An Overview of Computational Challenges in Online Advertising

Richard E. Chatwin

Abstract—Online advertising is a large and rapidly growing business. The major players in the space, namely advertisers, publishers, and ad exchanges, are developing increasingly sophisticated systems, methods, and tools to facilitate, manage, optimize, and report on the performance of online advertising marketplaces and campaigns. Developing solutions that are both mathematically sound and practical draws on techniques from a variety of disciplines including machine learning, stochastic optimal control, information retrieval, data mining, natural language processing, and econometrics. In this paper, we provide an overview of the online advertising space, and identify, frame, and describe solutions approaches to some of the major computational challenges in the space. We describe specific examples from industry applications, including ad inventory auctions, bidding and allocation strategies for ad inventory, inventory targeting, banner and landing page optimization, and performance estimation.

I. INTRODUCTION

Worldwide spending on online advertising exceeded $100 billion for the first time in 2012, and is projected to achieve annual growth on the order of 12% to reach $163 billion in 2016, thereby accounting for more than one quarter of all ad spending [1].

This spend increase reflects the preferences of both consumers and advertisers. Consumers increasingly engage with online media at the expense of print and TV media. Advertisers value the ability to target their advertising, based on consumers’ behavior and demographics, and to measure and quantify the success of online advertising. The targetability and measurability of online media engenders a broad range of fascinating problems in the areas of performance estimation, campaign optimization, and marketplace design that are amenable to formulation and solution via the techniques of control and optimization theory.

This paper provides an introduction and overview to the domain of online advertising. We highlight the broad array of challenges faced by the main participants, namely advertisers, online publishers, and ad exchanges. Many of these challenges have been met by formulating them as stochastic optimal control problems and, utilizing the latest research in the field, solving the resulting problems so as to generate actionable decisions. In other cases, we anticipate that the application of control theory may lead to improved solutions to the outstanding challenges.

In the following section we provide an overview of the online advertising space and describe at a high level several industry challenges that arise therein. The remaining sections focus on three specific examples that illustrate the diversity of problems that are encountered in online advertising.

Section III describes the problem of allocating a given ad inventory slot to: (i) a given advertiser (the publisher’s challenge); or (ii) a given campaign (the advertiser’s challenge). We illustrate how a form of Extended Kalman filter can successfully estimate advertiser/campaign performance and thereby support optimal decision making.

Section IV discusses how Search Engine Marketers can identify appropriate product attributes for inclusion in the ad copy displayed in response to user queries on search engines. We describe a solution that obtains product attribute sets through an unsupervised topic discovery model based on WordNet semantic similarity metrics.

Section V describes the problem faced by publishers of guaranteeing advertisers delivery of advertisement volumes in premium display ad inventory slots.

II. OVERVIEW OF THE ONLINE ADVERTISING SPACE

A. Online Advertising Ecosystem

The online advertising ecosystem is incredibly complex. A popular set of references for understanding this ecosystem are LUMAscapes: sector landscapes developed by LUMA Partners that attempt to organize the key sectors of digital media: Display, Search, Video, Mobile, Social, Commerce, Digital Capital, and Gaming [2]. For the purposes of this paper, we focus on the Display and Search channels that currently dwarf the other online advertising channels.

To simplify the discussion, we identify three critical players in the Display and Search channels: (i) publishers, i.e., web sites that display content to users navigating their sites or search engines that display search engine results pages in response to user search queries; (ii) advertisers who ultimately wish to sell their products to consumers and to this end want to show ads on publisher sites to either enhance their brand image or induce a direct response on the part of the consumer; and (iii) ad exchanges, who act as intermediaries between publishers and advertisers, facilitating the advertisers’ purchase of ad inventory from the publishers (often through automated auction-based pricing and buying in real-time). A related review paper of online advertising [3] makes a similar simplification, see, for example, Fig. 2 therein.

Publishers’ content pages typically include one or more spaces devoted to advertising; a challenge facing the publisher (or the ad exchange to which the publisher submits the inventory for sale) is how to match that advertising inventory

1R. E. Chatwin is VP Research at Adchemy, Inc., Foster City, CA 94404, USA richard at adchemy.com
Advertisers face the challenge of building an appropriate portfolio of advertising inventory to achieve their advertising objectives. We note the distinction between campaigns focused on brand advertising, where reach is the primary objective and no immediate response on the behalf of the consumer is expected, and direct or performance marketing, in which campaigns are designed to elicit an active response from the consumer, for example, in the form of a product purchase on the advertiser’s web site.

Before proceeding, we provide a little more color on some of the key intermediaries in the online advertising ecosystem. A good source of information regarding online advertising is the website of the Interactive Advertising Bureau (IAB) [4], an industry body that is dedicated to the growth of the interactive advertising marketplace, of interactive share of total marketing spend, and of its members share of total marketing spend. The following definitions are drawn from the IAB’s Wiki [5]:

- **Agency.** An organization that, on behalf of clients, plans marketing and advertising campaigns, drafts and produces advertisements, and places advertisements in the media (often through third-party ad servers).
- **An ad server** is a web server dedicated to the delivery of advertisements. This specialization enables the tracking and management of advertising related metrics.
- **Ad networks** provide an outsourced sales capability for publishers and a means to aggregate inventory and audiences from numerous sources in a single buying opportunity for media buyers.
- **A demand side platform (DSP)** provides centralized and aggregated media buying from ad exchanges, ad networks and sell side platforms, often leveraging real-time bidding capabilities of these sources.
- **A sell side platform (SSP)** provides outsourced media selling (to DSPs or ad networks) and ad network management services for publishers; it does not provide services for advertisers.
- **A data management platform (DMP)** facilitates the collection of audience intelligence by advertisers and ad agencies, thereby allowing better ad targeting in subsequent campaigns. **Online data aggregators,** e.g., Lotame and BlueKai, establish relationships with a large number of websites in order to gain a big-picture view of cookieed users that would be inaccessible to individual sites. **Offline data aggregators,** e.g., Acxiom and ChoicePoint, offer data acquired through both public record and private sources (like customer loyalty cards) for sale. This data is primarily sold to advertisers at the ZIP or ZIP+4 level, in order to maintain anonymity.

### B. Targeting Relevant Advertising

Performance marketers can maximize the economic return of their campaigns by identifying consumers with a high propensity to respond to the advertising and showing them ads to which they are most likely to respond.

An interesting example of such advertising in an offline setting is Target’s use of data regarding consumer product purchases to identify changes in purchase behavior that signal that the consumer is pregnant [6]. When consumers go through major life events, their shopping habits can become flexible and retailers have an opportunity to influence the consumer’s future shopping patterns for years. By identifying pregnant consumers, Target can include coupons for pregnancy-related products in offline mailings or emails to that consumer, and can potentially generate a lengthy future revenue stream.

Target’s advertising to pregnant consumers is predicated on two hypotheses that can be (and have been) validated through statistical experimentation and analysis: (i) Pregnant consumers have a higher propensity to change their future shopping habits; and (ii) Coupons targeted to specific stages of pregnancy can influence the consumer’s likelihood to change their shopping habits. This example illustrates several interesting points. Understanding the likely economic return of a particular ad shown to a particular user may be very difficult, so various assumptions must be made.

In many cases, a key assumption is that there is a strong correlation between ads that are most relevant to the user from an information theoretic sense and those that will generate the highest economic return. We see this assumption at work in search engine marketing, where advertisers are asked to bid on keywords and, in response to a user query, the search engine shows ads from those advertisers with the highest information-theoretic match between the advertiser keyword and the user query. The search engines refine their matching algorithms based on consumer responses to ads so that ultimately the match conflates the two concepts of information-theoretic relevance and economic return.

A second key assumption is that ads that share common characteristics will generate similar response rates, and users that share common characteristics will have a similar propensity to respond to a common ad. Thus, estimating the likely economic return or performance of an ad shown to a user will typically involve a feature-based or hierarchical model. Users and ads may be clustered based on their common features and historical observations of responses to ads.

Leveraging these assumptions and resulting matching-based algorithms and performance estimation models requires the ability to target advertising to users exhibiting specific attributes or in a specific context or at a particular time. The targeting can be based on user data available directly to the publisher or the advertiser or acquired from a data aggregator. There are a variety of targeting methods:

- **Geographic targeting.** Advertisers can show an ad specifically to visitors based on zip code, area code, city, DMA, state, and/or country. The user’s location may be identified from site registration information or inferred from the user’s IP address.
- **Demographic targeting.** Advertisers can show an ad specifically to visitors based on demographic information such as age, gender and income. The demographics may come from site registration information (in which
case the data is user specific) or may be inferred (in which case the data, partly to address privacy concerns, may be aggregate data, for example at the ZIP+4 level).

- **Intent-based targeting.** Advertisers can show an ad specifically to visitors who express a particular intent through actions they take. The primary example of this approach is in search engine marketing where advertisers target users (by bidding on keywords) who express an intent through the query they enter similar to that expressed by the keyword. Note that consumers can express intent through many types of actions and in many contexts, e.g., by searching for a particular restaurant type in a particular location on Yelp.com; by interacting with a mobile app; through liking something on Facebook; by tweeting a message on Twitter; through their credit card transactions. In many cases the infrastructure to support effective targeting of such consumers by advertisers is not yet in place, but we can anticipate that such infrastructure will be developed as mobile and social advertising markets continue to grow. Credit card issuers and processors are already targeting text messages to consumers who agree to receive real-time discounts based on their transactions. Plans to use a similar approach to target consumers online are afoot [7].

- **Site retargeting.** Advertisers can target ads to previous site visitors when they are on third-party web sites. For example, a consumer may visit an advertiser’s web site, put some items in the shopping cart on the site but then abandon the cart. By use of a pixel tag or other code a third-party (such as an ad server) can recognize that particular user when the user visits a different domain and can show the consumer a targeted ad (such as promoting the products that the consumer abandoned in the shopping cart on the advertiser’s site).

- **Creative retargeting.** Advertisers can show an ad specifically to visitors that previously were exposed to or interacted with the advertisers creative. The mechanism is similar to that used for site retargeting but the emphasis here is on showing the consumer a sequence of ads. This technique may be more effective for brand advertising rather than direct advertising.

C. Analytical Challenges

Trading and delivery of display advertising impressions is covered by one of two mechanisms: (i) private forward contract between publisher and advertiser (or third-party intermediary such as an ad network or ad exchange); and (ii) real-time spot auction managed by a publisher or ad exchange. Forward contracts, also known as advanced sales contracts, are long term and of large volume. In Section V we describe in more detail some of the challenges faced by a publisher offering such contracts, and in particular how to decide whether to accept a proposed new contract. Individual impressions may be sold on the spot market or allocated to a contract. Equally, a publisher may fulfill delivery on a contract by bidding in a spot market on behalf of the contract. Contract pricing may be flat-rated, i.e., the advertiser receives all impressions generated for an ad slot for a given period of time (for example, a movie studio promoting a new movie by purchasing the mantle on the Yahoo! or MSN Home Page for the entire day of the movie’s release); or on a cost-per-mille (CPM) basis, meaning the advertiser pays a fixed rate for every thousand impressions delivered.

Real-time spot auctions for display advertising are managed by ad exchanges, e.g., AdX, AdECN, RightMedia. Pricing in spot auctions may be on a CPM, CPC (cost-per-click), or CPA (cost-per-action) basis. Paid search advertising impressions are traded entirely through such auctions, which are typically managed by the search engine themselves, e.g., Google’s AdWords and Microsoft’s adCenter. Paid search pricing is exclusively on a CPC basis.

1) **Auction Mechanisms:** Ad exchanges and publishers operating real-time spot auctions require an effective auction mechanism. An auction is composed of two elements: an allocation algorithm, and a payment scheme (first and second price auctions have the same allocation algorithm, but differ in payment scheme). For ad exchanges and search engines, the most popular format is the Generalized Second Price (GSP) auction [8] (related references include [9], [10], [11], and [12]). The generalized nature of the auction refers to the fact that one auction is held to determine allocation of all paid search ad slots on a search engine results page (there may be 8-10 such slots). The second price nature of the auction refers to the fact that each advertiser pays only the amount required to secure its slot in the ranking of slots. Such auctions are not truthful; a truthful variant, the laddered auction of [13], has not been adopted by search engines.

Evaluating competing auction mechanisms for sponsored search is difficult due to model intractability, so [14] employs a niche-based co-evolutionary simulation approach to evaluate and compare key performance measures of several practical auction mechanisms, including the generalized first and second price auction, the Vickrey-Clarke-Groves mechanism, and a novel hybrid mechanism adopted by sogou.com, a major search engine in China.

The ranking of ads is based on the product of the advertiser’s bid and a quality score computed by the search engine. When the quality score encompasses only the estimated click-through rate for the advertiser’s ad, this ranking mechanism maximizes the immediate expected revenue to the search engine.

Obtaining accurate estimates of the (relative) click-through rates for the ads is a major challenge for search engines on a number of levels. Because search engines entertain a large number of queries, the historical data set is very large, requiring specialized techniques and algorithms involving parallelization for processing so-called Big Data, e.g., Hadoop and MapReduce. Further, a significant fraction of queries are unique (have never been searched on before), meaning effective techniques for relating such unique queries to non-unique queries are required. Because the auctions are conducted in real-time, the whole process of matching of query to potential ads, computing quality score for the ads
in relation to the query, and scoring and ranking the ads must happen very rapidly (so that the user does not notice a delay in the serving of the search engine results page).

Recent papers describing CTR estimation approaches include [15], which presents a Bayesian online learning algorithm used in Bing’s sponsored search advertising, and [16], which proposes a personalized click prediction model employing user-specific behavioral and demographic features.

In practice, quality score also incorporates relevance of the advertiser’s ad and landing page to the user query. In the long run, the search engine’s viability depends on it providing a good user experience. A user who clicks on an ad and is directed to a landing page that is irrelevant to their interests has a negative experience and is less likely to click on ads in the future. Equally, advertisers who pay for clicks that do not convert on their site may reduce future bids and budget on the search engine. It thus behooves the search engine to enforce a high degree of relevance of ads and landing pages to queries: quality score is the mechanism for achieving this. Developing algorithms that automatically measure the relevance of the billions of ads and landing pages and incorporate this into the quality score is a further challenge faced by search engines.

2) Matching: Matching impressions to ads is a complicated process. As a first step, advertisers publish a set of targeting criteria against which publishers or ad exchanges can match impressions.

For paid search campaigns this involves specifying a set of keywords with associated match types and ads; the search engines match the user query to the advertisers’ keywords and ads based on the match type (exact match requires the query to exactly match the keyword; phrase match requires the query to contain the keyword as a phrase; broad match gives the search engine latitude to match the query to the keyword based on some relevance judgment).

Developing a keyword portfolio and associated ad copy is a major challenge of search engine marketing. In practice, this challenge must be broken down to elemental steps. The first step is typically keyword suggestion, which involves expanding from a seed set of keywords; the major search engines offer tools for this based on query log mining. Many third-party tools submit the seed terms as queries to a search engine and identify semantically similar terms on the search engine results pages or on the top pages referenced in the results. Others utilize meta-tag spidering to identify relevant terms that web sites typically include in their meta-tags.

A variety of academic research in the area has explored alternative methods of defining semantic similarity between terms ([17], [18], [19], [20], [21], [22]). A recent technique proposed by [23] generates an advertiser-specific semantic graph that connects related keywords based on semantic similarity using a web-based kernel function. The graph can be traversed using algorithms that account for intensity of competition for a keyword. A multi-armed bandit framework is employed to determine the profitability of the keywords and arrive at an optimal set of keywords.

Other steps in sponsored search campaign development may involve organizing keywords into ad groups (using clustering techniques), developing ad copy and assigning ad copy to ad groups. Each of these may be broken down to further sub-steps. One such example is inserting relevant product attributes into ad copy. At Adchemy we have found that echoing in the ad copy the descriptive attributes expressed in the user query makes the ads more relevant to the user and leads to higher response rates. Identifying the right product attributes to echo is not straightforward. In Section IV we discuss an approach for this that uses a topic discovery approach to extract relevant attributes from WordNet.

For display campaigns, combinations of targeting criteria (ad location, ad context, user demographics, user geography, and user behavior) are called advertising lines. Advertisers must identify the right set of advertising lines on which to bid. One approach to doing this is to start a campaign with one or few advertising lines so as to collect a data set for analysis. Analysis involves clustering the targeting features into distinct advertising lines so that one or more performance metrics are relatively consistent within each advertising line, while limiting the total number of clusters (again, scale of the campaign in terms of number of advertising lines is an issue for both advertiser and ad exchange).

Advertisers may face the additional question of which campaigns (each campaign advertises a distinct product category) to target to which advertising lines. Section III describes an approach to solving this problem.

As described so far, advertisers publish a set of targeting criteria in an offline mode; when an impression arises in real-time, the ad exchange matches the impression to one or more advertisers’ ads, then either selects one or more ads or enters the ads into an auction to determine which to display to the user. When real-time bidding is enabled, advertisers are inserted into the online matching process. After the ad exchange matches the impression to an advertiser (based on the offline targeting criteria), the ad exchange queries the advertiser with the impression details. The advertiser can refine its bid and ad (or chose not to match to the impression). The advertiser’s new bid and ad are entered into the selection or auction process for determining which ads to display to the user. Recent literature focused on research topics related to ad exchanges includes [24], [25], [26], [27].

The publisher or ad exchange decision to allocate an impression to an advertiser can be complicated by longer-term considerations. One well studied example is the so-called AdWords problem [28], in which advertiser budgets are taken into account. To effectively utilize advertiser budgets when the allocation decisions are made in an online manner, the advertiser’s bid entered in the auction described above should be adjusted by a function of the proportion of the advertiser’s budget remaining. The standard formulation makes no assumption regarding the distribution of impression arrivals and assumes budgeted linear returns [28], [29]. Extensions involve (i) allowing concave return functions [30]; and (ii) making distributional assumptions regarding the arrival sequence of impressions [31], [32], [33], [34], [35], [36], [37], [38], [39], [40].

Advertisers must submit bids for all their targeted advertis-
ing lines or keywords. Bids are typically updated on a regular basis, say every day, to account for the latest performance data. There is a variety of literature discussing the bid management problem (see [41], [42], [43], [44], [45]), although much of it is not entirely practical. Most practical approaches require that the advertiser obtain an accurate estimate of the revenue per click for any keyword. Such estimates can be hard to develop because many keywords receive little data; for any given keyword portfolio, only a fraction will receive impressions, a fraction of those will generate clicks, and a fraction of those will generate revenue. Feature-based or hierarchical estimation models are commonly employed.

If budget constraints can be ignored (for example, the advertiser is willing to accept as much volume as the paid search campaign can generate, provided its ROAS target is met), then one approach is to bid on each keyword independently. The advertiser’s ROAS target is used to translate the expected revenue per click for the keyword into a cost per click that the advertiser is willing to pay for that keyword. Because the auction is second price, the advertiser will pay less than its bid; an estimate of the discount from bid to cost can be used to translate the cost per click into a bid value.

The method of [46] is slightly more sophisticated. It does allow for a budget constraint and assumes that for each keyword estimates of the following quantities are available: expected revenue per click, the volume of impressions per period, the click-through rate in the first position, the rate at which CTR decays with position, the number of competing bidders, and the distribution of competing bids. A Generalized Methods of Moment (GMM) approach is employed to estimate these parameters. In order to obtain identifiability of the model parameters, a period of essentially random bidding is required to generate sufficient variability in the bids and ad positions for each keyword across days. In a field test of the approach, a 26.4% increase in revenues relative to the approach used by an advertiser was observed.

3) Contextual advertising: A publisher or third-party ad server may select a campaign and ad to display based on the context of where the ad is to be displayed (banner and text ads on web pages, multimedia ads in video streams, mobile ads in mobile apps or mobile-version pages), and who is seeing them (geographic, demographic, and behavioral attributes of the visitor). Typically, this means maximizing the relevance of the ad to the context, although sometimes performance-based considerations are also included.

In many cases, contextual advertising focuses on the so-called Context Match problem of matching text ads (such as are displayed on sponsored search results pages) to web pages. Advertisers can use one campaign defined by keywords and ads to advertise both on the search engine itself and, through a Context Match program, on relevant web pages. Web pages include space for text ads and when a request to serve the page is received, the search engine is asked to provide relevant text ads for those slots. The search engine can match the web page to the keywords and ads trafficked by the advertisers. This matching may be accomplished by extracting keywords from the web page and matching those to the advertiser keywords.

A recent survey paper describes some of the challenges of and proposed solutions for contextual advertising [47]. The paper describes seven alternative approaches to contextual advertising; we append two additional approaches:

1) Keyword extraction. [48] uses natural language processing techniques and a logistic regression classifier to extract keywords from web pages. A particularly helpful feature is found to be the frequency of occurrence of the phrase in the MSN search query logs.

2) Impedance coupling. [49] expands the text of the webpage with new terms from similar pages to reduce vocabulary variance with relevant advertisements.

3) Genetic programming. [50] proposes a genetic programming framework to learn ranking functions that are very efficient in placing relevant ads on web pages.

4) Semantic matching. [51] combines semantic and syntactic matching of ads to web pages. Pages and ads are classified to nodes in a 6,000-node taxonomy of common commercial topics, then matched based on the distance between the nodes to which they are classified.

5) AdROSA. [52] utilizes a multi-agent architecture to create a personalized advertisement system depending on individual user browsing behavior rather than personal data input or demographic information.

6) Ant colony optimization. [53] uses historical data from user click-through patterns and Ant Colony Optimization (ACO) [54], which is a computational technique inspired by how an ant colony finds food.

7) Clickable terms. [55] matches web pages directly with a set of ad-side terms, independent of the page content. The relative clickability of ad-side terms on a given site is used to form a new set of features for training a maximum-entropy click model.

8) Sentiment detection. [56] propose the utilization of sentiment detection to select ads that are related to the positive (and neutral) aspects of blog posts.

9) Ontology matching. [57] propose an adaptive method to construct an ontology automatically for matching Chinese ads and web pages semantically. The method outperforms the impedance coupling method of [49] and the regression SVM method of [58].

Some of the approaches listed above incorporate some element of user behaviour (clicking on the ad, moving to another task immediately after clicking and so on) to help assess the relevance of the ad to the context. Other papers that focus more on user click-through data to select relevant ads include [59], [60], [61], and [62]. A recent paper focuses on optimizing long-run revenue from allocating ads to web pages [63]. The authors employ Partially Observable Markov Decision Processes (POMDPs) to manage the exploration-exploitation trade-off, leveraging correlations between ads to improve the efficiency of exploration. The algorithm significantly outperforms a conventional offline, content-based matching between webpages and ads when applied to a real-world ad dataset from a major search engine.
4) **Creative optimization**: As impressions are allocated to an advertiser’s campaign, the advertiser can choose which creatives to display so as to maximize a campaign objective such as the revenue or the number of actions or conversions generated. The creative selection decision may be made on an impression-by-impression basis, for example, if the creative are landing pages and the advertiser owns the page serving process. However, if the creative serving process is owned by a publisher or third-party ad server, then the advertiser will specify the fraction of impressions that should be devoted to each alternative over a given time period, e.g., a day.

Three approaches to creative optimization are commonly used, and apply whether the creative are banners in a display campaign, text ads in a paid search campaign, or the landing pages to which users clicking on banners or text ads are directed. All strategies involve two phases: (i) **exploration**, designed to learn accurate estimates of the expected payoff for each creative; and (ii) **exploitation**, where the focus is on the creative with the highest estimated expected payoff. In A/B testing and experimental design the two phases are distinct; in multi-arm bandits the two are intertwined.

In A/B testing a default (or control or champion) creative is tested against an experimental (or treatment or challenger) creative. The two creatives are shown in even rotation to a set of users. The test is run for a given number of impressions, chosen so as to be able to identify a specified performance difference with statistical significance. The best performing creative is used exclusively after the test. An A/B1/. . ./Z test includes multiple challengers to the current champion. Running sequential A/B tests allows multiple orthogonal design elements to be tested. [64], [65], and [66] describe online A/B testing; [67] presents a detailed survey.

When the creative design comprises multiple components, e.g., text content, layout, images, colors, etc, each with multiple available variations, the techniques of experimental design may be applied (originated by Fisher [68]; [69] describes its application to landing page optimization). Because the number of possible creatives (combinations of component variations) can be combinatorially large, methods such as fractional factorial design [70] and the Taguchi method [71], [72] aim to maximize the test output information for a given traffic volume. After the test the creative with the highest estimated expected payoff is used exclusively.

In multi-arm bandits [73] (review papers include [74] and [75]) an analogy is made between the creative optimization problem and a gambler playing a slot machine with multiple arms. The gambler learns the payoff distributions of the arms (creatives) as she plays them; she wants to play the arms in an order so as to maximize the long-run expected return. The problem can be modeled as a Markov decision process (MDP) and there are many strategies that provide an approximate solution [76], [77]. The most popular strategies are (i) UCB1 [78], [79], in which at each step an index for each arm is computed as an upper confidence bound on the expected payoff: the arm with the highest index is the one selected; and (ii) ε-greedy [80], in which the arm with the highest estimated payoff is selected with probability $1 - \varepsilon$, and a random arm is selected with probability $\varepsilon$. The value of $\varepsilon$ is time dependent and decays to zero as time increases. Recent research suggests that the Thompson sampling approach can be successful in practical settings [81], [82]. In this strategy each arm is pulled with probability equal to the current estimate of its being the best arm.

In the classical multi-armed bandit problem, the bandit arms are assumed to be independent. In creative optimization this is a strong limitation; two creatives having similar text but different colors have a dependency between them. If one of the creatives performs well, it can be inferred that the other will also. [83] proposes a generative model that groups the arms into different clusters. At each step, first a cluster is chosen in order to maximize expected reward probabilities and then an arm in that cluster is chosen. This algorithm is shown to achieve higher performance, both theoretically and empirically, as compared to the classical bandit algorithms.

### III. CAMPAIGN ALLOCATION

Both publishers and advertisers face the problem of allocating campaigns across inventory in order to achieve advertising objectives. Publishers will typically conduct the allocation at an advertiser level, i.e., must decide how to allocate advertising inventory among competing advertisers. In doing so publishers must balance competing objectives and constraints. In the short-term the publisher wishes to maximize revenue generated from the allocated campaigns while meeting advertiser requirements to deliver certain agreed upon volumes of impressions, clicks or actions. In the longer term, the publisher must ensure that each advertiser is able to achieve its internal ROI or ROAS objectives so that the advertiser is incented to continue purchasing advertising inventory from the publisher. Individual advertisers may run multiple advertising campaigns that focus on different business or product segments. For each impression allocated by the publisher to the advertiser, the advertiser may choose which of its campaigns to display for that impression, with the allocation decisions driven by volume, revenue, margin, budget and quality objectives. To illustrate how such a problem may be formulated, we focus on an advertiser allocating inventory between competing business segments.

Our example is drawn from the online lead generation business in which the lead generator runs display advertising campaigns to identify consumers who may be interested in its clients’ products. Each individual campaign is focused on one among a variety verticals, e.g., online education, mortgage refinace, auto insurance and credit cards. The lead generator will typically employ an ad serving solution to run its digital marketing campaigns across a number of publishers and placements within those publishers. The ad server targets the campaigns along various dimensions; we focus on three: placement, geography (state within the US), and day of week. A combination of placement, state and day of week is called a cell. The campaign allocation problem is to allocate the impressions within each cell across the campaigns. The allocation is updated and the new allocation trafficked to the ad server on a daily basis.
Determining how to allocate display advertising impressions within a cell involves estimating the likelihood that a typical user in a cell will respond to an ad (by ultimately completing and submitting a form on the lead generator’s landing page), estimating the likely revenue that will be generated by each such completed form, and using these estimates as inputs into an optimization problem.

### A. Formulation as an Optimization Problem

Denote the set of banner advertising campaigns by \( C = \{1, \ldots, C\} \). These campaigns can be run at one or more of a set of targeted cells \( G = \{1, \ldots, G\} \). Each campaign may have restrictions as to its eligible audience, so may be allocated only in a specified subset of the cells.

In the lead generation business, it is typical for the timeframe over which allocation is considered to be a calendar month; we divide the remainder of the current month into days denoted by \( d \in M \). Suppose that we know (from historical data, e.g., previous occasions on which we have made the buy at this placement) the number of banner impressions \( n_g^d \) shown to visitors in cell \( g \) during day \( d \), for each cell \( g \in G \) and day \( d \in M \).

Suppose also that we know (from historical data and simulations) the form per impression (FPI) conversion rate \( p_{gc}^d \), for impressions of campaign \( c \) shown to visitors in cell \( g \) on day \( d \), and the revenue per form (RPF) \( r_{gc}^d \) obtained from leads generated from impressions of campaign \( c \) shown to visitors in cell \( g \), for each cell \( g \in G \), campaign \( c \in C \), and day \( d \in M \).

The allocation decisions are what proportion \( x_{gc}^d \) of impressions in each cell \( g \in G \) to allocate to each campaign \( c \in C \) on each day \( d \in M \).

The formal statement of the problem (P) is:

\[
\begin{align*}
\text{maximize} & \quad \sum_{g \in G, c \in C, d \in M} n_g^d p_{gc}^d x_{gc}^d \\
\text{subject to} & \quad \sum_{g \in G, c \in C, d \in M} n_g^d p_{gc}^d x_{gc}^d \leq F_c \quad \forall c \in C \\
& \quad x_{gc}^d = 0 \quad \forall (g, c, d) \in \mathcal{E} \\
& \quad \sum_{c \in C} x_{gc}^d = 1 \quad \forall g \in G, d \in M \\
& \quad x_{gc}^d \geq 0 \quad \forall g \in G, c \in C, d \in M,
\end{align*}
\]

where \( \mathcal{E} \) defines the set of (cell, campaign, day) combinations that are prohibited. The objective is to maximize revenue \( R \) (we assume that the cost is fixed or at least uncorrelated) subject to four types of constraints. The first puts caps on the number of forms that can be generated for each campaign during the remainder of the month. The second simply prohibits certain campaigns being allocated in certain cells and on certain days. The third type of constraint simply says that the proportions of impressions allocated to the campaigns must sum to 1 within each cell and day. Finally, we require that the allocation proportions be non-negative.

Observe that (P) can be viewed as a generalized network flow problem, a special class of linear program, and thus can be solved efficiently. Consider a network with source nodes for each (cell, day) pair \((g, d)\) with supply \(n_g^d\). There are nodes for each campaign \(c\), and an arc from each source node \((g, d)\) to each campaign node \(c\) with loss \(p_{gc}^d\) and revenue multiplier \(r_{gc}^d\). There is one sink node with arcs from each campaign node to it. The upper bound on the flow in the arc from campaign node \(c\) to the sink is \(F_c\).

### B. Estimation of Problem Parameters

In practice, the numbers of impressions \(n_g^d\), the form per impression conversion rates \(p_{gc}^d\), and the revenue per form rates \(r_{gc}^d\) are unknown a priori and must be forecasted from historical data. Here we illustrate how a form of Extended Kalman Filter may be used for the estimation.

We consider a linear dynamical system driven by noise

\[
x(t + 1) = Ax(t) + B \varepsilon(t)
\]

where \(x, \varepsilon \in \mathbb{R}^n\), and \(A, B \in \mathbb{R}^{nxn}\). Here \(x\) represents the state of the system that evolves over time and \(\varepsilon\) is the state noise. We assume that the matrix \(B\) is invertible. We only consider the case where the \(\{\varepsilon(t)\}\) are white noise processes, i.e., a sequence of mutually uncorrelated error variables. We are mainly interested in the cases where \(\varepsilon(t) \sim \mathcal{N}(0, I)\) or \(\varepsilon(t)\) is a Laplace distribution with zero mean and covariance equal to the identity matrix.

At time \(t\) we observe the output \(y(t) \in \mathbb{R}^n\). The observations \(y(t)\) are related to \(x(t)\) by

\[
\mathbb{E}\{y(t)|x(t)\} = \mu(t) = h(x(t)).
\]

We will consider the case where \(h\) is the logistic function

\[
h(x) = \frac{e^x}{1 + e^x},
\]

and the entries of \(y(t)\) are equal to either \(-1\) or \(+1\).

We also assume that for \(t = 0\) we have \(x(0) \sim \mathcal{N}(\bar{x}(0), \Sigma_x(0))\). Assume for now that the matrices and vector \(A, B, \bar{x}(0), \Sigma_x(0)\) are given. We will discuss later the problem of estimating them. To simplify notation write

\[
x = \begin{bmatrix} x(0) \\ \vdots \\ x(T) \end{bmatrix}, \quad y = \begin{bmatrix} y(0) \\ \vdots \\ y(T) \end{bmatrix}, \quad \varepsilon = \begin{bmatrix} \varepsilon(0) \\ \vdots \\ \varepsilon(T - 1) \end{bmatrix}.
\]

Given the observations \(y\) we estimate \(x\) using the maximum likelihood estimate. If the probability of \(x\) given \(y\) is \(p(x|y)\), we have

\[
\hat{x} = \arg\max_x p(x|y) = \arg\max_x \log p(x|y).
\]

To further simplify notation, define

\[
\hat{P} = \begin{bmatrix} -B^{-1}A & B^{-1} & 0 & 0 & \cdots & 0 \\ 0 & -B^{-1}A & B^{-1} & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & -B^{-1}A & B^{-1} \end{bmatrix},
\]

so that we have

\[
\varepsilon = \hat{P}x.
\]
By taking the logarithm of the probability and removing the terms that do not depend on \( x \) we obtain the equivalent formulation in the case of Gaussian noise:

\[
\tilde{x} = \arg\min_x \left( \frac{1}{2}\|\Sigma^{-1/2}_x(x(0) - \tilde{x}(0))\|^2 + \frac{1}{2}x^T P \tilde{P} x + \sum_{j=1}^{(T+1)n} \log (1 + e^{x^T y_j}) \right).
\] (1)

This is nothing more that a logistic problem with a special quadratic penalty. We can also rewrite the problem for the case of Laplacian noise:

\[
\tilde{x} = \arg\min_x \left( \sqrt{2}\|\Sigma^{-1/2}_x(x(0) - \tilde{x}(0))\| + \sqrt{2}\|\tilde{P} x\| + \sum_{j=1}^{(T+1)n} \log (1 + e^{x^T y_j}) \right).
\] (2)

In this case this is a standard logistic problem with an \( L_1 \) penalty.

To simplify the notation we can introduce the matrix

\[
\tilde{Q} = \begin{bmatrix} \Sigma^{-1/2}_x & 0 & 0 & 0 & 0 & 0 \\ -B^{-1} A & B^{-1} & 0 & 0 & \cdots & 0 \\ 0 & -B^{-1} A & B^{-1} & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & -B^{-1} A & B^{-1} \end{bmatrix}
\]

and the vector

\[
\tilde{r} = \begin{bmatrix} \Sigma^{-1/2}_x(0) \\ 0 \\ \vdots \\ \vdots \\ 0 \end{bmatrix},
\]

and write the two problems as

\[
\tilde{x} = \arg\min_x \left( \frac{1}{2}\|\tilde{Q}x - \tilde{r}\|^2 + \sum_{j=1}^{(T+1)n} \log (1 + e^{x^T y_j}) \right)
\] (3)

and

\[
\bar{x} = \arg\min_x \left( \sqrt{2}\|\tilde{Q}x - \bar{r}\| + \sum_{j=1}^{(T+1)n} \log (1 + e^{x^T y_j}) \right)
\] (4)

C. Convex formulation

Clearly both problem (3) and problem (4) are convex problems [84] and can be solved efficiently using a primal-dual algorithm. A standard solver for regularized logistic problems can be used to solve these problems if it supports generic quadratic or \( L_1 \) penalty functions. We can generalize both problems into a common convex optimization problem:

\[
\text{minimize } \sum_{j=1}^n n_j \log (1 + e^{-(M_j + x)}) + \|M_3 x - m_2\|_1 + \|M_3 x - m_3\|_2^2.
\] (5)

Clearly using a solver that accepts a generic quadratic penalty of the form \( \|M_3 x - m_3\|_2 \) would be extremely inefficient for solving problem (5). The reason is that the solver would not make use of the structure of the matrix \( M_3 \). Utilizing the matrix structure can considerably improve the performance of the solver.

Previous work has employed non-linear Kalman filters to solve problems similar to (3) and (4), [85] uses a Kalman filter applied to a linearized system. The non-Gaussian distributions are replaced with a Gaussian approximation by linearizing them around the estimated \( x(t) \) of the previous Kalman iterations. This approach does not guarantee a correct solution and it can be improved by multiple iterations. [85] also proposes a Fisher-scoring technique in which, at each iteration, the linear estimation problem is solved using a Kalman filter. Similar approaches using a forward-backward Fisher-scoring technique are presented in [86].

D. Model estimation

In the exposition thus far we have assumed that the matrices describing the model and the a priori distribution of \( x(0) \) are known. In this section we look at the problem of estimating \( A, B, \tilde{x}(0) \), and \( \Sigma x(0) \). Working with the generic problem formulation (5), the challenge is to estimate the matrices \( M_1, M_2, \), and \( M_3 \).

One approach to solving this estimation problem is to use an EM algorithm. In the first step we assume that \( M_1, M_2, \), and \( M_3 \) are given and we solve problem (5) for variable \( x \). Once the new estimate of \( x \) is computed we can solve problem (5) with variables \( M_1, M_2, \), and \( M_3 \). We can then iterate to convergence. Both problems are convex problems. To account for the possible structure of the \( M_i \) matrices, we can add linear constraints on the \( M_i \).

E. Implementation details

Rather than trying to solve the non-differentiable problem (5), we focus on the equivalent problem:

\[
\begin{align*}
\text{minimize} & \; \sum_{j=1}^n n_j \log (1 + e^{-(M_j + x)}) + 1^T t \\
\text{subject to} & \; -t \leq M_3 x - m_3 \leq t.
\end{align*}
\] (6)

The gradient of the objective function with respect to \((x,t)\) is

\[
\begin{bmatrix}
M_1^T \\ M_3^T \\
\text{diag}(n \star \frac{x - M_j}{(1 + e^{-(M_j + x)})})
\end{bmatrix}
\]

where \(1/(1 + e^{-(M_j + x)})\) is a vector whose \( i \)-th component is \(1/(1 + e^{-(M_j + x)}))\), and \( \star \) represents elementwise vector multiplication. The Hessian of the objective function with respect to \((x,t)\) is

\[
\begin{bmatrix}
M_1^T \\ M_3^T \\
0
\end{bmatrix}
\]

At each iteration of the algorithm we need to solve the primal-dual system [87]

\[
\begin{bmatrix}
H & 0 & M_2^T & -M_3^T \\
0 & 0 & -I & -I \\
M_2 & -I & -D_1^{-1} & 0 \\
-M_2 & -I & 0 & -D_2^{-1}
\end{bmatrix}
\begin{bmatrix}
\delta x \\
\delta t \\
\delta z_1 \\
\delta z_2
\end{bmatrix}
= \begin{bmatrix}
c_1 \\
c_2 \\
c_3 \\
c_4
\end{bmatrix},
\]

where \( H \) is the Hessian, \( z_1 \) and \( z_2 \) are the dual variables associated with the two inequalities, and the \( \{c_i\} \) are the
residuals from the previous iteration. For \( i = 1, 2, D_i = \text{diag}(z_i/s_i) \), where \( s_i \) is the slack of the inequality. We seek an efficient way to solve these equations. We have

\[
\begin{bmatrix}
H + 4M_2^T D M_2 & 0 & 0 & 0 \\
-M_2^T & I & D_1^{-1} & 0 \\
-2M_2 D & 0 & I & 0 \\
0 & 0 & I & I
\end{bmatrix}
\begin{bmatrix}
\delta x \\
\delta t \\
\delta z_1 \\
\delta z_2
\end{bmatrix}
= 
\begin{bmatrix}
c_1 - M_2^T c_2 + 2M_2^T D (c_3 - c_4 + D_2^{-1}c_2) \\
-c_4 \\
D(c_4 - c_3 - D_2^{-1}c_2) \\
-c_2
\end{bmatrix},
\]

where

\[ D = (D_1^{-1} + D_2^{-1})^{-1}. \]

We can now solve for \( x \) and substitute the result to obtain \( z_1, t, \) and \( z_2 \) in this order. The matrix

\[
H + 4M_2^T D M_2 = 
M_2^T \text{diag} \left(n * \frac{e^{-M_1 x}}{(1 + e^{-M_1 x})^2}\right) M_1 + 2M_2^T M_2 + 4M_2^T D M_2
\]

has a banded arrow structure that can be easily solved with a block elimination or sparse Cholesky technique [88].

F. Results

To illustrate the performance of the Extended Kalman Filter estimation model in practice, we use an example from an online lead generator (Adchemy, Inc.) in the mortgage refinance vertical. A granular model was used to estimate both the forms per impression (FPI) and the revenue per form (RPF) in the month of January 2008. We report results averaged across all cells and campaigns in the vertical.

January 2008 was a very interesting month for the mortgage business, marking the start of the “Great Recession”, which saw declines in all major stock market indices and dramatic loss of value in housing markets around the world. With U.S. markets closed for MLK Day on January 21, 2008, markets across the rest of the world experienced severe one-day sell-offs [89]. The Federal Open Market Committee of the Federal Reserve Bank responded by cutting the Federal Funds interest rate by 75 basis points to 3.5% on the 22nd [90], and by a further 50bp to 3% on the 30th [91].

These two rates cuts spurred consumer interest in mortgage refinances, leading to much higher response rates to the display advertising of lead generators. Many such consumers were not qualified for a new loan (or had not realized that changes in the Fed Funds rate are not immediately reflected in prevailing mortgage rates). Consequently, mortgage lenders responded by cutting their caps, i.e., the numbers of leads they were willing to buy from lead generators.

These events caused rapid changes in both Adchemy’s FPI and RPF, as reflected in the charts of Fig. 1. The data in each chart have been normalized so that the observation on January 1 takes on the value 1.0. The two vertical lines in each chart are the days of the Federal Funds rate cuts. We make the following observations about these charts:

- The first half of the month was business-as-usual, with the RPF chart reflecting the usual pattern of relative strength during the week and weakness on the weekend.
- Adchemy had a large one-day distribution buy with a major publisher on the 17th, resulting in a spike in FPI and a dip in RPF.
- Adchemy experienced very high form volume the two days after the surprise Federal Funds rate cut (the 22nd), while two major lenders cut their caps. The day of the first rate cut Adchemy experienced a big drop in RPF (but note that the model readjusted after two days).
- The models successfully tracked the changes in FPI and RPF, particularly the steep increase in FPI for the rest of the month after the first Federal Funds rate cut.

While changes in performance metrics as dramatic as those observed in January 2008 are uncommon, relative swings in performance of cells and campaigns is very common and this Kalman-filter based estimation model can help to make close to optimal allocation decisions.

IV. UNSUPERVISED Topic Discovery for ATTRIBUTE EXTRACTION

In this section we discuss a problem that arises in Search Engine Marketing (SEM), namely identifying product attributes for inclusion in the ad copy displayed in response to user queries. This step is integral to developing ads that are relevant to the user queries [92].

Fig. 2 shows candidate ads for the search term “Leather Chaise Lounge”. Each ad comprises a headline, two lines of description, and a display url. Each element can be considered a short sentence comprising two parts: a) Intent, and b) Noun Phrase. The intent of the ad tells the primary purpose of the ad, while the noun phrase describes the product on which the ad focuses. For example, in the description line “IKEA Leather Chaise Lounge on sale”, “on sale” is the intent and “IKEA Leather Chaise Lounge” is the noun phrase, comprising the product “chaise lounge” and its attributes “IKEA” and “leather”. Formally, this can be expressed in context free grammar as:

\[
\langle ad \rangle = \langle (\langle intent \rangle) \rangle \langle (\langle noun phrase \rangle) \rangle \langle (\langle intent \rangle) \rangle
\]

\[
\langle noun phrase \rangle = \langle (\langle attribute \rangle) \rangle \langle (\langle product name \rangle) \rangle
\]

The \( \langle intent \rangle \) is common across most if not all product categories and hence is easy to obtain, but finding the \( \langle attribute \rangle \) set requires domain knowledge. The \( \langle attribute \rangle \) set of a product helps in defining the specificity of the ad by: a) targeting those consumers who are interested in specific attributes, and b) providing information about the category of products that are offered for the given \( \langle intent \rangle \) by the advertiser. For example, in the “Toy” product category, an ad with headline “Wooden brain-teaser puzzles for sale at Walmart”, with the use of “wooden” and “brain-teaser” attributes, targets the set of consumers who are interested in wooden brain-teaser puzzles. Current practice requires manually extracting attributes from the advertiser’s product catalog.
Fig. 1: Daily Estimated vs. Observed Normalized Metrics for Adchemy’s Mortgage Refinance lead generation business, January 2008.

 Clearly, automated attribute extraction methods would be beneficial. Previous work in this area has explored a variety of approaches: (i) Generative and structure learning-based extraction [93], [94], [95]; (ii) Ontology-based extraction [96], [97]; (iii) Tag-based supervised extraction [98], [99], [100]; and (iv) Semantics and rule-based extraction [101], [102], [103], [104], [105]. However, all such work has focused on: a) finding entities when the entity types are known, e.g., finding a person or location from a text, or b) extracting entities/relations using a seed set to train a model.

This can be problematic for entities heretofore unseen by the model. Here we describe a simple and scalable information extraction model, first presented in [106], to discover new entity types without any seed set or prior knowledge of the types to be extracted. This scenario is encountered in the SEM ad-copy creation process, where the attributes of a product being advertised typically vary from one product category to another. Seed labels from previously understood product categories can be of little use for a new product category. The earlier approach closest in spirit to our approach (as it too does not need any seed set or assume any rules) is unsupervised topic modeling [107]. However, that approach is very generic and does not exploit the unique properties of SEM tasks and datasets.

The unsupervised model described here uses WordNet semantic similarity metrics to obtain product attribute sets. It is completely unsupervised in terms of seed sets or rules/patterns. The input of the model is an online product category catalog and the output is a scored set of labeled clusters, each representing an attribute of the product. For the “Toy” product category mentioned above, the model discovers the attribute-clusters: \{wooden, leather, plastic, tin, ...\} and \{red, green, blue, black, ...\} and provides labels “material” and “color”, respectively, to these two clusters. The ad-generation scheme of (7) is revised by employing a new scheme for \(\langle\text{noun phrase}\rangle\) generation:

\[
\langle\text{noun phrase}\rangle = \langle\text{attribute}_1\rangle \cdots \langle\text{attribute}_n\rangle \langle\text{product name}\rangle \tag{8}
\]

The product attributes are assigned a certain order to give the ad semantically correct structure. For example, “coffee-colored women’s t-shirt” is semantically/aesthetically better than “women’s coffee-colored t-shirt”. Knowing attribute-cluster labels facilitates obtaining an appropriate attribute order [108], [109]. The attribute-cluster scores help in determining the relevance of an attribute for the product category and hence, whether to include the attribute in the ad copy.

A. The Attribute Extraction Model

The model treats the product catalog as a bag of words. Each word \(w\) present in catalog \(d\) is assigned a probability mass \(P(w|d)\) as follows:

\[
P(w|d) = \frac{N_d(w)}{\sum_{w \in d} N_d(w)} \tag{9}
\]
where \( N_d(w) \) is the count of word \( w \) in catalog \( d \). The sense-set \( \psi_w \) for each word \( w \) is the set of all senses of \( w \), i.e.,

\[
\psi_w = \{ w_s : w_s \in \text{synset}(w) \} 
\]

where \( \text{synset}(w) \) contains the SynSets of word \( w \) in WordNet [110]. Each member \( w_S \) of set \( \psi_w \) is a unique sense of word \( w \) and lies in a unique SynSet of \( w \). The catalog \( d \) is expanded to a “bag of senses”, \( \Psi \), where:

\[
\Psi = \bigcup_{w \in d} \psi_w 
\]

A naive approach to clustering these senses is to group two senses, \( w_{s_1} \) and \( w_{s_2} \), together if \( \text{hypernym}(w_{s_1}) = \text{hypernym}(w_{s_2}) \), i.e., \( w_{s_1} \) and \( w_{s_2} \) are immediate siblings in the WordNet hypernym tree. Fig. 3 shows an example where two immediate sense siblings, “cherry” and “pine” are clustered using a common parent “wood”, which becomes the cluster head of the new cluster. This approach can be extended to include “leather” in the cluster with “material” as the new cluster head. Now “material” represents a valid cluster label but the naive approach of clustering words based on the WordNet sense hierarchy does not know when to stop adding more hierarchies to the sense tree. This can be achieved by utilizing two important semantic metrics:

- **depth-metric**: the sense of a word increases in specificity as the word’s depth increases in WordNet [111]
- **hop-metric**: the smaller the hop-counts between two words in the WordNet taxonomy, the closer their senses are [112].

Model parameter tuning is based on the above two metrics. The model learning has two phases: 1) Cluster Discovery, and 2) Cluster Pruning.

B. Cluster Discovery (Phase I)

Algorithm 1 describes the cluster discovery process. The model iterates through each word-sense present in the “bag of senses”, \( \Psi \), obtained from (11), and clusters them together based on WordNet’s hyponym-hypernym (IS-A) relation. For each sense \( s \in \Psi \), the algorithm first iterates through all the discovered clusters in cluster set \( \Omega \) and checks whether there exists \( C_i \in \Omega \) such that \( s \) has a valid hypernym/hyponym relation with \( C_i \). If so, then \( s \) is added to \( C_i \) and \( C_{\text{head}} \) is modified appropriately. If there is no such \( C_i \), then the model iterates through the previously unclustered senses \( s_j \in \Psi \) such that \( s \) and \( s_j \) have a hypernym/hyponym relationship between them. If there exists such an \( s_j \) then a new cluster \( C_{\text{new}} \) is created with \( s \) and \( s_j \) inserted into \( C_{\text{new}} \) and \( C_{\text{new,head}} \) appropriately initialized. \( C_{\text{new}} \) is inserted into \( \Omega \). The model moves onto the next sense in \( \Psi \) and repeats the above steps. After iterating through all elements of \( \Psi \), \( \Omega \) returns with a set of candidate clusters/topics with each cluster’s head assigned as label for that topic. The labels of these clusters will become our discovered attributes of the products and the members will fill the tokens \( \langle \text{attribute} \rangle_i \) of (8). Each \( C_i \in \Omega \) is a cluster of word-senses with a sense-hierarchy among the elements present in it.

C. Cluster Pruning (Phase II)

The left half of Table I shows the top five prominent clusters discovered after applying Phase I to a product catalog from the Furniture category. Only one of these five clusters is a valid one and even this cluster is noisy. The clusters obtained in Phase I generally contain a lot of noise and are not sense specific. The Cluster Pruning phase addresses these issues by parameterizing the cluster properties based on the WordNet metrics defined in Section IV-A. The clusters have the following properties:

- **Cluster Depth** (\( C_{\text{depth}} \)): The cluster depth is the depth of the head-node of the cluster in the WordNet sense hierarchy, i.e., \( C_{\text{depth}} = C_{\text{head,depth}} \).
- **Cluster Breadth** (\( C_{\text{breadth}} \)): The cluster breadth is the vertical span of the the cluster tree in terms of WordNet depth \( C_{\text{breadth}} = \max(\text{node,depth} - C_{\text{depth}}) \) where \( \text{node} \in C \).
- **Cluster Probability Mass** (\( C_{\text{mass}} \)): The probability of a sense \( s \) in catalog \( d \) is defined as \( P(s|d) = P(w|d) \) where \( s \in \psi_w \), i.e., \( s \) is a sense of word \( w \). The cluster probability mass, \( C_{\text{mass}} \), based on this is \( C_{\text{mass}} = \sum_{s \in C} P(s|d) \).
- **Cluster Density** (\( C_{\text{density}} \)): The cluster density is defined as:

\[
C_{\text{density}} = \frac{C_{\text{mass}}}{C_{\text{breadth}}} \tag{12}
\]

- **Mutual Information** (\( Ml(C_1, C_2) \)): The mutual information between any two clusters \( C_1 \) and \( C_2 \) for a given catalog \( d \) measures the amount of common mass between the two clusters. A common word set, \( \Gamma_{C_1, C_2} \), between \( C_1 \) and \( C_2 \) is defined as:

\[
\Gamma_{C_1, C_2} = \{ w : w \in d \land (\exists s_1, s_2 \in \psi_w \ s.t. \\
(s_1 \in C_1 \land s_2 \in C_2) \} \tag{13}
\]

where \( w \) is a word in catalog \( d \). The mutual information, \( Ml(C_1, C_2) \) is defined as:

\[
Ml(C_1, C_2) = \frac{\sum_{w \in \Gamma_{C_1, C_2}} P(w|d)}{C_{1\text{mass}}} \tag{14}
\]
Table I: The result of the model for $\kappa = 5$, before (left of the vertical line) and after (right of the line) pruning for Furniture category. The top 10 elements in each cluster are shown.

<table>
<thead>
<tr>
<th>Unit</th>
<th>Set</th>
<th>Play</th>
<th>Event</th>
<th>Material</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clark</td>
<td>Quartet</td>
<td>Hole</td>
<td>Turn</td>
<td>Soda</td>
</tr>
<tr>
<td>London</td>
<td>Suite</td>
<td>Set</td>
<td>Label</td>
<td>Lime</td>
</tr>
<tr>
<td>Almond</td>
<td>Product</td>
<td>Sets</td>
<td>Carry</td>
<td>Products</td>
</tr>
<tr>
<td>Hamilton</td>
<td>Core</td>
<td>Case</td>
<td>Crib</td>
<td></td>
</tr>
<tr>
<td>Clock</td>
<td>Triplet</td>
<td>Pawn</td>
<td>Stool</td>
<td>Articulating</td>
</tr>
<tr>
<td>Bradley</td>
<td>Solitary</td>
<td>Sitting</td>
<td>Mocha-colored</td>
<td></td>
</tr>
<tr>
<td>Lotus</td>
<td>Prince</td>
<td>Adornment</td>
<td>Mahogany-colored</td>
<td>Tufted-seat</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Furniture</th>
<th>Equipment</th>
<th>Wood</th>
<th>Color</th>
<th>Leather</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dresser</td>
<td>Bird</td>
<td>Ash</td>
<td>Red</td>
<td>Kids</td>
</tr>
<tr>
<td>Bureau</td>
<td>Glove</td>
<td>Hardwood</td>
<td>White</td>
<td>Buff</td>
</tr>
<tr>
<td>Table</td>
<td>Hammer</td>
<td>Wicker</td>
<td>Yellow</td>
<td>Suede</td>
</tr>
<tr>
<td>Etagere</td>
<td>Set</td>
<td>Alder</td>
<td>Jade</td>
<td>Micro-suede</td>
</tr>
<tr>
<td>Sleeper</td>
<td>Bag</td>
<td>Birch</td>
<td>Tawny</td>
<td>Crushed</td>
</tr>
<tr>
<td>Seats</td>
<td>Wood-base</td>
<td>Knot</td>
<td>Two-tone</td>
<td>Alligator</td>
</tr>
<tr>
<td>Counter</td>
<td>X-base</td>
<td>Log</td>
<td>Grey</td>
<td>Morocco</td>
</tr>
<tr>
<td>Buffet</td>
<td>Club</td>
<td>Driftwood</td>
<td>Pastel</td>
<td>Cordovan</td>
</tr>
</tbody>
</table>

To obtain a more sense-specific set of clusters with valid attribute labels, the model constrains the above cluster properties through the following three model parameters, based on the semantic metrics mentioned in Section IV-A.

- $\delta$: Regulates the depth of discovered cluster $C$ to obtain suitable product attributes as clusters.
- $\beta$: In the WordNet sense-hierarchy, the sense-specificity spreads as one goes down. The model parameter $\beta$ controls this spread in the clusters.
- $\mu$: The mutual information, $MI(C_1, C_2)$, between two clusters gives an estimate of how much one cluster replicates the other. If this replication exceeds a threshold (regulated by $\mu$), the smaller cluster should be discarded as it does not contain any independent information.

Algorithm 2 shows the steps involved in the Cluster Pruning phase. Each cluster obtained from Phase I is tested on the three parameters $\delta$, $\beta$, and $\mu$. A cluster that does not satisfy the constraints imposed by any of the three is broken into smaller clusters, where the smaller clusters are the subtrees one hop down in the sense hierarchy in the cluster. The children of the previous cluster head are the cluster heads of the respective new clusters. When all clusters in $\Omega$ satisfy the constraints imposed by the model parameters, the model creates a ranked set of clusters $\Omega$, ranking the clusters in descending order of their cluster density defined in (12). This density is modified in cases when “hop-holes” are discovered, i.e., when the nearest child to a cluster-head is more than one hop away. The new density $C_{\text{density}}' = \frac{C_{\text{density}}}{(C_{\text{depth}} - \delta + \text{hop size})}$, where $C_{\text{depth}} = \sum_{C_i \in \Omega} C_{\text{depth}}(C_i)$ and hop size is the “hop-hole” size. This modified density accounts for the sudden jump in sense-specificity, due to “hop-holes”, when moving down from cluster-head to its children in the sense tree.

The parameters, $\delta$, $\beta$, and $\mu$, used in Algorithm 2, are learned over a training set, being tuned, for example, so as to optimize the number of valid clusters (as assessed by domain experts) found among the top 40 clusters returned by the model on one or more training sets. However, in practice we found that the optimized parameter values ($\delta = 6$, $\beta = 8$, and $\mu = 0.7$) vary minimally among alternative training sets.

D. Summary Results

Clusters to the right in Table I are the top five clusters discovered by the model after Phase II. Except for “Equipment”, the remaining four are valid attribute clusters of the Furniture category. Moreover, a quick comparison to the valid clusters on the left reveals that the new clusters are much less noisy.

In a comparison with a traditional LDA approach our model is able to find substantially more attribute clusters with a much higher cluster purity (proportion of valid terms in the cluster, as assessed by domain experts) than the LDA model. Table II reports the valid clusters found by our model and by the LDA model for $\kappa = 10$ for a product catalog from the Watches category. As such, the approach represents an effective mechanism for unsupervised semantics-based attribute extraction. The attributes uncovered would be very hard to discover by a tagging or a rule-based technique unless we knew what we were looking for.

The proposed model can be used as a bootstrapping method for tag-based extraction techniques. In that case the valid attribute clusters returned by the model can be used as a seed set for the corresponding attribute extraction. One drawback of the model is that it cannot discover clusters for attributes that are not included in WordNet, such as brands.

V. Guaranteed Delivery in Display Advertisement

In this section we study the pricing and allocation of display advertisements from the publisher’s point of view. We briefly introduce three research problems, then focus on one in more detail. The materials in this section are largely based on [113], but also rely on [114], [115], and [116].

A. Setting

Unlike sponsored search where all advertisements are solely priced and sold in spot auctions, display advertisements are also sold through bilaterally negotiated $\text{advanced sales contracts}$. This is particularly the case for premium ad-inventory, with contracts including $\text{guaranteed delivery}$ provisions that require the publisher to deliver a certain minimum number of impressions.

Advanced sales contracts allow for large volumes of impressions to be sold ahead of time. A natural use case is that of a branding campaign, where determining the value of each individual impression might be impossible, but the value of the complete campaign is easy to determine. Alternatively,
one can see advanced sales contracts as enabling complementarities between display advertisement impressions\(^1\).

In an advanced sales contract, together with the value of the contract, an advertiser usually selects a target audience, a minimum impression target per day, and a maximum duration. If the contract is accepted, the publisher needs to deliver at least the minimum of impressions per day, of the right type, for the duration of the contract.

The publisher now faces a problem similar to that faced by financial portfolio managers. Think of each potential contract as an available stock to hold, with the spot market as a risk-free alternative. For each impression, the publisher decides whether to invest in the risk-free option (sell the impression in the spot market) or invest in one of available risky options (allocate the impression to one of the guaranteed delivery contracts). For any given guaranteed delivery contract, the risk comes from payment being contingent on delivery of the right impressions.

The publisher has three types of decision to consider:

1. whether to allocate a given impression to the spot market or to a contract;
2. for an impression not allocated to the spot market, what contract to allocate it to; and
3. whether or not to accept a proposed new contract.

The first question is considered by [115], where the authors assume all impressions are available in the spot market, and contracts are implemented through bidding (in the spot market). A key assumption is that the price of an impression in the spot market is positively correlated to the value of the impression for all advertisers. We have thus an inherent substitutability of impressions both within and across advertisers’ contracts. The main contribution of the paper is a characterization of “maximally-representative” allocation (those for which the distribution of impressions used to satisfy a contract is consistent with the distribution of the impressions’ supply), together with a randomized bidding strategy to implement said allocation in the spot market.

The second question is considered by [114]. Based on the assumptions that impressions arrive online, the set of contracts is known, and sampling from the impressions’ distribution is possible, the authors design a near-optimal and compact allocation plan. The compactness constraint is crucial in the real-world setting where the allocation has to be implemented in a distributed system with low latency and limited computing power. The main contribution of the paper goes beyond the online allocation problem as the authors’ solution is applicable to a large class of “online assignment problems”, the “online allocation with forecast” being one specific instance of the general problem.

The third question is considered both by Feige et al. [116] and by Alaei et al. [113], Feige et al. explore the problem of approximating the optimal revenue the publisher can expect given that both the supply of impressions and the available advanced sales contracts are known. A key assumption is that contracts are binding: there is a large penalty to the publisher for breaching a contract it agreed to deliver on. The main contribution is the proposal of a greedy admission control algorithm whose performance guarantee improves as the performance of the optimum solution improves.

In the following subsections, we present in more detail the online setting defined in [113]. The key distinction from other work is the explicit introduction of temporal constraints in the definition of the advanced sales contracts.

### B. Preliminaries and Definitions

We assume advertisers are interested in acquiring display advertisements over different business periods, called rounds. We also assume said advertisers are interested in maximizing their utility after a given number of rounds. We call that number of rounds the time horizon, or business period for the given advertiser. As an example, a business period can be a month, and a round can represent a day. An advertiser interested in branding activities could be unwilling to let a single day go by without one of its ads being served to a target population.

Let \( T > 0 \) be the total number of rounds available, and label each round by \( t \in \{1, \ldots, T\} \). We make the assumption that, during a given round \( t \), all impressions are perfect substitutes, and that \( n_t \) are for sale. We define the following notation,

\[
S = \prod_{t=1}^{T} \{0, \ldots, n_t\}
\]

where \( s \in S \) represents a possible allocation of impressions to a given advertiser. Finally, we let \( N = \sum_{t=1}^{T} n_t \) be the total supply over the business period. For ease of exposition,

<table>
<thead>
<tr>
<th>Timepiece Metal</th>
<th>Color</th>
<th>Quartz</th>
<th>Leather</th>
<th>Jewelry</th>
<th>Band</th>
<th>Material</th>
<th>Brand</th>
</tr>
</thead>
<tbody>
<tr>
<td>wristwatch</td>
<td>stainless</td>
<td>brown</td>
<td>topaz</td>
<td>grain</td>
<td>band</td>
<td>stainless</td>
<td>hush</td>
</tr>
<tr>
<td>hunter</td>
<td>gunmetal</td>
<td>salmon</td>
<td>clip</td>
<td>chain</td>
<td>sapphire</td>
<td>quartz</td>
<td>crystal-accepted</td>
</tr>
<tr>
<td>chronograph</td>
<td>blue</td>
<td>agate</td>
<td>weed</td>
<td>carabiner</td>
<td>strap</td>
<td>leather</td>
<td>expansion</td>
</tr>
</tbody>
</table>

TABLE II: Valid clusters discovered for the Watches catalog with \( \kappa = 10 \). The first 7 clusters are discovered by our model; the last 2 are discovered by LDA. The first row in the table is the cluster label.
**ALGORITHM 1: Cluster Discovery**

Initialize:

- $\Psi$ /*obtained via (11)*/
- $\Xi \leftarrow \{\}$ /*stores clustered senses*/
- $\Omega \leftarrow \{\}$ /*set of discovered sense clusters*/
- flag $\leftarrow$ false

for each $s \in \Psi$, do
  for each $C_i \in \Omega$, do
    if hypernym($s$) = $C_{i\text{head}}$ then
      insert $s$ into cluster $C_i$
      flag $\leftarrow$ true
    end
    if hypernym($C_{i\text{head}}$) = $s$ then
      $C_{i\text{head}}$ $\leftarrow$ $s$
      flag $\leftarrow$ true
    end
  end
  if flag then
    insert $s$ into $\Xi$
    remove $s$ from $\Psi$
    flag $\leftarrow$ false
  end
  else if hypernym($s_j$) = $s$ then
  create new cluster $C_{\text{new}}$
  $C_{\text{new,head}}$ $\leftarrow$ $s_j$
  Insert $s$ in $C_{\text{new}}$
  flag $\leftarrow$ true
  break
  end
end

end

/* Prune for overlapping clusters*/

for each $C_i \in \Omega$, do
  for each $C_j \in \Omega$, do
    if MI($C_i, C_j$) $>$ $\mu$ then
      $C_{\text{smaller}}$ $\leftarrow$ $C_j$ if $C_{j\text{mass}}$ $>$ $C_{i\text{mass}}$ then
        $C_{\text{smaller}}$ $\leftarrow$ $C_i$
        end
      remove $C_{\text{smaller}}$ from $\Omega$
    end
  end
end

/* Rank the new clusters*/

$\hat{\Omega}$ $\leftarrow$ $\{\Omega\}$

for each $C_i \in \hat{\Omega}$, do
  for each $C_j \in \hat{\Omega}$, do
    if $C_i\text{density}$ $\geq$ $C_j\text{density}$ then
      swap $\hat{\Omega}[i]$ and $\hat{\Omega}[j]$
    end
  end
end

return $\hat{\Omega}$

/*function DisIntegrate*/

DisIntegrate(cluster $C$):

$\Lambda$ $\leftarrow$ $\{C\}$

while $\exists C_i \in \Lambda \land (C_{i\text{depth}} \leq \delta \lor C_{i\text{breadth}} > \beta)$ do
  for (each node $j \in C_i \land (C_{j\text{depth}} - \text{node}_{j\text{depth}}) = 1)$ do
    $C_j$ $\leftarrow$ child cluster with head $\text{node}_j$
    insert $C_j$ into $\Lambda$
    remove $C_i$ from $\Lambda$
  end
end

return $\Lambda$

---

**ALGORITHM 2: Cluster Pruning**

Initialize:

- $\hat{\Omega}$ $\leftarrow$ $\{\}$ /*array of ranked clusters*/

for each cluster $C \in \Omega$, do
  $\Lambda$ $\leftarrow$ $\{\}$ /*function DisIntegrate defined below*/
  if $C_{\text{head depth}} \leq \delta \land C_{\text{breadth}} > \beta$ then
    DisIntegrate($C$)
  end
  remove $C$ from $\Omega$
end

for each cluster $C_i \in \Lambda$, do
  insert $C_i$ into $\Omega$
end

/* Prune for overlapping clusters*/

for each cluster $C_i \in \Omega$, do
  for each cluster $C_j \in \Omega$, do
    if MI($C_i, C_j$) $>$ $\mu$ then
      $C_{\text{smaller}}$ $\leftarrow$ $C_j$ if $C_{j\text{mass}}$ $>$ $C_{i\text{mass}}$ then
        $C_{\text{smaller}}$ $\leftarrow$ $C_i$
        end
      remove $C_{\text{smaller}}$ from $\Omega$
    end
  end
end

/* Rank the new clusters*/

$\hat{\Omega}$ $\leftarrow$ $\{\Omega\}$

for each cluster $C_i \in \hat{\Omega}$, do
  for each cluster $C_j \in \hat{\Omega}$, do
    if $C_i\text{density}$ $\geq$ $C_j\text{density}$ then
      swap $\hat{\Omega}[i]$ and $\hat{\Omega}[j]$
    end
  end
end

return $\hat{\Omega}$
we assume advertisers arrive sequentially, and index them by \( k \in \{1, \ldots, K\} \). We denote advertiser \( k \) by \( A_k \).

We define an allocation \( S = \{s_1, \ldots, s_K\} \) as a \( k \)-partition of the set of impressions such that,

- for all \( k \in \{1, \ldots, K\} \), \( s_k \in S \); and
- \( \sum_k s_k \leq (n_1, \ldots, n_T) \).

For each \( k \), the vector \( s_k \) represents the set of impressions allocated to advertiser \( A_k \). The second condition ensures that all allocations are feasible with respect to supply constraints.

The utility model considered in the online case requires a mild technical assumption for the analysis to hold true. We present the full utility model as it is expressive enough to capture known features encountered by advertisers interested in guaranteed delivery display ads. Advertiser \( A_k \) is fully characterized by a 5-tuple \((t_k, \tau_k, q_k, b_k, V_k)\) where

- \( t_k \in \{1, \ldots, T\} \) is the desired start date;
- \( \tau_k \in \{1, \ldots, T\} \) is the desired duration;
- \( q_k \in \{1, \ldots, n\} \) is the minimum impressions per round advertiser \( A_k \) seeks starting at \( t_k \) and for \( \tau_k \) rounds;
- \( b_k \in \mathbb{R}^+ \) is the utility drawn from receiving \( q_k \) impressions per round during \( \tau_k \) consecutive rounds; and
- \( V_k \) is the additional value function

\[
V_k : \{0, 1, \ldots, N\} \to \mathbb{R} \\
q \mapsto V_k(q)
\]

where \( V_k(q) \) is the value drawn by advertiser \( A_k \) of being allocated \( q \) additional impressions given that it is guaranteed to receive \( q_k \) impressions per round during \( \tau_k \) consecutive rounds.

Given an allocation \( s_k \in S \), the utility to agent \( A_k \) is

\[
U(s_k; A_k) = b_k + V_k(|s_k| - q_k \tau_k)
\]

provided \( \forall t \in [t_k, t_k + \tau_k - 1] \), \( s_k(t) \geq q_k \). Otherwise the utility to agent \( A_k \) is assumed to be nil.

This utility function accounts for branding activities linked to display advertising that require minimum impression delivery per round. Also, note that ad-fatigue can be modeled by setting \( V_k \) to be negative.

Finally, we say that a function \( f \) is \( C \)-benevolent [117] if:

1. \( f(0) = 0 \) and \( f(p) > 0 \), for all \( p > 0 \); and
2. for all \((\varepsilon, p_1, p_2)\) such that \( 0 < \varepsilon \leq p_1 \leq p_2 \),

\[
f(p_1) - f(p_1 - \varepsilon) \leq \frac{f(p_2) - f(p_2 - \varepsilon) - f(p_1) + f(p_1 - \varepsilon)}{p_2 - (p_1 - \varepsilon)}
\]

It is then easy to see that all non-decreasing convex functions that satisfy the first condition are \( C \)-benevolent. We can now formulate the technical assumption needed for the analysis of our online algorithm. In all that follows, we are interested in advertisers whose utility satisfies the following assumptions:

1. for all \( k \), \( V_k(0) = 0 \); and
2. for all \( k \), \( b_k = f(\tau_k q_k) \), where \( f \) is \( C \)-benevolent.

We finish this subsection by succinctly defining the advanced sales contracts proposed by the publisher in our setting.

**Definition 1 ((Advanced Sales) Contracts):** A contract \( \Gamma(t, \tau, q, p) \) is a guarantee that the publisher will provide (the quantity) \( q \) impressions per round over the (duration) \( \tau \) consecutive rounds starting at round \( t \) at a price of \( p \).

### C. Online Problem

It is clear that the class of contracts from Definition 1 represents a guarantee in the delivery of a minimum number of impressions to a given advertiser over a period of time. Given the hardness results from [116] in the offline case, we assume contracts can be dropped without penalty. We also assume, for all \( t, n_t = n \), where \( n \) is a parameter we will use to characterize the performance of our algorithms. We make this assumption as, for \( n = 1 \), our problem can be easily reduced to that studied by Woeinger in [117]. Finally, we assume advertisers arrive online, and consider the worst-case scenario where no information is known about the distribution of the advertisers’ parameters.

We present a simple online greedy algorithm that achieves a competitive ratio of \( 8n^2 \). We then develop a lower bound of \( n \) for the performance of any deterministic online algorithm, thus showing our algorithm is near-optimal.

**Online Resource Allocation Algorithm (ORA)**

1. If the new coming contract does not conflict with any currently scheduled contracts, schedule it.
2. Otherwise, find\(^3\) the set of contracts with the minimum total profit that if we dropped them from the schedule the new coming contract can be scheduled. If the profit of the new contract is more than twice the total profit of these contracts, schedule it.
3. In the rest of situations, drop the new contract.

Note that Algorithm ORA is a modified simple greedy algorithm: anytime the input changes, the algorithm identifies the “least profitable” set of contracts one would need to drop in order to accept the new one. It accepts the new contract only if the value received is at least twice that of the identified set of contracts. In [113], the authors perform a set of simulations using real data from Yahoo! display advertisement business. Given the performance observed using inputs from real data, one can conclude that the instances typically encountered in a real setting are quite different from those needed to make our algorithm perform poorly.

To expand on the last point, we now provide a lower bound for the performance of any deterministic algorithm in the online setting. For ease of exposition, assume that each advertiser is interested in only one contract. The proof relies on the adversary using two extreme types of contracts:

- **Wide:** An advertiser \( A_i \) is called *Wide* if \( \tau_i = n \) and also \( q_i = n \) (the advertiser is interested in acquiring all impressions available over a short period of time).

- **Long:** An advertiser \( A_i \) is called *Long* if \( \tau_i = n^2 \) and \( q_i = 1 \) (the advertiser is interested in a small number of impressions for a long time).

\(^2\)The proof of the competitive ratio is rather involved: we refer the reader to the original paper [113].

\(^3\)This involves running an algorithm for a knapsack-like problem on the accepted set of contracts.
Note that a Wide advertiser is still interested in a potentially long business period \( (n) \), but said period is very short with respect to that of interest to Long advertisers \( (n^2) \). Also, all contracts will have the same value function \( b = f(\tau q) \).

We will now construct an adaptive adversary that, using only Wide and Long advertisers, ensures that no deterministic online algorithm has a worst case ratio better than \( n \).

**Theorem 1:** Assume \( f \) is a C-benevolent function. Then there is no deterministic online algorithm with a worst case ratio smaller than \( n \).

**Proof:** The proof relies on defining the following (adaptive) adversary. Let \( A \) be any deterministic algorithm used to decide, online, whether to accept or reject a given contract. The adversary implements the following strategy:

- Initially, the adversary sends only Wide contracts.
- Once \( A \) accepts a Wide contract, and as long as the contract is honored by \( A \), the adversary starts sending Long during the active period of the Wide contract.
- If \( A \) decides to drop the Wide contract and accept the Long contract instead, the adversary starts sending Wide contracts again, and continues to do so, as long as the Long contract is honored by \( A \).
- Wide Contracts are submitted as soon as the previous Wide Contract sent by the adversary is finished.
- The adversary will stop sending any more contracts if he either sends \( n \) Wide contracts or \( n \) Long contracts or the worst case ratio is already \( n \).

Let us now provide a simple analysis that shows the worst case ratio cannot be better than \( n \) under such an adaptive adversary. Assume \( A \) decides to honor the first Wide contract, then the worst case ratio is already \( n \) as the the algorithm has declined \( n \) Long contracts, and each Long contract has a value equal to that of a Wide contract.

Now assume \( A \) switches to, and honors, a Long contract. Then it would have lost the opportunity to deliver at least \( n \) Wide contracts, and thus the worst case ratio is already \( n \) again. Thus \( A \) should switch to a Wide contract again.

Given the impression demand of each contract, the impression supply, and the adversary’s behavior, it is clear that only one contract can be allocated at the end of each round. Also, if \( A \) keeps a contract until its expiration, we have seen that the worst case ratio would be at least \( n \). To avoid this, \( A \) must switch before the end of any contract.

Let us now see what the state of the optimal solution is after \( 2n + 1 \) switches. Given that each switch happens between contracts of different type, by the pigeonhole principle, we have seen at least \( n \) contracts of type Long. Finally, \( n \) Long contracts can be executed simultaneously, thus the optimal offline solution will allocate at least \( n^3 \) impressions, when the one picked by \( A \) can allocate at most \( n^2 \) impressions, which completes the proof.

**ACKNOWLEDGMENT**

The author thanks colleagues at Adchemy who have contributed to the development of some of the ideas presented herein. Particular thanks are due to Esteban Arcaute Aizpuru, Abhimanyu Kumar, Alessandro Magnani, and Manas Pathak.