Tel Aviv University
Raymond and Beverly Sackler Faculty of Exact Sciences
The Blavatnik School of Computer Science

Keyword Optimization in Search-Based Advertising Markets

Thesis submitted in partial fulfillment of the requirements
for the M.Sc. degree of Tel-Aviv University
by

Yuval Netzer

The research work for this thesis has been carried out at Tel-Aviv University under the
supervision of
Prof. Yishay Mansour

Acknowledgements

I am heartily thankful to my supervisor, Prof. Yishay Mansour, whose encouragement, guidance and endless patience allowed me to complete this thesis.

I would also like to thank my loved wife and kids Iris, Itai and Avigail, all who have supported me in this research, each in their own way :)

Last, I offer my regards and blessings to all of those who supported me in any respect during the completion of this project.

Table of Contents

Abstract

Search-based advertising is a multi-billion industry which is part of the growing electronic commerce market. In this work, we study the search-based advertising market from the advertiser's point of view. There are three natural participants in the search-based advertising: the advertisers, promoting their products to consumers based on their search queries, the users, which are searching for content, and the search providers, who match the advertisers and users by placing ads in search result pages. It is customary today for the advertisers to pay only for ad clicks. This guides both the advertisers' and the search providers' strategy. The advertisers' strategy, which is the focus of this work, requires them to select keywords, bids and a budget constraint imposed on their advertising spending. We abstract an optimization problem in which the advertisers set a daily budget and select a set of keywords on which they bid. We assume that the cost and benefit of keywords is fixed and known, sidestepping this important strategic issue and focusing on the keyword optimization problem. The advertisers' goal is to optimize the utility subject to its budget constraint. Clearly the advertisers would like to buy the most profitable keywords, subject to the budget constraint. The problem is that there is uncertainty regarding the type and number of queries in a day, and the advertisers have to fix a single policy for the online day. If too few keywords are selected, the advertisers remain with unused budget. If too many keywords are selected, at the time keywords associated with 'good' ads appear, the daily budget may have already been exhausted.

We study the advertisers' keyword optimization problem in three different settings: in an offline problem setting in which all problem parameters are known beforehand, in a stochastic model in which the advertiser knows only some of the parameters of the stochastic model, and in an adversarial model which makes no statistical assumptions about the generation of the query sequences. For each of these models we provide lower and upper bounds to the performance of learning algorithms while using the notion of regret minimization [1].

We support our theoretical results with extensive simulations in a simulated environment while comparing our algorithm's performance against the performance of algorithms suggested in [3].

Chapter 1

Introduction

1.1 Search-Based Advertising

In the last decade, search-based advertising has become a multi-billion industry and a market of major interest for many companies. Search-based advertising is a method of placing online advertisements on Web pages that show results from search engine queries. These advertisements are targeted to match key search terms (keywords) queried on search engines. Consumers often use the search engine to identify and compare purchasing options immediately before making purchasing decisions. The opportunity to present consumers with advertisements tailored to their immediate buying interests encourages consumers to click on search ads instead of unpaid search results, which are often less relevant. This highly focused targeting ability has contributed to the attractiveness of search advertising for advertisers.

Search based advertising usually takes the following form: when an Internet user enters a search query into a search engine, he gets back a page with results, containing both the links most relevant to the query and additional sponsored links which are advertisements. These sponsored links are distinguishable from the "organic" search results presented, and different searches yield different sponsored links. When a user clicks on these sponsored links, he is sent to the advertisers Web page. A company whose ad is displayed pays the search engine only when the consumer clicks on the ad. For each such click, the advertiser pays the search engine a payment determined by the auction mechanism, which is used by the search engines to sell the online advertising.

The three main players taking place in the search-based advertising market are the advertisers, the search engines and the consumers (search engine users):

1. The advertisers are companies or individuals interested in promoting their products to consumers based on their search queries. In most search-based advertising services, a company sets a daily budget, selects a set of keywords, determines a bid price for each keyword, and designates one or more ads associated with each selected keyword. The companies try to maximize their utility of this budget and measure metrics such as cost per click (CPC) and conversion rates (the percentage of clicks that result in a commissionable activity, i.e., sale or lead).

2. The search engines - in order to determine which ads to present given a user query, the search engines conduct auctions to sell ads according to bids received for keywords matching the query and the relative relevance of the user query to the ads in the inventory. Since the search engine is paid only for clicks, it is in its interest to present advertisement on which the user is likely to click. The number of ads that the search

engine can present to a user is limited, and different positions on the search results page yield different results, e.g., an ad shown at the top of a page is more likely to be clicked than an ad shown at the side. The mechanism most widely used by search engines are based on the generalized second-price (GSP) auction [2]. In a GSP auction the bidder pays the minimal bid that is required to get the position he was assigned.

3. Consumers - when a consumer searches using a search engine, their queries are matched against advertiser's selected keywords. The search engines then display the ads associated with the highest quality scores for those keywords on the search result page. The quality scores combine the bids and signals such as the keywords' expected clickthrough rates.

The advertisers, who wish to promote their goods and services via search-based advertising, need to decide between millions of available keywords and a highly uncertain clickthrough rate associated with the ads matched to them. Identifying the most profitable set of keywords, given the daily budget constraint, is a challenging task. This challenging optimization problem is the subject of this work.

In the rest of this chapter we describe related work and our main thesis results.

1.2 Keyword Optimization in Search-Based Advertising

Keyword optimization for search based advertising has been the focus of some theoretical and applicative work.

Rusmevichientong and Williamson ([3]) have formulated a model for keyword selection in search based advertising. In their model, the advertiser has a fixed daily budget and each keyword has fixed known cost and profit. However, the keyword click-through probabilities are unknown. The number of queries appearing in each of the days, as well as the distribution of keywords, are generated probabilistically with known parameters. They justify their assumptions about keyword costs and distribution by identifying multiple public available data sources that may be used to estimate these parameters.

The main result of [3] is to develop an algorithm that adaptively identifies a set of keywords to bid on, based on historical performance. The algorithm uses a prefix of a sorted list of keywords, which are sorted by descending order of profit-to-cost ratios, and uses an approximation algorithm for the stochastic knapsack [4]. They prove that by considering only subsets of keywords that are prefix, they achieve near optimal profits while reducing the size of the decision space from 2^N to N, where N is the number of keywords in the problem. The policy adaptively selects prefix keywords in order to maximize the total expected profit. There is a tradeoff between selecting too few profitable keywords, and not exhausting the entire daily budget, versus selecting too many keywords, and thus losing opportunities to receive clicks from profitable words that may arrive after the daily budget is exhausted. As the click-through rates are unknown, their algorithm balances between selecting keywords that yield high average profits based on past performance,

and selecting previously unused keywords in order to learn about their click-through rates. This exploration-exploitation dilemma is well known in the machine learning literature. Their algorithm mixes between selecting a prefix subset that has an expected cost which is close to the daily budget, using estimations of click-through rates given past observations (exploitation), and a random prefix subset (exploration).

Assuming the cost of each click is sufficiently small compared to the daily budget and that the expected number of searched keywords is close to its mean, they prove that in expectation the algorithm converges to near optimal profits, where the closeness to optimality depends on how well these assumptions are satisfied. They also perform numerical simulations to show that their algorithm outperforms existing multi armed bandit algorithms such as UCB1 [5] and EXP3 [6]. Their main point in this comparison is that the convergence rates of these bandit algorithms, which ignore the special structure of the problem, depends on the number of keywords, which in practice might be very large. In contrast their algorithm depends primarily on the largest l such that the expected cost of the prefix $\{1, 2, \ldots, l\}$ does not exceed the budget. They claim that in practice this number is significantly smaller then the total number of keywords.

Muthukrishnan et al. ([7]) study the keyword optimization problem under various stochastic models. In their work, they put a special emphasize on the stochastic models of the arriving queries, and study the evaluation (given a keyword selection - evaluating the expected profit) and optimization problems (selecting a bid solution which maximizes profit) of such models. They present algorithmic and complexity results for three stochastic models, where in all three models the optimization problem is non convex. Their work also differs from the work of [3] as they solve the problem in advance rather then by adaptive learning of the parameters. The three stochastic models discussed in [7] are:

1. Fixed proportions model - in this model the only random variable is the total number of clicks in a day and the proportions of clicks of each of the keywords remains constant. For this model they first prove that an optimal solution is a fractional prefix. In a fractional prefix the advertiser bids on all sorted keywords up to a selected keyword. For that last keyword in the prefix, the advertiser assign a probability p which is the probability that he will bid on queries from that keyword. They prove, using an interchange argument, that there is a fractional prefix which is optimal. They further show that finding the optimal prefix solution is non trivial since there are local maxima. Therefore setting the number of clicks to its expectation for each of the keywords and solving a deterministic problem, or alternatively, the greedy procedure which starts with an empty solution and keeps adding keywords - both fail due to possible multiple local maxima. Their solution to the optimization problem overcomes the infinite number of possible fractional prefix solutions by showing that it is sufficient to evaluate a polynomial set of "interesting" solutions which depends on the number of keywords and the number of different values the total number of clicks can take.

2. Independent keywords model - in this model, the number of clicks for each keyword has it's own probability distribution (which can be different for different keywords). The key distinguishing feature of this model is that for different keywords, the number

of clicks are independent. For this model they prove that the prefix solution may not be an optimal solution and there exists a prefix solution which is a 2-approximation to the optimal solution. They also present a PTAS for evaluating the expected profit of a proposed policy which combined with the previous result implies a $(2 + \epsilon)$ approximation algorithm.

3. Scenario model - this model attempts to capture the full generality of a joint distribution without the large number of bits needed to represent an arbitrary joint probability distribution. It does so by a limited (polynomial) number of scenarios in which the exact number of clicks for each word is given. A single scenario is taken from a given probability distribution over scenarios. In this model, the authors prove two negative results: using a reduction from $CLIQUE$ the keyword optimization problem under the scenario model is NP-hard, and the gap between the optimal fractional prefix solution and the optimal (integer or fractional) solution to the optimization problem can be arbitrarily large.

1.3 Search related auctions

There is variety of work on search related auctions in the presence of a limited budget but it has primarily focused on the game theoretic aspects and on the search engines perspective:

In [2] the authors investigate the "generalized second price" (GSP) auction used by the search engines to sell online advertising and show that, unlike the Vickrey-Clarke-Groves mechanism, it doesn't have an equilibrium in dominant strategies and truth telling is not an equilibrium of GSP.

Blum et al. [8] consider the problem of revenue maximization in online auctions from the auctioneer's point of view. They apply online learning techniques to an online auction problem in which bids are received and dealt with one-by-one. They present algorithms with asymptotically constant-competitive ratios. They also study a related problem, called posted-price auctions, in which the auctioneer posts a price to each bidder and the bidder either accepts or rejects the auctioneer's offer. They show algorithms for the posted price model with similar asymptotically constant competitive ratios with respect to the optimal fixed price revenue.

Metha et al. [9] derived an optimal online algorithm for the revenue maximization problem faced by search engines, deciding which ads to display with each query. The problem is to assign queries arriving during the day to advertisers, while respecting their daily budgets. Their optimal online algorithm achieves a competitive ratio of $1 - 1/e$ using a generalization of the online bipartite matching problem. They generalize their analysis to more realistic settings in which a bidder pays only when the user clicks on the ad, the search engine charges the bidder with the next highest bid, while maintaining the same competitive ratio.

Pandey and Olston [10] consider a similar problem in which the search engine has to handle new advertisers whose degree of appeal to users is yet to be determined. They study the tradeoff between exploration and exploitation as a multi-armed bandit problem

and extend traditional bandit formulations to account for budget constraints that occur in search engine advertising markets.

Ganchev et al. [11] utilize sponsored search data drawn from a wide array of Yahoo! Overture auctions and performed an exploratory analysis attempting to characterize and understand real world search auction data. They examined how bids are distributed, what kinds of models of advertiser value can reasonably be proposed and found an aggregate exponential decay of prices across auctions. Nevertheless, they showed that aggregate exponential decay does not fully describe bidding behavior on a per-auction basis where deviations are hypothesized to occur due to strategic bidding.

Borgs et al. [12] consider the problem of online keyword advertising auctions among multiple bidders, where the bidder are using a simple heuristic to optimize their utility. They show that existing auction mechanisms combined with this heuristic can experience cycling (as been observed empirically in current systems) and propose a modified class of mechanisms with small random perturbations which provably converges in the case of first price mechanisms and empirically converge in second price mechanisms.

1.4   Main results

The rest of the thesis is organized as follows. In Chapter 2 we study the advertiser's keyword optimization problem as an offline problem in which all parameters are known. We show that even in this simple model, the keyword optimization problem is NP-complete. We analyze the performance of a special family of solutions - prefix policies. In a prefix policy, an advertiser buys a prefix of a list of keywords, where the keywords are sorted by descending order of their profit to cost ratios. We present near-tight bounds for prefix policies, under the reasonable assumption that the ratio between the profit of the most valuable and the least valuable keywords is bounded by some constant.

In Chapter 3 we analyze a stochastic formulation of the problem in which the advertiser has only partial information about the parameters of the stochastic model. We show that in the stochastic model, prefix-solutions are optimal. Using the notion of 'regret', i.e. the difference between the expected profit of the optimal fixed policy in hindsight and the advertiser's expected profit, we derive on-line algorithms which guarantee sublinear regret in the number of days.

In Chapter 4 we study an adversarial setting. Under some assumptions about the length of the sequence and on the feedback an algorithm receives, we show that the regret of an advertiser compared to the best fixed policy, may be as big as $O(\sqrt{Tk})$, where T is the number of sequences (days) and k is the number of keywords in the problem instance. We show that this bound is tight (up to a logarithmic factor) by proving a matching upper bound.

Chapter 5 tests some of our ideas in a simulated environment, adopting the stochastic model in [3]. Our empirical results support our theoretical findings. We show that a model free learning algorithm can quickly converge to a near optimal policy while enjoying robustness which is lacking for model based algorithms.

Chapter 2

The Offline Advertiser's Keyword Optimization problem

In this chapter we analyze the advertiser's keyword optimization problem in an offline model. In this formulation of the problem, all parameters of the problem are known to the advertiser beforehand. Unlike the real world problem and the models presented in the next chapters, the optimization can be done offline. In this model, for each of the days the advertiser receives a detailed sequence with the query words that will appear in that day. The advertiser's task is to select a good fixed set of keywords to buy. His main constraint is his daily budget.

We show that even in this simple model, the keyword optimization problem in NP-complete. We further present tight bounds for a special family of solutions which we call "prefix-solutions", which albeit being optimal for single days, perform poorly when the per keyword profits are unbounded.

2.1 Model definition

A keyword optimization problem instance consists of a fixed set of k keywords $W = \{w_1, w_2, \ldots, w_k\}$ with associated fixed profits $\overrightarrow{\pi} = \{\pi_1, \pi_2, \ldots, \pi_k\}$ and costs $\overrightarrow{c} = \{c_1, c_2, \ldots, c_k\}$, where all costs and profits are positive integers, e.g., cents. We define a daily query sequence $s = (q_1, q_2, \ldots, q_m)$, as a sequence of queries where each query q_i is one of the keywords in W, i.e., $q_i \in W$ and $s \in W^m$. An advertiser has a fixed total daily budget $B_{day} > 0$, which limits his spend in a single day, and which resets daily.

A keyword optimization problem is therefore formalized as $\Gamma = (W, \overrightarrow{\pi}, \overrightarrow{c}, B_{day}, T, (m_t)_{t=1}^T, S)$ where T is the number of days in the problem and $S = (s_t)_{t=1}^T$, i.e., for every day $t \in [1, T]$, s_t is the t'th daily sequence with m_t queries: $(q_{t1}, q_{t2}, \ldots, q_{tm_t})$.

We distinguish between queries and keywords - a query q refers to an instance of a keyword w in a specific daily sequence s. The subscript indices tj of q_{tj} refer to the location j in the sequence s_t of day t, while the subscript index i of w_i refers to the keyword index in W.

The advertisement mechanism works as following: at the beginning of each day t, the advertiser's daily budget resets to B_{day}. For each query q_{tj} in the daily sequence s_t which is in the subset of keywords selected by the advertiser, the advertiser is charged with the keyword's cost $c(q_{tj})$ which is subtracted from his daily budget. Day t ends when either the advertiser's budget is exhausted or when there are no more queries left in the daily sequence s_t. The Advertiser's profit is the sum of daily profits gained for each of the queries bought in the sequences s_t, where the profit for each of the queries is determined according

6

to the queries' keyword matching profit.

In addition to the overall daily budget B_{day} we allow the advertiser to have a per keyword budget $\overrightarrow{B} = \{B_1, B_2, \ldots, B_k\}$. This additional constraint means that after using B_i budget on queries matching keyword w_i, no more queries which match keyword w_i are bought. These budgets also reset daily, and their motivation is described later in this chapter. For simplicity, we assume these keyword budgets are an integral multiplicative factor of the keyword costs, i.e., $\forall i, B_i = c_i \cdot x$ for some integer x.

The advertiser's policy $\overrightarrow{B}$ is therefore defined as the per keyword budgets $\{B_1, B_2, \ldots, B_k\}$, i.e., the subset of keywords selected by the advertiser is determined to be the w_i for which $B_i > 0$. This is in addition to the daily budget B_{day}.

The advertiser's profit is calculated by summing over the individual keyword profits of all queries which were bought by the advertiser. The quantities of each of the bought keywords are defined by the order of the queries in a given day and by the remaining total and per keyword budgets for each query. The costs of the queries are disregarded with respect to the profits and we only use them to limit the number of allowed queries an advertiser may buy (the cost limit is the total daily budget B_{day}).

Notice that when limiting the budget of a specific keyword w_i, the advertiser buys all of the instances of this keyword until it exhausts either of the keyword's Budget B_i or the total budget B_{day}. The length m_t of each of the daily sequences s_t may vary considerably, and as the advertiser is not allowed to select **which** of the instances of a keyword he buys - a different order for the queries in a sequence may exhaust the daily budget in different times. Also, in a given day, for a given query, if the advertiser still has remaining budget in his total daily budget, and he has remaining budget for the keyword matching this query, the advertiser must buy that query, and can not 'save' budget for 'better' queries. More precisely the advertiser fixes its policy (budgets) at the start of each day, and cannot modify it during the day.

Let $I(B(q_{tj}, j) \geq c(q_{tj})) \equiv buy(q_{tj})$ denote the event that query q_{tj} is being bought. I.e., I is an indicator function which in our case indicates that the budget of the keyword matching query q_{tj} at the time the query arrives is at least $c(q_{tj})$, where $c(q_{tj})$ is the cost of the keyword w_i in W matching q_{tj}, and $B(w_i, j)$ is the budget left for keyword w_i at time j.

Formally, given a query sequence s_t, and budgets B_{day} and $\overrightarrow{B}$, let y_t be the largest integer such that:
$\sum_{j=1}^{y_t} c(q_{tj}) \cdot buy(q_{tj}) \leq B_{day}$. The profit of using the policy $\overrightarrow{B}$ in day t is

$$\Pi(B_{day}, \overrightarrow{B}, s_t, \overrightarrow{\pi}, \overrightarrow{c}) = \sum_{j=1}^{y_t} \pi(q_{tj}) \cdot buy(q_{tj}),$$

where $\pi(q_{tj})$ is the profit of the keyword in W matching q_{tj}.

We use the abbreviation $\Pi(\Gamma, \overrightarrow{B})$ as the total profit of policy $\overrightarrow{B}$ over the T days in the problem instance Γ, i.e.,

$$\Pi(\Gamma, \overrightarrow{B}) = \sum_{t=1}^{T} \Pi(B_{day}, \overrightarrow{B}, s_t, \overrightarrow{\pi}, \overrightarrow{c}).$$

The advertiser's offline optimization problem is therefore the following: given $\Gamma = (W, \overrightarrow{\pi}, \overrightarrow{c}, B_{day}, T, (m_t)_{t=1}^T, (s_t)_{t=1}^T)$, to find an optimal fixed (over all of the days) policy $\overrightarrow{B}$ that maximize its overall profit $\Pi(\Gamma, \overrightarrow{B})$.

We define $OPT(\Gamma)$ as the maximal profit over all policies $\overrightarrow{B}$, i.e., $OPT(\Gamma) = \max_{\overrightarrow{B}} \Pi(\Gamma, \overrightarrow{B})$. A β-approximation for OPT guarantees for any instance Γ a profit of at least $\beta \cdot OPT(\Gamma)$, i.e., $\Pi(\Gamma, \overrightarrow{B}) \geq \beta \cdot OPT(\Gamma)$.

The following claim gives motivation to our per keyword budget model. It shows that an advertiser with (only) a total daily budget and no per keyword budget may loose as much as a third of the profit of an advertiser who can set a per keyword budget:

Claim 1 *There is a problem instance Γ, such that $\Pi(\Gamma, \{B_{day}, \ldots, B_{day}\}) \leq \frac{2}{3} OPT(\Gamma)$.*

Proof: Consider the simple case in which there are two keywords $W = \{w_1, w_2\}$ with profits $\overrightarrow{\pi} = \{4, 2\}$, costs $\overrightarrow{c} = \{1, 1\}$, and the total daily budget $B_{day} = 2l$ where $l \gg 1$. Given a single day sequence $s_1 = \{\underbrace{w_2, \ldots, w_2}_{2l \ times}, \underbrace{w_1, \ldots, w_1}_{l \ times}\}$, an advertiser with a total daily budget B_{day} may earn at most a total profit of $4l$ (either by buying w_1, i.e. $W_{buy} = \{w_1\}$, or by buying both w_1 and w_2, i.e., $W_{buy} = \{w_1, w_2\}$). An advertiser who uses a per keyword budget and buys the keywords with a budget vector $\overrightarrow{B} = \{l, l\}$ earns a total profit of $6l$. ∎

2.2 Solving the offline deterministic keyword problem

In order to study the computational hardness of the problem, we define the advertiser's offline keyword **decision** problem which matches our optimization problem: given $\Gamma = (W, \overrightarrow{\pi}, \overrightarrow{c}, B_{day}, T, (m_t)_{t=1}^T, (s_t)_{t=1}^T)$, decide whether there exists per keyword budgets $\overrightarrow{B}$ for which the total profit $\Pi(\Gamma, \overrightarrow{B}) \geq K$.

Claim 2 *The advertiser's offline deterministic keyword **decision** problem is NP-complete.*

Proof: The Knapsack decision problem is known to be NP-Complete (see e.g. [13] which refer to the basic offline Knapsack problem in which items may be selected at most once). We reduce the Knapsack problem to the offline deterministic advertiser keyword **decision** problem.

Formally, in the Knapsack problem we are given a set of items $S = \{1, \ldots, m\}$ with weights c_i and values p_i and are required to decide whether there exists a subset of items $S' \subseteq S$ for which $\sum_{i \in S'} c_i \leq C$ and $\sum_{i \in S'} p_i \geq K$.

For each item i with weight c_i and value p_i in a Knapsack problem instance, we add to the keywords set W in the advertiser budget decision problem instance a keyword w_i which we define with profit $\pi_i = p_i$, and cost c_i.

We build a single day sequence s_1 with the following queries: for each item with index i we add a single query of type w_i. The daily budget is taken to be the knapsack capacity C, i.e. $B_{day} = C$.

If exists a subset S' of items for which $\sum_{i \in S'} c_i \leq C$ and $\sum_{i \in S'} p_i \geq K$, the solution for the advertiser's keyword problem will be to set $B_i = B_{day}$ for all w_i, $i \in S'$ and $B_i = 0$ for all w_i, $i \notin S'$. In that case, the advertiser will buy all keywords in S' and gain a profit of at least K.

If $\vec{B}$ is a solution to the advertiser's keyword problem, we build S' such that $i \in S'$ iff $B_i \geq c_i$. In that case, obviously the advertiser buys the keywords in S' which lead to a total profit of at least K with at most C cost. In the matching knapsack problem, the profit from S' is also at least K and the cost is at most C.

Therefore, there exists a solution to the advertiser budget decision problem with budget $B_{day} = C$ and a total profit greater than K iff there exists a subset of items in the knapsack problem with capacity at most C and a total profit greater or equal to K.

This advertiser's offline deterministic decision problem is obviously in NP and the reduction can be done in polynomial time, therefore it is NP-complete. ∎

Thus the matching advertiser's offline **optimization** problem is NP-hard.

One might suggest a greedy algorithm that sorts the keywords by their profit to cost ratio, and iteratively, adds the next keyword according to this order. In the next problem instance, we show that in that case OPT may earn as much as twice the profit of such a greedy algorithm. For the knapsack problem, a small variant on the greedy approach yields a 2-approximation. That approximation scheme is not valid in our problem settings as the advertiser adwords optimization problem requires solving multiple knapsack-like problems simultaneously. Providing an approximation scheme that holds for the advertiser adwords optimization problem is an open problem.

Greedy Keyword Algorithm

1. Sort keywords in non increasing order of $\frac{\pi_i}{c_i}$.

2. Greedily add keyword w_i by setting B_i, $B_i \in [0, 1, \ldots, B_{day}]$ which maximizes the overall profit (notice that when adding a keyword, the profit may decrease, as in query series where the budget is already exhausted, adding worse keywords will be on the expense of the better queries).

Claim 3 *Greedy Keyword Algorithm may loose as much as half the profit of the optimal policy.*

Proof: Assume 3 keywords w_1, w_2, w_3 with $\vec{c} = \{m/2 + \varepsilon, m/2, m/2\}$, $\vec{\pi} = \{(1+\varepsilon) \cdot (m + 2\varepsilon), m, m\}$, $B_{day} = m$, and a single repeating sequence $s = (w_1, w_2, w_3)$.

The profit to cost ratios are $(2(1+\varepsilon), 2, 2)$. I.e., the list is already sorted by the profit to cost ratios. Therefore, the greedy algorithm will first add w_1 by setting $B_1 = B_{day}$, leading to the policy $\vec{B} = \{m/2 + \varepsilon, 0, 0\}$. After doing so, the greedy algorithm will not be able to benefit from w_2, w_3. On the other hand, OPT may select the policy $\vec{B} = \{0, m/2, m/2\}$ which earns a total profit of $2 \cdot m$ while the greedy algorithm gains only $(1+\varepsilon) \cdot (m + 2\varepsilon)$.
∎

One may claim that the failure of the greedy algorithm in the previous problem instance is due to the large query costs used, compared to the daily budget B_{day}. That is indeed in contrary to the real world problem where the cost of a single query is usually very small relative to the daily budget, i.e., $\forall i, c_i \ll B_{day}$. This claim is even more evident for the high budget advertisers. In order to study this more realistic setting, we therefore normalize the costs to 1 and redefine the profits as the profit to cost ("bang-per-buck") ratio $\pi_i' = \pi_i/c_i$ where each instance of keyword w_i is replaced with c_i instances of the normalized keyword w_i' with profit π_i'.

In the next claim, we show that the impact of this normalization is negligible:

Claim 4 $\forall s, B_{day}, \vec{B}, \vec{\pi}, \vec{c},$

$$|\Pi(B_{day}, \vec{B}, s', \vec{\pi'}, \vec{1}) - \Pi(B_{day}, \vec{B}, s, \vec{\pi}, \vec{c})| \leq \max_i \pi_i$$

where s' is created from sequence s by replacing each instance of keyword w_i with c_i instances of the normalized keyword w_i', i.e, with c_i instances of a keyword with profit π_i' and unit cost.

Proof: As we restricted keyword budgets to be multiplicative factors of the keyword costs, the per keyword budgets will not affect our normalization scheme. Nevertheless, using the normalized costs may result with a fraction of the last keyword w_i being bought in the sequence s' whenever in s remains a budget $B_{day} < c_i$ yet $B_{day} \geq 1$, that can not be used as keyword fractions are not allowed. ∎

This normalization simplifies our analysis and results in profit analysis inaccuracies of up to an additive term of at most $\sum_{t=1}^{T} \max_i \pi_i \leq T\pi_{max}$ in the offline model, where π_{max} is the maximal profit of a single keyword.

For simplicity of notation, in the rest of this chapter, we use the notation π_i for those normalized profits and sort the keywords in a non-increasing order of the normalized profits, i.e., $\forall i, i'$ if $i < i'$ then $\pi_i \geq \pi_i'$.

2.3 Prefix policies

As we are especially interested in simple structured policies - we examine the case in which the keywords have been normalized and sorted in a decreasing order of "bang-per-buck" ratios and our algorithm chooses to buy a set of keywords of the form $w_1, \ldots, w_i$ for some integer i with the budgets $\vec{B} = \{\underbrace{B_{day}, \ldots, B_{day}}_{i-1\ times}, b, \underbrace{0, \ldots, 0}_{k-i\ times}\}$. I.e., only a prefix of the keywords is bought, all keywords but the last are limited with the daily budget B_{day}, and the last keyword has a budget b.

We call such a set a prefix solution and notate the prefix solution which buys all words up to word w_i with maximal budget B_{day}, and word w_i with budget b, $PRE(B_{day}, i, b)$.

Claim 5 *Let Γ include a single day list s_1, there exists a prefix solution $PRE(B_{day}, i, b)$ which has an optimal profit, $\Pi(\Gamma, PRE(B_{day}, i, b)) = OPT(\Gamma)$.*

Proof: Let $i^* = arg \max_{j<k} \{\sum_{i=1}^{j} n_i \leq B_{day}\}$, where n_i is the number of instances of keyword w_i in the sequence s_1.

We select the prefix solution $PRE(B_{day}, i^*+1, B_{day} - B')$ where $B' = \sum_{j=1}^{i^*} n_j$, i.e., buy all keywords $w_i, i \leq i^*$, with budget B_{day}, and buy the keyword w_{i^*+1} with budget $B_{day} - B'$. If not all queries are bought, the daily budget is fully exhausted. It is also obvious that increasing budget for less valuable words (as the words are sorted by their profits) may only decrease the total profit. ∎

For the next two claims, we define the problem instance Γ_{WC} (following [7]):

Let k (the number of keywords) be some even integer, $\pi_i = m^{k-i}$ and $c_i = 1, \forall i \in [1, k]$, and let the daily budget $B_{day} = l$.

The daily sequences $S = (s_t)_{t=1}^{T}$ consists of the following $k/2$ query sequence lists (each with different number of occurrences as stated in the table):

Day list	Number of occurrences	Sequence
s_1	1	$\underbrace{w_2, \ldots, w_2}_{l\ times}, \underbrace{w_1, \ldots, w_1}_{l\ times}$
s_2	m^2	$\underbrace{w_4, \ldots, w_4}_{l\ times}, \underbrace{w_3, \ldots, w_3}_{l\ times}$
$\vdots$	$\vdots$	$\vdots$
s_i	m^{2i-2}	$\underbrace{w_{2i}, \ldots, w_{2i}}_{l\ times}, \underbrace{w_{2i-1}, \ldots, w_{2i-1}}_{l\ times}$
$\vdots$	$\vdots$	$\vdots$
$s_{k/2}$	m^{k-2}	$\underbrace{w_k, \ldots, w_k}_{l\ times}, \underbrace{w_{k-1}, \ldots, w_{k-1}}_{l\ times}$

Lemma 6 *Any prefix solution $PRE(B_{day}, i, b)$ has a profit:*

$$\Pi(\Gamma, PRE(B_{day}, i, b)) \leq (1/m + 2/k) \cdot OPT(\Gamma)$$

.

Proof: For Γ_{WC}, OPT which buys only the odd keywords earns for day lists of type s_i the profit of keyword w_{2i-1}, $l \cdot m^{2i-2}$ times, the total of which is : $l \cdot m^{2i-2} \cdot m^{k-(2i-1)} = l \cdot m^{k-1}$. This gives a profit of $k/2 \cdot l \cdot m^{k-1}$ during the entire S.

A prefix solution which selects prefix $PRE(B_{day}, i^*, B_{day})$ where i^* is odd, i.e., $i^* = 2i-1$, earns as OPT for a day of type s_i, i.e., $l \cdot m^{k-1}$. For each other day of type $s_{i'}, i' \neq i$ - it either earns $m^{2i'-2} \cdot l \cdot m^{k-2i'} = l \cdot m^{k-2}$ (for the even keyword $2i'$) or it earns nothing (if $2i' - 1 > i^*$).

Therefore, the best prefix is of the form $i^* = k - 1$, and the prefix policies are bounded by:

$$\Pi(\Gamma, PRE(B_{day}, i, b)) \quad \leq \quad \Pi(\Gamma, PRE(B_{day}, k-1, b))$$

$$
\begin{aligned}
&= l \cdot m^{k-1} + (k/2 - 1) \cdot l \cdot m^{k-2} \\
&\leq (\frac{2}{k} + \frac{1}{m})\frac{k}{2}l \cdot m^{k-1} \\
&= (\frac{2}{k} + \frac{1}{m}) \cdot OPT(\Gamma).
\end{aligned}
$$

We conclude the proof by noting that a prefix solution which selects prefix $PRE(B_{day}, i^*, B_{day})$ where i^* is even, i.e., $i^* = 2i$ buys the same queries as does a prefix solution with $i^* = 2i-1$ except for days s_i in which a less profitable keyword w_{2i} is bought (instead of w_{2i-1}), i.e., $PRE(B_{day}, 2i, B_{day}) \leq PRE(B_{day}, 2i - 1, B_{day})$. ∎

Claim 7 *A prefix solution is an $\Theta(\frac{1}{k})$ approximation.*

Proof: We first show that there exists a problem instance Γ with k keywords such that any prefix solution $\overrightarrow{B}$ has profit $\Pi(\Gamma, \overrightarrow{B}) \leq 3 \cdot OPT(\Gamma)/k$.

Consider Γ_{WC}, with $m = k$, by Lemma 14, we get that any prefix solution $PRE(B_{day}, i, b)$, has $\Pi(\Gamma, PRE(B_{day}, i, b)) \leq OPT(\Gamma) \cdot (3/k)$.

For the upper bound, if there are k keyword types, there is at least one keyword type w_i for which $1/k$ of the total profit is gained by OPT. We select the prefix set $\{1, 2, \ldots, i\}$. It is guaranteed that for each query we will get no less than π_i and in total at least $1/k$ of OPTs profit. ∎

Assuming unbounded single keyword profits seems unrealistic, and indeed if we restrict the ratio between single keyword profits, we get the following stronger result:

Claim 8 *For $k \geq 4$, a prefix solution is an $\Theta(\frac{1}{\log(\pi_1/\pi_k)})$ approximation, where π_1 and π_k are the maximal and minimal keyword profits, respectively. Also, there exists a problem instance Γ for which $\Pi(\Gamma, PRE(B_{day}, i, b)) = O(\frac{\log(\log(\pi_1/\pi_k))}{\log(\pi_1/\pi_k)} \cdot OPT(\Gamma))$*
For $k = 3$ there is an instance Γ in which a prefix solution has

$$
\tfrac{1}{3} \cdot OPT(\Gamma) \leq \max_{i,b} \Pi(\Gamma, PRE(B_{day}, i, b)) \leq \tfrac{1}{2 - \frac{\pi_k}{\pi_1}} \cdot OPT(\Gamma).
$$

We note that for $k = 3$ and large enough π_{max}, a prefix solution may loose as much as half of the profit earned by OPT.

Proof: For the lower bound we again use Γ_{WC}, by Lemma 14, now with $m = k - 1$. We have $\pi_1/\pi_k = (k - 1)^{k-1}$, i.e., $\log(\pi_1/\pi_k) = (k - 1)\log(k - 1)$ and $\log(\log(\pi_1/\pi_k)) = \log(k - 1) + \log(\log(k - 1))$.

Also, we notice that for $k > e + 1$, $\log(\log(k - 1)) > 0$ implying $\log(k - 1) = \log(\log(\pi_1/\pi_k)) - \log(\log(k - 1)) > \log(\log(\pi_1/\pi_k))$.

Therefore, using Lemma 14, for any prefix solution $PRE(B_{day}, i, b)$, for $k > e + 1$, we get that the total expected profit is bounded by:

$$
\begin{aligned}
\Pi(\Gamma, PRE(B_{day}, i, b)) &\leq (\frac{1}{m} + \frac{2}{k}) \cdot OPT(\Gamma) \\
&= (\frac{1}{k - 1} + \frac{2}{k}) \cdot OPT(\Gamma)
\end{aligned}
$$

$$\leq \quad \frac{3}{k-1} \cdot OPT(\Gamma)$$

$$\leq \quad \frac{3 \log(\log(\pi_1/\pi_k))}{\log(\pi_1/\pi_k)} \cdot OPT(\Gamma)$$

For the upper bound we first define a 'bucket' of keywords as the subset of keywords which have a profit in the range $2^l \leq \pi_i < 2^{l+1}$. Namely, we analyze the simple 'Bucket prefix' solution which partitions all keywords into adjacent buckets in which the max profit is no more than twice the profit of the min profit. There are at most $\lceil \log \frac{\pi_1}{\pi_k} \rceil$ buckets.

Therefore, there is a bucket which is responsible for at least $1/\lceil \log \frac{\pi_1}{\pi_k} \rceil$ of the total profit of the optimal (non prefix) solution. We compare it's profit to the prefix solution which buys all keywords in the buckets until and including of this bucket. For each of the keywords the prefix solution eventually buys, he is guaranteed to get at least $1/2$ of the profit of OPT, overall leading to a total profit of at least $\frac{1}{2 \cdot \lceil \log \frac{\pi_1}{\pi_k} \rceil}$ of OPT.

For the case where $k = 3$, for the lower bound we use the following setting: there are 3 keywords w_1, w_2, w_3 with $\overrightarrow{\pi} = \{\pi_{max}, 1, 1\}$, $B_{day} = l$.

Day list	Number of repetitions	Sequence
s_1	1	$\{\underbrace{w_2, \ldots, w_2}_{l \; times}, \underbrace{w_1, \ldots, w_1}_{l \; times}\}$
s_2	$\pi_{max} - 1$	$\{\underbrace{w_3, \ldots, w_3}_{l \; times}\}$

OPT uses $\overrightarrow{B} = \{B_{day}, 0, B_{day}\}$ and earns $(2\pi_{max} - 1)l$ whereas any prefix solution earns no more than $\pi_{max}l$ simply by considering all three prefix solutions. The upper bound for $k = 3$ is the same as in the general case, only that now we have 3 keywords of which at least one is responsible to at least $1/3$ of the optimal profit of OPT. ∎

Chapter 3

Online Stochastic Keyword Optimization

In this chapter we analyze a more realistic model, inspired by the model in [3]. This model introduces a few changes: rather than being given a problem instance to be solved offline, the advertiser is required to adaptively change it's policy in an online manner. We also assume that the series of query sequences $S = (s_t)_{t=1}^T$ is taken from a distribution with a known model but with unknown parameters.

We show that the prefix solutions we introduced in Chapter 2, are optimal in this model when considering the family of daily fixed policies (where an advertiser is not allowed to change his policy during a day, but is allowed to change the policy between days). We also show that finding the parameters for this optimal prefix solution is not trivial, as there might exist multiple local maxima for the expected reward as a function of the prefix parameters.

We further show that knowing the distribution over the length of the days is valuable, as a policy which does not know at the beginning of each day, the length of the daily sequence of queries, may earn a negligible fraction of the profit of a policy which does know the length of the daily sequences.

We adopt the notion of External Regret [1] for comparisons between the performance of online algorithms and the performance of the best single prefix policy in retrospect. We end this chapter by providing an algorithm, based on UCB1 [5], which achieves $O(k^{\frac{1}{2}} T^{\frac{2}{3}} lnT)$ regret compared to the optimal solution, where k is the number of keywords in the problem and T is the number of days. We also provide a regret bound of $O(T^{\frac{3}{4}} lnT)$, which is independent of the number of keywords k. These bounds can be compared to our $\Omega(k^{\frac{1}{2}} T^{\frac{1}{2}})$ lower bound (which is proved in Appendix A).

3.1   Model definition

In the online stochastic model, each of the daily sequences s_t in the series S, is sampled from the following model.

Let T be the number of daily sequences in the problem instance (which we assume is very large, i.e., we are interested in the asymptotic behavior). For each day $t \in [1, T]$, we first sample the length of the sequence $m_t = |s_t|$ of day t, from an unknown distribution P with a finite support. Given the length $m_t = |s_t|$, the sequence $s_t = (q_1, q_2, ..q_{m_t})$ is generated using a multinomial distribution: each query q_j, $1 \leq j \leq m_t$, is sampled i.i.d. from the set of keywords $W = \{w_1, w_2, \ldots, w_k\}$ using a multinomial distribution $\overrightarrow{\lambda} = \{\lambda_1, \lambda_2, .., \lambda_k\}$, i.e., we have that $Pr[q_j = w_i] = \lambda_i$.

14

Rather than selecting a fixed set of keyword budgets $\vec{B}$, as in the offline deterministic model, the advertiser is allowed to adaptively change the daily policy in which he may select a daily subset of keywords to buy, i.e., for each day $t \in [1, T]$ the advertiser selects a daily policy $\vec{X^t}$. Given the stochastic nature of the model, instead of the per keyword budgets $\vec{B}$ introduced in Chapter 2, we allow the advertiser to buy probabilistically queries matching keyword w_i using a 'weight' $b_i \in [0, 1]$ associated with keyword w_i. The daily policy vector of day t, $\vec{X^t} = (b_1, b_2, \ldots, b_k)$ defines a vector of probabilities for buying instances of keywords in W, i.e., a query $q = w_i$ is bought with probability b_i (assuming the advertiser has remaining budget when that query arrives).

The advertiser is required to select the daily policy before the daily query sequence arrives. The model assumes the advertiser knows the associated costs $\vec{c}$ and profits $\vec{\pi}$ of the keywords in W, but does not know the distribution $\vec{\lambda}$ of the keywords in W, and the distribution P over the length of the daily sequences s_i, i.e., $\vec{X^t} = f(W, \vec{\pi}, \vec{c}, B_{day}, (\Pi_j)_{j=1}^{t-1})$, where Π^j is the daily profit of day j. Also, at the end of each day, the advertiser is only given a single reward scalar which is the total profit for that day (without additional information such as the frequency of the keywords, or the amount of used budget, etc.). As P and $\vec{\lambda}$ are unknown to the advertiser, he is therefore required to adapt to the input, and to learn his policy online.

Formally, the advertiser's online stochastic keyword optimization problem is the following. Given a stochastic problem instance $\Phi = (W, \vec{\pi}, \vec{c}, B_{day}, T, P, \vec{\lambda})$ where $P, \vec{\lambda}$ are unknown to the advertiser, for each day $t \in [1, T]$ the advertiser has to select a daily policy $\vec{X^t}$ after which he receives a daily profit Π^t. The advertiser's goal is to maximize his overall profit, i.e., $\sum_{t=1}^{T} \Pi^t$.

We will denote the expected total profit of the policy $(\vec{X^t})_{t=1}^{T}$ in the T days of the problem instance Φ as $\Pi(\Phi, (\vec{X^t})_{t=1}^{T})$, where the expectation is over the distributions P and $\vec{\lambda}$. Let $OPT(\Phi)$ be the maximal expected profit over all fixed policies $\vec{X}$, i.e., $OPT(\Gamma) = \max_{\vec{X}} \Pi(\Gamma, (\vec{X}, \ldots, \vec{X}))$. As in the offline model, we assume that the cost of a single query is small relative to the daily budget, i.e., $\forall i, c_i \ll B_{day}$. Therefore we use the same normalization described there, i.e., $\forall i, c_i = 1$.

3.2   Prefix policies

As in the deterministic offline problem, we are especially interested in simple prefix policies where we assume that the keywords are sorted in a non increasing profit order. These are defined as policies in which the advertiser fully buys (as long as he has remaining daily budget) all instances of all keywords up to some daily selected index $ind(t)$. From keyword $w_{ind(t)}$, in expectation, only a b_t fraction of the instances are bought (until the budget is exhausted), where $b_t \in (0, 1]$, i.e., $X^t = (\underbrace{1, \ldots, 1}_{ind(t)-1 \; times}, b_t, \underbrace{0, \ldots, 0}_{n-ind(t) \; times})$. We denote such a daily prefix solution as $PRE(B_{day}, i, b)$.

We next show that prefix solutions are indeed optimal within the space of daily fixed policies (where an advertiser is not allowed to change his policy during a day, but is allowed to change the policy between days).

Our proof is similar to that of the stochastic proportional model in [7]. There, the only random variable is the number of daily queries $|s|$, and the number of occurrences of the keyword w_i is $\lambda_i \cdot |s|$, and $\overrightarrow{\lambda}$ is a known vector which determines the fixed proportions of the keyword instances. Using an interchange argument, the authors prove that the optimal solution for that case is a prefix solution, where for $j > i$ (words sorted by profit in a non increasing order) if there is $b_j > 0$ and $b_i < 1$ we can move some "weight" from a "bad" keyword to a "better" keyword. They calculate the appropriate amount of "weight" to move using the $\overrightarrow{\lambda}$ values (such that the total expected cost remains the same while the expected profit may only increase).

Our probabilistic model differs in the fact that the actual occurrences of the queries are distributed according to a multinomial distribution (with the unknown parameter vector $\overrightarrow{\lambda}$). Only the expectation of the actual number of occurrences of word w_i is $\lambda_i \cdot |s|$, and each of the keywords' number of occurrences has a binomial distribution $B(n, p)$ with parameters $n = |s|$ and $p = \lambda_i$ (different keywords are negatively correlated).

Claim 9 *For every stochastic problem instance $\Phi = (W, \overrightarrow{\pi}, \overrightarrow{c}, B_{day}, T, P, \overrightarrow{\lambda})$, there exists an optimal fractional prefix solution, such that $OPT(\Phi) = \Pi(\Phi, PRE(B_{day}, i, b))$, for some i and b.*

Proof: We show that if there are two keywords where the 'better' word is not fully bought while the 'worse' word is being bought, i.e., there are some w_i and w_j where $j > i$, $b_j > 0$ and $b_i < 1$ by moving 'weight' from b_j to b_i we can only improve the expected profit. In this case, for a given day with $|s|$ queries, both the number of occurrences of w_i and w_j have a binomial distribution. We further notice that by decreasing ε weight from a word w_j which is distributed as $B(|s|, \lambda_j)$, we do not buy $B(|s|, \varepsilon \cdot \lambda_j)$ number of words w_j. By adding $\varepsilon \cdot \frac{\lambda_j}{\lambda_i}$ to the weight of word w_i - a $B(S, \lambda_i \cdot \frac{\varepsilon \lambda_j}{\lambda_i})$ number of words w_i are bought instead, i.e., the two policies $(b_1 = 1, \ldots, b_{i-1} = 1, b_i < 1, \ldots, b_j > 0, \ldots)$ and $(b_1' = 1, \ldots, b_{i-1}' = 1, b_i' = b_i + \varepsilon \cdot \frac{\lambda_j}{\lambda_i}, \ldots, b_j' = b_j - \varepsilon, \ldots)$ have the same distribution of number of words being bought. We note that ε can be taken to be small enough as to keep b_i' and b_j' in the range $[0, 1]$. Therefore, we leave the cost distribution unchanged and can only improve expected profit by buying from a keyword with a higher (or equal) profit. This procedure can be used to convert any non prefix solution which is optimal for the series S into a prefix solution which has a profit at least as profitable as the non prefix solution. As by the model definition, the distribution of the query series s_t is constant for all days $t \in [1, T]$, exist a single fixed policy which is optimal for all days. ∎

Although we showed that for each non-prefix policy there exists a prefix policy which is at least as good, finding the best prefix solution, i.e., the optimal parameters i and b of $PRE(B_{day}, i, b)$ is not a trivial task. We first show that selecting a prefix that exhausts, in expectation, the daily budget is not optimal.

Claim 10 *A prefix solution $PRE(B_{day}, i, b)$ which buys a subset of keywords with an expected sum of costs which is equal to the daily budget B_{day} is not necessarily an optimal prefix, i.e., selecting a prefix $PRE(B_{day}, i, b)$ for which $E(|s| \cdot (\sum_{j=1}^{i-1} \lambda_j + b\lambda_i)) = B_{day}$, where the expectation is over P, does not guarantee an optimal profit.*

Proof: Consider the following example.

Keyword	w_1	w_2
Profit (π)	1000	2
Cost (c)	1	1
Probability (λ)	0.5	0.5

Assume the following distribution over the length of the daily query sequences: w.p 0.9 $|s| = B_{day}/10$, w.p 0.1 $|s| = 9.1 B_{day}$. It is clear that the prefix solution $PRE(B_{day}, 1, 1)$ has a better payoff than the prefix solution $PRE(B_{day}, 2, 1)$ although for the latter

$$E(|s| \cdot (\sum_{j=1}^{i-1} \lambda_j + b\lambda_i)) = 0.9 \cdot B_{day}/10 + 0.1 \cdot 9.1 B_{day} = B_{day}.$$

∎

We next show that the expected profit as a function of the prefix index is not necessarily a concave function and there might be multiple local maxima.

Claim 11 *For prefix policies, the Advertiser's expected profit $\Pi(\Phi, PRE(B_{day}, i, b))$ as a function of the prefix index i may have multiple local maxima.*

Proof: Consider the following example where $\alpha \in [\varepsilon, 1]$ and $\varepsilon \in [0, 1]$:

	w_1	w_2	w_3
Profit (π)	$2/\varepsilon$	2	2
Probability (λ)	ε^2	$\alpha - \varepsilon^2$	$1 - \alpha$

Let the distribution P over the length of daily sequences be: w.p. $1 - \varepsilon$, we have s_1 with $|s_1| = B_{day}$, and w.p. ε, we have s_2 with $|s_2| = \frac{B_{day}}{\varepsilon^2}$.

We use Chernoff's inequality $Pr[x < (1 - \delta)\mu] < e^{-\frac{\mu\delta^2}{2}}$ where $x = \sum x_i$ and $x_i \in [0, 1]$, with $e^{-\frac{\mu\delta^2}{2}} = \frac{1}{\sqrt{B_{day}}}$ implying $\delta = \sqrt{\frac{log(B_{day})}{B_{day}}}$, to show that in long days, w.p. greater than $1 - \sqrt{\frac{log(B_{day})}{B_{day}}}$ there are at least $B_{day}(1 - \frac{1}{\sqrt{B_{day}}})$ occurrences of w_1. Let X_1 denote the random variable counting how many times word w_1 appears in a day with $\frac{B_{day}}{\varepsilon^2}$ queries. Then

$$Pr[x_1 < (1 - \sqrt{\frac{log(B_{day})}{B_{day}}})\frac{B_{day}}{\varepsilon^2} \cdot \varepsilon^2] < e^{-\frac{\frac{B_{day}}{\varepsilon^2} \cdot \varepsilon^2 \cdot \frac{log B_{day}}{B_{day}}}{2}} = \frac{1}{\sqrt{B_{day}}}$$

.

The expected profit of the different prefix solutions are:

$$\Pi(\Phi, PRE(B_{day}, 1, 1)) \geq \varepsilon \cdot B_{day}(1 - \frac{1}{\sqrt{B_{day}}})(1 - \sqrt{\frac{log(B_{day})}{B_{day}}}) \cdot \frac{2}{\varepsilon} + (1 - \varepsilon) \cdot B_{day} \cdot \varepsilon^2 \cdot \frac{2}{\varepsilon}$$

$$\geq 2B_{day}(1 - \sqrt{\frac{log(B_{day})}{B_{day}}} - \frac{1}{\sqrt{B_{day}}})$$

$$= 2B_{day} - 2\sqrt{B_{day}} \cdot (\sqrt{log(B_{day})} + 1)$$

$$\geq 1.95 B_{day},$$

where the last inequality holds for $B_{day} \geq 2^{20}$. For the second prefix we have,

$$\Pi(\Phi, PRE(B_{day}, 2, 1)) \leq \varepsilon \cdot B_{day} \cdot [\frac{\varepsilon^2 \cdot \frac{2}{\varepsilon} + (\alpha - \varepsilon^2) \cdot 2}{\varepsilon^2 + (\alpha - \varepsilon^2)}] + (1 - \varepsilon) \cdot B_{day} \cdot [\varepsilon^2 \cdot \frac{2}{\varepsilon} + (\alpha - \varepsilon^2) \cdot 2]$$

$$\leq \varepsilon B_{day} \cdot [\frac{2\varepsilon + 2\alpha}{\alpha}] + B_{day} \cdot [2\alpha + 2\varepsilon]$$

$$\leq 2\alpha B_{day} + 6\varepsilon B_{day},$$

where the first term in the right side of the first inequality is due to the fact that the expected reward in a 'long' day is lower or equal than the expected reward of an infinite query series. For the third prefix we have,

$$\Pi(\Phi, PRE(B_{day}, 3, 1)) = B_{day}(\varepsilon^2 \cdot \frac{2}{\varepsilon} + (\alpha - \varepsilon^2) \cdot 2 + (1 - \alpha) \cdot 2)$$

$$= 2B_{day}(\varepsilon(1 - \varepsilon) + 1)$$

$$\geq 2B_{day}$$

For $\alpha = 0.5$ and $\varepsilon < 0.1$, we have $(6\varepsilon + 2\alpha)B_{day} < 1.6B_{day}$ and there are multiple local minima. ∎

Modeling the length of the daily sequences s_t is crucial. We show that an advertiser that does not know the length of the daily sequences (and can only try to estimate it) might earn a negligible fraction of the profit of a policy which does know beforehand the number of daily impressions in each of the days. Note that a policy that selects the best prefix at each given day is an optimal policy.

Claim 12 *Let $P-OPT$ denote the (oracle) policy that knows at the beginning of each day t (prior to selecting the daily policy) the length of the daily sequence $|s_t|$. There exists a problem instance Φ such that for any i, b, we have*

$$\Pi(\Phi, PRE(B_{day}, i, b)) \leq \frac{15}{m-1} \cdot \Pi(\Phi, P - OPT)$$

where m is a parameter and may be arbitrary large.

Proof: We define the problem instance Φ. Fix B_{day} and the parameter $m \geq 2$. Let $\overrightarrow{\pi} = \{m^{k-1}, m^{k-2}, \ldots, m^{k-i}, \ldots, m^0\}$, and let $\overrightarrow{\lambda} = \{\frac{Z}{m^{2k}}, \frac{Z}{m^{2k-2}}, \ldots, \frac{Z}{m^{2k-2i+2}}, \ldots, \frac{Z}{m^2}\}$, where Z is the normalizing constant, i.e., $Z = \frac{1}{\sum_{i=1}^{k} \frac{1}{m^{2k-2i+2}}}$.

The distribution P, which selects the number of queries in a day, supports k different day types:

Day type:	$|s|$	$P(|s|)$
s_1	$\frac{B_{day}}{\lambda_1} = O(\frac{B_{day}m^{2k}}{Z})$	$\frac{C}{\Pi(\Phi, P-OPT|s_1)}$
s_2	$\frac{B_{day}}{\lambda_1 + \lambda_2} = O(\frac{B_{day}m^{2k-2}}{Z})$	$\frac{C}{\Pi(\Phi, P-OPT|s_2)}$
$\vdots$	$\vdots$	$\vdots$
s_j	$\frac{B_{day}}{\sum_{i=1}^{j} \lambda_i} = O(\frac{B_{day}m^{2k-2j+2}}{Z})$	$\frac{C}{\Pi(\Phi, P-OPT|s_j)}$
$\vdots$	$\vdots$	$\vdots$
s_k	$\frac{B_{day}}{\sum_{i=1}^{k} \lambda_i} = O(\frac{B_{day}m^2}{Z})$	$\frac{C}{\Pi(\Phi, P-OPT|s_k)}$

where $\Pi(\Phi, P - OPT|s_j)$ is the expected profit of $P - OPT$ given that the day type is day s_j and C is the normalizing constant, i.e., $\frac{1}{\sum_{j=1}^{k} \frac{1}{\Pi(\Phi, P-OPT|s_j)}}$.

We notice that in our setting, the expected number of keywords that an unlimited budget prefix account $PRE(\infty, j, 1)$ will buy in a day of type s_j is constant for all $j \in [1, k]$ and is equal to B_{day}.

We first bound the profit of $P - OPT$ by noticing that for a day of type s_j, $P - OPT$ which knows the length of the daily sequence is $|s_j| = O(\frac{B_{day}m^{2k-2j+2}}{Z})$ chooses the prefix solution that is optimal for that day (according to Claim 9 there exists an optimal prefix solution). The expected profit of that optimal prefix is at least the expected profit of $PRE(B_{day}, j, 1)$. Using Chernoff's inequality $Pr[x < (1 - \delta)\mu] < e^{-\frac{\mu\delta^2}{2}}$ where $x = \sum x_i$ and $x_i \in [0, 1]$, with $\delta = \sqrt{\frac{2logm}{B_{day}}}$, we can bound the number of keywords from the set $\{w_1, w_2, \ldots, w_j\}$ that appear in a day of type s_j. W.p. greater than $1 - \frac{1}{m}$ there are at least $B_{day}(1 - \sqrt{\frac{2logm}{B_{day}}})$ queries from that set. Therefore, we get that the expected profit of

$P - OPT$ in a day of type s_j is:

$$\Pi(\Phi, P - OPT|s_j) \geq \Pi(\Phi, PRE(B_{day}, j, 1)|s_j)$$

$$\geq (1 - \frac{1}{m})(1 - \sqrt{\frac{2logm}{B_{day}}})[\frac{B_{day}}{\sum\limits_{i=1}^{j} \lambda_i} \cdot \sum\limits_{i=1}^{j} \lambda_i]\frac{\sum\limits_{i=1}^{j} \lambda_i \cdot \pi_i}{\sum\limits_{i=1}^{j} \lambda_i}.$$

We next analyze the expected profit of a fixed prefix policy, and show that for any such fixed prefix, i.e., $\forall j, b, \quad PRE(B_{day}, j, b)$, the total expected profit is at most $\frac{15}{m-1}$ of the expected profit of $P - OPT$.

Given a selection of j and b we divide the day types $\{s_1, \ldots, s_k\}$ into three sets:

1. For $s_{j'}$ such that $j' \in \{j-1, j, j+1\}$, the prefix algorithm $PRE(B_{day}, j, b)$ obviously gets no more than $P - OPT$. This happens at most, at three day types and by our choice of the distribution $P(|s_j|)$ which divides the expected profit uniformly over the day types, we are guaranteed that this accounts to no more than $\frac{3}{k} \cdot \Pi(\Phi, P - OPT)$.

2. For $s_{j'}$ such that $j' \geq j + 2$: we notice that the expected profit of a prefix solution $PRE(B_{day}, j, b)$ at a day type $s_{j'}$ is bounded from above by the expected profit of the same prefix solution in an infinite long day. Also, due to the fact that the profits $(\pi_i)_{i=1}^{k}$ are sorted, the expected profit of $PRE(B_{day}, j, b)$ from a single query as a function of j and b is monotonically decreasing, i.e., if $j_1 < j_2$ (or $j_1 = j_2$ and

$$b_1 < b_2), \quad \frac{\sum\limits_{i=1}^{j_1-1} \lambda_i\pi_i + b_1 \cdot \lambda_{j_1}\pi_{j_1}}{\sum\limits_{i=1}^{j_1-1} \lambda_i + b_1 \cdot \lambda_{j_1}} \geq \frac{\sum\limits_{i=1}^{j_2-1} \lambda_i\pi_i + b_2 \cdot \lambda_{j_2}\pi_{j_2}}{\sum\limits_{i=1}^{j_2-1} \lambda_i + b_2 \cdot \lambda_{j_2}}.$$

Therefore, for $j' \geq j + 2$ and any b,

$$\frac{\Pi(\Phi, PRE(B_{day}, j', b)|s_j)}{\Pi(\Phi, P - OPT|s_j)} \leq \frac{B_{day} \cdot \frac{\sum\limits_{i=1}^{j+1} \lambda_i\pi_i}{\sum\limits_{i=1}^{j+1} \lambda_i}}{(1 - \frac{1}{m})B_{day}(1 - \sqrt{\frac{2logm}{B_{day}}})\frac{\sum\limits_{i=1}^{j} \lambda_i \cdot \pi_i}{\sum\limits_{i=1}^{j} \lambda_i}}$$

$$\leq \frac{m}{m-1} \cdot \frac{1}{1 - \sqrt{\frac{2logm}{B_{day}}}} \cdot \frac{2}{m}$$

$$= \frac{2}{m-1} \cdot \frac{1}{1 - \sqrt{\frac{2logm}{B_{day}}}}$$

where we used $\frac{\sum\limits_{i=1}^{j+1} \lambda_i\pi_i}{\sum\limits_{i=1}^{j+1} \lambda_i} \leq \frac{\sum\limits_{i=1}^{j} \lambda_i \cdot \pi_i}{\sum\limits_{i=1}^{j} \lambda_i} \cdot \frac{2}{m}$ in the last inequality.

3. For $s_{j'}$ such that $j' \leq j - 2$: the expected profit of a prefix solution $PRE(B_{day}, j, b)$ at a day type s'_j is bounded from above by the expected profit of the same prefix solution with infinite daily budget. Again, using the monotonicity of the expected profit $PRE(B_{day}, j, b)$ from a single query as a function of j and b we get that for any $j' \leq j - 2$ and any b,

$$\frac{\Pi(\Phi, PRE(B_{day}, j', b)|s_j)}{\Pi(\Phi, P-OPT|s_j)} \leq \frac{[\frac{B_{day}}{\sum_{i=1}^{j} \lambda_i} \cdot \sum_{i=1}^{j-2} \lambda_i] \frac{\sum_{i=1}^{j-3} \lambda_i \cdot \pi_i}{\sum_{i=1}^{j-3} \lambda_i}}{(1 - \frac{1}{m})(1 - \sqrt{\frac{2logm}{B_{day}}})[\frac{B_{day}}{\sum_{i=1}^{j} \lambda_i} \cdot \sum_{i=1}^{j} \lambda_i] \frac{\sum_{i=1}^{j} \lambda_i \cdot \pi_i}{\sum_{i=1}^{j} \lambda_i}}$$

$$= \frac{\sum_{i=1}^{j-2} \lambda_i \frac{\sum_{i=1}^{j-3} \lambda_i \cdot \pi_i}{\sum_{i=1}^{j-3} \lambda_i}}{(1 - \frac{1}{m})(1 - \sqrt{\frac{2logm}{B_{day}}}) \sum_{i=1}^{j} \lambda_i \cdot \pi_i}$$

$$= \frac{m}{m-1} \cdot \frac{1}{1 - \sqrt{\frac{2logm}{B_{day}}}} \cdot \frac{\sum_{i=1}^{j-2} \lambda_i}{\sum_{i=1}^{j-3} \lambda_i} \cdot \frac{\sum_{i=1}^{j-3} \lambda_i \cdot \pi_i}{\sum_{i=1}^{j} \lambda_i \cdot \pi_i}$$

$$\leq \frac{m}{m-1} \cdot \frac{1}{1 - \sqrt{\frac{2logm}{B_{day}}}} \cdot 2m^2 \cdot \frac{2}{m^3}$$

$$= \frac{4}{m-1} \cdot \frac{1}{1 - \sqrt{\frac{2logm}{B_{day}}}}$$

By selecting $B_{day} > 8logm$ we have that $\frac{1}{1 - \sqrt{\frac{2logm}{B_{day}}}} < 2$, and by setting $k = m$ we conclude our proof. ∎

In the proof of Claim 12, we have used a problem setting in which the number of keywords required $k > 16$ for obtaining a non-trivial bound. In the next claim we show that even for small keyword sets, an oracle policy $P-OPT$ performs much better compared to prefix policies.

Claim 13 *There exist a stochastic problem instance Φ in which*

$$\Pi(\Phi, PRE(B_{day}, i, b)) \leq 0.67 \cdot \Pi(\Phi, P - OPT)$$

and in which $k = 2$.

Proof: Consider the following example:

	w_1	w_2
Cost	1	1
Profit	$2/\alpha$	2
Prob	ε	$1 - \varepsilon$

where $\alpha \in [0, 1)$, and assume the following distribution over the number of daily words:

w.p. $1 - p$, we have $|s| = B_{day}$, and w.p. p, we have $|s| = \frac{B_{day}}{\varepsilon}$.

We use Chernoff's inequality $Pr[x < (1 - \delta)\mu] < e^{-\frac{\mu\delta^2}{2}}$ where $x = \sum x_i$ and $x_i \in [0, 1]$, with $e^{-\frac{\mu\delta^2}{2}} = \frac{1}{\sqrt{B_{day}}}$ implying $\delta = \sqrt{\frac{log(B_{day})}{B_{day}}}$. This implies that in long days, w.p. greater than $1 - \sqrt{\frac{log(B_{day})}{B_{day}}}$, there are at least $B_{day}(1 - \frac{1}{\sqrt{B_{day}}})$ occurrences of w_1.

P-OPT chooses $PRE(B_{day}, 1, 1)$ for long days and $PRE(B_{day}, 2, 1)$ for short days and it has expected profit:

$$\Pi(\Phi, P - OPT) \geq p \cdot \frac{2}{\alpha} \cdot \frac{B_{day}}{\varepsilon} \cdot \varepsilon \cdot (1 - \sqrt{\frac{log(B_{day})}{B_{day}}}) \cdot (1 - \frac{1}{\sqrt{B_{day}}}) + (1 - p)(\frac{\varepsilon}{\alpha} + (1 - \varepsilon)) \cdot 2B_{day}$$

The expected profits of the two different prefix policies are:

$$\Pi(\Phi, PRE(B_{day}, 1, 1)) \leq p \cdot \frac{2B_{day}}{\alpha} + (1 - p)\varepsilon \cdot \frac{2B_{day}}{\alpha}$$

$$\Pi(\Phi, PRE(B_{day}, 2, 1)) = 2B_{day} \cdot (\frac{\varepsilon}{\alpha} + (1 - \varepsilon))$$

For fractional prefixes, we have for $b \in (0, 1]$:

$$\Pi(\Phi, PRE(B_{day}, 2, b)) \leq 2B_{day} \cdot [(1 - p)(\frac{\varepsilon}{\alpha} + (1 - \varepsilon)b) + p(\frac{\varepsilon}{\varepsilon + (1 - \varepsilon)b} \cdot \frac{1}{\alpha} + \frac{(1 - \varepsilon)b}{\varepsilon + (1 - \varepsilon)b})]$$

Selecting $\alpha = 0.5$, $p = 0.5$, $\varepsilon = 0.0005$ and $B_{day} > 10^8$ imply that:

$$\Pi(\Phi, P - OPT) \geq 2B_{day}(1 - \sqrt{\frac{log B_{day}}{B_{day}}})(1 - \frac{1}{\sqrt{B_{day}}}) + B \geq 2.999 B_{day}.$$

For the fractional prefixes, we notice that for $b \in [0, 1]$, $\Pi(\Phi, PRE(B_{day}, 1, b)) \leq \Pi(\Phi, PRE(B_{day}, 1, 1))$.

Also, noticing that the function $f(b) = \frac{(1 - \varepsilon)b}{2} + \frac{\varepsilon + (1 - \varepsilon)\frac{b}{2}}{\varepsilon + (1 - \varepsilon)b}$ has a single minimum in the range $b \in (0, 1]$, implies that $\Pi(\Phi, PRE(B_{day}, 2, b)) \leq \Pi(\Phi, PRE(B_{day}, 2, 1))$, i.e., for all i, b, $\Pi(\Phi, PRE(B_{day}, i, b)) \leq 2(1 + \varepsilon)B_{day} \leq 2.001 B_{day}$.

3.3 Solving the Online problem

Since the distribution $\vec{\lambda}$ over the keywords in W, and the distribution P over the length of the daily sequences s_i are unknown to the advertiser, an online learning approach is needed. In [5] the authors discuss the exploration-exploitation dilemma in the context of the multi-arm bandit problem. They prove that simple algorithms, such as the $UCB1$ algorithm [5], asymptotically achieve a logarithmic regret in the number of plays, a regret which has been showed by [14] to be the best possible for such problems.

We show how to apply the $UCB1$ algorithm to our problem. In order to cope with the infinite number of policies available with the continuous prefix policies (as the probability b_i of bidding on the keyword w_i can be set to any value in the range $(0,1]$) we limit ourself to prefix accounts where b_i is from the discrete set $\{\varepsilon, 2\varepsilon, \ldots, 1\}$ which we denote $PRE(B_{day}, i, b_\varepsilon)$. The following lemma bounds the loss due to the discretization.

Lemma 14 *For any problem instance Φ there exists an index i^* and an integer c such that $\Pi(\Phi, OPT) - \Pi(\Phi, PRE(B_{day}, i^*, c \cdot \varepsilon)) \leq \varepsilon \cdot B_{day} \pi_{i^*} T$, where T is the number of sequences in the series S (the number of days).*

Proof: By Claim 9, there exists an optimal prefix policy $PRE(B_{day}, i, b)$ with parameters $i = i^*$ and $b = b^*$. Therefore we can find a policy $PRE(B_{day}, i, b_\varepsilon)$ with the same $i = i^*$ and which has a bidding probability b' where $b^* - \varepsilon \leq b' \leq b^*$. Such a policy will buy the same keywords bought by $\Pi(\Phi, OPT))$ as long as $\Pi(\Phi, OPT))$ has not exhausted the daily budget, except at most $\varepsilon \cdot B_{day}$ keywords of type w_{i^*} (Note that if the daily budget $\Pi(\Phi, OPT))$ is exhausted, then our policy $PRE(B_{day}, i^*, b')$ can only decrease the difference by buying additional keywords from the set $\{w_1, \ldots, w_{i-1}\}$). ∎

For multi-arm bandits, the policy UCB1 [5], achieves logarithmic regret without any preliminary knowledge about the reward distribution. Let $UCB1_{Pre}$ denote a daily changing $PRE(B_{day}, i, b_\varepsilon)$ policy, selected from the set of $\frac{k}{\varepsilon}$ prefix policies with $i \in \{1, \ldots, k\}$ and $b \in \{\varepsilon, 2\varepsilon, \ldots, 1\}$, according to the $UCB1$ algorithm. I.e., the 'actions' from which the UCB1 policy selects are (i, b) where $i \in \{1, \ldots, k\}$ and $b \in \{\varepsilon, 2\varepsilon, \ldots, 1\}$.

The next claim bounds the loss of such a UCB1 policy compared to the optimal prefix policy:

Claim 15 *For any problem instance Φ we have*

$$
\begin{aligned}
\Pi(\Phi, OPT) - \Pi(\Phi, UCB1_{Pre}) &\leq \varepsilon B_{day} \pi_1 (T+1) + \frac{k}{\varepsilon} \left(\frac{8ln(T)}{\varepsilon} + 5 \cdot \pi_1 \cdot B_{day} \right) \\
&= O\left(\varepsilon \pi_1 B_{day} T + \frac{k ln(T)}{\varepsilon^2} + \frac{k \pi_1 \cdot B_{day}}{\varepsilon} \right)
\end{aligned}
$$

where T is the number of sequences in the series S.

Proof: As shown in [5], for a bandit problem with k actions, where action i has an expected profit of μ_i, the UCB1 algorithm guarantees

$$
\Pi(\Phi, OPT)) - \Pi(\Phi, UCB1) = \sum_i E((\mu_{i^*} - \mu_i) \cdot N_i(t)))
$$

where $N_i(t)$ is the number of times arm i has been chosen during the first t plays, and i^* is the arm with the maximal expected profit. In [5] it was proved that

$$E(N_i(t)) \leq \frac{8ln(t)}{(\mu_{i^*} - \mu_i)^2} + 1 + \frac{\pi^2}{3} \leq \frac{8ln(t)}{(\mu_{i^*} - \mu_i)^2} + 5$$

.

Therefore:

$$
\begin{aligned}
\Pi(\Phi, OPT) - \Pi(\Phi, UCB1_{Pre}) &\leq \varepsilon B_{day}\pi_{i^*}T + \sum_{i,b}(\mu_{i^*,b^*} - \mu_{i,b})(\frac{8ln(T)}{(\mu_{i^*,b^*} - \mu_{i,b})^2} + 5) \\
&\leq \varepsilon B_{day}\pi_{i^*}T + \varepsilon T + \sum_{i,b|\mu_{i^*,b^*} - \mu_{i,b} \geq \varepsilon}(\mu_{i^*,b^*} - \mu_{i,b})(\frac{8ln(T)}{(\mu_{i^*,b^*} - \mu_{i,b})^2} + 5),
\end{aligned}
$$

where we used Lemma 14 in the first inequality and where the last term comes from the fact that $\sum_{i,b|\mu_{i^*,b^*} - \mu_{i,b} < \varepsilon}(\mu_{i^*,b^*} - \mu_{i,b}) \cdot N_i(T) \leq \varepsilon \cdot T$.

In our case, $\mu_{i^*,b^*} - \mu_{i,b} \leq B_{day} \cdot \pi_1$ and we get that:

$\Pi(\Phi, OPT) - \Pi(\Phi, UCB1) \leq \varepsilon T(\pi_1 \cdot B_{day} + 1) + \frac{k}{\varepsilon}(\frac{8ln(T)}{\varepsilon} + 5 \cdot \pi_1 \cdot B_{day}))$. ∎

Corollary 1 *For any problem instance* Φ: $\Pi(\Phi, OPT) - \Pi(\Phi, UCB1) = O(k^{\frac{1}{2}}B_{day}T^{\frac{2}{3}}lnT)$.

Proof: Setting $\varepsilon = k^{\frac{1}{2}}T^{-\frac{1}{3}}$ gives us

$$
\begin{aligned}
\Pi(\Phi, OPT) - \Pi(\Phi, UCB1) &\leq T^{\frac{2}{3}}(k^{\frac{1}{2}} \cdot \pi_1 \cdot B_{day} + 1 + 8ln(T)) + 5T^{\frac{1}{3}}k^{\frac{1}{2}}\pi_1 B_{day} \\
&= O(k^{\frac{1}{2}}B_{day}T^{\frac{2}{3}}lnT),
\end{aligned}
$$

which derives the bound. ∎

3.4   Bucket policies

We are especially interested in regret bounds which are not dependent on the number of keywords k in the problem, as we are usually dealing with very large number of keywords. To do that, we group keywords with similar profit-to-cost ("bang-to-buck") ratios together in the following way. We partition the range $[\pi_k, \pi_1]$ into multiplicative buckets, guaranteing a maximal ratio of $1 + \varepsilon$ between two different keywords sharing a bucket. Let $\beta = \frac{\pi_1}{\pi_k}$, we create $\frac{log\beta}{\varepsilon}$ buckets: $(1 + \varepsilon)^x = \beta$, and $xlog(1 + \varepsilon) = log\beta$ as $log(1 + \varepsilon) \leq \varepsilon$. The keywords $w_1, w_2, \ldots, w_k$ (which are sorted by their profit to cost ratios) are therefore grouped into buckets A_i, where $i \in \{1, \ldots, \frac{log\beta}{\varepsilon}\}$. Similar to the prefix accounts we defined earlier, we define bucket prefix accounts with daily budget B_{day}, which buy all words from buckets $A_1, \ldots, A_{i-1}$ and a fraction b_ε of the words in bucket A_i as $BUC(B_{day}, i, b_\varepsilon)$. The following lemma shows that the discretization does not lose much.

Lemma 16 *For any problem instance Φ there exist integers i' and c such that*

$$\Pi(\Phi, OPT) - \Pi(\Phi, BUC(B_{day}, i', c \cdot \varepsilon)) \leq 2\varepsilon \cdot B_{day} \pi_{i'} T.$$

Proof: By Claim 9, there exists an optimal prefix policy $PRE(B_{day}, i, b)$ with parameters $i = i^*$ and $b = b^*$. We can therefore find a policy $BUC(B_{day}, i, b_\varepsilon)$ with the following parameters:

1. $i = i'$ such that i^* is in $A_{i'}$.

2. a bidding probability b_ε such that the cost of queries being bought from the bucket $A_{i'}$ is less than the cost of the queries being bought by OPT from A_i, with a cost difference of less than εB_{day}.

For any of the queries in bucket A_i, $i < i'$ the bucket prefix earns exactly the same profits earned by OPT. For the queries in bucket $A_{i'}$, the bucket prefix earns no less the $\frac{1}{1+\varepsilon}$ of the profit earned by OPT per query (due to the multiplicative design of the bucket) and by the selection of b_ε that profit is gained over no less then $1 - \varepsilon$ of the budget spent by OPT on that bucket's keywords. ∎

Let $UCB1_{Buc}$ denote a daily changing $BUC(B_{day}, i, b_\varepsilon)$ policy, where $i \in \{1, \ldots, \frac{log\beta}{\varepsilon}\}$, and $b \in \{0, \varepsilon, 2\varepsilon, \ldots, 1 - \varepsilon\}$. Algorithm $UCB1_{Buc}$ uses the UCB1 algorithm to select a policy from the set of $\frac{log\beta}{\varepsilon^2}$ bucket prefix policies $BUC(B_{day}, i, b_\varepsilon)$.

Corollary 2 *For any problem instance Φ: $\Pi(\Phi, OPT) - \Pi(\Phi, UCB1_{Buc}) = O(B_{day} T^{\frac{3}{4}} lnT)$.*

Proof: By using Claim 15, Lemma 16, replacing the number of actions $\frac{k}{\varepsilon}$ with $k = \frac{log\beta}{\varepsilon^2}$ and by setting $\varepsilon = T^{-\frac{1}{4}}$ we have:

$$
\begin{aligned}
\Pi(\Phi, OPT) - \Pi(\Phi, UCB1_{Buc}) &\leq 2\varepsilon B \pi_1 (T + 1) + \frac{log\beta}{\varepsilon^2}(\frac{8 lnT}{\varepsilon} + 5\pi_1 B_{day}) \\
&\leq 2 B_{day} T^{\frac{3}{4}} \pi_1 (B_{day} + \frac{1}{T}) + 8 log\beta T^{\frac{3}{4}} lnT(1 + \frac{5}{8} B_{day} \pi_1 T^{-\frac{1}{2}}) \\
&= O(log\beta \pi_1 B_{day} T^{\frac{3}{4}} lnT)
\end{aligned}
$$

∎

The special structure of our problem, namely, the fact that different keywords can be joined together based on their profit to cost ratio, allow Corollary 2 to derive an upper bound $O(B_{day} T^{\frac{3}{4}} lnT)$, which is not dependant on the number of keywords k. This is not the case in the general stochastic multi-arm bandit. There the regret from the selection of each non optimal arm i is $min\{E(N_i(t)) \cdot (\mu_{i^*} - \mu_i), T \cdot (\mu_{i^*} - \mu_i)\}$. Consider a setting in which all actions except the optimal one have, $\mu_{i^*} - \mu_i = \frac{1}{\sqrt{T}}$. As we showed in the proof of Claim 15, the per action regret is $min\{\frac{log(T)}{\mu_{i^*} - \mu_i}, T \cdot (\mu_{i^*} - \mu_i)\} = O(\sqrt{T})$. For such a setting, we get a regret bound of $O(k\sqrt{T})$. In the keywords optimization problem, the special structure of the problem allows us to reduce the dimensionality of the action space by joining multiple similar actions together. That allows us to derive a bound independent of the number of keywords k. That holds even for exponential action spaces such as in the case where $k = 2^T$. In that case, the classical k-arm bandit upper bound is trivial.

3.5 Stochastic Model Lower Bound

We conclude the analysis of the stochastic model by presenting a lower bound for the regret of algorithms which choose (daily) between different prefix policies when compared to a fixed prefix policy. Our proof to the lower bound is based on the proof used for the lower bound in the adversarial model presented in the next chapter. For the simplicity of presentation, and as the proof relies on some of the results presented there, we provide the proof in appendix A.

Theorem 17 *For any given T, $k < T^{1/3}$ there exists a problem instance Φ in which for any algorithm A over prefix policies, $\Pi(\Phi, OPT) - \Pi(\Phi, A) = \Omega(\sqrt{Tk})$. For $k > T^{1/3}$, $\Pi(\Phi, OPT) - \Pi(\Phi, A) = \Omega(T^{2/3})$.*

Chapter 4

The adversarial Advertiser's Keyword Optimization problem

The previous chapter assumed a simple and stationary stochastic model underlying the statistics of generated user query lists. In practice, these assumptions may fail to adequately model the real life variability of users' interest over time.

Therefore we introduce an adversarial variation of the problem, in which the advertiser faces an adversary who may create arbitrary generated sequences of queries. No statistical assumptions are made about the generation of the sequences and we only assume that these sequences are generated at the beginning of each day before the advertiser selects his keyword set, i.e., the adversary is oblivious to the algorithm's actions. The advertiser task is the same as in the stochastic version of the problem, i.e., to select for each day a good fixed set of keywords to buy, while maximizing his profit under the daily budget constraint.

We show, that under some assumptions about the length of the sequence, the loss of an advertiser in the adversarial setting compared to the best fixed policy, may be as big as $\Omega(\sqrt{Tk})$, where T is the number of sequences (days) and $k = O(T^{1/3})$ is the number of keywords in the problem instance. We show that this bound is tight (up to a logarithmic factor). Our lower bound does not hold for the case where $k = \Omega(T^{1/3})$. In that case, we provide a weaker lower bound of $\Omega(T^{2/3})$, which is independent of k.

Our proof is similar to the lower bound proof of Auer et al [6]. There, the authors introduce the adversarial multi-arm bandit problem, a variant of the classical bandit problem, in which no statistical assumptions are made about the generation of awards, but rather, each slot machine is initially assigned with an arbitrary and unknown sequence of rewards by an adversarial opponent. They prove a lower bound of $\sqrt{Tk}$ for any time horizon T and for any number of actions $k \geq 2$. Ironically, the power of the adversary in their lower bound comes not from adapting to the on-line algorithm A, but from adapting to the number of trials T.

We note that to prove the lower bound in our adversarial model, we can't use the lower bound result for adversarial multi-arm bandits in [6] directly. In their problem setting, the adversary directly sets the reward of each of the actions, whereas in our model, the adversary generates arbitrary sets of daily query sequences s_t. Therefore in the multi-arm bandit model, the adversary is more powerful: he could specify the reward for each action independently, whereas in our model, the adversary can only select the daily query sequence which determines the profits for all actions simultaneously. I.e., there is a dependency between the profits gained by selecting different sets of keywords given a daily query sequence. This implies that the adversary can not generate arbitrary profits (rewards for the different "arms") as in the bandit problem.

We conclude this discussion by noting an important difference between the stochastic and adversarial models. In the stochastic model, we clustered 'similar' words together into buckets, and by that reduced the action space with a small controllable cost. In the adversarial model, such clustering is not possible. For example, consider a problem instance in which there are only three words w_1, w_2 and w_3. Let the profits be such that $\pi_1 > \pi_2, \pi_2 = \pi_3$. In contrast to the stochastic model where trading w_2 and w_3 is possible (while taking care of their prior distribution $\overrightarrow{\lambda}$), in the adversarial model, the adversary can easily create daily sequences in which advertisers who buy w_2 hurt their overall profit while buying w_3 improves the overall profit (by e.g., creating sequences in which w_2 appear before w_1 and other sequences in which w_3 appear only after w_1). This is a key difference between the stochastic and the adversarial models.

4.1 Model definition

In the adversarial model, we attempt to capture the full generality of an arbitrary joint probability distribution. For that, we adopt a model similar to the scenario model in [7]. Instead of using P and λ to represent the distribution of queries from the keywords set W, the adversary may create an arbitrary set of daily sequences $\{s_0, s_1, \ldots, s_k\}$ and select an arbitrary distribution P over these sequences.

The advertiser is allowed to adaptively change the daily policy, i.e., for each day $t \in [1, T]$ the advertiser selects a daily policy $\overrightarrow{X^t}$, where $\overrightarrow{X^t} = (b_1, b_2, \ldots, b_k)$ defines a vector of probabilities for buying instances of keywords in W. As in the stochastic model, $\overrightarrow{X^t} = f(W, \overrightarrow{\pi}, \overrightarrow{c}, B_{day}, (r_t)_{t=1}^{t-1})$, where r_t is the daily profit of day t. I.e., the model assumes the advertiser knows the associated costs $\overrightarrow{c}$ and profits $\overrightarrow{\pi}$ of the keywords in W, but does not know the structure of the daily sequences s_t and the distribution P by which over them. Again, at the end of each day, the only information the advertiser sees is the profit scalar r_t which is the total profit for that day.

Formally, the advertiser's online adversarial keyword optimization problem is the following: given a stochastic problem instance $\Phi = (W, \overrightarrow{\pi}, \overrightarrow{c}, B_{day}, T, (s_t)_{t=1}^{T}, P)$ where $(s_t)_{t=1}^{T}$ and P are unknown to the advertiser, for each day $t \in [1, T]$ the advertiser has to select a daily policy $\overrightarrow{X^t}$ after which he receives a daily profit r_t. The advertiser's goal is to maximize his overall profit, i.e., $\sum_{t=1}^{T} r_t$. As before, we will denote the expected total profit of the policy $(\overrightarrow{X^t})_{t=1}^{T}$ in the T days of the problem instance Φ as $\Pi(\Phi, (\overrightarrow{X^t})_{t=1}^{T})$, where the expectation is over the distribution P.

We define $OPT(\Phi)$ as the maximal expected profit over all fixed policies $\overrightarrow{X}$, i.e., $OPT(\Phi) = \max_{\overrightarrow{X}} \Pi(\Phi, (\overrightarrow{X}, \ldots, \overrightarrow{X}))$.

Let $\eta_t = i$ denote the selection of the (non fractional) prefix account $PRE(B_{day}, i, 1)$ at time t. I.e, $b = 1$.

4.2   Adversarial problem setting

To derive our lower bound we define the following problem instance Φ_{WC}:

The daily sequences are selected by the adversary to be -

$$s_0 \equiv \underbrace{w_k \ldots w_k}_{m \ times}, \underbrace{w_{k-1} \ldots w_{k-1}}_{m \ times}, \ldots \underbrace{w_2 \ldots w_2}_{m \ times}, \underbrace{w_1 \ldots w_1}_{m \ times}, \underbrace{w_0 \ldots w_0}_{m \ times}$$

$$s_1 \equiv \underbrace{w_k \ldots w_k}_{m \ times}, \underbrace{w_{k-1} \ldots w_{k-1}}_{m \ times}, \ldots \underbrace{w_2 \ldots w_2}_{m \ times}, \underbrace{w_1 \ldots w_1}_{m \ times}$$

$$s_2 \equiv \underbrace{w_k \ldots w_k}_{m \ times}, \underbrace{w_{k-1} \ldots w_{k-1}}_{m \ times}, \ldots \underbrace{w_2 \ldots w_2}_{m \ times}$$

$$\ldots$$

$$s_{k-1} \equiv \underbrace{w_k \ldots w_k}_{m \ times}, \underbrace{w_{k-1} \ldots w_{k-1}}_{m \ times}$$

$$s_k \equiv \underbrace{w_k \ldots w_k}_{m \ times}$$

We denote by $P_{base} = (p_0, p_1, .., p_k)$ the following distribution over daily sequences $\{s_0, s_1, .., s_k\}$ above:

$$P_{base}(\{sequence = s_0\} \cup \{sequence = s_1\} \cup \ldots \cup \{sequence = s_i\}) = \frac{1/2}{1-i\varepsilon}.$$

Explicitly: $p_0 = \frac{1}{2}$ and $\forall i \in \{1, \ldots, K\}$,

$$p_i = \frac{1/2}{1-i\varepsilon} - \frac{1/2}{1-(i-1)\varepsilon} = \frac{1}{2} \cdot \frac{\varepsilon}{(1-i\varepsilon)(1-(i-1)\varepsilon)}.$$

In order to have a proper distribution that sums to one, we define a null series, $s_\perp$ with probability $1 - \frac{1/2}{1-k\varepsilon} = \frac{1/2-k\varepsilon}{1-k\varepsilon}$ and require that $\varepsilon \leq 1/2k$. We denote by P_j the distribution we get by removing an $\frac{p_{j+1}}{10}$ fraction from p_{j+1} and moving it to p_j (P_j is a perturbed version of P_{base}) for $j \in \{0, 1, \ldots, K-1\}$.

We define the profits vector $\overrightarrow{\pi}$ as following:

$$\pi_0 = 1, \pi_1 = 1 - \varepsilon, \pi_2 = 1 - 2\varepsilon, .., \pi_j = 1 - j\varepsilon, .., \pi_k = 1 - k\varepsilon$$

We keep all costs equal: $\forall i, \quad c_i = 1$ and the daily budget $B_{day} = m$.

4.3   Analysis of the daily profits

The following table gives the advertiser profit for each daily sequence s_i. We add also the probability of s_i under P_{base} and P_j.

s	P_{base}	$P_j, j \in [1, k-1]$	$\eta_t = 0$	$\eta_t = 1$	$\eta_t = 2$	$\ldots$	$\eta_t = j$	$\ldots$	$\eta_t = k-1$	$\eta_t = k$
s_0	p_0	p_0	$m \cdot \pi_0$	$m \cdot \pi_1$	$m \cdot \pi_2$	$\ldots$	$m \cdot \pi_j$	$\ldots$	$m \cdot \pi_{k-1}$	$m \cdot \pi_k$
s_1	p_1	p_1	0	$m \cdot \pi_1$	$m \cdot \pi_2$	$\ldots$	$m \cdot \pi_j$	$\ldots$	$m \cdot \pi_{k-1}$	$m \cdot \pi_k$
s_2	p_2	p_2	0	0	$m \cdot \pi_2$	$\ldots$	$m \cdot \pi_j$	$\ldots$	$m \cdot \pi_{k-1}$	$m \cdot \pi_k$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
s_{j-1}	p_{j-1}	p_{j-1}	0	0	0	$\ldots$	$m \cdot \pi_j$	$\ldots$	$m \cdot \pi_{k-1}$	$m \cdot \pi_k$
s_j	p_j	$p_j + \frac{1}{10}p_{j+1}$	0	0	0	$\ldots$	$m \cdot \pi_j$	$\ldots$	$m \cdot \pi_{k-1}$	$m \cdot \pi_k$
s_{j+1}	p_{j+1}	$\frac{9}{10}p_{j+1}$	0	0	0	$\ldots$	0	$\ldots$	$m \cdot \pi_{k-1}$	$m \cdot \pi_k$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
s_{k-1}	p_{k-1}	p_{k-1}	0	0	0	$\ldots$	0	$\ldots$	$m \cdot \pi_{k-1}$	$m \cdot \pi_k$
s_k	p_k	p_k	0	0	0	$\ldots$	0	$\ldots$	0	$m \cdot \pi_k$

The following table gives the distribution over the different types of earned keywords, under the base distribution P_{base}'s for each of the different prefix policies:

P_{base}:

$profit$	$\eta_t = 0$	$\eta_t = 1$	$\ldots$	$\eta_t = j$	$\ldots$	$\eta_t = k-1$	$\eta_t = k$
0	$1 - p_0$	$1 - p_0 - p_1$	$\ldots$	$1 - \sum_{i=0}^{j} p_i$	$\ldots$	$1 - \sum_{i=0}^{k-1} p_i$	$1 - \sum_{i=0}^{k} p_i$
$m \cdot \pi_0$	p_0	0	$\ldots$	0	$\ldots$	0	0
$m \cdot \pi_1$	0	$p_0 + p_1$	$\ldots$	0	$\ldots$	0	0
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$m \cdot \pi_j$	0	0	$\ldots$	$\sum_{i=0}^{j} \mathbf{p_i}$	$\ldots$	0	0
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$m \cdot \pi_{k-1}$	0	0	$\ldots$	0	$\ldots$	$\sum_{i=0}^{k-1} p_i$	0
$m \cdot \pi_k$	0	0	$\ldots$	0	$\ldots$	0	$\sum_{i=0}^{k} p_i$

The last table gives the distribution over the different types of earned keywords, under the different P_j:

$profit$	$\eta_t = 0$	$\eta_t = 1$	$\dots$	$\eta_t = j$	$\eta_t = j+1$	$\dots$	$\eta_t = k-1$	$\eta_t = k$
0	$1-p_0$	$1-p_0-p_1$	$\dots$	$1-\sum\limits_{i=0}^{j} p_i - \frac{1}{10}p_{j+1}$	$1-\sum\limits_{i=0}^{j+1} p_i$	$\dots$	$1-\sum\limits_{i=0}^{k-1} p_i$	$1-\sum\limits_{i=0}^{k} p_i$
$m \cdot \pi_0$	p_0	0	$\dots$	0	0	$\dots$	0	0
$m \cdot \pi_1$	0	p_0+p_1	$\dots$	0	0	$\dots$	0	0
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$m \cdot \pi_j$	0	0	$\dots$	$\sum\limits_{i=0}^{j} p_i + \frac{1}{10}p_{j+1}$	0	$\dots$	0	0
$m \cdot \pi_{j+1}$	0	0	$\dots$	0	$\sum\limits_{i=0}^{j+1} p_i$	$\dots$	0	0
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$m \cdot \pi_{k-1}$	0	0	$\dots$	0	0	$\dots$	$\sum\limits_{i=0}^{k-1} p_i$	0
$m \cdot \pi_k$	0	0	$\dots$	0	$\dots$		0	$\sum\limits_{i=0}^{k} p_i$

4.4   Information theoretic bound

Next, following [6], we derive a useful information theoretic based bound that will lead us to prove our lower bound. For any random variable Y depending on a sequence of daily sequences s_i we will write $E_{base}(Y)$, $E_j(Y)$ to denote the expectation of Y with respect to the distributions P_{base} and P_j, respectively.

We denote the profit gained by the algorithm choosing prefix η_t at time t as r_t, by $\mathbf{r}^t$ the vector of profits $(r_1, r_2, \ldots r_t)$, and by $\mathbf{r}$ the full vector of profits $\mathbf{r}^T$, where T is the number of daily sequences sampled from $(s_i)_{i=0}^k$.

Let N_i denotes the random variable specifying the number of times prefix i is selected by the algorithm. The expected profit for each prefix $\eta_t = i$, $i \in \{0, 1, \ldots, k\}$ over the distribution P_{base} is:

$$E_{base}[r_t|\eta_t = i] = (\sum_{n=0}^{i} p_n)m\pi_i = \tfrac{m/2}{1-i\varepsilon}(1 - i\varepsilon) = \tfrac{m}{2}.$$

For the distribution P_j we get $\forall i \in \{0, 1, \ldots, k-1\}, i \neq j$:

$$E_j[r_t|\eta_t = i] = (\sum_{n=0}^{i} p_n)m\pi_i = \tfrac{m/2}{1-i\varepsilon}(1 - i\varepsilon) = \tfrac{m}{2},$$

and for $i = j$:

$$
\begin{aligned}
E_j[r_t|\eta_t = j] &= (\sum_{n=0}^{j} p_n + \frac{1}{10}p_{j+1})m\pi_j \\
&= \frac{m/2}{(1 - j\varepsilon)}(1 - j\varepsilon) + \frac{m}{10}p_{j+1}(1 - j\varepsilon)
\end{aligned}
$$

$$= \frac{m}{2} + \frac{m}{10}\left(\frac{1/2}{1-(j+1)\varepsilon} - \frac{1/2}{1-j\varepsilon}\right)(1-j\varepsilon)$$

$$= \frac{m}{2} + \frac{m}{20}\frac{1-j\varepsilon-(1-j\varepsilon-\varepsilon)}{(1-j\varepsilon-\varepsilon)(1-j\varepsilon)}(1-j\varepsilon)$$

$$= \frac{m}{2} + \frac{m}{20}\frac{\varepsilon}{(1-j\varepsilon-\varepsilon)}$$

$$\geq \frac{m}{2} + \frac{m}{20}\varepsilon.$$

The following lemma bounds the distance between the expectation of two functions of the profit sequence $\mathbf{r}$, with respect to the different distributions P_{base} and P_j.

Lemma 18 *Let $f : \{1, 1-\varepsilon, 1-2\varepsilon, \dots, 1-k\varepsilon, 0\}^T \to [0, M]$ be any function defined on profits sequence $\mathbf{r}$. $E_i[f(\mathbf{r})] - E_{base}[f(\mathbf{r})] \leq \sqrt{(2\ln 2)KL(P_{base}\{\mathbf{r}\}||P_i\{\mathbf{r}\})}$*

Proof: For any funcion f we havge $E_i[f(\mathbf{r})] - E_{base}[f(\mathbf{r})] \leq \frac{M}{2}||P_i\{\mathbf{r}\} - P_{base}\{\mathbf{r}\}||_1$. Also, using the known relationship between the L_1 norm and the KL-divergence (see Lemma 12.6.1 of [15]) we get that $||P_i\{\mathbf{r}\} - P_{base}\{\mathbf{r}\}||_1^2 \leq (2\ln 2)KL(P_{base}\{\mathbf{r}\}||P_i\{\mathbf{r}\})$. ■

The next lemma refers to functions defined over profit sequences $\mathbf{r}$. We notice that the profit r_t is a function of the policy η_t selected by the algorithm A, which is a function of the profits $\mathbf{r}^{t-1}$. Therefore, N_i is also a function of $\mathbf{r}$, as, $N_i = \sum_{j=0}^{T} I(A(\mathbf{r}^{t-1}) = i)$ where I is the indicator function and $A(\mathbf{r}^t)$ is the policy η_{t+1} selected by the algorithm at time $t+1$ given the profits vector $\mathbf{r}^t$.

Lemma 19 *Let $f : \{1, 1-\varepsilon, 1-2\varepsilon, \dots, 1-k\varepsilon, 0\}^T \to [0, M]$ be any function defined on profits sequence $\mathbf{r}$. Then for $\varepsilon \leq \frac{1}{4k}$ and for any $i \in \{0, \dots, k-1\}$ we have*

$$E_i[f(\mathbf{r})] \leq E_{base}[f(\mathbf{r})] + 0.1\varepsilon M\sqrt{E_{base}[N_i]}$$

Proof: The proof follows the proof of Lemma A.1 in [6], except for the estimate of the relative entropy KL $(P_{base}\{r_t|\mathbf{r}^{t-1}\}||P_i\{r_t|\mathbf{r}^{t-1}\})$.

Given a deterministic algorithm A:

$$\text{KL}(\text{P}_{\text{base}}\{\mathbf{r}\}||\text{P}_i\{\mathbf{r}\}) \triangleq \sum_{\mathbf{r}\in\{m,m(1-\varepsilon),m(1-2\varepsilon),..,0\}^{\text{T}}} \text{P}_{\text{base}}\{\mathbf{r}\}\lg(\frac{\text{P}_{\text{base}}\{\mathbf{r}\}}{\text{P}_i\{\mathbf{r}\}})$$

$$= \sum_{t=1}^{\text{T}}\text{KL}(\text{P}_{\text{base}}\{r_t|\mathbf{r}^{t-1}\}||\text{P}_i\{r_t|\mathbf{r}^{t-1}\})$$

$$= \sum_{t=1}^{T}\sum_{\mathbf{r}^t\in\{m,m(1-\varepsilon),m(1-2\varepsilon),..,m(1-k\varepsilon),0\}^t}\text{P}_{\text{base}}\{\mathbf{r}^t\}\lg(\frac{\text{P}_{\text{base}}\{r_t|\mathbf{r}^{t-1}\}}{\text{P}_i\{r_t|\mathbf{r}^{t-1}\}})$$

$$
= \sum_{t=1}^{T} \sum_{\mathbf{r}^{t-1}\in\{m,m(1-\varepsilon),..,0\}^{t-1}} P_{\text{base}}\{\mathbf{r}^{t-1}\} \sum_{r_t\in\{m,m(1-\varepsilon),..,0\}} P_{\text{base}}\{r_t|\mathbf{r}^{t-1}\} \lg(\frac{P_{\text{base}}\{r_t|\mathbf{r}^{t-1}\}}{P_i\{r_t|\mathbf{r}^{t-1}\}})
$$

$$
= \sum_{t=1}^{T} \sum_{\substack{\mathbf{r}^{t-1}\in\{m,m(1-\varepsilon),..,0\}^{t-1}\\A(\mathbf{r}^{t-1})=i}} P_{\text{base}}\{\mathbf{r}^{t-1}\} \sum_{r_t\in\{m,m(1-\varepsilon),..,0\}} P_{\text{base}}\{r_t|\mathbf{r}^{t-1}\} \lg(\frac{P_{\text{base}}\{r_t|\mathbf{r}^{t-1}\}}{P_i\{r_t|\mathbf{r}^{t-1}\}})
$$

$$
+ \sum_{\substack{\mathbf{r}^{t-1}\in\{m,m(1-\varepsilon),..,0\}^{t-1}\\A(\mathbf{r}^{t-1})\neq i}} P_{\text{base}}\{\mathbf{r}^{t-1}\} \sum_{r_t\in\{m,m(1-\varepsilon),..,0\}} P_{\text{base}}\{r_t|\mathbf{r}^{t-1}\} \lg(\frac{P_{\text{base}}\{r_t|\mathbf{r}^{t-1}\}}{P_i\{r_t|\mathbf{r}^{t-1}\}}) \ )
$$

$$
= \sum_{t=1}^{T} (P_{\text{base}}\{\eta_t = i\} KL(\sum_{j=0}^{i} p_j || \sum_{j=0}^{i} p_j + \frac{1}{10}p_{i+1}))
$$

$$
= \sum_{t=1}^{T} (P_{\text{base}}\{\eta_t = i\} KL(\frac{1/2}{(1-i\varepsilon)} || \frac{1/2}{(1-i\varepsilon)} + \frac{1}{10}(\frac{1/2}{1-(i+1)\varepsilon} - \frac{1/2}{1-i\varepsilon})))
$$

$$
= \sum_{t=1}^{T} (P_{\text{base}}\{\eta_t = i\} KL(\frac{1/2}{v} || \frac{9}{10} \cdot \frac{1/2}{v} + \frac{1}{10} \cdot \frac{1/2}{v-\varepsilon}))
$$

where in the sixth identity we used the fact that for $A(\mathbf{r}^{t-1}) \neq i$, the conditional distributions $P_{\text{base}}\{r_t|\mathbf{r}^{t-1}\}$ and $P_i\{r_t|\mathbf{r}^{t-1}\}$ are identical. For $\varepsilon \leq 1/4k$ and $v \in [\frac{3}{4}, 1]$, using $\ln(1-x) \leq -x$ a rather tedious computation verifies that:

$$
KL(\frac{1/2}{v} || \frac{9}{10} \cdot \frac{1/2}{v} + \frac{1}{10} \cdot \frac{1/2}{v-\varepsilon})
$$

$$
= \frac{1/2}{v} \log \frac{1/2v}{\frac{9}{10} \cdot \frac{1/2}{v} + \frac{1}{10} \cdot \frac{1/2}{v-\varepsilon}} + (1 - \frac{1/2}{v}) \log \frac{1 - 1/2v}{1 - \frac{9}{10} \cdot \frac{1/2}{v} - \frac{1}{10} \cdot \frac{1/2}{v-\varepsilon}}
$$

$$
= \frac{1}{2v} \log \frac{1/2v}{\frac{9}{10} \cdot \frac{(v-\varepsilon)}{2v(v-\varepsilon)} + \frac{1}{10} \cdot \frac{v}{2v(v-\varepsilon)}} + (\frac{2v-1}{2v}) \log \frac{\frac{2v-1}{2v}}{\frac{2v(v-\varepsilon) - \frac{9}{10}v + \frac{9}{10}\varepsilon - \frac{1}{10}v}{2v(v-\varepsilon)}}
$$

$$
= \frac{1}{2v} \log \frac{v-\varepsilon}{\frac{9}{10} \cdot (v-\varepsilon) + \frac{1}{10} \cdot v} + (\frac{2v-1}{2v}) \log \frac{(2v-1)(v-\varepsilon)}{2v(v-\varepsilon) - (v - \frac{9}{10}\varepsilon)}
$$

$$
= \frac{1}{2v} \log \frac{v-\varepsilon}{v - \frac{9}{10}\varepsilon} + \frac{2v-1}{2v} \log \frac{(2v-1)(v-\varepsilon)}{(2v-1)(v-\varepsilon) - \frac{\varepsilon}{10}}
$$

$$
= \frac{1}{2v} \log \left(1 - \frac{\frac{1}{10}\varepsilon}{v - \frac{9}{10}\varepsilon}\right) + \frac{2v-1}{2v} \log \left(1 + \frac{\frac{1}{10}\varepsilon}{(2v-1)(v-\varepsilon) - \frac{\varepsilon}{10}}\right)
$$

$$
= \frac{1}{\ln 2} \cdot \left[\frac{1}{2v} \ln \left(1 - \frac{\frac{1}{10}\varepsilon}{v - \frac{9}{10}\varepsilon}\right) + \frac{(2v-1)}{2v} \ln \left(1 + \frac{\frac{1}{10}\varepsilon}{(2v-1)(v-\varepsilon) - \frac{\varepsilon}{10}}\right)\right]
$$

$$
\leq \frac{1}{\ln 2} \cdot \left[\frac{1}{2v} \left(-\frac{\frac{1}{10}\varepsilon}{v - \frac{9}{10}\varepsilon}\right) + \frac{(2v-1)}{2v} \frac{\frac{1}{10}\varepsilon}{(2v-1)(v-\varepsilon) - \frac{\varepsilon}{10}}\right]
$$

$$
\leq \frac{1}{\ln 2} \cdot \frac{\varepsilon/10}{2v} \cdot \frac{(2v-1)(v-\frac{9}{10}\varepsilon) - [(2v-1)(v-\varepsilon) - \frac{\varepsilon}{10}]}{[(2v-1)(v-\varepsilon) - \frac{\varepsilon}{10}](v - \frac{9}{10}\varepsilon)}
$$

$$\leq \quad \frac{1}{\ln 2} \cdot \frac{\varepsilon/10}{2v} \cdot \frac{\frac{\varepsilon}{10}(2v-1)+\frac{\varepsilon}{10}}{[(2v-1)(v-\varepsilon)-\frac{\varepsilon}{10}](v-\frac{9}{10}\varepsilon)}$$

$$\leq \quad \frac{1}{\ln 2} \cdot \frac{\varepsilon/10}{2v} \cdot \frac{\frac{\varepsilon}{10}2v}{[(2v-1)(v-\varepsilon)-\frac{\varepsilon}{10}](v-\frac{9}{10}\varepsilon)}$$

$$\leq \quad \frac{1}{\ln 2} \cdot \frac{\varepsilon^2/100}{[(\frac{1}{2}+2\varepsilon)\cdot\frac{3}{4}-\frac{\varepsilon}{10}]\frac{3}{4}} \leq \frac{1}{\ln 2} \cdot \frac{32}{900}\varepsilon^2$$

$$\leq \quad 0.0246 \cdot \varepsilon^2.$$

Therefore $KL(P_{base}\{\mathbf{r}\}||P_i\{\mathbf{r}\}) \leq \mathrm{E}_{\mathrm{base}}[N_i](0.0246\varepsilon^2)$. Using Lemma 18 we get:

$$E_i[f(\mathbf{r})] \leq E_{base}[f(\mathbf{r})] + \frac{M}{2}\varepsilon\sqrt{0.0246\cdot 2ln2}\sqrt{\mathrm{E}_{\mathrm{base}}[N_i]}$$

which completes the proof. ∎

4.5 Adversarial Lower Bound

We are now ready to prove a lower bound on the loss of profit any algorithm will encounter compared to the optimal prefix solution in the adversarial model.

In [6], the authors define a model of random payoffs (depending on T but not on the algorithm) such that the expected regret of any algorithm on a random sample from this distribution is $\Omega(\sqrt{Tk})$. Their proof used a selection of one out of the k actions uniformly at random, and designating it as the "good" action. For all other actions, the payoff in each round is a uniform random sample from $\{0,1\}$, whether for that good action the payoff is a biased sample from $\{0,1\}$ which is 1 with probability $1/2+\varepsilon$, where $\varepsilon = \Theta(\sqrt{k/T})$. Therefore, a strategy which knows the good action will achieve expected payoff $T/2+\Theta(\sqrt{Tk})$. It can be shown, for information-theoretic reasons, that no strategy can learn the good action rapidly and reliably enough to play it more than $T/k+\Theta(\varepsilon\sqrt{T^3/k})$ times in expectation, from which the lower bound on regret follows.

Similarly, in our keyword optimization problem, the adversary constructs a problem instance for which all prefixes have the exact same profit except a single randomly selected prefix, i.e., a distribution P for which one of the prefixes is selected randomly as the "good" prefix and it's profit is slightly ($\frac{m}{20}\varepsilon$) higher than the profit of all other prefixes. Using Lemma 19, we show it is impossible for an algorithm to learn to identify the 'good' prefix fast enough and to utilize from it's additional profit.

As in [6], there is a trade-off between choosing ε too large (which makes it easy to learn which prefix is the "good" prefix), or too small (which leads to a negligible difference between the profit of the "good" prefix and all others). The optimal trade-off is achieved for $\varepsilon = \sqrt{k/T}$.

As in [6], the adversary's selections depends on T and not on the algorithm A.

Theorem 20 *For any given T, there exists a problem instance Φ in which for any algorithm A, $\Pi(\Phi, OPT) - \Pi(\Phi, A) = \Omega(\sqrt{Tk})$ for $k \leq T^{1/3}$ and $\Pi(\Phi, OPT) - \Pi(\Phi, A) = \Omega(T^{2/3})$ for $k > T^{1/3}$.*

Proof: Given T, our adversary constructs the sequences $s_0, s_1, \ldots, s_k$ according to Φ_{WC} and randomly selects P_i uniformly from $i \in \{0, 1, \ldots, k-1\}$. We will use $P_*\{\cdot\}$ to denote probability with respect to this random uniform choice of P_i.

Given an algorithm A, i.e., a deterministic on-line prefix selection strategy $\eta_t = A(\mathbf{r}^{t-1})$, the random variable N_i is a function of $\mathbf{r}$, i.e., $N_i \triangleq f : \mathbf{r}^T \to [0, T]$. Therefore we can use Lemma 19 to conclude that $E_i[N_i] \le E_{base}[N_i] + 0.1\varepsilon T \sqrt{E_{base}[N_i]}$.

$$
\begin{aligned}
E_i[r_t | \mathbf{r}^{t-1}] &= \frac{m}{2} P_i\{\eta_t \ne i\} + \left(\frac{m}{2} + \frac{m}{20} \cdot \frac{\varepsilon}{1 - (i+1)\varepsilon}\right) P_i\{\eta_t = i\} \\
&= \frac{m}{2} + \frac{m}{20} \cdot \frac{\varepsilon}{1 - (i+1)\varepsilon} P_i\{\eta_t = i\}.
\end{aligned}
$$

Using Lemma 19 for $i \in \{0, \ldots, k-1\}$ we get:

$$
\begin{aligned}
E_i(\Pi_A) &= \sum_{t=1}^{T} E_i[r_t] = \frac{mT}{2} + \frac{m}{20} \cdot \frac{\varepsilon}{1 - (i+1)\varepsilon} E_i[N_i] \\
&\le \frac{mT}{2} + \frac{m\varepsilon}{20(1 - (i+1)\varepsilon)} (E_{base}[N_i] + 0.1\varepsilon T \sqrt{E_{base}[N_i]}).
\end{aligned}
$$

Before the daily sequences begin, one prefix $i \in \{0, \ldots, k-1\}$ was chosen uniformly at random by the adversary to be the 'good' prefix. Using Lemma 19 and Jensen's inequality for $f(x) = \sqrt{x}$ and $i \in [0, k-1]$, $1 \ge 1 - (i+1)\varepsilon \ge \frac{3}{4}$, we get:

$$
\begin{aligned}
E_*[\Pi_A] &= \frac{1}{k} \sum_{i=0}^{k-1} E_i[\Pi_A] \\
&\le \frac{1}{k} \sum_{i=0}^{k-1} \left(\frac{mT}{2} + \frac{m\varepsilon}{20(1 - (i+1)\varepsilon)} (E_{base}[N_i] + 0.1T\varepsilon \sqrt{E_{base}[N_i]})\right) \\
&= \frac{mT}{2} + \sum_{i=0}^{k-1} \frac{m\varepsilon}{20k(1 - (i+1)\varepsilon)} E_{base}[N_i] + \sum_{i=0}^{k-1} \frac{0.1mT\varepsilon^2}{1 - (i+1)\varepsilon} \frac{1}{k} \sqrt{E_{base}[N_i]} \\
&\le \frac{mT}{2} + \frac{2m\varepsilon T}{30k} + \frac{2m\varepsilon^2 T}{300k} \sqrt{\frac{T}{k}}
\end{aligned}
$$

where we used $T = \sum_{i=0}^{k-1} E_{base}[N_i]$ and $\sqrt{T} \ge \sum_{i=0}^{k-1} \sqrt{E_{base}[N_i]}$.

When the adversary selects P_i to be the 'good' prefix, the expected profit of the policy which selects the good prefix $\eta_t = i$ in all of the days is:

$$
\begin{aligned}
E_*[\Pi_{max}] &= \frac{mT}{2} + \frac{mT}{20} \cdot \frac{\varepsilon}{1 - (i+1)\varepsilon} \\
&\ge \frac{mT}{2} + \frac{mT\varepsilon}{20}
\end{aligned}
$$

Therefore:

$$E_*[\Pi_{max} - \Pi_A] \geq \frac{mT\varepsilon}{20} - \frac{2mT\varepsilon}{30k} - \frac{2m\varepsilon^2 T}{300k}\sqrt{\frac{T}{k}}$$

$$= \frac{mT\varepsilon}{60}\left(3 - \frac{4}{k}\right) - \frac{m}{150}\frac{T^{3/2}\varepsilon^2}{k^{3/2}}$$

We can maximize the regret between the algorithm and the best prefix solution using $\varepsilon = \Theta(\sqrt{\frac{k}{T}})$ giving a lower bound of $\Omega(m\sqrt{Tk})$ for $k \geq 2$. As we required $\varepsilon \leq 1/4k$, this implies that $k \leq \Theta(T^{1/3})$.

For $k > T^{1/3}$, we can maximize the regret by using $\varepsilon = \frac{1}{4k}$, giving a lower bound of $\Omega(m\frac{T}{k})$. In that case the adversarial can simply use only $T^{1/3}$ keywords and by that, get $\Omega(mT^{2/3})$ ■

We conclude this chapter by providing a tight upper bound (up to the logarithmic factor) for the regret of an online algorithm compared to the optimal prefix solution. For that, we use the algorithm EXP3 proven by [6] to have an upper bound of $O(\sqrt{gk\log k})$ expected weak regret, where g is an upper bound on the total return of the single globally best action at time horizon T.

Corollary 3 *For any given problem instance Φ and any given T, an algorithm A which uses the EXP3 policy over the k different prefix solutions achieves $\Pi(\Phi, OPT) - \Pi(\Phi, A) = O(\sqrt{Tk\log k})$.*

Chapter 5

Experiments

This chapter includes empirical results for our setting. We follow the experimental setting suggested by [3], in which they have compared their model based algorithm with the model free algorithms EXP3 and UCB1. We show that by using a simple modification on the UCB1 algorithm, inspired by the bucket policies in Chapter 3, a model free policy can rapidly converge, with high probability, to a near optimal profit. We further show that when removing some of the assumptions of the model, a model free approach can actually outperform a model based approach.

5.1 Experimental Settings

Our first set of experiments uses the exact same model and settings described in [3]. The model slightly differs from the probabilistic model presented in Chapter 3, as follows. First, each keyword has an additional parameter - a clickthrough probability - which ties a fixed probability of a query being clicked with each of the keywords w_i. The clickthrough probabilities are the only unknown parameters, i.e., the distribution P over the daily sequences length and the vector of keyword probabilities $\vec{\lambda}$ are known. Second, the model assumes a very concentrated distribution for the number of queries P, namely, a Poisson distribution with a known mean μ. Following the experiments described in [3], there are two settings:

1. SMALL - each SMALL experiment has 100 randomly generated problem instances. Each instance, has 8000 keywords where the costs are sampled uniformly at random from the interval $[0.1, 0.3)$, the profits are sampled uniformly at random from the interval $[0.0, 1.0)$ and the clickthrough rates associated with the keywords are sampled uniformly at random from the interval $[0.0, 0.2)$. The daily sequence length is sampled from a Poisson distribution with a mean of 40,000. There are 200 time periods and a budget of \$400 per instance.

2. LARGE - similarly to the SMALL settings, in the LARGE experiments, we have 50,000 keywords, where the cost, profit and clickthrough rates are generated in the same way. The budget is increased to \$1000 per instance and the number of daily queries is sampled from a Poisson distribution with a mean of 150,000.

Figure 5.1 shows a typical histogram of the keywords' profit to cost ratio in a LARGE problem instance. We can see that the keyword profit to cost ratio is evenly distributed up to some value, after which, high ratios become less likely. The vertical line in Figure 5.1 marks the profit to cost ratio of the last keyword in the optimal prefix of this specific problem instance (the location of this keyword, near the end of the evenly distributed area, is typical to these problem settings).
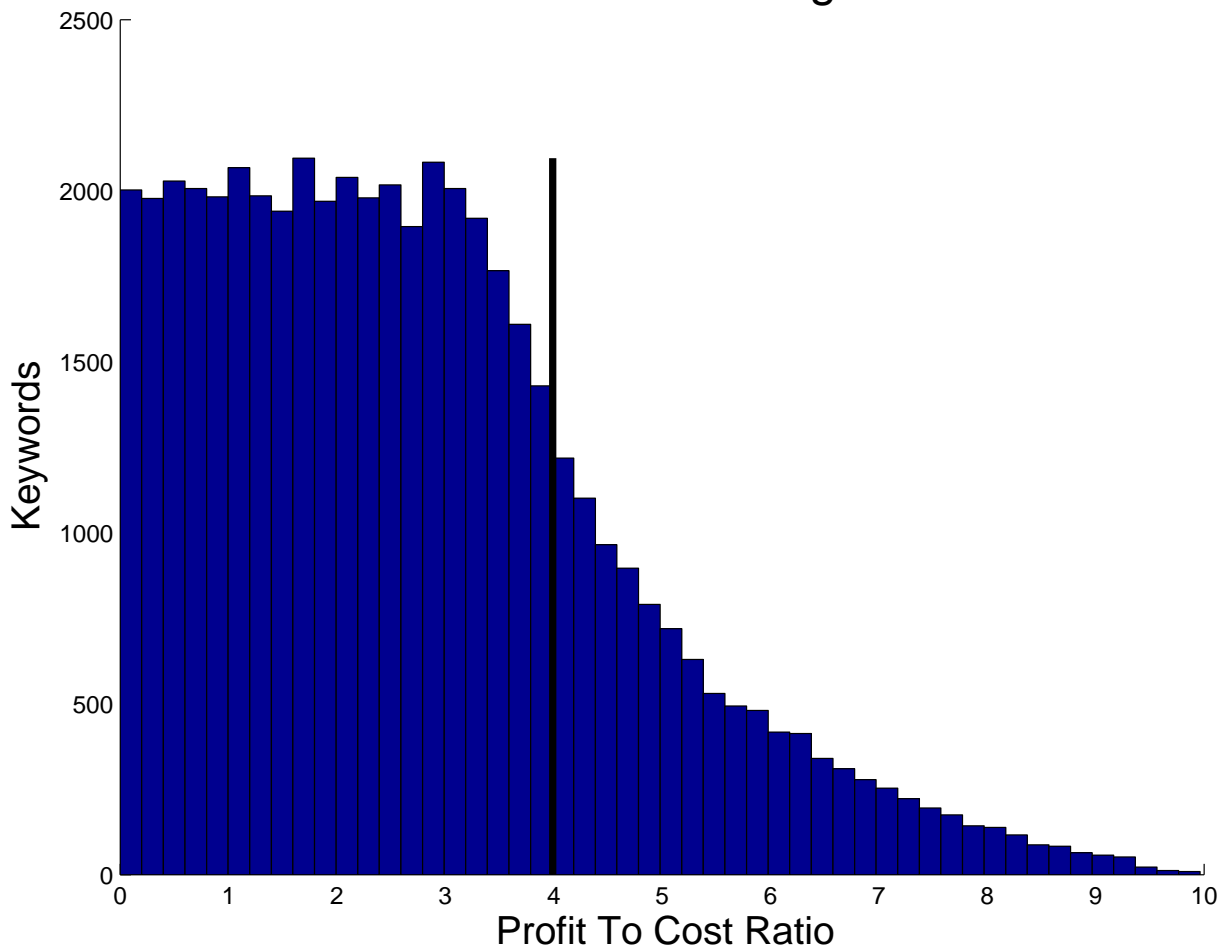
Figure 5.1: Keywords' profit to cost ratio histogram. The vertical line marks the location of the optimal prefix.

Figure 5.2 displays the profits as a function of the prefix index, for a sampled SMALL problem instance, averaged over 200 random days (the error bars illustrate the standard deviation). As can be seen in the figure, the profit curve seems to have a single global maxima, although from Figure 5.3, a zoomed version of the maxima area, we can see that finding the exact maxima with only a few samples may be a non-trivial task as the variance is quite large.

We conclude this discussion by noting that the experimental settings of [3], which we have adopted in our experiments, are very simple and directly match the theoretical model analyzed in [3]. Nevertheless, assuming a highly concentrated Poisson distribution for the number of daily queries, as well as assuming known and fixed keyword ratios, are assumptions which are hard to justify in practice.
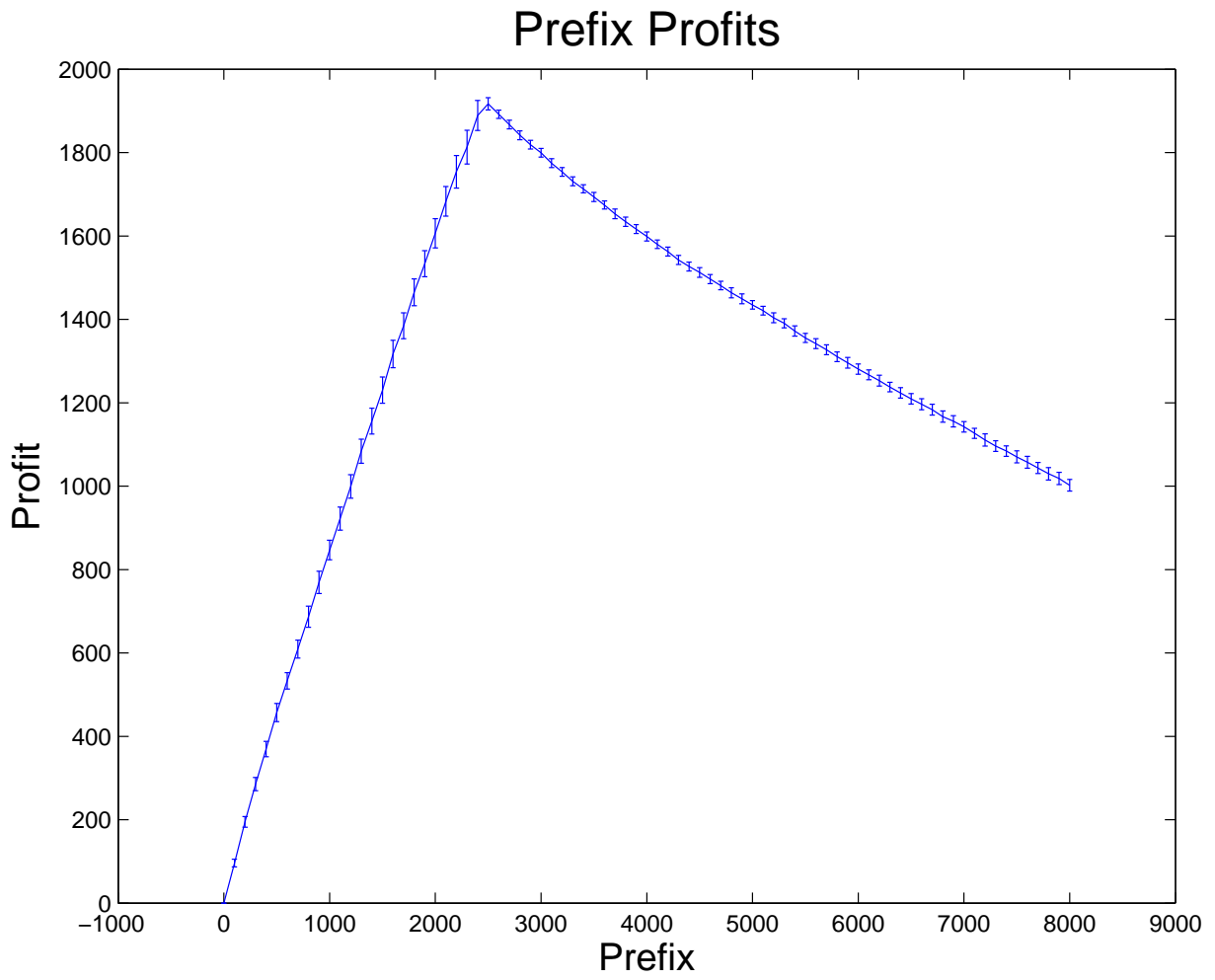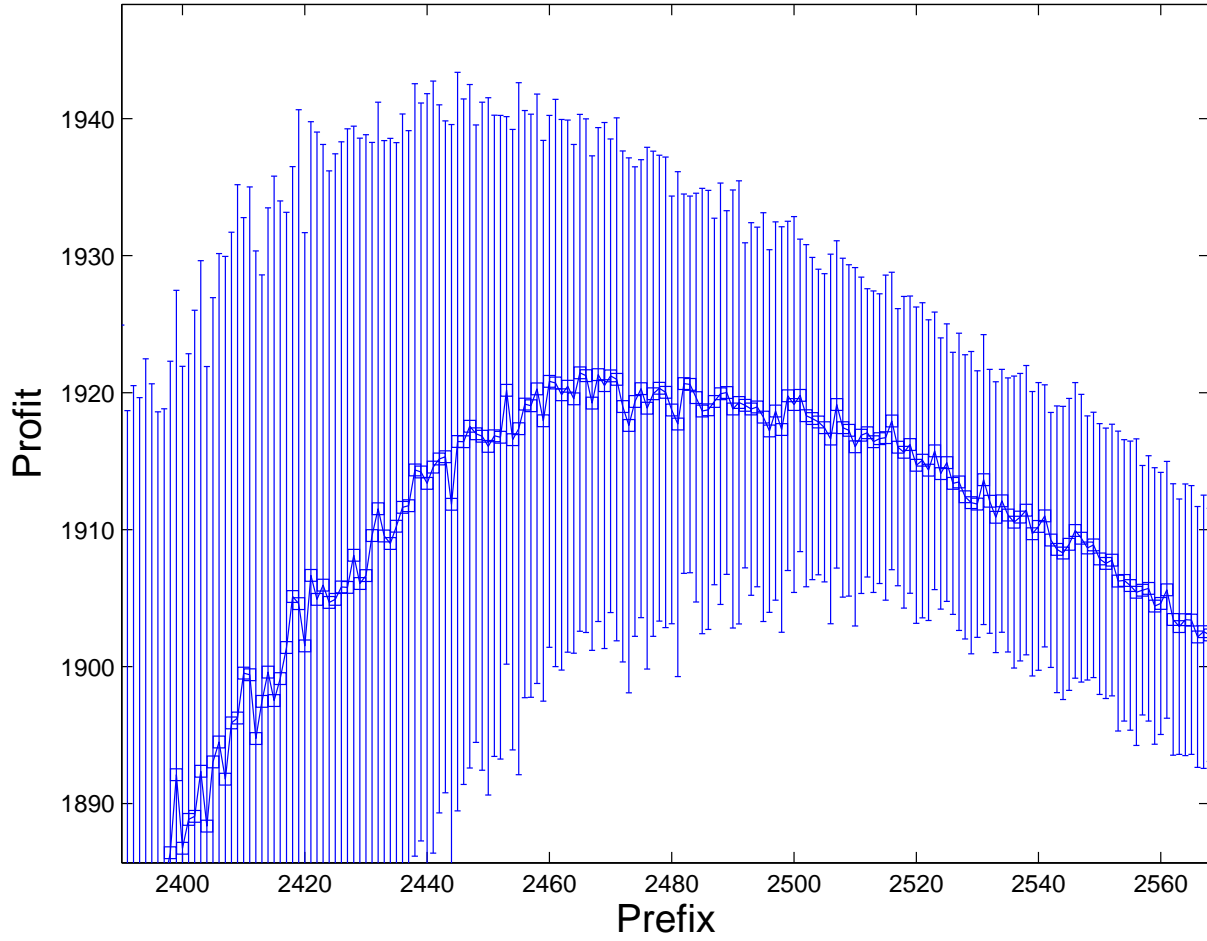
Figure 5.2: Prefix Profits

Figure 5.3: Prefix Profits - zoomed on the global maxima

5.2 The Adaptive Bidding algorithm

The adaptive bidding algorithm [3] is a model based algorithm which maintains a per-keyword estimate of the clickthrough rates based on the observed clicks and impressions. Given these estimates, the algorithm tries to optimize the profit by selecting the prefix solution which has an expected daily cost of the daily budget minus some slack. We emphasize that the ADAPTIVE BIDDING algorithm assumes all parameters of the model except clickthrough rates are known, namely, the distribution P over the daily sequences length, the keyword probabilities $\overrightarrow{\lambda}$, costs $\overrightarrow{c}$ and profits $\overrightarrow{\pi}$, are all known. Here is a formal presentation of the adaptive bidding algorithm.

ADAPTIVE BIDDING [3]:

1. INPUT: P, $\overrightarrow{\lambda}$, $\overrightarrow{c}$, $\overrightarrow{\pi}$, B_{day}.

    We are given a sequence $(\gamma_t \in [0,1] : t \geq 1)$ where γ_t will denote the probability that we choose a random decision at time t.

2. INITIALIZATION: For any keyword i, let y_i^t and x_i^t denote the number of impressions and number of clicks, respectively, that the ad associated with keyword i has received during day t. Set $y_i^0 = x_i^0 = 0$ and $\hat{p}_i^0 = p_{init}$ for all $0 \leq i \leq n$.

3. At time t:

    (a) Let l_t be the index such that:

    $$E(|S|) \cdot \sum_{u=0}^{l_t} c_u \lambda_u \hat{p}_u^{t-1} \leq B_{day}(1 - \frac{1}{k} - \frac{2}{k^{(1-\alpha)/2}}) \leq E(|S|) \cdot \sum_{u=0}^{l_t+1} c_u \lambda_u \hat{p}_u^{t-1},$$

    where k and α are parameters selected such that $k \geq 1$, $0 \leq \alpha < 1$, $c_i \leq 1/k$ and $\lambda_i \cdot E[|S|] \leq k^\alpha$ for all i.

    (b) With probability $1 - \gamma_t$ select the prefix policy l_t, otherwise, uniformly select a random prefix.

    (c) Observe the resulting impressions x_i^t and clicks y_i^t.

    (d) Updates: For any keyword i, update $\hat{p}_i^t = \dfrac{\sum_{j=1}^{t} x_i^j}{\sum_{j=1}^{t} y_i^j}$ if $\sum_{j=1}^{t} y_i^j > 0$, or $\hat{p}_i^t = p_{init}$ if

    $\sum_{j=1}^{t} y_i^j = 0$.

Notice that k and α place constrains on the magnitude of the cost and the expected number of arrivals associated with each keyword and their rational (as well as the way to choose them empirically) is detailed in [3]. In practice, we actually found that the Adaptive Bidding algorithm may be considerably improved by removing this slack, i.e., setting $k = \infty$ and selecting l_t such that:

$$E(|S|) \cdot \sum_{u=0}^{l_t} c_u \lambda_u \hat{p}_u^{t-1} \leq B_{day} \leq E(|S|) \cdot \sum_{u=0}^{l_t+1} c_u \lambda_u \hat{p}_u^{t-1}.$$

We refer to this variation as ADAPTIVE-BIDDING-ZERO-SLACK. Although a slack is necessary for the proof of the theoretical results in [3], in practice it seems to reduce the actual profits considerably.

In [3] the authors present a variety of initialization methods and randomization probabilities (γ_t). For the fairness of the comparison with our suggested algorithm, we used their best reported setting (which was verified also by our experiments). Namely, we used an initial probability p_{init} of 0 and a randomization probability of $\gamma_t = 1/t$.

5.3   An improved UCB1 algorithm

Closely observing the behavior of UCB1 [5] and EXP3 [6] algorithms show that in our settings, both algorithms perform poorly due to the large action space (50000 different prefixes in the LARGE setting and 8000 in the SMALL setting) compared to the low number of time periods (200). In such settings, both algorithms remain in the exploration stage and do not have a real chance to get to any exploitation of the learned actions values. Motivated by this observation and by the theoretical result of Chapter 3 (Corollary 2), we present an alternative model free algorithm which exploits the special structure of our problem. Our algorithm is a simple variation on the algorithm UCB1-TUNED from [5], where here, the bandit actions are buckets of consecutive keywords, as detailed in Chapter 3. The algorithm, ADAPTIVE-BUCKET-UCB1, follows UCB1-TUNED but has an important difference which allows convergence to near optimal performance with high probability. This is done using a refinement parameter, τ, which defines the rate in which some 'good' action/bucket is split into two smaller buckets. Each such split adds a single action to the action space. The criterion for selecting this 'good' bucket, which is initialized with the parameters of the bucket from which it was split, is detailed below.

ADAPTIVE-BUCKET-UCB1:
Let μ_i and σ_i denote the observed mean and standard deviation, respectively, of the profits gained by choosing all keywords from the prefix bucket policy i, i.e, selecting all keywords from $A_0, \ldots, A_i$.

1. INPUT: $\overrightarrow{c}$, $\overrightarrow{\pi}$, a refinement parameter τ and a regularization parameter α.

2. INITIALIZATION: Initialize a single prefix bucket A_1 of all sorted keywords.

3. At time t:

(a) Select the prefix solution $A_0, \ldots, A_i$ for the i which has the maximal $\mu_i + \hat{\sigma}_i + \alpha \cdot \chi_i$ where $\hat{\sigma}_i = \sqrt{\frac{ln(t)}{n_i} min\{1/4, V_i(n_i)\}}$, $V_i(n_i) = \sigma_i^2 + \sqrt{\frac{2lnt}{n_i}}$ is an upper confidence of the variance of action i, n_i is the number of times action i has been selected, and χ_i is a regularization term which is proportional, e.g., to the log of the size of the bucket A_i.

(b) If $t(mod\ \tau) \equiv 0$, for the i selected in (a), if $\chi_i > \chi_{i+1}$ split A_i, else split A_{i+1} (i.e., we select the larger bucket). Initialize the observed mean and standard deviation of the two new actions with the values μ_i and σ_i respectively.

In our experiments we have selected $\tau = 4$, $\alpha = 0.00003$. We note that by increasing the regularization parameter α, although slowing down the convergence rate, the algorithm can handle more complex profit landscapes and handle multiple maxima correctly.

5.4 Results

In order to examine the quality of the policies learned by the algorithms, we have terminated exploration after 100 days, i.e., at that time, the algorithm ADAPTIVE-BUCKET-UCB1 uses the observed means μ_i to select the best action. The algorithm ADAPTIVE BIDDING uses $\gamma_t = 0$, and the Algorithm EXP3 uses the action with the largest weight. This implies that we have 100 days for exploration and 100 days for exploitation.

Figures 5.4, 5.5 and 5.6, 5.7 show the simulation results for the SMALL and LARGE settings, respectively. These SMALL and LARGE settings overall result in qualitatively similar results. Figure 5.4, and 5.6 show the average profits (over 100 random problem instances) for the algorithms ADAPTIVE-BIDDING[3], ADAPTIVE-BIDDING-ZERO-SLACK, UCB1[5], ADAPTIVE-BUCKET-UCB1, and OPT (the optimal fixed prefix solution over the whole problem instance).

We can see that during the entire 100 days of exploration, UCB1 remains in an 'aggressive' exploration stage, while all other algorithms seem to quickly converge to much better policies (even during the exploration period). This should be expected due to the large action space handled by EXP3 in very few exploration steps and in fact, as seen in the exploitation stage, EXP3 is more likely to learn a poor policy. Our suggested ADAPTIVE-BIDDING-ZERO-SLACK algorithm seems to outperform ADAPTIVE-BIDDING from [3]. This happens due to a better use of the daily budget, which in the original ADAPTIVE-BIDDING algorithm tends to be partially un-exhausted. In fact, unlike the results presented in [3], terminating exploration in EXP3 in the LARGE problem instances actually outperforms the original ADAPTIVE-BIDDING algorithm in [3].

In Figure 5.5, and Figure 5.7 we consider an exponential average of the profit (with exponential decay factor of 0.95). We can see that in both the SMALL setting (Figure 5.5), and in the LARGE setting (Figure 5.7), ADAPTIVE-BUCKET-UCB1 and ADAPTIVE-BIDDING-ZERO-SLACK practically converge to near optimal average profits with high probability: in the SMALL setting, there is no significant difference between the mean profits learned by the two policies (0.44 P-Value in a paired T-Test), while in the LARGE setting, ADAPTIVE-BUCKET-UCB1 outperforms the ADAPTIVE-BIDDING-ZERO-SLACK
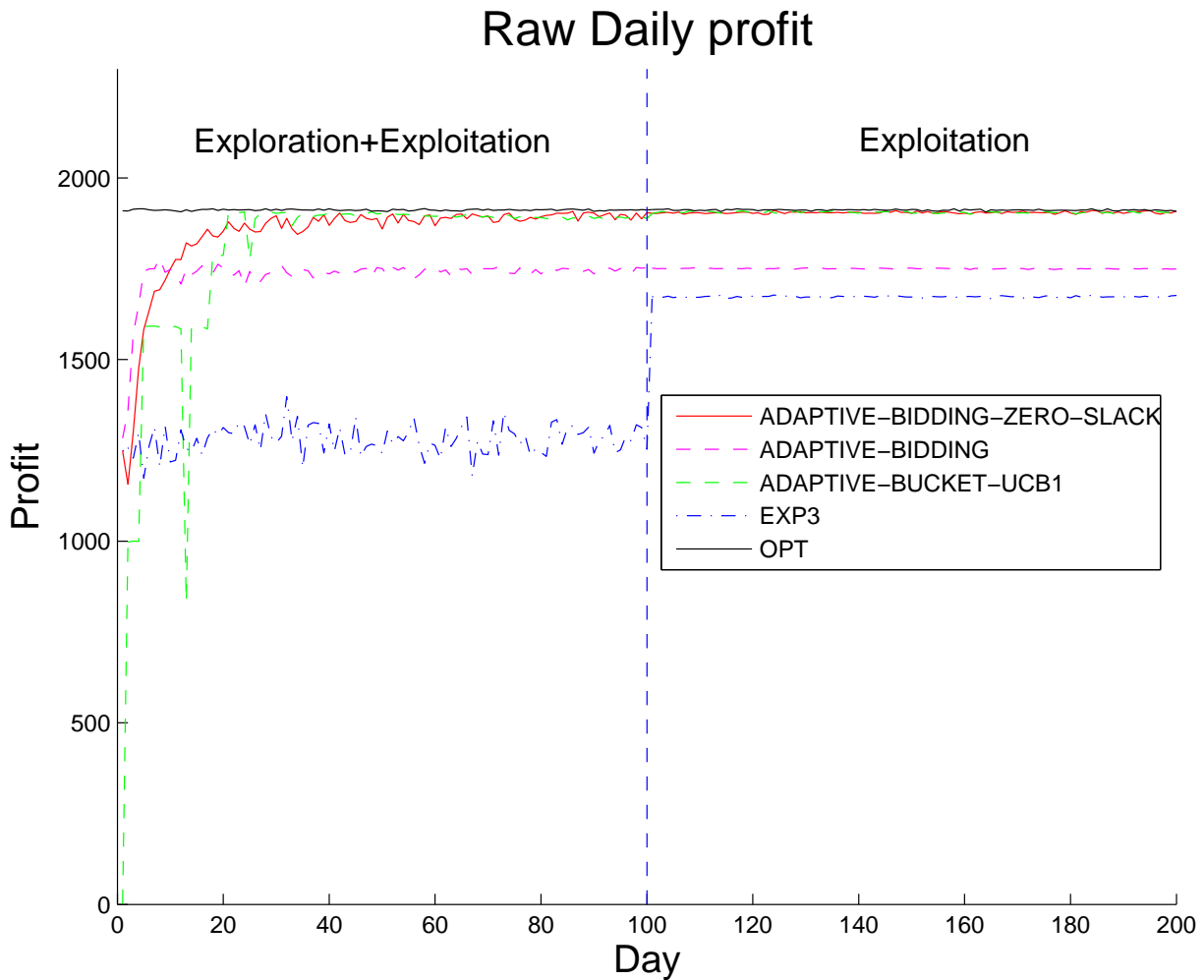
Figure 5.4: Raw profits - SMALL setting

method with a 0.2% relative improvement (statistically significant with a 8.4e-7 P-Value). ADAPTIVE-BUCKET-UCB1 only seconds ADAPTIVE-BIDDING-ZERO-SLACK with a slower convergence rate, which is due to the days 'wasted' on the first few bucket splits.

The bumps in the plots, which can mostly be seen in the exploration period, are due to 'bad' exploration steps.

We emphasize that ADAPTIVE-BIDDING-ZERO-SLACK requires the knowledge of the exact multi-nomial distribution $\overrightarrow{\lambda}$ and the average sequence length $E(|S|)$, while ADAPTIVE-BUCKET-UCB1 requires neither. In [3], the authors describe methods by which advertisers can model the multi-nomial distribution $\overrightarrow{\lambda}$ and the average sequence length $E(|S|)$, but modeling these thousands of parameters in a rapidly changing environment is a daunting task. In fact, the problem settings we have followed from [3] are obviously much simpler then real life problem setting in which the distribution P is much more complex and in which the keyword distribution $\overrightarrow{\lambda}$ is non-stationary.
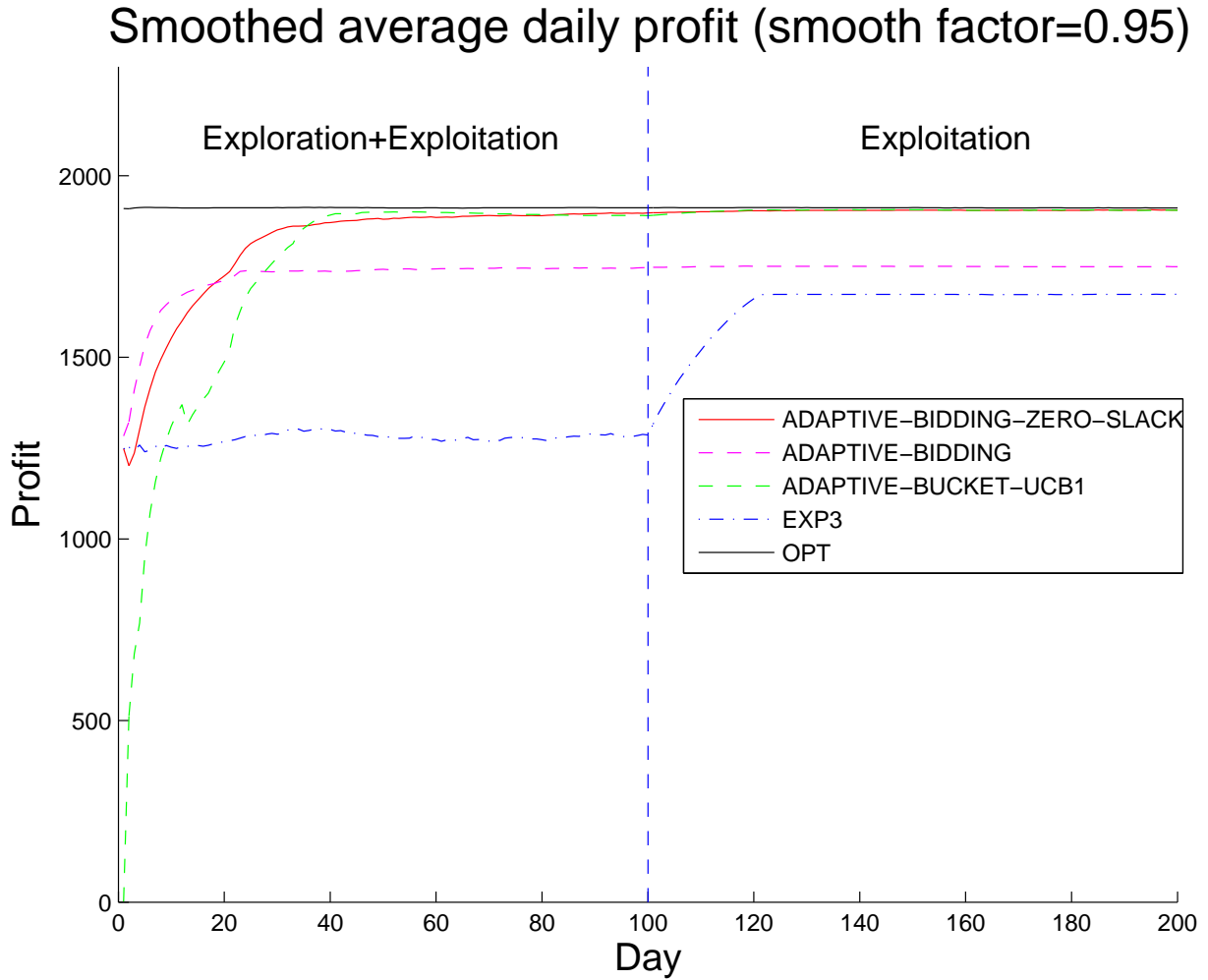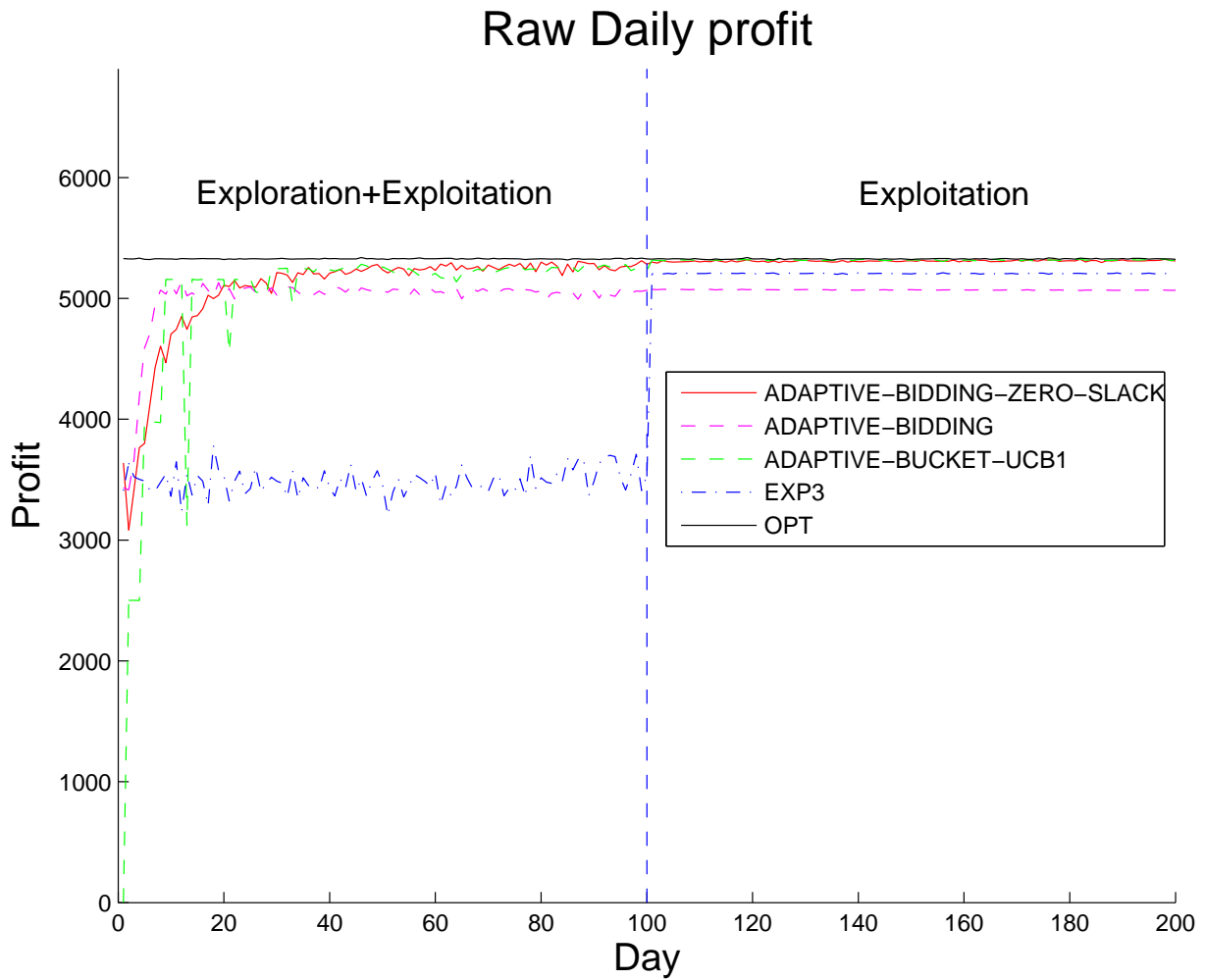
Figure 5.5: Smoothed profits - SMALL setting

Figure 5.6: Raw profits - LARGE setting

5.5 Robustness Of Model Free Algorithms

We end this chapter by repeating the previous experiment while introducing some error to the model parameters given to the algorithm. Our motivation comes from the fact that in the real life problem, assuming perfect knowledge of the distribution of the number of daily queries P, as well as the keyword distribution $\overrightarrow{\lambda}$ is far from from being feasible. A model free algorithm which doesn't model the complexity of the environment and in which a theoretical 'clean' setting may converge slower then a model based algorithm, may be a more robust approach to select.
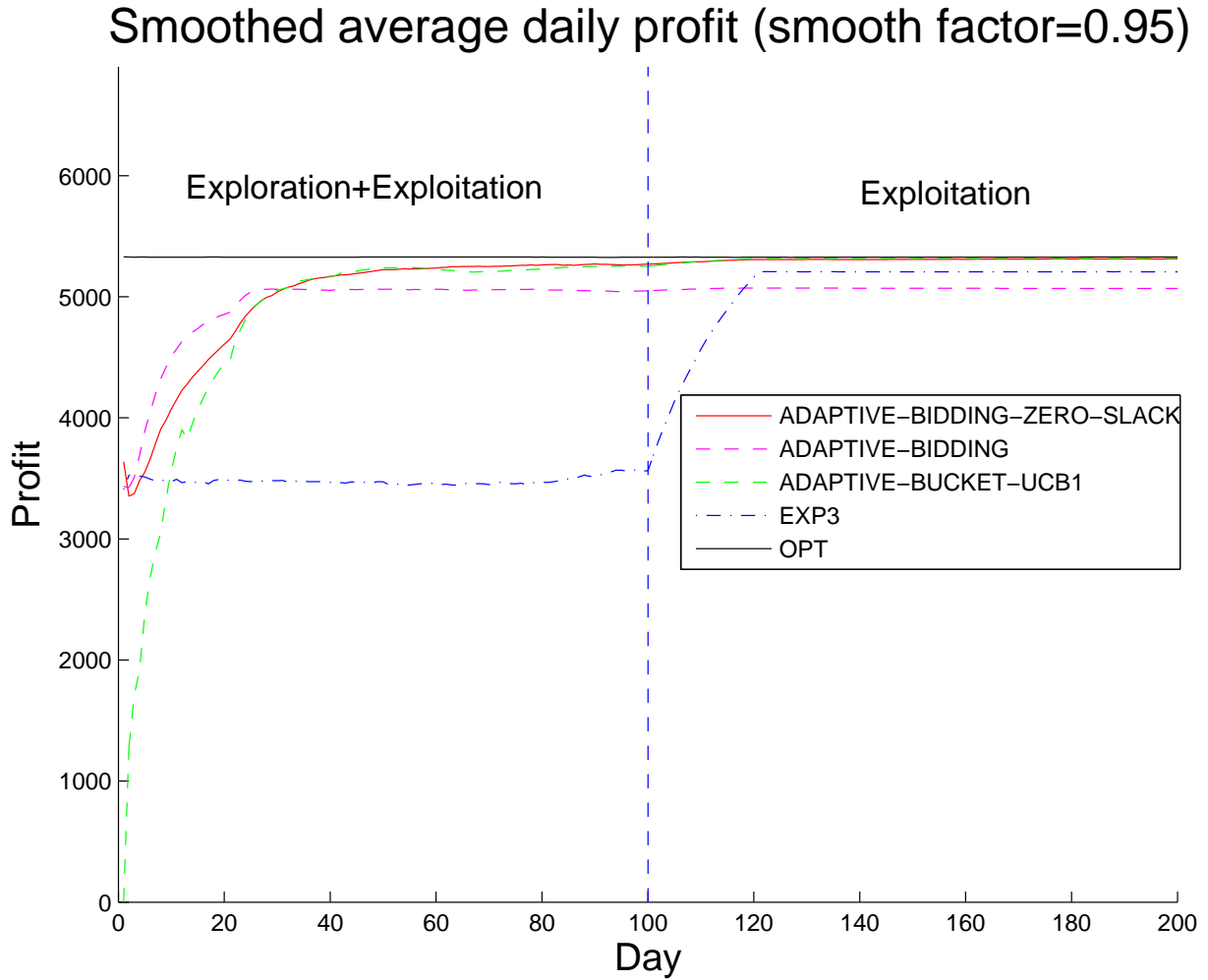
Figure 5.7: Smoothed profits - LARGE setting

In this experiment, rather then providing the ADAPTIVE-BIDDING-ZERO-SLACK algorithm with the correct mean series length $E(|S|)$, we provide it with $E(|S|) \cdot (1 + \epsilon)$ where ϵ varies between $[-0.2, 0.2]$, i.e., simulating a case where the mean series length is estimated with a relative error of up to 20%. The model free ADAPTIVE-BUCKET-UCB1 algorithm does not use the series length at all, nor does it use the keywords distribution $\overrightarrow{\lambda}$, in contrast to the ADAPTIVE-BIDDING-ZERO-SLACK algorithms which require to receive both as parameters.

Figure 5.8 shows that already at low error levels, the ADAPTIVE-BUCKET-UCB1 algorithm outperforms the ADAPTIVE-BIDDING-ZERO-SLACK algorithms. This implies that, as expected, the ADAPTIVE-BIDDING-ZERO-SLACK algorithms are very sensitive to model errors, a sensitivity to which a model-free algorithm such as ADAPTIVE-BUCKET-UCB1, does not suffer from.

One can also observe that the Adaptive-Bidding algorithm is much more sensitive to overestimates than to underestimates of the number of queries.
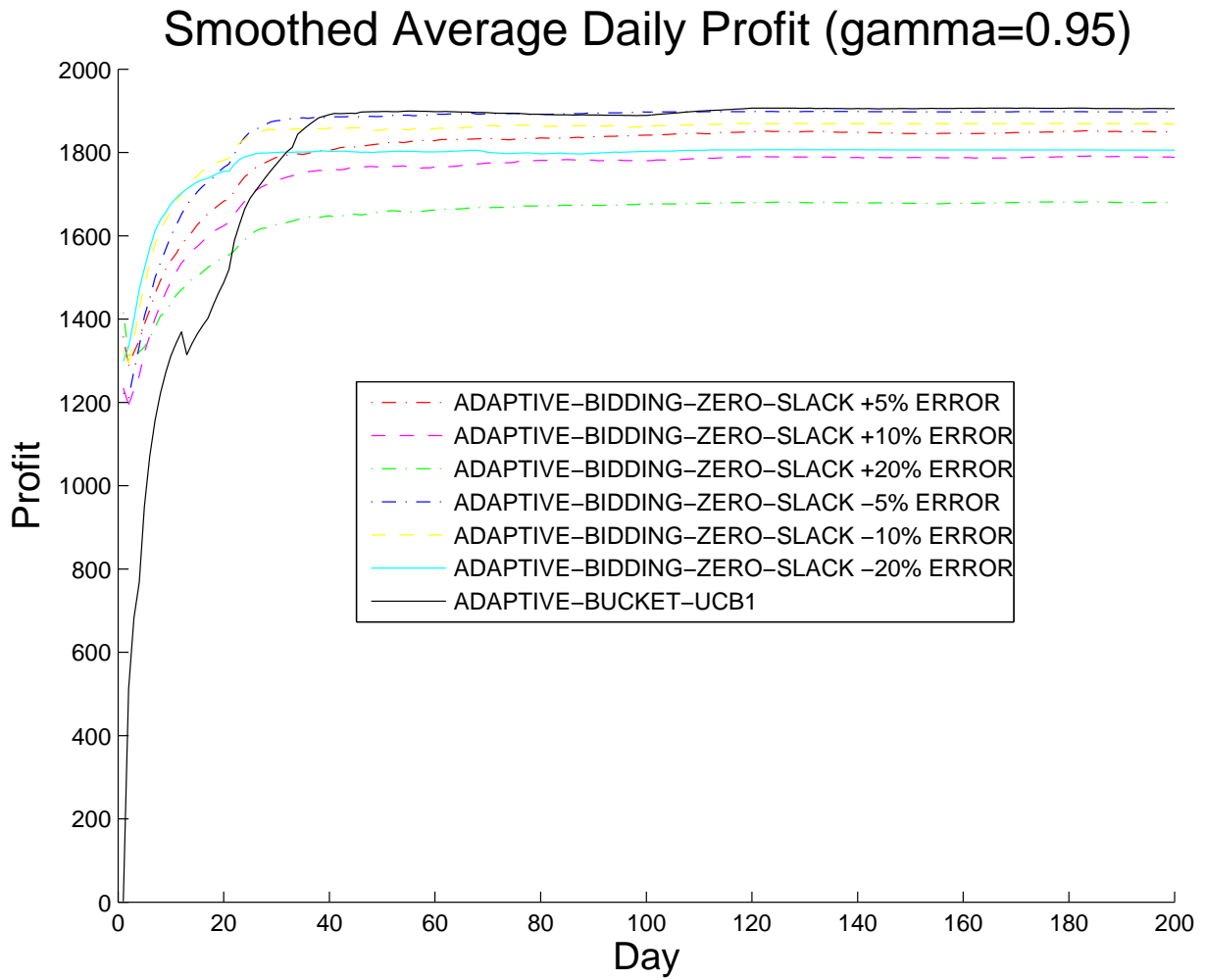
Figure 5.8: Smoothed profits, error in estimation of $E(|S|)$

Appendix A

Our proof to the lower bound in the probabilistic model is based on the proof used for the lower bound in the adversarial model presented in Chapter 4. We build a problem setting that with high probability produces the exact same profits presented there, which eventually lead to the same asymptotic regret, assuming long enough query sequences.

Proof of Theorem 17

Proof: We define the following problem instance Φ:

Let $\overrightarrow{\pi} = \{1, 1 - \varepsilon, \ldots, 1 - i\varepsilon, \ldots, 1 - k\varepsilon\}$, and let $\overrightarrow{\lambda} = \{\frac{Z}{x^k}, \frac{Z}{x^{k-1}}, \ldots, \frac{Z}{x^{k-i}}, \ldots, \frac{Z}{x^0}\}$, where Z is the normalizing constant, i.e., $Z = \frac{1}{\sum_{i=0}^{k} \frac{1}{x^{k-i}}}$. We keep all costs equal: $\forall i, \quad c_i = 1$ and the daily budget is set as $B_{day} = m$.

A day of type i, s_i, has a query sequence length: $|s_i| = m \cdot a \frac{x^{k-i}}{Z}$ where a and x are parameters that we set later, and the distributions over the day length P follow those presented in Chapter 4. I.e., $P_{base} = (p_0, p_1, .., p_k)$ is the following distribution over daily sequences $\{s_0, s_1, .., s_k\}$: $p_0 = \frac{1}{2}$ and for $i \in \{1, \ldots, K\}$,

$$p_i = \frac{1/2}{1 - i\varepsilon} - \frac{1/2}{1 - (i-1)\varepsilon} = \frac{1}{2} \cdot \frac{\varepsilon}{(1 - i\varepsilon)(1 - (i-1)\varepsilon)}.$$

In order to have a proper distribution that sums to one, we define a null query sequence, $s_\perp$ with probability $1 - \frac{1/2}{1-k\varepsilon} = \frac{1/2 - k\varepsilon}{1-k\varepsilon}$ and require that $\varepsilon \leq \frac{1}{2k}$.

We denote by P_i the distribution we get by removing an $\frac{p_{i+1}}{10}$ fraction from p_{i+1} and moving it to p_i (P_i is a perturbed version of P_{base}) for $i \in \{0, 1, \ldots, K-1\}$.

Claim 21 *For a day with $|s_i|$ queries, a prefix policy η_i gains a profit of $m\pi_i$ with probability at least $1 - \frac{2ma}{x} - e^{-\frac{ma}{2}}$.*

Proof: We bound the probability by considering the event that at least m queries of keyword w_i occur, and no query of keywords w_j, for $j \leq i - 1$ occur in a day with query sequence of length $|s_i|$.

We first show that in a day with $|s_i|$ queries, with high probability there will be more than m queries of keyword w_i. Let n_i be the r.v. of the number of queries of keyword w_i, given $|s_i|$ queries. By definition $E(n_i) = |s_i| \cdot \lambda_i = \frac{max^{k-i}}{Z} \frac{Z}{x^{k-i}} = ma$. using Chernoff's inequality $Pr[n_i < (1 - \delta)ma] < e^{-\frac{ma\delta^2}{2}}$, by setting $\delta = \frac{a-1}{a}$ and for $a \geq 2$ we can bound the number of queries n_i from keyword w_i that appear in a day of length $|s_i|$ as follows

$$Pr[n_i < m] = Pr[n_i < (1 - \delta)E(n_i)] = Pr[n_i < (1 - \frac{a-1}{a})ma) < e^{-\frac{ma}{2} \cdot (\frac{a-1}{a})^2} < e^{-\frac{ma}{2}}.$$

Second, for $x \geq 2$ we show that with high probability there will be no queries of any keyword $w_0, w_1, \ldots, w_{i-1}$ in a day with query sequence of length $|s_i|$:

$$Pr[\sum_{j=0}^{i-1} n_j > 0] \leq |s_i| \cdot \sum_{j=0}^{j=i-1} \lambda_j \leq |s_i| \cdot 2\lambda_{i-1} = ma\frac{x^{k-i}}{Z} \cdot 2\frac{Z}{x^{k-i+1}} = \frac{2ma}{x}.$$

Therefore, w.p. at least $1 - \frac{2ma}{x} - e^{-\frac{ab}{2}}$, we get that a prefix policy η_i earns an exact profit of $m \cdot \pi_i$. ∎

Claim 22 *For a day with $|s_i|$ queries, a prefix policy η_j for $j > i$ gains a profit of exactly $m\pi_j$ with probability at least $1 - \frac{2ma}{x} - e^{-\frac{ma}{2}}$.*

Proof: By definition, a day with query sequence of length $|s_i|$ has more queries than a day with query sequence of length $|s_j|$ for $j > i$. By Claim 21 for a day with query sequence of length $|s_j|$, after $|s_j|$ queries, w.p. at least $1 - \frac{2ma}{x} - e^{-\frac{ma}{2}}$, a prefix η_j has a profit of exactly $m \cdot \pi_j$. ∎

Claim 23 *For a day with $|s_i|$ queries, for all $j < i$, a prefix policy η_j gains no profit with probability equal or higher than $1 - \frac{2ma}{x}$.*

Proof: We can bound the number of queries from keywords $w_0, w_1, \ldots, w_j$ in a day with $|s_i|$ queries as follows:

$$Pr[\sum_{j=0}^{i-1} n_j > 0] \leq |s_i| \cdot \sum_{j=0}^{j=i-1} \lambda_j \leq |s_i| \cdot 2\lambda_{i-1} = ma\frac{x^{k-i}}{Z} \cdot 2\frac{Z}{x^{k-i+1}} = \frac{2ma}{x}.$$

∎

Therefore, using the union bound and the claims above, for T days, w.p. greater than $T(1 - \frac{6ma}{x} - 2e^{-\frac{ma}{2}})$ we get the exact same profit-per-prefix as in the proof of Theorem 20. Define $\delta = \frac{6ma}{x} + 2e^{-\frac{ma}{2}}$ and using the proof of Theorem 20 we get:

$$E_*[\Pi_A] \quad \leq \quad \delta T \cdot T \cdot m\pi_0 + (1 - \delta T)[\frac{mT}{2} + \frac{2m\varepsilon T}{30k} + \frac{2m\varepsilon^2 T}{300k}\sqrt{\frac{T}{k}}],$$

where we bounded the profit of the algorithm in the case where the profits do not match those in Chapter 4 with the maximum possible daily profit.

Also, the expected profit of the optimal prefix is lower bounded by:

$$E_*[\Pi_{max}] \quad \geq \quad (1 - \delta T)[\frac{mT}{2} + \frac{mT\varepsilon}{20}]$$

Therefore:

$$E_*[\Pi_{max} - \Pi_A] \quad \geq \quad \frac{mT\varepsilon}{60}(3 - \frac{4}{k} - 3\delta T) - \delta\frac{mT^2}{2} - \frac{m}{150}\frac{T^{3/2}\varepsilon^2}{k^{3/2}}$$

As in the proof of Theorem 20, by using $\varepsilon = \Theta(\sqrt{\frac{k}{T}})$ for $4 \leq k \leq \Theta(T^{1/3})$ we get:

$$
\begin{aligned}
E_*[\Pi_{max} - \Pi_A] &\geq \frac{m\sqrt{Tk}}{150}(\frac{15}{2} - 1 - 3\delta T - \frac{150T^{3/2}\delta}{2\sqrt{k}} - 1) \\
&= \frac{m\sqrt{Tk}}{150}(\frac{11}{2} - 3\delta T - \frac{150T^{3/2}\delta}{2\sqrt{k}})
\end{aligned}
$$

by setting $a = \frac{3}{m}(\log T + 2)$ and $x = \frac{225ma T^{3/2}}{\sqrt{K}}$, we get a lower bound of $\Theta(m\sqrt{kT})$ for $T < k^{1/3}$ and $\Theta(mT^{2/3})$ for $T \geq k^{1/3}$. ∎

References

[1] Blum, A. and Y. Mansour (2007). Learning, regret minimization, and equilibria. In N. Nisan, T. Roughgarden, E. Tardos, and V. Vazirani (Eds.), Algorithmic Game Theory. Cambridge, U.K.: Cambridge University Press.

[2] Benjamin Edelman, Michael Ostrovsky, and Michael Schwarz. Internet Advertising and the Generalized Second-Price Auction: Selling Billions of Dollars Worth of Keywords. American Economic Review, 97(1):242259, 2007.

[3] Paat Rusmevichientong and David P. Williamson. 2006. An adaptive algorithm for selecting profitable keywords for search-based advertising services. In Proceedings of the 7th ACM conference on Electronic commerce (EC '06). ACM, New York, NY, USA, 260-269. DOI=10.1145/1134707.1134736 http://doi.acm.org/10.1145/1134707.1134736

[4] Brian C. Dean, Michel X. Goemans, and Jan Vondrak. 2004. Approximating the Stochastic Knapsack Problem: The Benefit of Adaptivity. In Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science (FOCS '04). IEEE Computer Society, Washington, DC, USA, 208-217. DOI=10.1109/FOCS.2004.15 http://dx.doi.org/10.1109/FOCS.2004.15

[5] Peter Auer, Nicolo Cesa-Bianchi, and Paul Fischer. 2002. Finite-time Analysis of the Multiarmed Bandit Problem. Mach. Learn. 47, 2-3 (May 2002), 235-256. DOI=10.1023/A:1013689704352 http://dx.doi.org/10.1023/A:1013689704352

[6] Peter Auer, Nicolo Cesa-Bianchi, Yoav Freund, and Robert E. Schapire. 2003. The Nonstochastic Multiarmed Bandit Problem. SIAM J. Comput. 32, 1 (January 2003), 48-77. DOI=10.1137/S0097539701398375 http://dx.doi.org/10.1137/S0097539701398375

[7] S. Muthukrishnan, Martin Pal, and Zoya Svitkina. 2007. Stochastic models for budget optimization in search-based advertising. In Proceedings of the 3rd international conference on Internet and network economics (WINE'07), Xiaotie Deng and Fan Chung Graham (Eds.). Springer-Verlag, Berlin, Heidelberg, 131-142.

[8] Avrim Blum, Vijay Kumar, Atri Rudra, and Felix Wu. 2003. Online learning in online auctions. In Proceedings of the fourteenth annual ACM-SIAM symposium on Discrete algorithms (SODA '03). Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 202-204.

[9] Aranyak Mehta , Amin Saberi , Umesh Vazirani , Vijay Vazirani, AdWords and Generalized On-line Matching, Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science, p.264-273, October 23-25, 2005 [doi¿10.1109/SFCS.2005.12]

[10] S. Pandey, and C. Olston. Handling Advertisements of Unknown Quality in Search Advertising. NIPS 06

[11] Kuzman Ganchev, Alex Kulesza, Jinsong Tan, Ryan Gabbard, Qian Liu, and Michael Kearns. 2007. Empirical price modeling for sponsored search. In Proceedings of the 3rd international conference on Internet and network economics (WINE'07), Xiaotie Deng and Fan Chung Graham (Eds.). Springer-Verlag, Berlin, Heidelberg, 541-548.

[12] Christian Borgs, Jennifer Chayes, Nicole Immorlica, Kamal Jain, Omid Etesami, and Mohammad Mahdian. 2007. Dynamics of bid optimization in online advertisement auctions. In Proceedings of the 16th international conference on World Wide Web (WWW '07). ACM, New York, NY, USA, 531-540. DOI=10.1145/1242572.1242644 http://doi.acm.org/10.1145/1242572.1242644

[13] Michael R. Garey , David S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, W. H. Freeman & Co., New York, NY, 1979.

[14] T. L. Lai and H. Robbins. Asymptotically efficient adaptive allocation rules. Advances in Applied Mathematics, 6:4-22, 1985.

[15] T. M. Cover and J. A. Thomas, Elements of Information Theory. New York: Wiley, 1991.