OPTIMIZATION IN STOCHASTIC MODELS OF NETWORK APPLICATIONS

BY

BO TAN

DISSERTATION

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Electrical and Computer Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2012

Urbana, Illinois

Doctoral Committee:

Professor Rayadurgam Srikant, Chair
Professor M. Tamer Başar
Professor Bruce Hajek
Professor Venugopal Varadachari Veeravalli

ABSTRACT

In this dissertation, we propose stochastic models and develop optimal or near-optimal algorithms for resource allocation, for two important network applications: 1) video-on-demand (VoD) services in content distribution networks (CDNs) and 2) online advertising.

For the first application, we address the problem of content placement in VoD CDNs, with the objective of maximizing the utilization of CDN servers' uplink bandwidth resources. We consider system performance under a large network asymptotic. We first study the case of a finite content catalog, and distinguish two scenarios, namely a common "service-only" CDN for which requests are exogenous to the system, and an "ISP-managed peer-to-peer CDN" for which requests emanate from the servers (peers) themselves. For both scenarios, we consider a *loss network* model of performance, and determine asymptotically-optimal content placement strategies. We then turn to an alternative "large catalog model" where the content catalog size scales with the network size. Under this model, we establish that storage space per server must necessarily grow unboundedly if bandwidth utilization is to be maximized. We then identify a content placement strategy and a request acceptance policy which jointly maximize bandwidth utilization, provided storage space per server grows unboundedly, although arbitrarily slowly, with system size.

For the second application, which is online advertising, we propose a stochastic model to describe how search service providers charge client companies based on users' queries for the keywords related to these companies' ads by using certain advertisement assignment strategies. We formulate an optimization problem to maximize the long-term average revenue for the service provider under each client's long-term average budget constraint, and design an online algorithm which captures the stochastic properties of users' queries and click-through behaviors. We solve the optimization problem by making connections to scheduling

problems in wireless networks, queueing theory and stochastic networks. Unlike prior models, we do not assume that the number of query arrivals is known. Due to the stochastic nature of the arrival process considered here, either temporary "free" service, i.e., service above the specified budget (which we call "overdraft") or under-utilization of the budget (which we call "underdraft") is unavoidable. We prove that our online algorithm can achieve a revenue that is within $O(\epsilon)$ of the optimal revenue while ensuring that the overdraft or underdraft is $O(1/\epsilon)$, where ϵ can be arbitrarily small. With a view towards practice, we can show that one can always operate strictly under the budget. In addition, we extend our results to a click-through rate maximization model, and also show how our algorithm can be modified to handle non-stationary query arrival processes and clients with short-term contracts. Our algorithm also allows us to quantify the effect of errors in click-through rate estimation on the achieved revenue. We show that we lose at most $\frac{\Delta}{1+\Delta}$ fraction of the revenue if Δ is the relative error in click-through rate estimation. We further show that in the long run, an expected overdraft level of $\Omega(\log(1/\epsilon))$ is unavoidable (a universal lower bound) under any stationary ad assignment algorithm which achieves a long-term average revenue within $O(\epsilon)$ of the offline optimum.

To my parents, for their love and support

ACKNOWLEDGMENTS

My deepest gratitude is to my advisor, Prof. R. Srikant, for directing my research towards interesting and important problems, and for providing me with enough guidance and knowledge to conduct research correctly and effectively. He taught me how to start thinking about a big and complex problem from simple examples, how to extract intuitions and key ideas from complicated mathematical proofs, how to tackle a research topic from different perspectives, and how to make excellent presentations, etc. His patience and support helped me overcome many difficulties and finish this dissertation.

I thank Prof. Tamer Başar, Prof. Bruce Hajek and Prof. Venugopal V. Veeravalli for serving on my doctoral committee and providing valuable comments on my dissertation.

I am indebted to my co-authors with whom I had the pleasure of working on different theoretical problems in the field of networks during my graduate study, which either directly or indirectly contribute to this dissertation: Dr. Laurent Massoulié of Technicolor Paris Research Lab, Dr. Jian Ni of IBM T.J. Watson Research Center and Prof. Lei Ying of Iowa State University. Also, I would like to thank my mentors at Microsoft Research Redmond (Dr. Albert Greenberg, Dr. Changhoon Kim, Dr. David Maltz, Dr. Jitendra Padhye and Dr. Murari Sridharan) for providing me with an opportunity to work on a network problem in the system aspect.

I am very thankful to all the faculty members whom I have taken courses with, for their excellent instructions which prepared me with enough fundamental knowledge to complete this dissertation. Many past and current colleagues in the Coordinated Science Laboratory have been very helpful in many ways during my graduate school days; I thank all of them. My sincere thanks also go to all of my friends at UIUC, for kindly accompanying me though these five years and bringing happiness to my life.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Background and Motivations

The architecture of a communication network can be viewed as being composed of five layers with separate functionalities, including the physical layer, the data link layer, the network layer, the transport layer and the application layer [1]. Extensive theoretical research has been done in the wide field of optimization-based network architectures. Taking wireless networking as an example, various researchers have focused on optimizing a certain layer, such as power control (physical layer), link scheduling (MAC layer, i.e., "data link layer" in wireless networks), routing (network layer) and TCP congestion control (transport layer), while others aim at designing cross-layer protocols which jointly maximize the overall performance. See [2, 3, 4, 5] for a comprehensive survey.

Apart from what has been theoretically solved across the lower four layers so far, there are still a lot of interesting topics lying in the uppermost "application layer," arising from many new network applications supported by the increasing communication and computational capabilities. Some topics have already been tackled via system approaches, but still need more insights and designs from a modeling and analytical point of view, in order to achieve potential performance improvement.

One such important network application is the video-on-demand service. In recent years, the amount of multimedia traffic accessed via the Internet, including both live and on-demand streaming, is already of the order of exabytes (10^{18}) per month, and is expected to grow steadily in the coming future. In the centralized hosting model, data centers, where a large number of collocated servers are stacked together, have been built to provide such services. However, this architecture has many limitations such as underutilization, high

energy consumption and relatively high distances from end-users located at the edge of the network [6]. On the other hand, a content distribution network (CDN) architecture, in which video contents are duplicated in servers placed at various nodes of a network, can solve many of these problems, especially reducing the backbone load by localizing the traffic. To build such a CDN, an increasingly popular way for ISPs is to deploy small servers or leverage devices which are already deployed (such as set-top boxes and triple-play gateways) at subscribed users' premises (e.g., NaDa [6] and "People's CDN").[1] Economically, this is also much cheaper than scaling up a data center. Some of the issues in CDN deployment that have not been fully addressed include server placement, service routing and content placement.

Another important network application is online advertising. Providing online advertising services has been the major source of revenue for search service providers such as Google, Yahoo and Microsoft. When an Internet user queries a keyword, alongside the search results, the search engine may also display advertisements from some companies which provide services or goods related to this keyword. These companies (called"clients" in the following text) pay the search service providers for posting their ads with a specified amount of price for each ad on a pay-per-impression or pay-per-click basis.

Maximizing the revenue obtained from their clients is the key objective of search service providers. Research which targets this objective has followed two major directions. One is based on auction theory, in which the goal is to design mechanisms in favor of the service provider, and much of the research in this direction considers static bids (e.g. [7]; see [8] for a survey), while dynamic models such the one in [9] are still emerging. The other is from the perspective of online resource allocation without considering the impact of the service provider's mechanisms on the clients' bids, and the main focus of this kind of research is on designing an online algorithm which posts specific ads in response to each search query arriving online, in order to achieve a high competitive ratio with respect to the offline optimal revenue.

[1]There are many incentive mechanisms for subscribers to support this deployment, such as paying broadband subscription fees for volunteers and sending coupons.

1.2 Contributions

In this dissertation, we propose stochastic models and develop optimal or near-optimal algorithms for resource allocation, for both of the applications mentioned above. We will list our contributions respectively for each application in the following two subsections.

1.2.1 Video-on-Demand Services in CDNs

For the first application "video-on-demand service", we address specifically the content placement problem in CDNs which provide this service. In a CDN with small servers (called "boxes" in the following text) as described above, the critical resources at the boxes are storage space and uplink bandwidth. Our objective is to ensure that the largest fraction of traffic is supported by the CDN. More precisely, we look for content placement strategies that enable content downloaders to maximally use the boxes' uplink bandwidth, and hence maximally offload the servers in the data center. Such strategies must adjust to the distinct popularity of video contents, as a more popular content should be replicated more frequently.

Our main CDN model assumes that requests to download contents come only from external users, and boxes are only service providers (we call this network a "service-only" CDN). We are interested in the performance when the network size becomes very large. In terms of models of content catalog, we study both "finite catalog model" and "large catalog model," respectively, in Chapters 2 and 3.

In Chapter 2, where we assume that the content catalog size does not scale with the network size ("finite catalog model"), we have the following contributions:

- We have proved the optimality of a "Proportional-to-Product" placement strategy. Different from related work in the literature, we build our results upon a classical model called the "loss network."

- A simple demand-driven cache update algorithm based on a reversible Markov chain and a sampling-based preallocation algorithm have been proposed to realize the above strategy under different assumptions. The performances of these algorithms have been studied through extensive simulations.

- We extend our main CDN model to an ISP-managed P2P CDN model, which differs from the previous model in the sense that requests are "internal," i.e., requests emanate from boxes (peers) themselves (refer to Section 2.4 for a detailed description of this model). We show that a "Hot-Warm-Cold" placement strategy is asymptotically optimal by relating its performance to the solution of a linear program.

In Chapter 3, we study a "Large Catalog Model, " in which the content catalog size is assumed to scale with the network size. As far as we know, this model has not been studied in the classical literature on loss networks. By resorting to techniques such as large deviation inequalities (Azuma-Hoeffding and Chernoff) and coupling arguments, we establish that the "Proportional-to-Product" placement strategy (which is optimal under our main model) still leads to a zero loss rate in an underloaded system, provided that the cache size at each box grows unboundedly, although arbitrarily slowly, with the network size.

1.2.2 Online Advertising

For the second application "online advertising," we consider the revenue maximization problem assuming that the contracts between the content provider and the clients are fixed. More precisely, we formulate an optimization problem to maximize the long-term average revenue for the service provider under each client's long-term average budget constraint. Our main contributions are as follows:

- We propose a new stochastic model for online advertising, and design an online algorithm for ad placement which maximizes the long-term average revenue of the service provider.

- Our solution is obtained by making an interesting connection between scheduling in wireless networks and high-speed switches, and the online ad placement problem.

- Due to the stochastic nature of the arrival process considered here, either temporary "free" service, i.e., service above the specified budget (which we call "overdraft") or under-utilization of the budget (which we call "underdraft") is unavoidable. Given

an ϵ, we prove that our online algorithm can achieve a revenue that is within $O(\epsilon)$ of the optimal revenue while ensuring that the overdraft or underdraft is $O(1/\epsilon)$. With a view towards practice, we also show that one can always operate strictly under the budget.

- A key feature of our algorithm compared to previous algorithms in the literature, which also achieve near-optimal performance (in the sense of $1 - O(\epsilon)$ competitiveness), is that our algorithm does not require knowledge about the number of query arrivals. In addition, we extend our results to a click-through rate maximization model, and also show how our algorithm can be modified to handle non-stationary query arrival processes and clients with short-term contracts.

- Our algorithm allows us to quantify the effect of errors in click-through rate estimation on the achieved revenue. We show that we lose at most $\frac{\Delta}{1+\Delta}$ fraction of the revenue if Δ is the relative error in click-through rate estimation.

The above contributions which mainly focus on the design and performance analysis of our ad placement algorithm are all included in Chapter 4. In Chapter 5, we further establish a fundamental result for general ad placement algorithms (which can also be extended to other resource allocation problems with similar formulations): in the long run, an expected overdraft level of $\Omega(\log(1/\epsilon))$ is unavoidable (a universal lower bound) under any stationary ad assignment algorithm which achieves a long-term average revenue within $O(\epsilon)$ of the offline optimum.

CHAPTER 2

CONTENT PLACEMENT IN CDNS FOR VIDEO-ON-DEMAND SERVICES: FINITE CATALOG MODEL

In this chapter, we study optimal content placement strategies in a content distribution network (CDN) providing video-on-demand (VoD) services. We are interested in the performance of a large network, and assume that the content catalog size does not scale with the network size.

We consider the following mode of operation: video requests are first submitted to the CDN which is composed of "boxes"; if they are accepted, uplink bandwidth is used to serve them at the video streaming rate. They are rejected if their acceptance would require disruption of an ongoing request service. Rejected requests are then handled by the data center.

Alternative modes of operation could be envisioned (e.g., enqueueing of requests, service at rates distinct from the streaming rate, joint service by boxes and data center, etc.). The above proposed one, however, is appealing for the following reasons: it ensures zero waiting time for requests, which is desirable for VoD applications. We show that the resulting system can be modeled as a *loss network* [10], for which powerful theoretical results are available, and as our results show, simple placement strategies ensure optimal operation in this model.

2.1 Related Work

The number and location of replicas of distinct content objects in a CDN have a strong impact on such system's performance. Indeed, together with the strategy for handling incoming requests, they determine whether such requests must either be delayed, or served from an alternative, more expensive source such as a remote data center. Requests which cannot start service at once can either be enqueued (we then speak of a waiting model) or

redirected (we then speak of a loss model).

Peer-to-peer (P2P) technology is also increasingly used in building such CDNs to support VoD services (i.e., peers' storage and bandwidth resources are used). The main difference is that requests for downloading contents now emanate from peers (nodes that comprise the CDN) rather than external users. Some of the following works focus on VoD systems in the P2P context, but the basic ideas still apply for a general CDN.

Previous investigations of content placement for distributed VoD systems were conducted by Suh et al. [11] in the context of ISP-managed P2P systems. The problem tackled in [11] differs from our perspective; in particular, no optimization of placement with respect to content popularity was attempted in this work. Performance analyses of both queueing and loss models are considered in [11]. Valancius et al. [6] considered content placement dependent on content popularity, based on a heuristic linear program, and validated this heuristic's performance in a loss model via simulations.

Tewari and Kleinrock [12, 13] advocated to tune the number of replicas in proportion to the request rate of the corresponding content, based on a simple queueing formula, for a waiting model, and also from the standpoint of the load on network links. They further established via simulations that least recently used (LRU) storage management policies at peers emulated rather well their proposed allocation.

Wu et al. [14] considered a loss model, and a specific time-slotted mode of operation whereby requests are submitted to randomly selected peers, who accommodate a randomly selected request. They showed that in this setup the optimal cache update strategy can be expressed as a dynamic program. Through experiments, they established that simple mechanisms such as LRU or least frequently used (LFU) perform close to the optimal strategy they had previously characterized.

Kangasharju et al. [15] addressed file replication in an environment where peers are intermittently available, with the aim of maximizing the probability of a requested file being present at an available peer. This differs from our present focus in that the bandwidth limitation of peers is not taken into account, while the emphasis is on their intermittent presence. They established optimality of content replication in proportion to the *logarithm* of its popularity, and identified simple heuristics approaching this.

Boufkhad et al. [16] considered P2P VoD from yet another viewpoint, looking at the number of contents that can be simultaneously served by a collection of peers.

Content placement problem has also been addressed towards other different optimization objectives. For example, Almeida et al. [17] aim at minimizing total delivery cost in the network, and Zhou et al. [18] target jointly maximizing the average encoding bit rate and average number of content replicas as well as minimizing the communication load imbalance of video servers.

Cache dimensioning problem is considered in [19], where Laoutaris et al. optimized the storage capacity allocation for content distribution networks under a limited total cache storage budget, so as to reduce average fetch distance for the request contents with consideration of load balancing and workload constraints on a given node. We take a different perspective, focusing on many-user asymptotics so the results show that the finite storage capacity per node is never a bottleneck. (Even in the "large catalogue model", it also scales to infinity more slowly than the system size.)

There are obvious similarities between our present objective and the above works. However, none of these identifies explicit content placement strategies at the level of the individual peers, which lead to minimal fraction of redirected (lost) requests in a setup with dynamic arrivals of requests.

Finally, there is a rich literature on loss networks (see in particular Kelly [10]); however our present concern of optimizing placement to minimize the amount of rejected traffic in a corresponding loss network appears new.

2.2 Model Description

We now introduce our mathematical model and related notations. Denote the set of all boxes as $\mathcal{B}$. Let $|\mathcal{B}| = B$ and index the boxes from 1 to B. Box b has a local cache $\mathcal{J}_b$ that can store up to M contents, all boxes having the same storage space M. We further assume that each box can simultaneously serve U concurrent requests, where U is an integer, i.e., each box has an uplink bandwidth equal to U times the video streaming rate. In particular we assume identical streaming rates for all contents.

The set of available contents is defined as $\mathcal{C}$. Let $|\mathcal{C}| = C$ and index contents from 1 to C. Thus a given box b will be able to serve requests for content c for all $c \in \mathcal{J}_b$.

For a new request that needs a download service, an attempt is made to serve this request by some box holding content c, while ensuring that previously accepted requests can themselves be assigned to adequate boxes, given the cache content and bandwidth resources of all boxes. This potentially involves "repacking" of requests, i.e., reallocation of all the bandwidth resources in the system ("box-serving-request" mapping) to accommodate this new download demand pattern. If such repacking can be found, then the request is accepted; otherwise, it is rejected from the CDN.

It will be useful in the sequel to characterize the concurrent numbers of requests that are amenable to such repacking. Let $\mathbf{n} = \{n_c\}_{c \in \mathcal{C}}$ be the vector of numbers n_c of requests per content c. Clearly, a matching of these requests to server boxes is feasible if and only if there exist nonnegative integers z_{cb} (number of concurrent downloads of content c from box b) such that

$$\begin{aligned} \sum_{b: c \in \mathcal{J}_b} z_{cb} &= n_c, \ \forall\ c \in \mathcal{C}; \\ \sum_{c: c \in \mathcal{J}_b} z_{cb} &\leq U, \ \forall\ b \in \mathcal{B}. \end{aligned} \tag{2.1}$$

A more compact characterization of feasibility follows by an application of Hall's theorem [20], giving that $\mathbf{n}$ is feasible if and only if:

$$\forall\ \mathcal{S} \subseteq \mathcal{C}, \ \sum_{c \in \mathcal{S}} n_c \leq U \left| \{ b \in \mathcal{B} : \ \mathcal{S} \cap \mathcal{J}_b \neq \emptyset \} \right|. \tag{2.2}$$

We now introduce statistical assumptions on request arrivals and durations. New requests for content c occur at the instants of a Poisson process with rate ν_c. We assume that the video streaming rate is normalized to 1, and is the same for all contents. We further assume that all videos have the same duration, again normalized at 1. Under these assumptions, the amount of work per time unit brought into the system by content c equals ν_c.

With the above assumptions at hand, assuming fixed cache contents, the vector $\mathbf{n}$ of

requests under service is a particular instance of a general stochastic process known as a loss network model. Loss networks were introduced to represent ongoing calls in telephone networks, and exhibit rich structure. In particular, the corresponding stochastic process is reversible, and admits a closed-form stationary distribution. For our model, the stationary distribution reads:

$$\pi(\mathbf{n}) = \frac{1}{Z} \prod_{c \in \mathcal{C}} \frac{\nu_c^{n_c}}{n_c!} \mathcal{I}_{\{\mathbf{n} \text{ is feasible}\}}. \tag{2.3}$$

In words, the numbers of requests n_c are independent Poisson random variables with parameter ν_c, conditioned on feasibility of the whole vector $\mathbf{n}$.

Our objective is then to determine content placement strategies so that in the corresponding loss network model, the fraction of rejected requests is minimal. The difficulty in doing this analysis resides in the fact that the normalizing constant Z is cumbersome to evaluate. Nevertheless, simplifications occur under large system asymptotics, which we will exploit in the next sections.

We conclude this section by the following remark. For simplicity we assumed in the above description that a particular content is either fully replicated at a peer, or not present at all, and that a request is served from only one peer. It should however be noted that we can equally assume that contents are split into sub-units, which can be placed onto distinct peers, and downloaded from such distinct peers in parallel sub-streams in order to satisfy a request. This extension is detailed in Appendix A.3.

2.3 Optimal Content Placement

We first describe a simple adaptive cache update strategy driven by demand, and show why it converges to a "predetermined" content placement called "proportional-to-product" strategy. We then establish the optimality of this "proportional-to-product" placement in a large system asymptotic regime.

2.3.1 The Proportional-to-Product Placement Strategy

A simple method to adaptively update the caches at boxes driven by demand is described as follows:

Demand-Driven Cache Update

Whenever a new request comes, with probability ϵB (ϵ is chosen such that $\epsilon B \leq 1$), the server picks a box b uniformly at random, and attempts to push content c into this box's cache. If c is already in there, do nothing; otherwise, remove a content selected uniformly at random from the cache.

Since external demands for content c are according to a Poisson process with rate ν_c, we find that under the above simple strategy, content c is pushed at rate $\epsilon\nu_c$ into a particular box which is not caching content c. Recall that each box stores M distinct contents, and let j denote a candidate "cache state", which is a size M subset of the full content set $\mathcal{C}$. For convenience, let $\mathcal{J}$ denote the collection of all such j.

With the above strategy, the caches at each box evolve independently according to a continuous-time Markov process. The rate at which cache state j is changed to j', where $j' = j + \{c\} \setminus \{d\}$ for some contents $d \in j$, $c \notin j$, which we denote by $q(j, j')$, is easily seen to be $q(j, j') = \epsilon\nu_c/M$. Indeed, content d is evicted with probability $1/M$, while content c is introduced at rate $\epsilon\nu_c$.

It is easy to verify that the distribution $p(\cdot)$ given by

$$p(j) = \frac{1}{Z} \prod_{c \in j} \nu_c, \quad j \in \mathcal{J}, \tag{2.4}$$

for some suitable normalizing constant Z, verifies the following equation:

$$p(j)q(j, j') = p(j')q(j', j), \quad j, j' \in \mathcal{J}. \tag{2.5}$$

The latter relations, known as the local balance equations, readily imply that $p(\cdot)$ is a

stationary distribution for the above Markov process; since the process is irreducible, this is the unique stationary distribution.

Thus, we can conclude that under this cache update strategy, the random cache state at any box eventually follows this stationary distribution. This is what we refer to as the **"proportional-to-product" placement strategy**, and it is the one we advocate in the Distributed Server Network scenario.

Remark 1 *The customized parameter ϵ should not be too large, otherwise the burden on the server will be increased due to use of "push". Neither should it be too small, otherwise the Markov chain will converge too slowly to the steady state.* ◇

Under the cache update strategy, the distribution of cache contents needs time to converge to the steady state. However, if we have a priori information about content popularity, we can use a sampling strategy as an alternative way to directly generate proportional-to-product content placement in one go. One method works as follows:

Sampling-Based Preallocation

Select successively M contents at random in an i.i.d. fashion, according to the probability distribution $\{\hat{\nu}_c\}$, where $\hat{\nu}_c = \nu_c / \sum_{c' \in \mathcal{C}} \nu_{c'}$ is the normalized popularity. If there are duplicate selections of some content, re-run the procedure. It is readily seen that this yields a sample with the desired distribution.

An alternative sampling strategy which can be faster than the one described above when very popular items are present is given in the Appendix A.1.

2.3.2 A Loss Network under Many-User Asymptotics

We now consider the asymptotic regime called **"many user–fixed catalogue" scaling**: The number of boxes B goes to infinity. The system load, defined as

$$\rho \triangleq \frac{\sum_{c \in \mathcal{C}} \nu_c}{BU}, \tag{2.6}$$

is assumed to remain fixed, which is achieved in the present section by assuming that the content collection $\mathcal{C}$ is kept fixed, while the individual rates $\{\nu_c\}$ scale linearly with B. We also assume that the normalized content popularities $\{\hat{\nu}_c\}$ remain fixed as B increases. It thus holds that $\nu_c = \hat{\nu}_c \rho B U$ for all $c \in \mathcal{C}$. Note that although boxes are pure resources rather than users, scaling of $\{\nu_c\}$ with B to infinity actually indicates a "many-user" scenario.

To analyze the performance of our proposed proportional-to-product strategy, we require that the cache contents are sampled at random according to this strategy and are subsequently kept fixed. This can either reflect the situation where we use the previously introduced sampling strategy, or alternatively the situation where the cache update strategy has already made the distribution of cache states converge to the steady state, and occurs at a slower time scale than that at which new requests arise and complete.

Note that, as B grows large, the right-hand side in the feasibility constraint (2.2) verifies, by the strong law of large numbers,

$$|\{b \in \mathcal{B} : \ \mathcal{S} \cap \mathcal{J}_b \neq \emptyset\}| \sim B \sum_{j: j \cap S \neq \emptyset} m_j. \tag{2.7}$$

Here, $\{m_j\}$ corresponds to a particular content placement strategy, under which each box holds a size M content set j with probability m_j, and this happens independently over boxes. Specifically, $m_j = \frac{1}{Z} \prod_{c \in j} \hat{\nu}_c$ (where Z is a normalizing constant) corresponds to our proportional-to-product placement strategy.

We now establish a sequence of loss networks indexed by a large parameter B. For the B^{th} loss network, requests for content $c \in \mathcal{C}$ (regarded as "calls of type c") arrive at rate $\nu_c^{(B)} = (\rho U \hat{\nu}_c) \cdot B$, each "virtual link" $\mathcal{S} \subseteq \mathcal{C}$ has a capacity

$$W_{\mathcal{S}}^{(B)} \triangleq (U \sum_{j: j \cap S \neq \emptyset} m_j) \cdot B, \tag{2.8}$$

and $c \in \mathcal{S}$ represents that virtual link $\mathcal{S}$ is part of the "route" which serves call of type c.[1]

[1]Note that this construction in fact admits a form of fixed routing which is equivalently transformed from a dynamic routing model where each particular box is regarded as a link and calls of type c can use any single-link route corresponding to a box holding content c. This equivalent transform is based on the assumption that repacking is allowed (cf. Section 3.3. in [10]). We have already found this equivalent

This particular setup has been identified as the "large capacity network scaling" in Kelly [10]. There, it is shown that the loss probabilities in the limiting regime where $B \to \infty$ can be characterized via the analysis of an associated variational problem.

We now describe the corresponding results in [10] relevant to our present purpose. For the B^{th} loss network, consider the problem of finding the mode of the stationary distribution (2.3), which corresponds to maximizing $\sum_{c\in\mathcal{C}}(n_c^{(B)} \log \nu_c^{(B)} - \log n_c^{(B)}!)$ over feasible $\mathbf{n}^{(B)}$. Then, approximate $\log n_c^{(B)}!$ by $n_c^{(B)} \log n_c^{(B)} - n_c^{(B)}$ according to Stirling's formula and replace the integer vector $\mathbf{n}^{(B)}$ by a real-valued vector $\mathbf{x}^{(B)}$. This leads to the following optimization problem:

[OPT 1]

$$\max_{\mathbf{x}^{(B)}} \quad \sum_{c\in\mathcal{C}}(x_c^{(B)} \log \nu_c^{(B)} - x_c^{(B)} \log x_c^{(B)} + x_c^{(B)}) \tag{2.9}$$

$$s.t. \quad \forall\ \mathcal{S} \subseteq \mathcal{C},\ \sum_{c\in\mathcal{S}} x_c^{(B)} \le W_{\mathcal{S}}^{(B)} \tag{2.10}$$

$$\text{over} \quad \mathbf{x}^{(B)} \ge 0.$$

The corresponding Lagrangian is given by:

$$L(\mathbf{x}^{(B)}, \mathbf{y}^{(B)}) \;=\; \sum_{c\in\mathcal{C}}(x_c^{(B)} \log \nu_c^{(B)} - x_c^{(B)} \log x_c^{(B)} + x_c^{(B)}) + \sum_{\mathcal{S}\subseteq\mathcal{C}} y_{\mathcal{S}}^{(B)}(W_{\mathcal{S}}^{(B)} - \sum_{c\in\mathcal{S}} x_c^{(B)}),$$

where $\{y_{\mathcal{S}}^{(B)}\}_{\mathcal{S}\subseteq\mathcal{C}}$ are Lagrangian multipliers. The KKT conditions for this convex optimization problem comprise the original constraints and the following ones:

$$\bar{y}_{\mathcal{S}}^{(B)}(W_{\mathcal{S}}^{(B)} - \sum_{c\in\mathcal{S}} \bar{x}_c^{(B)}) = 0,\ \bar{y}_{\mathcal{S}}^{(B)} \ge 0,\ \forall\ \mathcal{S} \subseteq \mathcal{C},$$

$$\frac{\partial L(\bar{\mathbf{x}}^{(B)}, \bar{\mathbf{y}}^{(B)})}{\partial x_c^{(B)}} = \log \nu_c^{(B)} - \log \bar{x}_c^{(B)} - \sum_{\mathcal{S}:c\in\mathcal{S}} \bar{y}_{\mathcal{S}}^{(B)} = 0,\ \forall\ c \in \mathcal{C} \tag{2.11}$$

where $(\bar{\mathbf{x}}^{(B)}, \bar{\mathbf{y}}^{(B)})$ is a solution to the optimization problem. From equation (2.11), we further

transform by converting feasibility condition (2.1) to (2.2) in Section 2.2.

get

$$\bar{x}_c^{(B)} = \nu_c^{(B)} \exp(-\sum_{\mathcal{S}:c\in\mathcal{S}} \bar{y}_{\mathcal{S}}^{(B)}),\ \forall\ c \in \mathcal{C}. \tag{2.12}$$

Then the result that we will need from Kelly [10] is the following: for the B^{th} loss network, the steady state probability of accepting request for c, denoted by $A_c^{(B)}$, verifies

$$A_c^{(B)} = \exp\left(-\sum_{\mathcal{S}:c\in\mathcal{S}} \bar{y}_{\mathcal{S}}^{(B)}\right) + O\left(B^{-\frac{1}{2}}\right),\ \forall\ c \in \mathcal{C}, \tag{2.13}$$

where $\bar{y}_{\mathcal{S}}^{(B)}$ are the Lagrangian multipliers of the previous optimization problem.

2.3.3 Optimality of Proportional-to-Product Content Placement

Note that the global acceptance probability, denoted by A_{sys}, which also reads $A_{sys} = \sum_{c\in\mathcal{C}} \hat{\nu}_c A_c$, cannot exceed $\min(1, 1/\rho)$. Indeed, it is clearly no larger than 1. It cannot exceed $1/\rho$ either, otherwise the system would treat more requests than its available resources.

We now prove that the proportional-to-product content placement not only achieves the optimal global acceptance probability $A_{sys} = \min(1, 1/\rho)$, but also achieves fair individual acceptance probabilities, i.e., $A_c = A_{sys}$ for all c. More precisely, we have the following theorem:

Theorem 1 *By using $m_j = \prod_{c\in j} \hat{\nu}_c / Z$ for all $j \subseteq \mathcal{C}$ s.t. $|j| = M$, where Z is the normalizing constant, we have $\lim_{B\to\infty} A_c^{(B)} = \min\{1, 1/\rho\},\ \forall c \in \mathcal{C}$, for fixed ρ and $\mathcal{C}$.* [2] ⋄

Before giving the proof, we comment on the result. One point to note is that because of (2.7), the above optimal acceptance rate is achieved with probability one under any random sampling which follows the proportional-to-product scheme. Secondly, the optimality of the asymptotic acceptance probability does not depend on M, as long as $M \geq 1$. Thus for this particular scaling regime, storage space is not a bottleneck. As we shall see in the next two

[2] Prof. Hajek pointed out that the theorem also holds for any cache update strategy which leads to a steady-state distribution for the cache state. However, it is unclear whether other strategies can also achieve near-optimality in the asymptotic regime considered in the next chapter.

sections, increasing M **does** improve performance if either local services occur, as in the ISP-managed P2P CDN scenario (Section 2.4), or if the catalogue size C scales with the box population size B, a case not covered by the classical literature on loss networks, and to which we turn in Chapter 3.

Proof First, we consider $\rho \geq 1$. Letting

$$\exp\left(-\sum_{\mathcal{S}:c\in\mathcal{S}} \bar{y}_{\mathcal{S}}^{(B)}\right) = 1/\rho,\ \forall c \in \mathcal{C}, \tag{2.14}$$

we have

$$\forall c \in \mathcal{C},\ \sum_{\mathcal{S}:c\in\mathcal{S}} \bar{y}_{\mathcal{S}}^{(B)} = \log\rho. \tag{2.15}$$

Putting equation (2.15) into (2.12) leads to

$$\forall c \in \mathcal{C},\ \bar{x}_c^{(B)} = \nu_c^{(B)}/\rho.$$

Thus, inequality (2.10) in OPT 1 becomes

$$\forall \mathcal{S} \subseteq \mathcal{C},\ \sum_{c\in\mathcal{S}} \nu_c^{(B)} \leq \rho \sum_{j:j\cap\mathcal{S}\neq\emptyset} m_j BU. \tag{2.16}$$

Since $\nu_c^{(B)} = \rho BU \cdot \hat{\nu}_c$ and $\sum_{c\in\mathcal{C}} \hat{\nu}_c = 1$, inequality (2.16) further becomes, upon explicitly writing out the normalization constant Z:

$$\forall \mathcal{S} \subseteq \mathcal{C},\ \sum_{c\in\mathcal{S}} \hat{\nu}_c \cdot \sum_{\mathcal{G}:\ \substack{\mathcal{G}\subseteq\mathcal{C}\\|\mathcal{G}|=M}} \prod_{c\in\mathcal{G}} \hat{\nu}_c \leq \sum_{c\in\mathcal{C}} \hat{\nu}_c \cdot \sum_{\mathcal{G}:\ \substack{\mathcal{G}\cap\mathcal{S}\neq\emptyset\\ \mathcal{G}\subseteq\mathcal{C}\\|\mathcal{G}|=M}} \prod_{c\in\mathcal{G}} \hat{\nu}_c. \tag{2.17}$$

Two types of product terms (mapped to subsets $\mathcal{K} \subseteq \mathcal{C}$) appear on both sides:

I. $\prod_{c\in\mathcal{K}} \hat{\nu}_c$: $|\mathcal{K}| = M+1,\ \mathcal{K}\cap S \neq \emptyset$.

II. $(\prod_{c\in\mathcal{K}} \hat{\nu}_c)\cdot\hat{\nu}_{c'}$: $c' \in \mathcal{K}\cap S,\ |\mathcal{K}| = M$.

To show whether inequality (2.17) holds, we only have to prove that given any $\mathcal{S} \subseteq \mathcal{C}$,

for each product term (related to a $\mathcal{K}$) which appears in one inequality corresponding to a certain $\mathcal{S}$, its multiplicity on the left-hand side is no more than that on the right-hand side.

1. **For a product term of Type I:**

   - On the LHS: Since $\prod_{c\in\mathcal{K}} \hat{\nu}_c = \prod_{c\in\mathcal{G}} \hat{\nu}_c \cdot \hat{\nu}_{c'}$ for some $\mathcal{G} \subseteq \mathcal{C}$ and $c' \in \mathcal{S} \cap \mathcal{K}$, where $\mathcal{G}$ is a size M content set, $c' \notin \mathcal{G}$, and $\mathcal{K} = \mathcal{G} + \{c'\}$. It is easy to see that we have $|\mathcal{S} \cap \mathcal{K}|$ different choice of c' in a $\mathcal{K}$, so the multiplicity of this product term on the LHS equals $|\mathcal{S} \cap \mathcal{K}|$.
   - On the RHS: When $|\mathcal{S} \cap \mathcal{K}| \geq 2$, for any $c' \in \mathcal{K}$, $\mathcal{K} \setminus \{c'\}$ is a size M content set of which the intersect with $\mathcal{S}$ is not empty, hence the multiplicity equals $|\mathcal{K}|$ $(= M + 1)$. When $|\mathcal{S} \cap \mathcal{K}| = 1$, the exception to the above case is that if $c' \in \mathcal{S} \cap \mathcal{K}$, then $\mathcal{K} \setminus \{c'\}$ is a size M content set which has no intersect with $\mathcal{S}$ and is actually impossible to appear in the second summation term (over all size M content sets $\mathcal{G}$ s.t. $\mathcal{G} \cap \mathcal{S} \neq \emptyset$) in inequality (2.17). Thus, the multiplicity equals $|\mathcal{K}| - 1$ $(= M)$.

   From above, we can see that the multiplicity of the product term on the LHS is always no more than that on the RHS.

2. **For a product term of Type II:**
   $\mathcal{K}$ is actually already a size M content set $\mathcal{G}$ s.t. $\mathcal{G} \cap \mathcal{C} \neq \emptyset$. Therefore, it is easy to see that on both sides, the multiplicities of this product term are both 1.

Now we can conclude that inequality (2.17) holds for all $\mathcal{S} \subseteq \mathcal{C}$, and continue to check the complementary slackness. Given $\rho \geq 1$, one simple solution to equation (2.15) reads:

$$\forall\ \mathcal{S} \subseteq \mathcal{C},\ \bar{y}_{\mathcal{S}}^{(B)} = \log \rho \cdot \mathcal{I}_{\{\mathcal{S}=\mathcal{C}\}}. \tag{2.18}$$

Besides, inequality (2.17) is tight for $\mathcal{S} = \mathcal{C}$ (we even do not need to check this when $\rho = 1$). Therefore, complementary slackness is always satisfied with solution (2.18).

So far we have proved that the KKT condition holds when $\rho \geq 1$. When $\rho < 1$, we modify

(2.14) by letting

$$\exp\left(-\sum_{\mathcal{S}:c\in\mathcal{S}}\bar{y}_{\mathcal{S}}^{(B)}\right)=1,\ \forall c\in\mathcal{C}, \tag{2.19}$$

and hence there is an additional factor $1/\rho > 1$ on the RHS of inequality (2.17). Since the old version of inequalities (2.17) is proved to hold, the new version automatically holds, but none of them is tight now. However, from (2.19) we have $\bar{y}_{\mathcal{S}}^{(B)} = 0,\ \forall\ \mathcal{S}\subseteq\mathcal{C}$, which means complementary slackness is always satisfied (similar to $\rho = 1$).

Therefore, according to equation (2.13), it can be concluded that by using $m_j = \prod_{c\in j}\hat{\nu}_c/Z$ for all j, we can achieve

$$A_c^{(B)} = \min\{1, 1/\rho\} + O\left(B^{-\frac{1}{2}}\right),\ \forall c\in\mathcal{C},$$

so $\lim_{B\to\infty} A_c^{(B)} = \min\{1, 1/\rho\}$. ■

2.3.4 Simulation Results

In this subsection, we use extensive simulations to evaluate the performances of the two implementable schemes proposed in Subsection 2.3.1 which follow the "proportional-to-product" placement strategy, namely the sampling-based preallocation scheme and the demand-driven cache update (labeled as **"SAMP"** and **"CU"**, respectively).

We compare the results with the theoretical optimum (i.e., loss rate for each content equals $(1-1/\rho)^+$; the curves are labeled as **"Optimal"**) and a uniform placement strategy (labeled as **"UNIF"**) defined as the following: first, permute all the contents uniformly at random, resulting in a content sequence $\{c_i\}$, for $1\le i\le C$; then, push the M contents indexed by subsequence $\{c_{(j \mod C)}\}_{bM+1\le j\le(b+1)M}$ into the cache of box b, for $1\le b\le B$. UNIF is also used to generate the initial content placement for CU so that the loss rate can be reduced during the warm-up period.

If not further specified, the default parameter setting is as follows: The popularity of

contents $\{\hat{\nu}_c\}$ follows a zipf-like distribution (see e.g. [21]), i.e.,

$$\hat{\nu}_c = \frac{(c_0 + c)^{-\alpha}}{\sum_{c' \in \mathcal{C}} (c_0 + c')^{-\alpha}}, \tag{2.20}$$

with a decaying factor $\alpha > 0$ and the shift $c_0 \geq 0$. We use $\alpha = 0.8$ and $c_0 = 0$. The content catalogue size $C = 500$ and the number of boxes $B = 4000$. Each box can store $M = 10$ contents and serve at most $U = 4$ concurrent requests. The duration of downloading each content is exponentially distributed with mean equal to 1 time unit. The parameter ϵ in the cache update algorithm is set as $1/B$ such that upon a request, one box will definitely be chosen for cache update.

For every algorithm, we take the average over 10 independent repetitive experiments, each of which is observed for 10 time units. According to the sample path, the initial 1/5 of the whole period is regarded as a "warm-up" period and hence ignored in the calculation of final statistics.[3]

Some implementation details are not captured by our theoretical model, but should be considered in simulations. Upon a request arrival, the most idle box (i.e., with the largest number of free connections) among all the boxes which hold the requested content is chosen to provide the service, for the purpose of load balancing. If none of them is idle, we use a heuristic repacking algorithm which iteratively reallocates the ongoing services among boxes, in order to handle as many requests as possible while still respects load balancing. One important parameter which trades off the repacking complexity and the performance is the maximum number of iterations t_r^{max}, which is set as "undefined" by default (i.e., the iterations will continue until the algorithm terminates; theoretically there are at most C iterations). Other details regarding the repacking algorithm can be found in Appendix A.2. We will see an interesting observation about t_r^{max} later.

Figure 2.1 evaluates system loss rates under different traffic loads ρ. Our two algorithms SAMP and CU, which target the proportional-to-product placement, both match the theo-

[3]We can get enough samples during each observation period of 10 time units (for example, when $\rho = 1$, $B = 4000$ and $U = 4$, the average arrivals would be 160000). It has also been checked that after the warm-up period, the distribution of cache states well approximates the proportional-to-product placement and is kept quite stable for the remaining observation period.

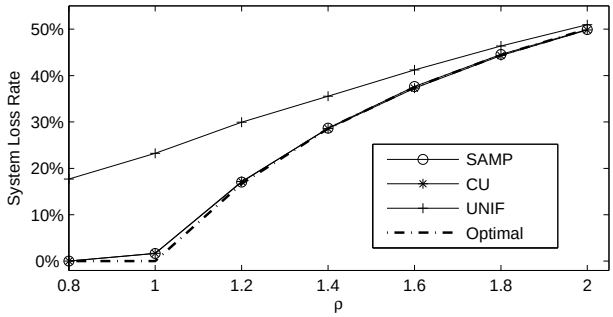


Figure 2.1: System loss rates under different traffic loads

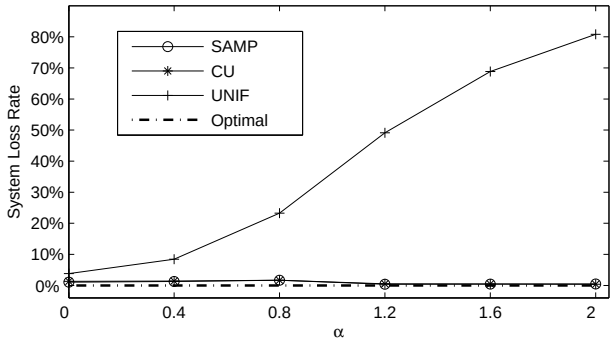


Figure 2.2: System loss rates with different α ($\rho = 1$)

retical optimum very well.[4] On the other hand, the UNIF algorithm, which does not utilize any information about content popularity, incurs a large loss even if the system is underloaded ($\rho < 1$). The gain of proportional-to-product placement over UNIF becomes less significant as the traffic load grows, which can be easily expected.

In Figure 2.2, when the decaying factor α in the zipf-like distribution increases, the distribution of placed contents generated by UNIF has a higher discrepancy from the real content popularity distribution, so UNIF performs worse. On the other hand, the two proportional-to-product strategies are insensitive to the change of content popularity, as we expected.

Figure 2.3 shows the effect of repacking on the system loss rate. In sub-figure (a), we find that under SAMP, repacking is not necessary. In sub-figure (b) which shows the performances of CU, when ρ is low, one iteration of repacking is sufficient to make the performance close enough to the optimum; when ρ is high, repacking also becomes unnecessary. The main

[4]In fact, around $\rho = 1$, they perform a little worse than the optimum. The reason is that $\rho = 1$ is the "critical traffic load" (a separation point between zero-loss and nonzero-loss ranges), under which the simulation results are easier to incur deviation from the theoretical value.

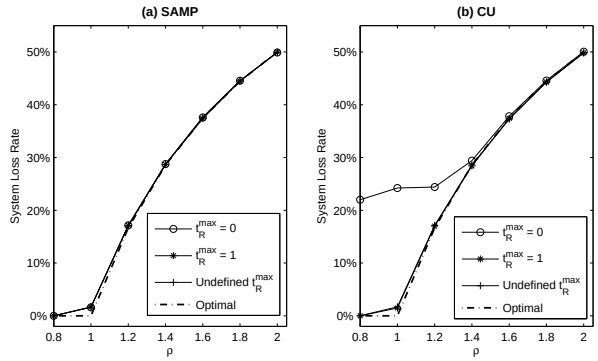


Figure 2.3: Effect of repacking on the system loss rate

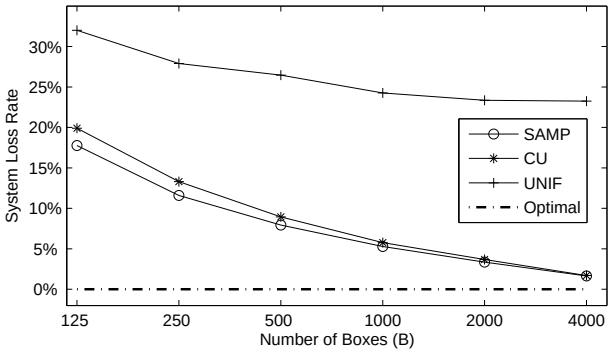


Figure 2.4: System loss rates with different number of boxes

take-away message from this figure is that we can execute a repacking procedure of very small complexity without sacrificing much performance. The reason is that when the server picks a box to serve a request, it already respects the rule of load balancing.

We then explain why CU still needs one iteration of repacking to improve the performance when ρ is low. Note that during the cache update, it is possible that the box is currently uploading the "to-be-kicked-out" content to some users. If repacking is enabled, those ongoing services can be repacked to other boxes (see details in Appendix A.2), but if $t_r^{max} = 0$ (no repacking), they will be terminated and counted as losses. When ρ is high, however, boxes are more likely to be busy, which leads to the failure of repacking, so repacking makes no difference.

Recall that the proportional-to-product placement is only optimal when the number of boxes $B \to \infty$. Figures 2.4 and 2.5 then show the impact of a finite B. In Figure 2.4, as B decreases, the system loss rate of every algorithms increases (compared to the two proportional-

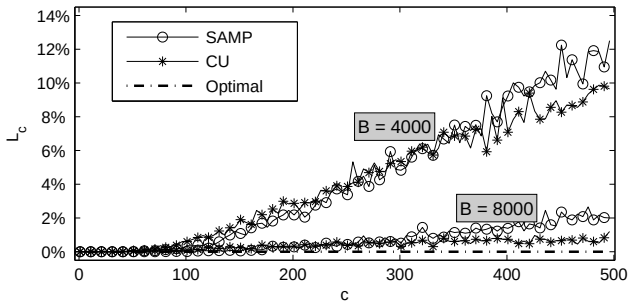


Figure 2.5: Loss rate of requests for each content ($\rho = 1$)

to-product strategies, UNIF is less sensitive to B). In Figure 2.5, non-homogeneity in the individual loss rates of requests for each content also reflects a deviation from the theoretical result (when $B \to \infty$, the loss rates of the requests for all the contents are proved to be identical). As expected, increasing the number of boxes (from 4000 to 8000) makes the system closer to the limiting scenario and the individual loss rates more homogeneous. Another observation is that as the popularity of a content decreases (in the figure, the contents are indexed in the descending order of their popularity), the individual loss rate increases. However, according to Figure 2.1, those less popular contents do not affect the system loss rate much even if they incur high loss, since their weights $\{\hat{\nu}_c\}$ are also lower.

In fact, if we choose a smaller content catalogue size C or a larger cache size M, simulations show the negative impact of a finite B will be reduced (the figures are omitted here). This tells us that if C scales with B rather than being fixed, the proof of optimality under the loss network framework in Subsection 2.3.2 is no longer valid and M must be a bottleneck against the performance of the optimal algorithm. We will solve this problem by introducing a certain type of "large catalogue model" later in Chapter 3.

2.4 Extension: ISP-Managed Peer-to-Peer CDN

In the CDN model we considered in Section 2.2, boxes do not generate requests. On the other hand, we also mentioned that in a peer-to-peer CDN, requests actually come from peers, i.e., those who contribute resources. If a peer has a request for a content which happens to be stored in its own cache, a "local service" will be provided and no bandwidth in the network

will be consumed. In other words, this type of request should be excluded from the traffic caused to a loss network, but it does contribute to the overall acceptance rate. That is why we need to formulate a new optimization problem to solve it.

Instead of a pure autonomous P2P CDN where content distribution cannot be controlled and the network population can be affected by "peer churn," we are going to focus on an ISP-Managed P2P CDN architecture, such as NaDa [6], where contents can be pushed to long-lived peers' boxes by a predefined placement strategy.

In fact, an ISP-managed P2P CDN like NaDa lies somewhere between a pure P2P system and a CDN composed of small servers which never download (i.e., the model in Section 2.2; "People's CDN" is a concrete example). In this architecture, usually gateways deployed at users with fiber connections are consistently put into contribution as servers, which can serve requests from users with a slower uplink (say DSL) and yet may not themselves generate massive amounts of requests. We can think of the case that requests are purely external and the case that requests are purely internal as two extreme cases in this architecture.

In the ISP-Managed P2P Network scenario, when box b has a request for content c which is currently in its own cache, a "local service" will be provided and no download bandwidth in the network will be consumed. To simplify our analysis, each request for a specific content is assumed to originate from a box chosen uniformly at random (this in particular assumes identical tastes of all users).

This means that the effective arrival rate of the requests for content c which generates traffic load actually equals $\tilde{\nu}_c \triangleq \nu_c(1-\tilde{m}_c)$, where $\tilde{m}_c$ is defined as the fraction of boxes who have cached content c. Let $\rho_c \triangleq \rho\hat{\nu}_c$ denote the traffic load generated by requests for content c, and λ_c denote the fraction of the system bandwidth resources used to serve requests for content c. Obviously, $\sum_{c\in\mathcal{C}} \lambda_c \leq 1$. The traffic load absorbed by the P2P system either via local services or via service from another box is then upper-bounded by

$$\tilde{\rho} = \sum_{c\in\mathcal{C}} \rho_c\tilde{m}_c + [\rho_c(1-\tilde{m}_c)] \wedge \lambda_c, \tag{2.21}$$

where "$\wedge$" denotes the minimum operator.

We will use this simple upper bound to identify an optimal placement strategy in the

present ISP-Managed P2P Network scenario. To this end, we shall establish that our candidate placement strategy asymptotically achieves this performance bound, namely absorbs a portion $\tilde{\rho}$ in the limit where B tends to infinity.

To find the optimal strategy, we introduce a variable $x_c \triangleq [\rho_c(1-\tilde{m}_c)] \wedge \lambda_c$ for all c. Note further that the fraction λ_c is necessarily bounded from above by $\tilde{m}_c$, as only those boxes holding c can devote their bandwidth to serving c. It is then easy to see that the quantity $\tilde{\rho}$ in (2.21) is no larger than the optimal value of the following linear program:

[OPT 2]

$$\begin{aligned} \max_{\tilde{\mathbf{m}},\boldsymbol{\lambda},\mathbf{x}} \quad & \sum_{c\in\mathcal{C}}(\rho_c\tilde{m}_c + x_c) \\ s.t. \quad & \forall\ c\in\mathcal{C},\ 0\le\tilde{m}_c\le 1,\ 0\le\lambda_c\le\tilde{m}_c; \\ & \forall\ c\in\mathcal{C},\ 0\le x_c\le\lambda_c,\ x_c\le\rho_c(1-\tilde{m}_c); \\ & \sum_{c\in\mathcal{C}}\tilde{m}_c = M,\ \sum_{c\in\mathcal{C}}\lambda_c\le 1. \end{aligned}$$

The following theorem gives the structure of an optimal solution to OPT 2, and as a result suggests an optimal placement strategy.

Theorem 2 *Assume that $\{\hat{\nu}_c\}$ are ranked in descending order. The following solution solves OPT 2:*

- *For $1\le c\le M-1$, $\tilde{m}_c = 1$, $\lambda_c = x_c = 0$.*
- *For $M\le c\le c^*$, $\tilde{m}_c = \lambda_c = x_c = \rho_c/(1+\rho_c)$, where c^* satisfies that*

$$\sum_{c=M}^{c^*}\frac{\rho_c}{1+\rho_c}\le 1,\ \textit{but}\ \sum_{c=M}^{c^*+1}\frac{\rho_c}{1+\rho_c} > 1.$$

- *For $c = c^*+1$, $\tilde{m}_c = \lambda_c = x_c = 1-\sum_{c=M}^{c^*}\tilde{m}_c$.*
- *For $c^*+2\le c\le C$, $\tilde{m}_c = \lambda_c = x_c = 0$.* ◇

The proof consists in checking that the KKT conditions are met for the above candidate solution. Details are given in Appendix B.2.

The above optimal solution suggests the following placement strategy:

"Hot-Warm-Cold" Content Placement Strategy

Divide the contents into three different classes according to their popularity ranking (in descending order):

- **Hot:** The $M-1$ most popular contents. At each box, $M-1$ cache slots are reserved for them to make sure that requests for these contents are always met via local service.
- **Warm:** The contents with indices from M to c^*+1 (or c^* if $\sum_{c=M}^{c^*} \tilde{m}_c = 1$). For these contents, a fraction $\tilde{m}_c$ of all the boxes will store content c in their remaining one cache slots, where the value of $\tilde{m}_c$ is given in Theorem 2. All requests for these contents (except c^*+1 if it is classified as "warm") can be served, at the expense of all bandwidth resources.
- **Cold:** The other less popular contents are not cached at all.

Remark 2 *The requests for the c^* most popular contents ("hot" contents and "warm" contents except content c^*+1) incur zero loss, while the requests for the $C-c^*-1$ least popular contents incur* 100% *loss. There is a partial loss in the requests for content c^*+1 if $\sum_{c=M}^{c^*} \tilde{m}_c < 1$.* ◇

Under this placement strategy, the maximum upper bound on the absorbed traffic load reads

$$\tilde{\rho} = \sum_{c=1}^{c^*} \rho_c + (\rho_{c^*+1}+1)\left(1 - \sum_{c=M}^{c^*} \frac{\rho_c}{1+\rho_c}\right).$$

We then have the following corollary:

Corollary 1 *Considering the large system limit $B \to \infty$, with fixed catalogue and associated normalized popularities $\{\hat{\nu}_c\}$ as considered in Subsection 2.3.2, the proposed "hot-warm-cold"*

placement strategy achieves an asymptotic fraction of absorbed load equal to the above upper bound $\tilde{\rho}$, and is hence optimal in this sense. $\diamond$

Proof With the proposed placement strategy, hot (respectively, cold) contents never trigger accepted requests, since all incoming requests are handled by local service (respectively, rejected). For warm contents, because each box holds only one warm content, it can only handle requests for that particular warm content. As a result, the processes of ongoing requests for distinct warm contents evolve independently of one another. For a given warm content c, the corresponding number of ongoing requests behaves as a simple one-dimensional loss network with arrival rate $\nu_c(1-\tilde{m}_c)$ and service capacity $\tilde{m}_c BU$. For $c = M, \ldots, c^*$, one has $\tilde{m}_c = \rho_c/(1+\rho_c)$ where $\rho_c = \nu_c/(BU)$, so both the arrival rate and the capacity of the corresponding loss network equal $\tilde{m}_c BU$. The asymptotic acceptance probability as $B \to \infty$ then converges to 1 and the accepted load due to both local service and services from other boxes converges to ρ_c. For content c^*+1 (if $\tilde{m}_{c^*+1} > 0$), the corresponding loss network has arrival rate $\nu_{c^*+1}(1-\tilde{m}_{c^*+1})$ and service capacity $\tilde{m}_{c^*+1}BU$. Then, in the limit $B \to \infty$, the accepted load (due to both local services and services from other boxes) reads $\rho_{c^*+1}\tilde{m}_{c^*+1} + \tilde{m}_{c^*+1}$ (which is actually smaller than ρ_{c^*+1}). Summing the accepted loads of all contents yields the result. ■

CHAPTER 3

CONTENT PLACEMENT IN CDNS FOR VIDEO-ON-DEMAND SERVICES: LARGE CATALOG MODEL

From the simulation results shown in Section 2.3.4, if the number of boxes is not large enough compared to the content catalog size, then the performance of the proportional-to-product placement is not as good as the theoretical optimum. This motivates us to seek another class of models in which content catalog size scales with the number of boxes rather than being fixed, which we term "large catalog models." These types of models do not seem to have been studied in the classical literature on loss networks.

3.1 Model Description

Throughout this chapter, we consider a specific "large catalogue model": The set of contents $\mathcal{C}$ is divided into a fixed number of "content classes", indexed by $i \in \mathcal{I}$. In class i, all the contents have the same popularity (arrival rate) ν_i. The number of contents within class i is assumed to scale in proportion to the number of boxes B, i.e., class i contains $\alpha_i B$ contents for some fixed scaling factor α_i. We further define $\alpha \triangleq \sum_i \alpha_i$.

With the above assumptions, the system traffic load ρ in equation (2.6) reads

$$\rho = \frac{1}{U} \sum_{i \in \mathcal{I}} \alpha_i \nu_i. \tag{3.1}$$

This model is mathematically convenient: by limiting the number of popularity values we limit the "dimensionality" of the request distribution, even though we now allow for a growing number of contents. The model is an approximation, that would result from batching into a single class all contents with a comparable popularity. Such classes can also capture the movie type (e.g. thriller, comedy) and age (assuming popularity decreases with content age).

We use $\hat{\upsilon}_i$ to denote the normalized popularity of content class $i \in \mathcal{I}$ and it satisfies $\sum_{i\in\mathcal{I}} \hat{\upsilon}_i = 1$. It is reasonable to regard each $\hat{\upsilon}_i$ as fixed. The quantity $\hat{\nu}_i \triangleq \hat{\upsilon}_i/(\alpha_i B)$ represents the normalized popularity of a specific content in class i, which decreases as the number of contents in this class $\alpha_i B$ increases, since users now have more choices within each class. In practice, an online video provider company which uses this CDN adds both boxes and available movies of each type to attract more user traffic, under a constraint of a maximum tolerable traffic load ρ.

We will focus on the case with only external requests and consider the following questions: What amount of storage is required to ensure that memory space is not a bottleneck? Furthermore, if this storage requirement is satisfied, is the proportional-to-product placement strategy still optimal under the large-catalogue scaling?

3.2 Necessity of Unbounded Storage

We first establish that bounded storage will strictly constrain utilization of bandwidth resources. To this end we need the following lemma:

Lemma 1 *Consider the system under large catalogue scaling, with fixed weights α_i and cache size M per box. Define $M' \triangleq \lceil 2M/\alpha \rceil$. Then*

(i) More than half of the contents are replicated at most M' times, and

(ii) For each of these contents, the loss probability is at least $E(\inf_i \nu_i, M'U) > 0$, where $E(\cdot,\cdot)$ is the Erlang function [10] defined as:

$$E(\nu, C) \triangleq \frac{\nu^C}{C!}\left[\sum_{n=1}^{C}\frac{\nu^n}{n!}\right]^{-1}.$$

◇

Proof We first prove part (i). Note that the total number of content replicas in the system equals BM. Thus, denoting by f the fraction of contents replicated at least $M'+1$ times, it

follows that $f\alpha B(M'+1) \leq BM$, which in turn yields

$$f \leq \frac{M}{\alpha\left(\lceil 2M/\alpha \rceil + 1\right)} \leq \frac{M}{2M+\alpha} < \frac{1}{2},$$

which implies statement (i).

To prove part (ii), we establish the following general property for a loss network (equivalent to our original system) with call types $j \in \mathcal{J}$, corresponding arrival rates ν_j, and capacity (maximal number of competing calls) C_l on link ℓ for all $\ell \in \mathcal{L}$. We use $\ell \in j$ to indicate that the route for calls of type j comprises link ℓ. Denoting the loss probability of calls of type j in such a loss network as p_j, we then want to prove

$$p_j \geq E(\nu_j, C'_j), \tag{3.2}$$

where $C'_j \triangleq \min_{\ell \in j} C_\ell$, i.e., the capacity of the bottleneck link on the route for calls of type j.

Note that the RHS of the above inequality is actually the loss probability of a loss network with only calls of type j and capacity C'_j. Fixing index j, we define this loss network as an auxiliary system and consider the following coupling construction which allows us to deduce inequality (3.2): let X_k be the number of active calls of type k in the original system for all k, and let X'_j denote the number of active calls of type j in the auxiliary system. Initially, $X_j(0) = X'_j(0)$. The non-zero transition rates for the joint process $(\{X_k\}_{k\in K}, X'_j)$ are given by

$$\begin{aligned}
&k \neq j:\ X_k \to X_k + 1 && \text{at rate } \nu_k \prod_{\ell \in j} \mathcal{I}_{\{\sum_{k \ni \ell} X_k < C_\ell\}},\\
&k \neq j:\ X_k \to X_k - 1 && \text{at rate } X_k,\\
&(X_j, X'_j) \to (X_j + 1, X'_j + 1) && \text{at rate } \nu_j^{both},\\
&(X_j, X'_j) \to (X_j + 1, X'_j) && \text{at rate } \nu_j^{ori},\\
&(X_j, X'_j) \to (X_j, X'_j + 1) && \text{at rate } \nu_j^{aux},\\
&(X_j, X'_j) \to (X_j - 1, X'_j - 1) && \text{at rate } X_j,\\
&(X_j, X'_j) \to (X_j, X'_j - 1) && \text{at rate } \left[X'_j - X_j\right]^+,
\end{aligned}$$

where

$$\begin{aligned}
\nu_j^{both} &\triangleq \nu_j \mathcal{I}_{\{X'_j < C'_j\}} \cdot \prod_{\ell \in j} \mathcal{I}_{\{\sum_{k \ni \ell} X_k < C_\ell\}}, \\
\nu_j^{ori} &\triangleq \nu_j \mathcal{I}_{\{X'_j = C'_j\}} \cdot \prod_{\ell \in j} \mathcal{I}_{\{\sum_{k \ni \ell} X_k < C_\ell\}}, \\
\nu_j^{aux} &\triangleq \nu_j \mathcal{I}_{\{X'_j < C'_j\}} \cdot \mathcal{I}_{\{\exists \ell \in j \text{ s.t. } \sum_{k \in \ell} X_k = C_\ell\}}.
\end{aligned}$$

It follows from Theorem 8.4 in [22] that $\{X_k\}$ is indeed a loss network process with the original dynamics, and that X'_j is a one-dimensional loss network with capacity C'_j and arrival rate ν_j. From the construction, we can see that all transitions preserve the inequality $X_j(t) \leq X'_j(t)$ for all $t \geq 0$, due to the following reason: once X_j increases by 1, X'_j either increases by 1 or equals the capacity limit C'_j, and for the latter case, the corresponding transition rate ν_j^{ori} implies that $X_j \leq C'_j = X'_j$. Similarly, once X'_j decreases by 1, either X_j also decreases by 1, or in the case that X_j does not decrease, it must be that the transition rate $X'_j - X_j$ is strictly positive. In any case, the above inequality is preserved.

We further let $A_j(t)$, $A'_j(t)$ denote the number of type j external calls, $L_j(t)$, $L'_j(t)$ the number of type j call rejections, and $D_j(t)$, $D'_j(t)$ the number of type j call completions, respectively in the original and auxiliary systems, during time interval $[0, t]$. It follows from our construction that whenever the service for a call of type j completes in the original system, the service for a call of type j also completes in the auxiliary system, hence $D_j(t) \leq D'_j(t)$ for all $t \geq 0$. Since $X_j(t) = A_j(t) - D_j(t) - L_j(t)$, $X'_j(t) = A'_j(t) - D'_j(t) - L'_j(t)$ and $A_j(t) = A'_j(t)$, we have $L_j(t) \geq L'_j(t)$. Upon dividing this inequality by $A(t)$ and letting t tend to infinity, one retrieves the announced inequality (3.2) by the ergodic theorem.

Back to the context of our CDN system, for those contents which are replicated at most M' times (i.e., the contents considered in part (i)), the rejection rate of content c of type j reads $p_j \geq E(\inf_i \nu_i, C'_j) \geq E(\inf_i \nu_i, M'U)$. ■

The above lemma readily implies the following corollary:

Corollary 2 *Under the assumptions in Lemma 1, the overall rejection probability is at least* $\frac{1}{2} E(\min_i \nu_i, M'U)$. *Indeed, for bounded* M, M' *is also bounded, and* $E(\min_i \nu_i, M'U)$ *is*

bounded away from 0. ◇

Thus, even when the system load ρ is strictly less than 1, with bounded M there is a non-vanishing fraction of rejected requests, hence a suboptimal use of bandwidth.

3.3 Efficiency of Proportional-to-Product Placement

3.3.1 Modified Proportional-to-Product Placement Strategy and Counter-Based Acceptance Rule

We consider the following **"Modified Proportional-to-Product Placement"**: each of the M storage slots at a given box b contains a randomly chosen content. The probability of selecting one particular content c is $\nu_i/(\rho BU)$ if it belongs to class i. In addition, we assume that the selections for all such MB storage slots are done independently of one another.

Remark 3 *This content placement strategy can be viewed as a "balls-and-bins" experiment. All the MB cache slots in the system are regarded as balls, and all the* $|\mathcal{C}|$ $(= \sum_i \alpha_i B)$ *contents are regarded as bins. We throw each of the MB balls at random among all the* $|\mathcal{C}|$ *bins. Bin c (corresponding to content c which belongs to class i) will be chosen with probability* $\nu_i/(\rho BU)$*. Alternatively, the resulting allocation can be viewed as a bipartite random graph connecting boxes to contents.* ◇

Note that this strategy differs from the "proportional-to-product" placement strategy proposed in Section 2.3, in that it allows for multiple copies of the same content at the same box. However, by the birthday paradox, we can prove the following lemma which shows that up to a negligible fraction of boxes, the above content placement does coincide with the proportional-to-product strategy.

Lemma 2 *By using the above content placement strategy, if* $M \ll \sqrt{(\min_i \alpha_i)B}$ *at a certain box, we have*

$$\Pr(\textit{all the } M \textit{ cached contents are different}) \approx 1. \tag{3.3}$$

$\diamond$

Proof In the birthday paradox, if there are m people and n equally possible birthdays, the probability that all the m people have different birthdays is close to 1 whenever $m \ll \sqrt{n}$. Here in our problem, at a certain box, the M cache slots are regarded as "people" and the $|\mathcal{C}|$ contents are regarded as "birthdays." Although the probability of picking one content is non-uniform, the probability of picking one content within a specific class is uniform. One can think of picking a content for a cache slot as a two-step process: With probability $\alpha_i\nu_i / \sum_j \alpha_j\nu_j$, a content in class i is chosen. Then conditioned on class i, a specific content is chosen uniformly at random among all the $\alpha_i B$ contents in class i.

Contents from different classes are obviously different. When $M \ll \sqrt{\alpha_i B}$, even if all the M cached contents are from class i, the probability that they are different is close to 1. Thus, $M \ll \sqrt{\min_i \alpha_i B}$ is sufficient for (3.3) to hold. ■

To prove that under this particular placement, inefficiency in bandwidth utilization vanishes as $M \to \infty$, we shall in fact consider a slight modification of the "request repacking" strategy considered so far for determining which contents to accept:

Counter-Based Acceptance Rule

A parameter $L > 0$ is fixed. Each box b maintains at all times a counter Z_b of associated requests. For any content c, the following procedure is used by the server whenever a request arrives: A random set of L distinct boxes, each of which holds a replica of content c, is selected. An attempt is made to associate the newly arrived request with all L boxes, but the request will be rejected if its acceptance would lead any of the corresponding box counters to exceed LU.

Remark 4 *Note that in this acceptance rule, associating a request to a set of L boxes does not mean that the requested content will be downloaded from all these L boxes. In fact, as before, the download stream will only come from one of the L boxes, but here we do not specify which one is to be picked.*

It is readily seen that the above rule defines a loss network. Moreover, it is a stricter acceptance rule than the previously considered one. Indeed, it can be verified that when all ongoing requests have an associated set of L boxes, whose counters are no larger than LU, there exist nonnegative integers Z_{cb} *such that* $\sum_{b:c\in\mathcal{J}_b} Z_{cb} = Ln_c,\ \forall\ c \in \mathcal{C}$ *and* $\sum_{c:c\in\mathcal{J}_b} Z_{cb} \leq LU,\ \forall\ b \in \mathcal{B}$*, then feasibility condition* (2.2) *holds a fortiori.* ◇

3.3.2 Asymptotically Zero Loss Rate in an Underloaded System

We introduce an additional assumption, needed for technical reasons.

Assumption 1 *A content which is too poorly replicated is never served. Specifically,* ***a content must be replicated at least*** $\boldsymbol{M^{3/4}}$ ***times to be eligible for service.*** ◇

Our main result in this context is the following theorem:

Theorem 3 *Consider fixed M,* α_i*,* ν_i*, and corresponding load* $\rho < 1$*. Then for suitable choice of parameter L, with high probability (with respect to placement) as* $B \to \infty$*, the loss network with the above "modified proportional-to-product placement" and "counter-based acceptance rule" admits a content rejection probability* $\phi(M)$ *for some function* $\phi(M)$ *decreasing to zero as* $M \to \infty$*.* ◇

The interpretation of this theorem is as follows: The fraction of lost service opportunities, for an underloaded system ($\rho < 1$), vanishes as M increases. Thus, while Corollary 2 showed that $M \to \infty$ is necessary for optimal performance, this theorem shows that it is also sufficient: there is no need for a minimal speed (e.g. $M \geq \log B$) to ensure that the loss rate becomes negligible.

Proof The proof has five sequential stages:

1) The chance for a content to be "good:"

Let N_c denote the number of replicas of content c of class i. Then, N_c admits a binomial distribution with parameters $(MB, \frac{\nu_i}{\rho BU})$. We call content c a "good" content if $|N_c - \mathbb{E}[N_c]| < M^{2/3}$, i.e.,

$$\left| N_c - \frac{\nu_i M}{\rho U} \right| < M^{2/3}. \tag{3.4}$$

As $N_c = \sum_{i=1}^{MB} Z_i$, where $Z_i \sim Ber(p)$ ($p \triangleq \frac{\nu_i}{\rho BU}$) are i.i.d., according to the Chernoff bound,

$$\Pr\left(N_c \geq M^{2/3} + \frac{\nu_i M}{\rho U} \right) \leq e^{-MB \cdot I(a)}, \tag{3.5}$$

where $a \triangleq \left(M^{2/3} + \frac{\nu_i M}{\rho U} \right) / MB$ and $I(x) \triangleq \sup_\theta \{ x\theta - \ln(\mathbb{E}[e^{\theta Z_i}]) \}$ is the Cramér transform of the Bernoulli random variable Z_i. Instead of directly deriving the RHS of inequality (3.5), which can be done but needs a lot of calculations (see Appendix B.3), we upper bound it by using a much simpler approach here: For the same deviation, a classical upper bound on the Chernoff bound of a binomial random variable is provided by the Chernoff bound of a Poisson random variable which has the same mean (see e.g. [22]). Therefore, the RHS of inequality (3.5) can be upper bounded by

$$\exp\left(-\frac{\nu_i M}{\rho U} \cdot \hat{I}\left(1 + \frac{\rho U}{\nu_i M^{1/3}} \right) \right),$$

where $\hat{I}(x)$ is the Cramér transform of a unit mean Poisson random variable, i.e., $\hat{I}(x) = x \log x - x + 1$. By Taylor's expansion of $\hat{I}(x)$ at $x = 1$, the exponent in the last expression is equivalent to

$$\begin{aligned} & -\frac{\nu_i M}{\rho U} \cdot \left(\frac{1}{2} \left(\frac{\rho U}{\nu_i M^{1/3}} \right)^2 + o\left(M^{-2/3} \right) \right) \\ = \; & -\frac{\rho U}{2\nu_i} M^{1/3} + o\left(M^{1/3} \right) = -\Theta\left(M^{1/3} \right). \end{aligned}$$

On the other hand, when M is large, $-M^{2/3} + \frac{\nu_i M}{\rho U} \geq 0$ holds, hence we have

$$\begin{aligned} & \Pr\left(N_c \leq -M^{2/3} + \frac{\nu_i M}{\rho U}\right) \\ = \ & \Pr\left(\sum_{i=1}^{MB} \hat{Z}_i \geq MB \cdot \hat{a}\right) \leq e^{-MB \cdot \hat{I}(\hat{a})}, \end{aligned} \tag{3.6}$$

where $(-\hat{Z}_i) \sim Ber(p)$, $\hat{a} \triangleq M^{-1/3}/B - p \in [-1, 0]$ when B is large, and it is easy to check that $\hat{I}(\hat{a}) = I(-\hat{a})$. Similarly as above by upper bounding $e^{-MB \cdot I(-\hat{a})}$, we can find that the exponent of the upper bound is also $-\Theta\left(M^{1/3}\right)$. Therefore,

$$\Pr(\text{content } c \text{ is good}) \geq 1 - 2e^{-\Theta(M^{1/3})}. \tag{3.7}$$

2) The number of "good contents" in each class:

Denoting by X_i the number of good contents in class i, we want to use a corollary of Azuma-Hoeffding inequality (see e.g. Section 12.5.1 in [23] or Corollary 6.4 in [22]) to upper bound the chance of its deviation from its mean. This corollary applies to a function f of independent variables $\xi_1, \ldots, \xi_n$, and states that if the function changes by an amount no more than some constant c when only one component ξ_i has its value changed, then for all $t > 0$,

$$\Pr(|f(\boldsymbol{\xi}) - \mathbb{E}[f(\boldsymbol{\xi})]| \geq t) \leq 2e^{-2t^2/(nc^2)}.$$

Back to our problem, each independent variable ξ_j corresponds to the choice of a content to be placed in a particular memory slot at a particular box (we index a slot by j for $1 \leq j \leq MB$), and $f(\boldsymbol{\xi})$ corresponds to the number of good contents in class i based on the placement $\boldsymbol{\xi}$, i.e., $X_i = f(\boldsymbol{\xi})$. It is easy to see that in our case $c = 1$, hence we have

$$\Pr(|X_i - \mathbb{E}[X_i]| \geq t) \leq 2e^{-2t^2/(MB)}, \ \forall t > 0.$$

Taking $t = (MB)^{2/3}$ in the above inequality further yields

$$\Pr\left(|X_i - \mathbb{E}[X_i]| \geq (MB)^{2/3}\right) \leq 2e^{-2(MB)^{1/3}}.$$

Thus, we have

$$\begin{aligned} & \Pr\left(X_i \geq \left(1-2e^{-\Theta(M^{1/3})}\right)\cdot \alpha_i B-(MB)^{2/3}\right) \\ \overset{(a)}{\geq} & \Pr\left(X_i \geq \mathbb{E}[X_i]-(MB)^{2/3}\right) \\ \geq & \Pr\left(|X_i-\mathbb{E}[X_i]|<(MB)^{2/3}\right) \\ \geq & 1-2e^{-2(MB)^{1/3}}, \end{aligned} \tag{3.8}$$

where (a) holds since

$$\begin{aligned} \mathbb{E}[X_i] &= \Pr(\text{content } c \text{ is good})\cdot \alpha_i B \\ &\geq \left(1-2e^{-\Theta(M^{1/3})}\right)\cdot \alpha_i B. \end{aligned}$$

Note that in order for the lower bound on X_i shown in the above probability to be $\Theta(B)$, $M \sim o(B^{1/2})$ is a sufficient condition.

3) The chance for a box to be "good:"

We call a replica "good" if it is a replica of a good content, and use C_i to denote the number of good replicas of class i. We also call a box "good" if the number of good replicas of class i held by this box lies within

$$\frac{\alpha_i \nu_i M}{\rho U} \pm O(M^{2/3}).$$

As we did for "good contents," we will also use the Chernoff bound to prove that a box is good with high probability.

Let $\mathcal{E}_i$ represent an event that the number X_i of good contents within class i satisfies

$$X_i \geq \left(1-2e^{-\Theta(M^{1/3})}\right)\alpha_i B-(MB)^{2/3}, \tag{3.9}$$

which has a probability of at least $1-2e^{-\Omega((MB)^{1/3})}$, according to inequality (3.8) when $M \sim o(B^{1/2})$. Conditional on $\mathcal{E}_i$, according to the lower bound in inequality (3.4) (i.e., the

definition of "good contents") and inequality (3.9), we have

$$\begin{aligned} C_i &\geq \left(\frac{\nu_i M}{\rho U} - M^{2/3}\right)\left(\left(1 - 2e^{-\Theta(M^{1/3})}\right)\alpha_i B \right. \\ &\quad \left. -(MB)^{2/3}\right) \\ &= MB \cdot \frac{\alpha_i \nu_i}{\rho U}\left(1 - O(M^{-1/3} + M^{2/3}B^{-1/3})\right). \end{aligned} \tag{3.10}$$

On the other hand, from the upper bound in inequality (3.4) and the fact $X_i \leq \alpha_i B$, we obtain that

$$C_i \leq MB \cdot \frac{\alpha_i \nu_i}{\rho U}\left(1 + O(M^{-1/3})\right). \tag{3.11}$$

Conditional on $\mathcal{E}_i$, to constitute a box, sample without replacement from the determined content replicas. Denote the number of good replicas of class i stored in a particular box (say, box b) by ζ_i, which actually represents the number of good replicas in the M samples sampled without replacement from all the MB replicas, among which C_i are good ones (conditional on $\mathcal{E}_i$). This means that, conditional on $\mathcal{E}_i$, ζ_i follows a hypergeometric distribution $H(MB, C_i, M)$. It can be found that (see e.g. Theorem 1 in [24]) conditional on $\mathcal{E}_i$, $H_i \leq_{st} \zeta_i \leq_{st} G_i$. Here, "$\leq_{st}$" represents stochastic ordering, and

$$\begin{aligned} G_i &\sim \text{Bin}\left(M, \frac{\alpha_i \nu_i}{\rho U}\left(1 + O(M^{-1/3})\right)\right), \\ H_i &\sim \text{Bin}\left(M, \frac{\alpha_i \nu_i}{\rho U}\left(1 - O(M^{-1/3} + M^{2/3}B^{-1/3})\right)\right), \end{aligned}$$

where the second parameters of the distributions of G_i and H_i are determined according to inequalities (3.11) and (3.10) respectively.

We will see why we need these two "binomial bounds" on ζ_i. By definition,

$$\begin{aligned}
& \Pr(\text{box } b \text{ is not good}) \\
= \; & \Pr\left(\bigcup_{i\in\mathcal{I}}\left\{\left|\zeta_i - \frac{\alpha_i\nu_i M}{\rho U}\right| \geq O(M^{2/3})\right\}\right) \\
\leq \; & \sum_{i\in\mathcal{I}} \Pr\left(\left|\zeta_i - \frac{\alpha_i\nu_i M}{\rho U}\right| \geq O(M^{2/3})\right), \qquad (3.12)
\end{aligned}$$

where for all $i \in \mathcal{I}$,

$$\begin{aligned}
& \Pr\left(\left|\zeta_i - \frac{\alpha_i\nu_i M}{\rho U}\right| \geq O(M^{2/3})\right) \\
= \; & \Pr\left(\left|\zeta_i - \frac{\alpha_i\nu_i M}{\rho U}\right| \geq O(M^{2/3}),\ \mathcal{E}_i\right) \\
& + \Pr\left(\left|\zeta_i - \frac{\alpha_i\nu_i M}{\rho U}\right| \geq O(M^{2/3}),\ \mathcal{E}_i^c\right) \\
\leq \; & \Pr\left(\left|\zeta_i - \frac{\alpha_i\nu_i M}{\rho U}\right| \geq O(M^{2/3}) \,\middle|\, \mathcal{E}_i\right) \cdot \Pr\left(\mathcal{E}_i\right) \\
& + \Pr\left(\mathcal{E}_i^c\right). \qquad (3.13)
\end{aligned}$$

By definition of stochastic ordering,

$$\begin{aligned}
& \Pr\left(\left|\zeta_i - \frac{\alpha_i\nu_i M}{\rho U}\right| \geq O(M^{2/3}) \,\middle|\, \mathcal{E}_i\right) \\
\leq \; & \Pr\left(G_i \geq \frac{\alpha_i\nu_i M}{\rho U} + O(M^{2/3})\right) \\
& + \Pr\left(H_i \leq \frac{\alpha_i\nu_i M}{\rho U} - O(M^{2/3})\right) \\
\overset{(a)}{\leq} \; & 2e^{-\Theta(M^{1/3})},
\end{aligned}$$

where (a) can be obtained using a similar Chernoff bounding approach as for N_c in Stage 1

of this proof. Thus, continuing from inequality (3.13), we further have

$$\begin{aligned}
&\Pr\left(\left|\zeta_i - \frac{\alpha_i \nu_i M}{\rho U}\right| \geq O(M^{2/3})\right) \\
\leq\ & 2e^{-\Theta(M^{1/3})} \cdot \Pr(\mathcal{E}_i) + (1 - \Pr(\mathcal{E}_i)) \\
=\ & 1 - (1 - 2e^{-\Theta(M^{1/3})}) \Pr(\mathcal{E}_i) \\
\leq\ & 1 - (1 - 2e^{-\Theta(M^{1/3})})(1 - 2e^{-\Omega((MB)^{1/3})}) \\
=\ & 2e^{-\Theta(M^{1/3})} - 2e^{-\Omega((MB)^{1/3})}. \qquad (3.14)
\end{aligned}$$

Putting inequality (3.14) back to inequality (3.12) immediately results in

$$\Pr(\text{box } b \text{ is good}) \geq 1 - 2|\mathcal{I}|e^{-\Theta(M^{1/3})}. \qquad (3.15)$$

4) The number of "good boxes:"

We use a similar approach as in Stage 2 to bound the number of good boxes, say Y, which can be represented as a function $g(\boldsymbol{\xi})$ where $\boldsymbol{\xi} = (\xi_1, \xi_2, \cdots, \xi_{MB})$ is the same content placement vector defined in Stage 2. Still, $g(\boldsymbol{\xi})$ changes by an amount no more than 1 when only one component ξ_i has its value changed, then for all $t > 0$, $\Pr(|Y - \mathbb{E}[Y]| \geq t) \leq 2e^{-2t^2/(MB)}$, and taking $t = (MB)^{2/3}$ further yields

$$\Pr\left(|Y - \mathbb{E}[Y]| \geq (MB)^{2/3}\right) \leq 2e^{-2(MB)^{1/3}}.$$

Similarly as we obtain inequality (3.8), we finally come to

$$\Pr\left(Y \geq B\left(1 - 2|\mathcal{I}|e^{-\Theta(M^{1/3})}\right)\right) \geq 1 - 2e^{-2(MB)^{1/3}}. \qquad (3.16)$$

5) The performance of a loss network:

Finally, consider the performance of the loss network defined by the "Counter-Based Acceptance Rule." We introduce an auxiliary system to establish an upper bound on the rejection rate. In the auxiliary system, upon arrival of a request for content c, L different requests are mapped to L distinct boxes holding a replica of c, but here they are accepted

individually rather than jointly, and no requests will be rejected. Letting Z_b (respectively, Z'_b) denote the number of requests associated to box b in the original (respectively, auxiliary) system, one readily sees that $Z_b \leq Z'_b$ at all times and all boxes and for each box b, the process Z'_b evolves as a $M/M/\infty$ system. We now want to upper bound the overall arrival rate of requests to a good box:

(a) Non-good contents

Assume that upon a request arrival, we indeed pick L content replicas, rather than L distinct boxes holding the requested content (as specified in the acceptance rule). This entails that, if two replicas of this content are present at one box, then this box can be picked twice. However, since a vanishing fraction of boxes will have more than one replicas of the same content when $M \ll \sqrt{(\min_i \alpha_i)B}$ (as proved in Lemma 2), we can strengthen the definition of a "good" box to ensure that, on top of the previous properties, a good box should hold M distinct replicas. It is easy to see that the fraction of good boxes will still be of the same order as with the original weaker definition.

With these modified definitions, consider one non-good content c of class i cached at a good box. Its unique replica will be picked with probability L/N_c when the sampling of L replicas among the N_c existing ones is performed. Thus, since we ignore requests for all content c with $N_c \leq M^{3/4}$ (according to Assumption 1), the request rate will be at most $\nu_i L M^{-3/4}$.

Besides, there are at most $O(M^{2/3})$ non-good content replicas held by one good box. The reason is as follows: By definition, a good box holds at least

$$\sum_{i \in \mathcal{I}} \left(\frac{\alpha_i \nu_i M}{\rho U} - O(M^{2/3}) \right) = M - O(M^{2/3}) \tag{3.17}$$

good content replicas among all classes, so the remaining slots, being occupied by non-good content replicas, are at most $O(M^{2/3})$. Therefore, the overall arrival rate of requests for non-good contents to a good box is upper bounded by

$$\overline{\nu}_{\text{non-good}} = O(M^{2/3} \cdot LM^{-3/4}) = O(LM^{-1/12}). \tag{3.18}$$

(b) Good contents

The rate generated by a good content c of class i is $\nu_i L/N_c$. Now, by definition of a good content, one has:

$$N_c \geq \frac{\nu_i M}{\rho U}(1 - O(M^{-1/3})).$$

This entails that the rate of requests for this content is upper bounded by

$$\frac{\rho LU}{M}(1 + O(M^{-1/3})).$$

By definition of a "good box," there are at most $\alpha_i \nu_i M/\rho U + O(M^{2/3})$ good content replicas of class i cached in this good box. Therefore, the overall arrival rate of requests for good contents to a good box is upper bounded by

$$\begin{aligned} \overline{\nu}_{\text{good}} &= \sum_{i \in \mathcal{I}} \left(\frac{\rho LU}{M}(1 + O(M^{-1/3})) \right) \\ &\quad \times \left(\frac{\alpha_i \nu_i M}{\rho U} + O(M^{2/3}) \right) \\ &= (\rho LU)(1 + O(M^{-1/3})). \end{aligned} \tag{3.19}$$

To conclude, for any good box b, the process Z'_b evolves as a one-dimensional loss network with arrival rate no larger than

$$\overline{\nu} = \overline{\nu}_{\text{non-good}} + \overline{\nu}_{\text{good}} = \rho LU + O(LM^{-1/12}),$$

by combining the two results in (3.18) and (3.19).

The number of good replicas in good boxes is, due to inequality (3.16) and equation (3.17), at least $MB(1 - O(M^{-1/3}))$, with a high probability (at least $1 - 2e^{-2(MB)^{1/3}}$). On the other hand, the total number of replicas of good contents is at most MB, which is the total number of replicas (or available cache slots).

Now pick some small $\epsilon \in (0, 1/3)$ and let $\tilde{X}$ denote the number of good contents which have at least $M^{2/3+\epsilon}$ replicas outside good boxes. Then necessarily, with a probability of at

least $1 - 2e^{-2(MB)^{1/3}}$,

$$\tilde{X}M^{2/3+\epsilon} \leq MB - MB(1 - O(M^{-1/3})) = O(BM^{2/3}),$$

i.e., $\tilde{X} \leq O(BM^{-\epsilon})$. According to inequality (3.8), the total number of good contents is $\Theta(B)$ (specifically, very close to $|\mathcal{C}| = \alpha B$) with a probability of at least $1 - 2|\mathcal{I}|e^{-2(MB)^{1/3}}$; hence, we can conclude that, with high probability, for a fraction of at least $1 - O(M^{-\epsilon})$ of good contents, each of them has at least a fraction $1 - O(M^{-1/3+\epsilon})$ of its replicas stored in good boxes (since a good content has $\frac{\nu_i}{\rho U}M \pm O(M^{2/3})$ replicas in total by definition). We further use $\tilde{\mathcal{C}}$ to represent the set of such contents.

Recall that A_c was defined in Subsection 2.3.2 as the steady-state probability of accepting a request for content c in the original system. For all $c \in \tilde{\mathcal{C}}$,

$$\begin{aligned} A_c &\geq \Pr(\text{all the } L \text{ sampled replicas are in good boxes}) \\ &\quad \times \Pr(Z_b < LU,\ \forall b \ s.t.\ \text{box } b \text{ is sampled}) \\ &\overset{(a)}{\geq} \left(1 - O(M^{-1/3+\epsilon})\right)^L \\ &\quad \times \Pr(Z'_b < LU,\ \forall b \ s.t.\ \text{box } b \text{ is sampled}). \end{aligned} \tag{3.20}$$

The argument why (a) holds is as follows: We have $N_c \approx \nu_i M/(\rho U)$ replicas (assuming that content c is of class i), among which $N'_c = N_c(1 - O(M^{-1/3+\epsilon}))$ are in good boxes. Then, the probability that L samples fall in the good boxes can be written explicitly as

$$\frac{N'_c(N'_c - 1)\cdots(N'_c - L + 1)}{N_c(N_c - 1)\cdots(N_c - L + 1)},$$

which can be approximated as the first part on the RHS we write above, under the assumption that $L \ll M$. The second part is due to the fact that $Z'_b \leq Z_b$ for all box b.

Next, we are going to further lower bound $\Pr(Z'_b < LU)$. Since $\overline{\nu}$ is an upper bound on the arrival rate, we have

$$\Pr(Z'_b \geq LU) \leq \Pr(S \geq LU) \leq e^{-\overline{\nu}I(LU/\overline{\nu})},$$

where $S \sim \text{Poi}(\overline{\nu})$, $I(x) = x \log x - x + 1$ and the second inequality is a Chernoff bound. Since $\overline{\nu} = \rho LU + O(LM^{-1/12})$, we can further show that $\Pr(Z'_b \geq LU) \leq e^{-\Theta(L)}$, under the condition that $\rho < 1$ (otherwise, the exponent will become 0 or $+\Theta(L)$). Thus, $\Pr(Z'_b < LU) \geq 1 - e^{-\Theta(L)}$. Continuing from inequality (3.20), we have

$$A_c \;\; \geq \;\; \left(1 - O(M^{-1/3+\epsilon})\right)^L \cdot \left(1 - Le^{-\Theta(L)}\right). \tag{3.21}$$

It should be recalled that within this stage of proof, finally coming to inequality (3.21) actually needs everything to be conditional on the following events:

- The number of good boxes is $\Theta(B)$;
- The number of good contents is $\Theta(B)$;
- A box caches M distinct replicas,

and as $B, M \to \infty$ and $M \ll \sqrt{(\min_i \alpha_i)B}$, all of them have high probabilities. Additionally, $\tilde{\mathcal{C}} \xrightarrow{p} \mathcal{C}$ as $B, M \to \infty$. Therefore, further letting $L \to \infty$ but keeping $L \ll M^{1/3-\epsilon}$, we will find that the RHS of inequality (3.20) is approximated as

$$1 - O(LM^{-1/3+\epsilon}) - Le^{-\Theta(L)} \approx 1,$$

and then conclude that the requests for almost all the contents will have near-zero loss. ■

CHAPTER 4

ONLINE ADVERTISEMENT, OPTIMIZATION AND STOCHASTIC NETWORKS

4.1 Introduction

In this chapter, we study another important network application: online advertising. Our online advertising model is formulated as the following:

Online Advertising Model:

Assume that queries for keyword q arrive to the search engine according to a stochastic process at rate ν_q queries per time slot, where we have assumed that time is discrete and a *"time slot"* is our smallest discrete time unit. In response to each query arrival, the search engine may display ads from some clients on the webpage. There are L different places (e.g., top, bottom, left, right, etc.) on a webpage where ads could be displayed. We will call these places *"webpage slots."* When client i's ad is displayed in webpage slot s when keyword q is queried, there is a probability with which the user who is viewing the page (the one who generated the query) will click on the ad. This probability, called the *"click-through rate,"* is denoted by c_{qis}.

A client specifies the amount of money ("bid") that it is willing to pay to the search service provider when a user clicks on its ad related to a specific query. We use r_{qi} to denote this per-click payment from client i for its ad related to a query for keyword q. Additionally, client i also specifies an average budget b_i which is the maximum amount that it is willing to pay per *"budgeting cycle"* on average, where a budgeting cycle equals to N time slots (we have introduced the notion of a budgeting cycle since the time-scale over which queries arrive may be different than the time-scales over which budgets may be settled).

The problem faced by the search service provider is then to assign advertisements to

webpage slots, in response to each query, so that its long-term average revenue is maximized.

Based on the above model, we design an online algorithm which achieves a long-term average revenue within $O(\epsilon)$ of the offline optimal revenue, where ϵ can be chosen arbitrarily small, indicating the near-optimality of our online algorithm. Before entering into the details, we will first survey the related literature, highlight the main contributions of our work, and discuss the differences between our model and previous ones.

4.1.1 Related Work

We will only survey the online resource allocation models here, and not the auction models. The online ads model in prior literature mainly includes two types, namely AdWords (AW) and Display Ads (DA), of which the difference lies in the constrained resource of each client. In the AW model, the resource is the client's budget, while in the DA model, the resource is the maximum number of impressions agreed on by the client and the service provider. Correspondingly, after each resource allocation step, the resource of a client whose ad is posted, is reduced by the bid value[1] in the AW model, or one impression in the DA model. Both of them belong to a general class of packing linear programs formulated in [25]. Most of the prior online algorithms for solving the AW and DA model respect the hard constraint on the client's resources. One exception is [26], where the authors argue that "free disposal" of resources makes the DA model more tractable (but not necessary for the AW model).

Mehta et al. [27] modeled the online ads problem as a generalization of an online matching problem [28] on a bipartite graph of queries and clients. Later in [29], Buchbinder et al. showed that matching clients to webpage slots (whether it is a single slot or multiple slots) can be solved as a maximum-weighted matching problem. Following [29], a number of other online algorithms using the maximum-weighted bipartite matching idea have been proposed in [30, 26, 31] and [25]. The algorithms in [32] and [27], which were earlier than [29], can also be regarded as maximum-weighted matching solutions on this bipartite graph of clients

[1]This refers to the pay-per-impression scheme. With a pay-per-click scheme, the reduction only happens if the ad is clicked.

and webpage slots.

In [32], the "b-matching" problem (related to the online ads context, bids are trivially 0 or 1 and budgets are all b) is solved by an $1 - 1/e$ competitive algorithm as $b \to \infty$ and the weights are the remaining budgets of those clients interested in the newly arrived query (i.e., the bid equals 1). For the online ads problem in which bids and budgets can have general and different values, [27] (its longer version is [33]) uses the "discounted" bids as the weights corresponding to each client. The discount factor is calculated by a function $\psi(x) = 1 - e^{x-1}$, of which the input x is the fraction of a client's budget that has been consumed. Their algorithm is also $1 - 1/e$ competitive, under an assumption that bids are small compared to budgets. By taking advantage of estimated numbers of query arrivals for each keyword within a given period and modifying the discount factor in [27], Mahdian et al. [30] designed a class of algorithms which achieve a considerably better competitive ratio with accurate estimates while still guaranteeing a reasonably good competitive ratio with inaccurate estimates, also assuming small bids.

The algorithms in [29, 26, 31, 25] and [34] all use a primal-dual framework to compute a maximum-weighted matching at each iteration, in which the dual variables (corresponding to each client) are used to determine the weights. The two $1 - 1/e$ competitive algorithms in [29] and [26] update the dual variables dynamically in their primal-dual type algorithms every time a decision is made. Specifically, each dual variable in [29], which implicitly tracks the fraction of budget that has been spent by the corresponding client, grows during each iteration at a rate parameterized by the fraction of the bid for the incoming query in this client's total budget, while [26] uses an "exponentially weighted average" of the up-to-date $n(i)$ most valuable impressions[2] assigned to client i as a new dual variable with respect to this client. On the other hand, the three dual type learning-based algorithms in [31], [25] and [34] achieve a competitive ratio of $1 - O(\epsilon)$ based on a random-order arrival model (rather than the adversarial model in most of the earlier work), assuming small bids and knowledge of the total number of queries. The main difference between them is that [31] and [25] use an initial ϵ fraction of queries to learn the optimal dual variables (with respect to

[2]In the DA model in [26], $n(i)$ is defined as the maximum number of impressions agreed for client i. After allowing free disposal, only the current $n(i)$ most valuable impressions assigned to client i will be considered.

this training set), while the algorithm in [34] repeats the learning process over geometrically growing intervals. Additionally, the "small bids" condition in [34] is slightly weaker than the condition in [31] and [25].

4.1.2 Our Contributions and Comparison to Prior Work

As in prior work (especially [29] and [26]), our solution relies on a primal-dual framework to solve a maximum-weighted matching problem on a bipartite graph of clients and webpage slots, with dynamically updated dual variables which contribute to the weights on the edges of the bipartite graph. However, unlike prior work, we are able to obtain a revenue which is $O(\epsilon)$ close to the optimal revenue using a purely adaptive algorithm without the need for the knowledge of the number of query arrivals over a time period or the average arrival rates.

Our solution is related to scheduling problems in wireless networks. In particular, we use the optimization decomposition ideas in [2, 35] and the stochastic performance bounds in [4]. Borrowing from that literature, we introduce the concept of an *"overdraft" queue*. The overdraft queue measures the amount by which the provided service temporarily exceeds the budget specified by a client.

Our online algorithm exhibits a trade-off between the revenue obtained by the service provider and the level of overdrafts. Specifically, our algorithm can achieve a revenue that is within $O(\epsilon)$ of the optimal revenue while ensuring that the overdraft is $O(1/\epsilon)$, where ϵ can be arbitrarily small. We can further modify our online algorithm so that clients can always operate strictly under their budgets. Finally, our algorithm and analysis naturally allow us to assess the impact of click-through rate estimation on the service providers revenue.

Besides the revenue maximization model, we study another model in which the objective is to maximize the average overall click-through rate, subject to a minimum impression requirement for each client. We also show that our results can be naturally extended to handle non-stationary query arrival processes and clients which have short-term contracts with the service provider.

Like the algorithm in [34], our algorithm can also be generalized to a wider class of linear programs within different application contexts, where the coefficients in the objective

function and constraints are not necessarily nonnegative.

There are two points of departure in our algorithm compared to existing models: the first one is that we assume a stochastic model in which the query arrival rates are unknown. Thus, there is no need to know the number of arrivals in a time period as in prior models. The other is that we assume an average budget rather a fixed budget over a time horizon. This allows us to better model permanent clients (e.g., big companies who do not stop advertising) and those who do not provide a fixed time-horizon budget. Clients who advertise for a limited amount of time are not explicitly modeled here. But we believe that our model will also handle such clients well since the algorithm is naturally adaptive. For such clients, one can divide their total budget by the time horizon over which they contract with the service provider to approximately model them in our framework.

A minor difference with respect to prior models is that our model assumes that time is slotted. This can be easily modified to assume that query arrivals can occur at any time according to some continuous-time stochastic process. The only difference is that our analysis would then involve continuous-time Lyapunov drift instead of the discrete-time drift used in this paper. From a theoretical point of view, our analysis is different from prior work which uses competitive ratios: our model and solution is similar in spirit to stochastic approximation [36] where gradients (here the gradient of the dual objective) are known only with stochastic perturbations. This point of view is essential to model stochastic traffic with unknown statistics.

Instead of the $1-O(\epsilon)$ competitive ratio in prior work, we show that our algorithm achieves a revenue which is within $O(\epsilon)$ of the optimal revenue. The $O(\epsilon)$ penalty arises due to the stochastic nature of our model. However, we do not require assumptions such as knowledge of the total number of queries in a given period [30, 31, 25, 34], or information of keyword frequencies [30].[3]

[3]It should be mentioned that another common assumption "small bids" (or "large budgets", "large offline optimal value") used in [32, 27, 30, 26, 31, 34, 29] and [25] is not essentially different from our "long-term" assumption.

4.1.3 Organization of This Chapter

The rest of this chapter is organized as follows: In Section 4.2, we formulate an optimization problem involving long-term averages. In Section 4.3, we start considering the stochastic version of our model and propose an online algorithm, which also introduces the concept of "overdraft queue." Performance analysis of this online algorithm, which includes the near-optimality of the long-term revenue and an upper bound on the overdraft level, will also be done in Section 4.3. The last two subsections of Section 4.3 present two extensions, namely the decisions based on estimated click-through rates and the "underdraft" mechanism. The second online ads model "click-through rate maximization problem" with its related extensions, algorithm design and analysis is given in Section 4.4. Section 4.5 further considers short-term clients and non-stationary query arrivals.

4.2 An Optimization Problem Involving Long-Term Averages

Based on the model described above, we first pose the revenue maximization problem as an optimization problem involving long-term averages. For this purpose, we define an assignment of clients to webpage slots as a matrix M of which the $(i,s)^{\text{th}}$ element is defined as follows:

$$M_{is} = \begin{cases} 1, & \text{if client } i \text{ is assigned to webpage slot } s \\ 0, & else. \end{cases}$$

The matrix M has to satisfy some practical constraints. First, a webpage slot can be assigned to only one client and vice versa. Furthermore, the assignment of clients to certain webpage slots may be prohibited for certain queries. For example, it may not make sense to advertise chocolates when someone is searching for information about treatments for diabetes. These constraints can be abstracted as follows: For the queried keyword q, the set of assignment matrices have to belong to some set $\mathcal{M}_q$. We also let $p_{_{qM}}$ be the probability of choosing matrix M when the queried keyword is q.

The optimization problem is then given by

$$\max_{\mathbf{p}} \bar{R}(\mathbf{p}) = \sum_{q} \nu_q \sum_{M \in \mathcal{M}_q} p_{qM} \sum_{i,s} M_{is} c_{qis} r_{qi} \tag{4.1}$$

subject to

$$N \sum_{q} \nu_q \sum_{M \in \mathcal{M}_q} p_{qM} \sum_{s} M_{is} c_{qis} r_{qi} \leq b_i, \qquad \forall i; \tag{4.2}$$

$$0 \leq p_{qM} \leq 1, \qquad \forall q,\ M \in \mathcal{M}_q; \tag{4.3}$$

$$\sum_{M \in \mathcal{M}_q} p_{qM} \leq 1, \qquad \forall q. \tag{4.4}$$

In the above formulation, the objective (4.1) is the average revenue per time slot and constraint (4.2) expresses the fact that the average payment over a budgeting cycle should not exceed the average budget. The optimization is a linear program and if all the problem parameters are known, in principle, it can be solved offline, returning probabilities $\{p_{qM}\}$ which can be used by a service provider to maximize its revenue. However, such an offline solution is not desirable for at least two reasons:

- Being a static approach, it does not use any feedback about the current state of the system. For example, the fact that the empirical average payment of a client has severely exceeded its average budget would have no impact on the subsequent assignment strategy. Since the formulation and hence, the solution, only cares about long-term budget constraint satisfaction, severe overdraft or underdraft of the budget can occur over long periods of time.

- The offline solution is a function of the query arrival rates $\{\nu_q\}$. Thus, a change in the arrival rates would require a recomputation of the solution.

In view of these limitations of the offline solution, we propose an online solution which adaptively assigns client advertisements to webpage slots to maximize the revenue. As we will see, the online solution does use feedback about the overdraft (or underdraft) level in future decisions, and does not require knowledge of $\{\nu_q\}$.

4.3 Online Algorithm and Performance Analysis

4.3.1 A Dual Gradient Descent Solution

To get some insight into a possible adaptive solution to the problem, we first perform a dual decomposition which suggests a gradient solution. However, a direct gradient solution will not take into account the stochastic nature of the problem and will also require knowledge of the query arrival rates $\{\nu_q\}$. We will address these issues in the following subsections, using techniques that, to the best of our knowledge, have not been used in prior literature on the online advertising problem.

We append the constraint (4.2) to the objective (4.1) using Lagrange multipliers $\delta_i \geq 0$ to obtain a partial Lagrangian function

$$\begin{aligned} L(\mathbf{p}, \boldsymbol{\delta}) &= \sum_q \nu_q \sum_{M \in \mathcal{M}_q} p_{qM} \sum_{i,s} M_{is} c_{qis} r_{qi} - \sum_i \delta_i \cdot \left(\sum_q \nu_q \sum_{M \in \mathcal{M}_q} p_{qM} \sum_s M_{is} c_{qis} r_i - \frac{b_i}{N} \right) \\ &= \sum_q \nu_q \sum_{M \in \mathcal{M}_q} p_{qM} \sum_{i,s} M_{is} c_{qis} r_{qi} (1 - \delta_i) + \sum_i \frac{\delta_i b_i}{N}, \end{aligned}$$

subject to constraints (4.3) and (4.4). The dual function is

$$D(\boldsymbol{\delta}) = \max_{\mathbf{p}} \sum_q \nu_q \sum_{M \in \mathcal{M}_q} p_{qM} \sum_{i,s} M_{is} c_{qis} r_{qi} (1 - \delta_i) + \sum_i \frac{\delta_i b_i}{N},$$

subject to constraints (4.3) and (4.4). Note that the maximization part in the dual function can be decomposed into independent maximization problems with regard to each queried keyword q, i.e., for all q,

$$\max_{\{p_{qM},\ M \in \mathcal{M}_q\}} \sum_{M \in \mathcal{M}_q} p_{qM} \sum_{i,s} M_{is} c_{qis} r_{qi} (1 - \delta_i) = \max_{M \in \mathcal{M}_q} \sum_{i,s} M_{is} c_{qis} r_{qi} (1 - \delta_i),$$

where it is easy to see that each maximization is solved by a deterministic solution. This suggests the following primal-dual algorithm to iteratively solve the original optimization

problem (4.1): at step k,

$$\forall q, \quad \hat{M}^*(q,k) \in \arg\max_{M \in \mathcal{M}_q} \sum_{i,s} M_{is} c_{qis} r_{qi} (1 - \delta_i(k));$$

$$\forall i, \quad \delta_i(k+1) = \left[\delta_i(k) + \epsilon \left(N \sum_q \nu_q \sum_s [\hat{M}^*(q,k)]_{is} \cdot c_{qis} r_{qi} - b_i \right) \right]^+,$$

where $\epsilon > 0$ is a fixed step-size parameter, and $[x]^+ = x$ if $x \geq 0$ or $[x]^+ = 0$ otherwise. Furthermore, defining $\hat{Q}_i(k) \triangleq \delta_i(k)/\epsilon$, the above iterative algorithm becomes

$$\forall q, \quad \hat{M}^*(q,k) \in \arg\max_{M \in \mathcal{M}_q} \sum_{i,s} M_{is} c_{qis} r_{qi} \left(\frac{1}{\epsilon} - \hat{Q}_i(k) \right);$$

$$\forall i, \quad \hat{Q}_i(k+1) = \left[\hat{Q}_i(k) + \hat{\lambda}_i(k) - b_i \right]^+,$$

where

$$\hat{\lambda}_i(k) \triangleq N \sum_q \nu_q \sum_s [\hat{M}^*(q,k)]_{is} c_{qis} r_{qi}. \tag{4.5}$$

Note that $\hat{Q}_i(k)$ can be interpreted as a queue which has $\hat{\lambda}_i(k)$ arrivals and b_i departures at step k. Although this algorithm already uses the feedback provided by $\{\hat{\mathbf{Q}}(k)\}$ (or $\{\boldsymbol{\delta}(k)\}$) about the state of the system, it is still using a priori information about the arrival rates of queries in $\{\hat{\boldsymbol{\lambda}}(k)\}$, hence not really "online." However, it motivates us to incorporate a queueing system with stochastic arrivals into the real online algorithm, which will be described in the next subsection.

4.3.2 Stochastic Model, Online Algorithm, and "Overdraft Queue"

In practice, a search service provider may not have a priori information about the query arrival rates $\{\nu_q\}$, and generally, query arrivals during each time slot are stochastic rather than constant. Let time slots be indexed by $t \in \mathcal{Z}^+ \cup \{0\}$. We specify our detailed statistical assumptions as follows:

- Query arrivals: Assume that a time slot is short enough so that query arrivals in each time slot can be modeled as a Bernoulli random variable with occurrence probability

ν. The probability that an arrived query is for keyword q is assumed to be ϑ_q and $\sum_q \vartheta_q = 1$. Let $\tilde{q}(t)$ represent the index of the keyword queried in time slot t, such that $\tilde{q}(t) = q$ w.p. $\nu_q = \nu\vartheta_q$ for all q (indexed by positive integers) and $\tilde{q}(t) = 0$ w.p. $1 - \nu$, which accounts for the case that no query arrives.

- Budget spending: We limit the values of budget spent in each budgeting cycle to be integers. To match the average budget b_i (when it is not an integer), the budget of client i in budgeting cycle k is assumed to be a random variable $\tilde{b}(k)$ which equals $\lceil b_i \rceil$ w.p. ϱ_i and $\lfloor b_i \rfloor$ otherwise, such that $E[\tilde{b}(k)] = \varrho_i \lceil b_i \rceil + (1 - \varrho_i)\lfloor b_i \rfloor = b_i$, i.e., $\varrho_i = \frac{b_i - \lfloor b_i \rfloor}{\lceil b_i \rceil - \lfloor b_i \rfloor} = b_i - \lfloor b_i \rfloor$. For the trivial case that b_i is already an integer, we let $\varrho_i = 1$.

- Click-through behaviors: In time slot t, after a query for keyword q arrives, if the ad of client i is posted on webpage slot s in response to this query, then whether this ad will be clicked is modeled as a Bernoulli random variable $\tilde{c}_{qis}(t)$ with occurrence probability c_{qis}.

We now want to implement the above iterative algorithm online based on this stochastic model. According to definition (4.5), $\hat{\lambda}_i$ includes average query arrivals and click-through choices within N time slots (i.e., one budgeting cycle). Thus, each iteration step in the online algorithm should correspond to a budgeting cycle. For convenience, we define

$$\mathbf{u}(k) \triangleq \{\tilde{q}(t), \tilde{\mathbf{c}}(t) \text{ for } kN \leq t \leq kN + N - 1\}$$

as a collection of random variables describing user behaviors (including stochastic query arrivals and click-through choices) in budgeting cycle k. The online algorithm is then described as follows:

Online Algorithm: (in each budgeting cycle $k \geq 0$)

In each time slot $t \in [kN, kN + N - 1]$, if $\tilde{q}(t) > 0$, choose the assignment matrix

$$\tilde{M}^*(t, \tilde{q}(t), \mathbf{Q}(k)) \in \arg \max_{M \in \mathcal{M}_{\tilde{q}(t)}} \sum_{i,s} M_{is} c_{\tilde{q}(t)is} r_{\tilde{q}(t)i} \left(\frac{1}{\epsilon} - Q_i(k) \right). \tag{4.6}$$

At the end of budgeting cycle k, for each client i, update

$$Q_i(k+1) = \left[Q_i(k) + A_i(k, \mathbf{Q}(k), \mathbf{u}(k)) - \tilde{b}_i(k)\right]^+, \tag{4.7}$$

where

$$A_i(k, \mathbf{Q}(k), \mathbf{u}(k)) \triangleq \sum_{t=kN}^{kN+N-1} \sum_{s} [\tilde{M}^*(t, \tilde{q}(t), \mathbf{Q}(k))]_{is} \cdot \tilde{c}_{\tilde{q}(t)is}(t) \cdot r_{\tilde{q}(t)i}. \tag{4.8}$$

Here, $A_i(k, \mathbf{Q}(k), \mathbf{u}(k))$ represents the revenue obtained by the service provider from client i during budgeting cycle k, and recall that $\tilde{b}_i(k)$ is a random variable which takes integer values whose mean is equal to the average budget per budgeting cycle.

In this algorithm, client i is associated with a virtual queue Q_i (maintained at the search service provider). During budgeting cycle k, the amount of money client i is charged by the search service provider $A_i(k, \mathbf{Q}(k), \mathbf{u}(k))$ is the arrival to this queue, and the average budget per budgeting cycle b_i is the departure from this queue. Note that if this queue is positive, it means that the total value of the real service already provided to the client has temporarily exceeded the client's budget, i.e., "free" service has been provided temporarily. Hence, we call this queue the "overdraft queue."

There are two different time scales here. The faster one is a time slot, the smallest time unit used to capture user behaviors (including stochastic query arrivals and click-through choices) and execute ad-posting strategies. The slower one is a budgeting cycle (equal to N time slots), at the end of which the overdraft queues are updated based on the revenue obtained over the whole budgeting cycle.

We make the following assumptions on the above stochastic model: $\{\tilde{q}(t)\}$ are i.i.d. across time slots t; $\{\tilde{c}_{qis}(t)\}$ are independent across q, i, s, and t; each variable in $\{\tilde{q}(t)\}$ and each variable in $\{\tilde{c}_{qis}(t)\}$ are mutually independent. In fact, the model can be generalized to allow for query arrivals correlated over time and across keywords, and other similar correlations inside the click-through choices or between these two stochastic processes. Such models would only make the stochastic analysis more cumbersome, but the main results will

continue to hold under these more general models.

In order to guarantee that the Markov chain which we will define later is both irreducible and aperiodic, we further assume that the probability of whether there is an arrival in a time slot $\nu \in (0, 1)$. We also assume that r_{qi} for all q and i can only take integer values. Together with the fact that $\tilde{b}(k)$ takes integer values, $\{\mathbf{Q}(k)\}$ becomes a discrete-time integer-valued queue. Note that assuming integer values is only for ease of analysis, but not necessary.

4.3.3 An Upper Bound on the Overdraft

According to the ad assignment step (4.6), if at the beginning of budgeting cycle k, $Q_i(k) > 1/\epsilon$, then for this budgeting cycle, the i^{th} row of $\tilde{M}^*(t, q, \mathbf{Q}(k))$ is always a zero vector, i.e., the service provider will not post the ads of client i until $Q_i(k)$ falls below $1/\epsilon$. Since by assumption the number of query arrivals per time slot is upper bounded, for any budgeting cycle k, one can bound the transient length of each overdraft queue as below:

$$Q_i(k) \leq \frac{1}{\epsilon} + N \cdot \arg\max_{q,s}\{r_{qi}c_{qis}\} - \lfloor b_i \rfloor, \ \forall i.$$

Therefore, $Q_i(k) \sim O(1/\epsilon)$ for all i, and stability is not an issue for these "upper bounded" queues. It further implies that this online algorithm satisfies the budget constraints in the long run, i.e., for all client i,

$$\lim_{K\to\infty} E\left[\frac{1}{K}\sum_{k=0}^{K-1} A_i(k, \mathbf{Q}(k), \mathbf{u}(k))\right] \leq b_i \tag{4.9}$$

must hold.

It should be mentioned that in [37], through using the LIFO queueing discipline, the authors show an $O((\log(1/\epsilon))^2)$ bound on the averaged waiting time encountered by most of the packets, which is tighter than the bound $O(1/\epsilon)$ under the FIFO queueing discipline (see e.g. [2]; our above result also fits this bound). While the length of a FIFO queue is proportional to the arrival rate according to Little's law [38], the length of a LIFO queue in [37] is still $O(1/\epsilon)$, even if it is occupied by very "old" packets which only accounts for a

negligible fraction $O(\epsilon^{\log(1/\epsilon)})$ of all the packets that have arrived. Unlike in a communication network where waiting time is usually the main concern and dropping a small fraction of old packets does almost no hurt to many online applications, what clients of online advertising service care about is how much they have paid beyond their budgets, which is measured by the overdraft queue in our model.

4.3.4 Near-Optimality of the Online Algorithm

We can show that, in the long term, the proposed online algorithm achieves a revenue that is close to the optimal revenue $\bar{R}(\mathbf{p}^*)$ (where $\mathbf{p}^*$ is the solution to the optimization problem (4.1)).

We start with the following lemma:

Lemma 3 *Consider the Lyapunov function $V(\mathbf{Q}) = \frac{1}{2}\sum_i Q_i^2$. For any $\epsilon > 0$, and each time period k,*

$$E[V(\mathbf{Q}(k+1))|\mathbf{Q}(k) = \mathbf{Q}] - V(\mathbf{Q}) \leq -\frac{N}{\epsilon}\left(\bar{R}(\mathbf{p}^*) - \bar{R}(\tilde{\mathbf{p}}^*(k,\mathbf{Q}))\right) + B_1 - B_2\sum_i Q_i.$$

Here,

$$\begin{aligned} B_1 \triangleq \frac{1}{2}\Big(& (N(N-1)L^2 + NL)(\arg\max_{q,i,s}\{c_{qis}r_{qi}\})^2 \\ & + \sum_i \lceil b_i \rceil^2 (b_i - \lfloor b_i \rfloor) + \lfloor b_i \rfloor^2 (1 - b_i + \lfloor b_i \rfloor)\Big), \end{aligned} \tag{4.10}$$

where L is the number of webpage slots;

$$B_2 \triangleq \min_i\{b_i - N\sum_q \nu_q \sum_{M\in\mathcal{M}_q} p^*_{qM} \sum_s M_{is}c_{qis}r_{qi}\}; \tag{4.11}$$

and $\tilde{\mathbf{p}}^*(k,\mathbf{Q}) \triangleq \{\tilde{p}^*_{qM}(k,\mathbf{Q}),\ \forall q, M \in \mathcal{M}_q\}$ where $\tilde{p}^*_{qM}(k,\mathbf{Q})$ equals 1 if $M = \tilde{M}^*(t,q,\mathbf{Q})$ for $kN \leq t \leq kN+N-1$ (i.e., the optimal matrix in the maximization step (4.6)) and 0 otherwise. ◇

The proof is given in Appendix B.4.

Now we are ready to present one of the major theorems in this chapter, indicating that the long-term average revenue achieved by our online algorithm is within $O(\epsilon)$ of the maximum revenue obtained by the offline optimal solution. The proof is given in Appendix B.5.

Theorem 4 *For any $\epsilon > 0$,*

$$0 \leq \lim_{K\to\infty} E\left[\bar{R}(\mathbf{p}^*) - \frac{1}{KN}\sum_{k=0}^{K-1} R(k)\right] \leq \frac{B_1\epsilon}{N}$$

for some constant $B_1 > 0$ defined as

$$B_1 \triangleq \frac{1}{2}\Big((N(N-1)L^2 + NL)(\arg\max_{q,i,s}\{c_{qis}r_{qi}\})^2 + \sum_i \lceil b_i\rceil^2(b_i - \lfloor b_i\rfloor) + \lfloor b_i\rfloor^2(1 - b_i + \lfloor b_i\rfloor)\Big), \tag{4.12}$$

where L is the number of webpage slots, and $R(k) \triangleq \sum_i A_i(k, \mathbf{Q}(k), \mathbf{u}(k))$ is defined as the revenue obtained during budgeting cycle k. ◇

Remark 5 *If we choose a very small ϵ, the matching in (4.6) behaves like a greedy solution until the queue lengths grows comparably large. This indicates a tradeoff between how close to the long-term optimal revenue the algorithm can achieve and the actual convergence time.*

Additionally, supposing that $\{r_{qi}\}$ and $\{b_i\}$ are both measured in another scale with a factor α, e.g., using cents instead of dollars ($\alpha = 100$), and assuming that α is unknown, it can be shown that the $O(\epsilon)$ convergence bound will also be scaled by α if we measure the revenue in the original scale. To change the algorithm into a "scale-free" version, $\{r_{qi}\}$ and $\{b_i\}$ should be divided by a common benchmark value, e.g., the largest budget specified by all the initially existing clients. Since the benchmark value is also implicitly multiplied by α if measured in another scale, the scaling factor will be canceled in the normalized $\{r_{qi}\}$ and $\{b_i\}$ and no longer affect the convergence bound. ◇

4.3.5 Impact of Click-Through Rate Estimation

In our online algorithm, the decision of picking an optimal ad assignment matrix in (4.6) in response to each query is based on the true click-through rates $\mathbf{c}$. In reality, an estimate $\hat{\mathbf{c}}$

based on historical click-through behaviors is used, i.e., in response to each query for keyword q, which arrives in time slot $t \in [kN, kN+N-1]$, we choose the assignment matrix

$$\tilde{M}^*(t, \tilde{q}(t), \mathbf{Q}(k)) \in \arg\max_{M \in \mathcal{M}_{\tilde{q}(t)}} \sum_{i,s} M_{is} \hat{c}_{\tilde{q}(t)is} r_{\tilde{q}(t)i} \left(\frac{1}{\epsilon} - Q_i(k) \right). \tag{4.13}$$

We then have the following corollary in addition to Theorem 4 in Subsection 4.3.4:

Corollary 3 *Assume that the estimated click-through rates* $\hat{\mathbf{c}} \in [\mathbf{c}(1-\Delta), \mathbf{c}(1+\Delta)]$ *with some* $\Delta \in (0,1)$. *Under our online algorithm with estimated click-through rates,* $\mathbf{Q}(k)$ *is still positive recurrent. Then, for any* $\epsilon > 0$,

$$\lim_{K \to \infty} E\left[\frac{1}{KN} \sum_{k=0}^{K-1} R(k) \right] \geq \left(\frac{1-\Delta}{1+\Delta} \right) \cdot \bar{R}(\mathbf{p}^*) - \frac{B_1 \epsilon}{N},$$

for some constant $B_1 > 0$ *(defined in equation (4.12)).* ◇

Proving this needs some minor changes to the proof of Lemma 3 and Theorem 4, which will be shown in Appendix B.6.

Remark 6 *Corollary 3 tells us that for small* ϵ, *the long-term average revenue achieved by our online algorithm with estimated click-through rates will be at least* $\left(\frac{1-\Delta}{1+\Delta}\right)$ *of the offline optimal revenue.* ◇

4.3.6 Underdraft: Staying under the Budget

In the previous sections, we allowed the provision of temporary free service to clients, which we call overdraft. If this is not desirable for some reason, the algorithm can be modified to have non-positive overdraft. We do this by allowing the queue lengths to become negative, but not positive. The practical meaning of negative queue lengths is to allow each client to accumulate a certain volume of "credits" if the current budget is under-utilized and use these credits to offset future possible overdrafts. We call this negative queue length *"underdraft."* Corresponding to this mechanism, we modify our online algorithm as follows: in response

to each query for keyword q, which arrives in time slot $t \in [kN, kN+N-1]$, choose the assignment matrix

$$\tilde{M}^*(t, \tilde{q}(t), \mathbf{Q}(k)) \in \arg\max_{M \in \mathcal{M}_{\tilde{q}(t)}} \sum_{i,s} M_{is} c_{\tilde{q}(t)is} r_{\tilde{q}(t)i} \left(\Gamma_i - Q_i(k)\right),$$

and at the end of budgeting cycle k, for each client i, update

$$Q_i(k+1) = \max\{Q_i(k) + A_i(k, \mathbf{Q}(k), \mathbf{u}(k)) - \tilde{b}_i(k), -C_i\},$$

where Γ_i denotes a customized "throttling threshold" (not necessarily $1/\epsilon$) and C_i denotes the maximum allowable credit volume for client i. Recall that $A_i(k, \mathbf{Q}(k), \mathbf{u}(k))$ is defined in equation (4.8). We can bound each overdraft queue as below:

$$Q_i(k) \leq \Gamma_i + N \cdot \arg\max_{q,s}\{r_{qi} c_{qis}\} - \lfloor b_i \rfloor, \ \forall i, k.$$

Thus, if our objective is to eliminate overdrafts (i.e., $Q_i(k) \leq 0$ for all k), we can set

$$\Gamma_i := \left[\lfloor b_i \rfloor - N \cdot \arg\max_{q,s}\{r_{qi} c_{qis}\}\right]^-, \ \forall i, \tag{4.14}$$

where in contrast to $[x]^+$, $[x]^-$ takes the non-positive part of x, i.e., $[x]^- = x$ if $x \leq 0$ or $[x]^- = 0$ otherwise. We further let

$$C_i := \frac{1}{\epsilon} - \Gamma_i, \ \forall i,$$

so that after converting $Q_i(k)$ to be nonnegative by using $\tilde{Q}_i(k) = Q_i(k) + C_i$ for all i, everything is transformed back to the original online algorithm except that each $Q_i(k)$ is replaced by $\tilde{Q}_i(k)$; hence, we can still show that the revenue achieved by this modified version of online algorithm is within $O(\epsilon)$ of the optimal revenue.

It might seem counter-intuitive that by letting ϵ go to zero, we can incur potentially large underdrafts (under-utilization of the budget) and yet are able to achieve maximum revenue. This is not a contradiction: for each fixed ϵ, in the long term, the average service provided to

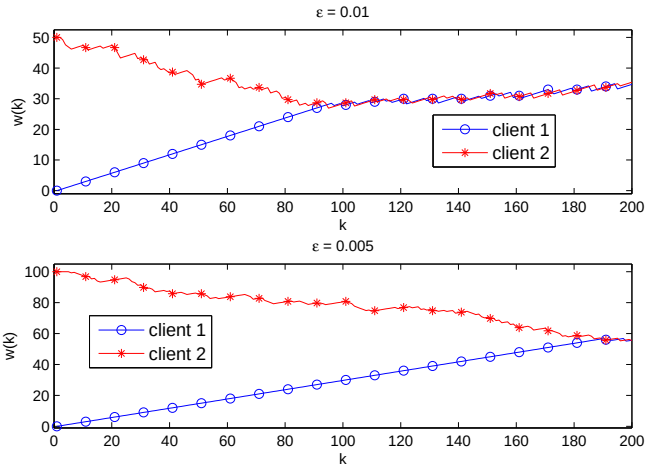


Figure 4.1: Temporary unfairness in service

each client is close to the average budget. The $O(1/\epsilon)$ is a fixed amount by which the total budget up to any time T is under-utilized, and, after divided by T, it goes to zero when T approaches infinity.

We note that while an underdraft does not seem to significantly hurt either the client, who actually benefits from an underdraft, or the service provider, whose long-run average revenue is still diminished only by $O(\epsilon)$, large values of the underdraft may result in temporary unfairness in the system.[4] If, for example, a client accumulates a large underdraft compared to the other clients, then it may receive priority over other clients for large periods of time. To illustrate this, we consider an example with two clients and one queried keyword. Assume that $\Gamma_i < 0$ for $i = 1, 2$, and at time slot k_0, $Q_1(k_0) = \Gamma_1$ and $Q_2(k_0) = -C_2$ (this occurs with a positive probability due to the ergodicity of the Markov chain $\{\mathbf{Q}(k)\}$ proved before). We simulate the sample paths of the weights in the maximization step (4.20) with the following setting: budgets $b_1 = b_2 = 0.6$, click-through rates $c_1 = c_2 = 0.5$, revenue-per-click $r_1 = r_2 = 1$; the number of query arrivals per time slot equals 2 w.p. 0.5 and 0 otherwise; a budgeting cycle equals to one time slot ($N = 1$) for simplicity. The results for both $\epsilon = 0.01$ and $\epsilon = 0.005$ ($k = 0$ corresponds to k_0 here) are shown in Figure 4.1. Client 2 keeps getting

[4]Note that this temporary unfairness is not an artifact of the underdraft mechanism. In fact, it occurs once a sample path enters a state where some clients have huge differences from others in their corresponding queue lengths, which can also happen under the original algorithm. We are just using the underdraft scheme to illustrate this phenomenon.

services until the weights of both clients reaches the same level, and the smaller ϵ is, the longer the "unfair serving" period lasts.

It should be mentioned that this underdraft idea can be used under any upper-bounded query arrival model, not restricted in the Bernoulli arrival model considered in this paper.

4.4 Click-Through Rate Maximization Problem

In this section, we consider another online ads model, in which the objective is to maximize the long-term average total click-through rate of all queries. Instead of average budget, client i specifies in the contract an average "*impression requirement*" m_i, which is the minimum number of times an ad of this client should be posted by the service provider per "*requirement cycle*" (equal to N time slots) on average. The other parameters are the same as in the model proposed in Section 4.1 for the revenue maximization problem.

The corresponding optimization formulation now becomes

$$\max_{\mathbf{p}\in\mathcal{F}} \bar{J}(\mathbf{p}) = \sum_q \nu_q \sum_{M\in\mathcal{M}_q} p_{qM} \sum_{i,s} M_{is} c_{qis} \tag{4.15}$$

where the feasible set $\mathcal{F}$ is characterized by

$$N \sum_q \nu_q \sum_{M\in\mathcal{M}_q} p_{qM} \sum_s M_{is} \geq m_i, \qquad \forall i; \tag{4.16}$$

$$0 \leq p_{qM} \leq 1, \qquad \forall q,\ M \in \mathcal{M}_q; \tag{4.17}$$

$$\sum_{M\in\mathcal{M}_q} p_{qM} \leq 1, \qquad \forall q. \tag{4.18}$$

Different from the revenue maximization problem, here the feasible set can become empty if some m_i is too high. Basically, without constraint (4.16), $\mathcal{F}$ is relaxed to

$$\mathcal{F}_0 \triangleq \{\mathbf{p} : 0 \leq p_{qM} \leq 1,\ \forall q, M \in \mathcal{M}_q;\ \sum_{M\in\mathcal{M}_q} p_{qM} \leq 1,\ \forall q\}. \tag{4.19}$$

We can then define the following capacity region which characterizes how large the average

number of impressions can be achieved for each client per requirement cycle:

$$\mathcal{C} \triangleq \left\{ \boldsymbol{\mu} : \mu_i = N \sum_q \nu_q \sum_{M \in \mathcal{M}_q} p_{qM} \sum_s M_{is},\ \forall i,\ s.t.\ \mathbf{p} \in \mathcal{F}_0 \right\}.$$

Clearly, $\mathbf{m} \in \mathcal{C}$ must hold to ensure the existence of a solution for the above optimization problem. Through a similar approach as in Subsection 4.3.1, we can write down a similar online algorithm based on the same stochastic model as defined in Subsection 4.3.2. We define $\mathbf{q}(k) \triangleq \{\tilde{q}(t),\ \text{for}\ kN \leq t \leq kN + N - 1\}$. Similar to $\tilde{b}_i(k)$, $\tilde{m}(k) = \lceil m_i \rceil$ w.p. $m_i - \lfloor m_i \rfloor$ and $\tilde{m}(k) = \lfloor m_i \rfloor$ otherwise.

Online Algorithm: (in each requirement cycle $k \geq 0$)

In each time slot $t \in [kN, kN + N - 1]$, if $\tilde{q}(t) > 0$, choose the assignment matrix

$$\tilde{M}^*(t, \tilde{q}(t), \mathbf{Q}(k)) \in \arg \max_{M \in \mathcal{M}_{\tilde{q}(t)}} \sum_{i,s} M_{is} \left(\frac{c_{\tilde{q}(t)is}}{\epsilon} + Q_i(k) \right). \tag{4.20}$$

At the end of requirement cycle k, for each client i, update

$$Q_i(k+1) = \left[Q_i(k) + \tilde{m}(k) - S_i(k, \mathbf{Q}(k), \mathbf{q}(k)) \right]^+,$$

where

$$S_i(k, \mathbf{Q}(k), \mathbf{q}(k)) \triangleq \sum_{t=kN}^{kN+N-1} \sum_s [\tilde{M}^*(t, \tilde{q}(t), \mathbf{Q}(k))]_{is}. \tag{4.21}$$

In real online advertising business, some clients may only have short-term contracts, i.e., clients may not be interested in the average number of impressions per time slot but may be interested in a minimum number of impressions in a given duration (such as a day). Further, query arrivals may not form a stationary process. In fact, they are more likely to vary depending on the time of day. These extensions are considered in Appendix E. Such extensions also make sense for the revenue maximization model considered in the previous

sections, but the approach is similar to Appendix E and so will not be considered here.

4.4.1 Performance Evaluation

$S_i(k, \mathbf{Q}(k), \mathbf{q}(k))$ defined in (4.21) represents the actual number of impressions for client i's ads during requirement cycle k. The queue length increases when the average impression requirements in a particular requirement cycle cannot be fulfilled. Hence, a positive queue represents accumulated "credits," which enhances the chance of being assigned with a webpage slot in the future, much like a negative queue in the revenue maximization problem. We thus call this queue a "*credit queue*." Unlike the revenue maximization problem in which an $O(1/\epsilon)$ upper bound on the transient queue length is automatically imposed by the online algorithm, here we need to prove the stability of the queues and show an upper bound on the mean queue length. Since $\{\mathbf{Q}(k)\}$ defines an irreducible and aperiodic Markov chain, in order to prove its stability (positive recurrence), we will first bound the expected drift of $\mathbf{Q}(k)$ for a suitable Lyapunov function.

Lemma 4 *Consider the Lyapunov function* $V(\mathbf{Q}) = \frac{1}{2}\sum_i Q_i^2$. *For any* $\epsilon > 0$ *and each requirement cycle* k,

$$E[V(\mathbf{Q}(k+1))|\mathbf{Q}(k) = \mathbf{Q}] - V(\mathbf{Q}) \leq \frac{D_3}{\epsilon} + D_1 - D_2 \sum_i Q_i. \tag{4.22}$$

Here,

$$D_1 \triangleq \frac{1}{2}\Big(N(N-1)L^2 + NL + \sum_i \lceil m_i \rceil^2 (m_i - \lfloor m_i \rfloor) + \lfloor m_i \rfloor^2 (1 - m_i + \lfloor m_i \rfloor)\Big), \tag{4.23}$$

where L *is the number of webpage slots;*

$$D_2 \triangleq \min_i \{N \sum_q \nu_q \sum_{M \in \mathcal{M}_q} \hat{p}_{qM} \sum_s M_{is} - m_i\}, \tag{4.24}$$

for some $\hat{\mathbf{p}} \in \mathcal{F}$ *such that* $D_2 > 0$*; and*

$$D_3 \triangleq N \cdot \max_{\mathbf{p} \in \mathcal{F}_0} \bar{J}(\mathbf{p}) \tag{4.25}$$

where $\mathcal{F}_0$ *is defined in* (4.19). ◇

The proof is similar to the proof of Lemma 3 with some modifications in the final steps, which will be given briefly in Appendix B.7. With this lemma, we can conclude that $\mathbf{Q}(k)$ is positive recurrent because the expected Lyapunov drift is negative except for a finite set of values of $\mathbf{Q}(k)$, according to Foster-Lyapunov theorem ([38, 39]).

Remark 7 *Note that compared to the definition of* B_2 *in* (4.11) *of Lemma 3 where* $B_2 \geq 0$*,* D_2 *needs to be strictly positive in order to prove the stability of queues. Such a* $\hat{\mathbf{p}}$ *in the definition of* D_2 *can always be found unless* $\mathcal{F}$ *is a degenerate set with at most one element.* ◇

The stability of the queues directly implies the following corollary:

Corollary 4 (Overservices in the long term)

$$\lim_{K \to \infty} E\left[\frac{1}{K} \sum_{k=1}^{K} S_i(k, \mathbf{Q}(k), \mathbf{q}(k))\right] \geq m_i, \quad \forall i.$$

◇

In addition to proving stability, Lemma 4 will be used to evaluate the upper bound on the expected total queue length in the steady state, as shown in the following theorem:

Theorem 5 *Under the online algorithm,*

$$E\left[\sum_i Q_i(\infty)\right] \leq \frac{1}{D_2^*}\left(D_1 + \frac{D_3}{\epsilon}\right), \tag{4.26}$$

where D_1 and D_3 are respectively defined in (4.23) *and* (4.25)*; D_2^* is defined as*

$$D_2^* \triangleq \max_{\mathbf{p}\in\mathcal{F}_0} D_2(\mathbf{p}) \tag{4.27}$$

where D_2 is defined in (4.24) *(regarded as a function of $\mathbf{p}$).* ◇

Proof Averaging both sides of inequality (4.22) over $0 \leq k \leq K-1$, taking $K \to \infty$ and doing some simple algebra, one obtains

$$\limsup_{K\to\infty} \frac{1}{K} \sum_{k=0}^{K-1} E\left[\sum_i Q_i(k)\right] \leq \frac{1}{D_2}\left(D_1 + \frac{D_3}{\epsilon}\right).$$

The LHS equals to $E\left[\sum_i Q_i(\infty)\right]$ according to Theorem 15.0.1 in [39]. The RHS is minimized through maximizing D_2 over all $\mathbf{p} \in \mathcal{F}_0$ (which will certainly satisfy $\mathbf{p} \in \mathcal{F}$ and $D_2 > 0$). This completes our proof. ■

The following theorem shows that the online algorithm proposed above achieves a long-term average click-through rate within $O(\epsilon)$ of the offline optimum. The proof is similar to the one for Theorem 4 and hence will be omitted.

Theorem 6 *For any $\epsilon > 0$,*

$$0 \leq \lim_{K\to\infty} E\left[\bar{J}(\mathbf{p}^*) - \frac{1}{KN}\sum_{k=0}^{K-1} J(k)\right] \leq \frac{D_1\epsilon}{N},$$

for some constant $D_1 > 0$ (defined in (4.23) *in Lemma 4). Here, $J(k)$ is defined as the total number of click-through events within requirement cycle k.* ◇

4.4.2 Customizing Impression Requirements $\{m_i\}$ Based on Query Arrival Rates $\{\nu_q\}$

Since a positive queue measures how much the service provider "owes" a client, reducing the coefficient of the $1/\epsilon$ term in the upper bound on the mean queue length becomes important.

Besides, we also need to guarantee $\mathbf{m} \in \mathcal{C}$. In order to handle these two issues, we introduce an approach to customizing $\{m_i\}$ based on known (or estimated) query arrival rates $\{\nu_q\}$,

Replacing D_2^* in Theorem 5 by a common D_2 defined in equation (4.24), if we want the expected total queue length to be upper bounded by Q_{max}, it suffices to let

$$D_2 \geq \xi \triangleq \frac{1}{Q_{max}} \left(D_1 + \frac{D_3}{\epsilon} \right), \tag{4.28}$$

where D_3 is already determined, and D_1 does not matter much given a small ϵ although it includes unknown $\{m_i\}$. We then solve the following optimization problem to determine $\{m_i\}$:

$$\begin{aligned} &\max_{\mathbf{p} \in \mathcal{F}_0, \mathbf{m}} \sum_i \log m_i \\ \text{s.t.} \quad & N \sum_q \nu_q \sum_{M \in \mathcal{M}_q} p_{qM} \sum_s M_{is} - m_i \geq \xi, \ \forall i. \end{aligned}$$

Here we use $\sum_i \log m_i$ as the objective function in order to guarantee a unique optimal solution and impose a certain fairness rule called "proportional fairness" (see e.g. [40]). Note that ξ cannot be set too large (i.e., Q_{max} cannot be set too small), otherwise there may not exist a feasible solution.

Naturally, a question would arise: now that we need to solve some mathematical programming like the above one based on knowledge of query arrival rates, why not also directly solve the original linear programming in (4.15) and use the offline optimal solution $\mathbf{p}^*$ to assign ads? The answer to this is similar to the max-weight algorithm for wireless networks. In [41] and [42], it has been shown that adaptive algorithms lead to much better queueing performance compared to static offline algorithms. We verify this assertion in our context through simulations in the next subsection.

4.4.3 Queue Update in a Faster Time Scale

In the original algorithm, the queue length is updated only at the end of each requirement cycle and used in the max-weight matching for the next whole requirement cycle. The longer

a requirement cycle lasts, the more obsolete the queue length information becomes, so with a large N, short-term performances may not be so good even if long-term performances are still guaranteed.

We then propose a solution which updates queue lengths in a faster time scale. Specifically, we divide each requirement cycle into T *queueing cycles* with equal lengths (assuming $N/T \in \mathcal{Z}^+$ without loss of generality). We use $\{\hat{\mathbf{Q}}(k,\tau) : 0 \le \tau \le T\}_{k\ge 0}$ to denote this new queueing system and assume $\hat{\mathbf{Q}}(-1,T) = \mathbf{0}$. At the beginning of each requirement cycle k before any decision, update

$$\hat{\mathbf{Q}}(k,0) = \hat{\mathbf{Q}}(k-1,T) + \tilde{\mathbf{m}}(k),$$

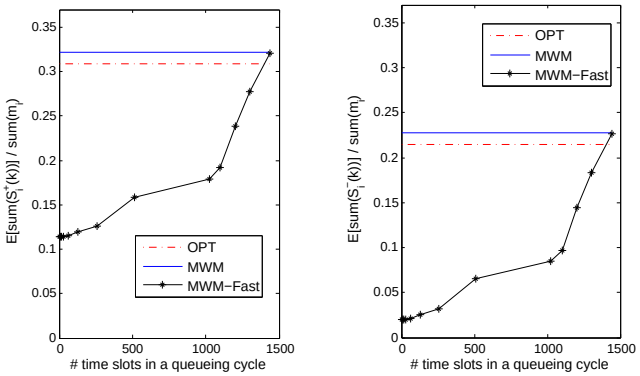
and at the end of the τ^{th} queueing cycle within this requirement cycle ($1 \le \tau \le T$), for all client i,

$$\hat{Q}_i(k,\tau) = \left[\hat{Q}_i(k,\tau-1) - \sum_{t=kN+(\tau-1)\frac{N}{T}}^{kN+\tau\frac{N}{T}-1} \sum_s [\tilde{M}^*(t,\tilde{q}(t),\hat{\mathbf{Q}}(k,\tau))]_{is}\right]^+.$$

Since $\|\hat{\mathbf{Q}}(k,T) - \mathbf{Q}(k)\| \le B$ for some constant B independent of the queue lengths, it can be shown that the long-term performances evaluated in Subsection 4.4.1 are still guaranteed (the idea behind such a proof would be similar to the one in [43] and so is omitted).

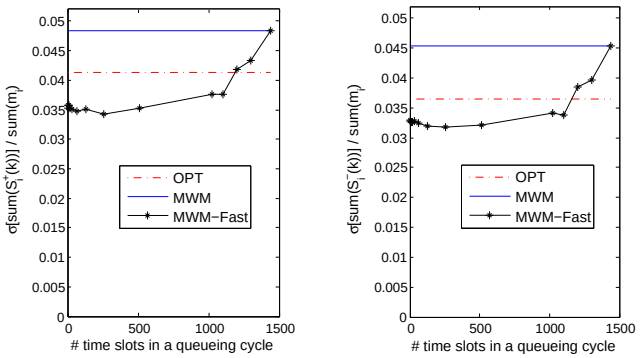
Next, we use simulations to compare three different algorithms, namely a randomized algorithm following the offline optimal solution (labeled as OPT) and two versions of our online algorithm "max-weight matching" with and without "fast queue update" respectively (labeled as MWM-Fast and MWM respectively). In each scenario we test, all the parameters are randomly generated. The impression requirements $\{m_i\}$ are chosen through the approach in Subsection 4.4.2.

We take an example scenario with 2 webpage slots, 5 keywords and 10 clients. The probability that a query arrives in a time slot equals 0.7. Specifically, for the five keywords, the query arrival rates are $\boldsymbol{\nu} = [0.2364, 0.0594, 0.1669, 0.0714, 0.1659]$. Table 4.1 shows the click-through rates for the ten clients ($C_1 \sim C_{10}$) corresponding to each keyword ($q_1 \sim q_5$), on webpage slots 1 and 2 respectively (a zero click-through rate indicates that the corresponding client is not related to this keyword). We use $N = 1440$ (say, one time slot is one minute



(a) Over-Service (b) Under-Service

Figure 4.2: Average overall over-service and under-service (normalized by the total impression requirement) impacted by the "fast queue update"



(a) Over-Service (b) Under-Service

Figure 4.3: The standard variance of overall over-service and under-service (normalized by the total impression requirement) impacted by the "fast queue update"

and one requirement cycle is one day), $\epsilon = 10^{-4}$ and $Q_{max} = 20/\epsilon$ (recall that Q_{max} is used to set up an upper bound on the mean queue length by the heuristic in Subsection 4.4.2). The simulation has been run for 1000 requirement cycles.

To compare the performances of all the three algorithms, instead of considering the long-term performance requirements that we have used in the theory, we introduce two new metrics: over-service $S_i^+(k) \triangleq [S_i(k) - \tilde{m}_i(k)]^+$ and under-service $S_i^-(k) \triangleq [\tilde{m}_i(k) - S_i(k)]^+$ to client i during requirement cycle k. Note that these metrics measure deviations from the guarantees over short time scales and so are more stringent requirements than the long-term guarantees used in the theory.

We show respectively in Figures 4.2(a) and 4.2(b) that the average overall over-service and

Table 4.1: Click-through rates for all the clients' ads

	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8	C_9	C_{10}
	Webpage Slot 1									
q_1	0	0.519	0.973	0	0.649	0	0	0	0.800	0
q_2	0	0	0	0.340	0	0	0.952	0	0	0
q_3	0.982	0.645	0.856	0.461	0.190	0	0.369	0.669	0.156	0
q_4	0.423	0	0	0	0.599	0	0.179	0	0.471	0.094
q_5	0	0	0	0.875	0	0.518	0	0	0	0
	Webpage Slot 2									
q_1	0	0.235	0.421	0	0.536	0	0	0	0.067	0
q_2	0	0	0	0.312	0	0	0.050	0	0	0
q_3	0.118	0.248	0.194	0.222	0.036	0	0.158	0.252	0.092	0
q_4	0.296	0	0	0	0.020	0	0.124	0	0.032	0.060
q_5	0	0	0	0.826	0	0.330	0	0	0	0

under-service normalized by the total impression requirement, i.e., $E[\sum_i S_i^+(k)]/\sum_i m_i$ and $E[\sum_i S_i^-(k)]/\sum_i m_i$, are both reduced by the fast queue update. Similarly, a "variance reduction" effect is shown by the fast queue update based on the statistics $\sqrt{var[\sum_i S_i^+(k)]}/\sum_i m_i$ and $\sqrt{var[\sum_i S_i^-(k)]}/\sum_i m_i$, respectively in Figures 4.3(a) and 4.3(b). In terms of the overall click-through rate, our simulation has verified that the three algorithms achieve approximately the same performance (the figure is omitted here) and further demonstrated in Figure 4.4 that the fast queue update can also reduce its variance. Note that these performances of each individual client also improve and we simply omit the figures here.

Observed from Figure 4.5, the offline optimal solution leads to very unstable queue dynamics. This essentially arises from the fact that the algorithm operates on an optimal point $\mathbf{p}^*$ for which some inequalities in constraint (4.16) may be tight. In contrast, our online algorithm guarantees the stability of queues, and the faster the queues update, the more stable the queue dynamics become (as an example we use $T = 24$, i.e., the number of time slots per queueing cycle equals 60). This is consistent with the above results which show a reduction of over-service and under-service in both mean and variance since these metrics directly measure the level of deviations around the equilibrium point of each stable queue.

Remark 8 *While a long-term client may only be concerned with average performances, a*

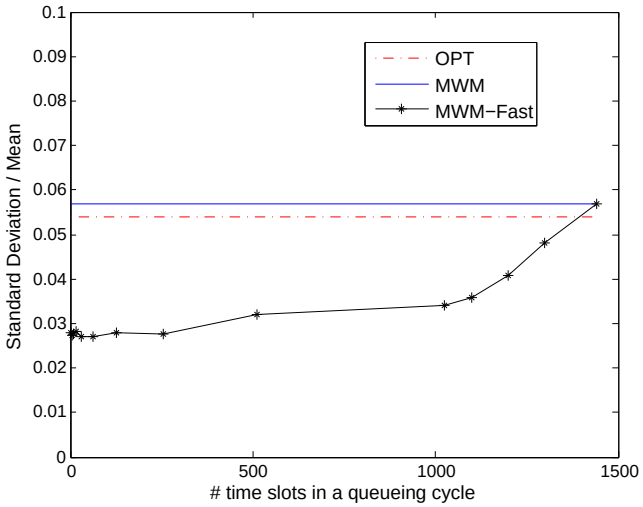


Figure 4.4: The "standard variance to mean ratio" of overall click-through rate impacted by the "fast queue update"

short-term client cares about both mean (the average level for all the clients of its type) and variance (related to its own individual level), especially for the performances of under-service and click-through rate.[5] *All of these are well handled by our online algorithm with fast queue updates.*

4.5 Short-Term Clients and Non-Stationary Query Arrivals

We focus on the click-through rate maximization problem, although a similar model and solution can be used for revenue maximization problem.

First, consider how to include short-term clients in the system. Let us index long-term clients from 1 to n, the i^{th} of which has an average impression requirement of m_i per requirement cycle. There are further $\tilde{n}$ types of short-term clients indexed by $n+1 \leq i \leq n+\tilde{n}$. Each short-term client of type i has a impression requirement of l_i per contract term. Without loss of generality, we assume that the contract term of any short-term client is equal to one requirement cycle. In each requirement cycle k, there are $X_i(k)$ clients of type i in the system, where $X_i(k)$ follows a stationary stochastic process with mean x_i and $X_i(k)$ is known at the beginning of requirement cycle k.

[5]Over-service are cared about by the online ads service provider.

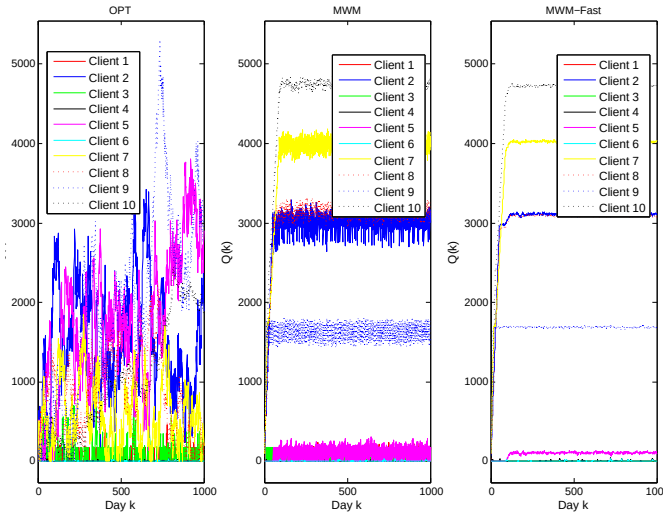


Figure 4.5: Queue dynamics under three algorithms

Correspondingly in an ad assignment matrix M, the first n rows and the subsequent $\tilde{n}$ rows represent the n long-term clients and the $\tilde{n}$ types of short-term clients, respectively. If short-term type j is assigned to some webpage slot, one out of $X_j(k)$ clients of this type is chosen uniformly at random due to their homogeneity.

Additionally, for a short-term client of type i, the algorithm is actually aimed to satisfy at least only $(1 - \alpha_i)l_i$, where $\alpha_i \in [0, 1]$ is called "unfulfilled rate" for clients of type i and to be determined by the algorithm. A strictly convex and monotonically increasing function $\phi(\alpha_i) \in [0, \infty)$ is then introduced to measure the "unhappiness" of short-term clients about unfulfilled impression requirements, and deducted from the original objective function "overall average click-through rate" in (4.15) after being scaled by some predetermined weight w_i which reflects the importance of the new metric "unfulfilled rate."

The second extension from the original model is to consider a more general query arrival pattern. We introduce a new time scale "stationary-arrival period" between the fast one "time slot" t and the slow one "requirement cycle" k, namely one requirement cycle equals H stationary-arrival periods (assuming that $N/H \in \mathcal{Z}^+$ and usually $N/H \gg 1$), and we assume that query arrivals with respect to each keyword q form a stationary stochastic process with rate $\nu_q(h)$ within the h^{th} stationary-arrival period in one requirement cycle for all $1 \leq h \leq H$. This is a more reasonable assumption for the query arrival pattern in the real Internet. For example, in one day, the query arrivals are stationary within each individual

hour, non-stationary across different hours, and stationary in the same hour across different days. This corresponds to $H = 24$, although setting a contract term (already assumed to be equal to one requirement cycle) as one day would only be a simplification for ease of exposition. Based on this example, in the following text we are going to use "day" and "hour" instead of "requirement cycle" and "stationary-arrival period" to better describe the basic ideas.

In summary, the new optimization problem is formulated as

$$\max_{\{\mathbf{p}(h),\forall h;\ \boldsymbol{\alpha}\}} \frac{1}{H}\sum_{q}\sum_{h=1}^{H}\nu_q(h)\sum_{M\in\mathcal{M}_q} p_{_{qM}}(h)\sum_{1\le i\le n+\tilde{n},s} M_{is}c_{qis} - \sum_{i=n+1}^{n+\tilde{n}} w_i\phi(\alpha_i)$$

subject to

$$\frac{N}{H}\sum_{q}\sum_{h=1}^{H}\nu_q(h)\sum_{M\in\mathcal{M}_q} p_{_{qM}}(h)\sum_{s} M_{is} \ge \begin{cases} m_i \quad , & \forall \quad 1\le i\le n \\ (1-\alpha_i)l_i x_i, & \forall\ n+1\le i\le n+\tilde{n} \end{cases}$$

and

$$0\le p_{_{qM}}(h)\le 1,\ \forall q,\ M\in\mathcal{M}_q,\ 1\le h\le H; \quad \sum_{M\in\mathcal{M}_q} p_{_{qM}}(h)\le 1,\ \forall q,\ 1\le h\le H.$$

The only modification in the online algorithm described in Subsection is to add the following two steps specially for each type of short-term clients:

- At the beginning of the k^{th} day, update

  $$\alpha_i^*(k) = \arg\max_{\alpha_i\in[0,1]}\left\{l_iQ_i(k)X_i(k)\cdot\alpha_i - \frac{Hw_i}{\epsilon}\phi(\alpha_i)\right\}$$

  which corresponds to the target "unfulfilled rate" for each type of short-term clients in this day. We can simply check three possible values for α_i, namely 0, 1 and $\psi\left(\frac{\epsilon l_i X_i(k)\cdot Q_i(k)}{Hw_i}\right)$, where the function $\psi \triangleq \left[\frac{d\phi}{d\alpha}\right]^{-1}$, to see which one maximizes the above expression.

- At the end of the k^{th} day, "credit queue" i maintained for type i of short-term clients

is updated as

$$Q_i(k+1) \;=\; Q_i(k) + (1 - \alpha_i^*(k)) \cdot l_i X_i(k) - S_i(k, \mathbf{Q}(k), \mathbf{q}(k)),$$

where $S_i(k, \mathbf{Q}(k), \mathbf{q}(k))$ is defined in (4.21).

The conclusions and proofs about near-optimality of the objective value, queueing stability and upper bound on the expected queue length are similar to those shown for the original problem in Subsection 4.4.1 and hence omitted here.

Note that the online algorithm is still "oblivious" to the query arrivals, when the arrival processes become non-stationary to some extent. This is an artifact of dual decomposition w.r.t. each hour h, in addition to a decomposition w.r.t. each keyword q as we have seen before.

CHAPTER 5

A UNIVERSAL LOWER BOUND ON THE EXPECTED OVERDRAFT LEVEL IN ONLINE ADVERTISING

In Chapter 4, we are able to show that our online algorithm achieves an overdraft level of $O(1/\epsilon)$. So a natural question is whether this bound is tight. We are unable to answer this question at this time. However, we will show that the overdraft has to increase when ϵ becomes small. In this chapter, we want to show that in the long run, an expected overdraft level of $\Omega(\log(1/\epsilon))$ is unavoidable under any stationary ad assignment algorithm which achieves a long-term average revenue within $O(\epsilon)$ of the offline optimum, when the queue length is only allowed to be nonnegative. An ad assignment algorithm ϖ is defined as a strategy which uses matrix $M^{\varpi}(t,q) \in \mathcal{M}_q$ for ad assignment when a query for keyword q arrives at each time slot t. During each budgeting cycle k, the revenue obtained from client i under algorithm ϖ is defined as

$$A_i^{\varpi}(k) \triangleq \sum_{t=kN}^{kN+N-1} \sum_{s} [M^{\varpi}(t, \tilde{q}(t))]_{is} \cdot \tilde{c}_{\tilde{q}(t)is}(t) \cdot r_{\tilde{q}(t)i}. \tag{5.1}$$

We then define average revenue obtained from client i per budgeting cycle as $\lambda_i^{\varpi} \triangleq E[A_i^{\varpi}(k)]$ in the steady state. The long-term average revenue (per time slot) is thus $\bar{R}^{\varpi} = \sum_i \lambda_i^{\varpi}/N$, and the overdraft level of client i evolves as

$$Q_i^{\varpi}(k+1) = \left[Q_i^{\varpi}(k) + A_i^{\varpi}(k) - \tilde{b}_i(k) \right]^+. \tag{5.2}$$

Note that our online algorithm is one particular ϖ, which makes the decision based on the current overdraft levels of all clients.

To seek a universal lower bound on expected overdraft level in the long run (here, equivalent to steady state), we only have to consider those algorithms ϖ such that $\bar{Q}_i^{\varpi} \triangleq$

$E[Q_i^{\varpi}(k)] < \infty$ for all i. To categorize these "stable" algorithms, we define "per-client revenue region," similar to the concept of "capacity region" in the context of queueing networks:

Definition 1 ("Per-Client Revenue Region")

$$\mathcal{C} \triangleq \left\{ \boldsymbol{\lambda}^{\varpi} = \{\lambda_i^{\varpi}\} \geq \mathbf{0} \colon \exists \varpi \ s.t. \ \lambda_i^{\varpi} \triangleq E\left[A_i^{\varpi}(k)\right] \leq b_i, \ \forall i \right\},$$

given fixed parameters $\{r_{qi}\}$, $\{b_i\}$, $\{c_{qis}\}$, N *and statistical properties of* $\tilde{q}(t)$ *and* $\{\tilde{c}_{qis}(t)\}$. $\diamond$

The offline optimal average revenue is then equal to $\max_{\boldsymbol{\lambda} \in \mathcal{C}} \sum_i \lambda_i / N$, which is denoted as $\bar{R}^*$.

Note that if the query arrival rates per budgeting cycle are too low, the average revenue drawn from some client will never hit its specified budget, no matter which algorithm ϖ s.t. $\boldsymbol{\lambda}^{\varpi} \in \mathcal{C}$ you pick (i.e., $\exists$ i s.t. no feasible solution $\mathbf{p}$ can make constraint (4.2) for this i tight). The system resources (here, budgets) are underutilized and it is not so important to consider the tradeoff between revenue and overdraft. To avoid this, we can assume a relatively large N (i.e., the number of time slots in one budgeting cycle) such that

$$N \geq \max_i \left\{ \frac{b_i}{\sum_q \nu_q r_{qi} \cdot \max_{M \in \mathcal{M}_q^i} c_{qis(i,M)}} \right\}, \tag{5.3}$$

where $\mathcal{M}_q^i \subseteq \mathcal{M}_q$ is defined as a set of ad assignment matrices, of which the i^{th} row has a "1", and $s(i, M)$ in $c_{qis(i,M)}$ refers to the column in M where that "1" stays. This guarantees that for each i, there exists an algorithm ϖ_i such that $\boldsymbol{\lambda}^{\varpi_i} \in \mathcal{C}$ and $\lambda_i^{\varpi_i} = b_i$. In the following text, we will assume the above condition for N.

5.1 One Keyword, One Client and One Webpage Slot

We start with the simplest model: one keyword, one client and one webpage slot (hence we omit all the subscripts in the corresponding notations). Under condition (5.3), the offline

maximum average revenue is trivially b/N.

Theorem 7 *Given a small $\epsilon > 0$, if an algorithm ϖ leads to $E[A^{\varpi}(k)] \geq b - \epsilon$ in the steady state, then*

$$\bar{Q}^{\varpi} \geq \frac{\log(1/\epsilon)}{2(1 - \log(\varphi P_+))} - 1,$$

where we assume that

$$\varphi \triangleq \Pr(\textit{no query arrival in a budgeting cycle}) > 0,$$

and $P_+ \triangleq \Pr(\tilde{b}(k) > 0) > 0$. $\diamond$

Note that this result works for any query arrival and budget spending model satisfying the above two stated assumptions, and not only restricted to the model we described in Subsection 4.3.2. In the proof below, we generally write $\tilde{b}(k)$ as a random variable which can possibly take all nonnegative integer values.

Proof We ignore the superscript ϖ for brevity. The dynamics of the queue is rewritten as $Q(k+1) = Q(k) + A(k) - \hat{b}(k)$, where the actual departure process is defined as

$$\hat{b}(k) \triangleq \begin{cases} \tilde{b}(k) & \text{if } \; Q(k) + A(k) - \tilde{b}(k) \geq 0; \\ Q(k) + A(k) & \text{otherwise.} \end{cases} \tag{5.4}$$

Let $p_i \triangleq \Pr(\hat{b}(k) = i)$ and $q_i \triangleq \Pr(\tilde{b}(k) = i)$ in the steady state. Note that

$$\begin{aligned} b - \epsilon \;&\leq\; E[A(k)] = E[\hat{b}(k)] = \sum_{i=1}^{\infty} \Pr(\hat{b}(k) \geq i) = \Pr(\hat{b}(k) \geq 1) + \sum_{i=2}^{\infty} \Pr(\hat{b}(k) \geq i) \\ &\overset{(a)}{\leq}\; (1 - p_0) + \sum_{i=2}^{\infty} \Pr(\tilde{b}(k) \geq i) = 1 - p_0 + \Big(b - \Pr(\tilde{b}(k) \geq 1)\Big) = 1 - p_0 + b - (1 - q_0) \\ &=\; q_0 - p_0 + b, \end{aligned}$$

where (a) holds because $\Pr(\hat{b}(k) \geq i) \leq \Pr(\tilde{b}(k) \geq i)$ for all $i \geq 0$. Thus, $p_0 \leq q_0 + \epsilon$. Since

$$\Pr(\hat{b}(k) = 0) = \Pr(\tilde{b}(k) = 0) + \Pr(\hat{b}(k) = 0,\ \tilde{b}(k) \geq 1),$$

we have $p_0 = q_0 + \tilde{p}_0$, where $\tilde{p}_0 \triangleq \Pr(\hat{b}(k) = 0,\ \tilde{b}(k) \geq 1)$. Therefore,

$$\tilde{p}_0 \leq \epsilon. \tag{5.5}$$

Next, we are looking for a lower bound on $\tilde{p}_0$ in relation to $\bar{Q}$. Letting $P_+ \triangleq \Pr(\tilde{b}(k) > 0)$ (which is surely a positive constant since $b > 0$), we then have

$$\begin{aligned} n\tilde{p}_0 &= \sum_{k=0}^{n-1} \Pr(\hat{b}(k) = 0,\ \tilde{b}(k) > 0) \overset{(a)}{\geq} \Pr\left(\bigcup_{k=0}^{n-1} \{\hat{b}(k) = 0,\ \tilde{b}(k) > 0\}\right) \\ &\overset{(b)}{\geq} \Pr(Q(0) \leq n-1;\ A(k) = 0, \tilde{b}(k) > 0,\ \forall\ 0 \leq k \leq n-1) \\ &= \Pr(Q(0) \leq n-1) \cdot \prod_{k=0}^{n-1} \Pr(A(k) = 0) \cdot \Pr(\tilde{b}(k) > 0) \\ &= (\varphi P_+)^n \cdot \Pr(Q(0) \leq n-1) \overset{(c)}{\geq} (\varphi P_+)^n \left(1 - \bar{Q}/n\right), \end{aligned} \tag{5.6}$$

where (a) holds according to the union bound, (b) holds since the event on the RHS implies the one on the LHS, and (c) holds due to the Markov inequality. If we pick $n := \lceil 2\bar{Q} \rceil \in [2\bar{Q}, 2\left(\bar{Q}+1\right)]$, inequality (5.6) further implies that

$$\tilde{p}_0 \geq \frac{(\varphi P_+)^n}{2n} \overset{(e)}{\geq} e^{-n(1-\log(\varphi P_+))} \geq e^{-2(\bar{Q}+1)(1-\log(\varphi P_+))}, \tag{5.7}$$

where (e) holds because $\frac{1}{2x} \geq e^{-x}$ for all $x > 0$. Combining inequalities (5.5) and (5.7) then completes the proof. ■

In the related literature, [44] comes up with an $\Omega(1/\sqrt{\epsilon})$ bound for a set of algorithms under some admissibility conditions, while [45] provides an $\Omega(\log(1/\epsilon))$ bound for more general algorithms.

Our proof uses the following ideas inspired by [45]: if the throughput is lower bounded by a number close to the average potential departure rate, then the probability of zero actual

departures given nonzero potential departures must be upper bounded by a small number; further, if the average queue length is given, then the probability of hitting zero must be upper bounded because otherwise, the queue length would become small. However, we cannot directly use the expression for the lower bound in [45] since it imposes certain strict convexity assumptions which do not apply to our model where the objective is linear. So we have provided a very simple derivation of the lower bound on the queue length for our specific model.

Additionally, our $\Omega(\log(1/\epsilon))$ bound based on a linear objective function can be extended to the multi-queue case (in Subsection 5.2). The $\Omega(1/\sqrt{\epsilon})$ bound in [44] has been extended to the multi-queue case in [46] but still under strict convexity assumption and for a restrictive class of algorithms. Whether the $\Omega(\log(1/\epsilon))$ bound in [45] can be easily extended to multiple queues still remains a question.

5.2 Multiple Keywords, Multiple Clients and Multiple Webpage Slots

We now extend this lower bound to the original general model, which can have multiple keywords, multiple clients and multiple webpage slots.

It is easy to see that the "per-client revenue region" $\mathcal{C}$ in Definition 1 is a polytope, which can then be rewritten as

$$\mathcal{C} = \left\{ \boldsymbol{\lambda} \geq \mathbf{0} : \ \sum_i h_i^{(n)} \lambda_i \leq d^{(n)}, \ \forall \ 1 \leq n \leq L \right\}, \tag{5.8}$$

where $h_i^{(n)} \geq 0$ and $d^{(n)} > 0$ for all i and n. The outer boundary of the polytope $\mathcal{C}$ consists of the L hyperplanes, i.e., $\sum_i h_i^{(n)} \lambda_i = d^{(n)}$ for all $n \in [1, L]$.

Under condition (5.3), L is at least equal to the number of clients (i.e., number of budget constraints), so (5.8) gives a more precise description of the stability condition for this "multi-queue system," compared to the original definition of $\mathcal{C}$. Thus, corresponding to the normal vector of each hyperplane, we convert the original multi-queue system into a new one with L queues: For each $n \in [1, L]$, we first scale the i^{th} queue described in (5.2) by $h_i^{(n)}$, so

that it has a queue length equal to $h_i^{(n)}Q_i(k)$, with $h_i^{(n)}A_i(k)$ arrivals and $h_i^{(n)}\tilde{b}_i(k)$ potential departures in time slot k, for all i. Next, we treat $\sum_i h_i^{(n)}Q_i(k)$ as the n^{th} queue, and since any $\boldsymbol{\lambda} \in \mathcal{C}$ satisfies $\sum_i h_i^{(n)}\lambda_i \leq d^{(n)}$, its maximum achievable average departure rate equals $d^{(n)}$, where $d^{(n)} \leq \sum_i h_i^{(n)}b_i$, because the potential departure rate of each individual scaled queue may not be fully achieved when all of them are coupled together.

We then come up with the formal definition of the class of algorithms which achieves a "near-optimal" average revenue.

Definition 2 ("ϵ-Neighborhood" of the maximum) *Let $\boldsymbol{\lambda}^*$ be one optimal point in $\mathcal{C}$ such that $\sum_i \lambda_i^* = \bar{R}^*$. The ϵ-neighborhood of $\boldsymbol{\lambda}^*$ is defined as*

$$\mathcal{N}_\epsilon \triangleq \{\boldsymbol{\lambda}^\varpi \in \mathcal{C} \setminus \partial\mathcal{C} : \ 0 < N \cdot (\bar{R}^* - \bar{R}^\varpi) \leq \epsilon\}, \tag{5.9}$$

where $\partial\mathcal{C}$ represents the outer boundary of $\mathcal{C}$, and it should be noted that the average revenue is evaluated per time slot while $\boldsymbol{\lambda}$ is evaluated per N time slots. $\diamond$

Note that in the above definition, since $\boldsymbol{\lambda}^\varpi \in \mathcal{N}_\epsilon$ is not on any boundary, $\bar{R}^*$ is strictly larger than $\bar{R}^\varpi$, which is easy to see from some basic principles of linear programming.

The following theorem shows the universal lower bound $\Omega(\log(1/\epsilon))$ for the general case.

Theorem 8 *For any algorithm ϖ s.t. $\boldsymbol{\lambda}^\varpi \in \mathcal{N}_\epsilon$, we have*

$$\sum_{i=1}^{M} \bar{Q}_i^\varpi \geq \frac{\log(1/\epsilon) - C_2}{C_1} - 1,$$

where $\varphi \triangleq \Pr(\text{no query arrival in a budgeting cycle}) = (1-\nu)^N > 0$, $P_+ \triangleq \Pr(\tilde{b}_i(k) > 0, \ \forall i) > 0$, and

$$\begin{aligned} C_1 &\triangleq 2(1 - \log(\varphi P_+)) \cdot \max_{i,n} h_i^{(n)} \in (0, \infty), \\ C_2 &\triangleq \max\{\log(\max_{i,n} h_i^{(n)}), 0\} \in [0, \infty). \end{aligned} \tag{5.10}$$

$\diamond$

Proof We ignore the superscript ϖ for brevity. According to some basic principles of linear programming, an optimal point $\boldsymbol{\lambda}^*$ is at a corner of $\mathcal{C}$. If there are several optimal points, any convex combination of them is also optimal. Denote this optimal point sets as Λ^* and $\forall \lambda^* \in \Lambda^*$, $\exists\ n^* \in [1, L]$, s.t. $\sum_i h_i^{(n^*)} \lambda_i^* = d^{(n^*)}$.

Given a $\boldsymbol{\lambda} \in \mathcal{N}_\epsilon$, $\exists\ \boldsymbol{\theta}$ s.t. $\sum_i \theta_i = \sum_i \lambda_i^*$ and $\theta_i \geq \lambda_i$ for all i (but at least one inequality is strict). Besides, for this $\boldsymbol{\theta}$, $\exists\ \tilde{n} \in [1, L]$, s.t. $\sum_i h_i^{(\tilde{n})} \theta_i \geq d^{(\tilde{n})}$ (otherwise, $\boldsymbol{\theta} \in \mathcal{C} \setminus \partial\mathcal{C}$ will hold and hence $\sum_i \theta_i < \sum_i \lambda_i^*$, which leads to a contradiction). Therefore,

$$\begin{aligned} d^{(\tilde{n})} - \sum_i h_i^{(\tilde{n})} \lambda_i \leq & \sum_i h_i^{(\tilde{n})} (\theta_i - \lambda_i) \overset{(a)}{\leq} h_{max}^{(\tilde{n})} \sum_i (\theta_i - \lambda_i) \\ = & \ h_{max}^{(\tilde{n})} \sum_i (\lambda_i^* - \lambda_i) \leq h_{max}^{(\tilde{n})} \epsilon, \end{aligned} \tag{5.11}$$

where $h_{max}^{(\tilde{n})} \triangleq \max_i h_i^{(\tilde{n})} > 0$ and inequality (a) holds because $\theta_i \geq \lambda_i$ for all i. Letting $P'_+ \triangleq \Pr(\sum_i h_i^{(\tilde{n})} \tilde{b}_i(k) > 0)$, it is easy to see that $P'_+ \geq \Pr(\tilde{b}_i(k) > 0,\ \forall i) = P_+ > 0$. Together with Theorem 7, we can conclude that

$$\sum_i h_i^{(\tilde{n})} \bar{Q}_i \ \geq \ \frac{\log(1/\epsilon) - \log(h_{max}^{(\tilde{n})})}{2(1 - \log(\varphi P'_+))} - 1 \geq \frac{\log(1/\epsilon) - \log(h_{max}^{(\tilde{n})})}{2(1 - \log(\varphi P_+))} - 1.$$

Since $\sum_i h_i^{(\tilde{n})} \bar{Q}_i \leq h_{max}^{(\tilde{n})} \sum_i \bar{Q}_i$, it is further concluded that

$$\sum_i \bar{Q}_i \geq \frac{\log(1/\epsilon) - \log(h_{max}^{(\tilde{n})})}{2h_{max}^{(\tilde{n})}(1 - \log(\varphi P_+))} - 1 \geq \frac{\log(1/\epsilon) - C_2}{C_1} - 1,$$

where the universal constants are defined in (5.10), and it is guaranteed that $C_1 \in (0, \infty)$ and $C_2 \in [0, \infty)$. This completes the proof. ■

Remark 9 *We briefly explain the idea behind choosing θ in the above proof: For those $\boldsymbol{\lambda} \in \mathcal{N}_\epsilon$ such that $\lambda_i \leq \lambda_i^*$ for all i (at least one is strict), $\boldsymbol{\theta}$ can be directly chosen as $\boldsymbol{\lambda}^*$ to make inequality (a) in (5.11) hold. But for the other $\boldsymbol{\lambda} \in \mathcal{N}_\epsilon$ which do not satisfy the above condition, it is necessary to introduce a $\boldsymbol{\theta}$ other than $\boldsymbol{\lambda}^*$, which both lies on the "maximum*

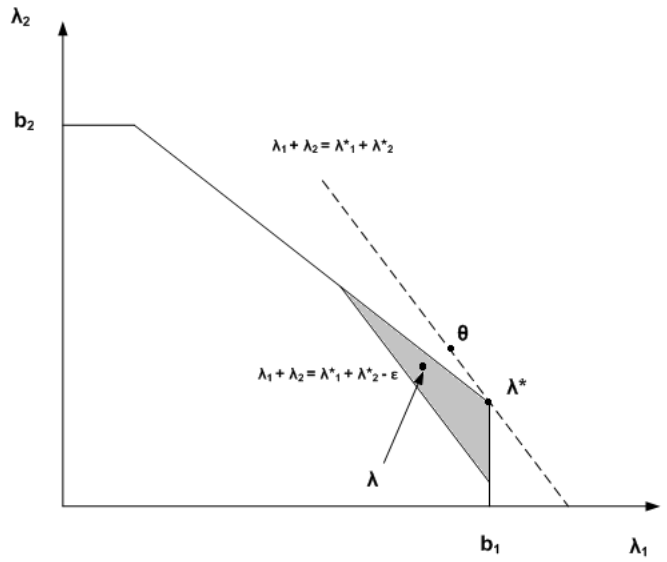


Figure 5.1: An illustration of the idea in the proof of Theorem 8

revenue line" (i.e., $\sum_i \theta_i = \sum_i \lambda_i^*$*) and dominates* $\boldsymbol{\lambda}$ *component-wise, in order to derive inequality (5.11). Note that* $\boldsymbol{\theta}$ *is not unique and furthermore,* $\boldsymbol{\theta}$ *lies either on* $\partial\mathcal{C}$ *or in the exterior of* $\mathcal{C}$ *and it can be chosen as a boundary point only if the optimal revenue point is not unique. Figure 5.1 illustrates this idea using an example with one keyword, two clients and one webpage slot, specifically for showing where such a* $\boldsymbol{\theta}$ *is located.* $\diamond$

The basic idea in our proof is to use Theorem 7 to first get a lower bound for those new single queues written as a "weighted sum" of the original queues (described above). This idea is similar to one part in the proof for the lower bound on the expected queue length of a departure-controlled multi-queue system in [47], but some techniques in their proof cannot directly apply to arrival-controlled queues like ours.

5.3 Tightness of the Lower Bound

We want to show that the $\Omega(\log(1/\epsilon))$ universal lower bound is tight, i.e., achievable by some algorithms. Consider the following simple queueing model: the arrival process $a(k)$ is i.i.d. across time, $a(k) = 2$ w.p. ν and $a(k) = 0$ otherwise. The service rate is constant and equal to 1. Assume that $\nu \in (1/2, 1)$. With the controlled arrival process $\hat{a}(k)$, we want to

achieve a throughput $E[\hat{a}(k)] \geq 1-\epsilon$ for a given small $\epsilon > 0$. A "threshold policy" based on a threshold T is proposed below:

- When $Q(k) > T$, reject all arrivals.
- When $Q(k) = T$, accept one arrival w.p. p_1, accept two arrivals w.p. p_2, and reject all of them otherwise.
- When $Q(k) < T$, accept all arrivals.

Defining π_i as the steady-state probability that $Q(k) = i$ $(0 \leq i \leq T+1)$ for the resulting Markov chain, the local balance equations are given below:

$$\begin{aligned} \pi_i \nu &= \pi_{i+1}(1-\nu), \ \forall\ 0 \leq i \leq T-2; \\ \pi_{_{T-1}} \cdot p_1 \nu &= \pi_{_T}(1-(p_1+p_2)\nu); \\ \pi_{_T} \cdot p_2 \nu &= \pi_{_{T+1}}; \\ \sum_{i=0}^{T+1} \pi_i &= 1. \end{aligned} \tag{5.12}$$

Combining these equations with the throughput requirement, we get

$$\nu \left[2 \sum_{i=0}^{T-1} \pi_i + \pi_{_T}(2p_2 + p_1) \right] = 1 - \epsilon, \tag{5.13}$$

and one can finally show that (ignoring detailed calculations)

$$T = \frac{\log(1/\epsilon) + \log C(\epsilon)}{\log\left(\frac{\nu}{1-\nu}\right)},$$

where

$$C(\epsilon) \triangleq \frac{(2\nu - 1 + \epsilon)(1 - \nu(p_1+p_2))}{\nu(2 - 2(1-\nu)p_2 - p_1)}.$$

The above result further implies that $\bar{Q} \sim \Theta(\log(1/\epsilon))$. We can also see that as $\nu \to 1$, $T \to 0$, which is consistent with the fact the lower bound given in Theorem 7 goes to 0 as the "zero arrival probability" $\varphi \to 0$.

Another example showing the tightness of an $\Omega(\log(1/\epsilon))$ bound is the dynamic packet dropping algorithm in [45] (note that this universal lower bound is proved based on a strict convexity assumption as mentioned before in Subsection 5.1).

Remark 10 *We note that the lower bound* $\Omega(\log(1/\epsilon))$ *is different from the upper bound which is* $O(1/\epsilon)$*. This is quite similar to related works in the case of wireless networks; however, the main difference here is that we have removed a crucial convexity assumption which makes the results for wireless networks inapplicable in our case. The natural question, therefore, is whether we can use strategies as in the case of wireless networks [37] to get tighter upper bounds. In particular, it has been shown in [37] that an* $O((\log(1/\epsilon))^2)$ *delay can be obtained by using a LIFO ("last in, first out") queueing discipline. However, in our model, delays do not have a meaning; only the overdraft has a meaning and the overdraft here is the equivalent of the queue length in [37]. In [37], while LIFO decreases the average delay, the queue length still continues to be of the form* $O(1/\epsilon)$*. Therefore, the ideas in [37] do not seem to apply to our model. The question of bridging the gap between the upper and lower bounds is an open problem.*

The key reason to prove a lower bound is that we have a controlled queueing system and so it is not clear a priori whether a non-zero overdraft has to be maintained. The lower bound proves that at least a $\log(1/\epsilon)$ *overdraft is necessary. On the other hand, the overdraft may or may not be a significant problem. It only results in "free" service to the clients but does not negatively impact them. From a service provider's perspective, free service may not be desirable but, on average, the overdraft is zero in the long run since the instantaneous overdraft is bounded in a stochastic sense in the steady state.* $\diamond$

CHAPTER 6

CONCLUSION

In this dissertation, we propose stochastic models and design optimal or near-optimal algorithms for resource allocation, in two popular network applications: video-on-demand services (supported by content distribution networks) and online advertising. We conclude the dissertation by summarizing the results and contributions for these two applications respectively.

6.1 Video-on-Demand Services in Content Distribution Networks

In content distribution networks which provide video-on-demand services, the information of content popularity can be utilized to design optimal content placement strategies, which minimizes the fraction of rejected requests in the CDN, or equivalently, maximizes the utilization of CDN servers' uplink bandwidth resources. We focused on CDNs where the network size is large. For the finite content catalog model, we proved the optimality of a proportional-to-product placement strategy in "service-only" CDNs (where requests are external), and proved optimality of "Hot-Warm-Cold" placement strategy in ISP-managed P2P CDNs (where requests are internal). For a certain large content catalog model, we also established that proportional-to-product placement strategy with a certain request acceptance policy leads to optimal performance in "service-only" CDNs, provided cache storage space per server grows unboundedly (although arbitrarily slowly with system size).

Many interesting questions remain. To name only two, more general popularity distributions (e.g. Zipf) for the large catalogue scenario could be investigated; the efficiency of adaptive cache update rules such as the one discussed in Section 2.3.1, or classical alternatives such as LRU, in conjunction with a loss network operation, also deserves more

detailed analysis. The complexity and necessity of performing "repacking" from a theoretical perspective is also an open question.

6.2 Online Advertising

For the second problem, "online advertising," we propose a stochastic model to describe how search service providers charge client companies based on users' queries for the keywords related to these companies' ads by using certain advertisement assignment strategies. We formulate an optimization problem to maximize the long-term average revenue for the service provider under each client's long-term average budget constraint, and design an online algorithm which captures the stochastic properties of users' queries and click-through behaviors. We solve the optimization problem by making connections to scheduling problems in wireless networks, queueing theory and stochastic networks. Our online algorithm is entirely oblivious to query arrivals and fully adaptive, so even non-stationary query arrival patterns and short-term clients can be handled.

With a small customizable parameter ϵ which is the step size used in each iteration of the online algorithm, we have shown that our online algorithm achieves a long-term average revenue which is within $O(\epsilon)$ of the optimal revenue and the overdraft level of this algorithm is upper bounded by $O(1/\epsilon)$. By allowing negative values for the length of overdraft queues, we can eliminate overdraft.

When estimated click-through rates instead of true ones are used in our online algorithm, we show that the achievable fraction of the offline optimal revenue is lower bounded by $\frac{1-\Delta}{1+\Delta}$, where Δ is the relative error in click-through rate estimation.

In another optimization formulation where the objective is to maximize the long-term average click-through rate and the constraints include a minimum impression requirement for each client, we further propose an approach to set impression requirements which make the contract feasible and limit the average accumulated under-service to clients. Simulations show that making queues update in a faster time scale will reduce both over-service and under-service, which benefits a system involving short-term clients.

In the context of revenue maximization problem, we further establish that in the long run,

an expected overdraft level of $\Omega(\log(1/\epsilon))$ is unavoidable (a universal lower bound) under any stationary ad assignment algorithm which achieves a long-term average revenue within $O(\epsilon)$ of the offline optimum. The tightness of this universal lower bound is also shown for a simple queueing model using a threshold policy. At this time, we do not know if there exist online ad assignment strategies that achieve the lower bound. This is an interesting open question.

APPENDIX A

ADDITIONAL ISSUES IN CDNS FOR VIDEO-ON-DEMAND SERVICES

A.1 Approximation to Proportional-to-Product Placement Using Bernoulli Sampling

An alternative sampling strategy to get the proportional-to-product placement is as follows:

To push contents to box b ($1 \leq b \leq B$), the server will

1. Generate C independent Bernoulli random variables $X_c \sim \text{Ber}(p_c)$ for all $c \in \mathcal{C}$, where $p_c = \beta\hat{\nu}_c/(1+\beta\hat{\nu}_c)$, $\hat{\nu}_c$ is the normalized version of ν_c, and β is a customized constant parameter.

2. If $\sum_{c\in\mathcal{C}} X_c = M$ (which means a valid cluster of size M is generated), push content c to box b if $X_c = 1$; Otherwise, go back to Step 1.

We now analyze why this scheme works: after generating a valid size-M subset, the probability that this subset is a certain subset $\mathcal{G}_j$ equals

$$\begin{aligned}
&\Pr(X_c = 1,\ \forall c \in \mathcal{G}_j;\ X_c = 0,\ \forall c \notin \mathcal{G}_j | \sum_{c\in\mathcal{C}} X_c = M) \\
&= \frac{\prod_{c\in\mathcal{G}_j} p_c \cdot \prod_{c\notin\mathcal{G}_j}(1-p_c)}{\Pr(\sum_{c\in\mathcal{C}} X_c = M)} \\
&= \prod_{c\in\mathcal{G}_j} \frac{p_c}{1-p_c} \cdot \left(\frac{\prod_{c\in\mathcal{C}} p_c}{\Pr(\sum_{c\in\mathcal{C}} X_c = M)}\right) \\
&= \prod_{c\in\mathcal{G}_j} \hat{\nu}_c/Z,
\end{aligned}$$

where $Z = \Pr(\sum_{c\in\mathcal{C}} X_c = M)/(\beta^M \prod_{c\in\mathcal{C}} p_c)$, which actually equals the normalizing factor for $\prod_{c\in\mathcal{G}_j} \hat{\nu}_c$.

We then consider the computational complexity of this approximation algorithm. Assuming that $\{\hat{\nu}_c\}$ is sorted in the descending order, we have

$$\begin{aligned}\Pr(\sum_{c\in\mathcal{C}} X_c = M) &\geq \prod_{c=1}^{M} p_c \cdot \prod_{c=M+1}^{C} (1-p_c) \\ &= \frac{\prod_{c=1}^{M} \beta\hat{\nu}_c}{\prod_{c=1}^{C}(1+\beta\hat{\nu}_c)} \triangleq P^*.\end{aligned}$$

So the computational complexity is upper bounded by $O(BC/P^*)$. Note that the constant parameter β can be adjusted to get a higher $\Pr(\sum_{c\in\mathcal{C}} X_c = M)$ in order to reduce computational complexity. To achieve this, we can just choose a β which maximizes its lower bound P^*, so

$$\frac{\partial \log P^*}{\partial \beta} = \frac{M}{\beta} - \sum_{c=1}^{C} \frac{\hat{\nu}_c}{1+\beta\hat{\nu}_c} = 0. \tag{A.1}$$

The server can use any numerical methods (e.g., Newton's method) to seek a root of equation (A.1). In fact, this lower bound P^* on $\Pr(\sum_{c\in\mathcal{C}} X_c = M)$ is not tight, since it is just the largest item in the sum expression. When the popularity is close to uniformity (e.g., in a zipf-like distribution, α is small), this largest item is no longer dominant, so the lower bound P^* is quite untight, which means we actually overestimate the computation complexity by only evaluating its upper bound. However, this will not affect the real gain we obtain after choosing the optimal β according to equation (A.1).

Recall that we also proposed a simple sampling strategy in Section 2.3.1. It is easy to see that when some contents are much more popular than the others (e.g., zipf-like α is large), the probability that duplicates appear in one size-M sample is high, hence largely increases the number of resampling. Thus, it would be faster if we choose the Bernoulli sampling. However, when the popularity is quite uniform, the simple sampling works very well. An extreme case is that under the uniform popularity distribution,

$$\Pr\{\text{a valid size-}M\text{ subset}\} = \frac{\binom{C}{M}\cdot M!}{C^M} = \prod_{i=1}^{M-1}\left(1-\frac{i}{C}\right),$$

which shows that when C is large, you can get a valid sample almost every time.

A.2 Detailed Implementation in the Simulations

A.2.1 A Heuristic Repacking Algorithm

We first describe the concept of "repacking." When the cache size $M = 1$, all the bandwidth resources at a certain box belongs to the content the box caches. When $M \geq 2$, however, this is not the case: all the contents cached in one box are actually competitors for the bandwidth resources at that box. Consider a simple example in which $B = 2$, $M = 2$ and $U = 1$: Box 1 which caches content 1 and 2 is serving a download of content 2, while box 2 which caches content 2 and 3 is idle. When a request for content 1 comes, the only potential candidate to serve it is box 1, but since the only connection is already occupied by a download of content 2, the request for content 1 has to be rejected. However, if this ongoing download can be "forwarded" to the idle box 2, the new request can be satisfied without breaking the old one. We call this type of forwarding "repacking."

In the feasibility condition (2.1) and its equivalent form (2.2), we actually allow perfect repacking to identify a feasible $\{n_c\}$. In a real system, perfect repacking needs to enumerate all the possible serving patterns and choose the best one based on some criterion, which is usually computationally infeasible. We then propose a heuristic repacking algorithm which is not so complex but can achieve similar functionality and improve performances, although imperfect.

Several variables need to be defined before we describe the algorithm:

- n_c: the system-wide ongoing downloads of content c, which does not count the downloads from the server.
- $\mathcal{B}_c^k$: The set of boxes which have content c ("potential candidate boxes") and k free connections, for $0 \leq k \leq U$.
- D_c: number of boxes which has content c. $D_c = \sum_{k=0}^{U} |\mathcal{B}_c^k|$.

- $\mathbf{u}_b$: a U-dimensional vector, of which the i-th component represents the content box b is using its i-th connection to upload (a value 0 represents a free connection).
- c_o: the "orphan content" which is affiliated with a new request or an ongoing download but has not been assigned with any box.
- $\mathcal{C}_o$: the set of contents which has once been chosen as orphan contents.
- t_R: the number of repacking already done.

Note that when choosing a box to serve a request, load balancing is already considered, which to some extent reduces the chance of necessary repacking in later operations. However, repacking is still needed for an incoming request for content c as soon as $\cup_{k>0}\mathcal{B}_c^k = \emptyset$.

Repacking Algorithm

After getting a request for content c while $\cup_{k>0}\mathcal{B}_c^k = \emptyset$, the server executes the following:

1. Initialize $c_o := c$, $\mathcal{C}_o := \{c\}$, and $t_R := 0$.
2. Let $\bar{\mathcal{C}} = \{c' : n_{c'}/D_{c'} > n_{c_o}/D_{c_o} \text{ and } c' \notin \mathcal{C}_o\}$, i.e., a set of contents which have not become orphans during this repacking process and of which the utilization factor (may be larger than 1) is larger than that of the current orphan content c_o. If $\bar{\mathcal{C}}_o = \emptyset$, regard c_o as a loss and TERMINATE.
3. Choose $c^* = \arg\max_{c' \in \bar{\mathcal{C}}}\{n_{c'}/D_{c'}\}$. Uniformly pick one (box, connection) pair from

$$\{(b, i) : \ b \in \mathcal{B}_c^0, \ c^* \text{ is the } i\text{-th component of } \mathbf{u}_b\}.$$

4. Use the chosen box b and its i-th connection to continue uploading the remaining part of content c_o. At the same time, c^* which was served using that connection becomes a new orphan, i.e., $c_o := c^*$. Update $\mathbf{u}_b$ and $\{n_c\}$. Set $t_R := t_R + 1$.
5. If $\cup_{k>0}\mathcal{B}_{c_o}^k \neq \emptyset$, i.e., there exists a free connection to serve the new c_o, then use the load-balancing-based box selection rule to select a box to continue uploading the remaining

part of c_o. The repacking process is perfect (no remaining orphan) and TERMINATE. Otherwise,

- If $t_R = t_R^{max}$, a customized algorithm parameter ($0 \leq t_R^{max} \leq C$), regard c_o as a loss and TERMINATE.
- Otherwise, set $\mathcal{C}_o := \mathcal{C}_o + \{c_o\}$, and go to Step 2.

A.2.2 A Practical Issue in Cache Update

When a box b is chosen for cache update (and it does not hold the content c corresponding to the request), it might still be uploading content c' which is to be replaced. This fact is not captured by the Markov chain model. In practice, those ongoing services must be terminated. Since we have introduced the repacking scheme, they become "orphans" ready for repacking. We implement the procedure as follows:

1. Rank these orphans by their remaining service time in the ascending order, i.e., the original download which is sooner to be completed is given higher priority.
2. Do repacking one by one until one orphan fails to be repacked. Note that here the repacking algorithm starts from Step 5, since there may already be some boxes with both content c and free connections.

A.3 Storage of Segments and Parallel Substreaming

We have mentioned before that compared to the "storage of complete contents and downloads by single streaming" setting, a more widely used mechanism in practice is that each box stores one specific segment of a video content and a download (streaming) comprises parallel substreaming from different boxes. To model this mechanism, we have the following simplifying assumptions: Each content is divided into K segments with equal length which are independently stored. Each box can store up to M segments (actually it does not matter

if we keep the original storage space of each box, i.e., M complete contents, which now can hold MK segments, since the storage space is a customized parameter) and these M segments do not necessarily belong to M distinct contents. The bandwidth of each box is kept as U, so now each box can accommodate UK parallel substreaming, each with download rate $1/K$ (the average service duration is still kept as 1 because each segment is $1/K$ of the original content length). The definition of "traffic load" ρ is then the same as in equation (2.6). A request for a content will be divided into sub-requests submitted to the boxes holding those corresponding segments of this content, generating K parallel substreaming flows in total (one box can serve more than one substreaming service for this request if it caches more than one distinct segments of this content).

Let θ represent a segment and $\theta \in c$ indicate that θ is a segment of content c. Recall that we use n_c to denote the number of concurrent downloads (now called "streams") of content c in the network. We further use n_θ to denote the number of substreams corresponding to segment θ.

Now the original feasibility constraint (2.1) becomes

$$\begin{aligned} \sum_{b:\ \theta \in \mathcal{J}_b} z_{\theta b} &= n_\theta,\ \forall\ \theta \in \Theta; \\ \sum_{\theta:\ \theta \in \mathcal{J}_b} z_{\theta b} &\leq UK,\ \forall\ b \in \mathcal{B}, \end{aligned} \tag{A.2}$$

where Θ represents the whole set of segments and $z_{\theta b}$ denotes the number of concurrent substreams downloading segment θ from box b. It is easy to see that the equivalent version which can be proved by Hall's theorem becomes:

$$\forall\ \mathcal{S} \subseteq \Theta,\ \sum_{\theta \in \mathcal{S}} n_\theta \leq KU\left|\{b \in \mathcal{B}:\ \mathcal{S} \cap \mathcal{J}_b \neq \emptyset\}\right|, \tag{A.3}$$

where with a little abuse of notation, $\mathcal{S}$ is used to denote a subset of Θ, instead of $\mathcal{C}$ as before.

Since we have assumed that video duration and video streaming rate are all the same, one naturally has $n_\theta = n_c$ for all $\theta \in c$. If we let randomness exist in the service duration, then

within one stream, some substreams may complete earlier than the others. Therefore, the above equality needs to be added as a constraint (and used to come up with the following result), i.e., the bandwidth for the K substreams should be reserved until the whole streaming is completed.

Then, in the proof of the optimality of "proportional-to-product" placement for DSN, every expression keeps the same, except that the feasibility constraint (2.10) is changed to

$$\forall\ \mathcal{S} \subseteq \Theta,\ \sum_{\theta \in \Theta} \sum_{c:\theta \in c} x_c^{(B)} \leq \sum_{j:j \cap \mathcal{S} \neq \emptyset} m_j BUK, \tag{A.4}$$

and the "proportional-to-product" placement $\{m_j\}$ is now with respect to each segment, i.e., $m_j = \prod_{\theta \in j} \hat{\nu}_\theta / Z$ for all $j \subseteq \Theta$ s.t. $|j| = M$, where Z is the normalizing constant and $\hat{\nu}_\theta = \hat{\nu}_c$ if $\theta \in c$. With an observation that $\sum_{\theta \in \Theta} \hat{\nu}_\theta = K \sum_{c \in \mathcal{C}} \hat{\nu}_c = K$, we can still come to an inequality same with inequality (2.17), except that c and $\mathcal{C}$ are replaced by θ and Θ respectively. All the succeeding steps are exactly the same in the proof of optimality.

APPENDIX B

PROOFS

B.1 Proof of Equivalence between Feasibility Conditions (2.1) and (2.2)

B.1.1 Sufficiency of Condition (2.2)

We use Hall's theorem to prove the sufficiency.

[Hall's theorem] Suppose $\mathcal{J} = \{J_1, J_2, \cdots\}$ is a collection of sets (not necessarily countable). A SDR ("System of Distinct Representatives") for $\mathcal{J}$ is defined as $X = \{x_1, x_2, \cdots\}$, where $x_i \in J_i$. Then, there exists a SDR (not necessarily unique) iff. $\mathcal{J}$ meets the following condition:

$$\forall\ \mathcal{T} \subseteq \mathcal{J},\ |\mathcal{T}| \leq |\bigcup_{A \in \mathcal{T}} A|. \tag{B.1}$$

$\diamond$

In our VoD CDN system, denote the content set as $\mathcal{C} = \{c_1, c_2, \cdots, c_N\}$. Given the ongoing download services of each content $\{n_i\}_{i=1}^{N}$, we get a "distinguishable content set"

$$\begin{aligned}\bar{\mathcal{C}} \ = \ & \{c_1^{(1)}, c_1^{(2)}, \cdots, c_1^{(n_1)}; c_2^{(1)}, c_2^{(2)}, \cdots, c_1^{(n_2)}; \cdots; \\ & c_N^{(1)}, c_N^{(2)}, \cdots, c_N^{(n_N)}\},\end{aligned}$$

where $c_i^{(k)}$ represents the k-th download service of content i for $1 \leq k \leq n_i$, and has its "potential connection set"

$$J_i^{(k)} = \{l_b^{(j)} :\ 1 \leq j \leq U,\ c_i \in b,\ b \in \mathcal{B}\},$$

i.e., the set of all the connections of those boxes which have content c_i. A collection of the "potential connection sets" for all $\{c_i^{(k)}\}$ is then

$$\mathcal{J} = \{J_1^{(1)}, J_1^{(2)}, \cdots, J_1^{(n_1)}; \cdots; J_N^{(1)}, J_N^{(2)}, \cdots, J_N^{(n_N)}\},$$

and a SDR for $\mathcal{S}$ is

$$X = \{x_1^{(1)}, x_1^{(2)}, \cdots, x_1^{(n_1)}; \cdots; x_N^{(1)}, x_N^{(2)}, \cdots, x_N^{(n_N)}\},$$

s.t. $x_i^{(k)} \in J_i^{(k)}$, which means each $c_i^{(k)}$ is affiliated with a distinct connection (i.e., a feasible solution in our model).

Now we want to prove the existence of such a SDR, i.e., to prove equation (B.1). For $\forall\ \mathcal{T} \subseteq \mathcal{J}$, there is a one-to-one mapping between $\mathcal{T}$ and a $\bar{\mathcal{S}} \subseteq \bar{\mathcal{C}}$. Further, this $\bar{\mathcal{S}}$ can be mapped to a $\mathcal{S} \subseteq \mathcal{C}$ where

$$\mathcal{S} = \{c_i :\ \exists 1 \leq k \leq n_i,\ s.t.\ c_i^{(k)} \in \bar{\mathcal{S}}\},$$

i.e., $\mathcal{S}$ is the set of all contents considered in $\bar{\mathcal{S}}$ without considering multiple services of each content. Then, $\forall\ \mathcal{T} \subseteq \mathcal{J}$,

$$\begin{aligned} \text{RHS} &= |\bigcup_{J_i^{(k)} \in \mathcal{T}} J_i^{(k)}| = \sum_{b: \exists c_i \in \mathcal{S}\ s.t.\ c_i \in b} U \\ &= U\,|\{b \in \mathcal{B} :\ \mathcal{S} \cap \mathcal{J}_b \neq \emptyset\}| \end{aligned}$$

and

$$\text{LHS} = |\mathcal{T}| = |\bar{\mathcal{S}}| \leq \sum_{c_i \in \mathcal{S}} n_i.$$

Therefore, if

$$\forall S \subseteq \mathcal{C}, \sum_{c_i \in \mathcal{S}} n_i \leq = U\,|\{b \in \mathcal{B} :\ \mathcal{S} \cap \mathcal{J}_b \neq \emptyset\}|$$

holds, then equation (B.1) holds. The sufficiency is proved.

B.1.2 Necessity of Condition (2.2)

For any $\mathcal{S} \subseteq \mathcal{C}$,

$$\begin{aligned}\sum_{c\in\mathcal{S}} n_c &= \sum_{c\in\mathcal{S}}\sum_{b:c\in\mathcal{J}_b} Z_{cb} = \sum_{\substack{b:\ \exists c\in\mathcal{S}\\ s.t.\ \ c\in\mathcal{J}_b}}\ \sum_{c\in\mathcal{S}\cap\mathcal{J}_b} Z_{cb} \\ &\overset{(a)}{\leq} \sum_{b:\ \exists c\in\mathcal{S}\ \ s.t.\ \ c\in\mathcal{J}_b} U = U\left|\{b\in\mathcal{B}:\ \mathcal{S}\cap\mathcal{J}_b\neq\emptyset\}\right|,\end{aligned}$$

where the inequality (a) is due to the second constraint in condition (2.1). Hence, the necessity is proved.

B.2 Proof of Theorem 2

The Lagrangian of OPT 2 is

$$\begin{aligned}& L(\tilde{\mathbf{m}},\boldsymbol{\lambda},\mathbf{x};\mathbf{u},\mathbf{v},\mathbf{y},\mathbf{z},\mathbf{w},\eta,\gamma) \\ =\ & \sum_{c\in\mathcal{C}}\Big[\rho_c\tilde{m}_c + x_c - u_c(\tilde{m}_c-1) - v_c(\lambda_c-\tilde{m}_c) \\ & -y_c(x_c-\lambda_c) - z_c(x_c-\rho_c+\rho_c\tilde{m}_c) + w_c\lambda_c\Big] \\ & -\eta\left(\sum_{c\in\mathcal{C}}\lambda_c - 1\right) - \gamma\left(\sum_{c\in\mathcal{C}}\tilde{m}_c - M\right).\end{aligned}$$

The KKT condition includes the feasible set defined in OPT 2 and the following:

$$\begin{aligned}
\frac{\partial L}{\partial x_c} = 1 - y_c - z_c &= 0,\ \forall c; \\
\frac{\partial L}{\partial \tilde{m}_c} = \rho_c - u_c + v_c - \rho_c z_c - \gamma &= 0,\ \forall c; \\
\frac{\partial L}{\partial \lambda_c} = -v_c + y_c - \eta + w_c &= 0,\ \forall c; \\
u_c(\tilde{m}_c - 1) &= 0,\ u_c \geq 0,\ \forall c; \\
v_c(\lambda_c - \tilde{m}_c) &= 0,\ v_c \geq 0,\ \forall c; \\
y_c(x_c - \lambda_c) &= 0,\ y_c \geq 0,\ \forall c; \\
z_c(x_c - \rho_c + \rho_c \tilde{m}_c) &= 0,\ z_c \geq 0,\ \forall c; \\
w_c \lambda_c &= 0,\ w_c \geq 0,\ \forall c.
\end{aligned}$$

We then put the solution stated in the theorem into KKT condition to check whether the condition is satisfied. The analysis is as follows:

- For $1 \leq c \leq M-1$, since $\tilde{m}_c = 1$ and $\lambda_c = x_c = 0$, we obtain that $v_c = 0$, $y_c + z_c = 1$, $\rho_c(1 - z_c) = u_c + \gamma$, and $y_c = \eta - w_c$. Letting $w_c = 0$, we further have: $u_c = \rho_c \eta - \gamma,\ y_c = \eta,\ z_c = 1 - \eta$. To keep $u_c, y_c, z_c \geq 0$, we must have $\eta \in [0, 1]$ and $\gamma \leq \rho_c \eta$, for $1 \leq c \leq M-1$. Thus, since $\{\rho_c\}$ are also ranked in the descending order, we have

$$\gamma \leq \rho_{M-1} \eta. \tag{B.2}$$

- For $M \leq c \leq c^*$, since $\tilde{m}_c = \lambda_c = x_c = \rho_c/(1+\rho_c)$, we obtain that $u_c = w_c = 0,\ y_c + z_c = 1,\ \rho_c(1 - z_c) = \gamma - v_c,\ y_c = \eta + v_c$. We further have:

$$v_c = \frac{\gamma - \rho_c \eta}{\rho_c + 1},\ y_c = \frac{\eta + \gamma}{\rho_c + 1},\ z_c = 1 - \frac{\eta + \gamma}{\rho_c + 1}.$$

  To keep $v_c, y_c, z_c \geq 0$, we must have $\rho_c \eta \leq \gamma \leq \rho_c + 1 - \eta,$ for $M \leq c \leq c^*$. Thus,

$$\rho_M \eta \leq \gamma \leq \rho_{c^*} + 1 - \eta. \tag{B.3}$$

- For $c = c^* + 1$, when $m_c = 0$, it degenerates to the next case. When $\tilde{m}_c > 0$, since $\tilde{m}_c = \lambda_c = x_c < \rho_c(1 - \tilde{m}_c)$, we obtain that $u_c = w_c = z_c = 0,\ y_c = 1,\ \rho_c + v_c = \gamma,\ \eta + v_c = 1$. We further have

$$\gamma = \rho_{c^*+1} + 1 - \eta. \tag{B.4}$$

- For $c^* + 2 \leq c \leq C$, since $\tilde{m}_c = \lambda_c = x_c = 0$, we obtain that $u_c = z_c = 0,\ y_c = 1,\ v_c = \gamma - \rho_c,\ w_c = \eta + v_c - 1 = \eta + \gamma - \rho_c - 1$. To keep $v_c, w_c \geq 0$, and due to the fact that $\eta \in [0, 1]$, we must have $\gamma \geq \rho_c,$ for $c^* + 2 \leq c \leq C$. Thus,

$$\gamma \geq \rho_{c^*+2}. \tag{B.5}$$

For inequalities (B.2), (B.3), (B.5) and equation (B.4) to hold simultaneously, we can choose a η which satisfies

$$\frac{\rho_{c^*+1} + 1}{\rho_{M-1} + 1} \leq \eta \leq \frac{\rho_{c^*+1} + 1}{\rho_M + 1},$$

which also satisfies $\eta \in [0, 1]$. Therefore, the theorem is proved.

It should be mentioned that when $\sum_{c=M}^{c^*} \tilde{m}_c = 1$, i.e., $\tilde{m}_{c^*+1} = 0$, the case "$c = c^* + 1$" can be combined with the next case "$c^* + 2 \leq c \leq C$", hence equation (B.4) does not exist while inequality (B.5) is changed to $\gamma \geq \rho_{c^*+1}$. Then, we can just choose a η which satisfies

$$0 \leq \eta \leq \frac{\rho_{c^*+1} + 1}{\rho_M + 1}.$$

B.3 Another Approach to Bound the Chance of "Good Contents" in Proving Theorem 3

At the first stage of proving Theorem 3, we mentioned that we can also directly derive the Chernoff bound on the RHS of inequality (3.5) to get the result. The derivation is given below:

Recall that $I(x) = \sup_\theta\{x\theta - \ln(\mathbb{E}[e^{\theta Z_i}])\}$ is the Cramér transform of the Bernoulli random

variable Z_i. It is easy to check that

$$I(x) = \begin{cases} a \ln\left(\frac{x}{p}\right) + (1-x)\ln\left(\frac{1-x}{1-p}\right) & \text{if } x \in [0,1] \\ +\infty & \text{else} \end{cases}.$$

Also recall that $a \triangleq \left(M^{2/3} + \frac{\nu_i M}{\rho U}\right)/MB = M^{-1/3}/B + p$, where $p \triangleq \frac{\nu_i}{\rho BU}$. Since we are considering a large B, $a \in [0,1]$ holds. Thus, denoting $\bar{p} = 1-p$ for brevity, the exponent of RHS of inequality (3.5) reads

$$\begin{aligned} & -MB \cdot I(a) \\ = \; & -(pMB + M^{2/3}) \cdot \ln\left(1 + \frac{1}{pM^{1/3}B}\right) \\ & -(\bar{p}MB - M^{2/3}) \cdot \ln\left(1 - \frac{1}{\bar{p}M^{1/3}B}\right) \\ = \; & -\frac{pMB + M^{2/3}}{pM^{1/3}B} + \frac{pMB}{2(pM^{1/3}B)^2} \\ & +\frac{\bar{p}MB - M^{2/3}}{\bar{p}M^{1/3}B} + \frac{\bar{p}MB}{2(\bar{p}M^{1/3}B)^2} + o(M^{1/3}) \\ = \; & -\frac{M^{1/3}}{2B}\left(\frac{1}{p} + \frac{1}{\bar{p}}\right) + o(M^{1/3}) \\ = \; & -\frac{M^{1/3}}{2}\left(\frac{\rho U}{\nu_i} + \frac{1}{B(1-\frac{\nu_i}{\rho U})}\right) + o(M^{1/3}) \\ = \; & -\Theta\left(M^{1/3}\right). \end{aligned} \tag{B.6}$$

With similar steps as above, we can show the exponent of the RHS of inequality (3.6) is also $-\Theta\left(M^{1/3}\right)$. Therefore, inequality (3.7) is proved.

B.4 Proof of Lemma 3

$$
\begin{aligned}
& E[V(\mathbf{Q}(k+1))|\mathbf{Q}(k)=\mathbf{Q}] - V(\mathbf{Q}) \\
= \;& \frac{1}{2}E\left[\sum_i \left(\left[Q_i + A_i(k,\mathbf{Q},\mathbf{u}(k)) - \tilde{b}_i(k)\right]^+\right)^2 - Q_i^2\right] \\
\leq \;& \frac{1}{2}E\left[\sum_i \left(Q_i + A_i(k,\mathbf{Q},\mathbf{u}(k)) - \tilde{b}_i(k)\right)^2 - Q_i^2\right] \\
= \;& E\left[\sum_i Q_i\left(A_i(k,\mathbf{Q},\mathbf{u}(k)) - \tilde{b}_i(k)\right) + \frac{1}{2}\sum_i \left(A_i(k,\mathbf{Q},\mathbf{u}(k)) - \tilde{b}_i(k)\right)^2\right] \\
\leq \;& \sum_i Q_i\left(\lambda_i(k,\mathbf{Q}) - b_i\right) + \frac{1}{2}\sum_i (E[A_i^2(k,\mathbf{Q},\mathbf{u}(k))] + E[\tilde{b}_i^2(k)]), \qquad \text{(B.7)}
\end{aligned}
$$

where it was already defined in equation (4.8) that for all i,

$$
A_i(k,\mathbf{Q}(k),\mathbf{u}(k)) = \sum_{t=kN}^{kN+N-1}\sum_s [\tilde{M}^*(t,\tilde{q}(t),\mathbf{Q}(k))]_{is}\cdot \tilde{c}_{\tilde{q}(t)is}(t)\cdot r_{\tilde{q}(t)i}.
$$

We further define

$$
\lambda_i(k,\mathbf{Q}(k)) \;\triangleq\; E[A_i(k,\mathbf{Q}(k),\mathbf{u}(k))|\mathbf{Q}(k)] = N\sum_q \nu_q \sum_s [\tilde{M}^*(q,t,\mathbf{Q}(k))]_{is} c_{qis} r_{qi}.
$$

Since each client can at most get one webpage slot for each query, we can further bound

$$
\sum_i A_i^2(k,\mathbf{Q},\mathbf{u}(k)) \leq (N(N-1)L^2 + NL)(\arg\max_{q,i,s}\{c_{qis}r_{qi}\})^2.
$$

Besides,

$$
E[b_i^2(k)] \;=\; \lceil b_i\rceil^2\varrho_i + \lfloor b_i\rfloor^2(1-\varrho_i) = \lceil b_i\rceil^2(b_i - \lfloor b_i\rfloor) + \lfloor b_i\rfloor^2(1-b_i+\lfloor b_i\rfloor).
$$

Thus, by defining

$$
B_1 \triangleq \frac{1}{2}\Big((N(N-1)L^2 + NL)(\arg\max_{q,i,s}\{c_{qis}r_{qi}\})^2 + \sum_i \lceil b_i\rceil^2(b_i - \lfloor b_i\rfloor) + \lfloor b_i\rfloor^2(1-b_i+\lfloor b_i\rfloor)\Big),
$$

and continuing from inequality (B.7), we have

$$
\begin{aligned}
& E[V(\mathbf{Q}(k+1))|\mathbf{Q}(k)=\mathbf{Q}] - V(\mathbf{Q}) \\
\leq\ & N\sum_q \nu_q \sum_{i,s} Q_i[\tilde{M}^*(q,t,\mathbf{Q})]_{is} c_{qis} r_{qi} - \sum_i Q_i b_i + B_1 \\
=\ & -N\sum_q \nu_q \sum_{i,s} \left(\frac{1}{\epsilon} - Q_i\right)[\tilde{M}^*(q,t,\mathbf{Q})]_{is} c_{qis} r_{qi} \\
& + \frac{N}{\epsilon}\sum_q \nu_q \sum_{i,s} [\tilde{M}^*(q,t,\mathbf{Q})]_{is} c_{qis} r_{qi} + B_1 - \sum_i Q_i b_i \\
=\ & -N\sum_q \nu_q \sum_{i,s} \left(\frac{1}{\epsilon} - Q_i\right)[\tilde{M}^*(q,t,\mathbf{Q})]_{is} c_{qis} r_{qi} \\
& + \frac{N}{\epsilon}\bar{R}(\tilde{\mathbf{p}}^*(k,\mathbf{Q})) + B_1 - \sum_i Q_i b_i \qquad \text{(B.8)} \\
\overset{(a)}{\leq}\ & -N\sum_q \nu_q \sum_{i,s} \left(\frac{1}{\epsilon} - Q_i\right) \sum_{M\in\mathcal{M}_q} p^*_{qM} M_{is} c_{qis} r_{qi} + \frac{N}{\epsilon}\bar{R}(\tilde{\mathbf{p}}^*(k,\mathbf{Q})) + B_1 - \sum_i Q_i b_i \\
=\ & -\frac{N}{\epsilon}\left(\bar{R}(\mathbf{p}^*) - \bar{R}(\tilde{\mathbf{p}}^*(k,\mathbf{Q}))\right) + B_1 \\
& - \sum_i Q_i \cdot \left(b_i - N\sum_q \nu_q \sum_{M\in\mathcal{M}_q} p^*_{qM} \sum_s M_{is} c_{qis} r_{qi}\right), \qquad \text{(B.9)}
\end{aligned}
$$

where inequality (a) holds because equation (4.6) in the online algorithm is equivalent to

$$
\forall q, \quad \tilde{\mathbf{p}}^*_q(k,\mathbf{Q}(k)) \in \arg\max_{\substack{\{p_{qM},\\ M\in\mathcal{M}_q\}}} \sum_{M\in\mathcal{M}_q} p_{qM} \sum_{i,s} M_{is} c_{qis} r_{qi} \left(\frac{1}{\epsilon} - Q_i(k)\right), \qquad \text{(B.10)}
$$

which means that evaluating the objective function in (B.10) with $\mathbf{p} = \mathbf{p}^*$ cannot achieve a larger value. Letting

$$
B_2 \triangleq \min_i \{b_i - N\sum_q \nu_q \sum_{M\in\mathcal{M}_q} p^*_{qM} \sum_s M_{is} c_{qis} r_{qi}\},
$$

from inequality (B.9), we finally obtain

$$E[V(\mathbf{Q}(k+1))|\mathbf{Q}(k)=\mathbf{Q}]-V(\mathbf{Q})\leq-\frac{N}{\epsilon}\left(\bar{R}(\mathbf{p}^*)-\bar{R}(\tilde{\mathbf{p}}^*(k,\mathbf{Q}))\right)+B_1-B_2\sum_i Q_i.$$

B.5 Proof of Theorem 4

The first inequality which shows that the online algorithm cannot do better than the offline optimal solution is too obvious, so we just ignore it here (proving it in a very rigorous way is also very easy, after defining the "per-client revenue region" in Subsection 5.2 and then using the fact that the average revenue vector $\boldsymbol{\lambda}$ corresponding to our online algorithm falls inside that region, according to inequality (4.9) which is implied by stability).

We now focus on the second inequality, i.e., the $O(\epsilon)$ convergence bound. From Lemma 3,

$$\begin{aligned}
& E\left[\bar{R}(\mathbf{p}^*)-\bar{R}(\tilde{\mathbf{p}}^*(k,\mathbf{Q}(k)))\right] \\
\leq\ & \frac{\epsilon}{N}\cdot E\left[B_1-B_2\sum_i \mathbf{Q}(k)+V(\mathbf{Q}(k))-E[V(\mathbf{Q}(k+1))|\mathbf{Q}(k)]\right] \\
\leq\ & \frac{\epsilon}{N}\cdot(B_1-E[V(\mathbf{Q}(k))]-E[V(\mathbf{Q}(k+1))]).
\end{aligned}$$

Adding the terms for $0\leq k\leq K-1$ and dividing by K, we get

$$\begin{aligned}
\frac{1}{K}\sum_{k=0}^{K-1}E\left[\bar{R}(\mathbf{p}^*)-\bar{R}(\tilde{\mathbf{p}}^*(k,\mathbf{Q}(k)))\right] &\leq \frac{\epsilon}{N}\left(B_1-\frac{E[V(\mathbf{Q}(K))]}{K}+\frac{V(\mathbf{Q}(0))}{K}\right) \\
&\leq \frac{\epsilon}{N}\left(B_1+\frac{V(\mathbf{Q}(0))}{K}\right).
\end{aligned}$$

Since $V(\mathbf{Q}(0))<\infty$, we get the following limit expression:

$$\lim_{K\to\infty}\frac{1}{K}\sum_{k=0}^{K-1}E\left[\bar{R}(\mathbf{p}^*)-\bar{R}(\tilde{\mathbf{p}}^*(k,\mathbf{Q}(k)))\right]\leq\frac{B_1\epsilon}{N}. \tag{B.11}$$

Finally, because

$$E\left[N\bar{R}(\mathbf{p}^*)-R(k)\right]=E\left[E\left[N\bar{R}(\mathbf{p}^*)-R(k)|\mathbf{Q}(k)\right]\right]=N\cdot E\left[\bar{R}(\mathbf{p}^*)-\bar{R}(\tilde{\mathbf{p}}^*(k,\mathbf{Q}(k)))\right],$$

inequality (B.11) is equivalent to

$$\lim_{K\to\infty} E\left[\bar{R}(\mathbf{p}^*) - \frac{1}{KN}\sum_{k=0}^{K-1} R(k)\right] \le \frac{B_1\epsilon}{N}.$$

B.6 Proof of Corollary 3

Continuing from inequality (B.8) in Appendix B.4 (the proof of Lemma 3), we get

$$\begin{aligned}
&E[V(\mathbf{Q}(k+1))|\mathbf{Q}(k)=\mathbf{Q}] - V(\mathbf{Q}) \\
&\overset{(a)}{\le} -\frac{1}{1+\Delta}N\sum_q \nu_q \sum_{i,s}\left(\frac{1}{\epsilon} - Q_i\right)[\tilde{M}^*(q,t,\mathbf{Q})]_{is}\hat{c}_{qis}r_{qi} + \frac{N}{\epsilon}\bar{R}(\tilde{\mathbf{p}}^*(k,\mathbf{Q})) + B_1 - \sum_i Q_i b_i \\
&\overset{(b)}{\le} -\frac{1}{1+\Delta}N\sum_q \nu_q \sum_{i,s}\left(\frac{1}{\epsilon} - Q_i\right)\sum_{M\in\mathcal{M}_q} p^*_{qM} M_{is}\hat{c}_{qis}r_{qi} + \frac{N}{\epsilon}\bar{R}(\tilde{\mathbf{p}}^*(k,\mathbf{Q})) + B_1 - \sum_i Q_i b_i \\
&\overset{(c)}{\le} -\frac{1-\Delta}{1+\Delta}N\sum_q \nu_q \sum_{i,s}\left(\frac{1}{\epsilon} - Q_i\right)\sum_{M\in\mathcal{M}_q} p^*_{qM} M_{is}c_{qis}r_{qi} + \frac{N}{\epsilon}\bar{R}(\tilde{\mathbf{p}}^*(k,\mathbf{Q})) + B_1 - \sum_i Q_i b_i \\
&= -\frac{N}{\epsilon}\left(\frac{1-\Delta}{1+\Delta}\bar{R}(\mathbf{p}^*) - \bar{R}(\tilde{\mathbf{p}}^*(k,\mathbf{Q}))\right) + B_1 \\
&\quad -\sum_i Q_i \cdot \left(b_i - \frac{1-\Delta}{1+\Delta}N\sum_q \nu_q \sum_{M\in\mathcal{M}_q} p^*_{qM}\sum_s M_{is}c_{qis}r_{qi}\right).
\end{aligned} \tag{B.12}$$

Here, inequalities (a) and (c) hold respectively because $\hat{\mathbf{c}} \le \mathbf{c}(1+\Delta)$ and $\hat{\mathbf{c}} \ge \mathbf{c}(1-\Delta)$, with the fact that all the coefficients in this summation are nonnegative. Inequality (b) holds because equation (4.13) in the online algorithm with estimated click-through rates is equivalent to

$$\forall q, \quad \tilde{\mathbf{p}}^*_q(k,\mathbf{Q}(k)) \in \arg\max_{\substack{\{p_{qM},\\ M\in\mathcal{M}_q\}}} \sum_{M\in\mathcal{M}_q} p_{qM}\sum_{i,s} M_{is}\hat{c}_{qis}r_{qi}\left(\frac{1}{\epsilon} - Q_i(k)\right), \tag{B.13}$$

which means that evaluating the objective function in (B.13) with $\mathbf{p} = \mathbf{p}^*$ cannot achieve a larger value. Letting

$$B_2' \triangleq \min_i \left\{ b_i - \frac{1-\Delta}{1+\Delta} \cdot N \sum_q \nu_q \sum_{M \in \mathcal{M}_q} p^*_{qM} \sum_s M_{is} c_{qis} r_{qi} \right\},$$

from inequality (B.9), we finally obtain

$$E[V(\mathbf{Q}(k+1))|\mathbf{Q}(k) = \mathbf{Q}] - V(\mathbf{Q}) \leq -\frac{N}{\epsilon} \left(\frac{1-\Delta}{1+\Delta} \bar{R}(\mathbf{p}^*) - \bar{R}(\tilde{\mathbf{p}}^*(k, \mathbf{Q})) \right) + B_1 - B_2' \sum_i Q_i.$$

Therefore, similarly as in the proof of Theorem 4, we can finally show that

$$\lim_{K \to \infty} E\left[\frac{1}{KN} \sum_{k=0}^{K-1} R(k) \right] \;\geq\; \left(\frac{1-\Delta}{1+\Delta} \right) \cdot \bar{R}(\mathbf{p}^*) - \frac{B_1 \epsilon}{N}.$$

B.7 Proof of Lemma 4

By a similar approach as in the proof of Lemma 3 (Appendix B.4), we have

$$\begin{aligned}
& E[V(\mathbf{Q}(k+1))|\mathbf{Q}(k) = \mathbf{Q}] - V(\mathbf{Q}) \\
\leq\; & -N \sum_q \nu_q \sum_{i,s} \left(Q_i + \frac{c_{qis}}{\epsilon} \right) [\tilde{M}^*(q,t,\mathbf{Q})]_{is} + \frac{N}{\epsilon} \bar{J}(\tilde{\mathbf{p}}^*(k,\mathbf{Q})) + D_1 + \sum_i Q_i m_i \\
\overset{(a)}{\leq}\; & -N \sum_q \nu_q \sum_{i,s} \left(Q_i + \frac{c_{qis}}{\epsilon} \right) \sum_{M \in \mathcal{M}_q} \hat{p}_{qM} M_{is} + \frac{N}{\epsilon} \bar{J}(\tilde{\mathbf{p}}^*(k,\mathbf{Q})) + D_1 + \sum_i Q_i m_i \\
=\; & -\frac{N}{\epsilon} \left(\bar{J}(\hat{\mathbf{p}}) - \bar{J}(\tilde{\mathbf{p}}^*(k,\mathbf{Q})) \right) + D_1 \\
& - \sum_i Q_i \left(N \sum_q \nu_q \sum_{M \in \mathcal{M}_q} \hat{p}_{qM} \sum_s M_{is} - m_i \right),
\end{aligned} \tag{B.14}$$

where D_1 is an upper bound on $\frac{1}{2}\sum_i (E[S_i^2(k,\mathbf{Q},\mathbf{u}(k))] + E[\tilde{m}_i^2(k)])$ and defined as

$$D_1 \triangleq \frac{1}{2} \Big(N(N-1)L^2 + NL + \sum_i \lceil m_i \rceil^2 (m_i - \lfloor m_i \rfloor) + \lfloor m_i \rfloor^2 (1 - m_i + \lfloor m_i \rfloor) \Big).$$

Note that inequality (B.14) has the same form as inequality (B.9) in the proof of Lemma 3, except that the offline optimum $\mathbf{p}^*$ is replaced by some $\hat{\mathbf{p}} \in \mathcal{F}$. Letting

$$D_2 \triangleq \min_i \{N \sum_q \nu_q \sum_{M \in \mathcal{M}_q} \hat{p}_{qM} \sum_s M_{is} - m_i\},$$

it is always possible to pick a $\hat{\mathbf{p}} \in \mathcal{F}$ such that $D_2 > 0$ (unless $\mathcal{F}$ is a degenerated set which has at most one element). We further bound the above inequality as

$$E[V(\mathbf{Q}(k+1))|\mathbf{Q}(k) = \mathbf{Q}] - V(\mathbf{Q}) \leq \frac{D_3}{\epsilon} + D_1 - D_2 \sum_i Q_i.$$

Here, $D_3 \triangleq N \cdot \max_{\mathbf{p} \in \mathcal{F}_0} \bar{J}(\mathbf{p})$ where $\mathcal{F}_0$ is defined in (4.19). This concludes our proof.

REFERENCES

[1] J. Kurose, K. Ross, and B. Anand, *Computer Networking: A Top-Down Approach*. Pearson/Addison Wesley, 2008.

[2] L. Georgiadis, M. J. Neely, and L. Tassiulas, "Resource allocation and cross-layer control in wireless networks," *Foundations and Trends in Networking*, vol. 1, no. 1, pp. 1–144, 2006.

[3] M. Chiang, S. H. Low, A. R. Calderbank, and J. C. Doyle, "Layering as optimization decomposition: A mathematical theory of network architectures," *Proceedings of the IEEE*, vol. 95, no. 1, pp. 255–312, 2007.

[4] X. Lin, N. B. Shroff, and R. Srikant, "A tutorial on cross-layer optimization in wireless networks," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 8, pp. 1452–1463, Aug. 2006.

[5] S. Shakkottai and R. Srikant, "Network optimization and control," *Foundations and Trends in Networking*, vol. 2, no. 3, pp. 271–379, 2007.

[6] V. Valancius, N. Laoutaris, L. Massoulie, C. Diot, and P. Rodriguez, "Greening the internet with nano data centers," in *Proc. of the 5th international conference on Emerging networking experiments and technologies (CoNEXT)*, 2009, pp. 37–48.

[7] G. Iyengar and A. Kumar, "Characterizing optimal adword auctions," Technical Report, Nov. 2006, http://arxiv.org/abs/cs.GT/0611063.

[8] J. Feldman and S. Muthukrishnan, "Algorithmic methods for sponsored search auctions," *SIGMETRICS Tutorial. Chapter in Performance Modeling and Engineering*, 2008.

[9] I. Menache, A. Ozdaglar, R. Srikant, and D. Acemoglu, "Dynamic online-advertising auctions as stochastic scheduling," in *Proc. of the Workshop on the Economics of Networks, Systems and Computation (NetEcon)*, Stanford, CA, USA, Jul. 2009.

[10] F. Kelly, "Loss networks," *The Annals of Applied Probability*, vol. 1, no. 3, pp. 319–378, 1991.

[11] K. Suh, C. Diot, J. Kurose, L. Massoulie, C. Neumann, D. Towsley, and M. Varvello, "Push-to-peer video-on-demand system: Design and evaluation," *IEEE Journal on Selected Areas in Communications*, vol. 25, no. 9, pp. 1706–1716, 2007.

[12] S. Tewari and L. Kleinrock, "On fairness, optimal download performance and proportional replication in peer-to-peer networks," in *Proc. of IFIP Networking*, 2005.

[13] S. Tewari and L. Kleinrock, "Proportional replication in peer-to-peer networks," in *Proc. of IEEE INFOCOM*, 2006.

[14] J. Wu and B. Li, "Keep cache replacement simple in peer-assisted VoD systems," in *Proc. of IEEE INFOCOM Mini-Conference*, 2009.

[15] J. Kangasharju, K. W. Ross, and D. A. Turner, "Optimizing file availability in peer-to-peer content distribution," in *Proc. of IEEE INFOCOM*, 2007.

[16] Y. Boufkhad, F. Mathieu, F. de Montgolfier, D. Perino, and L. Viennot, "Achievable catalog size in peer-to-peer video-on-demand systems," in *Proc. of the Seventh International Workshop on Peer-to-Peer Systems (IPTPS)*, 2008.

[17] J. M. Almeida, D. L. Eager, M. K. Vernon, and S. J. Wright, "Minimizing delivery cost in scalable streaming content distribution systems," *IEEE Transactions on Multimedia*, vol. 6, no. 2, pp. 356–365, Apr. 2004.

[18] X. Zhou and C.-Z. Xu, "Efficient algorithms of video replication and placement on a cluster of streaming servers," *Journal of Network and Computer Applications*, vol. 30, no. 2, pp. 515–540, Apr. 2007.

[19] N. Laoutaris, V. Zissimopoulos, and I. Stavrakakis, "On the optimization of storage capacity allocation for content distribution," *Computer Networks*, vol. 47, pp. 409–428, 2003.

[20] B. Bollobás, *Modern Graph Theory*. New York: Springer, 1998.

[21] L. Breslau, P. Cao, L. Fan, G. Philips, and S. Shenker, "Web caching and zipf-like distributions: Evidence and implications," in *Proc. of IEEE INFOCOM*, Mar. 1999.

[22] M. Draief and L. Massoulie, "Epidemics and rumours in complex networks," in *the London Mathematical Society Lecture Note Series*. Cambridge University Press, 2010.

[23] M. Mitzenmacher and E. Upfal, *Probability and computing: randomized algorithms and probabilistic analysis*. Cambridge University Press, 2005.

[24] A. Klenke and L. Mattner, "Stochastic ordering of classical discrete distributions," *Advances in Applied probability*, vol. 42, no. 2, pp. 392–410, 2010.

[25] J. Feldman, M. Henzinger, N. Korula, V. S. Mirrokni, and C. Stein, "Online stochastic packing applied to display ad allocation," *Algorithms–ESA, Lecture Notes in Computer Science*, vol. 6346, pp. 182–194, 2010.

[26] J. Feldman, N. Korula, V. Mirrokni, S. Muthukrishnan, and M. Pál, "Online ad assignment with free disposal," *Internet and Network Economics, Lecture Notes in Computer Science (LNCS)*, vol. 5929, pp. 374–385, Dec. 2009.

[27] A. Mehta, A. Saberi, U. V. Vazirani, and V. Vazirani, "Adwords and generalized online matching," in *Proc. of the 46th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 2005, pp. 264–273.

[28] R. Karp, U. Vazirani, and V. Vazirani, "An optimal algorithm for on-line bipartite matching," in *Proc. of the 22nd Annual ACM Symposium on Theory of Computing (STOC)*, 1990, pp. 352–358.

[29] N. Buchbinder, K. Jain, and J. S. Naor, "Online primal-dual algorithms for maximizing ad-auctions revenue," *Algorithms–ESA, Lecture Notes in Computer Science (LNCS)*, vol. 4698, pp. 253–264, 2007.

[30] M. Mahdian, H. Nazerzadeh, and A. Saberi, "Allocating online advertisement space with unreliable estimates," in *Proc. of the 8th ACM conference on Electronic Commerce (EC)*, 2007, pp. 288–294.

[31] N. R. Devenur and T. P. Hayes, "The adwords problem: online keyword matching with budgeted bidders under random permutations," in *Proc. of the 10th ACM Conference on Electronic Commerce (EC)*, Stanford, CA, USA, 2009, pp. 71–78.

[32] B. Kalyanasundaram and K. Pruhs, "An optimal deterministic algorithm for online b-matching," *Theoretical Computer Science*, vol. 233, no. 1–2, pp. 319–325, 2000.

[33] A. Mehta, A. Saberi, U. V. Vazirani, and V. Vazirani, "Adwords and generalized online matching," *Journal of the ACM*, vol. 54, no. 5, Oct. 2007, article No. 22.

[34] S. Agrawal, Z. Wang, and Y. Ye, "A dynamic near-optimal algorithm for online linear programming," *Submitted to Mathematics of Operations Research*, 2010.

[35] J. J. Jaramillo and R. Srikant, "Optimal scheduling for fair resource allocation in ad hoc networks with elastic and inelastic traffic," in *Proc. of IEEE INFOCOM*, San Diego, CA, USA, Mar. 2010.

[36] V. Borkar, *Stochastic Approximation: A Dynamical Systems Viewpoint*. Cambridge University Press, 2008.

[37] L. Huang, S. Moeller, M. Neely, and B. Krishnamachari, "LIFO-Backpressure achieves near optimal utility-delay tradeoff," in *Proc. of 9th Intl. Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt)*, 2011.

[38] S. Asmussen, *Applied Probability and Queues*. Springer-Verlag, 2003.

[39] S. Meyn and R. Tweedie, *Markov Chains and Stochastic Stability*. Springer-Verlag, London, 1993.

[40] F. Kelly, A. Maulloo, and D. Tan, "Rate control for communication networks: shadow prices, proportional fairness and stability," *The Journal of the Operational Research Society*, vol. 49, no. 3, pp. 237–252, 1998.

[41] S. Shakkottai, "Effective capacity and QoS for wireless scheduling," *IEEE Transactions on Automatic Control*, vol. 53, no. 3, pp. 749–761, Apr. 2008.

[42] L. Ying, R. Srikant, A. Eryilmaz, and G. Dullerud, "A large deviations analysis of scheduling in wireless networks," *IEEE Transactions on Information Theory*, vol. 52, no. 11, pp. 5088–5098, Nov. 2006.

[43] A. Eryilmaz, R. Srikant, and J. Perkins, "Stable scheduling policies for fading wireless channels," *IEEE/ACM Transactions on Networking*, vol. 13, no. 2, pp. 411–424, Apr. 2005.

[44] R. A. Berry and R. G. Gallager, "Communication over fading channels with delay constraints," *IEEE Transactions on Information Theory*, vol. 48, no. 5, pp. 1135–1149, May 2002.

[45] M. J. Neely, "Intelligent packet dropping for optimal energy-delay tradeoffs in wireless downlinks," *IEEE Transactions on Automatic Control*, vol. 54, no. 3, pp. 565–579, Mar. 2009.

[46] M. J. Neely, "Optimal energy and delay tradeoffs for multiuser wireless downlinks," *IEEE Transactions on Information Theory*, vol. 53, no. 9, pp. 3095–3113, 2007.

[47] D. Shah and D. Wischik, "Lower bound and optimality in switched networks," in *Proc. of 46th Annual Allerton Conference on Communication, Control, and Computing*, 2008, pp. 1262–1269.