SPRINGER BRIEFS IN OPERATIONS MANAGEMENT

Subodha Kumar

Optimization Issues in Web and Mobile Advertising

Past and Future Trends

Springer

SpringerBriefs in Operations Management

Series Editor

Suresh P. Sethi
The University of Texas at Dallas, TX, USA

More information about this series at http://www.springer.com/series/13082

Subodha Kumar

Optimization Issues in Web and Mobile Advertising

Past and Future Trends

Springer

Subodha Kumar
Department of Information
and Operations Management
Texas A&M University
College Station, TX, USA

ISSN 2365-8320 ISSN 2365-8339 (electronic)
SpringerBriefs in Operations Management
ISBN 978-3-319-18644-3 ISBN 978-3-319-18645-0 (eBook)
DOI 10.1007/978-3-319-18645-0

Library of Congress Control Number: 2015951986

Springer Cham Heidelberg New York Dordrecht London

Printed on acid-free paper

Springer International Publishing AG Switzerland is part of Springer Science+Business Media (www.springer.com)

This book is dedicated to:
my children Aarna and Medha,
my wife Susmita, and
my parents

Preface

For many websites, revenue generated from advertisements (or ads for short) is critical for survival. This in turn raises the question: How should ads at a website be scheduled to maximize revenue? This monograph aims to provide an overview of such optimization issues in web and mobile advertising. This monograph begins with discussing the evolution of web advertising over time. This is followed by the discussion of prominent pricing models. The initial models mainly considered the problem of scheduling ads competing to be placed on a web page. Here, the ads were specified by geometry and display frequency, and both of these factors were considered in developing a solution to the ad scheduling problem. These models were similar in nature to the problem of scheduling ads on newspaper or television, but the pricing structure was different. As the web advertising evolved, the initial models were augmented by considering how the schedule of ads is changed based on individual user click behavior. Thus, these models considered personalization in web advertising. The landscape of web advertising changed further with the growth of ad-firms. The ad-firms usually match an advertiser's ads with a publisher's website and with a web user who visits the website. The recent studies analyze the problem from the perspective of ad-firms. More recently, there has been tremendous growth in mobile advertising. In this domain, the most popular vehicle for displaying ads is a mobile application (or app for short). In mobile advertising, an important problem for ad-firms is demand-supply optimization problem. The last chapter discusses some of the key future trends in web and mobile advertising, and associated challenges. This book is written for (a) academicians and professionals involved in web/mobile advertising with an interest in optimization issues, and (b) academicians and professionals involved in optimization problems with an interest in web/mobile advertising.

College Station, TX, USA — Subodha Kumar

Contents

Chapter 1
Evolution of Web Advertising

Abstract The number of Internet users is increasing dramatically. With the sharp increase in online users, the market for web advertising is also growing. In the last decade, the web advertising revenues in the United States, as well as worldwide, have increased significantly. With the growth of web advertising, several web sites have now begun to depend heavily on the revenue generated by the advertisements displayed on the site. For such sites, revenue generated from advertisements is critical for survival. This chapter discusses how the web advertising evolved over time.

1.1 History of Web Advertising

We begin with discussing the growth of web advertising in its early phases.

1.1.1 Early Phases of Web Advertising

Advertising on the World Wide Web became mainstream in early 1994 when HotWired, a web magazine (part of Wired magazine), sold a banner ad to AT&T and displayed it on their web page [3, 17]. The exploding supply of web pages in 1994 led to the birth of several search engines [5]. These search engines also sold advertising to make money. During the early period, fairly traditional methods of advertising were mimicked on the web. These included web versions of business directories similar to the yellow pages such as yellowpages.com; web versions of newspaper classified ads such as Craigslist; and web versions of direct mail and telephone marketing such as CheetahMail [5].

In 1995, fewer than one third of U.S. households had computers, and fewer than half of those machines had decent Internet connections [21]. Hence, there was lot of skepticism about the web advertising. This skepticism grew further in late 1995.

S. Kumar, *Optimization Issues in Web and Mobile Advertising*, SpringerBriefs in Operations Management, DOI 10.1007/978-3-319-18645-0_1

However, it did not stop advertisers from using web as a new media of advertising. While web advertising was going through a defining phase in 1995, the two most debated issues were: (a) tracking of users' visits to the websites, and (b) pricing for advertising located on content sites [21]. In this period, many small advertisers took a "wait and see" approach, but big names (such as Fox, Chrysler, Sports Illustrated, and ESPN) started using web advertising. The rates for web advertisement (hereafter also referred to as "ad") in late 1995 varied from a few thousand dollars to $1 million. For example, HotWired charged $30,000 for an 8 week sponsorship, and ESPN charged $1 million for a 1 year sponsorship of ESPNET [21].

Advertisers in 1995 mainly focused on attracting clickers via the web itself, rather than using traditional media (such as magazines, newspaper, and TV). However, according to a 1996 survey by Market Facts, half of 500 respondents said that they visited a website because they saw the web address in a magazine or newspaper ad [21]. Hence, in 1996, companies started launching ads for websites on traditional media. *Piggybacking* (also referred to as *affiliate marketing*) was another innovative approach that gained popularity in 1996. In this approach, websites created business-to-business links with partners in order to increase traffic. Sports sites, such as ESPN, were among the first to use this approach by providing links for their partners on their sites, with the expectation that the link would be reciprocated [21].

In the first half of 1996, the web advertising revenue reached an all-time high at approximately $81.8 million [9]. Also, the studies showed that users were going online primarily to gather news and information [21]. Hence, many advertisers switched from entertainment-oriented ads of 1995 to information-oriented ads. Such ads were referred to as *advertorials* that combined ads with editorial content. CDNow was an early innovator in web advertising. It started use of editorial content as a means to promote interest in all sorts of music. It also created the first affiliate marketing program on the Internet, which was later perfected by Amazon.com.

1.1.2 Targeting Phase of Web Advertising

As Rich LeFurgy (Chairman and Senior Vice President of Internet Advertising Bureau) stated: "1997 was definitely a breakthrough year for Internet advertising" [10]. This was also reflected by the fact that the web advertising revenue in 1997 reached $906.5 million (compared to $267 million in 1996) [9, 10]. In this period, search engines found new revenue streams by rounding up cobranding sponsorships. For example, in the second quarter of 1997, Yahoo reported $13.5 million in revenue (up 42 % from the first quarter), Excite's reported $9.5 million (up 26 % from the first quarter), and Infoseek reported $7.7 million (up 25 % from the first quarter) [21].

In this phase, the web advertisers matured and started focusing on personalization of web advertising. Instead of links, advertisers became more concerned with proximity and immediacy [21]. This is mainly attributed to the technological

advances in 1997. Until 1997, it was difficult to know who was using the web. However, in 1997 (and onwards), several companies started providing services related to targeting customers on web. For example, a company called *PC Meter* provided two types of services: (a) comparative research on web sites, and (b) tracking consumer and business usage trends. Similarly, another company, called *RelevantKnowledge*, started providing data about business usage by ranking top websites in a variety of categories. Until 1997, one of the major drawbacks of web advertising was considered to be geographic limitations. The advertisers and website started overcoming this limitation in 1997 by asking users to register their zip codes. Several advertisers and search engines reported that the response improved by geographic targeting. One of the reasons why the web advertising landscape changed in this period is attributed to the fact that the efforts of traditional media (such as Newspaper, radio, and TV stations) to go online became intense. For example, *The Wall Street Journal* launched a career site to attract online advertisers.

The innovations in web advertising continued in 1998. In this period, another form of ads, called *Interstitial Ads*, started becoming more common. These ads appear while users wait for screens to download. Another major event that occurred in 1998 was that Google received its first funding with a contribution of $100,000 from Andy Bechtolsheim, a co-founder of Sun Microsystems [19]. By 1998, Google had indexed 60 million pages with their search algorithm and began ranking them. This was an important step in the domain of web advertising.

1.1.3 Growth of Web Advertising in the Twenty-First Century

Although Yahoo started search ads around 1996, a big leap in web advertising happened in 2000 when Google developed AdWords. Google's AdWords allowed advertisers to display their ads in the Google content network. Another important development in the twenty-first century was the growth of firms as *DoubleClick*. DoubleClick was one of the earliest known Application Service Provider (ASP) for Internet "ad-serving." Such companies offered technology products and services for advertising agencies and media companies to allow clients to traffic, target, deliver, and report on their interactive advertising campaigns. With the growth of these services, the advertisers were able to improve targeting and personalization for their ads, which resulted in increased expenditure on web advertising.

The growth of social media (such as Facebook, Twitter, and Linkedin) in the twenty-first century provided a big boost to web advertising. For example, around $5.1 billion were spent on social advertising in 2013 [20]. According to the studies by BI Intelligence and BIA/Kelsey, the U.S. social-media ad revenue will reach nearly $14–15 billion in 2018 [2, 7]. In the third quarter of 2014, Facebook brought in $2.96 billion in advertising revenue, and Twitter generated $320 million [1]. Mobile advertising is also closely related to web advertising, but its reach is far greater. Hence, it is expected that advertisers and media industry will increasingly take account of a bigger and fast-growing mobile market, especially in the social

media. It is evident from the fact that, in the third quarter of 2014, 66 % of Facebook advertising revenue came from mobile ads, and Twitter generated 85 % of its total revenue from mobile advertising [1]. According to BIA/Kelsey, the year 2014 represented the greatest year-over-year jump in social media ad revenues, growing to $8.4 billion, largely due to increases in mobile advertising [2].

Overall, the revenue from web advertising grew tremendously in the twenty-first century. In 2013, it increased by 17 % over 2012 to reach $42.78 billion [11]. This trend is expected to continue: It is estimated that the web advertising revenue in the United States will reach $50 billion by 2015 [4]. Another estimate shows that the U.S. web advertising will reach $77 billion in 2016, and will comprise 35 % of all advertising spending, overtaking television advertising [8]. Finally, Internet advertising is not purely a U.S. phenomenon. In the United Kingdom, online advertising revenues in 2013 increased by 15.2 % over 2012 to reach almost $6.3 billion [13]. In 2013, Internet passed newspapers to become the world's second-largest ad medium, behind TV—Internet now captures one in five ad dollars [16]. According to a report by Digital TV Research, global Internet advertising spending will reach $143 billion in 2017 [18].

1.2 Types of Web Ads

Web advertising is broadly classified into following categories [5]: (a) Display Advertising, (b) Search Engine Marketing, (c) Online Classified Advertising, (d) Affiliate Marketing (also called "Lead Generation"), (e) E-Mail Advertising, and (f) Interstitial Advertising. As discussed earlier, two new and emerging delivery methods for web advertising are: (a) Social Media Advertising, and (b) Mobile Advertising.

Display advertising is the earliest form of web advertising that began in the early 1990s. Display ads on websites look like those in newspapers and magazines. These can be further classified into following categories [5]: (a) Banners, (b) Sponsorships, (c) Rich Media, (d) Slotting Fees, and (e) Digital Video. A banner ad is a small, typically rectangular, graphic ad displayed within a web page. In the early 1990s, a commercial online service, *Prodigy*, displayed banners at the bottom of the screen to promote Sears products [6]. The first clickable web ad was sold in 1993 by *Global Network Navigator* to a Silicon Valley law firm [3]. As discussed earlier, the web banner advertising became mainstream in 1994 when HotWired sold banner ads to AT&T and other companies [3, 17]. Next, sponsorships represent custom content created for an advertiser that may or may not include ad elements (for example, reskinning a section of a website with the advertiser's branding) [5]. Rich media ads include some interactivity using different features, such as animation and sound. Slotting fees ads are specific types of display ads in which a fee is charged for premium ad placement and/or exclusivity. Finally, the digital video ads include commercials that appear in live, archived, and/or downloadable streaming content [3]. This format of display ads has gained popularity in the last few years.

Search engine marketing (SEM) is designed to increase the visibility of a website in search engine results. Search engines provide sponsored results and organic (non-sponsored) results based on the query of a web searcher [14]. Search ad refers to paying Internet companies to present an ad linked to a specific search word or phrase [5]. It includes: (a) paid listings in sponsored results (text links appear at the top or side of search results for specific keywords), and (b) paid inclusion in organic results (guarantees that a marketer's URL is indexed by a search engine). Search engines usually employ visual cues to differentiate sponsored results from organic results. Since 2005, search ads have been the most prominent format of web advertising (in terms of revenue). Before that, display ads were the most prominent format. In 2013, search revenues accounted for 43 % of the total web advertising revenues [11].

When the web advertising started, it was believed that if media and advertising standards were not put in place, the online advertising medium could never mature and capture a brand's advertising spend [15]. Unlike traditional media (which had established formats and specifications across the multiple channels), every web page could be a separate media company with different specifications and measurements in place. Hence, in 1996, the Interactive Advertising Bureau (IAB) was founded to empower the media and marketing industries to thrive in the digital economy. One of the core objectives of IAB is: "Coalesce around market-making measurement guidelines and creative standards" [12].

1.3 Differences Between Web Advertising and Offline Advertising

Although the goal of both web advertising and offline advertising is to attract customers, these two have some fundamental differences. First of all, the structure of online media allows advertisers and websites to gather much more information about the users. For example, in the online channel, using the IP address of the computer and the cookies, the advertiser can attempt to understand the intent as well as the precise location of the individual user. These cookies can also provide important information regarding what other sites the user has visited. However, in an offline media, such as TV, radio, and newspaper, it is usually not possible to collect such information about each individual user. Using these information, the online advertisers can present personalized and targeted ads to each user. Therefore, the effectiveness of well-executed web ad is usually much better than that of an offline ad.

Second, it is much easier to evaluate the effectiveness of a web ad compared to that of an offline ad. In the online media, the advertisers precisely know when the ad is displayed to each user. However, in an offline media, such as newspaper, it is not possible to know when exactly the ad is displayed to the user. Further, if an online user clicks on a web ad, it is registered directly in the system. However, such measurement is quite difficult in an offline media. The information about the time also helps the online advertiser in customizing the ad. The reach of an online ad is also

much higher than that of an offline ad, because an online ad can be placed anywhere globally. Another advantage of online advertising is that it can be customized based on the actions of the users (even in the real-time). However, it is difficult to change an offline ad based on the actions of the users.

Because of the differences discuss above, the optimization issue in web advertising are different from those in offline advertising. The purpose of this book is to discuss the optimization models for web advertising that exploit the unique aspects of online channel. In this book, we also discuss the new optimization issues arising because of the emerging trends, such as mobile advertising and ad exchanges. Before discussing the optimization models, in Chap. 2, we present different pricing models used in web advertising.

References

1. Bergen M (2014) Twitter beats revenue expectations again but user engagement slows. AdvertisingAge, October 27
2. BIA/Kelsey (2014) U.S. social media advertising revenues to reach $15B by 2018. http://www.biakelsey.com/Company/Press-Releases/140515-U.S.-Social-Media-Advertising-Revenues-to-Reach-$15B-by-2018.asp. Retrieved9Feb2015
3. Briggs R, Hollis N (1997) Advertising on the web: is the response before clickthrough? J Advert Res 37(2):33–45
4. eMarketer (2011) Online advertising market poised to grow 20% in 2011. eMarketer Newsroom, June 8
5. Evans DS (2009) The online advertising industry: economics, evolution, and privacy. J Econ Perspect 23(3):37–60
6. Gibson M (2012) History of online display advertising. Vantage Local, July 12
7. Hoelzel M (2014) There's a simple rule that's helping marketers advertise more effectively on social networks. Business Insider, October 16
8. Hof R (2011) Online ad spend to overtake TV by 2016. Forbes, August 26
9. Interactive Advertising Bureau (1997) Internet advertising bureau announces 1996 advertising revenue reporting program results. http://www.iab.net/about_the_iab/recent_press_releases/press_release_archive/press_release/4226.Retrieved9Feb2015
10. Interactive Advertising Bureau (1998) Internet advertising sees breakthrough year in 1997. http://www.iab.net/about_the_iab/recent_press_releases/press_release_archive/press_release/4230.Retrieved9Feb2015
11. Interactive Advertising Bureau (2014) IAB Internet advertising revenue report: an industry survey conducted by PwC and sponsored by the Interactive Advertising Bureau (IAB), 2013 full year results. http://www.iab.net/media/file/IAB_Internet_Advertising_Revenue_Report_FY_2013.pdf.Retrieved9Feb2015
12. Interactive Advertising Bureau (2015) About the IAB. http://www.iab.net/about_the_iab.Retrieved9Feb2015
13. Interactive Advertising Bureau UK (2014) 2013 Full year digital adspend results. http://www.iabuk.net/research/library/2013-full-year-digital-adspend-results.Retrieved9Feb2015
14. Jansen BJ, Mullen T (2008) Sponsored search: an overview of the concept, history, and technology. Int J Electron Bus 6(2):114–131.

15. Joel M (2013) We need a better definition of “Native Advertising.” Harvard Business Review, February 13
16. Johnson B (2013) 10 things you should know about the global ad market. AdvertisingAge, December 8
17. Kaye BK, Medoff N (2001) Just a click away: advertising on the Internet. Allyn and Bacon, Massachusetts
18. Kemp S (2012) Study: global online ad revenue to hit $143 billion by 2017. The Hollywood Reporter, November 14
19. Kopytoff V, Fost D (2004) For early Googlers, key word is $$$/Founders, backers could reap billions when company goes public. SFGATE, April 29
20. Lieb R (2014) Why it’s not the end of social media marketing. iMedia Connection, July 1
21. Thorson E, Wells WD, Rogers S (1999) Web advertising’s birth and early childhood as viewed in the pages of advertising age. In: Schumann DW, Thorson E (eds) Advertising and the world wide web. Lawrence Erlbaum Associates Inc Publishers, Hillsdale

Chapter 2
Pricing Models in Web Advertising

Abstract The method to determine the price of web advertising is one of the most striking innovations. In the early phases, most of the web advertisements were sold through traditional exposure-based pricing models, such as the flat fee model or the cost per impression model. However, over time, some other models also emerged in practice where the pricing is based on a visitor taking some specifically defined action in response to an advertisement. In fact, such models are now becoming more popular than the traditional models. This chapter presents different pricing models used in web advertising and analyze them.

2.1 Traditional Exposure-Based Pricing Models

In the early days of web advertising (around late 1994 and early 1995), few websites or advertisers understood the real value of the web as an advertising medium, let alone what price to assign to web advertising. The advertising managers certainly understood the value of a 30 s television spot or a classified ad in the back of the book, due to factors including industry norms, program ratings, years of marketing research, and managerial experience. But, except for a handful of pioneering online programs like CDNOW's BuyWeb, which produced measurable results, there was simply no way to understand the value of web advertising [9]. This presented a dilemma for those managers who wished to advertise online.

Given little knowledge about the effectiveness of web advertising, the flat fee pricing model were the earliest web advertising pricing model [10]. Flat fee pricing charges the advertiser for their ads on a website in a certain period (e.g., per month). Flat fee can be without or with traffic guarantees [2]. Naturally, it would be advantageous for the advertiser to request guarantees of traffic level. At a minimum, accurate information on site traffic must be made available to the advertiser, so that the advertiser may evaluate alternative web media vehicles. Hence, the advertisers started requesting for the traffic information. Assuming accurate traffic information,

S. Kumar, *Optimization Issues in Web and Mobile Advertising*, SpringerBriefs in Operations Management, DOI 10.1007/978-3-319-18645-0_2

flat fee prices may be readily converted into a CPM (cost-per-thousand-exposures) model. In the CPM model, an advertiser pays an amount based on the number of impressions (exposures) of an ad [12]. The number of impressions during a visit is the number of times that the ad is displayed to the user during the visit [16]. As with conventional broadcast and print advertising, this approach measures only the amount of advertising delivered, broken down, at best, by demographic or psychographic segments.

In the early phases of web advertising, CPMs were inflated and flat fee deals were even worse. There were no guarantees and there was no rationality to the market. Many of the early online selling efforts were conducted by magazine salespeople with little experience regarding how new media actually worked. Thinking of banner ads in the same way as they thought of a 30 s television spot or a print ad in a magazine, they sought to base their prices on the number of people who would see an ad—what in the trade was called "exposure-based cost-per-thousand pricing" [9]. Hence, they quite literally invented CPM rates in the low $70s or charged flat fees from $5000 to $10,000 per month to advertise on some of the earliest commercial websites. It did not take long for the advertisers to determine that those current CPMs were inflated and an obvious bad buy. At the flat rate of $10,000, the ad would never even generate enough revenues to cover its costs. Consider the following. Suppose the price of a single banner ad exposure (i.e., one page view) was seven cents, because the web provider demanded a $70 CPM. If 1 % of the visitors that saw the banner actually clicked, a single click would cost $7. But, since only a small percentage of those visitors were converted into paying customers, the customer acquisition costs were actually sky high. Assuming a 1 % conversion rate, the cost to acquire that new customer was $700 [9]. As advertisers started realizing such problems with the high costs of CPM pricing model, the web advertising industry increasingly moved towards performance-based pricing models. However, until 2005, the CPM pricing model was the most prominent model in web advertising. Even in 2013, approximately 33 % of the total web advertising revenues were priced on the basis of CPM [11].

2.2 Performance-Based Pricing Models

The performance-based pricing has been the most prevalent pricing model since 2006. In 2013, approximately 65 % of the web advertising revenues were priced on a performance basis [11]. The earliest performance-based pricing model in web advertising is the *cost-per-click* pricing model.

2.2.1 Cost-Per-Click Pricing Model

Paying by the number of viewers (i.e., the CPM model) remained the norm until Procter & Gamble negotiated a deal with Yahoo! in 1996 that compensated the web portal for ads based on the "cost-per-click," commonly known as "CPC" [4]. Yahoo! was paid only when a user clicked on the ad; this was the web-version of paying for direct response commonly used by advertisers for things such as mail and telephone solicitations. By 2002, the CPC model had been adopted by both Google and Yahoo!, and it became the most widely used pricing model in paid search advertising [3].

A relatively small proportion of those exposed to a banner ad actually click on the banner. In 2006, DoubleClick reported that 4 % of web site visitors who were exposed to a banner ad clicked on the ad the first time they saw it [10]. The top 25 % performing ads in the DoubleClick Network had an average click-through rate (defined as the number of times that the ad is clicked upon divided by the number of times that it is exposed) of 8 %, with some click rates as high as 12–15 %. Click-through rates decline after the first exposure, falling to 2 % for the second and third exposures, and to 1 % or less at four exposures. Therefore, the payment based upon the number of clicks guarantees that the visitor is not only exposed to the banner ad, but also actively decides to click on the banner (to become exposed to the target communication). Therefore, the payment based on the number of clicks may be viewed as payment for target communication exposures. In other words, the CPC model was an attempt to develop a more accountable way of charging for web advertising [12].

In spite of its benefits, CPC model also had its controversies. Some website providers continued to feel that this pricing strategy was unfair, arguing that the click-through was at least partially a function of the level of creativity of the ad and the level of interest generated in the viewer by it, which were not under the control of website providers [16]. On the other hand, as discussed above, applying only traditional exposure-based models to the web does not take into account its unique, interactive nature. Additionally, the Internet is the first commercial medium in which it is actually possible to measure consumer response, not just to assume it. Although the CPC model might not represent the optimal approach to measuring the value of interactivity, it offered a departure point from which to proceed.

2.2.2 Cost-Per-Action Pricing Model

Although the payment based on CPC model guarantees that there was visitor exposure to target communications, it does not guarantee that the visitor liked the communication or even spent any substantial time viewing it. Hence, it is proposed that an additional measure of the value of an ad should be based upon the degree to which the visitor interacts with the target communication [10]. An interactivity metric might be based upon the duration of time spent viewing the communication,

the depth or number of pages of the target communication accessed, the number of repeat visits to the target communication, or some other elements. Such a practice was announced for the first time in 1996 when a member of the Internet mailing list posted to the list that Modem Media, the interactive advertising agency, had developed a pricing model in which its clients would pay, not for exposures or clicks, but only for activity at the client's website.

This development raised anew the controversy surrounding the best web media pricing models, with website providers arguing that the problem with performance-based measures, such as the number of clicks or some action, is that the website provider cannot be held responsible for the activity related to an ad. An analogy is drawn to print, with the web publisher arguing that the print medium charges for ads, regardless of whether they lead to sales. Not surprisingly, advertisers and their agencies continue to argue that, since the web medium allows for accountability, it is possible and desirable to develop models that measure consumer behavior [16]. Therefore, some leading practitioners are now moving toward measures as the "cost-per-action" (CPA) pricing model proposed by Snap.com, where an action refers to a realized purchase, download, registration, subscription, or lead [5]. A number of merchants are following this trend, including Google, ZiXXo (pay-per-print), Ingenio (pay-per-call), and others [3]. In 2006, Google attracted media attention when it started to test a CPA model [7, 8]. Google regards CPA as the "Holy Grail" of targeted advertising [6], and many online advertising companies have adopted it, including eBay, ValueClick, and Snap.com.

2.2.3 Outcome-Based Pricing Model

The advertisers are eventually interested in outcomes, and the ultimate outcome is purchase. As Stephen Klein, former I/PRO manager, stated: "One hundred thousand people going to a site is worth something, but a site that only five people visit can be worth more if they are the right five people" [14]. The metrics discussed above relates to early stages of the purchase process: Banner ads affect the consumer's awareness, and the action affects the consumer's comprehension and understanding. Beyond these initial stages are the marketing objectives of attitude change, purchase intention, and, ultimately, purchase. An outcome-based pricing model begins by specifying precisely the advertiser's goal for the target communication. Examples of typical outcomes include influencing attitudes, motivating the consumer to provide personal information, or leading the consumer to purchase. Whatever the marketing objective, the web provides a vehicle for integrated marketing campaigns, which allows the marketer to track and to measure the effectiveness of the ad. An example is per-inquiry (PI) ads where the royalty is paid only on actual product sales, and no other payment is required [10].

2.3 Hybrid Pricing Models

The CPM model and the performance-based model represent two extreme pricing strategies. Pure CPM pricing favors the website provider because there is no risk: The website provider is paid whether or not the ad is clicked upon. On the other hand, the performance-based pricing model favors the advertiser because the website provider carries all the risk: If the performance criterion of the advertiser is not met, no revenue accrues to the website provider [12]. Several recent studies analyze and compare different pricing models (e.g., see [1, 5, 13, 15]). These studies recommend a hybrid pricing model that shares the risk between the website provider and the advertiser. In [12], a hybrid pricing model is considered for optimal scheduling and placement of web ads. In 2013, approximately 2 % of the web advertising revenues were priced on a hybrid basis [11].

References

1. Asdemir K, Kumar N, Jacob VS (2012) Pricing models for online advertising: CPM vs. CPC. Inf Syst Res 23(3):804–822
2. Dickinger A, Zorn S (2008) Compensation models for interactive advertising. J Universal Comput Sci 14(4):557–565
3. Economist (2006) The ultimate marketing machine. Econ Spec Represent 380(8485):61–64
4. Evans DS (2009) The online advertising industry: economics, evolution, and privacy. J Econ Perspect 23(3):37–60
5. Feng J, Xie J (2012) Performance-based advertising: advertising as signals of product quality. Inf Syst Res 23(3):1030–1041
6. Gardiner B (2007) Google seeks “Holy Grail” of cost per action ad pricing. Wired, August 23
7. Gonsalves A (2006) Google confirms testing new ad-pricing model. InformationWeek, June 22
8. Helft M (2007) Google tests an ad idea: pay only for results. The New York Times, March 21
9. Hoffaman DL, Novak TP (2010) How to acquire customers on the web. Harvard Business Review, May
10. Hoffaman DL, Novak TP (2010) Advertising and pricing models for the web. In: Hurley D, Kahin B, Varian H (eds) Internet publishing and beyond: the economics of digital information and intellectual property. MIT Press, Cambridge
11. Interactive Advertising Bureau (2014) IAB Internet advertising revenue report: an industry survey conducted by PwC and sponsored by the Interactive Advertising Bureau (IAB), 2013 Full Year Results. http://www.iab.net/media/file/IAB_Internet_Advertising_Revenue_Report_FY_2013.pdf.Retrieved9Feb2015
12. Kumar S, Dawande M, Mookerjee VS (2007) Optimal scheduling and placement of Internet banner advertisements. IEEE Trans Knowl Data Eng 19(11):1571–1584
13. Mangàni A (1996) Online advertising: pay-per-view versus pay-per-click. J Revenue Pricing Manage 2(4):295–302
14. Murphy IP (1996) On-line ads effective? Who knows for sure? Market News 30(20):1–38
15. Najafi-Asadolahi S, Fridgeirsdottir K (2014) Cost-per-click pricing for display advertising. Manuf Serv Oper Manage 16(4):482–497
16. Novak TP, Hoffman DL (1997) New metrics for new media: toward the development of web measurement standards. World Wide Web J 3(1):213–246

Chapter 3
Scheduling Advertisements on a Web Page

Abstract Many web sites (e.g., Hotmail, Yahoo, Google) provide free services to the users while generating revenues from advertising. Advertising revenue is, therefore, critical for these sites. This in turn raises the question, how should ads at a web site be scheduled in a planning horizon to maximize revenue. Consider a set of ads competing to be placed in a planning horizon that is divided into time intervals called *slots*. An ad is specified by its size and display frequency. The size represents the amount of space the ad occupies in a slot. An ad is said to be scheduled if a pre-specified number of copies of that ad are placed in the slots subject to the restriction that a slot contains at most one copy of an ad. In this chapter, we present two problems. The MINSPACE problem minimizes the maximum fullness among all slots in a feasible schedule where the fullness of a slot is the sum of the sizes of ads assigned to the slot. For the MAXSPACE problem, in addition, we are given a common maximum fullness for all slots. The total size of the ads placed in a slot cannot exceed that common maximum fullness. The objective is to find a feasible schedule of ads such that the total occupied slot space is maximized. We examine the complexity status of both problems and present heuristics with performance guarantees. For the MAXSPACE problem, we also present a hybrid genetic algorithm (GA).

3.1 Introduction

As discussed earlier, a banner ad is a small graphic image that is linked to a target ad. Many different types of banner ads with different sizes are being used in web advertisement. Rectangular ads are the most common type of banner ads. These ads usually appear on the side, top, or bottom of a screen as a distinct, clickable image [19]. Banner ad size is usually measured in pixels: width $\times$ height. Many different accepted sizes, in pixels, are identified by the Standards and Practices Committee of the Interactive Advertising Bureau [14]. Various types of banner

S. Kumar, *Optimization Issues in Web and Mobile Advertising*, SpringerBriefs in Operations Management, DOI 10.1007/978-3-319-18645-0_3

ads can be observed. For example, www.cheaptickets.com displays three side banner ads and www.travelocity.com displays six side banner ads of different sizes [5]. Typically, a set of ads competes for space on a web page in a planning horizon, say an hour. If ads are updated every 40 s on the web page, we have 90 time intervals to schedule ads. In each time interval, a rectangular slot (e.g., side banner) consisting of ads is displayed for viewers. For example, UtopiAd's desktop companion Magellan (www.utopiAd.com) updates ads after every 20 s.

This chapter presents two problems namely, MINSPACE and MAXSPACE, which are theoretical models designed to address the problem of scheduling ads competing to be placed on a web page. A *block* is a rectangular space allocated on a web page, usually on the left, right, top or bottom, which houses one or more banner ads. A given planning horizon is divided into a number of time intervals called *slots*. Ads scheduled in a block for a particular slot are displayed for users who visit the site during that time interval. In scheduling ads, these problems consider two factors: (a) the size of the ad and (b) the number of times (slots) that an advertiser would like the ad to appear on the web site during a planning horizon.

Since the maximum space to be used by ads on a web page is limited, the advertisement provider may not be able to place all the ads that are competing for space in a given planning horizon. The advertisement provider is then faced with the problem of selecting and placing ads in such a way that his/her profit is maximized. This issue is addressed by the MAXSPACE problem. The use for the MINSPACE problem is primarily during the design stage of a web page whose contents may change over time. The advertisement provider wants to place all ads that are requested by customers in a given planning horizon. Here, the space to be allocated for all ads needs to be decided optimally.

3.2 Problem Description

Consider a set of n ads $A = \{A_1, \ldots, A_n\}$. A planning horizon is divided into N equal time intervals (slots). An ad placed in a particular slot will be displayed for the time interval corresponding to that slot. Each ad has two characteristics: *size* and *frequency*. The size of an ad A_i is denoted by s_i and represents the amount of space A_i occupies in a slot. The frequency of an ad A_i is denoted by w_i and is the number of slots which should contain a copy of the ad. Advertisers do not want to display more than one copy of an ad in the same slot and therefore an ad A_i can be displayed at most once in a slot.

Definition. Ad A_i is said to be *scheduled* if a total of w_i copies of A_i appear in the slots with at most one copy in a slot.

Clearly, $w_i \leq N$, $\forall\ i$. We assume that the width of the ads placed in a slot is the same as the width of the slot as this is the ad presentation scheme followed by most web sites. The *fullness* of a slot j is defined as $f_j = \sum_{A_i \in B_j} s_i$, where B_j is a set of ads which have a copy in slot j. The *height* of the schedule, f, is the maximum fullness of the slots, i.e., $f = max_j f_j$.

In the MINSPACE problem, all the ads are to be placed in N slots. The objective is to find a schedule with minimum height, say f^*. For example, consider the problem instance given in Fig. 3.1a. A feasible schedule is shown in Fig. 3.1b. Here, the fullness of slot 2 is $s_1 + s_3 = 9$ and the height of the schedule is $f = 10$. Figure 3.1c shows an optimal schedule with $f^* = 9$. It should be noted that during any time interval only one of these five slots is shown to a user accessing the web page. The choice of which slot to display in a time interval can be made by cycling through a deterministic permutation of the N slots.

In the MAXSPACE problem, a common upper bound, S, on the fullness of each slot is specified. For this problem, a feasible schedule is a placement of a subset $A' \subseteq A$ of ads such that (a) for each $A_i \in A'$, exactly w_i copies are placed in the slots with at most one copy of an ad in each slot and (b) the fullness of any slot must not exceed S, i.e., $\sum_{A_i \in B_j} s_i \leq S, \forall\ j$, where $B_j \subseteq A'$ is a set of ads which have a copy in slot j. Without loss of generality, we can assume that for each $A_i \in A'$, $s_i \leq S$. The objective of the problem is to find a feasible schedule of $A' \subseteq A$ ads such that the total *weight* of the scheduled ads ($\sum_{A_i \in A'} s_i w_i$) is maximized. Here, an underlying assumption is that the profit for the web site owner increases with the size of the scheduled ad (i.e., the total number of pixels required to display the ad). This is a reasonable assumption and is often valid in practice [20]. A feasible schedule for the problem instance given in Fig. 3.1a with $S = 6$ is shown in Fig. 3.1d with $\sum_{A_i \in A'} s_i w_i = 27$, where $A' = \{A_1, A_3, A_6, A_7, A_8\}$. An optimal schedule is shown in Fig. 3.1e with $\sum_{A_i \in A'} s_i w_i = 30$, where $A' = \{A_1, A_3, A_5, A_6, A_7\}$.

3.3 Earlier Results for Special Cases of the Problem

A special case of the MINSPACE problem with $w_i = 1$ for all ads A_i, $i = 1, \ldots, n$, is equivalent to the classical problem of multiprocessor scheduling [2, 17, 22] that can be stated as follows: n independent tasks require processing on any of m identical parallel processors. The objective here is to find a schedule of tasks that minimizes the total completion time of all tasks (or the makespan of the schedule). In this special case, N (number of slots) is equivalent to m parallel processors and the problem reduces to the problem of scheduling n ads (tasks) in N slots (machines). Here, the size s_i of each ad A_i corresponds to the processing time of a task in the context of the multiprocessor scheduling problem. One popular algorithm for the multiprocessor scheduling problem is the LPT (largest processing time first) algorithm. The worst bound of LPT is $\frac{4}{3} - \frac{1}{3m}$ [11].

When $w_i = 1$ for all ads $i = 1, 2, \ldots, n$, the MAXSPACE problem has some similarity to the bin-packing problem [9]. Given a list of real numbers between 0 and 1, the bin-packing problem is the problem of placing all the elements of the list in a minimum number of bins so that no bin contains numbers whose sum exceeds 1.

a

Ad A_i	A_1	A_2	A_3	A_4	A_5	A_6	A_7	A_8
s_i	5	5	4	4	2	1	1	1
w_i	3	2	2	1	2	2	1	1

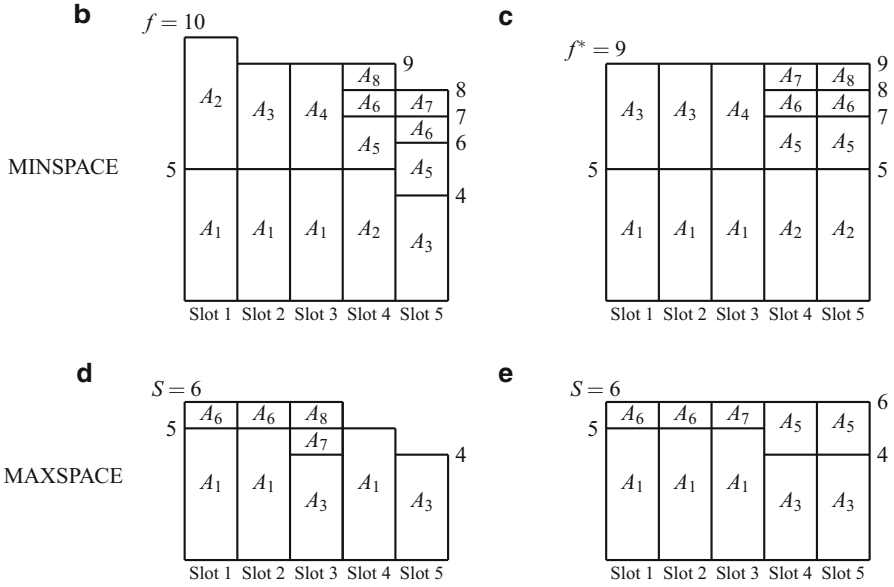

Fig. 3.1 An example to illustrate MINSPACE and MAXSPACE problems. (**a**) Problem data. (**b**) A feasible, nonoptimal schedule. (**c**) An optimal schedule. (**d**) A feasible, nonoptimal schedule. (**e**) An optimal schedule

However, in the MAXSPACE problem, the number of bins, N, is fixed and items are placed to maximize the space utilization of N bins. A survey of approximation algorithms for bin-packing is available in [4].

An application of bin-packing problem to the multiprocessor scheduling problem shows that the bin-packing problem is the dual problem of scheduling n independent tasks on m identical processors [3]. Based on the first-fit decreasing algorithm for bin-packing, an algorithm for multiprocessor scheduling, called the multifit algorithm, satisfies a bound of $\frac{6}{5}$ [3, 7], which is better than the bound given by LPT (i.e., $\frac{4}{3}$). Algorithms such as improved multifit [8] with a guarantee of $\frac{72}{61}$ and $1+\varepsilon$ approximation scheme [12] are also available for the multiprocessor scheduling problem.

3.4 MINSPACE Problem: Formulation and Complexity

We start by providing an integer programming formulation for the MINSPACE problem. Later, in Sect. 3.4.1.1, we present an approximation algorithm based on the linear programming relaxation of this formulation.

$$\begin{aligned} & \min F \\ \text{subject to} \quad & F \geq \sum_{i=1}^{n} s_i x_{ij}, \qquad j = 1,2,\ldots..,N. \qquad (IP_{min}) \\ & \sum_{j=1}^{N} x_{ij} = w_i, \qquad i = 1,2,\ldots..,n. \end{aligned}$$

where

$x_{ij} = 1$, if ad A_i is assigned to slot j,
$= 0$, otherwise.

where $w_i =$ frequency of ad A_i $i = 1,2,\ldots..,n$, $s_i =$ size of ad A_i $i = 1,2,\ldots..,n$. The first set of constraints, together with the objective, minimize the height of the schedule. The second set of constraints ensure that for ad A_i, a total of w_i copies are assigned to the slots. Since $x_{ij} \in \{0,1\}$, at most one copy of ad A_i can be assigned to a slot j. The MINSPACE problem is NP-hard in the strong sense (Theorem 2.1 in [1]).

3.4.1 Approximation Results for the MINSPACE Problem

An algorithm, called Largest-Size Least-Full (LSLF), for the MINSPACE problem works as follows [1]:

Algorithm LSLF [1]

Step 1: Sort the ads in nonincreasing order of their size. Let $s_1 \geq s_2 \ldots \geq s_n$.
Step 2: Assign the ads in the sorted order, where ad A_i is assigned to the w_i least full slots.

Step 1 requires time $O(n \log n)$. In Step 2, assigning each ad requires sorting of the N slots and hence takes time $O(N \log N)$. Therefore the total time required for Step 2 is $O(nN \log N)$. The time complexity of the algorithm is thus $O(n \log n + nN \log N)$. It is shown in [1] that LSLF is a 2-approximation for the MINSPACE problem. Moreover, in the special case when the ads, sequenced in nonincreasing order of their size, satisfy that $\frac{s_i}{s_{i+1}}$ is an integer, $i = 1,\ldots,n-1$, LSLF finds an optimum solution.

In [5], the authors propose a polynomial time approximation algorithms for the MINSPACE problem and prove the corresponding performance guarantees. Recall that, for a given schedule h, f^h denotes the maximum slot fullness among all the

slots, i.e., $f^h = \max_{1 \le j \le N} f_j^h$. In this context, scheduling refers to "deciding which ads will appear in a particular slot (time interval) and to determine the sequence in which the slots will appear." However, we deal only with the problem of assigning the ads to slots. The choice of which slot to display in a time interval can be made by cycling through a deterministic permutation of the slots corresponding to a given planning horizon.

The objective is to find a schedule, h^*, with $f^{h^*} = \min_h f^h$ where the minimum is taken over all valid schedules. For simplicity of notation, we denote the optimum solution value by f^* and use f to denote the solution value given by an approximation algorithm. Note that f^* is the optimum solution value for (IP_{min}). In [5], the authors present the following linear programming based 2-approximation algorithm for the MINSPACE problem, and then give an improved algorithm with a performance guarantee better than 2.

3.4.1.1 A Linear Programming Relaxation Based 2-Approximation Algorithm

Denote the linear programming (LP) relaxation of (IP_{min}) by LP_{min}. The assignment of ads to slots can be viewed as assignments in a bipartite graph, $G(V_1 \cup V_2, E)$, where nodes V_1 represent the ads and the nodes V_2 represent the slots. An edge $(i, j) \in E$ represents the assignment of ad A_i to slot j. Let $\bar{x}$ be an optimum basic feasible solution to LP_{min} and let f_{lp} denote the value of the optimum LP solution. Let g denote the vector of fractional variables of $\bar{x}$. Define the *residual graph*, R_G, as the subgraph of G induced by the edges of g. In [5], the authors present the following structural result.

Lemma 3.1. *The residual graph, R_G, of the linear programming relaxation LP_{min} is a forest [5].*

We can consider each connected component of R_G separately. Let $\bar{f}_j$ denote the fullness of slot j in $\bar{x}$. In R_G, if a slot node has degree 1, then we refer to that slot as a *singleton* slot. A simple rounding step can be used to remove a singleton slot from R_G. Consider a singleton slot j and let edge (i, j), where node i corresponds to ad A_i, be the single edge incident to slot j. Let $g_{ij} = \bar{x}_{ij}$. Let $g_{ij_1}, g_{i,j_2}, \ldots, g_{i,j_p}$ be the values corresponding to other edges in R_G incident on node i (see Fig. 3.2a).

Note that the quantity $g_{ij} + \sum_{k=1}^{p} g_{ij_k}$ is an integer, say z, since the frequency, w_i, of ad A_i is an integer. The rounding step sets $g_{ij} = 1$ and $g_{ij_k} = (\frac{z-1}{z-g_{ij}}) g_{ij_k}$, $k = 1, \ldots, p$ (Fig. 3.2b). Observe that the frequency of ad A_i remains unchanged in the solution after this rounding step. Moreover, the singleton slot j disappears from R_G since there are no fractional edges incident on it.

Note A rounding step on a singleton slot node removes at least one edge from the residual graph. We continue to denote the updated residual graph by R_G.

Each ad node in the residual graph has degree at least two since the ad frequency is an integer and the residual graph contains only the edges corresponding to the fractional variables. Since R_G has no cycle, there must exist at least one singleton slot. Otherwise, each slot node will have degree at least two and since each ad node

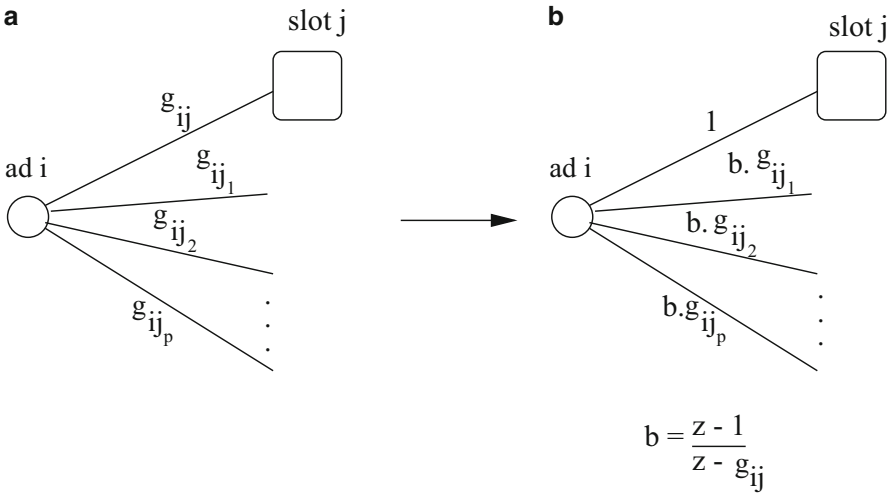

Fig. 3.2 Rounding step for a singleton slot (**a**) before rounding (**b**) after rounding

has degree at least two in R_G, there would exist a cycle in R_G which is a contradiction to Lemma 3.1. Moreover, at least one singleton slot continues to exist as the rounding procedure proceeds.

As the rounding procedure proceeds, the residual graph R_G continues to be acyclic since the original residual graph was acyclic and the rounding step does not introduce any new edges. Therefore, R_G has at least one node with degree 1. Moreover, this node cannot be an ad node as this would mean that there exists an ad node with degree 1 in R_G which contradicts the property that for each ad node i, the sum of edge weights incident on node i is the integer w_i. Therefore, as the rounding procedure proceeds by executing the rounding step on singleton slots in the residual graph R_G, it will always find at least one singleton slot unless the residual graph is empty (i.e. has no edges).

The description of the LP Rounding (LPR) algorithm is thus straightforward.

Algorithm LPR

Step 1: Obtain a vertex solution $\bar{x}$ to the LP_{min}. Construct the residual graph R_G based on the fractional variables.

Step 2: Find a singleton slot node in R_G and perform the rounding step as shown in Fig. 3.2. Update R_G. Repeat Step 2 until R_G is empty (i.e. R_G has no edges).

Using the properties discussed above, the following result is presented in [5].

Theorem 3.2. *LPR is a 2-approximation for the MINSPACE problem and this bound is asymptotically tight [5].*

Solving the linear programming relaxation in Step 1 requires time $O(n^3N^3L)$ where L is the length of the binary encoding of LP_{min} [15, 21]. Since the residual graphs are acyclic and the number of edges decreases by at least one in each iteration, Step 2

is repeated at most $n+N-1$ times. The rounding procedure for a singleton slot requires time $O(N)$. The time complexity of algorithm LPR is $O(n^3N^3L+N(n+N-1))$.

An interesting consequence of Theorem 3.2 is that the integrality gap, defined as the difference in the optimal solutions of an integer program and its linear programming relaxation, for IP_{min} is bounded above by s_{max}. That is, $f^* - f_{lp} < s_{max}$.

3.4.1.2 An Improved Approximation Algorithm

Unlike the LSLF algorithm, the next algorithm for the MINSPACE problem, called Largest-Frequency Least-Full (LFLF), assigns ads in the nonincreasing order of their frequency. The description of the algorithm follows.

Algorithm LFLF [5]

Step 1: Sort the ads by frequency, from largest to smallest.
Step 2: Assign the ads in the sorted order, where ad A_i is assigned to the w_i least full slots.

Step 1 requires time $O(n\log n)$ and Step 2 requires time $O(nN\log N)$. The time complexity of the algorithm is thus $O(n\log n+nN\log N)$. Since LSLF makes direct use of the size of an ad in its packing criterion, it is expected to perform better than LFLF. The next result, which is somewhat counter-intuitive, shows that the performance guarantee of LFLF is better than LSLF.

Theorem 3.3. *The performance bound for the algorithm LFLF for the MINSPACE problem is* $\frac{f}{f^*} \leq r$ *where: [5]*

1. $r = 2 - \frac{1}{N-w_0+1}$ *when* $w_0 \leq \frac{N+1}{2}$ *and*
2. $r = max\{1, \frac{2N}{N+2}(\frac{S^*-s_0}{S^*})\}$ *when* $w_0 > \frac{N+1}{2}$

and $w_0 = \min\{w_1,\ldots,w_n\}$, $s_0 = \min\{s_1,\ldots,s_n\}$, $S^* = \sum_{i=1}^{n} s_i$. *In both cases, the bound is tight.*

3.4.1.3 Constant Factor Approximation Algorithms

Further, in [6], the authors design polynomial time heuristics for the MINSPACE problem that provide solutions with a performance guarantee. In particular, they provide a solution within a *constant* factor of the optimum solution. Their results include:

1. An online $(2-\frac{1}{N})$-approximation algorithm.
2. A (offline) $(1+\frac{1}{\sqrt{2}})$-approximation algorithm.
3. A (offline) $\frac{3}{2}$-approximation algorithm.

3.4.2 Special Cases

In this section, we present improved approximation algorithms for three special cases of theoretical interest.

(1) when all of the ad frequencies are the same. That is, $w_i = w,\ i = 1,\ldots,n$. In this situation, we denote the MINSPACE problem by MIN_w.
(2) when all of the ad frequencies are the same and furthermore, the number of slots N is an integer multiple of the common frequency w. That is, $k = \frac{N}{w}$ is an integer. Here, we denote the MINSPACE problem by $\text{MIN}_{N|w}$.
(3) when $s_i = s,\ i = 1,\ldots,n$. Let MIN_s denotes this special case for the MINSPACE problem.

Since the MINSPACE problem is NP-hard in the strong sense (Theorem 2.1 in [1]), the problem MIN_w is also strongly NP-hard. We now observe that LSLF algorithm provides a much tighter bound for $\text{MIN}_{N|w}$. Next, in [5], the authors present the following result.

Theorem 3.4. *Algorithm LSLF is a* $(\frac{4}{3} - \frac{1}{3k})$*-approximation for* $MIN_{N|w}$ *and this bound is tight [5].*

Remark 1: As a consequence of the result above, multiprocessor scheduling algorithms such as multifit [7] and improved multifit [8] with performance guarantees of $\frac{6}{5}$ and $\frac{72}{61}$, respectively, and the $1 + \varepsilon$ approximation scheme [12] are also available for problem $\text{MIN}_{N|w}$.

Remark 2: The best known bound for problem MIN_w is the one established in Theorem 3.3 for algorithm LFLF.

Algorithms LSLF and LFLF, when used for problem MIN_s, provide a solution value of $s\lceil\frac{\sum_{i=1}^{n} w_i}{N}\rceil$. It is easy to see that this is optimal, thus proving that MIN_s is polynomially solvable. We record the result below.

Theorem 3.5. *Algorithms LSLF and LFLF solve problem* MIN_s *optimally.*

3.5 The MAXSPACE Problem

Given a set of ads $\{A_1,\ldots,A_n\}$, the number of slots, N, a common upper bound S on the fullness of each slot and the values for s_i, w_i for each A_i, we may not be able to assign all the ads given in A. Thus, we are required to find a subset $A' \subseteq A$ which forms a valid schedule and maximizes $\sum_{i \in A'} s_i w_i$. An integer programming formulation for the MAXSPACE problem is given below. Later, in Sect. 3.6, we use this formulation to present the optimum solution or an upper bound on the optimum solution.

$$\max \sum_{j=1}^{N} \sum_{i=1}^{n} s_i x_{ij}$$

$$\text{subject to} \quad \sum_{i=1}^{n} s_i x_{ij} \leq S, \qquad j = 1,2,\ldots..,N \qquad (IP_{max})$$

$$\sum_{j=1}^{N} x_{ij} = w_i y_i, \qquad i = 1,2,\ldots..,n.$$

where

$$x_{ij} = \begin{cases} 1, \text{ if ad } A_i \text{ is assigned to slot } j \\ 0, \text{ otherwise.} \end{cases} \qquad y_i = \begin{cases} 1, \text{ if ad } A_i \text{ is assigned,} \\ 0, \text{ otherwise.} \end{cases}$$

Note that w_i = frequency of ad A_i $i = 1,2,\ldots..,n$, s_i = size of ad A_i $i = 1,2,\ldots..,n$. The first set of constraints restrict the sum of ad sizes in a slot to be at most S. The second set of constraints ensure that, when an ad A_i is assigned, exactly w_i copies of the ad are placed in the slots. Since $x_{ij} \in \{0,1\}$, at most one copy of an ad can be placed in a slot. The MAXSPACE problem is NP-hard in the strong sense (Theorem 2.5 in [1]).

We now introduce the following subroutine which will simplify our presentation of subsequent algorithms:

LSLF(u,S): Application of Algorithm LSLF for the ads in set u, discarding any ad that, when placed, violates the limit S for the size of a slot.

3.5.1 Approximation Results for the MAXSPACE Problem

For the MAXSPACE problem, an algorithm called SUBSET-LSLF is presented in [1] that has arbitrarily bad performance. In this section, we present approximation algorithms for the MAXSPACE problem and the associated performance bounds. For an algorithm in this section, let $A' \subseteq A$ be the set of ads scheduled and let $p = \sum_{i \in A'} s_i w_i$ be the weight assigned. Let p^* be the weight assigned by an optimal solution.

We start by describing an algorithm, called MAX1, for the MAXSPACE problem that has been proposed in [5]. This algorithm decomposes the set of ads into two subsets based on frequencies. Based on the total weight of ads in each subset, the procedure gives priority to ads in one subset and achieves a performance guarantee of $\frac{1}{4} + \frac{1}{4N}$. Before proceeding with the description of the algorithm, we record two trivial upper bounds on p^*, the optimum solution value.

$$p^* \leq \sum_{i=1}^{n} s_i w_i = p_0 \qquad (3.1)$$

and

$$p^* \leq SN. \tag{3.2}$$

Algorithm MAX1 [5]

Step 1: Given s_i, w_i, $i = 1,2,\ldots,n$, the number of slots N and the size of each slot S, let $\underline{u} = \{A_i \mid w_i < \frac{N+1}{2}\}$, $\overline{u} = \{A_i \mid w_i \geq \frac{N+1}{2}\}$, $B_{\underline{u}} = \sum_{A_i \in \underline{u}} s_i w_i$, $B_{\overline{u}} = \sum_{A_i \in \overline{u}} s_i w_i$.

Step 2: Let u be the set among $\underline{u}, \overline{u}$ for which $B_u \geq \frac{p_0}{2}$. Then,

2(a). Apply LSLF(u,S); let S_u be the resulting partial assignment.

2(b). Starting with S_u, apply LSLF$(A-u,S)$.

The time complexity of MAX1 is $O(n\log n + nN\log N)$. The proof is available in [5].

Theorem 3.6. *The performance bound for the algorithm MAX1 for the MAXSPACE problem is* $\frac{p}{p^*} \geq \frac{1}{4} + \frac{1}{4N}$. *Moreover, this bound is tight [5].*

To achieve a better performance guarantee, a more detailed analysis of the problem input is required. Our next algorithm, MAX2 (proposed in [5]), uses the knowledge of ad sizes and frequencies in a more effective way to obtain a bound of $\frac{3}{10}$. The description and the analysis of this algorithm follows.

Algorithm MAX2 [5]

Step 1: Given s_i, w_i, $i = 1,2,\ldots,n$, the number of slots N and the size of each slot S, let $\underline{v} = \{A_i \mid w_i \leq \frac{2N}{5}\}$, $\overline{v} = \{A_i \mid w_i > \frac{2N}{5}\}$, $B_{\underline{v}} = \sum_{A_i \in \underline{v}} s_i w_i$, $B_{\overline{v}} = \sum_{A_i \in \overline{v}} s_i w_i$. Set $p_0 = SN$.

Step 2: If $B_{\underline{v}} \geq \frac{3p_0}{10}$, then apply LSLF$(\underline{v},S)$ and terminate.

Step 3: If $B_{\overline{v}} \geq \frac{7p_0}{10}$, then ads in $\overline{v}$ are divided into two sets as follows. Let $\underline{c} = \{A_i \mid s_i \geq \frac{S}{4}\}$, $\overline{c} = \{A_i \mid s_i < \frac{S}{4}\}$.

Step 4: Placing ads in $\underline{c} = \{A_i \mid s_i \geq \frac{S}{4}\}$. Find $k \leq 3$ and ads $A_{r_1},\ldots,A_{r_k}$ in $\underline{c}$ having $\sum_{j=1}^{k} s_{r_j} w_{r_j} \geq \frac{3p_0}{10}$ and which can be assigned without violating the size limit for any slot. If successful, assign ads $A_{r_1},\ldots,A_{r_k}$ and terminate.

Step 5: Placing ads in $\overline{c} = \{A_i \mid s_i < \frac{S}{4}\}$. Apply LSLF$(\overline{c},S)$. If $B_{\overline{c}} = \sum_{A_i \in \overline{c}} s_i w_i \geq \frac{3p_0}{10}$ or if at least one ad is rejected when placing the ads from $\overline{c}$, terminate.

Step 6: If LSLF$(\underline{v},S)$ assigns all the ads in $\underline{v}$, then apply LSLF$(\underline{v},S)$, LSLF$(\overline{c},S)$ and the result of Step 4 to solve the problem optimally for the sets of ads $\underline{v}$, $\overline{c}$ and $\underline{c}$. Let $p^*(\underline{v})$, $p^*(\overline{c})$ and $p^*(\underline{c})$ be the corresponding values. Select the schedule associated with the largest of these three values.

The time complexity of algorithm MAX2 is $O(n^3N + nN\log N)$. The proof is available in [5].

Theorem 3.7. *The performance bound for the algorithm MAX2 for the MAXSPACE problem is* $\frac{p}{p^*} \geq \frac{3}{10}$ *and this bound is tight [5].*

3.5.2 Hybrid Genetic Algorithm

For the MAXSPACE problem, in [16], the authors first propose an algorithm, called LSMF, based on the idea of a classical bin-packing algorithm. Then, they develop a genetic algorithm (GA) to solve the problem. Finally, they design a hybrid GA by combining GA with LSMF and an existing algorithm. In order to improve the performance of GA, they use the design of experiments to find the good GA parameter settings before running the GA.

3.5.3 Special Cases

In this section, we present improved approximation algorithms for three special cases of theoretical interest.

(1) when all of the ad frequencies are the same. That is, $w_i = w,\ i = 1,\ldots,n$. In this situation, we denote the MAXSPACE problem by MAX_w.
(2) when all of the ad frequencies are the same and furthermore, the number of slots N is an integer multiple of the common frequency w. That is, $k = \frac{N}{w}$ is an integer. Here, we denote the MAXSPACE problem by $\text{MAX}_{N|w}$.
(3) when $s_i = s,\ i = 1,\ldots,n$. Let MAX_s denotes this special case for the MAXSPACE problem.

Problems MAX_w and $\text{MAX}_{N|w}$ are NP-hard since an arbitrary instance of the 0–1 knapsack problem [18] is a special case of both problems. MAX_s is NP-hard since the subset-sum problem [18] is a special case. Below, we describe a $\frac{1}{3}$-approximation algorithm (proposed in [5]), called MAX3, for problem MAX_w. The same algorithm is a $\frac{1}{2}$-approximation for $\text{MAX}_{N|w}$.

Algorithm MAX3 [5]

Step 1: If $w > \frac{N}{2}$, then
1(a). Find an ad A_{r_1} having $s_{r_1} \geq \frac{NS}{3w}$. If successful, assign the ad A_{r_1} and go to the post-processing step, Step 3.
1(b). Find any two ads A_{r_1} and A_{r_2} having $s_{r_1} + s_{r_2} \geq \frac{NS}{3w}$ and which can be assigned without violating the size limit for any slot. If successful, assign ads A_{r_1} and A_{r_2} and go to post-processing step, Step 3.
1(c). Apply LSLF(A, S).
Step 2: If $w \leq \frac{N}{2}$, then
2(a). Sort the ads by size s_i from largest to smallest. The N slots are divided into $b+1$ groups where the first b groups contains exactly w slots and $bw > (N-w)$. The last group contains $(N-bw)$ slots, where $(N-bw) < w$.
2(b). Ads are placed in such a way that all the copies of an ad are assigned to the slots in the same group and an ad A_i is assigned to the slots in group $l+1$ if and only if it cannot be assigned to the slots in groups 1,...,l without violating the size limit of slots in these groups.

Step 3: **Post-processing step:** Sort the remaining unassigned ads by size s_i from largest to smallest. Assign the ads in the sorted order such that each ad A_i is assigned to w least full slots. Discard any ad that, when placed, would violate the size limit for some slot.

The time complexity of algorithm MAX3 is $O(n^2N + nN\log N)$. The proof of the following theorem is available in [5].

Theorem 3.8. *Algorithm MAX3 is a* $\frac{1}{3}$*-approximation for* MAX_w *[5].*

For problem $\text{MAX}_{N|w}$, the heuristic solution, p, from algorithm MAX3 satisfies $\frac{p}{p^*} \geq \frac{1}{2}$. The proof of this result follows from that of Theorem 3.8 by noting that $w \leq \frac{N}{2}$ and $bw = N$. Again, the tightness of this bound is easy to show. We state the result below.

Corollary 3.9. *Algorithm MAX3 provides a* $\frac{1}{2}$ *approximation for problem* $MAX_{N|w}$ *and this bound is tight.*

It is easy to observe that problem MAX_s can be solved by solving the well-known 0–1 knapsack problem [10, 18, 21]: The item weights as well as item profits for the knapsack problem are the frequencies w_i, $i = 1, \ldots, n$ and the knapsack capacity is $N\lfloor \frac{S}{s} \rfloor$. As such, a pseudopolynomial algorithm is available for solving the problem to optimality [18]. Also, a number of approximation algorithms are known for the problem including a $1 + \varepsilon$ approximation scheme [13].

3.6 Computational Experience

In this section, we report on the computational performance of the algorithms presented in this chapter. In order to examine the performances of algorithms proposed by the authors in [5] for both MINSPACE and MAXSPACE problems (i.e., LFLF, MAX1, and MAX2), they conduct experiments with a test bed of 10 sets of problems with each set consisting of 10 problem instances. They find that the average performance of the heuristics is much stronger than their provable worst case guarantees: The average percentage gap between the heuristic and optimum solutions is approximately within 30 %, 15 % and 20 % for heuristics LFLF, MAX1, and MAX2, respectively.

In [16], the authors also conduct experiments to examine the performance of their hybrid GA for the MAXSPACE problem. They find that hybrid GA provides the optimal solution for all the test problems with known optimal values. For other test problems, the hybrid GA solutions are within 3 % of the upper bounds generated by CPLEX on the average. It provides up to 87 % improvement over the existing heuristic proposed by Adler et al. [1]. Further, in [16], the authors implement the proposed hybrid GA for an actual dataset. This dataset is obtained by observing the ads on ValuePay's pIggy Adbar for 1 h. In this case study, the improvements were of the order of 4–24 %. They report that it can result in substantial improvement

in revenue. For example, a 4 % increase in revenue in the case study could lead to $22,176 increase in monthly revenue, while a 24 % increase could lead to an increase in revenue of $111,240 per month.

References

1. Adler M, Gibbons PB, Matias Y (2002) Scheduling space-sharing for internet advertising. J Sched 5(2):103–119.
2. Chen B, Potts CN, Woeginger G (1998) A review of machine scheduling: complexity, algorithms and approximability. In: Du DZ, Pardalos PM (eds) Handbook of combinatorial optimization, vol. 3, pp. 21–169. Kluwer Academic Publishers, New York
3. Coffman EG Jr, Garey MR, Johnson DS (1978) An application of bin-packing to multiprocessor scheduling. SIAM J Comput 7(1):1–17
4. Coffman EG Jr, Garey MR, Johnson DS (1997) Approximation algorithms for bin-packing: a survey. In: Hochbaum DS (ed) Approximation algorithms for *NP*-hard problems. PWS Publishing Company, Boston, pp 46–93
5. Dawande M, Kumar S, Sriskandarajah C (2003) Performance bounds of algorithms for scheduling advertisements on a web page. J Sched 6(4):373–393
6. Dawande M, Kumar S, Sriskandarajah C (2005) Scheduling web advertisements: a note on the MINSPACE problem. J Sched 8(1):97–106
7. Friesen DK (1984) Tighter bounds for multifit processor scheduling algorithm. SIAM J Comput 13:170–181
8. Friesen DK, Langston MA (1986) Evaluation of a multifit-based scheduling algorithm. J Algorithms 7:35–59
9. Garey MR, Johnson DS (1979) Computers and intractability: a guide to the theory of *NP*-completeness. Freeman, San Francisco
10. Garfinkel RS, Nemhauser GL (1972) Integer programming. Wiley, New York
11. Graham RL (1969) Bounds on multiprocessing timing anomalies. SIAM J Appl Math 17:416–429
12. Hochbaum DS, Shmoys DB (1987) Using dual approximation algorithms for scheduling problems: theoretical and practical results. J Assoc Comput Mach 34:144–162
13. Ibarra OH, Kim CE (1975) Fast approximation algorithms for the knapsack and sum of subset problems. J Assoc Comput Mach 22:463–468
14. Interactive Advertising Bureau (2015) Guidelines, specifications & best practices. http://www.iab.net/guidelines.Retrieved9Feb2015
15. Kipp MR (1999) Large scale linear and integer optimization. Kluwer Publishers, Boston
16. Kumar S, Jacob VS, Sriskandarajah C (2006) Scheduling advertisements on a web page to maximize revenue. Eur J Oper Res 173:1067–1089
17. Lawler EL, Lenstra JK, Rinnooy Kan AHG, Shmoys DB (1993) Sequencing and scheduling: algorithms and complexity. In: Graves SC et al (eds) Handbooks in operations research and management science, vol 4, pp 445–552. Elsevier Ltd, Oxford, UK
18. Martello S, Toth P (1989) Knapsack problems. Wiley, New York
19. McCandless M (1998) Web advertising. IEEE Intell Syst 3:8–9
20. McGuire T (2001) Internet: a stream of light. Intermedia 29:19–21
21. Nemhauser GL, Wolsey LA (1988) Integer programming and combinatorial optimization. Wiley, New York
22. Pinedo M (1995) Scheduling theory, algorithms, and systems. Prentice Hall, Upper Saddle River

Chapter 4
Personalization of Web Advertising

Abstract As discussed in Chap. 2, the performance-based pricing has become the most prevalent web advertising pricing model. Hence, in this chapter, we consider a hybrid pricing model where the price advertisers pay is a function of (a) the number of exposures of the ad, and (b) the number of clicks on the ad. The problem is to find an ad schedule to maximize web site revenue under a hybrid pricing model. Further, as discussed in earlier chapters, the Internet is the first commercial medium in which it is actually possible to measure consumer response, not just to assume it. Both the advertisers and the website providers attempt to exploit this unique, interactive nature of the medium. Therefore, in this chapter, we consider two versions of the problem: *static* and *dynamic*, and present a variety of efficient solution techniques that provide near-optimal solutions. In the dynamic version, the schedule of ads is changed based on individual user click behavior. We show that a schedule that adapts to user click behavior consistently outperforms one that does not. We also demonstrate that to benefit from observing user click behavior, the associated probability parameter need not be estimated accurately. For both these versions, we present the sensitivity of the revenue with respect to the model parameters.

4.1 Introduction

Typically, a web site displays a specific sequence of ads to each user during her visit to the site. For example, the site may display a specific set of ads to a user during the first minute of the visit and a different set of ads during the next minute of the visit. Ads compete for exposure during each time interval (e.g., 1 min) and the goal is usually to optimize some objective calculated for the visit. For the purposes of this study, the *planning horizon* is the duration of a user's visit. For example, a typical planning horizon may consist of 10 min. A scheduling problem that arises is one of choosing a subset of ads for each time interval within the planning horizon.

S. Kumar, *Optimization Issues in Web and Mobile Advertising*, SpringerBriefs in Operations Management, DOI 10.1007/978-3-319-18645-0_4

The models discussed in the previous chapter considered the web ad scheduling similar to (or identical to) the ad scheduling in traditional media, such as a magazine or a newspaper. However, in this chapter, we consider the unique aspects of web as a medium for advertising. First, we focus on optimizing advertising objectives at the level of a single user. In the last decade, web advertising targeted at the individual user has gained popularity [3, 4, 8]. Second, the model in this chapter incorporates results studies on clickstream analysis that predict the probability of an ad being clicked upon during a user's visit. In particular, the model in this chapter is based on the observations that the repeated exposure of an ad during a user's visit has a negative linear effect and a positive quadratic effect on the click rate. Here, the magnitude of the negative coefficient of the linear term is significantly larger than the positive coefficient of the quadratic term. Thus, the positive quadratic effect begins to dominate only after a certain number of exposures of an ad. These observations are central to our discussion in this chapter. Third, a novel aspect of the model in this chapter is the ability to adapt the sequence of ads shown to a user based upon the user's actual click behavior.

4.2 Problem Statement, Assumptions, and Approach

4.2.1 Problem Statement and Assumptions

Given a set of ads that can be displayed to a user and a planning horizon consisting of a sequence of slots, the goal is to select a subset of ads in each slot to maximize a revenue objective of the web site owner. Next, we discuss problem constraints and revenue calculation.

There are three problem constraints: *size constraint*, *exposure constraints*, and *pairwise ad constraints*. The size constraint in this model is the same as that in the MAXSPACE problem (in previous chapter) where the sum of the heights of the ads selected for a slot is at most the height of the slot. The exposure constraints—motivated by the web ad scheduling practice—ensure that, for each slot, at most one copy of any ad is selected for display. The pairwise ad constraints are of two kinds: (a) *inclusion constraints*, and (b) *exclusion constraints*. An inclusion constraint is a constraint that applies to an ordered pair of ads and requires that the first ad in the pair be exposed in a slot only if the second ad in the pair is exposed in that slot. Such constraints represent situations that correspond to complementary products (e.g., a real estate and a mortgage ad could be required to be exposed together). An inclusion constraint may also be used to model a competitive situation where two firms with competing ads require that their ads be exposed simultaneously in a slot. An exclusion constraint, on the other hand, imposes that whenever one of the ads in the pair is exposed, the other ad in the exclusion pair cannot be shown.

In general, exclusion constraints capture two types of situations: (a) where a very effective ad could adversely affect the value of another less effective ad, and (b) two competitors do not want their ads to be displayed together.

The objective is to maximize the total expected revenue in the planning horizon (i.e., the duration of a user's visit) which is the sum of the expected revenues in each slot. For a slot, the expected revenue is the sum of the expected revenue of each ad displayed in the slot. The expected revenue, per unit size, of an ad in a slot is derived using a hybrid pricing model. In this model, an ad is charged a fixed amount for exposure in a slot and a variable amount that depends on whether or not the ad is clicked upon by the user as a result of that exposure.

The probability of a click resulting from the kth exposure of an ad during a user's visit depends on two effects: *exposure effect* and *re-click effect* [6].

- *exposure effect*: In web advertising, the differential impact of each successive ad exposure during a user's visit is initially negative and non-linear but becomes positive later at higher levels of ad exposure. This U-shaped functional form has been observed in recent research on individual clickstream analysis using data from a commercial web site and is different from the *inverted* U-shaped response found in traditional broadcast media. Due to the exposure effect, the click probability in the kth exposure of ad A_i is

$$e_{ik} = -a_{i1}k + b_{i1}k^2 + a_{i0} \tag{4.1}$$

  where a_{i0}, a_{i1} and b_{i1} are positive constants with $a_{i1} >> b_{i1}$.
- *re-click effect*: A user clicking on an ad during a visit increases her probability of clicking on that ad in future exposures during the *same* visit. Thus, if an ad has been clicked upon, the re-click effect increases the click probability by an amount p in all subsequent exposures of the ad during the current visit.

The two effects, exposure and re-click, capture the typical behavior of a consumer on a website. The decreasing portion of the U-shaped exposure effect has been attributed to *wear-out*, while the increasing portion occurs due to *wear-in* or *familiarity* [2, 9]. The re-click effect has been attributed to a consumer's interest in an ad [7].

We consider two versions of the problem. In the *static* version, click events are not observed and scheduling decisions are therefore made based upon the total expected click probability. In the *dynamic* version, the user click behavior during a visit is observed and this information is exploited in scheduling decisions.

In the static version, the total expected click probability, C_{ik}, for the kth exposure of ad A_i can be calculated as follows. For the first exposure, $C_{i1} = e_{i1}$, where e_{ik} is the click probability in the kth exposure of ad A_i due to the exposure effect. For the second exposure, the total expected click probability is $C_{i2} = C_{i1}(e_{i2} + p) + (1 - C_{i1})e_{i2} = pC_{i1} + e_{i2}$, representing an expectation of the two possible values of the click probability in the second slot: $e_{i2} + p$, if the ad is clicked upon in the first slot (with probability C_{i1}) and e_{i2} otherwise. In general, for $k \geq 2$,

$$C_{ik} = \left\{ 1 - \prod_{l=1}^{k-1} (1 - C_{il}) \right\} p + e_{ik} \tag{4.2}$$

In the dynamic version, the probability of a click in the kth exposure of ad A_i is given by:

$$C_{ik} = e_{ik} + p' \tag{4.3}$$

where $p' = 0$ if ad A_i has not been clicked upon thus far during the visit (i.e., in exposures 1 through $k-1$), and $p' = p$ otherwise.

In both the static and dynamic versions, the expected revenue, per unit size, from the kth exposure of ad A_i is given by:

$$R_{ik} = a_{i2} + b_{i2} C_{ik} \tag{4.4}$$

where a_{i2} and b_{i2} are positive constants.

4.2.2 Approach

We first present a linear integer programming formulation for the static version of the revenue maximization problem. Although realistic instances can be solved to optimality in a few minutes using state-of-the-art integer programming solvers, we present a heuristic solution for two reasons. First, since the problem class defined by the static version is strongly NP-hard, it is conceivable that obtaining an optimum solution may become difficult for extremely large instances. Second, an effective algorithm (discussed in Sect. 4.5) to solve the dynamic version requires repeated solutions of the static version. Since providing real-time solutions is a requirement for the dynamic version, a fast and near-optimal heuristic for the static version becomes a necessity. The heuristic proposed in [6] solves a sequence of knapsack-like problems (one for each slot) and is easy to implement, runs very quickly, scales reasonably well, and provides good quality solutions.

4.3 Formulating the Static Version

Consider a set of n ads $A = \{A_1, \ldots, A_n\}$ competing for space in a planning horizon that is divided into N time intervals (slots). Each slot is a rectangle of height S and width W. Ad A_i has height s_i. We assume that the width of all the ads is W. Ad A_i is said to be *scheduled* in slot j if one copy of A_i appears in that slot. A feasible schedule is a placement of a subset $A'_j \subseteq A$ of ads in slot j such that the following condition is satisfied: for $j = 1, \ldots, N$, the sum of ad sizes assigned to slot j must not exceed S. That is, $\sum_{A_i \in A'_j} s_i \leq S \; \forall \; j$. Clearly, $s_i \leq S \; \forall \; i$.

Notation:

N	number of slots.
n	number of ads.
S	height of a slot.
D_{in}	set of pairs of ads on which an inclusion constraint is imposed: $(A_u, A_v) \in D_{in}$ implies that ad A_u is exposed in a slot only if ad A_v is also exposed in that slot.
D_{ex}	set of pairs of ads on which an exclusion constraint is imposed: $(A_u, A_v) \in D_{ex}$ implies that at most one of A_u and A_v can be exposed in a slot.
R_{ik}	expected revenue, per unit size, from the kth exposure of ad A_i.
r_{ik}	expected revenue from the kth exposure of ad A_i; $r_{ik} = s_i R_{ik}$.
T_{ik}	expected revenue from k exposures of ad A_i; $T_{ik} = \sum_{l=1}^{k} r_{il}$.
z_{ik}	1 if ad A_i is exposed a total of k times; 0 otherwise.
x_{ij}	1 if ad A_i is scheduled in slot j; 0 otherwise.

$$\text{Maximize} \quad \sum_{i=1}^{n} \sum_{k=1}^{N} T_{ik} z_{ik}$$

subject to

$$\sum_{j=1}^{N} x_{ij} = \sum_{k=0}^{N} k.z_{ik}; \quad i = 1,2,\ldots,n \tag{4.5}$$

$$\sum_{k=0}^{N} z_{ik} = 1; \quad i = 1,2,\ldots,n \tag{4.6}$$

$$x_{uj} + x_{vj} \leq 1; \quad j = 1,2,\ldots,N; \ (A_u, A_v) \in D_{ex} \tag{4.7}$$

$$x_{uj} - x_{vj} \leq 0; \quad j = 1,2,\ldots,N; \ (A_u, A_v) \in D_{in} \tag{4.8}$$

$$\sum_{i=1}^{n} s_i x_{ij} \leq S; \quad j = 1,2,\ldots,N \tag{4.9}$$

$$x_{ij} \in \{0,1\}, \quad i = 1,2,\ldots,n; \quad j = 1,2,\ldots,N$$

$$z_{ik} \in \{0,1\}, \quad i = 1,2,\ldots,n; \quad k = 1,2,\ldots,N$$

$z_{ik} = 1$ implies that ad A_i has been displayed a total of k times and, hence, $\sum_{j=1}^{N} x_{ij} = k$. Constraints (4.5) and (4.6) ensure this relationship. For each pair of ads in D_{ex}, Constraint (4.7) imposes that at most one ad can be exposed in each slot. For each ordered pair of ads $(A_u, A_v) \in D_{in}$, Constraint (4.8) imposes that the first ad in the pair be exposed in a slot only if the second ad in the pair is also exposed in that slot. Thus, for each slot j, $x_{uj} = 1$ implies $x_{vj} = 1$. Constraint (4.9) ensures that the sum of the heights of the ads selected for a slot is at most the height of the slot.

It is easy to see that the static problem is strongly NP-hard: an arbitrary instance of the strongly NP-hard Multiple Subset Sum Problem [1] can be polynomially reduced to an instance of the static problem. While we could easily solve realistic

instances of the static problem to optimality using the CPLEX optimizer, the computation times required to obtain an optimal solution can be substantial for very large instances. Also, most state-of-the-art integer program solvers employ a variety of techniques—search strategies, different types of cutting planes, heuristics, etc—to make the search for integer solutions efficient. Therefore, there is typically a significant amount of variability in the performance of different solvers on the same class of problems. Thus, in [6], the authors propose an efficient heuristic that provides near-optimal solutions quickly, and scales well with problem size.

4.3.1 A Successive Slot Knapsack (SSK) Heuristic

The basic idea behind the SSK heuristic is easy to explain: if the planning horizon consisted of a single slot, then the revenue maximization problem is similar to the classical 0–1 knapsack problem [5] with the only difference being the presence of the inclusion and exclusion constraints. For a problem involving more than one slot, such a problem can be solved successively for each slot. That is, we can first determine the optimal allocation of ads in first slot. Then, using the result of this allocation (i.e., the set of ads which are displayed in the first slot), the expected revenue of the ads displayed in the first slot can be updated and an optimal allocation for the second slot can be obtained. The allocation for the second slot then sets up the problem for the third slot and so on.

Consider an ad $A_i \in A$. Let m_{ij} be the expected revenue from assigning ad A_i to slot j. The value m_{ij} depends on the number of exposures of ad A_i in the slots preceding slot j (i.e., slots $1,2,\ldots,j-1$). Using the same notation as in the integer programming formulation of the previous subsection, it is easy to see that, for $j \geq 2$, $m_{ij} = r_{ik}$, where $k = \sum_{q=1}^{j-1} x_{iq} + 1$; and $m_{i1} = r_{i1}$. Thus, given the ad assignments for the slots preceding slot j, the values m_{ij} can be computed and the following modified-knapsack problem can be solved to maximize the expected revenue for slot j:

$$
\begin{aligned}
MKP(j) = \text{Maximize} \quad & \sum_{i=1}^{n} m_{ij} x_{ij} \\
\text{subject to} \quad & \\
x_{uj} + x_{vj} &\leq 1;\ (A_u, A_v) \in D_{ex} \\
x_{uj} - x_{vj} &\leq 0;\ (A_u, A_v) \in D_{in} \\
\sum_{i=1}^{n} s_i x_{ij} &\leq S \\
x_{ij} &\in \{0,1\},\ \ i = 1,2,\ldots,n
\end{aligned}
$$

The optimal allocation obtained by solving problem $MKP(j)$ can then be used to set up the problem for the next slot. A formal description of the SSK heuristic follows.

Algorithm SSK [6]

Step 1: Define $m_{i1} = r_{i1}$; $i = 1,2,\ldots,n$. Set $j = 1$.
Step 2: Solve problem $MKP(j)$. Let $x_{ij}, i = 1,2,\ldots,n$, be an optimal solution for $MKP(j)$. Set $j = j+1$.
Step 3: If $j \leq N$, update the values of $m_{ij}, i = 1,2,\ldots,n$, as follows: $m_{ij} = r_{ik}$, where $k = \sum_{q=1}^{j-1} x_{iq} + 1$ and go to Step 2; otherwise, terminate.

Algorithm SSK is essentially a greedy algorithm wherein the emphasis, while deciding the schedule for a slot, is on maximizing the revenue of only that slot. The problem to be solved for each slot, namely, the modified-knapsack problem is itself an NP-hard problem [5], but can be solved quite efficiently in practice using state-of-the-art integer program solvers.

4.4 Computational Experience: Static Version

The computational experiments conducted in [6] indicate that the quality of the solutions obtained by the SSK heuristic is very good over a wide variety of problem instances. Also, they report that the SSK heuristic is very quick. In [6], the authors also conduct sensitivity analysis to examine the importance of arriving at precise estimates for the click probability parameters for different families of ad scheduling problems. A secondary goal of sensitivity analysis is to demonstrate the impact of the revenue parameters on the scheduling objective, namely, the expected revenue to the web site.

The results of sensitivity analysis in [6] show that the impact of a particular click probability parameter on the revenue is usually significant when small changes in the value of the parameter permit or restrict ads from being displayed in the increasing part of the click probability curve. However, the effect of a click probability parameter is influenced by the primary parameter settings: when the planning horizon is short (or there are many ads), changes in the click probability parameters may have very little impact on the revenue. The sensitivity analysis also demonstrate that the revenue parameters (a_2 and b_2) have a direct and linear impact on the revenue.

4.5 A Dynamic Version

A drawback of the revenue maximization model in Sect. 4.3 is that it does not take advantage of a user's click behavior during a visit. Therefore, in [6], the authors propose a dynamic version that aims at exploiting this behavior. Recall that the main idea in the dynamic version is to revise, at the end of a slot, the user's probability of clicking on an ad, and incorporate this revised estimate while deciding on the schedule in forthcoming slots. In [6], the authors use a fixed positive probability p to capture the re-click effect.

Note that the probability revision, to be used for scheduling ads during a visit, requires the knowledge of the *actual* click behavior during all previous slots in the same visit. We therefore have a dynamic (or online) version of the revenue maximization problem. The dynamic algorithm begins by solving the static problem for N slots. From the corresponding solution, we use the allocation of ads in the first slot. Then, actual click events for these ads in the first slot are observed and an allocation for the second slot is obtained by again solving the static problem for $N-1$ slots and so on.

Let u_{ik} be the click probability of an ad A_i in its kth exposure and η be the number of slots that have been displayed when u_{ik} is being calculated. The value of u_{ik} depends on the actual click events of ad A_i in the η displayed slots. Let us denote the number of exposures of ad A_i in these η slots as $M_{i\eta}$. Note that $M_{i\eta} = \sum_{q=1}^{\eta} x_{iq}$. We need to calculate u_{ik} for each ad $A_i \in A$ and each exposure k where $M_{i\eta} < k \leq (N - M_{i\eta})$. It is easy to see that $u_{i1} = e_{i1}$. Next, we calculate the value of u_{ik} for $k \geq 2$. If ad A_i has been clicked in any of the η displayed slots, then $u_{ik} = e_{ik} + p$; otherwise u_{ik} is calculated in a manner similar to Eq. (4.2) as follows: for $k = M_{i\eta} + 1$, $u_{ik} = e_{ik}$; and for $k > M_{i\eta} + 1$, $u_{ik} = e_{ik} + \left\{1 - \prod_{q=M_{i\eta}+1}^{k-1}(1 - u_{iq})\right\} p$. Recall from Sect. 4.3 that T_{ik} is the total expected revenue from k exposures of Ad A_i. Using these values of u_{ik} and Eq. (4.4), we can easily calculate the values of T_{ik}. The optimal allocation for $(\eta+1)$th slot can now be obtained by solving the static problem for the last $N - \eta$ slots using the values of T_{ik}. A formal description of the algorithm is given below.

A Look-Ahead Dynamic Algorithm [6]

Step 1: Solve the static problem (Sect. 4.3) for N slots. Based on the solution, schedule the ads in the first slot. Set $\eta = 1$.
Step 2: Observe actual click events for the ads scheduled in slot η.
Step 3: As explained above, calculate the updated values of u_{ik} for each ad $A_i \in A$ and each exposure k where $M_{i\eta} < k \leq (N - M_{i\eta})$. Here, $M_{i\eta} = \sum_{q=1}^{\eta} x_{iq}$.
Step 4: Using Eq. (4.4) and the values of u_{ik}, calculate the updated values of T_{ik}.
Step 5: Solve the static problem again for the last $N - \eta$ slots using the updated values of T_{ik}. Schedule the ads in $(\eta+1)$th slot based on the solution of this problem.
Step 6: If $\eta = N - 1$, then terminate; otherwise set $\eta = \eta + 1$ and go to Step 2.

4.5.1 Generation of Hypotheses and Statistical Validation

The dynamic version should outperform the static version because of its information advantage (pertaining to click events) over the static version. Our goal in this section is to study the effect of the different parameters on the amount of gain resulting from this information advantage. The mathematical analysis of the comparison of the static and dynamic versions encounters two difficulties. First, since the scheduling

problem is NP-hard, there is no known non-enumerative way to express its solution. Second, tracking the information advantage of the dynamic version across many slots is difficult. However, tracking this advantage from one slot to the next is the crux of the issue.

Based on the above discussion, in [6], the authors propose the following three hypotheses:

Hypothesis 1 *If* $p < \min_i \min_{\{k:e_{ik} > e_{i(k+1)}\}} (e_{ik} - e_{i(k+1)})$*, and there are sufficient ads available so that it is possible to fill all the slots without repeating an ad, then the expected revenue is the same in the static and dynamic versions.*

Hypothesis 2 *If the expected click probability from repeating an ad* A_i *is higher than that from selecting a new ad (i.e.,* $C_{i(k+1)} > C_{j1},\ \forall j \neq i, \forall k$*), and there are sufficient ads available so that it is possible to fill all the slots without repeating an ad, then the difference in the expected revenue between the dynamic and static versions increases or remains the same with an increase in* a_{i1}*.*

Hypothesis 3 *The difference in the expected revenue between the dynamic and the static versions increases or remains the same with an increase in p.*

In [6], the authors first present the mathematical justification of these hypotheses based on the analysis of the restricted version of the problem where equal-sized ads with the same effectiveness level and the same revenue parameter values are scheduled over two slots. For the general case (i.e., arbitrary ad sizes with different effectiveness levels, and more than two slots), they conduct experiments with a wide variety of problem instances [6]. Their experimental results provide support for all three hypotheses.

4.5.2 A Myopic SSK Heuristic for the Dynamic Version

Obtaining a dynamic schedule requires us to repeatedly solve the static problem. Since the static problem is strongly NP-hard, there is a need for an efficient heuristic that can provide an optimal or a near-optimal solution for large instances. Therefore, in [6], the authors propose a dynamic version of the SSK heuristic, and call it the *myopic algorithm*.

The dynamic version of the SSK heuristic exploits the knowledge gleaned from the user click behavior during a visit [6]. The only difference between this heuristic and the static version of algorithm SSK (given in Sect. 4.3.1) is in calculation of the value of m_{ij} at Step 3. Recall that m_{ij} is the expected revenue from assigning ad A_i to slot j. In the dynamic version, $m_{ij} = a_{i2} + b_{i2}(e_{ik} + p')$, where $k = \sum_{q=1}^{j-1} x_{iq} + 1$. Here, $p' = p$ if the ad A_i has been clicked in any of the slots preceding slot j, or $p' = 0$ otherwise.

The experimental results in [6] show that the revenue of the myopic algorithm is within 0.2 % of the revenue of the look-ahead dynamic algorithm. Therefore, we

can conclude that the myopic algorithm is an effective and efficient alternative to the look-ahead dynamic algorithm, especially for large instances. Since the dynamic schedule operates in real-time, the myopic algorithm (which solves very quickly) should be a strong candidate for use in practice.

4.6 Discussions and Recommendations

We now summarize and discuss the main results, and offer guidelines for solving the ad-scheduling problem in practice. Here, we address the overall question regarding the choice of a scheduling method, namely static versus dynamic. The analysis and computational tests presented in this chapter illustrate that the dynamic approach, if implemented correctly, should increase revenue to the web site. However, the dynamic approach depends on our knowledge of the re-click probability p. A natural question arises: how critical is it to predict the value of p accurately? Clearly, if the success of the dynamic approach critically depends on an accurate estimation of p, then its use may be limited by this requirement.

The results of the experiments conducted in [6] show that the revenue changes only for those values of p that are significantly lower than the correct p value. Also, the revenue is not affected for any positive error values. In [6], the authors offer the following explanation: for values of p that are higher than the correct value, if an ad has been clicked upon in any slot, its click probability in all subsequent exposures becomes relatively higher. Consequently, an overestimation should not affect the scheduling decision. However, large negative error values (i.e., significantly underestimating the value of p) may reduce the chances of clicked ads being selected in subsequent slots and may impact the schedule.

In [6], the authors also observe that an error in p value affects the static and dynamic schedule in a similar fashion. Moreover, the dynamic revenue is always greater than the static revenue at any error level. Therefore, we conclude that it is always better to use the dynamic schedule even when we do not have a good estimate of the value of p. Also, it is better to overestimate rather than underestimate the value of p.

Although the look-ahead dynamic approach outperforms the static approach, one practical concern that is relevant is the time needed to execute the dynamic approach. Note that while the static approach computes the entire schedule in advance, the look-ahead dynamic approach must recompute an updated schedule after every slot. When a real-time solution is desired, the computational requirements of the look-ahead dynamic approach may be excessive. The myopic implementation of the dynamic approach solves very quickly and provides solutions that almost match the performance of the look-ahead dynamic approach [6]. The myopic implementation may have another advantage over the look-ahead version. We discuss this next. The look-ahead dynamic approach requires the knowledge of N, the planning horizon. In practice, however, there is no guarantee that a customer will stay for the entire duration of the horizon. Interestingly, the myopic version benefits when the customer departs earlier than planned [6].

To summarize, on the issue of static versus dynamic scheduling, it is recommended to use the dynamic approach. Within dynamic approaches, for practical reasons (real-time requirements, and uncertain planning horizons), the use of the myopic implementation is favored [6].

References

1. Caprara A, Kellerer H, Pferschy U (2000) The multiple subset sum problem. SIAM J Optim 11(2):308–319
2. Dahlen M (2001) Banner advertisements through a new lens. J Advert Res 41(4):23–30
3. Dudley B (2005) Microsoft touts ad-selling system as step ahead of its competitors. The Seattle Times, March 17
4. Gallagher K, Foster KD, Parsons J (2001) The medium is not the message: advertising effectiveness and content evaluation in print and on the web. J Advert Res 41(4):57–70
5. Garey MR, Johnson DS (1979) Computers and intractability: a guide to the theory of *NP*-completeness. Freeman, San Francisco
6. Kumar S, Dawande M, Mookerjee VS (2007) Optimal scheduling and placement of Internet banner advertisements. IEEE Trans Knowl Data Eng 19(11):1571–1584
7. MacInnis DJ, Moorman C, Jaworski BJ (1991) Enhancing and measuring consumers' motivation, opportunity, and ability to process brand information from ads. J Mark 55(4):32–53
8. Murthi BPS, Sarkar S (2003) The role of the management sciences in research on personalization. Manag Sci 49(10):1344–1362
9. Tellis GJ (1988) Advertising exposure, loyalty, and brand purchase: a two-stage model of choice. J Mark Res 25:134–44

Chapter 5
Internet Advertising Firms

Abstract In this chapter, we analyze a problem faced by an Internet advertising firm (Chitika) that operates in the Boston area. Chitika contracts with publishers to place relevant ads over a specified period on publisher websites. Ad revenue accrues to the firm and the publisher only if a visitor clicks on an ad (i.e., we are considering the CPC model). This might imply that all visitors to the publisher's website be shown ads. However, this is not the case if the publisher imposes a click-through-rate constraint on the ad-firm. This performance constraint captures the publisher's desire to limit ad clutter on the website and holds the ad-firm responsible for the publisher's opportunity cost of showing an ad that does not result in a click. We present a predictive model of a visitor clicking on a given ad. Using this prediction of the probability of a click, we present a decision model that uses a threshold to decide whether or not to show an ad to the visitor. The decision model's objective is to maximize the advertising firm's revenue subject to a click-through-rate constraint. We present and contrast two competing solutions: (1) a *static* solution, and (2) a *rolling-horizon* solution that re-solves the problem at certain points in the planning horizon. The static solution is shown to be optimal when accurate information on the input parameters to the problem is known. However, when the parameters to the model can only be estimated with some error, the rolling-horizon solution can perform better than the static solution. When using the rolling-horizon solution, it becomes important to choose the appropriate re-solving frequency.

5.1 Introduction

Other than the visitor, there are three main entities involved in Internet advertising: (1) the advertiser (whose advertisement is displayed), (2) the publisher (that provides the real-estate where the advertisement is displayed), and, (3) the advertising firm (usually referred to as "ad-firm"). The ad-firm's business is one of monetizing the traffic for publishers (website owners, mobile application providers, etc.).

S. Kumar, *Optimization Issues in Web and Mobile Advertising*, SpringerBriefs in Operations Management, DOI 10.1007/978-3-319-18645-0_5

An ad-firm gathers data on visitor behavior from its large publisher network and uses this knowledge to target advertisements (hereafter also referred to as "ads") to visitors on a variety of websites. Such firms collect ads from advertisers (or ad aggregators) and display these ads to targeted visitors on different publisher sites.

As discussed earlier, despite the obvious attractiveness (to advertisers) of a performance-based payment scheme (such as a CPC model), at the end, it is the publisher that must bear the opportunity cost of an ad display that does not result in a click. To address this incentive misalignment, an emerging trend in the ad industry is to require the ad-firm to manage ad display at a publisher's site such that an efficiency (or a click-through-rate) constraint is respected. Typically, the publisher and the ad-firm enter into a contract. At the end of each month, the ad-firm pays a publisher a (contractually agreed upon) fraction of the revenue accrued from clicks that were generated by the ads displayed on the publisher's site. In addition, the publisher enforces a click-through-rate constraint on the ad-firm, which requires that the monthly average click-through-rate (over the contract period that could be several months) is above a specified value. Such a constraint ensures that the publisher's space for ads on its website is used efficiently. An impression (an ad placed) that did not result in a click is an indication of a wasted opportunity for the publisher. The publisher could have either placed more relevant ads, or used the space for additional content. The click-through-rate constraint essentially balances two opposing goals of the publisher: (1) generate as much revenue as possible from ads, and (2) keep the website content interesting so that visitors continue to patronize the website. If a publisher becomes too greedy and shows an excessive number of ads, this could come at the expense of the content; the main driver of traffic to the website. In this case, the publisher would likely suffer in the long run. On the other hand, if a visitor clicks on an ad, it often indicates that she likes it. In such cases, the ad could be considered to be almost indistinguishable from content.

Earlier chapters have focused on solving the problem of choosing the best ad to a given visitor or a given slot. The implicit assumption in most extant studies on campaign management is that ads are shown to every arriving visitor. Given the emphasis on efficiency (namely, the click-through-rate constraint), this chapter focuses on deciding when *not* to show an ad to a visitor. This filtering of ads must occur in a manner to balance the opposing goals of revenue and efficiency. If too many ads are filtered, the efficiency constraint would be met (or even exceeded), but opportunities to generate clicks would likely be lost. On the other hand, if not enough filtering is done, more clicks would likely be generated at the cost of not meeting the target efficiency constraint.

5.2 Overview of Solution

The ad-firm usually manages multiple publishers. However, the problem for each publisher can be considered a separate problem as long as there is no constraint from the advertisers that links two publishers. In the problem setting considered in

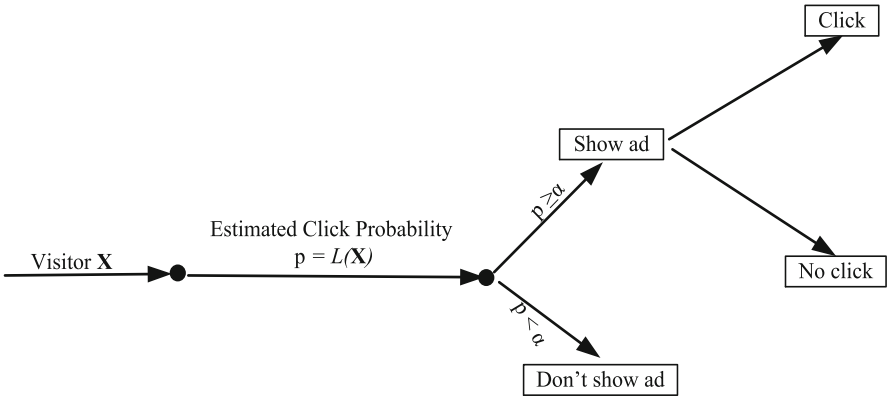

Fig. 5.1 Overall solution process

this paper, the advertisers do not impose any such constraint. Therefore, we analyze the problem for each publisher separately.

Essentially, any solution to the user profiling problem can only address the problem by some variant of an ad *filter*: using this filter, only some users are selected to be shown ads. In past studies, this filter is also referred to as *targeting* [1, 2]. The actionable information associated with a user includes the available details of the *impression*. Based on these details, a decision needs to be made to show or not to show an ad. One convenient way to use the impression details is to map them to a *click-probability*. That is, given these details, what is the probability of a click for the ad associated with the impression? Next, for this click-probability, should we show the ad? The above discussion suggests the use of a *threshold policy*. This policy can be stated as follows. *If the click-probability of an impression is greater than a threshold, then show the ad, else do not show it.* We later show that such a threshold policy is optimal. Figure 5.1 illustrates the overall solution process. The figure depicts that the solution to the user profiling problem consists of two parts: (1) a step involving data analytics (to predict the click-probability, p), and (2) a follow-up step involving decision analytics (to choose the threshold, α).

With regard to the problem of choosing the correct threshold, the natural question arises: should the threshold be held constant over the planning horizon, or should the value of the threshold be varied depending on the current *state* of the problem, i.e., the impressions and clicks that have been observed so far and the time left in the planning horizon (typically, this horizon is 1 month)? Put differently, should the display criterion (the threshold) be relaxed or tightened, depending on how much time is left in the planning horizon and whether the current click-through-rate is above or below the target level that has to be achieved by month end (on average). If we are ahead (i.e., the current click-through-rate is above the target value), then should the threshold be lowered (so more ads can be shown) to potentially earn more ad revenue? On the other hand, if we are behind (i.e., the current click-through-rate is below the target value), then should the threshold be increased (so that ads are only

shown to more interested visitors), in an attempt to meet the click-through-rate constraint? Increasing the threshold sacrifices ad revenue, but we may need to take this action to achieve the target click-through-rate at the end of the planning horizon.

5.3 Model and Solution

5.3.1 Description of the Data Analytic Model

Here we describe how the probability of a click and the distribution of these probabilities in the visitor population for a publisher is estimated.

5.3.1.1 Predicting the Probability of a Click

The prediction method uses a vector of observations collected from the visitor's cookie as well as meta data available from the `http` header [3]. These observations include variables such as the visitor's search string, Internet browser, operating system, previous click data, and so on. One of the major strengths of an ad-firm lies in its ability to develop a large publisher network. This enables the firm to have repeated interactions with visitors across different websites, thus enabling the firm to develop a *profile* of each visitor. The Logit model uses this profile information together with other information about the publisher's website to estimate the chances that a visitor will click on a given ad. This model can be expressed as a function $p = L(\mathbf{X})$, where p is the estimated click-probability, and $\mathbf{X}$ is the vector of variables used for the prediction. The model uses over 50 different variables; some of the important ones are given in [4, 5].

5.3.1.2 The Click-Probability Distribution

Using the above Logit model, the value of p for any given visitor to the publisher's site can be estimated. We can estimate the click-probability distribution for a given publisher using a sample of these probabilities. The click-probability distribution provides us with vital information. As we show later, the optimal policy requires us to show an ad to a visitor only if the click-probability (p) meets or exceeds a given threshold value (say, α). That is, show the ad only if $p \geq \alpha$, where $0 \leq \alpha \leq 1$. For any given threshold, the click-probability distribution allows us to estimate the probability that an ad will be shown to a randomly selected visitor and the conditional probability that a visitor will click on an ad given that the click-probability exceeds the threshold. Conceptually, the probability that an ad will be shown is the upper tail of the distribution (i.e., the area under the density curve above α), and the click-probability that a visitor will click on an ad is the conditional expectation of the upper tail of the distribution.

Let $f(p)$ denote the probability density of p and $\beta(\alpha)$ denote the probability that an ad is shown to a visitor for a given threshold α. Then, $\beta(\alpha)$ can be calculated as

$$\beta(\alpha) = \int_{\alpha}^{1} f(p)dp. \tag{5.1}$$

It is clear from the expression for $\beta(\alpha)$ that the probability of showing an impression decreases with the threshold, α. Let $\delta(\alpha)$ denote the conditional probability that a visitor will click on an ad, given that the click-probability exceeds the threshold α. This probability is given by

$$\delta(\alpha) = \frac{\int_{\alpha}^{1} pf(p)dp}{\beta(\alpha)}. \tag{5.2}$$

Now, in the following proposition, we present an important property of $\delta(\alpha)$. Proofs for all the propositions in this chapter are available in [5].

Proposition 5.1. *The conditional probability of a click (i.e., $\delta(\alpha)$) increases with α, i.e., $\frac{d\delta(\alpha)}{d\alpha} \geq 0$ [5].*

The above result highlights the trade-off between the revenue and the click-through-rate. As α increases, the probability of showing an impression (i.e., $\beta(\alpha)$) decreases. Therefore, the expected number of clicks (hence, the revenue of the ad-firm) also decreases with α. However, as shown in Proposition 5.1, the conditional probability of a click increases with α. Hence, as α increases, the click-through-rate increases.

5.3.2 Description of the Decision Analytic Model

Table 5.1 summarizes the main notation in the decision analytic model. We divide the planning horizon into K periods where each period corresponds to an arrival of a visitor to the publisher's website. For each arrival, we need to decide whether or not to show an ad. As discussed earlier, the objective is to maximize the expected number of clicks subject to the following click-through-rate constraint:

$$\mathbb{E}\left[\frac{\tilde{r}}{\tilde{m}}\right] \geq \eta. \tag{5.3}$$

This constraint can be difficult to calculate. However, it is possible to approximate the constraint very accurately using the ratio of expectations, rather than finding the expectation of the ratio. We present this result below.

Proposition 5.2. *As the number of arrivals (i.e., K) approaches ∞, $\mathbb{E}\left[\frac{\tilde{r}}{\tilde{m}}\right]$ approaches $\frac{\mathbb{E}[\tilde{r}]}{\mathbb{E}[\tilde{m}]}$ [5].*

Symbol	Definition
K	Number of periods in the planning horizon
$\tilde{m}$	Random variable for the number of impressions in the planning horizon
$\tilde{r}$	Random variable for the number of clicks in the planning horizon
p	Click-probability for a visitor
$f(p)$	Density function for the click-probability
α	Click-probability Threshold **(Decision Variable)**
$\beta(\alpha)$	Probability of impression for threshold α
$\delta(\alpha)$	Conditional probability of click for threshold α given that the ad is shown
η	Publisher's click-through-rate constraint

Table 5.1 Model parameters and variables

We therefore consider the revised optimization problem with the left-side of Constraint (5.3) suitably replaced. The revised constraint can be calculated as follows. Let $I[x,i,j,l]$ denote the $1-0$ decision variable to show or not an ad to the i^{th} arrival with characteristics $\mathbf{X} = x$, where j and l represent the number of impressions and the number of clicks, respectively, before the i^{th} arrival, $i = 1,2,\ldots,K$. Further, $\mathbb{P}[x]$ represents the click-probability for a visitor with characteristics x. Then, we have

$$\mathbb{E}[\tilde{r}] = \mathbb{E}\left[\sum_{i=1}^{K} I[x,i,j,l]\mathbb{P}[x]\right], \text{ and}$$
$$\mathbb{E}[\tilde{m}] = \mathbb{E}\left[\sum_{i=1}^{K} I[x,i,j,l]\right].$$

The optimization problem can now be formally expressed as

Problem P:

$$\max_{I(.)} \mathbb{E}\left[\sum_{i=1}^{K} I[x,i,j,l]\mathbb{P}[x]\right],$$

s.t.

$$\mathbb{E}\left[\sum_{i=1}^{K} I[x,i,j,l]\mathbb{P}[x]\right] - \eta\mathbb{E}\left[\sum_{i=1}^{K} I[x,i,j,l]\right] \geq 0.$$

The Lagrange Dual Function for this problem can be written as

$$L(\psi) = \max_{I(.)} \left\{ \mathbb{E}\left[\sum_{i=1}^{K} I[x,i,j,l]\mathbb{P}[x]\right] + \psi\left(\mathbb{E}\left[\sum_{i=1}^{K} I[x,i,j,l]\mathbb{P}[x]\right] - \eta\mathbb{E}\left[\sum_{i=1}^{K} I[x,i,j,l]\right]\right)\right\}, \tag{5.4}$$

where ψ is the Lagrange multiplier. This dual function can be rewritten as

$$\max_{I(.)} \mathbb{E}\left[\sum_{i=1}^{K} I[x,i,j,l](\mathbb{P}[x] + \psi(\mathbb{P}[x] - \eta))\right].$$

Thus, for each arrival, an ad should be displayed iff

$$\mathbb{P}[x] \geq \frac{\psi\eta}{1+\psi}.$$

Hence, the optimal policy is a threshold policy (i.e., $\alpha = \frac{\psi\eta}{1+\psi}$), and it is static in the sense that the decision to display an ad does not depend on the state $< i, j, l >$.

An upper bound for Problem P is

$$\min_{\psi \geq 0} L(\psi).$$

When $\psi = 0$, the first term of the dual function in Eq. (5.4) is maximum and the second term is 0. If the constraint in Problem P is satisfied at $\psi = 0$, then it is optimal to set $\psi = 0$. This is because any increase in ψ will make it strictly positive and the constraint will also be strictly greater than zero, thus violating complementary slackness. If the constraint is not satisfied at $\psi = 0$, then we should increase the value of ψ to a value (say $\bar{\psi}$) where the constraint is just satisfied at equality. Any further increase in ψ will violate complementary slackness. Thus, the optimal solution is either $\psi = 0$ (if the constraint is satisfied at this value of ψ) or the smallest value of ψ that satisfies the constraint at equality. The constraint at equality can be written as

$$\mathbb{E}\left[\sum_{i=1}^{K} I[x, i, j, l]\,(\mathbb{P}[x] - \eta)\right] = 0.$$

One possible solution to this equation is $I[x, i, j, l] = 0\ \forall i$. In this solution, the objective function value is clearly zero. Another possible solution is obtained by setting $\mathbb{P}[x] = \eta$, where $\mathbb{P}[x] = \delta\left(\frac{\psi\eta}{1+\psi}\right) = \delta(\alpha)$. In this solution, the objective function is greater than or equal to zero. Hence, this solution is optimal.

The constraint is satisfied at $\psi = 0$ only when $\delta(0) \geq \eta$. Clearly, $\delta(0) = \mathbb{E}[p]$. Based on the above discussion, we now present the optimal solution in the following proposition.

Proposition 5.3. *The optimal solution to* Problem P *is: [5]*

(a) $\alpha = 0$*; if* $\mathbb{E}[p] \geq \eta$.
(b) The smallest value of α *that satisfies* $\delta(\alpha) = \eta$*; otherwise.*

Observe that the above proposition states that the optimal policy is *static*. That is, there is no need to use different threshold values for different states.

5.4 Impact of Inaccurate Problem Parameters

We now examine the impact of inaccurate problem parameters on the solution presented in Proposition 5.3. With inaccurate parameters, let us assume that the true functions β and δ are wrongly estimated as $\hat{\beta}$ and $\hat{\delta}$ respectively. To address

inaccurate parameters, we propose a *rolling-horizon* approach. In a rolling-horizon approach, Problem P (presented in Sect. 5.3.2) is re-solved at certain points during the planning horizon. Formally, after s arrivals (or $(K-s)$ remaining arrivals), $s \in (1,2,\ldots,K-1)$, given that the impressions and the clicks that have occurred are m_s and r_s, respectively, the following optimization problem is solved

$$\max_{I(.)} \mathbb{E}\left[\sum_{i=s+1}^{K} I[x,i,j,l]\mathbb{P}[x]\right],$$

s.t.

$$\mathbb{E}\left[\sum_{i=s+1}^{K} I[x,i,j,l]\mathbb{P}[x]\right] - \eta\,\mathbb{E}\left[\sum_{i=s+1}^{K} I[x,i,j,l]\right] \geq m_s\eta - r_s.$$

Then, as shown earlier, the threshold that is used for the $(s+1)^{th}$ arrival is given by the solution of

$$\frac{r_s + (K-s)\hat{\beta}(\alpha)\hat{\delta}(\alpha)}{m_s + (K-s)\hat{\beta}(\alpha)} = \eta.$$

The rolling-horizon approach can be expected to cope well in presence of inaccurate model parameters because it uses actual state information to update the threshold in each period. We explain the basic idea behind varying the thresholds as follows. Imagine that we are at the beginning of the planning horizon, i.e., the first period of the planning horizon. We set the threshold at the smallest level such that the target click-through-rate level should just be achieved at the end of the month (on average). We keep this threshold value constant for the current period. At the end of the period, we collect data on the number of impressions and the number of clicks. The current click-through-rate value is the number of clicks divided by the number of impressions. If this value is better than the required click-through-rate at the end of the month, we can afford to lower the threshold and show more ads. Conversely, if the current click-through-rate value is below the threshold, we must set the next period's threshold higher. At the beginning of each period, we set the threshold to a value that, if kept constant for the rest of the planning horizon, would just achieve, on an expected basis, the revised target threshold for the rest of the planning horizon.

In Fig. 5.2, we consider a simple example to illustrate how the rolling-horizon approach works. Imagine that there are 3 days in the planning horizon and that the value of the threshold (α) is updated on a daily basis. Let the click-through-rate constraint be 0.01. At the beginning of the first day, we find the lowest value of α that (if held constant) would just achieve the required click-through-rate. Suppose this value is α_1. We set the threshold to α_1 for the first day. At the end of the first day, we observe the number of clicks (say r_1) and the number of impressions (say m_1). We next use these values (r_1 and m_1) to find the lowest value of α that, if held constant for the remaining 2 days, would just achieve the required click-through-rate of 0.01. Suppose this value is α_2. Intuitively, this value should be higher than α_1 if the click-through-rate in the first day was below the required target (0.01), otherwise it should

	Day 1	Day 2	Day 3
Impressions	m_0	m_1	m_2
Clicks	r_0	r_1	r_2
Decision Variables	α_1	α_2	α_3

Fig. 5.2 The mechanics of the rolling-horizon approach

be lower. After considering the fact that we are behind (ahead) on the constraint, we set the value of α to α_2 for the second day. Similarly, at the beginning of the third day, we use the actual clicks and impressions that have occurred so far to calculate the lowest value of α needed to achieve the click-through-rate constraint (say, α_3). For this problem, the static approach would have used a constant threshold value (α_1) for the entire duration, that is, the threshold value used by the rolling-horizon approach in the first period.

The rolling-horizon approach has the advantage that it can increase or reduce the threshold to incorporate state information. In any particular state, we may be better or worse than what we need to be with respect to the target click-through-rate constraint. If we are doing better, we can afford to use a lower value of α. Conversely, if real-world feedback is such that we are behind on the constraint, we need to tighten (increase) the threshold to catch up.

5.5 Experimental Results

The primary goal of the numerical experiments in this section is to provide answers to two questions that are of practical interest: (1) What update frequency should be used in the rolling-horizon approach? (2) In the presence of inaccurate problem parameters (or *noise*), which method (static or rolling-horizon) should be used? The first question essentially boils down to asking: within practical constraints, should the most frequent update frequency be used? The danger of updating too frequently (more importantly, if updating is based on the outcomes of a small number of arrivals) is that the threshold could be changed based upon a unlikely random draw (of impressions and clicks). Thus, in an attempt to cope with noise, an excessively frequent update policy could lead to poor decisions, and hurt overall performance. The second question may be especially relevant when we expect there to be a moderate or small amount of noise. In such situations, there is a tension between the benefits of using an optimal approach (i.e., static) versus the virtues of using a sub-optimal one that is able to cope with noise.

We describe the findings of experiments corresponding to the two questions discussed above: how to choose update frequency, and which method to use when there is noise. To study noise, we consider the use of an inaccurate shape parameter corresponding to the click probability distribution. The factors varied are the

click-through-rate constraint (η), the wrong shape parameter (k), and the update frequency (μ). The correct shape parameter is denoted by k' and the scale parameter is denoted by q. Therefore, the expected value of the click probability p is $k'q$, i.e., $\mathbb{E}[p] = k'q$. Before discussing the details of experiments, we present the following analytical result regarding the performance of the static approach in presence of wrong shape parameter.

Proposition 5.4. *In presence of the wrong shape parameter, the performance of the static approach is described as follows: [5]*

(a) The static approach provides an optimal solution when either (i) $\eta \leq \min [k'q, kq]$*, or (ii)* $k = k'$.
(b) The static approach provides a sub-optimal solution when either (i) $kq < \eta \leq k'q$*, or (ii)* $kq < k'q \leq \eta$.
(c) The static approach provides an infeasible solution when either (i) $k'q < \eta \leq kq$*, or (ii)* $k'q < kq \leq \eta$.

In [5], the authors examine all the conditions presented in this proposition experimentally. Clearly, the rolling-horizon approach may perform better than the static approach for the conditions presented in parts (b) and (c) of the proposition.

5.5.1 Choice of Update Frequency

For some publishers that have relatively low traffic, the rolling-horizon could suffer from having to update the threshold after a relatively small sample of arrivals (impressions, clicks). These updates may not be as reliable as the ones that were made with a large sample of impressions and clicks. Because of the inherent instability of inferences drawn from small samples, the rolling-horizon approach may sometimes be led astray and its performance could suffer vis-a-vis the static approach that does not rely on actual events to choose the threshold. It is here that we could expect to observe the malicious side of frequent updating. These speculations are borne out in our experiments that are discussed below.

Let us consider a relatively extreme case of a publisher with approximately 10 arrivals per period ($= \lambda$). We find that, when $\eta = 0.005$ or 0.0075, both the static and the rolling-horizon methods show ads to all the visitors. Hence, at these levels of η, there is no benefit of using the rolling-horizon approach. For all other values of η, the static approach performs better than the rolling-horizon approach. Further, the solution of rolling-horizon approach does not even meet the click-through-rate constraint as the update frequency increases. Thus, to summarize, the rolling-horizon approach indeed suffers when the update decisions are based on relatively small samples.

To avoid this problem with the rolling-horizon approach, we use an update period that ensures that the number of arrivals is large enough such that the update decision is not based on an unlikely draw of random events. Let the number of arrivals in one period be λ. We need to find a threshold ($\hat{\lambda}$) such that it is safe to update every

period if $\lambda > \hat{\lambda}$. Further, let α be the threshold at the beginning of a given period. The update decision at the end of the period depends on the number of impressions and clicks in that period, as well as the total clicks and impressions observed until the beginning of that period.

Recall that the new value of the threshold at the end of a period is calculated using

$$\frac{R + \tilde{r} + (K-s)\hat{\beta}(\alpha)\hat{\delta}(\alpha)}{M + \tilde{m} + (K-s)\hat{\beta}(\alpha)} = \eta,$$

where $\tilde{r}$ and $\tilde{m}$ are the number of clicks and impressions observed in the current period and R and M are the total clicks and impressions observed until the beginning of the period. The randomness in the quantities $\tilde{r}$ and $\tilde{m}$ is what affects the quality of the update decision. If these quantities are close to their mean values, then the update decision will likely be of good quality. However, if there is large variation in these quantities, then the update decision can be poor and would likely hurt the performance of the rolling-horizon approach.

From the above equation, it is clear that the distribution of the quantity $(\eta\tilde{m} - \tilde{r})$ is of key interest. The mean number of impressions is $\mathbb{E}[\tilde{m}] = \lambda\beta(\alpha)$ and the mean number of clicks is $\mathbb{E}[\tilde{r}] = \lambda\beta(\alpha)\delta(\alpha)$. Using Wald's theorem for variance, it can be shown that the variance of the number of impressions is $\mathbb{V}[\tilde{m}] = \lambda\beta(\alpha)(1-\beta(\alpha))$ and the variance of the number of clicks is $\mathbb{V}[\tilde{r}] = \lambda\beta(\alpha)\delta(\alpha)(1-\beta(\alpha)\delta(\alpha))$, where the functions β and δ are defined in Eqs. (5.1) and (5.2). The variance of the quantity $(\eta\tilde{m} - \tilde{r})$ is given by $\eta^2\mathbb{V}[\tilde{m}] + \mathbb{V}[\tilde{r}] - 2\mathbb{COV}[\eta\tilde{m}, \tilde{r}]$. Further, we can approximate $(\eta\tilde{m} - \tilde{r})$ to be Normally distributed with mean $\mu = \eta\mathbb{E}[\tilde{m}] - \mathbb{E}[\tilde{r}]$ and variance $\sigma^2 = \eta^2\mathbb{V}[\tilde{m}] + \mathbb{V}[\tilde{r}] - 2\mathbb{COV}[\eta\tilde{m}, \tilde{r}]$.

We have, $\mathbb{COV}[\tilde{m}, \tilde{r}] = \mathbb{E}[\tilde{m}\tilde{r}] - \mathbb{E}[\tilde{m}]\mathbb{E}[\tilde{r}]$. Also,

$$\begin{aligned}
\mathbb{E}[\tilde{m}\tilde{r}] &= \sum_{m=0}^{\lambda}\sum_{r=0}^{m} mr\mathbb{P}[r|m]\mathbb{P}[m] \\
&= \sum_{m=0}^{\lambda} m\mathbb{P}[m]\sum_{r=0}^{m} r\mathbb{P}[r|m] \\
&= \sum_{m=0}^{\lambda} m\mathbb{P}[m]\sum_{r=0}^{m} r\frac{m!}{(m-r)!(r!)}\delta(\alpha)^r(1-\delta(\alpha))^{m-r} \\
&= \sum_{m=0}^{\lambda} m\mathbb{P}[m]\cdot m\delta(\alpha) \\
&= \delta(\alpha)\sum_{m=0}^{\lambda} m^2\mathbb{P}[m] \\
&= \delta(\alpha)(\lambda\beta(\alpha)(1-\beta(\alpha)) + (\lambda\beta(\alpha))^2).
\end{aligned}$$

Thus, $\mathbb{COV}[\tilde{m}, \tilde{r}] = \lambda\beta(\alpha)\delta(\alpha)(1-\beta(\alpha))$.

Using the above expression, the variance of the quantity $(\eta\tilde{m} - \tilde{r})$ is

$$\sigma^2 = \lambda\beta(\alpha)\left(\eta^2(1-\beta(\alpha)) - 2\eta\delta(\alpha)(1-\beta(\alpha)) + \delta(\alpha)(1-\beta(\alpha)\delta(\alpha))\right).$$

Hence, in order to ensure that any random draw of the quantity $\eta\tilde{m} - \tilde{r}$ is relatively *stable*, we can derive a minimum sample size that ensures that the quantity stays within an error of χ around its mean with a probability of $100(1-\phi)$ percent. Using standard procedures for constructing confidence intervals, the lower bound on the number of arrivals in each period (i.e., $\hat{\lambda}$) can be calculated as follows:

$$\hat{\lambda} = \frac{Z^2_{1-\phi/2}\left[\eta^2(1-\beta(\alpha)) + \delta(\alpha)(1-\beta(\alpha)\delta(\alpha)) - 2\eta\delta(\alpha)(1-\beta(\alpha))\right]}{\beta(\alpha)\chi^2(\eta-\delta(\alpha))^2}.$$

We use the above expression for the minimum number of arrivals before which an update can be safely performed. When $\delta(\alpha) = \eta$, the above expression is not valid. This condition usually satisfies only in the first period. After the first period, because of randomness, $\delta(\alpha)$ is usually not equal to η. When $\delta(\alpha) = \eta$, since the mean of the random quantity is zero, we use the notion of an absolute error (e) and calculate the threshold using

$$\frac{1}{\eta^2(1-\beta(\alpha)) + \delta(\alpha)(1-\beta(\alpha)\delta(\alpha)) - 2\eta\delta(\alpha)(1-\beta(\alpha))}\left(\frac{e^2}{\beta(\alpha)Z^2_{1-\phi/2}}\right).$$

Below the threshold, the update could be based on an unlikely draw of events and hurt performance. For example, when $\eta = 0.0175, \alpha = 0, \chi = \phi = 0.05, k' = 2.25$, and $q = 0.005$, $\hat{\lambda} = \frac{(1.96)^2\times 2.25\times 0.005[1-(2.25\times 0.005)]}{(0.05)^2[0.0175-(2.25\times 0.005)]^2} = 437{,}573$, or 0.5 million (approximately). Thus, for a publisher with the above characteristics, and a traffic of about 100,000 hits per day, a daily update policy is not appropriate. That is, we need to wait for about 5 days to get a reasonable sample size to perform an update. To provide a sense for the above traffic volumes, we observe that at Chitika, *enterprise* level publishers have a traffic in excess of 1 million hits per day. Many publishers, however, are in the range of 50,000–100,000 hits per day. There is also a long tail; a significant number of publishers have a daily traffic of about 5000 hits per day.

We will refer to a publisher as a *high-volume* publisher if the total number of arrivals per month exceeds the update threshold for that publisher ($\hat{\lambda}$). Note that we need to calculate the threshold for all possible values of $\alpha \in [0,1]$ and use the maximum threshold value across these values of α. If the monthly traffic is not sufficient to warrant at least one update, we will refer to such a publisher as a *low-volume* one.

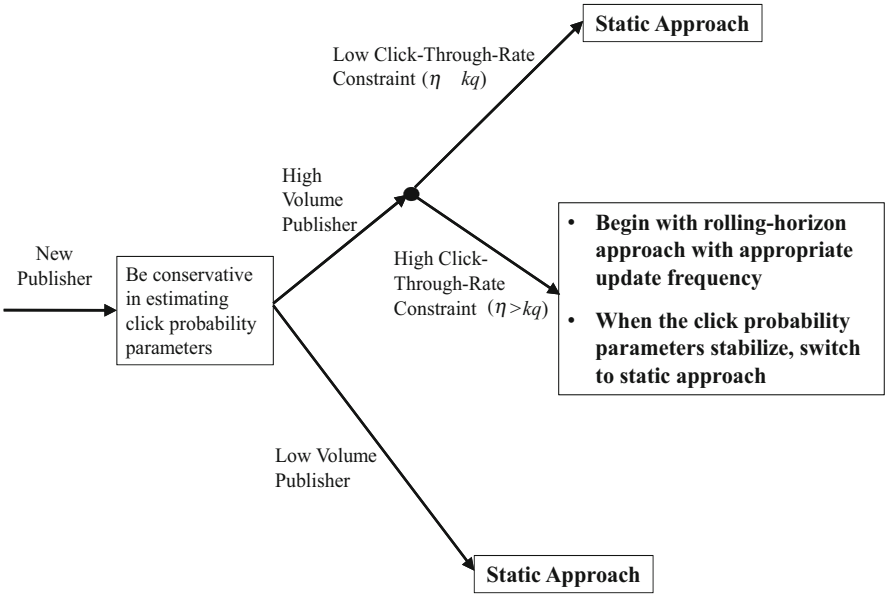

Fig. 5.3 Final recommendations

5.5.2 Final Recommendations

Based on the experimental results in [5], the final recommendations are as follows. For low-volume publishers, the static approach is recommended to avoid the danger of updating based on small samples. Also, when η is low, the static approach and the rolling-horizon approach have similar performance for any level of error. Here, it may be better to use the static approach to avoid the overhead of implementing an approach that attempts to update the threshold, but ends up not needing to since the value of η is low. For high-volume publishers and high levels of η, the rolling-horizon approach outperforms the static approach for most error levels. In the case of overestimation errors, neither approach meets the click-through-rate constraint but the rolling-horizon approach comes closer to meeting this constraint. Since overestimation errors often lead to infeasible solutions, it is better to be conservative while estimating the parameters of the click-probability distribution for a new publisher (Fig. 5.3).

References

1. Gerken DA (2008) System and method for selectively acquiring and targeting online advertising based on user IP address. United States Patent No: US 7,376,714 B1
2. Goldfarb A, Tucker C (2011) Online display advertising: targeting and obtrusiveness. Mark Sci 30(3):389–404
3. Kolluri V, Dorosario A, Mookerjee V, Mookerjee R, Caswell C (2013) Method and system for determining user likelihood to select an advertisement prior to display. United States Patent No: US 2013/0124344 A1
4. Mookerjee R, Kumar S, Mookerjee VS (2012) To show or not show: using user profiling to manage Internet advertisement campaigns at Chitika. Interfaces 42(5):449–464
5. Mookerjee R, Kumar S, Mookerjee VS (2015) Optimizing performance based Internet advertisement campaigns. Working paper

Chapter 6
Mobile Advertising

Abstract In this chapter, we present the problem of the mobile business unit of a web advertising firm, Chitika. In mobile advertising, the delivery of ads is achieved with the joint efforts of three distinct parties: the App aggregator (supply-side), the Ad aggregator (demand-side), and the Ad network (an intermediary between the supply and demand sides). We study a problem of optimal ad delivery from the perspective of the Ad network. An App aggregator generates a request that must be assigned to a particular Ad aggregator that provides an Ad for the request. Because of the presence of supply-side and demand-side constraints, a constrained allocation problem must be solved to maximize the profit earned by the Ad network. This problem is formulated and solved using a standard solver (CPLEX). The solution is examined to provide specific guidance to the Ad network to manage its supply and demand side contracts to increase profit.

6.1 Introduction

The context of this chapter is mobile advertising. In this domain, the most popular vehicle for displaying ads is a mobile application (or *app* for short). When a mobile phone user clicks on an app (for example, to display the current weather), an ad is inserted in the response (often at the bottom of the screen of the phone device). An interesting feature of the mobile ad industry is the presence of aggregators in both the supply and demand sides of the market. Supply side aggregators (or app aggregators) integrate with hundreds of app owners (analogous to publishers). Examples of app aggregators are Jiwire, Tweetcaster, etc. These app aggregators connect a variety of apps to an ad network for obtaining ads. On the demand-side, the ad network (e.g., Chitika, Rocketfuel, etc.) calls an ad aggregator for supplying ads. Examples of ad aggregators are Super Pages, Ad Marvel, etc.

S. Kumar, *Optimization Issues in Web and Mobile Advertising*, SpringerBriefs in Operations Management, DOI 10.1007/978-3-319-18645-0_6

To provide a concrete example, consider a mobile I-phone user who taps on the *Yelp* app on her phone device. Let the app aggregator for Yelp be Click-to-Call. Suppose that Click-to-Call works with the ad network Chitika. Upon receiving the ad request from Click-to-Call, Chitika forwards the request to Super Pages, an ad aggregator that works with a large number of merchants (who are often service providers, such as dentists, plumbers, restaurants, etc.). Chitika's call to Super Pages returns an ad for a restaurant that is in the vicinity of end user of the phone device. If the user taps on the ad, it could, for example, connect a call to the restaurant, presumably for making a reservation.

In this chapter, we consider a demand-supply optimization problem faced by an ad network, such as Chitika. Chitika has different contracts with supply-side partners that allow it access to the supply side of the market. For example, in a CPM contract, Chitika pays the app aggregator for each ad request that it accepts from the supply partner. On the other hand, in a revenue sharing contract, Chitika shares a part of the revenue it earns from the ad aggregator when there is a qualified action (e.g., a click, a call, etc.) on the ad. On the demand-side, Chitika earns some revenue from the ad aggregator if there is a qualified action on an ad. Upon receiving a request from an app aggregator, Chitika must choose a particular ad aggregator to call for an ad. If this call fails (i.e., the ad aggregator fails to provide an ad), Chitika can contact another ad aggregator for an ad and so on. However, there is a time window for which an ad can be obtained: in practice, no more than about 3 calls can be made for obtaining an ad. Beyond 3 calls, the ad display would be delayed beyond an acceptable amount. In such cases (no ad is obtained by Chitika), the opportunity to display an ad is lost. This process is illustrated in Fig. 6.1.

While attempting to maximize profit, Chitika must respect certain supply and demand side constraints. An important supply side constraint that Chitika must respect is a *coverage* constraint. Coverage is defined as the proportion of ad requests

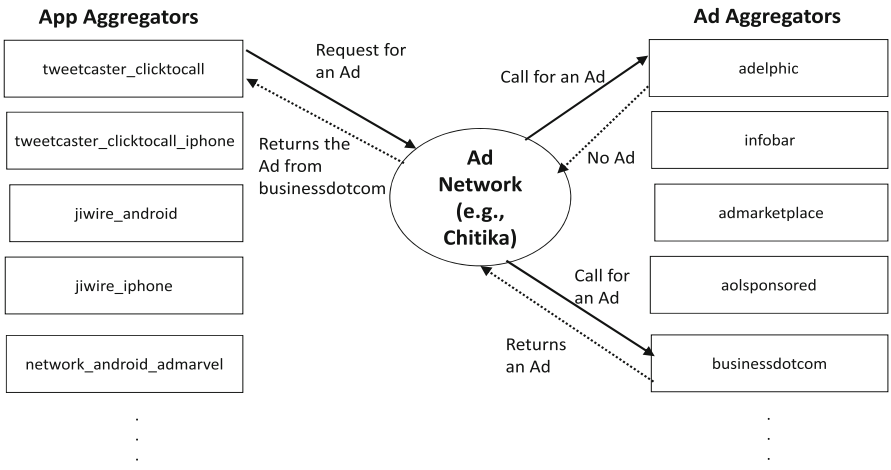

Fig. 6.1 Illustration of the process

from a supplier that are satisfied by an ad. Suppliers do not like it if the coverage provided by Chitika falls below a specified amount, and are likely to withdraw their supply if Chitika fails to deliver at or above a specified coverage threshold. On the demand side, ad aggregators place a limit on the number of calls (usually totaled for a day) made to them for ads. Because ad aggregators often work with multiple ad networks, this constraint reflects an ad aggregator's way of throttling the share of demand that Chitika can access from them.

The core problem that Chitika must solve is a *matching* problem. When it receives a request from a supply partner, it must judiciously choose an appropriate demand partner to match the request. The demand partner should be chosen based on a variety of factors. One consideration is the probability that the ad aggregator will return an ad corresponding to the request. Another consideration is the revenue that Chitika will earn if there is a click. Also, the probability that there is a click depends on the demand-supply pair. The above mentioned considerations must be balanced in the presence of the call limits imposed on Chitika by demand side partners.

The demand-supply matching problem is formulated as a Mixed Integer Program and solved using a standard solver (CPLEX) [3]. While the solution to the optimization problem is one of the goals of Mookerjee et al. [3], the solution also provides some guidance on solving a higher-level problem of managing contracts with Chitika's demand and supply partners. One example is as follows. For certain supply partners, Chitika pays an amount that is the higher of a revenue share and a CPM contract. To illustrate this, consider that the cost of an impression from a particular supply partner is $\$m$ per 1000 impressions. Further, let us assume a revenue share of 60 % with this partner. Then, the amount Chitika pays the supply partner is the higher of $\$mn$ and $\$0.6rc$, where n is the number of impressions (in thousands), r is the number of clicks, and c is the average revenue earned by Chitika per click. Using the solution of the optimization problem, it is possible to recommend to Chitika to negotiate a better contract with a particular supply or demand partner. In the example above, we could find out that Chitika would earn a higher profit if it (for example) offers a higher revenue share (70 % instead of 60 %) but replaces the hybrid contract with a pure revenue sharing contract.

There are commercial interests that drive both the supply and demand sides of the mobile advertising market. For an ad network such as Chitika, the problem in this study can be considered as a generalization of the multiple, fractional bin packing problem, which is an NP-hard problem [2]. The total number of requests made by an app aggregator needs to be allocated across several ad aggregators. Although a particular request can result in multiple calls (typically, not more than 3), the number of ads returned cannot be greater than the number of requests. Drawing an analogy with multiple fractional bin packing, this aspect of the problem corresponds to the possibility an item being fractionally distributed across multiple bins. However, in our problem, a request may not result in an ad, and there may be more calls than requests. Despite the complex nature of the optimization problem, we can solve the instances of realistic sizes using solvers as CPLEX. However, we also discuss some special cases of the problem that can be solved either optimally using a polynomial

time algorithm or approximately using a good heuristic. Based on the results of this model, we also discuss how supply and demand side contracts can be better configured to improve Chitika's revenue.

6.2 Optimization Model

As discussed earlier, a mobile app is registered with an app aggregator. Similarly advertisers contract with the ad aggregators and make their ads available to them. When an end user clicks on an app on her mobile device, a request for an ad is sent by the app aggregator to Chitika. Chitika then forwards this request to the ad aggregator who (likely) responds with an ad. Chitika then sends the ad to the mobile device. The information included in ad request are: type of phone, time of day, day of week, zip code, app generating the request, etc. The possible reasons for an ad aggregator not responding with any ad are: contracted advertisers are out of budget, no appropriate ad is available, there is technical time-out, etc. The payment system works as follows: Chitika pays the app aggregator (i.e., the supplier) based on any one of the following three schemes: (a) by impression, (b) by click, or (c) by best (for the supplier) of the two. On the other hand, Chitika is paid by the ad aggregator (i.e., the demand side) using the cost-per-click model.

The constraint imposed by an app aggregator on Chitika (referred to as *supplier constraint*) is as follows: Each app aggregator has a coverage constraint such that

$$\frac{\text{Number of impressions}}{\text{Number of ad requests}} \geq Coverage.$$

On the other hand, the constraints imposed by an ad aggregator on Chitika (referred to as *demand side constraints*) are as follows: (a) There is an upper bound on the number of calls that an ad aggregator will accept per day. (b) There is an upper bound on the number of impressions that an ad aggregator can provide per day. (c) There is an upper bound on the number of clicks that an ad aggregator wants per day. (d) There is an upper bound on the amount paid by an ad aggregator per day.

The following information is available to Chitika: (a) For each demand partner, $\mathbb{P}$[supplying an ad to Chitika], (b) $\mathbb{P}$[impression] for each demand/supplier combination, (c) $\mathbb{P}$[click] for each demand/supplier combination, and (d) Earnings/Click (for Chitika) for each demand/supplier combination. Currently, Chitika calls its demand partners one-by-one based on some static rankings. Problem challenge arises from the fact that though the coverage constraint is provided for the whole day, demand side coverage as well as the request rate varies hourly.

6.2.1 Notations

We begin with presenting the notations used in the model.
Parameters:

T	Number of time slots (hours) in a planning horizon (a day).
n	Number of ad aggregators. $n \approx 15$.
m	Number of app aggregators requesting ads. $m \approx 20$.
R_{it}	Expected number of ad-requests from app aggregator i at time slot t.
p_{ijt}	Probability that an ad is delivered to app aggregator i if it is allocated to ad aggregator j at time slot t.
q_{ijt}	Probability that an ad is clicked by an user associated with app aggregator i when it is delivered by ad aggregator j at time slot t.
r_{ij}	Amount of revenue made by the company (Chitika) if the ad is clicked by an user associated with app aggregator i when it is delivered by ad aggregator j.
D_j	Upper bound on the number of requests (calls) made to ad aggregator j during the planning horizon.
I_j	Upper bound on the number of impressions provided (to the app aggregators) from ad aggregator j during the planning horizon.
C_j	Upper bound on the number of clicks made (by the users) for ads from ad aggregator j during the planning horizon.
P_j	Upper bound on the amount paid by ad aggregator j during the planning horizon.
s_i	Lower bound on the fraction of ad-requests from app aggregator i resulting into impression during the planning horizon.
Π	Profit of the company (Chitika).
A_1	A set of app aggregators paid (by Chitika) per impression.
A_2	A set of app aggregators paid (by Chitika) per click.
A_3	A set of app aggregators paid (by Chitika) for better (for the app aggregator) of the above two.
d_i	Payment per impression for app aggregator $i \in A_1 \cup A_3$.
c_i	Payment per click for app aggregator $i \in A_2 \cup A_3$.

Variables:

Z_{ijt}	Planned number of requests (calls) to ad aggregator j to fulfill requests from app aggregator i at time slot t.
X_{ijt}	Expected number of impressions to app aggregator i from ad aggregator j at time slot t, where $X_{ijt} = p_{ijt}Z_{ijt}$.
E_i	Total payment (from Chitika) to app aggregator $i \in A_3$ over the planning horizon.

6.2.2 Formulation

We now present the formulation for the optimization problem.

$$\text{MIP(0): Max } \Pi = \tag{6.1}$$
$$\sum_{t=1}^{T}\sum_{j=1}^{n}\sum_{i=1}^{m} r_{ij}q_{ijt}X_{ijt} - \sum_{t=1}^{T}\sum_{j=1}^{n}\sum_{i\in A_1} d_i X_{ijt} - \sum_{t=1}^{T}\sum_{j=1}^{n}\sum_{i\in A_2} c_i q_{ijt} X_{ijt} - \sum_{i\in A_3} E_i$$

$$\sum_{j=1}^{n} X_{ijt} \le R_{it},\ i = 1,2,\ldots,m;\ t = 1,2,\ldots,T \tag{6.2}$$

$$\sum_{t=1}^{T}\sum_{j=1}^{n} X_{ijt} \ge s_i \sum_{t=1}^{T} R_{it},\ i = 1,2,\ldots,m \tag{6.3}$$

$$X_{ijt} = p_{ijt}Z_{ijt},\ i = 1,2,\ldots,m;\quad j = 1,2,\ldots,n;\quad t = k,k+1,\ldots,T \tag{6.4}$$

$$\sum_{t=1}^{T}\sum_{i=1}^{m} Z_{ijt} \le D_j,\ j = 1,2,\ldots,n \tag{6.5}$$

$$\sum_{t=1}^{T}\sum_{i=1}^{m} p_{ijt}Z_{ijt} \le I_j,\ j = 1,2,\ldots,n \tag{6.6}$$

$$\sum_{t=1}^{T}\sum_{i=1}^{m} q_{ijt}p_{ijt}Z_{ijt} \le C_j,\ j = 1,2,\ldots,n \tag{6.7}$$

$$\sum_{t=1}^{T}\sum_{i=1}^{m} q_{ijt}p_{ijt}r_{ij}Z_{ijt} \le P_j,\ j = 1,2,\ldots,n \tag{6.8}$$

$$E_i \ge \sum_{t=1}^{T}\sum_{j=1}^{n} d_i X_{ijt},\ i \in A_3 \tag{6.9}$$

$$E_i \ge \sum_{t=1}^{T}\sum_{j=1}^{n} c_i q_{ijt} X_{ijt},\ i \in A_3 \tag{6.10}$$

$$X_{ijt} \ge 0, E_i \ge 0, Z_{ijt} \in \mathbb{Z}^+,\quad i = 1,2,\ldots,m;\ j = 1,2,\ldots,n;\ t = 1,2,\ldots,T \tag{6.11}$$

In this formulation, Constraint set (6.2) suggests that the total number of ad impressions to app aggregator i at time slot t should not be more than the specified value R_{it}. Next, the app aggregator i requires that a minimum faction, s_i $(0 < s_i < 1)$, of total number of requests from i results into impressions during the planning horizon. This is expressed in Constraint set (6.3). Constraint set (6.4) suggests that in order to obtain X_{ijt} number of ad impressions to app aggregator i from ad aggregator j at time slot t, Z_{ijt} number of ad requests (calls) needs to be sent to j from i. Constraint set (6.5) imposes an upper bound D_j on the number of requests made to ad aggregator j during the planning horizon. Constraint set (6.6) imposes an upper bound I_j on the number of impressions provided (to the app aggregators) from ad aggregator j during the planning horizon. Next, Constraint set (6.7) imposes an

upper bound C_j on the number of clicks made for ads provided by ad aggregator j during the planning horizon. Constraint set (6.8) ensures that the maximum amount paid by ad aggregator j during the planning horizon is P_j.

6.2.3 Special Case 1

We consider a special case where set A_3 is empty, and I_j, C_j, and P_j are very large numbers so that constraints (6.6)–(6.8) can be ignored. That is $I_j = C_j = P_j = \infty$ for all j. By replacing X_{ijt} by $p_{ijt}Z_{ijt}$, the special case can be formulated as follows.

$$\text{MIP: Max } \Pi =$$
$$\sum_{t=1}^{T}\sum_{j=1}^{n}\sum_{i=1}^{m} r_{ij}q_{ijt}p_{ijt}Z_{ijt} - \sum_{t=1}^{T}\sum_{j=1}^{n}\sum_{i\in A_1} d_i p_{ijt}Z_{ijt} - \sum_{t=1}^{T}\sum_{j=1}^{n}\sum_{i\in A_2} c_i q_{ijt}p_{ijt}Z_{ijt}$$
$$\sum_{j=1}^{n} p_{ijt}Z_{ijt} \le R_{it},\ i = 1,2,\ldots,m;\ t = 1,2,\ldots,T$$
$$\sum_{t=1}^{T}\sum_{j=1}^{n} p_{ijt}Z_{ijt} \ge s_i \sum_{t=1}^{T} R_{it},\ i = 1,2,\ldots,m$$
$$\sum_{t=1}^{T}\sum_{i=1}^{m} Z_{ijt} \le D_j,\ j = 1,2,\ldots,n$$
$$Z_{ijt} \ge 0, \quad i = 1,2,\ldots,m;\ j = 1,2,\ldots,n;\ t = 1,2,\ldots,T$$

This special case can be formulated as generalized min-cost flow problem with leakage (or losses) in the sense that if we send Z_{ijt} calls, then only $p_{ijt}Z_{ijt}$ will be realized [3]. If all $p_{ijt} = 1$, then the problem is a regular min-cost flow problem that can be solved efficiently by a polynomial time algorithm providing integer solution $Z_{ijt} \in \mathbb{Z}^+$ to the problem [1, 4]. The generalized min-cost flow problem with leakage can be solved efficiently as well if Z_{ijt} are allowed to be continuous variables [5]. Although the generalized min-cost flow problem with leakage is NP-hard, there are good approximate algorithms available for integer solution Z_{ijt} [6].

6.2.4 Special Case 2

We now consider another special case where set A_3 is empty, and D_j, I_j, C_j, and P_j are very large numbers so that constraints (6.5)–(6.8) can be ignored. That is $D_j = I_j = C_j = P_j = \infty$ for all j. Since all column constraints related to j are ignored, the problem can be decomposed into m subproblem for each i, and we can solve them independently for each $i = 1,2,\ldots,m$. For a given i, the special case can be formulated as follows.

$$\text{MIP: Max } \Pi_i = \sum_{t=1}^{T}\sum_{j=1}^{n} r_{ij}q_{ijt}p_{ijt}Z_{ijt} - \sum_{t=1}^{T}\sum_{j=1}^{n} d_i p_{ijt}Z_{ijt} - \sum_{t=1}^{T}\sum_{j=1}^{n} c_i q_{ijt}p_{ijt}Z_{ijt}$$

$$\sum_{j=1}^{n} p_{ijt}Z_{ijt} \le R_{it},\ t = 1,2,\ldots,T$$

$$\sum_{t=1}^{T}\sum_{j=1}^{n} p_{ijt}Z_{ijt} \ge s_i \sum_{t=1}^{T} R_{it}$$

$$Z_{ijt} \ge 0, \quad j = 1,2,\ldots,n; t = 1,2,\ldots,T$$

If $Z_{ijt} > 0$, then each unit of this call will generate profit w_{ijt}, where $w_{ijt} = (r_{ij}q_{ijt}p_{ijt} - d_i p_{ijt} - c_i q_{ijt}p_{ijt})$. Here, $c_i = 0$ for all $i \in A_1$, and $d_i = 0$ for all $i \in A_2$. This problem again can be formulated as the generalized min-cost flow problem with leakage [3]. However, the following special case can be solved by a polynomial time algorithm as described below.

For given i, if $s_i = 1$ and all $p_{ijt} = 1$, then this problem can be solved by the following greedy algorithm: for each time t, find j^* such that w_{ij^*t} is the largest resolving the tie arbitrarily. Then, set $Z_{ij^*t} = R_{it}$ and $Z_{ijt} = 0$ if $j \ne j^*$. It is easy to show that this algorithm delivers an optimal solution, and runs in polynomial time in the order of $O(nT)$ [3].

6.3 Comparison with the Existing Practice at Chitika

In Figs. 6.2 and 6.3, we illustrate the effectiveness of presented optimization model using an example with five app aggregators and five ad aggregators [3] . For simplicity, we consider one time slot in this example, i.e., $T = 1$. Figure 6.2 presents the solution of an existing greedy approach used by Chitika, whereas Fig. 6.3 presents the solution of the proposed optimization approach for the same example. Even though we use the actual names for app aggregators and ad aggregators, the values are normalized for confidentiality.

First and foremost, as can be seen in the figures, the improvement in profit (shown in the last column of figures, in 10^4 dollars per day) using the solution of the optimization model is $\frac{1.08-0.78}{0.78} \times 100\,\% = 38.46\,\%$. This improvement results in the increased profit of about \$3000 per day for Chitika. Further, in the greedy approach, the calls are made such that the realized coverage is almost the same as the coverage constraint for all app aggregators. However, in the solution of the optimization model, the realized coverage is much higher than the coverage constraint for two app aggregators (tweetcaster_clicktocall and tweetcaster_clicktocall_iphone). Hence, the optimization model not only improves the profit of Chitika, but also improves the coverage for the app aggregators. This, in turn, improves Chitika's relationship with the app aggregators.

We would also like to point out that, in both approaches, the cost is more than the revenue for some app aggregators (jiwire_iphone and network_android_admarvel). This is caused due to the coverage constraint for each app aggregator. This result

App Aggregators	Number of Ad Requests (R_it)	Coverage Constraint (s_i)	Realized Coverage	Number of Calls for Ad Agrregators					Cost	Revenue	Total Profit
				adelphic	infobar	admarketplace	aolsponsored	businessdotcom			
tweetcaster_clicktocall	1000	0.60	0.64	0	0	0	220	3000	0.386	0.982576	
tweetcaster_clicktocall_iphone	2000	0.60	0.60	0	1200	0	0	0	1.200	1.370386	
jiwire_android	2000	0.60	0.60	1000	300	2500	880	0	0.905	1.154554	0.78
jiwire_iphone	1500	0.40	0.40	0	0	0	600	0	0.420	0.284934	
network_android_admarvel	1000	0.30	0.30	0	0	0	300	0	0.300	0.198965	
			Total	1000	1500	2500	2000	3000	3.210	3.991417	
Upper bound on the number of calls (D_j)				1000	1500	2500	2000	3000			

Fig. 6.2 Solution of the existing practice at Chitika

App Aggregators	Number of Ad Requests (R_it)	Coverage Constraint (s_i)	Realized Coverage	Number of Calls for Ad Agrregators					Cost	Revenue	Total Profit
				adelphic	infobar	admarketplace	aolsponsored	businessdotcom			
tweetcaster_clicktocall	1000	0.60	1.00	0	0	2500	0	0	0.600	0.858000	
tweetcaster_clicktocall_iphone	2000	0.60	0.99	0	0	0	1568	3000	1.985	2.676448	
jiwire_android	2000	0.60	0.60	330	1200	0	432	0	0.900	1.107771	1.08
jiwire_iphone	1500	0.40	0.40	670	0	0	0	0	0.420	0.389226	
network_android_admarvel	1000	0.30	0.30	0	300	0	0	0	0.300	0.251752	
			Total	1000	1500	2500	2000	3000	4.205	5.283196	
Upper bound on the number of calls (D_j)				1000	1500	2500	2000	3000			

Fig. 6.3 Solution of the proposed optimization model

indicates that Chitika needs to be careful in signing contract with app aggregators. More specifically, it may be beneficial for Chitika not to sign contract with some app aggregators. However, the optimal selection of app aggregators is itself an optimization problem. We are currently working on this problem, and plan to present some of these results during conference presentation. This result also suggests that Chitika should negotiate the coverage constraint with some of the app aggregators, such as jiwire_iphone and network_android_admarvel, maybe at the expense of increasing the payment per impression and/or payment per click.

References

1. Ahuja RK, Magnanti TL, Orlin JB (1993) Network flows: theory, algorithms, and applications. Prentice Hall, Englewood Cliffs
2. Garey MR, Johnson DS (1979) Computers and intractability: a guide to the theory of *NP*-completeness. Freeman, San Francisco
3. Mookerjee R, Kumar S, Mookerjee VS, Sriskandarajah C (2014) Demand-supply optimization in mobile advertising. In: Proceedings of the 24th annual workshop on information technologies and systems
4. Orlin JB (1993) A faster strongly polynomial minimum cost flow algorithm. Oper Res 41(2):338–350
5. Wayne KD (1999) A polynomial combinatorial algorithm for generalized minimum cost flow. In: Proceedings of the 31th annual ACM symposium on theory of computing
6. Wayne KD, Fleischer L (1999) Faster approximation algorithms for generalized flow. In: Proceedings of the 10th annual ACM-SIAM symposium on discrete algorithms

Chapter 7
Future Trends and Challenges in Web and Mobile Advertising

Abstract In this chapter, we present future trends in web and mobile advertising, and associated challenges.

7.1 Inclusion of Advertiser Constraints in the Problem of Ad-Firm

For the ad-firms, in addition to the publisher's constraint about exceeding a given click-through-rate (CTR), the advertiser often poses an additional constraint, a cost-per-conversion constraint. Hence, although this problem is similar to the one described in Chap. 5, here the solution must also respect the performance constraint of advertiser. The advertiser's constraint requires that the cost-per-conversion value is below a specified constant. The cost-per-conversion value is the ratio of the total advertising cost, that is, the per-click cost times the number of clicks, divided by the number of conversions (e.g., a sale or registration) that are generated from the clicks. Once again, for a constant cost-per-click value, we can express the advertiser's constraint as one that imposes a lower limit on the ratio of the number of conversions to the number of clicks (or conversion ratio).

To solve this problem, we can use the data analytics step to predict both the probability of a click and the probability of a conversion, similar to that in Chap. 5 [1]. The decision to show an ad depends on both probabilities. That is, we use two thresholds (one for the click probability and the other for the conversion probability), both of which must be satisfied to display an ad. Figure 7.1 depicts the overall solution process [1]. The inclusion of advertiser constraints often applies to situations in which the ad-firm contracts directly with the merchant for ad display, rather than obtaining these ads from a partner.

S. Kumar, *Optimization Issues in Web and Mobile Advertising*, SpringerBriefs in Operations Management, DOI 10.1007/978-3-319-18645-0_7

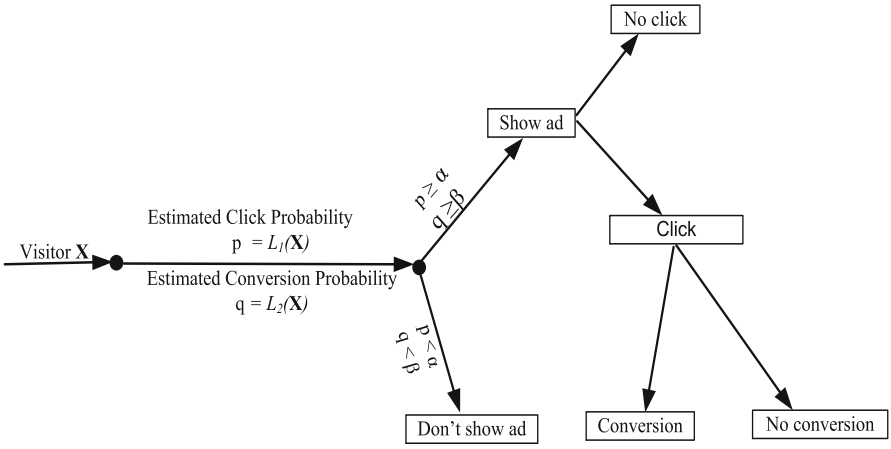

Fig. 7.1 Solution process for a problem with advertiser constraints

7.2 Real-Time Media Buying

To extend its access to publishers, ad-firms as Chitika actively consider buying impressions at advertising exchanges. These exchanges allow advertising networks such as Chitika to place a bid to buy the rights to show ads to a particular visitor on a particular website, which is outside the advertising network's current publisher network. Because of its large publisher network, the ad-firm has often previously encountered these visitors within its network. Therefore, it has visitor profiles on them, which it can leverage to target them with relevant ads. The main difference between the problem addressed in Chap. 5 and the real-time media buying problem is that, in real-time bidding, the ad-firm pays for the impression whether or not a visitor clicks. Thus, instead of a CTR constraint, the cost of the impression becomes part of the ad-firm's objective function. In this problem, the CTR constraint disappears, but a cost-per-impression value is included in the objective function. Thus, Ad-firm's objective in the media buying case is to maximize profit—revenue from clicks minus cost of impressions—by choosing an optimal value of the threshold α. If the advertiser is a direct client of ad-firm, a conversion constraint might still be applicable in the real-time media buying problem. A huge market opportunity awaits the firm that is able to fine-tune and adapt the ideas presented in Chap. 5 to media buying [1].

7.3 Fading Ads

The concept behind fading ads is to show an ad, but fade away after a specific amount of time (e.g., τ), depending on the visitor's previous click history. This is a relaxation of the model discussed in Chap. 5, in which either the time for which the

ad is shown is zero (for the case $p < \alpha$) or the ad is shown for a fixed amount of time. This click history includes data on whether or not the visitor has clicked on ads and the time it took the visitor to generate these clicks. Such an innovation, if implemented correctly, would provide the ad-firm with a very distinctive product in the Internet advertising space [1].

Reference

1. Mookerjee R, Kumar S, Mookerjee VS (2012) To show or not show: using user profiling to manage internet advertisement campaigns at Chitika. Interfaces 42(5):449–464

Index

S. Kumar, *Optimization Issues in Web and Mobile Advertising*, SpringerBriefs in Operations Management, DOI 10.1007/978-3-319-18645-0