2011

Optimal Advertisement Scheduling in Breaks of Random Lengths

Ajay Srinivasan ARAVAMUDHAN
Singapore Management University, ajaysa.2009@mom.smu.edu.sg

Optimal Advertisement Scheduling in Breaks of Random Lengths

by
Ajay S. Aravamudhan

Submitted to Lee Kong Chian School of Business in partial fulfillment
of the requirements for the Degree of
Master of Science in Operations Management

Thesis Committee

Pascale Crama (Supervisor / Chair)
Assistant Professor of Operations Management
Singapore Management University

Onur Boyabatli
Assistant Professor of Operations Management
Singapore Management University

Sharafali Moosa
Associate Professor of Operations Management (Education)
Singapore Management University

Singapore Management University
2011

Abstract

Optimal Advertisement Scheduling
in Breaks of Random Lengths

by Ajay S. Aravamudhan

Broadcasters generate a large part of their revenue through advertising, especially in live sports. Scheduling advertisements can be challenging in live broadcasting, however, for sports such as Cricket that have breaks of random lengths and number during which the ads are shown. This uncertainty, coupled with the high price of spots for major competitions, means that improving ad scheduling can add significant value to the broadcaster. This problem shares similarities with the stochastic cutting stock problem and the dynamic stochastic knapsack problem, with applications in the wood, steel and paper industry and the transportation industry respectively.

This dissertation adds to the existing literature on advertising scheduling by taking stochasticity in break sizes into consideration. We propose an optimal scheduling rule under simplifying assumptions and prove that our policy outperforms traditional scheduling methods. We also study the performance of several heuristics, and find that a flexible heuristic that does not depend on creating bundles performs the best.

Contents

1 Introduction **1**

2 Literature Review **5**

2.1 Revenue Management with Random Capacity 5

2.2 Dynamic Stochastic Knapsack Problem 7

2.3 Stochastic Cutting Stock Problem 8

2.4 Revenue Management in Media Applications 10

3 Optimal Policy **12**

3.1 Assumptions 12

3.2 Known Break Size 14

3.2.1 Base Case 14

3.2.2 Stochastic Number of Breaks 18

3.2.3 Multiple Break Sizes 22

3.2.4 Stochastic Number of Breaks of Multiple Sizes 27

3.3 Unknown Break Size 31

3.3.1 Base Case 32

3.3.2 Stochastic Number of Breaks 36

3.3.3 Multiple Break Sizes and its variants 39

4 Numerical Analysis **41**

4.1 Performance with deterministic number of breaks 41

4.1.1 Parameters 41

4.1.2 Impact of variation in air time sold 42
4.1.3 Value of flexibility 44
4.1.4 Impact of Variability 46

5 Applications in Practice: Scheduling ads for Cricket 52
5.1 Data Description 52
5.1.1 Break Lengths 52
5.1.2 Ad Lengths, Service Levels and Demands 53
5.1.3 Spot Values 54
5.2 Assumptions 55
5.3 Knapsack Model 55
5.3.1 Knapsack model with service guarantee 57
5.4 Heuristics 58
5.4.1 Greedy Policy 58
5.4.2 Certainty Equivalent Heuristic 58
5.4.3 Dynamic Certainty Equivalent Heuristic 58
5.4.4 Dynamic Modified Certainty Equivalent Heuristic 59
5.4.5 Perfect Information 59
5.5 Comparative Statics with Service Constraints 59
5.5.1 Parameters 59
5.5.2 Results 60
5.6 Comparison of Greedy and Optimal Policies 61
5.6.1 Results 62

6 Conclusion 65

Bibliography 67

A Proofs of theorems 70
A.1 Proof of Theorem 3.1 70
A.2 Proof of Theorem 3.2 72
A.3 Proof of Theorem 3.3 75

A.4 Proof of Theorem 3.4 77
A.5 Proof of Theorem 3.5 81
A.6 Proof of Theorem 3.6 83

B Tables **85**

B.1 Revenues 86
B.2 Service Levels 88
B.3 Utilization of breaks 92

Acknowledgements

It is a pleasure to thank the many people who made this thesis possible.

Firstly, I would like to thank Dr. Pascale Crama, my thesis advisor, for teaching me the ways of research and the art of writing well. She has been friendly, approachable, and always right. Her gentle encouragement and words of advice have been invaluable in shaping my development as a research student.

I would also like to thank Dr. Dana Popescu, for clarifying the assumptions and the model we would work with. Her enthusiasm and insight made my work easier.

I would like to express my gratitude to the unnamed sports broadcaster who first introduced the problem to us, and the managers therein. Without them, this thesis would not have been conceivable.

I wish to thank my professors, who provided me with the foundations I required to work on this dissertation. In particular, I would like to thank Dr. Onur Boyabatli and Dr. Sharafali Moosa, who gave me valuable advice during my thesis proposal.

I am indebted to Dr. Reddi Kotha for encouraging me to apply to SMU. I wish I could have lived up better to the potential he saw in me.

Most importantly, I would like to thank my wife, Gina Han, firstly for never complaining while I struggled to provide her with the comforts she deserves, and secondly for being a rock and providing encouragement when I struggled with self-doubt. I look forward to the years in her company.

Chapter 1

Introduction

Revenue for television broadcasters is generated primarily through the sale of local, regional, and national advertising on the local stations and their networks. At CBS, the most-watched U.S. broadcast network, TV advertising accounted for two-thirds of its revenue[1]. Major sporting events, such as the Super Bowl, the Olympics or the Football World Cup, greatly increase advertising revenues as advertisers are willing to pay a premium to air their ads during live broadcast.

In India, cricket is the main revenue earner for sports broadcasting networks. According to a study from TAM Media Research's advertising measurement arm AdEx, ad volumes in cricket saw a growth of more than three times in the five years from 2002 to 2007 with the volumes showing an extra spurt during the World Cups in 2003 and 2007[2]. The biggest spurt was seen in the 2007 World Cup where the volumes rose nearly 100%, with 22% of the advertising volumes in live cricket telecasts.

In recent times, the importance of advertising in cricket has increased even more, with the introduction of shorter formats of the game such as T-20, which is aimed at prime-time television viewers. Sony, the broadcaster for the Indian Premier League 2010, was expected to have earned approximately USD 150 millions from live broadcasting for the tournament alone, with advertising spots valued at more than USD 1100 per second[3]. Thus, even a small percentage gain in advertising revenue can translate to

[1] Bloomberg Businessweek 2010
[2] www.indiantelevision.com/
[3] Wall Street Journal, Jan 2010

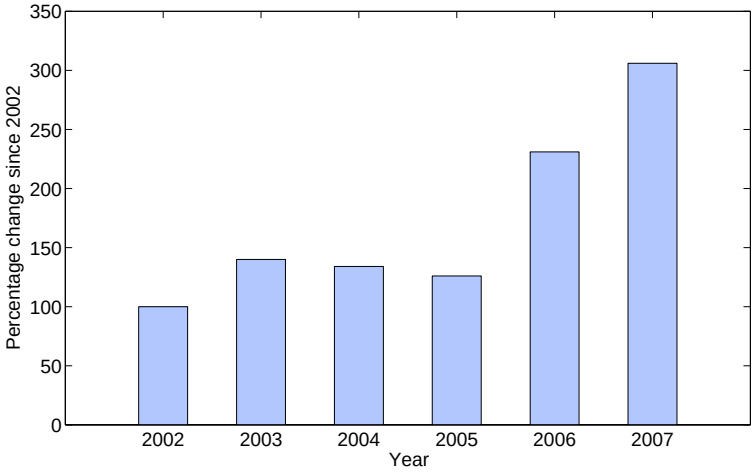

Figure 1.1: Growth of investment in advertising in live cricket, 2002-2007

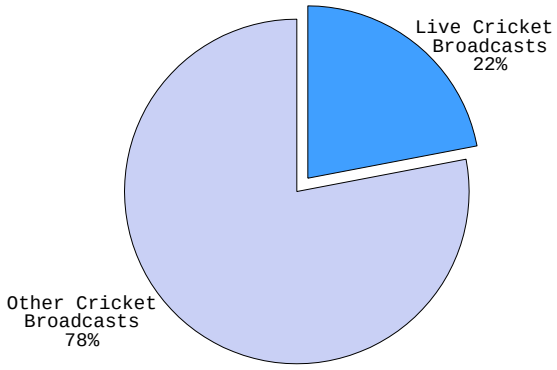

Figure 1.2: Slice of revenue earned through live cricket broadcasts, 2007

a large increase in earnings.

Scheduling advertisements in live broadcasting can be challenging, however, especially for sports that have breaks that are non-deterministic. This is the case for cricket, in which the duration and also the number of breaks can be random. Breaks in cricket are taken between overs, when a wicket falls, or there is a break in the game for refreshments or due to an injury. While the length of the game is usually predictable (especially in the case of limited overs games such as ODIs or T-20s), it is not fixed. Teams have to take up to ten wickets within the allotted overs, however overs may run out before all wickets are taken. Teams may also find that all their wickets have fallen before they have managed to bat out the overs. Thus, the number of breaks, time when they occur, and lengths of breaks in a game cannot be foretold, and broadcasters have to make ad scheduling decisions *while* the break is ongoing.

To do this, networks usually employ several people with specialized knowledge of the sport when scheduling commercials. These ad schedulers have a view of the live game as it happens from a centralized control center, which also has a list of ads available on a mainframe, along with the orders (number of times each ad has to be shown). Before the match starts, the ad scheduling team creates a few sample bundles, giving priority to tournament sponsors. The bundles are initially scheduled as planned, but are broken and ads are scheduled on the fly because break sizes are stochastic. The scheduling team is advised by an on-field director, who can judge the state of the game and inform them about how long the break could be. Based on this advice, and the known break size distribution, the schedulers select ads to be aired in each break.

The job of the ad schedulers is stressful because of the need to develop good schedules while under constant pressure to satisfy scheduling constraints. For instance, if live action begins while an ad is running, then the ad has to be stopped midway (the ad is said to have ‘crashed’) in order to air live action, thus forgoing any revenue from the crashed ad. Furthermore, networks have to satisfy service levels promised to agencies that get them the ad contracts, and these service levels are based on sponsorship statuses and geographical location. Thus, by reducing dependency on human intervention, there is an opportunity to maximize revenues by automating the commercial scheduling

process while generating near optimal schedules to meet all goals.

In this dissertation, we consider two approaches to study the problem. We begin with Chapter 2, where we review literature from the streams of advertising, random yield, stochastic knapsack and stochastic programming with recourse to help us gain insights for doing our study. In Chapter 3, we approach the problem analytically, and study optimal scheduling policies when faced with stochastic break sizes for a simplified setting, in which we do not consider constraints other than crashing. We consider two cases, where the scheduling team either has prior information about the break duration for the ongoing break, or it does not. In each case, we assume that the inventory contains ads of two sizes, S and $2S$, and consider scenarios where break sizes range from zero to any multiple of S, and where the number of breaks can be stochastic. In Chapter 4, we do a numerical analysis to study the sensitivity of the Optimal Policy to variations in problem parameters relative to the myopic Greedy rule, which we define as the ad schedule which generates the maximum revenue for the break in hand but does not consider the subsequent breaks. Finally, in Chapter 5, we study several heuristics inspired by the current scheduling practice in order to improve a broadcaster's revenues. We use data provided by a major cricket broadcaster to analyze the performance of several heuristics which create bundles beforehand. We consider generating bundles at various points during a match, and compare performance with the standard Greedy heuristic. We include some constraints for this study, such as minimum service levels for each client, to analyze how creating bundles beforehand affects revenue earned, and how often they should be created.

Finally, we summarize our findings and present our conclusions with directions for future research in Chapter 6.

Chapter 2

Literature Review

In this dissertation, we take two approaches to study the problem of optimally scheduling ads for random breaks: first we propose optimal policies for scheduling ads during breaks of random durations under simplifying assumptions, and secondly, we analyze several heuristics inspired by current scheduling practice hoping to provide managerial insight to sports broadcasters. This research has links to revenue management with random capacity, the dynamic stochastic knapsack problem, the stochastic cutting stock problem and revenue management in media applications.

2.1 Revenue Management with Random Capacity

Our problem is related to revenue management with random capacity / random yield, with typical applications in production planning. The most common choice to model random yield has been stochastically proportional yield, in which the yield is proportional to the order.

Ciarallo et al. (1994) [7] are the first to explore the impact of random capacity. The authors find that an order-up-to policy is optimal to minimize production costs. The order target includes a safety stock to account for random capacity in future periods and is higher than the myopic order-up-to level. Wang and Gerchak (1996) [27] revisit the results found in Ciarallo et al. (1994) to offer a more rigorous proof of the order-up-to policy. Khang and Fujiwara (2000) [15] prove under which conditions the myopic

order-up-to policy is optimal in a multi-period setting. Hwang and Singh (1998) [14] extend the analysis to a multi-stage production process and find an optimal policy characterized by a sequence of two critical numbers for each stage: a minimum input level below which no production takes place and a maximum desired production level. Finally, Wang and Gerchak (1996) [26] incorporate random yield and capacity and show that the optimal policy is characterized by a single reorder point in each period. That critical point is not a constant and depends on the inventory in hand.

Yano and Lee(1995)[29] review the literature in the area of lot sizing with random yields, focusing on single-stage continuous review models and single-stage periodic review models. They cover modeling of costs, modeling of yield uncertainty, and measures of performance of the system. Grosfeld-Nir and Gerchak (2004)[13] review papers discussing multiple lot sizing in production to order in multistage systems, and review situations where both yield and demand are random. Bollapragada and Morton(1999)[6] provide heuristics for dealing with random inventory by focusing on inventory at the end of the period, after the demand is met. They show that the random yield problem is analogous to the newsvendor problem with the demand distribution dependent on the quantity ordered. Their research supports the argument that myopic and near-myopic methods are useful across a wide spectrum of stochastic inventory problems.

Our model differs in two important aspects from the random yield and random capacity papers above. First, we maximize revenues rather than minimize costs. All the papers assume a single product whereas we work in a multi product setting with different prices and production costs. Therefore we need to schedule those products based on their profitability and their capacity usage. Second, we assume integer units or fixed-size order runs. Therefore, we cannot simply use capacity to its maximum and hold inventory to complete an order across multiple periods. Each order needs to be entirely processed within one production period.

2.2 Dynamic Stochastic Knapsack Problem

The knapsack problem is one of the simplest and thus oldest formulation of a maximization problem. The knapsack problem has been extensively studied in operations research, and has various industrial applications in areas such as resource allocation, capital budgeting, portfolio selection problem, cargo loading, and cutting stock problems. Knapsack problems of this type are deterministic because all parameters are known with certainty. However, in many situations, these parameters may be random variables having a certain distribution. Kleywegt and Papastavrou have written a series of papers on this topic ((1996) [19] [21],(2001) [20]), that define the Dynamic Stochastic Knapsack problem as one in which items to be packed arrive according to a known distribution, and determine the optimal policy that maximizes expected value, given the costs associated with waiting. They expand their research to cases where the rewards associated with an item are stochastic, and when the size of each item is also stochastic. Our research, however, attempts to solve a problem with multiple knapsacks, whose sizes are stochastic, and the items are of known size and value.

More recently, Perry and Hartman (2009) [22] model a multi-period, single resource capacity reservation problem as a dynamic, stochastic, multiple knapsack problem with stochastic dynamic programming. They propose an approximation approach which utilizes simulation and deterministic dynamic programming in order to allow for the solution of longer horizon problems and ensure good time zero decisions. Their simulation based approach, however, does not sufficiently capture the complexities of our problem.

Witchakul, Ayudhya, Charnsethikul(2008) [28] discuss random Knapsack capacity with deterministic weights and costs. They model the Knapsack's capacity as a random variable with a known distribution. They use the expectation of Knapsack size, and both underage and overage penalty costs, to estimate optimum selection of ads. They provide a heuristic for solving Stochastic Knapsack with Continuous/Discrete Random Capacity, and prove the validity of their heuristic analytically and numerically using a Monte Carlo Simulation. To the best of our knowledge, this is one of the few papers

that address stochastic knapsack sizes. The problem we face, however, can be seen as a modified bin packing problem where multiple bins of stochastic capacity have to be optimally filled. Besides, unlike the above paper, we do not consider overage and underage penalties.

2.3 Stochastic Cutting Stock Problem

The cutting stock problem originated as a knapsack problem which minimizes unused capacity rather than maximize revenue from the included items. It is based on industry applications which require to solve how to cut stock of a certain dimension into smaller, heterogeneous order sizes in such a way as to minimize waste of material, e.g. in the paper or steel industry. The problem was introduced by Gilmore and Gomory (1961) [10], and over a series of papers, the authors proposed a set of specialized techniques to solve the cutting stock problem (Gilmore and Gomory 1963 [11], 1965 [12]). One line of extensions to this problem looks at stock with stochastic dimensions. The randomness can be due to the nature of the stock, e.g., raw material like wood or stone slabs may come in unequal sizes, quality variation within the stock or defects at the edges of the stock. Scull (1981) [23] introduces a stochastic cutting stock problem in which the uncertainty in the stock length is due to defects at the edges. The stock is then cut into standard-length units, and the authors find the optimal distance from the edge at which to start cutting in order to minimize expected waste if inspection of the stock and its defects is not possible. Ghodsi and Sassani (2005) [9] introduce quality and length variability of the stock and the orders. A cut pattern needs to be decided upon arrival of each piece of stock. The authors propose a dynamic algorithm which first prioritizes the orders based on their quality level and quantity and then proposes a suitable cut pattern for the incoming stock. Even though the orders have different quality requirements, their revenue is assumed constant, and the objective is to minimize waste. Fathi and Kianfar (2009) [16] acknowledge that variability in quality may also lead to difference in revenue, and formulate a similar problem with quality and length variability with the objective to maximize revenue. They formulate the cut pattern problem as a dynamic

program and conduct a numerical experiment to show that it is feasible to solve this problem in real time. The authors, however, do not comment on the performance of their algorithm with respect to revenue or compare it across different heuristics.

Much research has been devoted in the field of stochastic programming to solve multi-stage recourse problems, and Birge (1997)[3] gives a summary of formulations and solution techniques. He gives a general model of the *multistage stochastic linear programming with recourse*. This formulation shares some characteristics with the problem at hand, since the inventory of ads available changes from one break to the next based on the realization of break size. Birge goes on to describe solution procedures such as extreme-point methods, interior point methods and column splitting. However, the challenges we face are different as our problem is an integer programming problem, while considerations of integrality are not touched upon.

Techniques for solving Stochastic Integer Programs are available in Birge and Louveaux (1997) [4]. The modified L-Shaped method suggested by them integrates *branch and bound* with the standard L-shaped method, thus an extra step is added where integrality constraints are checked for every feasibility cut introduced as part of the L-Shaped method. However they also state that "loosely stated, for this class of problems, is very unlikely that an algorithm will be found that would solve the problem in a number of operations polynomial in the problem data... If the second stage of a stochastic problem corresponds to an NP-hard problem, it is pointless to design an exact method that would require the solution of the second stage for each realization of the random variable". Thus, it warrants a study of heuristics or alternate algorithms that can actually run in polynomial time - even if their solutions are global sub-optimal - that can improve on the performance of schedules created manually. More recently, Haneveld and Van der Vlerk(1999) [18] survey structural properties of and algorithms for stochastic integer programming models, mainly considering linear two stage models with mixed integer recourse (and their multi-stage extensions). However, they also observe that "special purpose algorithms will turn out to be necessary to obtain good computational results for many real-life applications." Sen (2005) [24] studies algorithms for both two-stage as well as multi-stage stochastic mixed-integer programs. He

presents stage wise (resource directive) decomposition methods for two-stage models, and scenario (price directive) decomposition methods for multi-stage models. He also studies a variety of structures ranging from models that allow randomness in all data elements, to those that allow only specific elements to be influenced by randomness. He discusses *branch and price* and *Lagrangian relaxation* for multi-stage SMIP, but states that stage wise decomposition algorithms for the two stage case but states that scalability of the stage wise decomposition to multi-stage scenarios is suspect.

The literature survey on integer stochastic programming with recourse conveys that a specialized algorithm is in order for us to solve the multistage stochastic integer program we have in hand.

2.4 Revenue Management in Media Applications

Literature for revenue management for advertising in TV broadcasting has looked at the joint order acceptance and scheduling problem with deterministic break lengths. For example, Kimms and Muller-Bungart(2007)[17] formulate an integer program that maximizes the broadcaster's revenue, while taking into account non-conflicting product constraints and specific scheduling requests. The authors also propose several heuristics and conduct extensive numerical analyses that compare performance across the different solution methods. Bollapragada and Garbiras(2003)[5] also discuss ad scheduling but assume a deterministic audience distribution and given client preferences. They automate the commercial scheduling process while generating near optimal schedules to meet constraints (such as product conflict requirements and position percentage), and have implemented it in NBC. Zhang(2006)[30] uses a hierarchical structure using a model that uses a two step hierarchical approach, where winners (advertisers) are selected first and then slots are assigned to selected commercials.

There has also been work done in the area of slot allocation and contract selection for deciding on the inventory of ads a network has at the time of broadcasting. Araman and Popescu(2007) [1] develop a model for allocating advertising slots between up front and scatter markets under audience uncertainty in up front and operational planning

decisions. Kimms and Muller Bungart(2007) [17] discuss simultaneous optimization of optimal contract selection and ad scheduling. They provide heuristics for optimal ad scheduling based on contract constraints (such as at which position in the break an ad can be shown, etc.) Our focus in this dissertation, however, will be on the optimal selection of ads in a given break and not on contract selection.

Finally, Degraeve and DeReyck(2003)[8] discuss broadcast ad scheduling using SMS. Their model uses a linear decomposition of three schedules that are prepared before the broadcast begins, given a limited capacity of broadcast time slots, maximizing customer response and revenues from retailers paying for each broadcast. The problem we analyze has different parameters from those outlined in the papers above. Traditional ad scheduling heuristics consider deterministic break sizes, and differ only in the number and type of constraints they face. To the best of our knowledge, the problem of scheduling ads in stochastic break sizes under similar constraints as those faced when break sizes are deterministic has never been studied before.

Chapter 3

Optimal Policy

In this chapter, we will determine optimal selection policies that maximize expected revenue under some simplifying assumptions. Our objective is to find a general set of rules that aids ad schedulers when faced with random break lengths.

These rules are also applicable for a class of bin scheduling and cutting stock problems where bin sizes are non-deterministic and items are of known weight and value. Previous research in this area has focused on randomness in item value and weight; our contribution will be to add to this literature by considering randomness in bin sizes, and extend it to cases where the number of bins is also stochastic. We will use the terms "breaks" and "ads" throughout this document, but these can be substituted with "bins" and "items" for the general stochastic bin scheduling problem.

We propose a dynamic programming solution methodology where the return function is a preference selection criterion and illustrate the conditions required to guarantee optimality of the selection. We also contrast the behavior of the Optimal Policy with that of the Greedy Policy, and draw insights and implications.

3.1 Assumptions

We consider a scenario where breaks $\{b_1, b_2, \ldots, b_n\}$ occur sequentially. The number of breaks has an upper bound $N \geq k \geq 1$, and the capacities of the breaks follow a known distribution, and are IID.

A decision has to be made as to what ads are to be put in the next available break. For simplicity, we assume that we are always planning for break b_1, and the index n is the number of remaining breaks expected to occur.

We assume that we only have two types of ads in the inventory: small ads of size S and large ads of size $L = 2S$. This is representative of the types of ads currently used in American television networks, where ads are usually of either 15 second or 30 second durations.

We assume that the number of ads we have in our inventory is infinite, with possibly a fraction of those ads having a non-zero value. Further, the ads are arranged in descending order of value.

Thus, if $\mathbb{S}$ and $\mathbb{L}$ are the sets of small and large ads respectively, then

$$\mathbb{S} = \{s_1, s_2, \ldots, 0, \ldots\}$$
$$\mathbb{L} = \{l_1, l_2, \ldots, 0, \ldots\}$$

where $s_i \geq s_{i+1} \geq 0$ and $l_i \geq l_{i+1} \geq 0 \ \forall i$. We consider two scenarios: the size of the current break is either known to the scheduler before he begins scheduling, or it is unknown. For each scenario, we consider the following cases:

1. A base case, where the size of each break is limited to either S or L, and the number of breaks is fixed and known in advance,

2. An extension where the number of breaks is bounded but not fixed,

3. An extension where break sizes are any bounded multiple of S,

4. An extension where break sizes are any bounded multiple of S and the number of breaks is stochastic.

We assume that the revenue earned by each ad includes the sponsor status of an advertiser. We do not discuss per-client service levels in this chapter, instead focusing on service levels of large and small ads in general. We discuss per-client service levels in Chapter 5.

We define the Greedy Policy as follows:

Definition 3.1 (Greedy Policy)**.** *For each break, select the combination of ads that earns the highest revenue in that break.*

We also assume that ads that are not fully aired do not earn any revenue. Thus, by Definition 3.1, the Greedy Policy only selects ads that can be aired completely within the break.

3.2 Known Break Size

Our motivation to study this scenario comes from Cricket broadcasts where the on-field director can predict the length of the current break, depending on the type of break being taken and the state of the game. The ad schedulers then decide what ads to show in the current break based on the advice given by the on-field director, keeping in mind that the sizes of subsequent breaks are unknown.

This scenario could also occur in freight shipping where the shipper knows the capacity available in the next arriving ship but not those of subsequent ships, and has to build an appropriate consignment given that an certain number of ships are expected to follow.

Although the size of the current break is known, the scheduler should look ahead to decide his selection, so that revenue earned over all k breaks is maximized. The Greedy Policy is not globally optimal because it fails to consider the subsequent breaks and the stochasticity in their sizes.

We now look at the four cases mentioned in section 3.1, and discuss Optimal Policy for each.

3.2.1 Base Case

We assume that breaks are either of size S or of size $L = 2S$, and there is no uncertainty about the number of breaks remaining. Breaks of size S arrive with probability p and breaks of size L arrive with probability $1 - p$.

One break

Only one ad of size S can be scheduled in a small break. Thus, if $b_1 = S$, s_1 is always packed. It is trivial to prove that this is the case irrespective of the number of breaks left, hence we will not discuss selection policy when the current break size is S in subsequent sections.

In a large break, however, one ad of size L or two ads of size S can be aired. Selection for a large break when $k = 1$ reduces to a Greedy Policy. The scheduler chooses between the more profitable of (s_1, s_2) and l_1. We state the following lemma without proof:

Lemma 3.1. *When $k = 1$, and $b_1 = L$, the optimal policy is to:*

1. *select l_1 if $l_1 \geq s_1 + s_2$*
2. *select (s_1, s_2) if $s_1 + s_2 > l_1$*

Two breaks

b_1	b_2	Optimal selection	
S	S	s_1, s_2	
S	L	$s_1, s_2 + s_3$	if $s_2 + s_3 \geq l_1$
		s_1, l_1	otherwise
L	S	$s_1 + s_2, s_3$	if $s_1 + s_2 \geq l_1$
		l_1, s_1	otherwise
L	L	$s_1 + s_2, s_3 + s_4$	if $s_3 + s_4 \geq l_1$
		$s_1 + s_2, l_1$	if $s_1 + s_2 \geq l_1 > s_3 + s_4$
		$l_1, s_1 + s_2$	if $l_1 > s_1 + s_2 \geq l_2$
		l_1, l_2	otherwise

Table 3.1: Possible selection options for $k = 2$

For $k = 2$, we can write an exhaustive list of all the possible cases. These are listed out in Table 3.1 From the table, we see that when $b_1 = L$, the scheduler would choose (s_1, s_2) if:

$$(s_1 + s_2) + ps_3 + (1-p)\max\{l_1, s_3 + s_4\} \geq l_1 + ps_1 + (1-p)\max\{l_2, s_1 + s_2\} \quad (3.1)$$

We use the above equation to prove the optimal policy.

Lemma 3.2. *When $k = 2$, and $b_1 = L$, the optimal policy is to:*

1. *select l_1 if $l_1 \geq s_2 + s_3$*
2. *select (s_1, s_2) if $l_1 < s_2 + s_3$*

Proof. **Case 1**:
Let $l_1 \geq s_2 + s_3$. Let R_l and R_s be the revenues earned by selecting l_1 first and $s_1 + s_2$ first respectively. From equation 3.1, we can write:

$$\begin{aligned}
&R_l - R_s \\
&\quad = [l_1 + ps_1 + (1-p)\max\{l_2, s_1 + s_2\}] \\
&\quad\ - [(s_1 + s_2) + ps_3 + (1-p)\max\{l_1, s_3 + s_4\}] \\
&\quad \geq [l_1 + ps_1 + (1-p)(s_1 + s_2)] - [(s_1 + s_2) + ps_3 + (1-p)l_1] \\
&\qquad (\because l_1 \geq s_2 + s_3 \geq s_3 + s_4) \\
&\quad \geq p(l_1 - (s_2 + s_3)) \\
&\quad \geq 0 \ \ (\because l_1 \geq s_2 + s_3)
\end{aligned}$$

Therefore, when $l_1 \geq s_2 + s_3$, $R_l \geq R_s$, so l_1 is packed first. Similarly, we can prove **Case 2** by showing that $R_s - R_l \geq 0$ when $l_1 < s_2 + s_3$. □

While the Greedy Policy compares l_1 and $s_1 + s_2$, Lemma 3.2 compares l_1 and $s_2 + s_3$, setting a lower threshold for l_1 to be optimal, and improving its chances of being selected.

If $b_2 = L$ and **Case 1** applies, then the Optimal Policy selects $s_1 + s_2$ for b_2, (from Lemma 3.1), while the Greedy Policy selects l_1, and both policies earn equally.

Consider, however, the case when $b_2 = S$. The Greedy Policy then earns $s_1 + s_2 + s_3$. The Optimal Policy, however, earns $l_1 + s_1$ which is more than the revenue earned by the Greedy Policy, since $l_1 \geq s_2 + s_3$.

When $l_1 < s_2 + s_3$, and we schedule l_1 in b_1, then we are forced to schedule s_1 in b_2 when $b_2 = S$, earning $l_1 + s_1$. However, we know that $s_1 + s_2 + s_3 > l_1 + s_1$, therefore it is sub-optimal to schedule l_1 in b_1. Thus, the condition in **Case 1** describes the optimal threshold for l_1 to be an attractive candidate for b_1.

We next take a look at the multiple break case.

Multiple breaks

When the number of breaks remaining is greater than two, we see that the policy outlined in Lemma 3.2 extends to look ahead to *all* the remaining breaks.

Theorem 3.1. *When $k = n$, and $b_1 = L$, the optimal policy is to:*

1. *select l_1 if $l_1 \geq s_n + s_{n+1}$*
2. *select (s_1, s_2) if $l_1 < s_n + s_{n+1}$*

Proof. Proof is in Appendix A. □

Consider the situation where all breaks subsequent to b_1 are of size S. If we scheduled(s_1, s_2) in b_1, we would have scheduled ads $(s_1, s_2, \ldots, s_n, s_{n+1})$ at the end of the planning period. Therefore, we get a higher revenue by selecting $(l_1, s_1, s_2, \ldots, s_{n-1})$, since $l_1 \geq s_n + s_{n+1}$.

It is trivial to see that if one or more large breaks arrive instead, the Optimal Policy would earn at least as much as the Greedy.

As the number of breaks remaining increases, the threshold above which it becomes optimal to select l_1 decreases, and selecting l_1 becomes more attractive. Service levels of large ads, therefore, are higher with the Optimal Policy than with the Greedy Policy when breaks of size L occur and values of large ads fall in the range $(s_n + s_{n+1},\ s_1 + s_2)$.

Service levels of small ads are higher with the Greedy Policy than with the Optimal Policy, since the Greedy Policy has a higher threshold for scheduling large ads, and is more likely to schedule small ads for the large breaks as well.

In extreme cases, the Greedy Policy can schedule *all* small ads in the long breaks and be left with small ads of value zero, losing the opportunity to earn from small breaks.

3.2.2 Stochastic Number of Breaks

In this section, we consider a scenario where the number of breaks remaining has an upper bound, but is not fixed. To model this scenario, we assume that *all* breaks arrive, but that some breaks have size zero. We assign probabilities p_0 that a break arrives with size zero, p_1 that the break has size S and p_2 that the break has size L, where $p_0 + p_1 + p_2 = 1$.

The motivation to model this scenario comes from cases where broadcasting networks cannot predict how many breaks they can take in the game (but know the maximum possible number of breaks possible). In cricket, for instance, a break is taken every time a wicket falls. The maximum number of wickets that can fall in an innings is ten, however, the actual number of wickets that fall in each innings may be lower. This scenario can occur in other situations as well, for instance in freight shipping, where ships can arrive but have no space for accommodating the consignment to be shipped.

We study the decisions of the scheduler when the number of breaks remaining are one and two, then use induction to find the optimal policy for the general case.

One break

Selecting ads to be scheduled for one break of known length is trivial, and exactly the same as outlined in Lemma 3.1. The introduction of breaks of size zero does not affect ad selection because we already know the size of the (one) break that has to be scheduled. As before, if the break is of size S, we schedule s_1, and if it is L, we choose the larger of l_1 and $(s_1 + s_2)$. If it is of size zero, we schedule nothing.

Two breaks

When there are two breaks in the planning period, the Optimal Policy must look ahead to b_2 to decide on the optimal selection for b_1. The second break could be of size zero, which would make the two break case the same as the one break case. Therefore, with probability p_0, the greater of l_1 and (s_1+s_2) should be selected. However, with probability p_1+p_2, b_2 could be non-zero, and Lemma 3.2 applies. We state the Optimal Policy for two breaks formally below.

Lemma 3.3. *When $k=2$, and $b_1=L$, the optimal policy when $p_0 \geq 0$ is to:*

1. *select l_1 if $(p_0+p_1)l_1 \geq p_0(s_1+s_2)+p_1(s_2+s_3)$*
2. *select (s_1,s_2) otherwise*

Proof. **Case 1**:
Let $(p_0+p_1)l_1 \geq p_0(s_1+s_2)+p_1(s_2+s_3)$, and let R_l and R_s be the revenues earned by selecting l_1 first and s_1+s_2 first respectively. Then:

$$\begin{aligned} R_l - R_s = &[l_1 + p_0 0 + p_1 s_1 + p_2 \max\{l_2, s_1+s_2\}] \\ &- [(s_1+s_2) + p_0 0 + p_1 s_3 + p_2 \max\{l_1, s_3+s_4\}] \end{aligned}$$

However,

$$\begin{aligned} & (p_0+p_1)l_1 \geq p_0(s_1+s_2)+p_1(s_2+s_3) \\ \implies & (p_0+p_1)l_1 \geq p_0(s_3+s_4)+p_1(s_3+s_4) \\ \implies & l_1 \geq (s_3+s_4) \text{ assuming } p_0+p_1>0 \end{aligned}$$

We assume that $p_0+p_1>0$, otherwise we would have the degenerate case where $p_2=1$

and all breaks would be of the same size L. Substituting for $R_l - R_s$, we get:

$$\begin{aligned} R_l - R_s &\geq [l_1 + p_1 s_1 + p_2(s_1 + s_2)] - [(s_1 + s_2) + p_1 s_3 + p_2 l_1] \\ &\geq (p_0 + p_1) l_1 - p_0(s_1 + s_2) - p_1(s_2 + s_3) \\ &\geq 0 \ \ (\because (p_0 + p_1) l_1 \geq p_0(s_1 + s_2) + p_1(s_2 + s_3)) \end{aligned}$$

Therefore, when $(p_0 + p_1) l_1 \geq p_0(s_1 + s_2) + p_1(s_2 + s_3)$, $R_l \geq R_s$, so l_1 is selected. Similarly, we can prove **Case 2** by showing that $R_s - R_l \geq 0$ when $(p_0 + p_1) l_1 < p_0(s_1 + s_2) + p_1(s_2 + s_3)$. □

If $b_2 = 0$, we compare l_1 and $(s_1 + s_2)$, since nothing can be scheduled into b_2. In other words, select l_1 if:

$$p_0 l_1 \geq p_0(s_1 + s_2) \tag{3.2}$$

If $b_2 = S$, selecting l_1 in b_1 earns $l_1 + s_1$ and selecting (s_1, s_2) in b_1 earns $(s_1 + s_2 + s_3)$, so we would select l_1 if:

$$\begin{aligned} p_1(l_1 + s_1) &\geq p_1(s_1 + s_2 + s_3) \\ \implies p_1 l_1 &\geq p_1(s_2 + s_3) \end{aligned} \tag{3.3}$$

Finally, when $b_2 = L$, the Optimal Policy selects (s_1, s_2) in b_2 and the Greedy Policy selects l_1 in b_2, therefore both policies earn equally. Therefore, Lemma 3.3 checks the expected value earned by selecting l_1 against the expected value earned by selecting (s_1, s_2) when b_2 is not of size L. Combining Equation 3.2 and Equation 3.3, we get the condition described in **Case 1**.

We next look at the multiple break case and use induction to prove the Optimal Policy.

Multiple Breaks

When there are multiple breaks remaining, we look ahead to all remaining breaks to form the rule. We state the Optimal Policy formally as follows.

Theorem 3.2. *When $k = n$, $p_0 \geq 0$, and $b_1 = L$, the optimal policy is to:*

1. *select l_1 if* $(p_0 + p_1)^{n-1} l_1 \geq \sum_{i=0}^{n-1} \left[\binom{n-1}{i} p_0^{(n-1)-i} p_1^i \, (s_{i+1} + s_{i+2}) \right]$
2. *select (s_1, s_2) otherwise*

Proof. Proof is in Appendix A. □

The intuition behind Theorem 3.2 is similar to that of Lemma 3.3. If a break of size L were to occur subsequent to the current break, both the Greedy Policy and the Optimal Policy would earn equal revenues.

The Optimal Policy estimates the value of scheduling l_1 assuming that none of the subsequent breaks are of size L. The reasoning is similar to the one used in Equation 3.2 and Equation 3.3, except in the multiple break case where the number of breaks is $n-1$, i breaks of size S and $n-1-i$ breaks of size zero can occur with probability $\binom{n-1}{i} p_0^{(n-1)-i} p_1^i$. The rest of the reasoning follows.

The threshold above which l_1 is an attractive ad to be scheduled has increased due to the introduction of p_0. For instance, in the base case, for l_1 to be optimal when two breaks remained, l_1 had to be greater than $s_2 + s_3$. However, l_1 now has to be greater than

$$\frac{1}{(p_0 + p_1)} (p_0(s_1 + s_2) + p_1(s_2 + s_3)) \geq s_2 + s_3$$

The Optimal Policy weighs the advantages of scheduling the large ad against the probability that many of the subsequent breaks could be of size zero and hence earn nothing. It does this by adjusting the threshold for scheduling l_1 based on the break size distribution.

The Optimal Policy helps networks decide on their preferred mix of ads based on the distribution of break sizes. When there is randomness in the number of breaks,

networks require a higher value for large ads to be shown, compared to the case where the number of breaks is fixed.

3.2.3 Multiple Break Sizes

In this section, we study the Optimal Policy when break sizes are distributed between $[S, 2S, \ldots, MS]$. In addition, the size of the current break is assumed to be mS, where $m \leq M$. As before, we study Optimal Policy when the number of breaks remaining are one and two, and use induction to prove the Optimal Policy for the multiple break case.

One Break

For the one break case, we choose a set of ads that give us the best possible revenue within the known break size mS. This corresponds exactly with the Greedy Policy.

Lemma 3.4. *If $b_1 = mS$ and $k = 1$, then the Optimal Policy is to select $(l_1, \ldots, l_\lambda, s_1, \ldots, s_{m-2\lambda})$, where λ is the largest index such that:*

- $l_\lambda \geq s_{m-2\lambda+1} + s_{m-2\lambda+2}$
- $2\lambda \leq m$

Proof. Let $\hat{O} = (l_1, \ldots, l_\lambda, s_1, \ldots, s_{m-2\lambda})$ be the set that we want to prove is optimal. To prove optimality of $\hat{O}$, we have to prove that any change to $\hat{O}$ will cause the revenue to decrease.

We note that if $\exists \lambda : l_\lambda \in \hat{O}$, then $\forall i \leq \lambda$, $l_i \in \hat{O}$, since

$$l_i \geq l_\lambda \geq s_{m-2\lambda+1} + s_{m-2\lambda+2} \geq s_{m-2i+1} + s_{m-2i+2} \text{ and } 2i \leq 2\lambda \leq m$$

Now consider the case where we do not select some $l_i \in [l_1, l_\lambda]$. Then, we can either include $(s_{m-2\lambda+1}, s_{m-2\lambda+2})$, or we can include $l_{\lambda+1}$.

Selecting $(s_{m-2\lambda+1}, s_{m-2\lambda+2})$ is inferior because $l_i \geq l_\lambda \geq (s_{m-2\lambda+1} + s_{m-2\lambda+2})$, therefore our revenue will decrease. Similarly, $l_i \geq l_{\lambda+1}$, so substituting l_i with $l_{\lambda+1}$ will also decrease our revenue. It is trivial to prove that changing any $s_i \in [s_1, s_{m-2\lambda}]$

with $s_{m-2\lambda+1}$ will similarly cause a drop in revenue. Therefore $\hat{O}$ is the optimal set of ads to be scheduled when $b = 1$. □

The proof for Lemma 3.4 shows that every $l_i \in [l_1, l_\lambda]$ should be scheduled for maximum revenue. In general, it is sufficient to prove that l_λ (as defined in Lemma 3.4) must be scheduled; since every $l_i \geq l_\lambda$ belongs to the optimal set, optimality of selecting l_λ implies optimality of selecting l_i.

We now look at the two break case.

Two Breaks

The two breaks case follows Lemma 3.2; we now check if the large ad l_λ earns more than the sum of the small ads at indexes $m-2\lambda+2$ and $m-2\lambda+3$. We state the Lemma formally below.

Lemma 3.5. *If $b_1 = mS$ and $k = 2$, then the Optimal Policy is to select $(l_1, \ldots, l_\lambda, s_1, \ldots, s_{m-2\lambda})$, where λ is the largest index such that:*

- $l_\lambda \geq s_{m-2\lambda+2} + s_{m-2\lambda+3}$
- $2\lambda \leq m$

Proof. As before, let $\hat{O} = (l_1, \ldots, l_\lambda, s_1, \ldots, s_{m-2\lambda})$.

Let:

$$O_l = (l_1, \ldots, l_\lambda, l_{\lambda+1}, s_1, \ldots, s_{m-2\lambda-2})$$
$$O_s = (l_1, \ldots, l_{\lambda-1}, s_1, \ldots, s_{m-2\lambda+2})$$

Let R_o, be the revenue earned by selecting the ads in $\hat{O}$, and let R_l and R_s be the revenues earned by selecting O_l and O_s respectively.

We have to prove that:

1. $R_o \geq R_l$ and
2. $R_o \geq R_s$

It is trivial to prove that if $l_{\lambda+1} \geq s_{m-2\lambda} + s_{m-2\lambda+1}$ and $2(\lambda+1) > m$, then substituting $l_i \geq l_{\lambda+1}$ from $\hat{O}$ and introducing $l_{\lambda+1}$ is sub optimal (for the same reason that we pack l_1 in a break of length L, not l_2).

We use $V_i(l,s)$ to denote expected revenue earned from scheduling ads for break i onwards, and the indexes of the first large ad and the first small ad in our inventory are l and s respectively. Then,

$$R_o = \sum_{i=1}^{\lambda} l_i + \sum_{j=1}^{m-2\lambda} s_j + V_2(\lambda+1, m-2\lambda+1) \tag{3.4}$$

$$R_l = \sum_{i=1}^{\lambda+1} l_i + \sum_{j=1}^{m-2\lambda-2} s_j + V_2(\lambda+2, m-2\lambda-1) \tag{3.5}$$

$$R_s = \sum_{i=1}^{\lambda-1} l_i + \sum_{j=1}^{m-2\lambda+2} s_j + V_2(\lambda, m-2\lambda+3) \tag{3.6}$$

Case 1:

$$\begin{aligned} R_o - R_l = &\sum_{i=1}^{\lambda} l_i + \sum_{j=1}^{m-2\lambda} s_j + V_2(\lambda+1, m-2\lambda+1) \\ &- \left[\sum_{i=1}^{\lambda+1} l_i + \sum_{j=1}^{m-2\lambda-2} s_j + V_2(\lambda+2, m-2\lambda-1) \right] \end{aligned}$$

$$\begin{aligned} &= -l_{\lambda+1} + s_{m-2\lambda-1} + s_{m-2\lambda} \\ &\quad + V_2(\lambda+1, m-2\lambda+1) - V_2(\lambda+2, m-2\lambda-1) \end{aligned}$$

If $l_{\lambda+1} < s_{m-2\lambda} + s_{m-2\lambda+1}$, then $l_{\lambda+2} < s_{m-2\lambda-2} + s_{m-2\lambda-1}$. Then by induction, if $b_2 \geq 2S$, $V_2(\lambda+2, m-2\lambda-1)$ would earn $(s_{m-2\lambda-1} + s_{m-2\lambda})$ followed by the sum of values of ads selected from the set $\{l_{\lambda+2}, \ldots, s_{m-2\lambda+1}, \ldots\}$. For a break of corresponding size, $V_2(\lambda+1, m-2\lambda+1)$ would earn *at least* $l_{\lambda+1}$ followed by the sum of values

of ads from the set $\{l_{\lambda+2},\ldots,s_{m-2\lambda+1},\ldots\}$. Thus, we can write:

$$\begin{aligned} V_1(\lambda+1,m-2\lambda+1)-V_1(\lambda+2,m-2\lambda-1) \\ \geq p_1(s_{m-2\lambda+1}-s_{m-2\lambda-1}) \\ +\sum_{r=2}^{M} p_r\,(l_{\lambda+1}-[s_{m-2\lambda-1}+s_{m-2\lambda}]) \end{aligned} \tag{3.7}$$

Substituting for $R_o - R_l$, we get:

$$\begin{aligned} R_o - R_l &\geq -l_{\lambda+1}+s_{m-2\lambda-1}+s_{m-2\lambda} \\ &\quad + p_1(s_{m-2\lambda+1}-s_{m-2\lambda-1}) \\ &\quad + \sum_{r=2}^{M} p_r\,(l_{\lambda+1}-[s_{m-2\lambda-1}+s_{m-2\lambda}]) \\ &\geq p_1\,[-l_{\lambda+1}+s_{m-2\lambda-1}+s_{m-2\lambda}+s_{m-2\lambda+1}-s_{m-2\lambda-1}] \\ &\geq p_1\,[-l_{\lambda+1}+s_{m-2\lambda}+s_{m-2\lambda+1}] \\ &> 0 \quad (\because l_{\lambda+1} < s_{m-2\lambda}+s_{m-2\lambda+1}) \end{aligned}$$

Case 2: $R_o - R_s \geq 0$ can be similarly proved:

$$\begin{aligned} R_o - R_s &= \sum_{i=1}^{\lambda} l_i + \sum_{j=1}^{m-2\lambda} s_j + V_1(\lambda+1,m-2\lambda+1) \\ &\quad - \left[\sum_{i=1}^{\lambda-1} l_i + \sum_{j=1}^{m-2\lambda+2} s_j + V_1(\lambda,m-2\lambda+3)\right] \\ &= l_\lambda - (s_{m-2\lambda+1}+s_{m-2\lambda+2}) \\ &\quad + V_1(\lambda+1,m-2\lambda+1) - V_1(\lambda,m-2\lambda+3) \end{aligned}$$

Using similar arguments as before,

$$\begin{aligned} V_1(\lambda+1,m-2\lambda+1)-V_1(\lambda,m-2\lambda+3) \\ \geq p_1(s_{m-2\lambda+1}-s_{m-2\lambda+3}) \\ +\sum_{r=2}^{M} p_r\,(-l_\lambda+[s_{m-2\lambda+1}+s_{m-2\lambda+2}]) \end{aligned}$$

After substituting for $R_o - R_s$, we get:

$$\begin{aligned} R_o - R_s &\geq l_\lambda - (s_{m-2\lambda+1} + s_{m-2\lambda+2}) \\ &\quad + p_1(s_{m-2\lambda+1} - s_{m-2\lambda+3}) \\ &\quad + \sum_{r=2}^{M} p_r\left(-l_\lambda + [s_{m-2\lambda+1} + s_{m-2\lambda+2}]\right) \\ &\geq p_1(l_\lambda - (s_{m-2\lambda+2} + s_{m-2\lambda+3})) \\ &\geq 0 \quad (\because l_\lambda \geq s_{m-2\lambda+2} + s_{m-2\lambda+3}) \end{aligned}$$

□

By setting the threshold on the *least* valuable large ad that can be scheduled, the scheduler only needs to check backwards from $l_{\lfloor m/2 \rfloor}$ for the least valuable large ad that satisfies **Case 1**. When the appropriate ad is found, all large ads that have greater value are scheduled, and the remaining time in the break is filled with the most valuable small ads.

Multiple Breaks

The multiple breaks case uses induction, and the intuition behind the proof is similar to that used in Lemma 3.5. We state the Theorem formally below.

Theorem 3.3. *If $b_1 = mS$ and $k = n$, then the Optimal Policy is to select $(l_1, \ldots, l_\lambda, s_1, \ldots, s_{m-2\lambda})$, where:*

- $l_\lambda \geq s_{m-2\lambda+n} + s_{m-2\lambda+n+1}$
- $2\lambda \leq m$

Proof. Proof is given in Appendix A. □

Consider the case where $k = 1$, and the break of size mS is split into $\lfloor \frac{m}{2} \rfloor$ breaks of size $2S$ (and an additional break of size S, if m is an odd number). Then by Theorem 3.1, l_1 would be compared to $(s_{m/2} + s_{m/2+1})$, and if it is lower, to $(s_{m/2+1} + s_{m/2+2})$, and so on until either l_1 is greater than some combination of $s_i + s_{i+1}$, where $i > m/2$, or the

large breaks are exhausted. If l_1 is not scheduled in the first $\lfloor \frac{m}{2} \rfloor - 1$ breaks, then l_1 will finally be compared to $(s_{m-1} + s_m)$, which corresponds to $(s_{m-2\lambda+1} + s_{m-2\lambda+2})$ where $\lambda = 1$.

To see why this is true for any l_λ where $\lambda > 1$, assume that it is true for $l_1, \ldots, l_{\lambda-1}$. If $l_{\lambda-1}$ has been selected for airing, then $l_{\lambda-1} \geq (s_{m-2\lambda+3} + s_{m-2\lambda+4})$. If $l_{\lambda-1}$ is scheduled for the *last* large break, then ads $(s_{m-2\lambda+3}, s_{m-2\lambda+4})$ are not scheduled, and the last small ads to be scheduled are $(s_{m-2\lambda+1}, s_{m-2\lambda+2})$. For l_λ to be an attractive candidate to be scheduled, therefore, it has to be more valuable than the two least valuable small ads which *have* been selected: $(s_{m-2\lambda+1}, s_{m-2\lambda+2})$.

When $k > 1$, the index of small ads that l_i has to be compared against increases by exactly k, because the optimal policy assumes the worst case where every break subsequent to the current one is small, similar to the intuition in Section 3.2.1.

From a managerial perspective, the Optimal Policy reduces complexity; the Greedy Policy would have to generate every combination of ads that fits the break and select the most profitable. Thus, despite the added complexity of having breaks of multiple sizes, the Optimal Policy scales well.

3.2.4 Stochastic Number of Breaks of Multiple Sizes

In this section, we study the Optimal Policy when break sizes are distributed between $[0, S, 2S, \ldots, MS]$. The size of the current break is assumed to be mS, where $m \leq M$. As we did in Section 3.2.2, we allow breaks of size zero to model the case where the number of breaks is stochastic. We use probability p_i to denote the probability of a break of size i occurring, where $i \in [0, M]$ and $\sum_{i=0}^{M} p_i = 1$.

We study the Optimal Policy when the number of breaks remaining are one and two, and use induction to prove the Optimal Policy for the multiple break case.

One Break

When the current break is the only break to be scheduled, and the size of the break is known, we use the same policy as outlined in Lemma 3.4. We reiterate the lemma here

without proof.

Lemma 3.6. *If $b_1 = mS$, $p_0 \geq 0$, and $k = 1$, then the Optimal Policy is to select $(l_1, \ldots, l_\lambda, s_1, \ldots, s_{m-2\lambda})$, where λ is the largest index such that $2\lambda \leq m$ and:*

$$l_\lambda \geq s_{m-2\lambda+1} + s_{m-2\lambda+2}$$

Two Breaks

We know that Lemma 3.6 applies for b_2 when it is the only break remaining. We will use induction to prove optimality when two breaks remain.

As we did for Lemma 3.5, we prove that it is optimal to select l_λ when it satisfies the rule for optimality, from which we can infer optimality of selecting all $l_i \geq l_\lambda$.

Lemma 3.7. *If $b_1 = mS$, $p_0 \geq 0$, and $k = 2$, then the Optimal Policy is to select $(l_1, \ldots, l_\lambda, s_1, \ldots, s_{m-2\lambda})$, where λ is the largest index such that $2\lambda \leq m$ and:*

$$(p_0 + p_1) l_\lambda \geq p_0(s_{m-2\lambda+1} + s_{m-2\lambda+2}) + p_1(s_{m-2\lambda+2} + s_{m-2\lambda+3})$$

Proof. Let:

$$\hat{O} = (l_1, \ldots, l_\lambda, s_1, \ldots, s_{m-2\lambda})$$
$$O_l = (l_1, \ldots, l_\lambda, l_{\lambda+1}, s_1, \ldots, s_{m-2\lambda-2})$$
$$O_s = (l_1, \ldots, l_{\lambda-1}, s_1, \ldots, s_{m-2\lambda+2})$$

Let R_o, R_l, and R_s denote the revenues earned by selecting $\hat{O}$, O_l and O_s respectively. We have to prove that

1. $R_o \geq R_l$
2. $R_o \geq R_s$

We again use $V_i(l, s)$ to denote expected revenue earned when selecting ads for break i onwards, when the indexes of the first large ad and the first small ad are l and s respectively.

Case 1: By definition:

$$(p_0+p_1)l_\lambda \geq p_0(s_{m-2\lambda+1}+s_{m-2\lambda+2})+p_1(s_{m-2\lambda+2}+s_{m-2\lambda+3}) \tag{3.8}$$

$$\text{and } (p_0+p_1)l_{\lambda+1} < p_0(s_{m-2\lambda-1}+s_{m-2\lambda})+p_1(s_{m-2\lambda}+s_{m-2\lambda+1}) \tag{3.9}$$

Then,

$$\begin{aligned} R_o - R_l &= \sum_{i=1}^{\lambda} l_i + \sum_{j=1}^{m-2\lambda} s_j + V_2(\lambda+1, m-2\lambda+1) \\ &\quad - \left[\sum_{i=1}^{\lambda+1} l_i + \sum_{j=1}^{m-2\lambda-2} s_j + V_2(\lambda+2, m-2\lambda-1)\right] \\ &= -l_{\lambda+1} + s_{m-2\lambda-1} + s_{m-2\lambda} \\ &\quad + V_2(\lambda+1, m-2\lambda+1) - V_2(\lambda+2, m-2\lambda-1) \end{aligned}$$

From eq. 3.9, we have:

$$\begin{aligned} (p_0+p_1)l_{\lambda+1} &< p_0(s_{m-2\lambda-1}+s_{m-2\lambda})+p_1(s_{m-2\lambda}+s_{m-2\lambda+1}) \\ &< p_0(s_{m-2\lambda-2}+s_{m-2\lambda-1})+p_1(s_{m-2\lambda-1}+s_{m-2\lambda}) \\ \implies (p_0+p_1)l_{\lambda+2} &< p_0(s_{m-2\lambda-2}+s_{m-2\lambda-1})+p_1(s_{m-2\lambda-1}+s_{m-2\lambda}) \\ &\quad (\text{since } l_{\lambda+2} \leq l_{\lambda+1}) \end{aligned}$$

Then by induction, for $b_2 \geq 2S$, $V_2(\lambda+2, m-2\lambda-1)$ earns us $(s_{m-2\lambda-1}+s_{m-2\lambda})$ followed by the sum of values of ads selected from the set $\{l_{\lambda+2}, \ldots, s_{m-2\lambda+1}, \ldots\}$. On the other hand, with $V_2(\lambda+1, m-2\lambda+1)$ we earn *at least* $l_{\lambda+1}$ followed by the sum of values of ads from the set $\{l_{\lambda+2}, \ldots, s_{m-2\lambda+1}, \ldots\}$. Thus, we can write:

$$\begin{aligned} V_2(\lambda+1, m-2\lambda+1) - V_2(\lambda+2, m-2\lambda-1) \\ \geq p_1(s_{m-2\lambda+1} - s_{m-2\lambda-1}) \\ + \sum_{i=2}^{M} p_i\,(l_{\lambda+1} - [s_{m-2\lambda-1}+s_{m-2\lambda}]) \end{aligned}$$

Substituting for $R_o - R_l$, we get:

$$\begin{aligned}
R_o - R_l &\geq -l_{\lambda+1} + s_{m-2\lambda-1} + s_{m-2\lambda} \\
&\quad + p_1(s_{m-2\lambda+1} - s_{m-2\lambda-1}) \\
&\quad + \sum_{i=2}^{M} p_i\,(l_{\lambda+1} - [s_{m-2\lambda-1} + s_{m-2\lambda}]) \\
&\geq (p_0 + p_1)\,[-l_{\lambda+1} + s_{m-2\lambda-1} + s_{m-2\lambda}] + p_1\,[s_{m-2\lambda+1} - s_{m-2\lambda-1}] \\
&\geq -(p_0 + p_1) l_{\lambda+1} + p_0(s_{m-2\lambda-1} + s_{m-2\lambda}) + p_1(s_{m-2\lambda} + s_{m-2\lambda+1}) \\
&> 0 \quad \text{(from eq. 3.9)}
\end{aligned}$$

Case 2 $R_o - R_s \geq 0$ can be similarly proved using eq. 3.8. □

It can be seen that the intuition behind Lemma 3.7 is similar to our discussion in Section 3.2.3: the revenue earned by the Optimal Policy is equal to the case where we have $\lfloor m/2 \rfloor$ breaks of size $2S$ (and one break of size S, if m is an odd number). Since $p_0 \geq 0$, Theorem 3.2 would apply for each break of size $2S$.

The intuition behind the probabilities follows the discussion in Section 3.2.2. The Optimal Policy adjusts the threshold above which large ads are attractive based on the break size distribution, and these thresholds increase as p_0 increases and other probabilities decrease.

Thus the case of multiple break sizes with stochastic number of breaks can be seen as a combination of Sections 3.2.2 and 3.2.3.

Multiple Breaks

The multiple breaks case uses induction, and the intuition behind the proof is similar to that of Lemma 3.7. We state the Theorem formally below.

Theorem 3.4. *If $b_1 = mS$, $p_0 \geq 0$, and $k = n$, then the Optimal Policy is to select $(l_1, \ldots, l_\lambda, s_1, \ldots, s_{m-2\lambda})$, where λ is the largest index such that $2\lambda \leq m$ and:*

$$(p_0 + p_1)^{n-1} l_\lambda \geq \sum_{i=0}^{n-1} \left[\binom{n-1}{i} p_0^{n-1-i} p_1^i (s_{m-2\lambda+i+1} + s_{m-2\lambda+i+2}) \right]$$

Proof. Proof is given in Appendix A □

As discussed in section 3.2.2, the threshold has once again increased for l_1; this is because of the introduction of breaks of size zero. Our insights from Section 3.2.2 still apply. The Optimal Policy sets a threshold, based on the break size distribution, that each large ad should exceed to be scheduled. As we have studied, the higher the probability of breaks of size zero, the higher the value that large ads should have to make them attractive candidates for scheduling, given the same set of small ads.

In this section, we have studied the Optimal Policy for scheduling ads in breaks of stochastic sizes, when the size of the break for which we are currently scheduling is known but those of subsequent breaks is not. We conclude with the most general case, Theorem 3.4 where setting M and p_0 to appropriate values will give us the rules described in Theorem 3.1, Theorem 3.2 and Theorem 3.3. Thus we have described the rule for the full set of scenarios when the scheduler is aware of the size of the break to be scheduled next.

In the next section, we study the Optimal Policy for cases when the scheduler does not know the size of any of the breaks, but knows only the distribution of break sizes based on which he can create a sequence of ads to be scheduled.

3.3 Unknown Break Size

In this section we study scenarios where we do not know the size of any of the breaks at the time of scheduling. Our motivation arises from cases where the break begins without the on-field director being able to advice the schedulers on what the break length is expected to be, and an ad schedule has to be made based only on the break length distribution. We assume that break sizes are IID.

As before, we discuss Optimal Policy for the four cases mentioned in section 3.1.

3.3.1 Base Case

In the Base Case, breaks can be of two sizes, small (S) and large ($L = 2S$). A small break occurs with probability p, and a large break occurs with probability $1 - p$. We study our selection options when a choice has to be made between selecting either (s_1, s_2) or l_1 before the break size is observed.

One break

When scheduling for one break unknown size, we choose the maximum of the expected revenues from selecting either l_1 or (s_1, s_2). If a small break occurs, selecting (s_1, s_2) earns s_1, since only s_1 can be completely aired. Selecting l_1 earns nothing, since it cannot be fully aired. If a large break occurs, selecting (s_1, s_2) and l_1 earn $s_1 + s_2$ and l_1 respectively.

Therefore the expected revenue from selecting (s_1, s_2) is $ps_1 + (1-p)(s_1 + s_2)$, and the expected revenue from selecting l_1 is $(1-p)l_1$.

We state the following lemma without proof:

Lemma 3.8. *If $k = 1$, and the break size is unknown, the Optimal Policy is to:*

1. *select l_1 if $(1-p)l_1 \geq ps_1 + (1-p)(s_1 + s_2)$*
2. *select (s_1, s_2) otherwise*

We next look at the Optimal Policy when two breaks remain.

Two breaks

Unlike Lemma 3.2, where the Optimal Policy compared l_1 and $s_n + s_{n+1}$, the absence of *ex-ante* information forces us to choose myopically between the expected revenues earned by selecting l_1 and (s_1, s_2) irrespective of the number of breaks remaining. We state the optimal policy formally as follows.

Lemma 3.9. *When $k = 2$, and the break size is unknown, the optimal policy is to:*

1. *select l_1 if $(1-p)l_1 \geq ps_1 + (1-p)(s_1 + s_2)$*

2. *select* (s_1, s_2) *if* $(1-p)l_1 < ps_2 + (1-p)(s_2+s_3)$

3. *select either* l_1 *or* (s_1, s_2) *otherwise*

Proof. From Lemma 3.8, the rule holds good for one break. Let R_l and R_s denote the expected revenues from scheduling l_1 and (s_1, s_2) respectively. Let $V_i(l,s)$ denote the revenue earned from break i onwards, when the indexes of the first large ad and the first small ad are l and s respectively.

Case 1: When $(1-p)l_1 \geq ps_1 + (1-p)(s_1+s_2)$,

$$\begin{aligned}
R_l - R_s &= p(0 + V_2(1,1)) + (1-p)(l_1 + V_2(2,1)) \\
&\quad - p(s_1 + V_2(1,2)) - (1-p)(s_1 + s_2 + V_2(1,3)) \\
&= (1-p)l_1 - ps_1 - (1-p)(s_1+s_2) \\
&\quad + p[V_2(1,1) - V_2(1,2)] + (1-p)[V_2(2,1) - V_2(1,3)] \\
&\geq (1-p)l_1 - ps_1 - (1-p)(s_1+s_2) \\
&\quad + p[(1-p)l_1 - (1-p)l_1] \\
&\quad + (1-p)[p(s_1) + (1-p)(s_1+s_2) - (1-p)l_1] \\
&\qquad (\because V_2(2,1) \geq p(s_1) + (1-p)(s_1+s_2)) \\
&\geq p[(1-p)l_1 - ps_1 - (1-p)(s_1+s_2)] \\
&\geq 0
\end{aligned}$$

$R_l - R_s \geq 0 \implies$ it is optimal to select l_1 first.

Case 2: When $(1-p)l_1 < ps_2 + (1-p)(s_2+s_3)$

$$\begin{aligned}
R_s - R_l &= ps_1 + (1-p)(s_1+s_2) - (1-p)l_1 \\
&\quad + p[V_2(1,2) - V_2(1,1)] + (1-p)[V_2(1,3) - V_2(2,1)] \\
&\geq ps_1 + (1-p)(s_1+s_2) - (1-p)l_1 + p[(1-p)l_1 - ps_1 - (1-p)(s_1+s_2)] \\
&\quad + (1-p)[(1-p)(l_1) - ps_1 - (1-p)(s_1+s_2)]
\end{aligned}$$

$$
\begin{aligned}
&(\because V_2(1,2) \text{ and } V_2(1,3) \geq (1-p)(l_1)) \\
&\geq (1-p-(1-p))[ps_1+(1-p)(s_1+s_2)-(1-p)l_1] \\
&\geq 0
\end{aligned}
$$

$R_s - R_l \geq 0 \implies$ it is optimal to select (s_1, s_2) first.

Case 3: When $ps_1+(1-p)(s_1+s_2) > (1-p)l_1 \geq ps_2+(1-p)(s_2+s_3)$,

$$
\begin{aligned}
R_l - R_s &= (1-p)l_1 - ps_1 - (1-p)(s_1+s_2) \\
&\quad + p[V_2(1,1)-V_2(1,2)] + (1-p)[V_2(2,1)-V_2(1,3)] \\
&= (1-p)l_1 - ps_1 - (1-p)(s_1+s_2) \\
&\quad + p\Big[ps_1+(1-p)(s_1+s_2)-(1-p)l_1\Big] \\
&\quad + (1-p)\Big[ps_1+(1-p)(s_1+s_2)-(1-p)l_1\Big] \\
&= 0
\end{aligned}
$$

$R_l = R_s \implies$ we are indifferent between selecting $l_1$and (s_1, s_2). □

The Optimal Policy shows that there is a region where selecting l_1 is strongly preferable, a region where selecting (s_1, s_2) is strongly preferable, and a region where we are indifferent between selecting l_1 and (s_1, s_2). We shall look into the multiple break case before discussing the implications of such a partition.

Multiple breaks

Theorem 3.5. *When $k = n$, and the break size is unknown, the Optimal Policy is to:*

1. *select l_1 if $(1-p)l_1 \geq ps_1+(1-p)(s_1+s_2)$*
2. *select (s_1, s_2) if $(1-p)l_1 < ps_n+(1-p)(s_n+s_{n+1})$*
3. *select either l_1 or (s_1, s_2) otherwise*

Proof. Proof is given in Appendix A □

Theorem 3.5 extends Lemma 3.9. There are two regions where the scheduler has a strong preference over the possible choices, separated by a region of indifference.

In the first region (**Case 1**), l_1 is strongly preferable based on a myopic comparison of the expected revenues earned from selecting l_1 and (s_1, s_2). Since s_1 and s_2 are the *most* profitable small ads, l_1 earns a higher expected revenue than any combination of small ads to be selected.

The same can be said of the third region (**Case 2**); since expected revenue from selecting l_1 is lower than that from selecting (s_n, s_{n+1}), we select the smaller ads first. If we are faced with a series of only small breaks, we schedule $(s_1, \ldots, s_n)$ through the match, earning more than if we had scheduled l_1 first. If a large break were to arrive subsequently, the strategy would check if **Case 1** applies, earning at least as much as the Greedy Policy.

The indifference exists because when breaks can only be of size S or L, and $(1-p)l_1 \geq s_n + (1-p)s_{n+1}$, then the Optimal Policy expects to schedule l_1 when $(1-p)l_1$ is greater than some $(s_j + (1-p)s_{j+1})$, where $j > 1$, and **Case 1** applies.

In the worst case, suppose $(1-p)l_1 = s_n + (1-p)s_{n+1}$, and small ads have been scheduled for the first $n-1$ breaks, which are found to be short. For the n^{th} break, expected revenue from selecting l_1 will be compared to expected revenue from scheduling (s_n, s_{n+1}), and **Case 1** applies. Therefore l_1 is guaranteed to be selected for some break in the match.

Similarly, since selecting (s_1, s_2) earns higher expected revenue than *any* large ad, then in the worst case, l_1 is repeatedly selected for the first $n-1$ breaks which turn out to be short. Then for the n^{th} break, **Case 2** will apply and (s_1, s_2) will be selected. Thus (s_1, s_2) is also guaranteed to be selected for some break in the match.

Therefore we are indifferent between selecting l_1 and (s_1, s_2) for b_1.

From a managerial perspective, the region of indifference gives the network flexibility when accepting orders with client constraints at the start of the match. Consider a scenario where a client wishes the network to schedule a large ad when a particular wicket falls (and breaks in the match can only be of size S or L). The network can accept this scheduling constraint on a long ad as long as the expected value from airing

the ad is higher than the expected value of airing two small ads in their inventory which fall in the n^{th} and $(n+1)^{th}$ position, where n is the total number of breaks in the match. It is possible, however, that the above condition is satisfied, but the ad is not the most valuable large ad in the inventory when the break occurs. The network can still use the Optimal Policy and the break size distribution to quote a value to the client to make the ad a viable candidate for that break. The exact methodology is beyond the scope of our current discussion, however pricing policies based on the Optimal Policy is an area for future research.

3.3.2 Stochastic Number of Breaks

In this subsection we consider an extension where the number of breaks is stochastic. As in Section 3.2.2, we introduce a probability p_0 of having a break of size 0, while we assume that break could be of size S with probability p_1, and of size $L = 2S$ with probability p_2. We do not consider breaks of size greater than L.

As before, we assume that a break of size zero might arrive at any point in the match, and that we are always aware of the arrival of such a break, and discount the number of breaks remaining accordingly. All other assumptions and notations as listed in Section 3.1 still remain.

One break

With only one break possible, the scheduler chooses greedily between the expected values of selecting l_1 and (s_1, s_2). If the last break turns out to be of size zero, either strategy earns zero; if it is of size S, we earn zero with l_1 and s_1 with (s_1, s_2); if it is of size L we earn l_1 and $s_1 + s_2$ respectively.

As before, we propose a lemma without proof for the one break case:

Lemma 3.10. *When $k = 1$, $p_0 \geq 0$, and break sizes are unknown, the Optimal Policy is to:*

1. *select l_1 if $p_2 l_1 \geq p_1 s_1 + p_2(s_1 + s_2)$*

2. *select* (s_1, s_2) *otherwise*

Two breaks

For the two break case, the strategy continues to be a myopic choice between the expected revenues earned from selecting l_1 and (s_1, s_2), despite the introduction of breaks of size zero.

Lemma 3.11. *When* $k = 2$, $p_0 \geq 0$, *and break sizes are unknown, the Optimal Policy is to:*

1. *select* l_1 *if* $p_2 l_1 \geq p_1 s_1 + p_2(s_1 + s_2)$

2. *select* (s_1, s_2) *otherwise*

Proof. Let $p_2 l_1 = t + p_1 s_1 + p_2(s_1 + s_2)$. We use R_l, R_s, and $V_i(l, s)$ as defined in the previous subsections. From Lemma 3.10, the rule holds good for one break.
With the introduction of p_0, expected revenues earned are as follows:

$$R_l = (p_0 + p_1)V_1(1,1) + p_2(l_1 + V_1(2,1))$$
$$R_s = p_0 V_1(1,1) + p_1(s_1 + V_1(1,2)) + p_2(s_1 + s_2 + V_1(1,3))$$

Case 1: When $(1-p)l_1 \geq p s_1 + (1-p)(s_1 + s_2)$,

$$\begin{aligned}
R_l - R_s &= (p_0 + p_1)V_1(1,1) + p_2(l_1 + V_1(2,1)) \\
&\quad - p_0 V_1(1,1) - p_1(s_1 + V_1(1,2)) - p_2(s_1 + s_2 + V_1(1,3)) \\
&= p_1(V_1(1,1) - s_1 - V_1(1,2)) + p_2(l_1 + V_1(2,1) - (s_1 + s_2) - V_1(1,3)) \\
&= t + p_1[V_1(1,1) - V_1(1,2)] + p_2[V_1(2,1) - V_1(1,3)] \\
&\geq t + p_1[p_1 s_1 + p_2(s_1 + s_2) - p_2 l_1] + p_2[p_1 s_1 + p_2(s_1 + s_2) - p_2 l_1] \\
&\geq t + p_1(-t) + p_2(-t)
\end{aligned}$$

$$\geq p_0 t$$
$$\geq 0 \because p_0 \geq 0,\ t \geq 0$$

Thus, $R_l - R_s \geq 0 \implies$ it is optimal to selection l_1 first.

Case 2: When $(1-p)l_1 < ps_1 + (1-p)(s_1+s_2)$, let $t + p_2 l_1 = p_1 s_1 + p_2(s_1+s_2)$, where $t > 0$.

Proceeding along similar lines as **Case 1**, we can prove that $R_s - R_l \geq p_0 t \geq 0$. □

Case 1 and **Case 2** correspond with the cases in Lemma 3.9. The addition of p_0 does not affect the two regions of strong preference because the expected revenue when the break is of size zero is the same whether we selection l_1 or (s_1, s_2). Therefore, our decision is solely based on p_1 and p_2, and the expected revenues earned thereby.

There is no **Case 3** corresponding to Lemma 3.8 because S and L are not the only break sizes possible. When the expected revenue from selecting l_1 is less than that from selecting (s_1, s_2), **Case 2** applies. From the proof for Lemma 3.11, the difference between the two expected revenues is at least $p_0 t$, where $t \geq 0$ is the difference in expected revenue. Since $p_0 > 0$, we are indifferent between scheduling l_1 and (s_1, s_2) only when $t = 0$; i.e. $p_2 l_1 = p_1 s_1 + p_2(s_1+s_2)$.

Multiple breaks

From Lemmas 3.10 and 3.11, we can see that the myopic rule holds good when we plan for either one break or two. In this section, we use the previous results and prove by induction that the rule holds good for any number of breaks remaining.

Theorem 3.6. *When $k = n$, $p_0 \geq 0$, and break sizes are unknown, the Optimal Policy is to:*

1. *select l_1 if $p_2 l_1 \geq p_1 s_1 + p_2(s_1+s_2)$*
2. *select (s_1, s_2) otherwise*

Proof. Proof is given in Appendix A □

Theorem 3.6 can be seen as a special case of Theorem 3.5 where $p_0 > 0$. As before, we have two regions where the scheduler has strong preferences, but there is no region of indifference (except for the point where expected revenue from selecting l_1 equals that from selecting (s_1, s_2)). Therefore, when the number of breaks is not fixed and break sizes are not known in advance, the scheduler is forced to select between l_1 and (s_1, s_2) myopically based on the expected revenues earned.

3.3.3 Multiple Break Sizes and its variants

In this section, we consider the case where breaks can be of size $(S, 2S, 3S, ..., MS)$, and the size of the break is not known in advance.

As shown in Theorem 3.5 when breaks are of sizes S or $2S$, the Optimal Policy when break sizes are unknown is based on a myopic comparison of the expected revenues earned. When the maximum break size is $2S$, we choose between l_1 and (s_1, s_2). This policy is independent of the number of breaks remaining, since breaks are IID and break size is unknown for each break.

When the maximum break size is MS, the Optimal Policy should provide the *permutation* of ads to be scheduled based on the break size distribution. Consider, for example, the case when $M = 3$. Let us assume that the probability of breaks of sizes S,$2S$ and $3S$ occurring is p_1,p_2 and p_3, where $p_1 + p_2 + p_3 = 1$. Then the ad schedules that can be generated and the revenues earned are shown in Table 3.2.

Schedule	Revenue earned
s_1, s_2, s_3	$s_1 + (p_2 + p_3)s_2 + p_3 s_3$
s_1, l_1	$s_1 + p_3 l_1$
l_1, s_1	$(p_2 + p_3)l_1 + p_3 s_1$

Table 3.2: Revenues earned with each possible schedule

As shown, the revenue earned with each schedule is different, and the Optimal Policy should select the schedule which generates the maximum revenue based on the probabilities p_i and the values of the ads scheduled.

It can be shown that the number of possible combinations of ads for each value of M is a Fibonacci sequence as shown in table 3.3.

M	Number of ad schedules
1	1
2	2
3	3
4	5
5	8
6	13
7	21
⋮	⋮

Table 3.3: Number of possible ad schedules for each value of M

This can be explained as follows: let us assume that each break is divided into 'slots' of length S, thus when $M = n$, there are n slots available to be scheduled. Let σ_n and σ_{n+1} be the number of ad schedules possible when $M = n$ and $M = n+1$ respectively. Then for $M = n+2$, an additional slot is added at the end of $n+1$ slots, and this slot can either be programmed with an ad of size S or an ad of size $L = 2S$, starting from slot $n+1$. If an ad of size S is scheduled in slot $n+2$, the previous $n+1$ slots can be scheduled in σ_{n+1} ways. If, however, an ad of size L is scheduled across slots $n+1$ and $n+2$, then the previous n slots can be scheduled in σ_n ways, thus giving $\sigma_{n+2} = \sigma_n + \sigma_{n+1}$.

By the well known Binet's formula[1], when $M = n$, the number of possible combinations $\Phi(n)$ is

$$\Phi(n) = \frac{\varphi^n - (1-\varphi)^n}{\sqrt{5}} = \frac{\varphi^n - (-1/\varphi)^n}{\sqrt{5}}$$

Where $\varphi = \frac{1+\sqrt{5}}{2} \approx 1.6180339887\ldots$

The number of ad schedules that can be programmed increases exponentially with the value of M, resulting in a 'Hughes effect', or a 'curse of dimensionality' [2], a classical problem that arises when dealing with problems of stochastic recourse. For any $M = n$, the Optimal Policy would be the maximum of the expected revenues earned from each of the $\Phi(n)$ schedules. As discussed in Section 2.3, the problem is better solved with specialized heuristics, rather than attempting an analytical solution for the general n- case.

[1] Theory of Binet formulas for Fibonacci and Lucas p-numbers[25]

[2] Dynamic programming[2]

Chapter 4

Numerical Analysis

In this chapter we perform a numerical analysis of the Optimal Policy discussed in Chapter 3. We compare the Optimal Policy to the Greedy Policy, in order to find the conditions under which the Optimal Policy most outperforms the Greedy Policy. We also study the impact of service level commitments and the impact of uncertainty on the performance of the Optimal Policy.

4.1 Performance with deterministic number of breaks

We begin with cases where the number of breaks is fixed, and there are no breaks of size zero. A study of how the Optimal Policy performs as parameters change will give us an idea of how stable it is, and allow us to find conditions where it is most beneficial to use the Optimal Policy.

We begin with listing the parameters we will use for the study, and subsequently study the impact of various parameters on revenues and service levels.

4.1.1 Parameters

To evaluate the performance of the policies, we generate the values for the large and small ads based on the parameters listed in Table 4.1.

Results were averaged over 500 runs, and at each iteration the inventory and spot

values were changed. This was done to ensure that we tested the strategies across a broad range of data sets, and that the results were representative of average scenarios.

Parameter	Value
Number of breaks	50
Break size (s)	10 sec or 20 sec
Number of large ads	50
Number of small ads	100
Large ads values ($)	1000 – 1200
Small ads values ($)	200 – 1000

Table 4.1: Parameters for numerical analysis of Optimal Policy and Greedy Policy

In Table 4.1, the number of breaks, 50, is typical of the number of breaks found in a T-20 match, where each of two innings has twenty over breaks, and ten wickets are expected to fall during the course of the match. We assume that breaks can be of sizes 10 seconds or 20 seconds only with equal probability, hence the mean break length is 15 seconds. The range of values earned by long ads and short ads are typical of orders received by major sports broadcasters for international T-20 tournaments, and we select random values within these ranges.

We next look at performances of the Optimal Policy and Greedy Policy when air time sold (i.e., number of ads available in the inventory) varies.

4.1.2 Impact of variation in air time sold

We analyze how change in the amount of air time sold affects the performance of the Optimal Policy and the Greedy Policy. We start with 50 large ads and 100 short ads, as listed in Table 4.1, and remove two short ads for every large ad removed from the inventory, to keep the ratio of air time between large and small ads constant. Air time sold ranged from half the expected air time over the course of a match, to more than twice.

Cricket broadcasters oversell air time for important tournaments, particularly those that involve India, and expect to make good the ads not shown during live broadcast in non-live segments later. By overselling, networks have more flexibility in what ads

they show, and are able to cash in on opportunities when actual break time in matches exceeds the expected break time significantly.

Underselling air time is usually done when the penalty of not achieving service levels is severe, or the network expects a curtailed match, for instance, due to rain. By decreasing the air time sold, networks often aim to provide higher service levels, while putting a premium on the spot value per second. We observe changes in service levels of large and small ads, along with change in revenues earned for each policy as the amount of air time sold is varied.

The result of the numerical analysis when lengths of breaks and their number are known at the time of scheduling is summarized in Table 4.2.

When air time sold is close to the expected air time available ($\sim 750s$), we see that the Optimal Policy outperforms the Greedy Policy by almost 3%. This translates to an average of $1262 per match, equivalent to the expected value earned by airing two small ads more per match than the Greedy Policy.

Service levels for the Optimal Policy and the Greedy Policy show that the Optimal Policy consistently schedules more large ads than then Greedy Policy does, whereas the Greedy Policy relies more on small ads. As a consequence, we see that when the air time sold is 800s, approximately 97% of the small ads have been shown by the Greedy Policy, yet almost one small break in a 50 break match has *nothing* scheduled in it. This is because the Greedy Policy schedules small ads even for large breaks early in the match and runs out of ads to schedule when small breaks occur.

This is an important result for broadcast networks. The over reliance of the Greedy Policy on small ads to earn revenue may lead to lost opportunities, whereas the Optimal Policy schedules large ads whenever possible and holds a reserve of small ads for small breaks, leading to improved service levels and revenues overall.

We plot the revenues earned by the Optimal Policy and the Greedy Policy against air time sold in Figure 4.1. The difference in revenues earned is pronounced when the service level is between 80% and 90%, which corresponds to airtime sold of around 800 seconds (from Table 4.2), which is roughly equal to the expected air time. The service level mark of 80% is significant, since this is the service level usually promised by

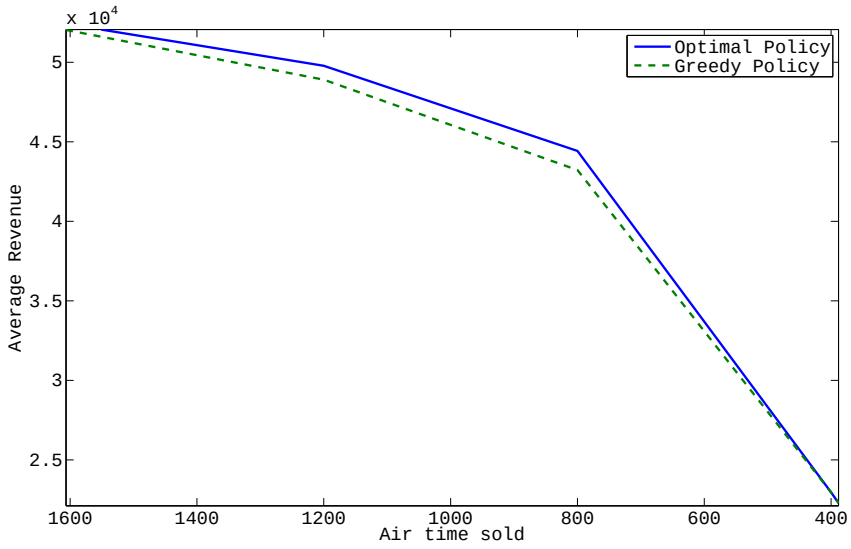

Figure 4.1: Average total revenue versus air time sold

the network to advertisers during most tournaments. Thus the Optimal Policy is significantly better than the Greedy Policy in conditions that approach real world situations.

We next consider the case where the number of breaks in a match is not fixed. The maximum difference between the Optimal Policy and the Greedy Policy occurs when the time sold is 800 seconds, which corresponds to 40 small ads and 10 large ads sold. Since the number of breaks is 50, the Greedy Policy schedules the small ads in the large breaks and is left with almost one small break left unscheduled (Table 4.2). When airtime sold is higher, both policies have a greater choice of ads to choose from and the difference between policies reduces; and when the airtime sold is lower, both policies suffer from a lack of ads equally.

The efficient frontier helps managers determine what service level is most optimal. Promising lower service levels can yield higher revenues, which should be balanced, however, with the possible loss of goodwill. Managers can thus decide on a target service level by considering both the benefits and costs involved.

4.1.3 Value of flexibility

Small ads can be shown in both small and large breaks, whereas large ads can only be shown in large breaks. Thus, small ads offer more flexibility to the broadcaster as to

which break they can be shown in. To analyze the value of flexibility, we study the change in revenue as we change the mix of small and large ads in the inventory. We begin with the parameters as outlined in Table 4.1, and split the large ads randomly into two, thus creating equally valued small ads, and increasing the ratio of small ads to large ads in the inventory.

We begin with a numerical analysis when the number of breaks per match is fixed. Table 4.3 shows the average revenue earned per match by the Optimal Policy and the Greedy Policy as the mix of large and small ads is varied and the number of breaks is fixed.

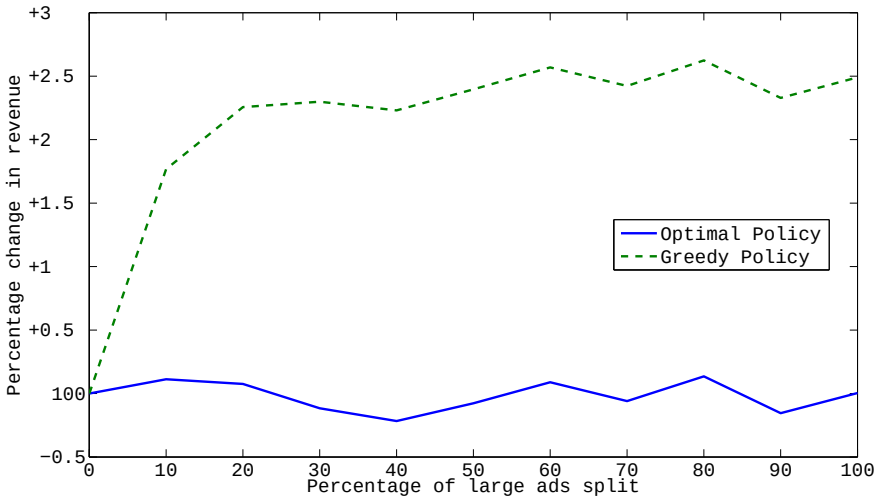

Figure 4.2: Change in revenue with split of large ads

The percentage change in revenue for each policy is the amount of change that each policy earns in comparison to the case when the ratio of air time is equally split between small ads and large ads, which is our starting ratio. The last column lists the percentage difference in revenues between the Greedy Policy and the Optimal Policy. We see that as the ratio of small ads to large ads increases, the difference between the two policies decreases. We plot the percentage change of the Optimal Policy and the Greedy Policy in Figure 4.2. The Optimal Policy does not vary much from its original value, whereas the Greedy Policy displays an increase of almost 2.5% from its original.

The increase in the revenue earned by the Greedy Policy is explained by the preference of the Greedy Policy for small ads: as discussed in section 4.1.2, the Greedy

Policy schedules more small ads than long ads, therefore revenues increase when the number of small ads increase.

From a managerial perspective, the ability of the Optimal Policy to maintain revenue earned despite substantial change in the ratio of small and large ads is of importance. Thus the Optimal Policy is a robust strategy despite changing inventory mix, and ensures the network a stable revenue regardless of the inventory composition.

Figure 4.3 plots the difference between the Optimal Policy and the Greedy Policy. We see that the difference is most significant when the air time is equally divided between the large ads and the small ads, and this difference decreases as the proportion of small ads increases. As expected, when the inventory consists only of small ads, there is no difference in revenue earned between the Optimal Policy and the Greedy Policy.

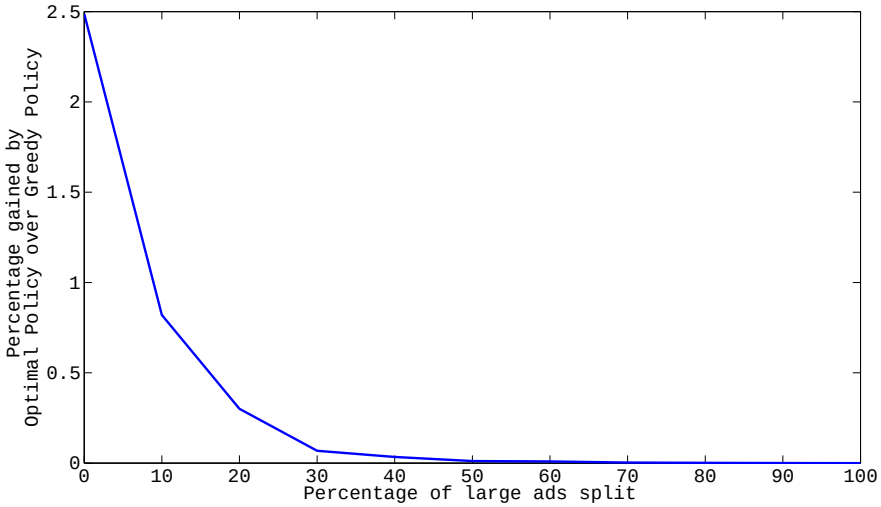

Figure 4.3: Difference between Greedy and Optimal with split of large ads

4.1.4 Impact of Variability

We next investigate the impact of variability on the performance of the Optimal Policy, as defined in Theorem 3.3. The distribution of break lengths and the size of our inventory are listed in Table 4.4, and all other parameters are the same as listed in Table 4.1.

To simulate variability, we use a Uniform Distribution with a mean of 60 seconds Although the break lengths were generated from a Uniform Distribution, they

are rounded down to the nearest multiple of 10 seconds, which is the size of the small ad. Large ads are of size 20 seconds, as before. We vary the support for the break size distribution from a constant 60 to $[10, 110]$. This corresponds to varying the standard deviation between $[0, \frac{50}{\sqrt{3}}]$, and the range between $[0, 100]$. As before, however, break sizes that are not a multiple of 10 are rounded down to the nearest multiple of 10, since the remaining break time will remain unutilized. For simplicity, we will only consider the range when discussing variability. Finally, we note that the expected air time has increased substantially, necessitating an increase in the size of our inventory as shown in Table 4.4.

The results have been tabulated in Table 4.5. We find that both the Optimal Policy and the Greedy Policy are not affected significantly by increase in variability. The 'percentage change' row for each policy shows the change in revenues earned compared to zero variability case. In this case, we see that there is negligible change in the revenues earned by the Optimal Policy and the Greedy Policy as variability increases. Service levels also do not show significant changes with variability, thus supporting our inference that the effect of variability on revenues earned is negligible. Figure 4.4 shows the percentage change in revenues with increase in range.

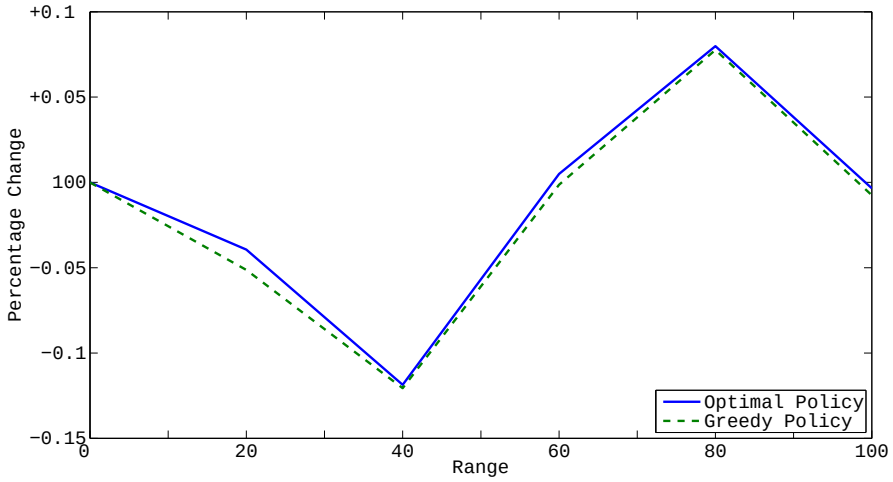

Figure 4.4: Change in revenue with variability

From a managerial perspective, as long as the expectation of break lengths is steady, increased variability does not significantly affect revenues. As variability increases,

breaks of large sizes are complemented by breaks of small sizes, keeping the overall revenues earned from fluctuating. The order in which large breaks and small breaks occur causes the small gains for the Optimal Policy over the Greedy Policy. The Greedy Policy has a preference for scheduling small ads in the large breaks, so when matches have a sequence of large breaks followed by small ones, the Greedy Policy is left with less valuable small ads for the small breaks. In contrast, the Optimal Policy has a lower threshold for scheduling large ads as discussed in section 3.2.3, therefore it is able to gain more from the short breaks in the latter part of the match.

In this chapter, we studied the behavior of the Optimal Policy numerically, and derived insights from the results. In the next chapter, we will discuss specialized algorithms and heuristics that will attempt to solve harder problems, such as having ads of multiple lengths with diversity constraints.

Air time sold (sec)			2000	1600	1200	800	400
Revenue	OP		54,935.46	52,349.08	49,523.60	44,489.78	23,034.12
	GP		54,733.86	51,927.26	48,673.94	43,227.58	23,034.12
	% Difference		0.37	0.81	1.72	2.84	0.00
Service Levels (%)	OP	*L* ads	22.52	39.26	67.15	99.44	100.00
		S ads	52.37	54.42	57.92	87.34	100.00
	GP	*L* ads	16.50	29.31	50.35	88.11	100.00
		S ads	58.40	64.37	74.72	96.62	100.00
Unused Breaks	OP	*L* breaks	-	-	-	0.03	14.64
		S breaks	-	-	-	0.01	5.39
	GP	*L* breaks	-	-	-	0.02	10.70
		S breaks	-	-	-	0.84	13.15

Table 4.2: Revenues and Service levels with change in air time sold
(*L*: large, *S*: small, OP: Optimal Policy, GP: Greedy Policy)

Long ads split (%)	Optimal Policy		Greedy Policy		Difference (%)
	Average Value	% change	Average Value	% change	
0	48,304	0	47,132	0	2.43
10	48,358	+0.11	47,965	+1.77	0.81
20	48,340	+0.08	48,195	+2.26	0.30
30	48,248	-0.17	48,215	+2.30	0.07
40	48,199	-0.22	48,183	+2.23	0.03
50	48,267	-0.08	48,261	+2.40	0.01
60	48,347	+0.09	48,343	+2.57	0.01
70	48,275	-0.06	48,274	+2.42	0
80	48,369	+0.14	48,369	+2.62	0
90	48,229	-0.15	48,229	+2.33	0
100	48,306	+0.00	48,306	+2.49	0

Table 4.3: Value of Flexibility

Parameter	Value
Break lengths	$\sim \mathscr{U}(\mu-\delta, \mu+\delta)$
	$\mu = 60$, $\delta \in [0, 50]$
Number of large ads	100
Number of small ads	200

Table 4.4: Parameters for analyzing impact of variability on Optimal Policy

Range			0	20	40	60	80	100
Average Revenue	OP		191,411	191,335	191,184	191,420	191,564	191,404
	% change		0.00	-0.04	-0.12	+0.01	+0.08	-0.003
	GP		191,411	191,313	191,180	191,408	191,559	191,396
	% change		0.00	-0.05	-0.12	-0.001	+0.08	-0.01
Service Levels (%)	OP	*L* ads	88.94	89.22	89.30	89.01	88.82	89.16
		S ads	61.06	60.78	60.70	60.99	61.18	60.84
	GP	*L* ads	88.94	87.90	88.84	88.08	88.34	88.49
		S ads	61.06	62.10	61.16	61.92	61.66	61.51

Table 4.5: Impact of variability
(*L*: large, *S*: small, OP: Optimal Policy, GP: Greedy Policy)

Chapter 5

Applications in Practice: Scheduling ads for Cricket

In previous chapters, we have studied the Optimal Policy for a stylized model of the real world problem. The motivation for this research came from our discussions with a major cricket broadcaster, who also provided us with real-world data, based on which we generated parameters for numerical analysis. We now test several scheduling heuristics under more constraints and present the analysis of the data, a study of the heuristics tested and the results obtained, and create relevant managerial insights.

5.1 Data Description

In order to have an estimate of the parameters and constraints in which ad scheduling was done, we received production logs of ads aired and the spot price for each of those ads during a T-20 tournament.

5.1.1 Break Lengths

Our findings are shown in Table 5.1, and the break length distribution is shown in Fig 5.1. The breaks recorded here were measured between the end of an over and the start of the next one, creating a slight skew towards the right (since actual time available

to show ads is shorter than the time between two such overs).

Parameter	Value(seconds)
Average break length	68.4
Standard Deviation	23
Minimum break length	10
Maximum break length	170
Number of samples (breaks)	983

Table 5.1: Analysis of break length data

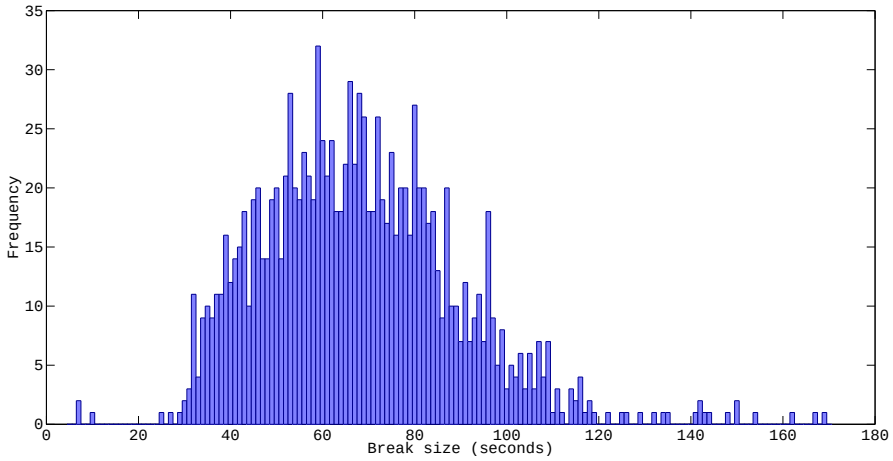

Figure 5.1: Break length distribution

For our numerical simulation, we estimated the break length to have a mean of 40 seconds and to be uniformly distributed between 10 seconds and 70 seconds.

For our study of T-20 matches, we estimated a total of 50 breaks per match (40 breaks in between overs and 10 wicket breaks). As a simplification of match conditions, we set the number of breaks to be 50 while estimating the performance of our heuristic.

5.1.2 Ad Lengths, Service Levels and Demands

The lengths of the ads contracted from clients was given directly by the sports broadcaster, and a summary of the data is shown in Table 5.2. For our analysis, we consider ad lengths of 10 seconds, 20 seconds, and 30 seconds. This agrees closely with the data from the broadcaster. On average, ads from 20 advertisers were shown in each match, and their demands were as shown in Table 5.3:

Parameter	Value(seconds)
Average ad length	20.82
Standard Deviation	7.84
Minimum ad length	10
Maximum ad length	40
Number of samples (ads)	2393

Table 5.2: Analysis of ad length data

Parameter	Value(seconds)
Average Demand	83.82
Standard Deviation	9.84
Minimum Demand	20
Maximum Demand	180
Number of samples (matches)	25

Table 5.3: Analysis of Client Demands

In our discussions with the broadcaster, we found that the broadcaster had to satisfy service level commitments of between 75% to 80% of the total demand of each advertiser (i.e. 75%-80% of the demand had to be successfully aired). These commitments could be satisfied across the duration of the tournament, but for our study we limit our service level commitments to each game. For our simulation, we estimate demands from advertisers such that we can satisfy the service levels promised to most, if not all advertisers. To achieve this, we use 'penalties' in the model, so that unsatisfied demand below the promised service level decrease the profits earned. The IP model with service level guarantee is given in Section 5.3.1.

5.1.3 Spot Values

Spot values in the tournament we analyzed were linear with ad length. This made it easier to characterize the value of each advertiser in terms of how much revenue per second each of his ads earned, so broadcaster concerns such as giving higher value to 'sponsors' of a tournament could be incorporated by adding value to the revenue per second that that advertiser earns. For our simulation, we draw random values from the range $[3500, 5500]$. The number of advertisers per match varied in the tournament, with

Parameter	Value($)
Average Rev/Sec	4110
Standard Deviation	583
Minimum Rev/Sec	3714.28
Maximum Rev/Sec	5714.29
Number of samples (advertisers / match)	20

Table 5.4: Analysis of spot values

an average of 20 advertisers per match.

From the data collected, we could characterize the advertisers, their demands, and the break lengths for each match and run simulations that closely reflected real world requirements. We discuss the heuristics considered for the simulation below.

5.2 Assumptions

To simplify the models we examine, we assume the following:

1. Spot prices for ads are linear in ad length. This is supported by the data we received from the sports broadcaster (see Section 5.1.3).

2. Two ads from the same advertiser cannot be shown in the same break; but there is no restraint on showing two ads from one advertiser in subsequent breaks.

In later sections, we will add an assumption that service level commitments must be met, and the broadcaster pay a penalty if he doesn't meet those constraints.

5.3 Knapsack Model

In this section we study the basic Knapsack model and a variation with service level guarantees. The Knapsack model aims to fit the best possible combination of ads into a break. The size of the knapsack is taken as the expected length of the break. Similar to Witchakul et al [28] we consider penalties for crashes and underutilization, but unlike them, we build a model for multiple periods, where ad inventory changes from break to break.

The model is as follows:

$$\begin{aligned}
\max z &= \sum_n \sum_a \sum_l r_{al} * x_{aln} \\
\text{subject to:} & \\
\sum_l x_{aln} &\leq 1 \,, \forall a, n && (5.1)\\
\sum_a \sum_l (l * x_{aln}) &\leq b \,, \forall n && (5.2)\\
\sum_n x_{aln} &\leq N_{al} \,, \forall a, l && (5.3)\\
x_{aln} &= 0 \text{ or } 1 && (5.4)
\end{aligned}$$

where

a is the index of advertisers,

l is the length of each ad

n is the break sequence number

b is the expected / predicted break length

N_{al} is the number of ads of length l contracted from advertiser a

x_{aln} is the decision variable

Constraint(5.1) restricts the number of times an advertiser's ad can be shown in a break;

Constraint(5.2) specifies that the sum of all ads per break should be lower than the break length expected;

Constraint(5.3) ensures that we only show as many ads as we have a contract for;

Constraint(5.4) makes this model a binary integer programming model

5.3.1 Knapsack model with service guarantee

Sports broadcasters, ensure service levels (usually of around 80%) to their clients, and usually have to make good or pay a penalty when the service level commitments are not met. We include this constraint to the earlier Knapsack model discussed in section 5.3:

$$\begin{aligned}
\max z &= \sum_n \sum_a \sum_l r_{al} * x_{aln} - P \sum_a s_a \\
&\text{subject to:} \\
\sum_l x_{aln} &\leq 1 \,, \forall a, n && (5.5) \\
\sum_a \sum_l (l * x_{aln}) &\leq b \,, \forall n && (5.6) \\
\sum_n x_{aln} &\leq N_{al} \,, \forall a, l && (5.7) \\
\sum_n^{n+1} x_{aln} &\leq 1 \,, \forall a, l, n && (5.8)
\end{aligned}$$

$$\begin{aligned}
\sum_l \sum_n (l * x_{aln}) + s_a &\geq S * \sum_l (l * N_{al}), \ \forall a && (5.9) \\
x_{aln} &= 0 \text{ or } 1 && (5.10) \\
s_a &\geq 0 && (5.11)
\end{aligned}$$

where

P is the penalty for not meeting the promised service level

s_a is the duration by which the service level was not met

S is the promised service level

and all other variables have the same meaning as before (see section 5.3).

Constraint(5.9) ensures that the service level guarantees, if not met, are penalized in the Objective.

5.4 Heuristics

In this section we describe a few heuristics that we evaluate for recommendation to the sports broadcaster. These heuristics work on data as outlined in section 5.1. Finally, we compare the revenues earned by each approach with the revenue earned in the perfect information case (PI), where the lengths of *all* breaks are known in advance of the schedule generation, which is the theoretical upper bound.

5.4.1 Greedy Policy

The Greedy Policy assumes that break lengths are known in advance, and then uses the Knapsack model for each (known) break without planning for subsequent breaks. We have compared a simplified version of the Greedy Policy to the Optimal Policy in Chapter 3, and we will extend that study and compare the Greedy Policy with other heuristics.

5.4.2 Certainty Equivalent Heuristic

The Certainty Equivalent heuristic (CE) builds a schedule of ads based on the expected break length (based on the knapsack model with service level guarantee outlined in section 5.3.1). Having generated the breaks (which are all of length equal to the mean break length), we schedule them against the actual breaks and find out how it performs. This gives us a lower bound on how any variation of the certainty equivalent heuristic should perform.

5.4.3 Dynamic Certainty Equivalent Heuristic

For Dynamic Certainty Equivalent (DCE) , we generate ‘bundles’ to fit an expected *distribution* of break lengths and that satisfies all constraints. The IP is as outlined in section 5.3.1. The scheduler, who knows the length of the break, schedules the bundle that best matches the break size. If there are multiple bundles of equal size that fit in the break, the scheduler chooses the first one among them.

5.4.4 Dynamic Modified Certainty Equivalent Heuristic

The Dynamic Modified Certainty Equivalent Heuristic (DMCE) is a variation of DCE, we now generate bundles not only at the beginning of the match but also periodically during of the match. The periods of bundle generation could be varied: bundles could be generated at specific points during the match, or when a break occurs with no perfectly matching bundles at hand. We study both cases and report insights.

5.4.5 Perfect Information

The Perfect Information heuristic (PI) is the theoretical 'upper bound', so we can compare the performance of heuristics as a percentage the maximum revenue attainable. We assume that sizes of all breaks are known before the first break, and run a knapsack that schedules ads with the given constraints in all breaks.

5.5 Comparative Statics with Service Constraints

In this section, we study results of numerical analysis done based on the heuristics proposed in Section 5.4, under service constraints. The main aim of this study was to find out which of the heuristics was most promising, and to be able to suggest the most promising direction in which the sports broadcaster may direct their efforts to maximize the revenue in real world situations.

5.5.1 Parameters

Our assumptions for the following sections are as given in Section 5.2. The parameters for the numerical simulation are listed in Table 5.5.

For this simulation, we assume that advertisers order advertising time from the network, and that the network commits to a certain service level that is a percentage of the time sold to each advertiser. In our simulation, we assume that this value is 80%. Further, to discourage not meeting the service level, we set a penalty value of 1000$ for each second short of the promised service level. In the real world, networks either make

Parameter	Value
Number of breaks per match	50
Number of advertisers	20
Ad lengths (s)	10, 20, 30
Revenue per second per advertiser	Randomly drawn from $[3500, 5500]$
Break length distribution (s)	$\sim U(10, 60)$
Number of random trials (matches)	100
Target Service Level	80% of time sold
Penalty for not reaching service level	1000$ for each second below target

Table 5.5: Parameters for Numerical Simulation

good on their contract in subsequent tournaments, or show ads at the end of the game to make up on advertising time. For our simplified setup, a penalty of 1000$ suffices to show us the general direction in which we must direct our efforts.

5.5.2 Results

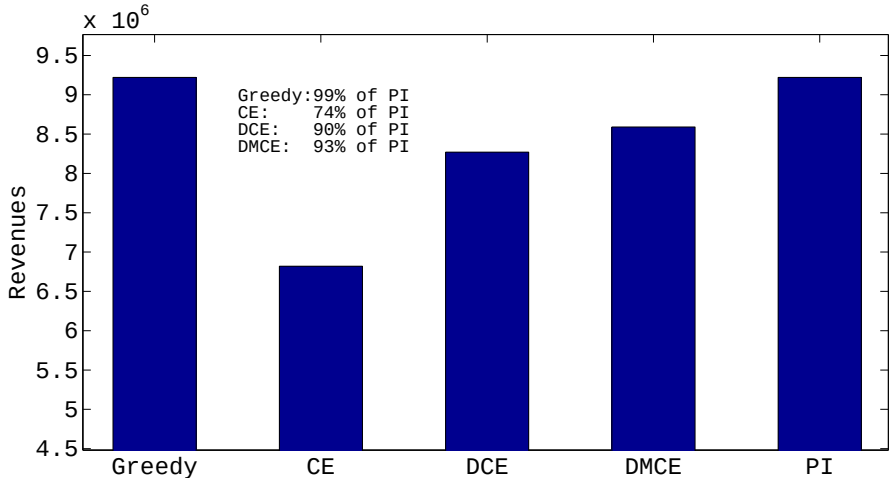

Figure 5.2: Performance of heuristics with service constraints

Among the heuristics selected, while the Greedy does not create bundles beforehand, the CE, DCE and DMCE heuristics rely on creating bundles, either before the match or during the match. From the results, we see that the Greedy performs almost on par with the PI, while heuristics that depend on creating bundles before the match do not perform as well.

Heuristic	% of PI
Greedy	99%
CE	74%
DCE	90%
DMCE	93%

Table 5.6: Performance of heuristics relative to PI

When break sizes are known before the commencement of the break, creating bundles and trying to fit the best one offers us no advantage over myopically choosing the best ads to fit the break. Since the size of the bundle is fixed, when we run out of appropriately sized bundles to fit into the break, we are forced to schedule a bundle that is of smaller size than the break, hence losing out on earning opportunities. Having bundles that fit the break by regenerating them (DMCE) does not guarantee us optimum ad selection, since valuable ads that could have been scheduled in the current break may be included in a bundle of a different size, and hence not scheduled. Therefore, the Greedy is able to best capitalize on the advance knowledge of break sizes.

We note that the Optimal Policy as described in Chapter 3 does not rely on creating bundles before the realization of breaks. In our discussions with the sports broadcaster, we found that the ad scheduling team did create bundles beforehand, but the bundles were discarded when they didn't have the right bundle for a break. The network should therefore stick to a flexible schedule that does not depend on pre-created bundles, affording flexibility in scheduling and giving them a better chance to earn higher revenues.

5.6 Comparison of Greedy and Optimal Policies

We next study numerically the conditions that determine how well the Optimal Policy performs over the Greedy Policy. In this study, we only consider the Base Case as presented in Section 3.2.1, where breaks and ads are of two sizes, short (15 seconds) and long (30 seconds), and the number of breaks is fixed.

We vary parameters across the relative value of small and large ads, the distribution of the two types of ads, and the Service Level, defined as the percentage of air time sold

that is expected to be aired (the lower the percentage, the higher the air time sold). The parameters are summarized in Table 5.7.

Parameter	Values
Ratio of values of small to large ads	1, 0.55, 0.50, 0.45, 0.40, 0.35, 0.30
Ratio of number of small to large ads	3, 2, 1, 1/2, 1/3
Service Level	80%,90%,100%,110%

Table 5.7: Parameters for comparison of Greedy and Optimal Policy

The results are presented in Appendix B. The Revenues tables in Section B.1 show the potential revenue that could be earned if all orders that were accepted could be shown, as well as the performance of the Greedy Policy and the Optimal Policy as a percentage of this total. The Service Level tables shown in Section B.1 show the percentage of ads, small and long, that were shown, and the Utilization tables in Section B.3 show the percentage of total break time that was utilized for showing ads.

5.6.1 Results

The percentage gain of the Optimal Policy over the Greedy Policy against the variation in the relative value of small ads is shown in Figure 5.3, where each graph is drawn for a particular service level.

We observe that gains of Optimal Policy over Greedy Policy monotonically decrease as the value of small ads decreases in comparison to large ads. While the greatest gains are seen when small ads are almost as valuable as large ads, when the relative value of small ads is 0.45 or less, the Optimal Policy shows no gains over the Greedy Policy.

This behavior can be explained by the Optimal Policy having a lower threshold for large ads, and therefore its tendency to schedule more large ads than the Greedy Policy (as discussed in Section 3.2.1 and Section 4.1.2). When small ads are as valuable as large ads, the Greedy Policy schedules small ads up front in the large breaks, since it earns twice as much with two small ads than one large ad. This behavior causes it to run out of small ads earlier than the Optimal Policy would, and it fails to schedule small ads in the small breaks that occur at some point after it runs out of small ads.

The difference in the service levels (tables in Section B.2) of large ads is acute when the ratio of small ads to large ads is 1:2, where we have half as many small ads as large ads, causing the Greedy Policy to run out of small ads early in a match.

When the ratio of small ads to large ads is 1:3, however, the gains made by Optimal Policy drop below those made when the ratio is 1:2. While the Optimal Policy does not exhaust its inventory of small ads as early as the Greedy Policy, the number of small ads is small enough for it to lose scheduling opportunities in small breaks at the later stages of a game. Tables in Section B.3 show that the utilization of breaks by the Optimal Policy when we have a 1:3 distribution is consistently less than 100%, and is always less than the utilization when ads are distributed by a 1:2 ratio.

When the relative value of small ads is 0.45 or less, the Greedy Policy schedules large ads just as often as the Optimal Policy, since two small ads no longer have as much value as one large ad. Thus we see no difference in either the revenues earned, or the service levels of small and large ads.

It can be argued that when the relative value of small ads is greater than 1, we would see that the gains made by the Optimal Policy decrease once again (compared to the case where the relative value of small ads is 1). Despite the Optimal Policy having a low threshold for large ads, the small ads be valuable enough for the most valuable large ad to not make that threshold, causing the Optimal Policy and the Greedy Policy to schedule similarly. Having small ads of relative value greater than 1, however, is only of academic interest, and we do not discuss it in detail.

From a managerial perspective, we see that the Greedy Policy is just as effective as the Optimal Policy when the inventory has small ads that are less than half the value of large ads. The Greedy Policy is easily implemented, and the network broadcaster need not invest in forward looking heuristics in such a case. Conversely, as the relative value of small ads increases above the 0.5 mark, the network broadcaster can significantly improve his revenues by implementing the Optimal Policy.

We also note that the greatest gains made by using the Optimal Policy occur when the ratio of small ads to large ads is 1:2. This may occur when the broadcaster has priced his small ads to a level where advertisers see more value in buying large ads,

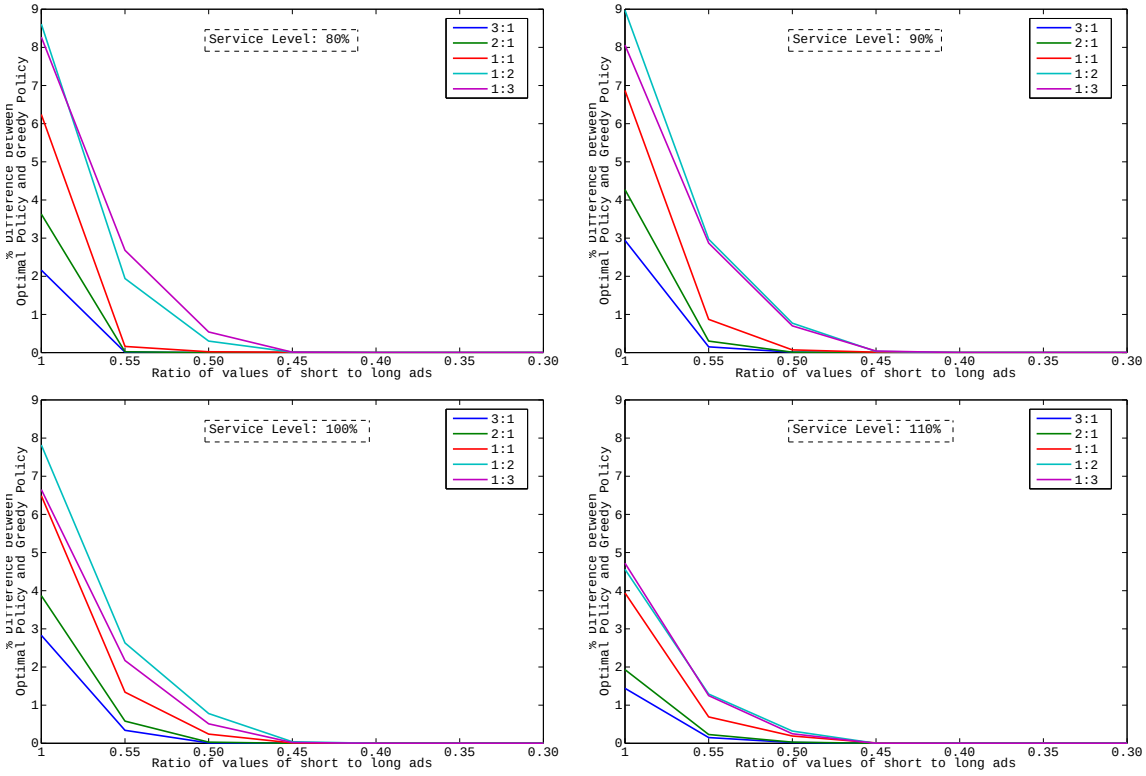

Figure 5.3: Performance of Optimal Policy over Greedy Policy

thereby skewing the ad distribution.

Finally, we note that service levels between 80% and 90% provide the best returns for using the Optimal Policy, and we note that our discussions with the network broadcaster showed that service levels in that range were usually targeted.

Chapter 6

Conclusion

TV networks showing live sports are often challenged by having breaks of non deterministic size, and the high profit margins in live sports broadcasting demand a better way of scheduling ads in such situations. In this dissertation we have discussed a method to schedule ads optimally when breaks are of random size and number, and the broadcaster has ads of two lengths.

Earlier literature related to advertising scheduling assume fixed break durations, and do not sufficiently answer how ads must be scheduled when faced with uncertainty in break sizes. Literature related to Random Yield, Stochastic Knapsack, and Stochastic Recourse do not sufficiently match the setting typical of our problem.

We find that the Optimal Policy when faced with non-deterministic breaks is a forward looking Greedy implementation. Bundling strategies fail to sufficiently account for the stochasticity in break sizes, and earn less than a flexible heuristic such as the Greedy Policy. Further, we show that the Optimal Policy outperforms the Greedy Policy when small ads have a value equal to or greater than half the value of a large ad.

While we do not account for all the constraints that broadcasters face, our model is general enough to be applied to a class of bin packing problems, for instance, cargo shipping when containers have non-deterministic capacity. This simple model, however, does not fully meet the network broadcaster's requirements. There is scope for the following extensions to this work:

1. Inclusion of diversity constraints into the Optimal Policy: two ads from the same advertiser, or from competing advertisers, cannot be shown in a break.

2. Optimal Policy when breaks are not all IID: for instance a rain break or an injury break takes a longer time than mid-over breaks.

3. Incorporating service levels constraints for advertisers, agencies, and geographic regions.

4. Extension to help managers in making pricing decisions and accepting spot orders based on the Optimal Policy.

5. A study of Broadcaster-Advertiser behavior based on Game Theoretic principles when the broadcaster employs the Optimal Policy.

To conclude, the area of Optimal scheduling of items (ads, cargo, etc) in non-deterministic containers is an area that has many possibilities for research and development. It is hoped that this dissertation is a stepping stone in establishing improved heuristics in this area.

Bibliography

[1] V. Araman and I. Popescu. Stochastic revenue management models for media broadcasting. Technical report, Working paper, 2007.

[2] R. Bellman. *Dynamic programming*. Dover Pubns, 2003.

[3] J. Birge. Stochastic programming computation and applications. *INFORMS Journal on Computing*, 9(2):111–133, 1997.

[4] J. Birge and F. Louveaux. *Introduction to stochastic programming*. Springer Verlag, 1997.

[5] S. Bollapragada and M. Garbiras. Scheduling commercials in broadcast television. *Operations Research*, 52(3):337–345, 2003.

[6] S. Bollapragada and T. Morton. Myopic heuristics for the random yield problem. *Operations Research*, 47(5):713–722, 1999.

[7] F. Ciarallo, R. Akella, and T. Morton. A periodic review, production planning model with uncertain capacity and uncertain demand-optimality of extended myopic policies. *Management Science*, 40(3):320–332, 1994.

[8] B. De Reyck and Z. Degraeve. Broadcast scheduling for mobile advertising. *Operations Research*, 51(4):509–517, 2003.

[9] R. Ghodsi and F. Sassani. Online cutting stock optimization with prioritized orders. *Assembly Automation*, 25(1):66–72, 2005.

[10] P. Gilmore and R. Gomory. A linear programming approach to the cutting stock problem. *Operations research*, 9(6):849–859, 1961.

[11] P. Gilmore and R. Gomory. A linear programming approach to the cutting stock problem-Part II. *Operations research*, 11(6):863–888, 1963.

[12] P. Gilmore and R. Gomory. Multistage cutting stock problems of two and more dimensions. *Operations Research*, 13(1):94–120, 1965.

[13] A. Grosfeld-Nir and Y. Gerchak. Multiple lotsizing in production to order with random yields: Review of recent advances. *Annals of Operations Research*, 126(1):43–69, 2004.

[14] J. Hwang and M. Singh. Optimal production policies for multi-stage systems with setup costs and uncertain capacities. *Management Science*, 44(9):1279–1294, 1998.

[15] D. Khang and O. Fujiwara. Optimality of myopic ordering policies for inventory model with stochastic supply. *Operations Research*, 48(1):181–184, 2000.

[16] K. Kianfar and Y. Fathi. Generalized mixed integer rounding inequalities: facets for infinite group polyhedra. *Mathematical Programming*, 120(2):313–346, 2009.

[17] A. Kimms and M. Muller-Bungart. Revenue management for broadcasting commercials: the channel's problem of selecting and scheduling the advertisements to be aired. *International Journal of Revenue Management*, 1(1):28–44, 2007.

[18] W. Klein Haneveld and M. van der Vlerk. Stochastic integer programming: General models and algorithms. *Annals of Operations Research*, 85:39–57, 1999.

[19] A. Kleywegt and J. Papastavrou. The dynamic and stochastic knapsack problem. *Operations Research*, 46(1):17–35, 1998.

[20] A. Kleywegt and J. Papastavrou. The dynamic and stochastic knapsack problem with random sized items. *Operations Research*, 49(1):26–41, 2001.

[21] J. Papastavrou, S. Rajagopalan, and A. Kleywegt. The dynamic and stochastic knapsack problem with deadlines. *Management Science*, 42(12):1706–1718, 1996.

[22] T. Perry and J. Hartman. An approximate dynamic programming approach to solving a dynamic, stochastic multiple knapsack problem. *International Transactions in Operational Research*, 16(3):347–359, 2009.

[23] D. Scull. A stochastic cutting stock procedure: cutting rolls of insulating tape. *Management Science*, 27(8):946–952, 1981.

[24] S. Sen. Algorithms for stochastic mixed-integer programming models. *Handbook of Discrete Optimization*, pages 515–558, 2005.

[25] A. Stakhov and B. Rozin. Theory of Binet formulas for Fibonacci and Lucas p-numbers. *Chaos, Solitons & Fractals*, 27(5):1162–1177, 2006.

[26] Y. Wang and Y. Gerchak. Continuous review inventory control when capacity is variable. *International Journal of Production Economics*, 45(1-3):381–388, 1996.

[27] Y. Wang and Y. Gerchak. Periodic review production models with variable capacity, random yield, and uncertain demand. *Management Science*, 42(1):130–137, 1996.

[28] S. Witchakul, P. Ayudhaya, and P. Charnsethikul. A Stochastic Knapsack Problem with Continuous Random Capacity. *Journal of Mathematics and Statistics*, 4(4):269–276, 2008.

[29] C. Yano and H. Lee. Lot sizing with random yields: A review. *Operations Research*, 43(2):311–334, 1995.

[30] X. Zhang. Mathematical models for the television advertising allocation problem. *International Journal of Operational Research*, 1(3):302–322, 2006.

Appendix A

Proofs of theorems

Notation	Explanation
$V_i(l,s)$	Expected revenue earned from break i onwards, when the indexes of the first large ad and the first small ad available are l and s respectively
R_l	Expected revenue earned by selecting l_1 in the current (large) break
R_s	Expected revenue earned by selecting s_1+s_2 in the current (large) break
b	Number of breaks remaining

Table A.1: Summary of notation

A.1 Proof of Theorem 3.1

Proof. From Lemma 3.1 and Lemma 3.2, we know the rule to be true when one or two breaks remain.

Let us assume the rule be true for breaks

$2,\ldots,n$. We will use then use induction to prove this theorem.

Then,

$$R_l = l_1 + V_2(2,1)$$

$$R_s = s_1 + s_2 + V_2(1,3)$$

$$V_n(1,1) = max\{R_l, R_s\}$$

We prove that $R_l - R_s \geq 0$ when $l_1 \geq s_n + s_{n+1}$.

$$\begin{aligned} R_l - R_s &= l_1 + V_2(2,1) - [s_1 + s_2 + V_2(1,3)] \\ &= (l_1 - (s_1 + s_2)) + (V_2(2,1) - V_2(1,3)) \end{aligned}$$

Expanding the above to a look ahead of two breaks, we get:

$$\begin{aligned} R_l - R_s = (l_1 - (s_1 + s_2)) & \\ + p[s_1 + V_3(2,2)] - p[s_3 + V_3(1,4)] & \\ + (1-p)\max \begin{cases} l_2 + V_3(3,1), & \text{if } l_2 > s_{n-1} + s_n \\ s_1 + s_2 + V_3(2,3) & \text{otherwise} \end{cases} & \\ - (1-p)(l_1 + V_3(2,3)) & \end{aligned}$$

Though we have no information about l_2 and $s_{n-1} + s_n$, the *max* operator guarantees that the value of $V_2(2,1)$ must at least be $s_1 + s_2 + V_3(2,3)$. Thus we get:

$$\begin{aligned} R_l - R_s &\geq (l_1 - (s_1 + s_2)) \\ &\quad + p[s_1 + V_3(2,2) - s_3 - V_3(1,4)] \\ &\quad + (1-p)[s_1 + s_2 + V_3(2,3) - l_1 - V_3(2,3)] \\ &\geq p\big[l_1 - (s_2 + s_3) + V_3(2,2) - V_3(1,4))\big] \end{aligned}$$

Continuing enumeration to look ahead for $i < n$ breaks, we get

$$R_l - R_s \geq p^{i-1}\Big[l_1 - (s_i + s_{i+1}) + V_{n-i}(2,i) - V_{n-i}(1,i+2)\Big]$$

Thus, for $i = n-1$, we get:

$$\begin{aligned}
R_l - R_s &\geq p^{n-2}\Big[l_1 - (s_{n-1}+s_n) + V_n(2,n-1) - V_n(1,n+1)\Big] \\
&\geq p^{n-2}\Big[l_1 - (s_{n-1}+s_n) \\
&\quad + p(s_{n-1}-s_{n+1}) + (1-p)(s_{n-1}+s_n-l_1)\Big] \\
&\geq p^{n-2}\Big[p(l_1-(s_n+s_{n+1}))\Big] \\
&\geq 0 \quad (\because p \geq 0,\, l_1 \geq (s_n+s_{n+1})
\end{aligned}$$

Therefore we have proved that when the number of breaks remaining is n and $l_1 \geq s_n + s_{n+1}$, it is optimal to pack l_1 first.

The reverse case, i.e. $R_s - R_l \geq 0$ when $s_n + s_{n+1} \geq l_1$ can be proved similarly. □

A.2 Proof of Theorem 3.2

Proof. From Lemma 3.1 and Lemma 3.3, we know the rule to be true when one and two breaks remain.

Let us assume the rule be true when the number breaks remaining are $1, 2, \ldots, n-1$. We will use then use induction to prove this theorem. As before, we use R_l and R_s to denote the revenues earned by scheduling l_1 and (s_1, s_2) respectively in b_1. We have:

$$R_l = l_1 + V_2(2,1)$$
$$R_s = s_1 + s_2 + V_2(1,3)$$

Then,

$$\begin{aligned}
R_l - R_s &= l_1 + V_2(2,1) - [s_1 + s_2 + V_2(1,3)] \\
&= (l_1 - (s_1 + s_2)) + (V_2(2,1) - V_2(1,3)) \\
&= (l_1 - (s_1 + s_2)) \\
&+ \Big[p_0(V_3(2,1) - V_3(1,3)) \\
&+ \ \ p_1(s_1 + V_3(2,2) - s_3 - V_3(1,4)) \\
&+ \ \ p_2(\max\{l_2 + V_3(3,1), s_1 + s_2 + V_3(2,3)\} \\
&\quad - \max\{l_1 + V_3(2,3), s_3 + s_4 + V_3(1,5)\}) \Big]
\end{aligned}$$

We can trivially prove that

$$\begin{aligned}
(p_0 + p_1)^{n-1} l_1 &\geq \sum_{i=0}^{n-1} \left[C_i^{n-1} \ p_0^{n-1-i} p_1^i \ (s_{i+1} + s_{i+2}) \right] \\
\implies (p_0 + p_1)^{n-2} l_1 &\geq \sum_{i=0}^{n-2} \left[C_i^{n-2} \ p_0^{n-2-i} p_1^i \ (s_{i+3} + s_{i+4}) \right] \qquad \text{(A.1)}
\end{aligned}$$

Using Equation A.1 and using induction, we can say that

$$V_2(1,3) = p_0 V_3(1,3) + p_1(s_3 + V_3(1,4)) + p_2(l_1 + V_3(2,3))$$

Substituting for $R_l - R_s$, we get:

$$\begin{aligned}
R_l - R_s &\geq (l_1 - (s_1 + s_2)) \\
&\quad + \Big[p_0(V_3(2,1) - V_3(1,3)) \\
&\quad + \ \ p_1(s_1 + V_3(2,2) - s_3 - V_3(1,4)) \\
&\quad + \ \ p_2(s_1 + s_2 + V_3(2,3) - l_1 - V_3(2,3)) \Big] \\
&\quad \geq (1 - p_2)(l_1 - (s_1 + s_2)) \\
&\quad + \Big[p_0(V_3(2,1) - V_3(1,3)) \\
&\quad + \ \ p_1(s_1 + V_3(2,2) - s_3 - V_3(1,4)) \Big]
\end{aligned}$$

$$
\begin{aligned}
&\geq (p_0+p_1)l_1 - p_0(s_1+s_2) - p_1(s_2+s_3) \\
&+ \Big[p_0(V_3(2,1) - V_3(1,3)) \\
&+ \;\; p_1(V_3(2,2) - V_3(1,4))\Big]
\end{aligned}
$$

On expanding, and applying induction throughout,

$$
\begin{aligned}
p_0(V_3(2,1) - V_3(1,3)) + p_1(V_3(2,2) - V_3(1,4)) \\
&\geq p_0\Big[p_0(V_4(2,1) - V_4(1,3)) \\
&+ \;\; p_1(s_1 + V_4(2,2) - s_3 - V_4(1,4)) \\
&+ \;\; p_2(s_1+s_2+V_4(2,3) - l_1 - V_4(2,3))\Big] \\
&+ p_1\Big[p_0(V_4(2,2) - V_4(1,4)) \\
&+ \;\; p_1(s_2 + V_4(2,3) - s_4 - V_4(1,5)) \\
&+ \;\; p_2(s_2+s_3+V_4(2,4) - l_1 - V_4(2,4))\Big]
\end{aligned}
$$

Substituting for $R_l - R_s$, we get:

$$
\begin{aligned}
R_l - R_s \geq (p_0+p_1)^2 l_1 \\
&- p_0^2(s_1+s_2) - 2p_0p_1(s_2+s_3) - p_1^2(s_3+s_4) \\
&+ \;\; p_0^2(V_4(2,1) - V_4(1,3)) \\
&+ \;\; 2p_0p_1(V_4(2,2) - V_4(1,4)) \\
&+ \;\; p_1^2(V_4(2,3) - V_4(1,5))
\end{aligned}
$$

Continuing to enumerate in this fashion and applying induction, we get:

$$\begin{aligned}
R_l - R_s &\geq (p_0+p_1)^{n-2} l_1 \\
&\quad - p_0^{n-2}(s_1+s_2) - (n-2)p_0^{n-3}p_1(s_2+s_3) - \ldots - p_1^{n-2}(s_{n-1}+s_n) \\
&\quad + \ p_0^{n-2}(V_n(2,1) - V_n(1,3)) \\
&\quad + \ (n-2)p_0^{n-3}p_1(V_1(2,2) - V_1(1,4)) \\
&\quad \vdots \\
&\quad + \ p_1^{n-2}(V_1(2,n-1) - V_n(1,n+1)) \\
&\quad \geq (p_0+p_1)^{n-1} l_1 - \sum_{i=0}^{n-1} \left[C(n-1,i)\ p_0^{n-1-i} p_1^i\ (s_{i+1}+s_{i+2}) \right] \\
&\quad \geq 0 \text{ (by definition)}
\end{aligned}$$

The proof for the reverse case, i.e. $R_s - R_l \geq 0$ when

$$(p_0+p_1)^{n-1} l_1 < \sum_{i=0}^{n-1} \left[C(n-1,i)\ p_0^{n-1-i} p_1^i\ (s_{i+1}+s_{i+2}) \right]$$

can be proved similarly. □

A.3 Proof of Theorem 3.3

Proof. We have proved the strategy to be true when $k = 1$ (Lemma 3.4) and $k = 2$ (Lemma 3.5).

To prove the strategy is true when $k = n$, let us assume the strategy hold good for breaks $(2,3,\ldots,n)$, i.e. for all subsequent breaks.

As before, let $\hat{O} = (l_1,\ldots,l_\lambda,s_1,\ldots,s_{m-2\lambda})$.

Let $O_l = (l_1,\ldots,l_\lambda,l_{\lambda+1},s_1,\ldots,s_{m-2\lambda-2})$ and $O_s = (l_1,\ldots,l_{\lambda-1},s_1,\ldots,s_{m-2\lambda+2})$. Let R_o, be the revenue earned by selecting the ads in $\hat{O}$, and let R_l and R_s be the revenues

earned by selecting O_l and O_s respectively.

$$R_o = \sum_{i=1}^{\lambda} l_i + \sum_{j=1}^{m-2\lambda} s_j + V_2(\lambda+1, m-2\lambda+1) \tag{A.2}$$

$$R_l = \sum_{i=1}^{\lambda+1} l_i + \sum_{j=1}^{m-2\lambda-2} s_j + V_2(\lambda+2, m-2\lambda-1) \tag{A.3}$$

$$R_s = \sum_{i=1}^{\lambda-1} l_i + \sum_{j=1}^{m-2\lambda+2} s_j + V_2(\lambda, m-2\lambda+3) \tag{A.4}$$

We have to prove that:

1. $R_o \geq R_l$ and
2. $R_o \geq R_s$

Case 1:

$$\begin{aligned} R_o - R_l &= \sum_{i=1}^{\lambda} l_i + \sum_{j=1}^{m-2\lambda} s_j + V_2(\lambda+1, m-2\lambda+1) \\ &\quad - \left[\sum_{i=1}^{\lambda+1} l_i + \sum_{j=1}^{m-2\lambda-2} s_j + V_2(\lambda+2, m-2\lambda-1) \right] \\ &= -l_{\lambda+1} + s_{m-2\lambda-1} + s_{m-2\lambda} \\ &\quad + V_2(\lambda+1, m-2\lambda+1) - V_2(\lambda+2, m-2\lambda-1) \end{aligned}$$

Note that $l_{\lambda+2} < s_{m-2\lambda+n-2} + s_{m-2\lambda+n-1}$. By a similar argument as given in Equation 3.7, we can write:

$$\begin{aligned} V_2(\lambda+1, m-2\lambda+1) - V_2(\lambda+2, m-2\lambda-1) & \\ &\geq p_1 \Big(s_{m-2\lambda+1} + V_3(\lambda+1, m-2\lambda+2) \\ &\quad - s_{m-2\lambda-1} - V_3(\lambda+2, m-2\lambda) \Big) \\ &\quad + \sum_{k=2}^{M} p_k \left(l_{\lambda+1} - [s_{m-2\lambda-1} + s_{m-2\lambda}] \right) \end{aligned}$$

Substituting in $R_o - R_l$, we get:

$$\begin{aligned}
R_o - R_l &\geq -l_{\lambda+1} + s_{m-2\lambda-1} + s_{m-2\lambda} \\
&+ p_1\Big(s_{m-2\lambda+1} + V_3(\lambda+1, m-2\lambda+2) \\
&\quad - s_{m-2\lambda-1} - V_3(\lambda+2, m-2\lambda)\Big) \\
&+ \sum_{k=2}^{M} p_k\,(l_{\lambda+1} - [s_{m-2\lambda-1} + s_{m-2\lambda}]) \\
&\geq p_1[-l_{\lambda+1} + s_{m-2\lambda} + s_{m-2\lambda+1} \\
&+ V_3(\lambda+1, m-2\lambda+2) - V_3(\lambda+2, m-2\lambda)]
\end{aligned}$$

Continuing enumeration of the above to expand to all n breaks, we get:

$$\begin{aligned}
R_o - R_l &\geq p_1^{n-1}\,[-l_{\lambda+1} + s_{m-2\lambda+n-2} + s_{m-2\lambda+n-1}] \\
&\geq 0\ (\because l_{\lambda+1} < s_{m-2(\lambda+1)+n} + s_{m-2(\lambda+1)+n+1})
\end{aligned}$$

Similarly, we can prove **Case 2** by showing that

$$R_o - R_s \geq p_1^{n-1}\,[l_\lambda - s_{m-2\lambda+n} - s_{m-2\lambda+n+1}] \geq 0$$

□

A.4 Proof of Theorem 3.4

Proof. We have proved the strategy to be true when $k = 1$ (Lemma 3.6) and $k = 2$ (Lemma 3.7).

To prove the strategy is true when $k = n$, let us assume the strategy hold good for breaks $(2, \ldots, n)$, i.e. for all subsequent breaks.

We define $\hat{O}, O_l, O_s$ as before.

Let R_o, R_l, and R_s denote the revenues earned by selecting $\hat{O}$, O_l and O_s respectively. We have to prove that

1. $R_o \geq R_l$

2. $R_o \geq R_s$

Case 1:

By definition:

$$(p_0+p_1)^{n-1} l_\lambda \geq \sum_{i=0}^{n-1} \left[\binom{n-1}{i} p_0^{n-1-i} p_1^i (s_{m-2\lambda+i+1} + s_{m-2\lambda+i+2}) \right] \tag{A.5}$$

$$\text{and } (p_0+p_1)^{n-1} l_{\lambda+1} < \sum_{i=0}^{n-1} \left[\binom{n-1}{i} p_0^{n-1-i} p_1^i (s_{m-2\lambda+i-1} + s_{m-2\lambda+i}) \right] \tag{A.6}$$

As before,

$$\begin{aligned} R_o - R_l = &- l_{\lambda+1} + s_{m-2\lambda-1} + s_{m-2\lambda} \\ &+ V_2(\lambda+1, m-2\lambda+1) - V_2(\lambda+2, m-2\lambda-1) \end{aligned}$$

From eq. A.6,

$$\begin{aligned} &(p_0+p_1)^{n-1} l_{\lambda+1} < \sum_{i=0}^{n-1} \left[\binom{n-1}{i} p_0^{n-1-i} p_1^i (s_{m-2\lambda+i-1} + s_{m-2\lambda+i}) \right] \\ \implies &(p_0+p_1)^{n-1} l_{\lambda+2} < \sum_{i=0}^{n-1} \left[\binom{n-1}{i} p_0^{n-1-i} p_1^i (s_{m-2\lambda+i-1} + s_{m-2\lambda+i}) \right] \\ \implies &(p_0+p_1)^{n-1} l_{\lambda+2} < \sum_{i=0}^{n-1} \left[\binom{n-1}{i} p_0^{n-1-i} p_1^i (s_{m-2\lambda+i-3} + s_{m-2\lambda+i-2}) \right] \end{aligned}$$

Then by induction, for $b_2 \geq 2S$, $V_2(\lambda+2, m-2\lambda-1)$ earns us $(s_{m-2\lambda-1} + s_{m-2\lambda})$ followed by the sum of values of ads selected from the set $\{l_{\lambda+2}, \ldots, s_{m-2\lambda+1}, \ldots\}$. On the other hand, with $V_2(\lambda+1, m-2\lambda+1)$ we earn *at least* $l_{\lambda+1}$ followed by the sum of values of ads from the set $\{l_{\lambda+2}, \ldots, s_{m-2\lambda+1}, \ldots\}$.

Therefore:

$$
\begin{aligned}
&V_2(\lambda+1,m-2\lambda+1)-V_2(\lambda+2,m-2\lambda-1)\\
&\quad\geq p_0(V_3(\lambda+1,m-2\lambda+1)-V_3(\lambda+2,m-2\lambda-1))\\
&\quad+p_1\left(s_{m-2\lambda+1}-s_{m-2\lambda-1}+V_3(\lambda+1,m-2\lambda+2)-V_3(\lambda+2,m-2\lambda)\right)\\
&\quad+\sum_{i=2}^{M}p_i\left(l_{\lambda+1}-[s_{m-2\lambda-1}+s_{m-2\lambda}]\right)
\end{aligned}
$$

Substituting for R_o-R_l, we get:

$$
\begin{aligned}
R_o-R_l&\geq -l_{\lambda+1}+s_{m-2\lambda-1}+s_{m-2\lambda}\\
&\quad+p_0(V_3(\lambda+1,m-2\lambda+1)-V_3(\lambda+2,m-2\lambda-1))\\
&\quad+p_1(s_{m-2\lambda+1}-s_{m-2\lambda-1}+V_3(\lambda+1,m-2\lambda+2)-V_3(\lambda+2,m-2\lambda))\\
&\quad+\sum_{i=2}^{M}p_i\left(l_{\lambda+1}-[s_{m-2\lambda-1}+s_{m-2\lambda}]\right)\\
&\quad\geq -(p_0+p_1)l_{\lambda+1}+p_0(s_{m-2\lambda-1}+s_{m-2\lambda})+p_1(s_{m-2\lambda}+s_{m-2\lambda+1})\\
&\quad+p_0(V_3(\lambda+1,m-2\lambda+1)-V_3(\lambda+2,m-2\lambda-1))\\
&\quad+p_1(V_3(\lambda+1,m-2\lambda+2)-V_3(\lambda+2,m-2\lambda))\\
&\quad\geq -(p_0+p_1)l_{\lambda+1}+p_0(s_{m-2\lambda-1}+s_{m-2\lambda})+p_1(s_{m-2\lambda}+s_{m-2\lambda+1})\\
&\quad+p_0^2(V_4(\lambda+1,m-2\lambda+1)-V_4(\lambda+2,m-2\lambda-1))\\
&\quad+p_0p_1(s_{m-2\lambda+1}-s_{m-2\lambda-1})+p_0p_1(V_4(\lambda+1,m-2\lambda+2)-V_4(\lambda+2,m-2\lambda))\\
&\quad+p_0\sum_{i=2}^{M}(l_{\lambda+1}-(s_{m-2\lambda-1}+s_{m-2\lambda}))\\
&\quad+p_1p_0(V_4(\lambda+1,m-2\lambda+2)-V_4(\lambda+2,m-2\lambda))\\
&\quad+p_1^2(s_{m-2\lambda+2}-s_{m-2\lambda})+p_1^2(V_4(\lambda+1,m-2\lambda+3)-V_4(\lambda+2,m-2\lambda+1))\\
&\quad+p_1\sum_{i=2}^{M}(l_{\lambda+1}-(s_{m-2\lambda}+s_{m-2\lambda+1}))
\end{aligned}
$$

$$
\begin{aligned}
&\geq l_{\lambda+1}(-p_0-p_1+p_0(1-p_0-p_1)+p_1(1-p_0-p_1))\\
&+s_{m-2\lambda-1}(p_0-p_0p_1-p_0(1-p_0-p_1))\\
&+s_{m-2\lambda}(p_0+p_1-p_0(1-p_0-p_1)-p_1^2-p_1(1-p_0-p_1))\\
&+s_{m-2\lambda+1}(p_1+p_0p_1-p_1(1-p_0-p_1))\\
&+s_{m-2\lambda+2}(p_1^2)\\
&+p_0^2(V_4(\lambda+1,m-2\lambda+1)-V_4(\lambda+2,m-2\lambda-1))\\
&+2p_0p_1(V_4(\lambda+1,m-2\lambda+2)-V_4(\lambda+2,m-2\lambda))\\
&+p_1^2(V_4(\lambda+1,m-2\lambda+3)-V_4(\lambda+2,m-2\lambda+1))\\
&\geq -l_{\lambda+1}(p_0+p_1)^2\\
&+p_0^2(s_{m-2\lambda-1}+s_{m-2\lambda})+2p_0p_1(s_{m-2\lambda}+s_{m-2\lambda+1})+p_1^2(s_{m-2\lambda+1}+s_{m-2\lambda+2})\\
&+p_0^2(V_4(\lambda+1,m-2\lambda+1)-V_4(\lambda+2,m-2\lambda-1))\\
&+2p_0p_1(V_4(\lambda+1,m-2\lambda+2)-V_4(\lambda+2,m-2\lambda))\\
&+p_1^2(V_4(\lambda+1,m-2\lambda+3)-V_4(\lambda+2,m-2\lambda+1))
\end{aligned}
$$

Continuing to enumerate in this fashion, and applying induction, after expanding up to V_n we get:

$$
\begin{aligned}
R_o-R_l &\geq -l_{\lambda+1}(p_0+p_1)^{n-2}+\sum_{i=0}^{n-2}\binom{n-2}{i}p_0^{n-2-i}p_1^i(s_{m-2\lambda+i-1}+s_{m-2\lambda+i})\\
&+\sum_{i=0}^{n-2}\binom{n-2}{i}p_0^{n-2-i}p_1^i(V_n(\lambda+1,m-2\lambda+i+1)-V_n(\lambda+2,m-2\lambda+i-1))\\
&\geq -(p_0+p_1)^{n-1}l_{\lambda+1}\\
&+\sum_{i=0}^{n-1}\left[\binom{n-1}{i}p_0^{n-1-i}p_1^i(s_{m-2\lambda+i-1}+s_{m-2\lambda+i})\right]\\
&\geq 0 \text{ from eq A.6}
\end{aligned}
$$

Case 2: $R_o-R_s\geq 0$ can be similarly proved using eq. A.5. □

A.5 Proof of Theorem 3.5

Proof. We have proved all three cases to be true when one break and two breaks remain. Let the rule be true for all subsequent breaks, $b_2, ..., b_n$. We will then prove by induction that it is true for break b_1.

Case 1: Let $(1-p)l_1 = ps_1 + (1-p)(s_1+s_2) + t$, where $t \geq 0$.

Then,

$$R_l = pV_2(1,1) + (1-p)(l_1 + V_2(2,1))$$
$$R_s = p(s_1 + V_2(1,2)) + (1-p)(s_1 + s_2 + V_2(1,3))$$

Thus, we get:

$$\begin{aligned} R_l - R_s &= p[V_2(1,1) - s_1 - V_2(1,2)] \\ &\quad + (1-p)[l_1 - (s_1+s_2) + V_2(2,1) - V_2(1,3)] \\ &= t + p[V_2(1,1) - V_2(1,2)] + (1-p)[V_2(2,1) - V_2(1,3)] \end{aligned}$$

Expanding the above to look ahead two breaks, we get:

$$\begin{aligned} R_l - R_s &= t + p[V_2(1,1) - V_2(1,2)] + (1-p)[V_2(2,1) - V_2(1,3)] \\ &\geq t + p\Big[pV_3(1,1) + (1-p)(l_1 + V_3(2,1)) \\ &\quad - pV_3(1,2) - (1-p)(l_1 + V_3(2,2)\Big] \\ &\quad + (1-p)\Big[p(s_1 + V_3(2,2)) + (1-p)(s_1 + s_2 + V_3(2,3)) \\ &\quad - pV_3(1,3) - (1-p)(l_1 + V_3(2,3)\Big] \\ &\geq t + p\Big[p(V_3(1,1) - V_3(1,2)) + (1-p)V_3(2,1)\Big] \\ &\quad + (1-p)\Big[ps_1 + (1-p)(s_1+s_2) - pV_3(1,3) - (1-p)l_1\Big] \\ &\geq t + p\Big[p(V_3(1,1) - V_3(1,2)) \\ &\quad + (1-p)(V_3(2,1) - V_3(1,3))\Big] - (1-p)t \end{aligned}$$

$$\geq p\Big[t + p[V_3(1,1) - V_3(1,2)] + (1-p)[V_3(2,1) - V_3(1,3)]\Big]$$

Continuing to expand the above to look ahead for $n-1$ breaks, we get:

$$\begin{aligned}
R_l - R_s &\geq p^{n-2}\Big[t + p[V_n(1,1) - V_n(1,2)] + (1-p)[V_n(2,1) - V_n(1,3)]\Big] \\
&\geq p^{n-2}\Big[t + p[(1-p)l_1 - (1-p)l_1] \\
&\quad + (1-p)[ps_1 + (1-p)(s_1+s_2) - (1-p)l_1]\Big] \\
&\geq p^{n-2}\Big[t + (1-p)(-t)\Big] \\
&\geq p^{n-1}t \\
&\geq 0, \text{ since } t \geq 0,\ p \geq 0
\end{aligned}$$

Therefore $R_l - R_s \geq 0 \implies$ selecting l_1 is profitable when $(1-p)l_1 \geq ps_1 + (1-p)(s_1 + s_2)$

Case 2: Let $(1-p)l_1 < ps_n + (1-p)(s_n + s_{n+1})$ Then

$$\begin{aligned}
R_s - R_l &= ps_1 + (1-p)(s_1+s_2) - (1-p)l_1 \\
&\quad + p[V_2(1,2) - V_2(1,1)] + (1-p)[V_2(1,3) - V_2(2,1)] \\
&\geq ps_1 + (1-p)(s_1+s_2) - (1-p)l_1 \\
&\quad + p\Big[pV_3(1,2) + (1-p)(l_1 + V_3(2,2)) \\
&\quad - p(s_1 + V_3(1,2)) - (1-p)(s_1 + s_2 + V_3(1,3)\Big] \\
&\quad + (1-p)\Big[pV_3(1,3) + (1-p)(l_1 + V_3(2,3)) \\
&\quad - p(s_1 + V_3(2,2)) - (1-p)(s_1 + s_2 + V_3(2,3))\Big] \\
&\qquad \Big[\because V_2(1,i) \geq (1-p)(l_1 + V_3(2,i)) \forall i < n\Big] \\
&\geq 0
\end{aligned}$$

$R_s - R_l \geq 0 \implies$

it is optimal to schedule $s_1 + s_2$ when $(1-p)l_1 < ps_n + (1-p)(s_n + s_{n+1})$.

Case 3: Let $ps_1 + (1-p)(s_1+s_2) = (1-p)l_1 + t$ where $t \geq 0$

$$\begin{aligned}
R_l - R_s &= (1-p)l_1 - ps_1 - (1-p)(s_1+s_2) \\
&\quad + p[V_2(1,1) - V_2(1,2)] + (1-p)[V_2(2,1) - V_2(1,3)] \\
&= -t + p\Big[p(s_1 + V_3(1,2)) + (1-p)(s_1+s_2+V_3(1,3)) \\
&\quad - pV_3(1,2) - (1-p)(l_1 + V_3(2,2))\Big] \\
&\quad + (1-p)\Big[p(s_1+V_3(2,2)) + (1-p)(s_1+s_2+V_3(2,3)) \\
&\quad - pV_3(1,3) - (1-p)(l_1+V_3(2,3))\Big]
\end{aligned}$$

In the above step, note that for $V_2(1,1)$, in break b_2, $(1-p)l_1$ need not exceed $ps_{n-1} + (1-p)(s_{n-1}+s_n)$, however $ps_1 + (1-p)(s_1+s_2) \geq (1-p)l_1$. Since we assume that the policy holds good in subsequent breaks, either Case 2 ((s_1,s_2) preferred) or Case 3 (indifference) will apply, therefore we select (s_1,s_2). The expansion for the other terms is similarly explained.

$$\begin{aligned}
\implies R_l - R_s &= -t + p\Big[t + p[V_3(1,2) - V_3(1,2)] + (1-p)[V_3(1,3) - V_3(2,2)]\Big] \\
&\quad (1-p)\Big[t + p[V_3(2,2) - V_3(1,3)] \\
&\quad + (1-p)[V_3(2,3) - V_3(2,3)]\Big] \\
&= -t + pt + (1-p)t \\
&= 0
\end{aligned}$$

$R_l - R_s = 0 \implies$ selecting l_1 and s_1+s_2 are both equally profitable. □

A.6 Proof of Theorem 3.6

Proof. Let $p_2 l_1 = t + p_1 s_1 + p_2(s_1+s_2)$ where $t \geq 0$.

Case 1:

$$\begin{aligned}
R_l - R_s &= (p_0+p_1)V_2(1,1) + p_2(l_1 + V_2(2,1)) \\
&\quad - p_0V_2(1,1) - p_1(s_1+V_2(1,2)) - p_2(s_1+s_2+V_2(1,3)) \\
&= p_1(V_2(1,1) - s_1 - V_2(1,2)) \\
&\quad + p_2(l_1 + V_2(2,1) - (s_1+s_2) - V_2(1,3)) \\
&= t + p_1[V_2(1,1) - V_2(1,2)] + p_2[V_2(2,1) - V_2(1,3)] \\
&\geq t + p_1\Big[p_0V_3(1,1) + p_1(s_1+V_3(1,2)) + p_2(s_1+s_2+V_3(1,3) \\
&\quad - (p_0+p_1)(V_3(1,2) - p_2(l_1+V_3(2,2)\Big] \\
&\quad + p_2\Big[p_0V_3(2,1) + p_1(s_1+V_3(2,2)) + p_2(s_1+s_2+V_3(2,3) \\
&\quad - (p_0+p_1)(V_3(1,3) - p_2(l_1+V_3(2,3)\Big] \\
&\geq p_0\Big[t + p_1(V_3(1,1) - V_3(1,2)) + p_2(V_3(2,1) - V_3(1,3))\Big]
\end{aligned}$$

Continuing enumeration, we get:

$$\begin{aligned}
R_l - R_s &\geq p_0^{n-2}\Big[t + p_1(V_n(1,1) - V_n(1,2)) + p_2(V_n(2,1) - V_n(1,3))\Big] \\
&\geq p_0^{n-2}\Big[t + p_1(p_1s_1 + p_2(s_1+s_2) - p_2l_1) + p_2(p_1s_1 + p_2(s_1+s_2) - p_2l_1)\Big] \\
&\geq p_0^{n-2}\Big[t + p_1(-t) + p_2(-t)\Big] \\
&\geq p_0^{n-1}t \\
&\geq 0
\end{aligned}$$

$R_l - R_s \geq 0 \implies$ it is optimal to schedule l_1 first.

Case 2 can be similarly proved by showing that $R_s - R_l \geq p_0^{n-1}t \geq 0$.

For both cases above, $R_l > R_s$ and $R_s > R_l$ if $t > 0$ and $p_0 > 0$. There is no interval where $R_l = R_s$, unlike the cases where the number of breaks is fixed. □

Appendix B

Tables

In this section, we present the results of the study of Section 5.6.

B.1 Revenues

Value of Small ad to Large ad = 1:1

Ad Distribution	3:1			2:1			1:1			1:2			1:3			
	Sold	**Earned**	**%**	**Sold**	**Earned**	**%**	**Sold**	**Earned**	**%**	**Sold**	**Earned**	**%**	**Sold**	**Earned**	**%**	Service Level
G	562507	487050	86.59%	526768	446389	84.74%	468266	378272	80.78%	422645	324126	76.69%	400530	301423	75.26%	0.8
OP		499215	88.75%		465490	88.37%		407503	87.02%		360522	85.30%		334538	83.52%	
G	499790	456156	91.27%	467940	421150	90.00%	418699	362125	86.49%	373653	312025	83.51%	355398	292008	82.16%	0.9
OP		470848	94.21%		441146	94.27%		390884	93.36%		345551	92.48%		320642	90.22%	
G	450114	430013	95.53%	422869	399202	94.40%	373440	341916	91.56%	337703	300653	89.03%	319617	282890	88.51%	1
OP		442744	98.36%		415556	98.27%		366165	98.05%		327070	96.85%		304156	95.16%	
G	409715	402782	98.31%	382869	374863	97.91%	342268	327816	95.78%	305894	288663	94.37%	292337	271519	92.88%	1.1
OP		408688	99.75%		382253	99.84%		341317	99.72%		302620	98.93%		285320	97.60%	

Value of Small ad to Large ad = 0.55:1

Ad Distribution	3:1			2:1			1:1			1:2			1:3			
	Sold	**Earned**	**%**	**Sold**	**Earned**	**%**	**Sold**	**Earned**	**%**	**Sold**	**Earned**	**%**	**Sold**	**Earned**	**%**	Service Level
G	372654	311747	83.66%	366942	308653	84.11%	362189	302933	83.64%	360137	291868	81.04%	355506	280061	78.78%	0.8
OP		311766	83.66%		308720	84.13%		303530	83.80%		298844	82.98%		289581	81.46%	
G	331519	303806	91.64%	327327	300211	91.72%	324898	294839	90.75%	318249	280060	88.00%	315820	271274	85.90%	0.9
OP		304306	91.79%		301177	92.01%		297660	91.62%		289513	90.97%		280349	88.77%	
G	297895	290187	97.41%	295353	286524	97.01%	289347	278910	96.39%	287066	268923	93.68%	283672	261446	92.16%	1
OP		291189	97.75%		288230	97.59%		282774	97.73%		276476	96.31%		267598	94.33%	
G	270900	269744	99.57%	267059	265670	99.48%	265620	262879	98.97%	260188	253669	97.49%	260286	250240	96.14%	1.1
OP		270154	99.72%		266293	99.71%		264705	99.66%		257028	98.79%		253500	97.39%	

Value of Small ad to Large ad = 0.50:1

Ad Distribution	3:1			2:1			1:1			1:2			1:3			
	Sold	**Earned**	**%**	**Sold**	**Earned**	**%**	**Sold**	**Earned**	**%**	**Sold**	**Earned**	**%**	**Sold**	**Earned**	**%**	Service Level
G	351843	293129	83.31%	350388	292226	83.40%	350378	293012	83.63%	353217	291053	82.40%	350352	282769	80.71%	0.8
OP		293132	83.31%		292236	83.40%		293066	83.64%		292116	82.70%		284670	81.25%	
G	312939	287301	91.81%	311786	286621	91.93%	313862	287615	91.64%	312619	281689	90.11%	311536	272300	87.41%	0.9
OP		287322	91.81%		286643	91.94%		287836	91.71%		284084	90.87%		274473	88.10%	
G	281754	275030	97.61%	281371	274845	97.68%	280336	273326	97.50%	281636	268788	95.44%	279427	262200	93.83%	1
OP		275066	97.63%		274935	97.71%		273992	97.74%		270985	96.22%		263617	94.34%	
G	256159	255259	99.65%	254232	253425	99.68%	256459	255184	99.50%	255425	251154	98.33%	255463	248140	97.13%	1.1
OP		255300	99.66%		253509	99.72%		255659	99.69%		251969	98.65%		248766	97.38%	

Value of Small ad to Large ad = 0.45:1

Ad Distribution	3:1			2:1			1:1			1:2			1:3			
	Sold	**Earned**	**%**	**Sold**	**Earned**	**%**	**Sold**	**Earned**	**%**	**Sold**	**Earned**	**%**	**Sold**	**Earned**	**%**	Service Level
G	330546	276433	83.63%	332031	278749	83.95%	338309	284475	84.09%	345789	285941	82.69%	345354	279550	80.95%	0.8
OP		276436	83.63%		278755	83.95%		284501	84.10%		285980	82.70%		279576	80.95%	
G	293251	270251	92.16%	295951	272470	92.07%	303227	277522	91.52%	305453	277815	90.95%	306411	270744	88.36%	0.9
OP		270254	92.16%		272475	92.07%		277552	91.53%		277914	90.98%		270860	88.40%	
G	264550	258437	97.69%	266697	261356	98.00%	270413	264983	97.99%	276806	265810	96.03%	275777	259288	94.02%	1
OP		258438	97.69%		261358	98.00%		265007	98.00%		265910	96.06%		259351	94.04%	
G	240346	239698	99.73%	241203	240636	99.76%	247575	246843	99.70%	250021	247082	98.82%	252560	245994	97.40%	1.1
OP		239698	99.73%		240636	99.76%		246860	99.71%		247094	98.83%		246005	97.40%	

Value of Small ad to Large ad = 0.40:1

Ad Distribution	3:1			2:1			1:1			1:2			1:3			
	Sold	**Earned**	**%**	**Sold**	**Earned**	**%**	**Sold**	**Earned**	**%**	**Sold**	**Earned**	**%**	**Sold**	**Earned**	**%**	Service Level
G	310024	260721	84.10%	314426	266555	84.78%	326934	276571	84.60%	339081	280942	82.85%	340170	275459	80.98%	0.8
OP		260722	84.10%		266557	84.78%		276577	84.60%		280943	82.85%		275459	80.98%	
G	274868	253379	92.18%	279775	258561	92.42%	292649	269167	91.98%	299547	272878	91.10%	301994	265616	87.95%	0.9
OP		253380	92.18%		258561	92.42%		269173	91.98%		272879	91.10%		265617	87.95%	
G	247739	242651	97.95%	252752	247337	97.86%	261218	256259	98.10%	269972	260204	96.38%	271632	255966	94.23%	1
OP		242651	97.95%		247337	97.86%		256260	98.10%		260204	96.38%		255966	94.23%	
G	225236	224529	99.69%	228246	227722	99.77%	239049	238394	99.73%	245975	242504	98.59%	248498	241756	97.29%	1.1
OP		224529	99.69%		227722	99.77%		238394	99.73%		242504	98.59%		241756	97.29%	

Value of Small ad to Large ad = 0.35:1

Ad Distribution	3:1			2:1			1:1			1:2			1:3			
	Sold	**Earned**	**%**	**Sold**	**Earned**	**%**	**Sold**	**Earned**	**%**	**Sold**	**Earned**	**%**	**Sold**	**Earned**	**%**	Service Level
G	288319	245652	85.20%	296923	254473	85.70%	315499	268947	85.24%	332214	274394	82.60%	335219	269381	80.36%	0.8
OP		245652	85.20%		254473	85.70%		268947	85.24%		274394	82.60%		269381	80.36%	
G	256171	237395	92.67%	264060	245706	93.05%	282793	261611	92.51%	293474	266182	90.70%	297596	261492	87.87%	0.9
OP		237395	92.67%		245706	93.05%		261611	92.51%		266182	90.70%		261492	87.87%	
G	229753	225361	98.09%	239045	234222	97.98%	252063	247013	98.00%	265018	255561	96.43%	268123	251432	93.77%	1
OP		225361	98.09%		234222	97.98%		247013	98.00%		255561	96.43%		251432	93.77%	
G	210459	209950	99.76%	215751	215153	99.72%	230812	230244	99.75%	240302	236896	98.58%	245440	238007	96.97%	1.1
OP		209950	99.76%		215153	99.72%		230244	99.75%		236896	98.58%		238007	96.97%	

Value of Small ad to Large ad = 0.30:1

Ad Distribution	3:1			2:1			1:1			1:2			1:3			
	Sold	**Earned**	**%**	**Sold**	**Earned**	**%**	**Sold**	**Earned**	**%**	**Sold**	**Earned**	**%**	**Sold**	**Earned**	**%**	Service Level
G	266957	230602	86.38%	279147	243588	87.26%	304171	261839	86.08%	326007	268216	82.27%	330878	265376	80.20%	0.8
OP		230602	86.38%		243588	87.26%		261839	86.08%		268216	82.27%		265376	80.20%	
G	237807	221652	93.21%	247977	232355	93.70%	271416	253118	93.26%	286391	260789	91.06%	293131	257646	87.89%	0.9
OP		221652	93.21%		232355	93.70%		253118	93.26%		260789	91.06%		257646	87.89%	
G	213766	209595	98.05%	223994	220168	98.29%	243056	238571	98.15%	259721	249160	95.93%	263681	247027	93.68%	1
OP		209595	98.05%		220168	98.29%		238571	98.15%		249160	95.93%		247027	93.68%	
G	194713	194133	99.70%	203484	202990	99.76%	223077	222287	99.65%	234956	231532	98.54%	241067	234358	97.22%	1.1
OP		194133	99.70%		202990	99.76%		222287	99.65%		231532	98.54%		234358	97.22%	

B.2 Service Levels

Value of Small ad to Large ad = 1:1

Heuristic	Ad length	**3:1**	**2:1**	**1:1**	**1:2**	**1:3**	Service Level
G	15	100%	100%	100%	100%	100%	0.8
	30	41%	49%	57%	60%	62%	
OP	15	99%	99%	99%	99%	99%	
	30	51%	61%	71%	75%	74%	
G	15	100%	100%	100%	100%	100%	0.9
	30	61%	66%	69%	71%	72%	
OP	15	99%	99%	99%	99%	100%	
	30	75%	81%	85%	87%	85%	
G	15	100%	100%	100%	100%	100%	1
	30	80%	81%	80%	81%	82%	
OP	15	100%	100%	99%	100%	100%	
	30	93%	95%	96%	95%	92%	
G	15	100%	100%	100%	100%	100%	1.1
	30	93%	93%	90%	90%	89%	
OP	15	100%	100%	100%	100%	100%	
	30	99%	100%	99%	98%	96%	

Value of Small ad to Large ad = .55:1

Heuristic	Ad length	**3:1**	**2:1**	**1:1**	**1:2**	**1:3**	Service Level
G	15	88%	90%	94%	99%	100%	
	30	68%	71%	73%	72%	71%	
OP	15	88%	90%	92%	95%	99%	0.8
	30	69%	71%	75%	76%	74%	
G	15	96%	97%	99%	100%	100%	
	30	80%	82%	84%	82%	80%	
OP	15	96%	97%	97%	99%	100%	0.9
	30	81%	84%	86%	87%	84%	
G	15	99%	100%	100%	100%	100%	
	30	93%	93%	94%	90%	89%	
OP	15	99%	99%	99%	100%	100%	1
	30	95%	95%	96%	95%	92%	
G	15	100%	100%	100%	100%	100%	
	30	99%	99%	98%	96%	95%	
OP	15	100%	100%	100%	100%	100%	1.1
	30	99%	99%	99%	98%	96%	

Value of Small ad to Large ad = .50:1

Heuristic	Ad length	**3:1**	**2:1**	**1:1**	**1:2**	**1:3**	Service Level
G	15	80%	81%	82%	92%	99%	
	30	80%	80%	79%	76%	74%	
OP	15	80%	80%	80%	88%	97%	0.8
	30	80%	80%	80%	78%	75%	
G	15	90%	90%	92%	99%	100%	
	30	90%	90%	89%	86%	83%	
OP	15	90%	90%	90%	96%	100%	0.9
	30	90%	90%	90%	88%	84%	
G	15	97%	97%	98%	100%	100%	
	30	97%	97%	96%	93%	91%	
OP	15	97%	97%	97%	100%	100%	1
	30	97%	97%	97%	95%	92%	
G	15	100%	100%	100%	100%	100%	
	30	100%	100%	99%	98%	96%	
OP	15	100%	100%	100%	100%	100%	1.1
	30	100%	100%	100%	98%	96%	

Value of Small ad to Large ad = .45:1

Heuristic	Ad length	**3:1**	**2:1**	**1:1**	**1:2**	**1:3**	Service Level
G	15	72%	71%	69%	84%	97%	
	30	92%	90%	87%	79%	75%	
OP	15	72%	71%	68%	83%	96%	0.8
	30	92%	91%	87%	79%	75%	
G	15	85%	83%	80%	93%	99%	
	30	98%	97%	94%	88%	84%	
OP	15	85%	83%	79%	92%	99%	0.9
	30	98%	97%	95%	89%	84%	
G	15	95%	95%	95%	99%	100%	
	30	100%	100%	99%	95%	92%	
OP	15	95%	95%	94%	99%	100%	1
	30	100%	100%	99%	95%	92%	
G	15	99%	99%	99%	100%	100%	
	30	100%	100%	100%	98%	96%	
OP	15	99%	99%	99%	100%	100%	1.1
	30	100%	100%	100%	98%	97%	

Value of Small ad to Large ad = .40:1

Heuristic	Ad length	**3:1**	**2:1**	**1:1**	**1:2**	**1:3**	Service Level
G	15	66%	63%	58%	81%	96%	
	30	99%	98%	92%	80%	75%	
OP	15	66%	62%	57%	81%	96%	0.8
	30	99%	98%	92%	80%	75%	
G	15	82%	80%	74%	92%	99%	
	30	100%	100%	97%	89%	84%	
OP	15	82%	80%	74%	92%	99%	0.9
	30	100%	100%	97%	89%	84%	
G	15	95%	94%	93%	99%	100%	
	30	100%	100%	99%	95%	92%	
OP	15	95%	94%	93%	99%	100%	1
	30	100%	100%	99%	95%	92%	
G	15	99%	99%	99%	100%	100%	
	30	100%	100%	100%	98%	96%	
OP	15	99%	99%	99%	100%	100%	1.1
	30	100%	100%	100%	98%	96%	

Value of Small ad to Large ad = .35:1

Heuristic	Ad length	**3:1**	**2:1**	**1:1**	**1:2**	**1:3**	Service Level
G	15	66%	61%	57%	81%	96%	0.8
	30	100%	100%	93%	80%	75%	
OP	15	66%	61%	57%	81%	96%	
	30	100%	100%	93%	80%	75%	
G	15	82%	80%	74%	92%	99%	0.9
	30	100%	100%	97%	89%	84%	
OP	15	82%	80%	74%	92%	99%	
	30	100%	100%	97%	89%	84%	
G	15	95%	94%	93%	99%	100%	1
	30	100%	100%	99%	95%	92%	
OP	15	95%	94%	93%	99%	100%	
	30	100%	100%	99%	95%	92%	
G	15	99%	99%	99%	100%	100%	1.1
	30	100%	100%	100%	98%	96%	
OP	15	99%	99%	99%	100%	100%	
	30	100%	100%	100%	98%	96%	

Value of Small ad to Large ad = .30:1

Heuristic	Ad length	**3:1**	**2:1**	**1:1**	**1:2**	**1:3**	Service Level
G	15	66%	62%	55%	82%	96%	0.8
	30	100%	100%	93%	79%	75%	
OP	15	66%	62%	55%	82%	96%	
	30	100%	100%	93%	79%	75%	
G	15	82%	80%	74%	92%	99%	0.9
	30	100%	100%	98%	89%	84%	
OP	15	82%	80%	74%	92%	99%	
	30	100%	100%	98%	89%	84%	
G	15	95%	94%	93%	99%	100%	1
	30	100%	100%	99%	95%	92%	
OP	15	95%	94%	93%	99%	100%	
	30	100%	100%	99%	95%	92%	
G	15	99%	99%	99%	100%	100%	1.1
	30	100%	100%	100%	98%	96%	
OP	15	99%	99%	99%	100%	100%	
	30	100%	100%	100%	98%	96%	

B.3 Utilization of breaks

Value of Small ad to Large ad = 1:1

Ad dist:	3:1	2:1	1:1	1:2	1:3	Service Level
G	96%	93%	89%	85%	84%	0.8
OP	100%	100%	100%	100%	97%	
G	94%	92%	88%	85%	84%	0.9
OP	100%	100%	100%	99%	95%	
G	92%	90%	87%	84%	83%	1
OP	97%	97%	97%	96%	92%	
G	88%	87%	85%	83%	82%	1.1
OP	91%	90%	91%	89%	88%	

Value of Small ad to Large ad = .55:1

Ad dist:	3:1	2:1	1:1	1:2	1:3	Service Level
G	100%	100%	100%	97%	93%	0.8
OP	100%	100%	100%	100%	97%	
G	100%	99%	99%	95%	91%	0.9
OP	100%	100%	100%	99%	95%	
G	97%	97%	95%	93%	90%	1
OP	97%	98%	97%	96%	92%	
G	91%	90%	90%	88%	87%	1.1
OP	91%	91%	91%	90%	88%	

Value of Small ad to Large ad = .50:1

Ad dist:	3:1	2:1	1:1	1:2	1:3	Service Level
G	100%	100%	100%	99%	96%	0.8
OP	100%	100%	100%	100%	97%	
G	100%	100%	100%	98%	94%	0.9
OP	100%	100%	100%	99%	95%	
G	97%	97%	97%	95%	92%	1
OP	97%	97%	97%	96%	92%	
G	91%	91%	91%	90%	88%	1.1
OP	91%	91%	91%	90%	88%	

Value of Small ad to Large ad = .45:1

Ad dist:	3:1	2:1	1:1	1:2	1:3	Service Level
G	100%	100%	100%	100%	97%	0.8
OP	100%	100%	100%	100%	97%	
G	100%	100%	100%	99%	95%	0.9
OP	100%	100%	100%	99%	95%	
G	98%	97%	97%	96%	92%	1
OP	98%	97%	97%	96%	92%	
G	91%	90%	91%	89%	88%	1.1
OP	91%	90%	91%	89%	88%	

Value of Small ad to Large ad = .40:1

Ad dist:	3:1	2:1	1:1	1:2	1:3	Service Level
G	100%	100%	100%	100%	97%	0.8
OP	100%	100%	100%	100%	97%	
G	100%	100%	100%	99%	95%	0.9
OP	100%	100%	100%	99%	95%	
G	97%	97%	97%	96%	92%	1
OP	97%	97%	97%	96%	92%	
G	91%	90%	91%	90%	88%	1.1
OP	91%	90%	91%	90%	88%	

Value of Small ad to Large ad = .35:1

Ad dist:	3:1	2:1	1:1	1:2	1:3	Service Level
G	100%	100%	100%	100%	97%	0.8
OP	100%	100%	100%	100%	97%	
G	100%	100%	100%	99%	95%	0.9
OP	100%	100%	100%	99%	95%	
G	97%	97%	97%	96%	92%	1
OP	97%	97%	97%	96%	92%	
G	91%	91%	91%	90%	88%	1.1
OP	91%	91%	91%	90%	88%	

Value of Small ad to Large ad = .30:1

Ad dist:	3:1	2:1	1:1	1:2	1:3	Service Level
G	100%	100%	100%	100%	97%	0.8
OP	100%	100%	100%	100%	97%	
G	100%	100%	100%	99%	95%	0.9
OP	100%	100%	100%	99%	95%	
G	98%	97%	97%	96%	92%	1
OP	98%	97%	97%	96%	92%	
G	91%	91%	91%	90%	88%	1.1
OP	91%	91%	91%	90%	88%	