Worst-Case Complexity Analyses for the Dobson-Kalish Optimal Pricing Algorithm and its Relatives

by

Derya Demirtaş

An essay
presented to the University of Waterloo
in fulfillment of the
essay requirement for the degree of
Master of Mathematics
in
Combinatorics and Optimization

Waterloo, Ontario, Canada, 2010

I hereby declare that I am the sole author of this essay. This is a true copy of the essay, including any required final revisions, as accepted by my examiners.

I understand that my essay may be made electronically available to the public.

Abstract

Revenue Management is one of the most active applied areas in Operations Research. Optimal pricing problem is one of the main mathematical optimization problems to be considered in this area. This problem is NP-hard. One of the most visible heuristic algorithms for this problem was proposed by Dobson and Kalish in the 1980s. In this essay, we focus on the worst-case behaviour of the Dobson-Kalish algorithm. In addition to this original version we also consider a modified version of the algorithm. To analyze the worst-case behaviour of these algorithms we express their main steps, with respect to a pattern that we choose, as a system of inequalities where the variables represent the data for the original optimal pricing problem. For the modified Dobson-Kalish algorithm our system of inequalities can be analyzed by linear optimization techniques. We discuss how to use this approach to prove lower bounds as well as upper bounds on the worst-case behaviour of the Dobson-Kalish algorithm.

Acknowledgements

I would like to thank Levent Tunçel for his excellent supervision. I would also like to thank my reader Ricardo Fukasawa for many helpful suggestions and comments. As well, thank-you to my family for all their continued support.

Contents

List of Figures

Chapter 1

Introduction

In today's business environment, any organization that wants to stay competitive needs to find new and better ways to manage its resources, products and people. Complexity of the processes and dynamic structure of the markets make the effective management even harder. Operations Research, the discipline of applying advanced analytical methods to help make better decisions, is a great tool to tackle business problems. By using techniques from many fields including optimization, statistics and economics, Operations Research analyzes complex systems and increases their efficiency.

The Operations Research discipline was born during the Second World War. During the war, both British and US army formed science teams to study the tactics and strategies of military operations. These teams had remarkable success in supporting the military effort and made significant contribution in areas such as radar systems, anti-submarine warfare and bombing strategy.

After its success during the war, Operations Research began to be applied to the industrial environment. Major industries, including energy, airline, manufacturing, transportation and telecommunications, established Operations Research groups to help solving their strategic business problems. The Operations Research field has grown enormously since then. Many mathematical methodologies have been developed and adapted to solve real-life problems. Operations Research has gotten even more promising with the use of optimization techniques and the increasing computational power.

One of the most well known success stories of Operations Research is the practice of revenue management in airline industry. In late 1970s, many low-cost airlines entered in the market and became strong competitors to big airline companies. American Airlines (AA) realized that they could not afford cutting prices further and needed to find another strategy to stay profitable. Finally, they came up with a solution based on purchase restrictions and capacity controlled fares. Specifically, they forecasted the number of business and

leisure customers that would take each flight and decided on the number of discounted tickets accordingly. To prevent high-paying business customers from buying these low price tickets, AA introduced purchase restrictions such as in advance purchase, length of stay requirements and strict refund policies. They implemented this strategy using computerized systems and gained a big success. AA estimates that these systems generated 1.4 billion dollars in additional incremental revenue over a three year period [8]. In fact, in 1991, INFORMS awarded the Franz Edelman Award to AA for the Best Operations Research Project in the World based on this revenue management practice. As a result of this history, revenue management has been recognized as an important field of Operations Research and rapidly spread into other industries.

Revenue management (RM), also known as yield management, is the practice of making sales decisions using analytical methods in order to maximize revenue or profit. Sales decisions that are addressed by revenue management can be classified in two categories [9]. Structural sales decisions are the ones concerned with a company's long term sale strategy: what sales channels to use, which selling formats to use (such as auctions, posted prices or negotiations); which segmentation or differentiation mechanisms to use; which terms of trade to offer, etc. Price and quantity decisions, in contrast, are taken more often and can be changed more easily. "How" questions are usually decided in this category: how a product should be priced in the different channels; how prices should be adjusted in time and due to seasonality; how to price across product categories; how to discount over the product lifetime, how to allocate output or capacity to different segments, products or channels, etc.

The nature of the business mainly determines the importance and relevance of these decisions. In many service industries, such as airlines, car rental companies, and hotels, quantity related strategies are more applicable than price related strategies. For instance, it is very common in airline industry, where RM emerged, to allocate different amounts of seats to different customer segments (such as business and discretionary travelers) and limit number of discount seats sold on each flight. In fact, the computer reservation systems enable airline companies to make these allocation decisions in real time. On the other hand, retailers cannot change available quantities as easily during the period since inventory decisions are usually made at the beginning of the period. Therefore, they use price as a tool to have more control on the demand.

Revenue management methodology and tools differ depending on whether quantity or price-based strategies are used. Thus, RM is usually separated into two broad categories, namely quantity-based revenue management and price-based revenue management.

Quantity-based revenue management usually requires that different customer segments are willing to pay a different price for the same product. For example, in airline industry case, "customer segments" refers to business customers and leisure/ discretionary customers. While the latter segment is very price sensitive, the former one values time

flexibility and service quality more. The first airline company that used revenue management, American Airlines, utilized these differences between their customers to compete with low-cost carriers.

In contrast to quantity-based revenue management, price-based RM can be applied in any business where changing the prices frequently is feasible and inexpensive. In price-based RM, all the customers pay the same price for a particular product and demand is controlled by price adjustments. The aim is to price the products so as to maximize the revenue.

In this study, we are interested in the following price-based RM problem: A company produces m different products and its competitors produce similar products in the same market. The market is composed of n distinct customer segments where all the customers in a particular segment share similar preferences and needs. We assume that the customers are rational and hence, try to maximize their own utility.

Our goal is to determine optimal prices for each product to maximize the total profit. There are two main challenges in pricing the products. The first challenge is the price war against competitors: Those products that are to be bought should be a better deal for some customers than competitors' products. The second one is the competition between our own products. We should not let a low-profit product steal the demand for a high-profit product and decrease the total profit. The phenomenon that a product of a company takes the market share away from another product of the same company is known as "cannibalization." To maximize the profit, we should set the prices in a way that a low-profit product never cannibalizes a high-profit product. We will explain how our model takes care of this "cannibalization" challenge after we define our mathematical model (1.8).

Let reservation price of segment i for product j, $R_{ij} \geq 0$, $i = 1, \ldots, n$, $j = 0, \ldots, m$, be the maximum amount that segment i is willing and able to pay for product j. Also, let π_j, v_j and f_j denote the price, variable cost and fixed cost of product j respectively. Note that, product 0 is created to deal with "no purchase" case. If a segment does not buy any product, it is assigned to product 0 where $\pi_0 = 0, v_0 = 0, f_0 = 0$ and $R_{i0} = 0 \ \forall i$.

Customer surplus of segment i for product j is defined as the difference between the reservation price and product price, i.e. $R_{ij} - \pi_j$. Moreover, CS_i represents the maximum surplus of segment i across all competitor products. Then, "rational customer" assumption requires that each customer will buy the product that maximizes his/her consumer surplus and does not buy any product if his/her surplus is nonpositive for all products. In other words, segment i will buy product $j \neq 0$ only if:

$$j = \underset{k=1,\ldots,m}{\operatorname{argmax}} \left\{ R_{ik} - \pi_k \right\}, \tag{1.1}$$

$$R_{ij} - \pi_j > CS_i, \tag{1.2}$$

and

$$R_{ij} - \pi_j > 0. \tag{1.3}$$

If we make the relations in (1.2) and (1.3) "$\geq$" instead of "$>$" then the model we construct will be overly optimistic in the sense that whenever there is a tie for the maximum surplus of a customer segment, an optimal solution will assign that customer segment to a product that maximizes our company's total profit. However, even the strict inequality case would have problems in practice since R_{ij}'s are usually estimated as a result of market studies and they may be thought of as the mean values of the corresponding distributions.

There is a solution to these issues in the literature. Shioda et al. [7] introduce a utility tolerance $\delta_i > 0$ for customer segment i and require the winning product to beat every other product by at least a margin of δ_i. However, in this essay we focus on the original model studied by Dobson and Kalish and others which used "$\geq$". Therefore, (1.1), (1.2) and (1.3) become segment i will buy product $j \neq 0$ only if:

$$j = \underset{k=1,\ldots,m}{\operatorname{argmax}} \{R_{ik} - \pi_k\}, \tag{1.4}$$

$$R_{ij} - \pi_j \geq CS_i, \tag{1.5}$$

and

$$R_{ij} - \pi_j \geq 0. \tag{1.6}$$

We can preprocess the data such that

$$R_{ij} \leftarrow \max(0, R_{ij} - CS_i),$$

and

$$CS_i \leftarrow 0.$$

In the above, we may assume $CS_i \leq R_{ij}$ (otherwise $CS_i > R_{ij}$ and customer segment i is never interested in any of our products even if we set our prices to 0; so in this case we can delete customer segment i). Then statements (1.4), (1.5) and (1.6) simplify to

$$\sum_{j=0}^{m} (R_{ij} - \pi_j) x_{ij} \geq R_{ik} y_k - \pi_k, 0 \leq \pi_k \qquad \forall i, \forall k, \tag{1.7}$$

where

$$x_{ij} := \begin{cases} 1, & \text{if customer segment } i \text{ buys product } j, \\ 0, & \text{otherwise,} \end{cases}$$

and

$$y_j := \begin{cases} 1, & \text{if product } j \text{ is offered,} \\ 0, & \text{otherwise.} \end{cases}$$

To justify the equivalence, we use the fact that $0 \leq CS_i \leq R_{ij}$ and the dummy product $k = 0$.

In this problem, we further make the following assumptions:

1. Customer segments are formed such that reservation prices are the same for each customer in a given segment.
2. Customer behaviour is fully determined by reservation prices and product prices and not affected by anything else.
3. Every customer segment pays the same price for each product.
4. Each customer segment buys at most one type of product and each customer buys at most one unit of a product.
5. Competitors do not react to our prices.
6. There is no production capacity constraint.

This problem can be formulated as a nonlinear mixed integer programming (MIP) problem [2]. The parameters employed in the mathematical formulation are:

i	customer segments ($i = 1 \dots n$),
j	product lines ($j = 0 \dots m$),
v_j	variable cost of product j,
f_j	fixed cost of product j,
N_i	number of customers in segment i,
R_{ij}	reservation price of segment i for product j,

and the decision variables are:

π_j	price of product j,
$x_{ij} := \begin{cases} 1, & \text{if customer segment } i \text{ buys product } j, \\ 0, & \text{otherwise,} \end{cases}$	
$y_j := \begin{cases} 1, & \text{if product } j \text{ is offered,} \\ 0, & \text{otherwise.} \end{cases}$	

The nonlinear MIP formulation is:

$$\max \sum_{i=1}^{n}\sum_{j=0}^{m} N_i(\pi_j - v_j)x_{ij} - \sum_{j=1}^{m} f_j y_j, \tag{1.8}$$

$$\begin{aligned}
\text{s.t.} \quad \sum_{j=0}^{m}(R_{ij} - \pi_j)x_{ij} &\geq R_{ik}y_k - \pi_k, \forall i,k,\\
\sum_{j=0}^{m} x_{ij} &= 1, \forall i,\\
x_{ij} &\leq y_j, \forall i,j,\\
\pi_0 &= 0,\\
\pi_j &\geq 0, \forall j,\\
x_{ij}, y_j &\in \{0,1\}, \forall i,j.
\end{aligned}$$

In this model, the objective function maximizes the total profit subject to six groups of constraints. The first group of constraints is simply the inequality (1.7) which insures that a particular segment maximizes its surplus by choosing a product over other products offered. It also insures the nonnegativity of the surplus since the right hand side is 0 for $k = 0$. Note that this group of constraints is inactive when $y_k = 0$ s.t. $k \neq 0$. The second group guarantees that a given segment never buys more than one product. It is an "=" sign instead of "$\leq$" as buying product 0 represents not buying any real product. The third group enforces that a product has to be produced if it is assigned to a customer. The fourth group of constraints sets the price of the null product to 0. The fifth one ensures the nonnegativity of the variable π_j while the last one ensures that the binary variables x_{ij} and y_j are restricted to 0,1. Note that the nonlinear MIP maximizes the total profit made by the company and at an optimal solution of (1.8) no cannibalization can occur.

To simplify the above formulation, we modify the data once more such that

$$R_{ij} \leftarrow R_{ij} - v_j,$$

$$\pi_j \leftarrow \pi_j - v_j,$$

and

$$v_j \leftarrow 0.$$

Therefore, the objective function changes to $\sum_{i=1}^{n}\sum_{j=0}^{m} N_i\pi_j x_{ij} - \sum_{j=1}^{m} f_j y_j$ while rest of the formulation and solution set remain the same. Note that the first constraint is still inactive when $y_k = 0$ s.t. $k \neq 0$ although now it forces $\pi_k \geq v_k, \forall k$ s.t. $y_k = 0$. However, this does not bring any additional limitation because there is no constraint on the price of a non-offered product and hence, it can be set to any value.

It is shown that problem (1.8) is $\mathcal{NP}$-hard [3] (see proof in Appendix A). It is also $\mathcal{APX}$-hard [4].

This model can be linearized by introducing continuous auxiliary variables p_{ij} [7]. See also [5] for a similar formulation for optimal bundle pricing problem. Let the auxiliary variable be:

$$p_{ij} = \begin{cases} \pi_j, & \text{if } x_{ij} = 1, \\ 0, & \text{otherwise.} \end{cases}$$

This is guaranteed by the following constraints:

$$\begin{aligned} p_{ij} &\geq 0, \\ p_{ij} &\leq R_{ij} x_{ij}, \\ p_{ij} &\leq \pi_j, \\ p_{ij} &\geq \pi_j - \bar{R}_j(1 - x_{ij}), \end{aligned} \tag{1.9}$$

where

$$\bar{R}_j := \max_i \{R_{ij}\}.$$

The corresponding linearized model is:

$$\max \sum_{i=1}^{n} \sum_{j=0}^{m} N_i p_{ij} - \sum_{j=1}^{m} f_j y_j, \tag{1.10}$$

$$\begin{aligned} \text{s.t.} \quad \sum_{j=0}^{m} (R_{ij} x_{ij} - p_{ij}) &\geq R_{ik} y_k - \pi_k, \forall i, k, \\ R_{ij} x_{ij} - p_{ij} &\geq 0, \forall i, j, \\ \sum_{j=0}^{m} x_{ij} &= 1, \forall i, \\ x_{ij} &\leq y_j, \forall i, j, \\ \pi_0 &= 0, \\ p_{ij} &\leq \pi_j, \forall i, j, \\ p_{ij} &\geq \pi_j - \bar{R}_j(1 - y_{ij}), \forall i, j, \\ x_{ij}, y_j &\in \{0, 1\}, \forall i, j, \\ \pi_j, p_{ij} &\geq 0, \forall i, j. \end{aligned}$$

Note that $\pi_j \geq 0$, $\forall j$ implies in the old variables $\pi_j \geq v_j$ (the price is at least the cost) which is a reasonable restriction.

Because of the difficulty of solving the problem, various heuristics have been proposed. One of the most efficient heuristics is developed by Dobson and Kalish [2]. They first formulated the problem assuming assignments are given and used the dual of this linear programming problem to design the heuristic.

Now, consider the problem where segment assignments to products are known in advance (i.e., x_{ij} and y_j are given). From now on, we will refer to it as "price setting subproblem." Let C_j denote the set of customer segments who buy product j, B denote the set of products that are bought at least by one segment, i.e. $B = \{j : C_j \neq \emptyset, j \neq 0\}$, and M_j denote the total number of customers who buy product j, i.e. $M_j = \sum_{i \in C_j} N_i$. Then the model (1.10) simplifies to the following linear programming model:

$$\begin{aligned} \max \quad & \sum_{j=1}^{m} M_j \pi_j - \sum_{j \in B} f_j, \\ \text{s.t.} \quad & R_{ij} - \pi_j \geq R_{ik} - \pi_k, \quad \forall j,k \in B, \forall i \in C_j, \\ & R_{ij} - \pi_j \geq 0, \quad \forall j \in B, \forall i \in C_j, \\ & \pi_0 = 0, \\ & \pi_j \geq 0, \quad \forall j. \end{aligned} \tag{1.11}$$

Since it is a maximization problem, π_j are always nonnegative in the optimal solution. Therefore, we can remove the nonnegativity constraint from the model. Moreover, given the assignments, fixed cost portion of the objective function becomes constant and hence, can be removed. With some additional arrangement, it further simplifies to:

$$\begin{aligned} \max & \sum_{j=1}^{m} M_j \pi_j, \\ \text{s.t.} \quad & \pi_j - \pi_k \leq \min_{i \in C_j} \{R_{ij} - R_{ik}\}, \forall j \in B, \forall k \in B \backslash \{j\}, \\ & \pi_0 = 0, \\ & \pi_j \leq \min_{i \in C_j} \{R_{ij}\}, \forall j \in B. \end{aligned} \tag{1.12}$$

In this model, the objective function maximizes the revenue subject to three groups of constraints. The first group of constraints insures that a particular customer segment maximizes its surplus by choosing product j over other products. The second group of constraints simply sets price of the null product to 0. The last group enforces that a customer segment never buys a product having a higher price than its reservation price for that product.

The dual of (1.12) is:

$$\begin{aligned} \min \quad & \sum_{j,k} r_{jk}\omega_{jk} + \sum_{j} \sigma_j w_j, && (1.13)\\ \text{s.t.} \quad & \sum_{k\neq j} \omega_{jk} - \sum_{k\neq j} \omega_{kj} + w_j = M_j, \quad \forall j \in B, \\ & \sum_{k\neq j} \omega_{kj} = 0, \quad \forall j \notin B, \\ & w_0 = 0, \\ & \omega_{jk}, w_j \geq 0, \quad \forall j,k, \end{aligned}$$

where

$$r_{jk} := \min_{i\in C_j} \{R_{ij} - R_{ik}\}$$

and

$$\sigma_j := \min_{i\in C_j} \{R_{ij}\}.$$

In fact, we can simplify this formulation by removing all $\omega_{kj}, j \notin B$ since the second constraint implies that $\omega_{kj} = 0, \forall j \notin B$ together with the nonnegativity constraint. In addition, we can add a single redundant constraint to convert (1.13) into a set of $|B|$ shortest path problems:

$$\begin{aligned} \min \quad & \sum_{j,k} r_{jk}\omega_{jk} + \sum_{j} \sigma_j w_j, && (1.14)\\ \text{s.t.} \quad & \sum_{k\neq j} \omega_{jk} - \sum_{k\neq j} \omega_{kj} + w_j = M_j, \quad \forall j \in B, \\ & -\sum_{j} w_j = -\sum_{j} M_j, \\ & \omega_{jk}, w_j \geq 0, \quad \forall j \in B, k \in B. \end{aligned}$$

In the corresponding network in Figure 1.1, there exists a node for each product $j \in B$ and one dummy node (node 0) to represent "no products." The arc from node k to j has cost $r_{jk} = \min_{i\in C_j} \{R_{ij} - R_{ik}\}$ while the arc from node 0 to j has cost $\sigma_j = \min_{i\in C_j} \{R_{ij}\}$. Node 0 is the source node and the problem is equal to finding the shortest path from node 0 to every node $j \in B$. Thus, optimal price of product j is the length of the shortest path from node 0 to node j.

Let's see the structure of the problem on an example:

Example 1.1. *Consider the case that there are two products and three segments, i.e.* $m = 2$ *and* $n = 3$. *Let* $N_1 = N_2 = N_3 = 1$ *and the reservation prices be:*

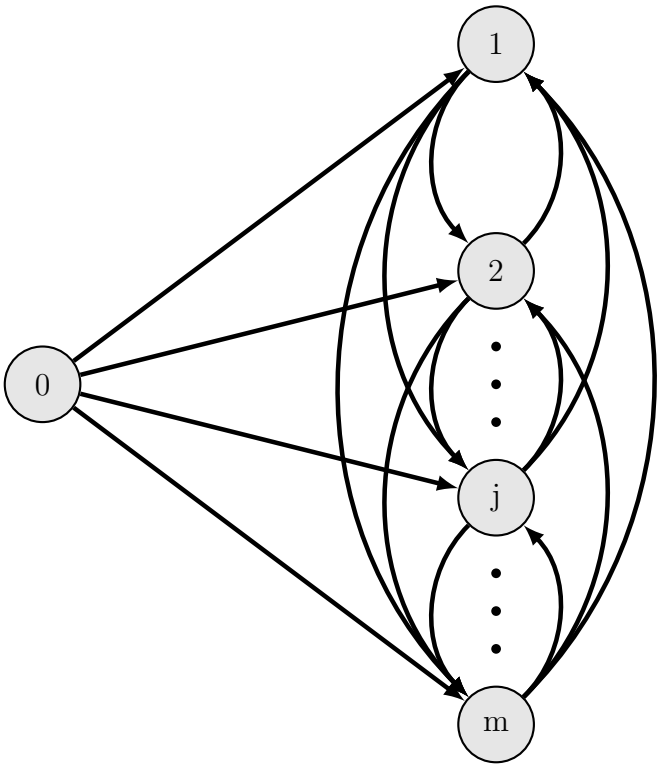


Figure 1.1: Underlying network of problem(1.14).

R_{ij}	*Product A*	*Product B*
Segment 1	*100*	*60*
Segment 2	*120*	*180*
Segment 3	*110*	*130*

Suppose that segment 1 *is assigned to product A and segment* 2 *and* 3 *are assigned to product B. Then, the network of this particular problem is as in Figure 1.2: Given the assignments, optimal price of product A is* 100*, the length of the shortest path from 0 to A. Similarly, optimal price of product B is* 120*. Thus, total revenue is* 340*. Surpluses of segments* 1*,* 2 *and* 3 *are* $100 - 100 = 0$*,* $180 - 120 = 60$ *and* $130 - 120 = 10$ *respectively. Note that, if segment* 3 *bought product A, its surplus would be still* 10 *(*$110 - 100 = 10$*). Therefore, if the price of product B is increased, segment 3 would switch to product A. Notice that, segment* 3 *is the segment determining the length of arc from A to B.*

In fact, this situation is not a coincidence and stated in the following proposition:

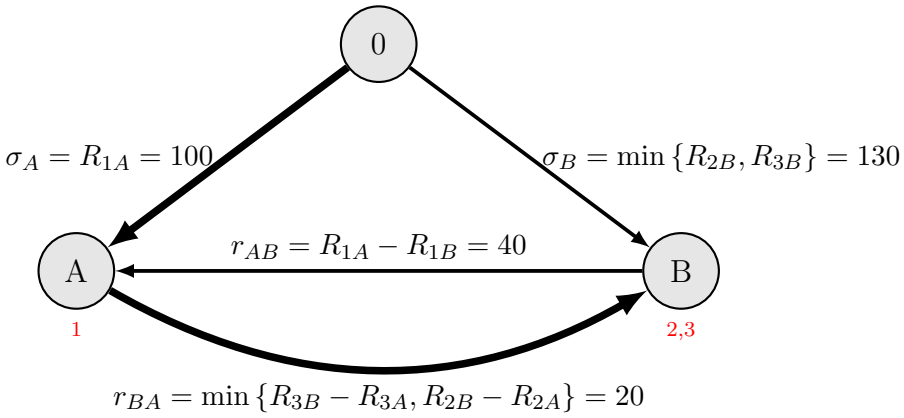


Figure 1.2: Underlying network of example 1.1.

Proposition 1.1. *If product j is the child of product k in the solution of the shortest paths problem above, then some segment i^* assigned to j prevents us from raising its price. In particular,*

$$i^* \in \underset{i \in C_j}{\operatorname{argmin}} \{R_{ij} - R_{ik}\}$$

We will refer to this segment as "critical segment" from now on.

Proof. If product j is the child of product k in the solution tree, then arc kj is part of the tree and hence, $\omega_{kj} = 1$. By complementary slackness, $\pi_j - \pi_k = \min_{i \in C_j} \{R_{ij} - R_{ik}\} = R_{i^*j} - R_{i^*k}$. Thus, $\pi_j - R_{i^*j} = \pi_k - R_{i^*k}$ which means that if π_j is increased, segment i^* would switch to product k. □

Since the price setting subproblem is equal to $|B|$ shortest path problems, its worst-case running time is $O(m^2)$, given the assignments. With n customer segments and m products, number of different possible assignments is $(m+1)^n$ since each customer either buys one of m products or does not buy at all. Although the price setting subproblem itself is solvable in polynomial time, solving it for each assignment obviously has exponential complexity. Instead, there is a more efficient heuristic found by Dobson and Kalish [2]. The heuristic, called Dobson-Kalish Reassignment Heuristic, starts with a feasible initial assignment, solves $|B|$ shortest path problems, finds optimal prices consistent with the assignment and then begins iterating, where at each iteration one critical segment is reassigned or deleted, and optimal prices are computed. As it is stated at Proposition 1.1, critical segments are the segments that constrains us from charging higher prices. Therefore, assigning a critical segment to another product or removing it completely would result an increase in the revenue.

For instance, in example 1.1, the critical segment is segment 3. If we move that segment from child node, node B, to the parent node, node A, the new graph and costs are as in Figure 1.3:

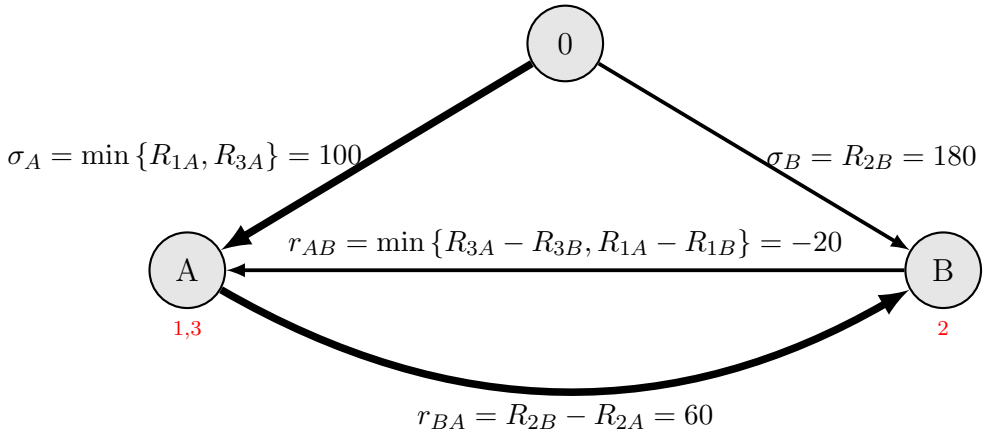


Figure 1.3: Reassignment of Segment 3.

As it can be seen in the figure, the shortest path tree remained the same while costs of the arcs are changed. Based on the figure, optimal prices of product A and B are 100 and 160 respectively. Thus, total revenue is 360. The reassignment resulted a 20 units increase in the total revenue.

On the other hand, the reassignment of a critical segment does not necessarily increase the revenue. It is possible that the reassigned segment becomes critical for the product which it is assigned to and restrains the firm from increasing the price of that product. Therefore, the heuristic needs to compare the revenue before to the revenue after for each reassignment. In fact, the Dobson-Kalish Reassignment Heuristic evaluates all possible critical segment reassignments at each iteration and picks the most profitable one. Then, the network is updated based on the new assignments and the shortest paths problem is solved on the updated network. This procedure is repeated until none of the reassignments increases the revenue. See Algorithm 1.1 for the formal description of the heuristic algorithm.

Now the question is how to find a feasible initial assignment. A simple yet good heuristic is *Maximum Reservation Price Heuristic* [7]. This heuristic assigns each segment to the product that it "wants" the most. It is very efficient since it generates a product-segment assignment easily and guarantees its feasibility since all the arc costs in the shortest path network will be nonnegative. Therefore, we will use MaxR to produce an initial assignment for the price setting subproblem.

Algorithm 1.1 Dobson-Kalish Reassignment Heuristic

Require: a feasible product-segment assignment and its corresponding optimal spanning tree solution from solving (1.14)

repeat
  for all products/nodes j where $C_j \neq \emptyset$ **do**
    Suppose arc (k, j) is in the spanning tree solution.
    if $k \neq 0$ **then**
      For every i^* such that $i^* = \arg\min_{i \in C_j}\{R_{ij} - R_{ik}\}$, reassign Segment i^* to product/node k.
    else
      For every i^* such that $i^* = \arg\min_{i \in C_j} R_{ij}$, delete Segment i^* (i.e., Segment i^* buys no products).
    end if
    Resolve the shortest path problem on the new network and record the change in the objective value.
    Restore the original network.
  end for
  Perform the reassignment that resulted in the maximum increase in the objective value. Resolve shortest path problems and update the optimal spanning tree.
until no reassignment improves the objective value.

Maximum Reservation Price Heuristic (MaxR)

Set $C_j = \{i : j = \arg\max_k\{R_{ik}\}\}, \forall j$.
Solve shortest path problems on the network defined by $C_j, \forall j$.

Dobson and Kalish [2] claim that the complexity of the Dobson-Kalish Reassignment Algorithm is $O(m^4 n)$. To show this, they claim that each segment can be reassigned at most m times before it is eliminated (assigned to 0). However, Shioda et al. [7] showed that the algorithm can reassign a customer segment more than m times. This result made the complexity of the Dobson-Kalish algorithm into an open problem.

In this study, we present a modified version of the Dobson-Kalish Reassignment Algorithm and discuss the worst-case running time of the Dobson-Kalish algorithm through the modified version. We use an innovative approach to complexity analysis, namely use linear programming to create examples proving the existence of worst-case instances. In Chapter 2, we discuss the modified Dobson-Kalish Reassignment Algorithm and provide a lower bound for the worst-case complexity of the algorithm. In Chapter 3, we study the original Dobson-Kalish Reassignment Algorithm and present a system of inequalities guaranteeing the worst-case behavior. Chapter 4 is a brief discussion of minimally infeasible systems and upper bounds on the number of reassignments. Finally, Chapter 5 summarizes our

findings and describes our current and future research.

Note that, in problem (1.8), we make some assumptions which might look restrictive at a first sight (see page 5). However they are not as limiting as they seem:

The first assumption states that the reservation prices are the same for every customer in each segment. Since there is no limit on the number of segments, separate customer segments can be formed for different reservation prices. The extreme case is that no pair of customers has the same reservation price and hence, a segment is formed for each customer.

The second assumption is quite common in many models based on reservation prices. In this study, we are interested in solving optimal pricing problem based on reservation prices. One can find more information about customer choice models in [9].

The third one is non-differentiated pricing assumption. This is not a restrictive assumption either. If a company wants to use price differentiation mechanisms based on time of purchase, customer profile, location and so on, this can be realized by replicating a product many times and making required changes to the reservation prices. A typical example is the airline industry. Often, airline companies charge different prices to different customers for the same type of seat and service on the same flight. For instance, a refundable flight ticket is usually more expensive than a nonrefundable one. The same ticket is cheaper if it is purchased at least one month in advance. Some discounts require a Saturday night stay or a minimum of two nights stay. All these price incentives violate the non-differentiated pricing assumption. However, this can be overcome by defining a different product for each single and combination of price incentives. Let's suppose that airline company A offers two price incentives for economy class travelers: advance purchase and Saturday night stay. Then, we need four types of economy class tickets instead of one: advance purchase with Saturday night stay, advance purchase without Saturday night stay, normal purchase with Saturday night stay and normal purchase without Saturday night stay.

Assumption 4 is the unit demand assumption. Similar to the previous approach, if a customer is willing to buy more than one product and/or unit, this can be handled by replicating the customer many times and making necessary changes to the reservation prices.

Assumption 5, static competition assumption, is possibly the most restricting one among all. We need this assumption to keep the model easy to solve. However, even this one is not as limiting as it seems. If we run the model frequently enough with the updated competitor prices and reservation prices, it might still give a good solution. In fact, in [7], the authors test the MIP formulations of the optimal pricing problem similar to (1.10) on real data sets. They run the Dobson-Kalish heuristic against a case with $n = 3095$ and $m = 2274$ and find a solution within 7.05% optimality in about an hour and a half. The prices obtained therefrom are used as an initial feasible solution (possibly with different assignments x_{ij}) for the MIP when the price of a competitor product is changed.

Updating the solution takes only 55.252 seconds of CPU time when the price change causes the competitor surplus to decrease by \$200 for 10 customer segments. They also test the case where the competitor surplus increases by \$200 for 10 customer segments due to the change in the price of a competitor product. Updating the solution in this case takes only 18.304 seconds of CPU time.

The last assumption, no capacity constraint assumption may not be realistic in all cases. For example, airline ticket optimal pricing problem is always capacitated by nature. Usually, the model of the airplane—hence the capacity—is determined before the tickets go on sale and it is not changed according to the demand. Therefore, capacity constraints need to be added to the mixed integer programming model. For instance, suppose the total seating capacity of a plane is 130; 22 seats in the first class and 108 seats in the economy class. Let product 1 denote the first class tickets and product 2 denote the economy class tickets. Then, adding the following inequalities guarantees that the capacity is not exceeded:

$$\sum_{i=1}^{n} N_i x_{i1} \leq 22,$$

and

$$\sum_{i=1}^{n} N_i x_{i2} \leq 108.$$

In fact, even if the airline company uses price differentiation, capacity restrictions can be included in the model. Remember the airline company A example where the corresponding model includes four different products to represent economy class tickets with different price incentives. To incorporate the above capacity constraints in this model, we enforce the following linear constraints:

$$\sum_{i=1}^{n} N_i x_{i1} \leq 22,$$

and

$$\sum_{i=1}^{n} N_i (x_{i2} + x_{i3} + x_{i4} + x_{i5}) \leq 108.$$

Note that, introducing capacity constraints does not affect the complexity of the model. However, it would require significant changes to the Dobson-Kalish algorithm. For a more general discussion of assumptions/ axioms of various revenue management models, see [10].

Chapter 2

Modified Dobson-Kalish Analysis (m=3)

In this chapter, we study a modified version of the Dobson-Kalish Reassignment Algorithm to simplify the analysis and to improve our understanding of the related algorithms and structures. After computing the shortest path tree for the given assignment, the *modified algorithm* simply chooses any (not necessarily the most profitable) reassignment of a customer segment to its parent node if it leads to a strict improvement in the objective function value. If no such improving reassignment exists, the algorithm stops.

In this chapter, we further focus on the special case of three products and a single customer in each segment (i.e. $m = 3$ and $N_i = 1$, $\forall i$). The goal is to prove that there exist instances of the optimal pricing problem (1.8) on which the modified Dobson-Kalish algorithm performs $\Omega(n^2)$ reassignments of the customer segments.

We want the segments initially assigned to the products so that the Hamming distances of all segments from node 0 are $m = 3$ except the segments 1 and n.

Hamming distance from node 0	:	1	2	3
Number of segments at each distance	:	[1,	1,	n-2]

Moreover, we force the pattern for the Hamming distance of the number of customer segments to change as follows:

$$
\begin{array}{lll}
[\ 1, & 1, & n\text{-}2\ ] \\
[\ 1, & 2, & n\text{-}3\ ] \\
\ldots & & \\
[1, & n\text{-}2, & 1] \\
[2, & 1, & n\text{-}3] \\
[2, & 2, & n\text{-}4] \\
\ldots & & \\
[2, & n\text{-}3, & 1] \\
[3, & 1, & n\text{-}4] \\
[3, & 2, & n\text{-}5] \\
\ldots & & \\
\ldots & & \\
[n\text{-}2, & 1, & 1]
\end{array}
$$

We call this sequence of reassignments *Quadratic Growth Reassignment (QGR)* pattern. Whether the establishment of this pattern change is possible or not, helps us prove bounds for the worst-case complexity.

To create the worst-case instance, we treat the data of the optimal pricing problem (1.8) as set of variables. Since $N_i = 1, \forall i$, our variables are $R_{ij}, i \in \{1, 2, \ldots, n\}, j \in \{1, 2, 3\}$. For convenience, we further define

$$
\begin{array}{rcl}
\alpha_i & := & R_{i2} - R_{i1}, \forall i \in \{1, 2, \ldots, n\}, \\
\beta_i & := & R_{i3} - R_{i1}, \forall i \in \{1, 2, \ldots, n\}, \\
\gamma_i & := & R_{i3} - R_{i2}, \forall i \in \{1, 2, \ldots, n\}.
\end{array} \tag{2.1}
$$

Initially, segment n is assigned to product 1, segment 1 is assigned to product 2 and segments 2,3,...,n-1 are assigned to product 3. In terms of linear inequalities it suffices to have:

$$
\begin{array}{rcl}
\alpha_1 & \geq & 0, \\
0 & \geq & \alpha_n, \\
\beta_i & \geq & 0, \forall i \in \{2, 3, \ldots, n-1\}, \\
0 & \geq & \beta_n, \\
0 & \geq & \gamma_1, \\
\gamma_i & > & 0, \forall i \in \{2, 3, \ldots, n-1\}.
\end{array} \tag{2.2}
$$

For simplicity, we set (w.l.o.g)

$$\begin{aligned} R_{13} &:= R_{n2} := R_{n3} = 0, \\ \gamma_2 &\leq \gamma_3 \leq \cdots \leq \gamma_{n-1}, \\ \alpha_\ell &\leq \alpha_j, \forall \ell \in \{1, 2, \ldots, \lfloor (n-4)/2 \rfloor + 1\}, \forall j \in \{\ell, \ell+1, \ldots, n\}. \end{aligned} \tag{2.3}$$

Indexing γ_j's is clearly w.l.o.g. On the other hand, the ordering on α_j's are w.l.o.g only for given j's and l's since later we order part of β_j's to enforce some shortest paths.

To achieve the desired Hamming distances, the initial shortest path tree should look like the following:

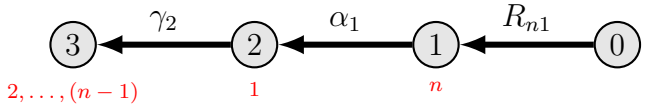


Figure 2.1: Desired worst-case shortest path tree at the start of the algorithm for $m = 3$.

In this figure, the numbers under the nodes denote the customer segments that are assigned to respective products. Conditions for the above shortest path tree to be realized are:

$$\begin{aligned} R_{j2} &\geq \alpha_1 + R_{n1}, \forall j \in \{1, 2, \ldots, n-2\}, \\ R_{k3} &\geq \gamma_j + \alpha_1 + R_{n1}, \forall j \in \{2, 3, \ldots, n-1\}, \forall k \in \{j, j+1, \ldots, n-1\}, \\ \beta_k &\geq \alpha_1 + \gamma_j, \forall j \in \{2, 3, \ldots, n-1\}, \forall k \in \{j, j+1, \ldots, n-1\}, \end{aligned} \tag{2.4}$$

where the first inequality guarantees the path $0-1-2$ is shorter than the arc $0-2$, the second inequality guarantees the path $0-1-2-3$ is shorter than the arc $0-3$ and the last one guarantees the path $1-2-3$ is shorter than the arc $1-3$.

First, all the segments at product 3 but one are moved to product 2 to decrease their Hamming distance. Note that, segment 2 is a critical segment in this shortest path tree so it is the first one to be reassigned. After moving segment 2, segment 3 becomes critical since γs are in increasing order. Eventually, all the segments 2,3,...,n-2 are moved to product 2 in order. For these movements to be realized, each movement should strictly improve the objective function. In other words, the following *moving constraint* must hold:

$$-\gamma_j + (n-j-1)(\gamma_{j+1} - \gamma_j) > 0, \forall j \in \{2, 3, \ldots, n-2\}. \tag{2.5}$$

The corresponding shortest path tree is as in Figure 2.2. Then, we force segment 1 to move from product 2 to product 1 and the shortest path tree structure to change with this

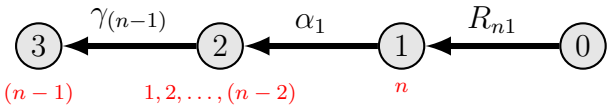


Figure 2.2: Worst-case shortest path tree after moving segments 2,3,. . . ,n-2 from product 3 to product 2.

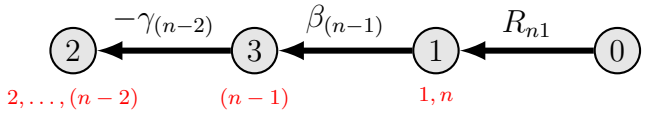


Figure 2.3: Worst-case shortest path tree after moving segment 1 from product 2 to product 1.

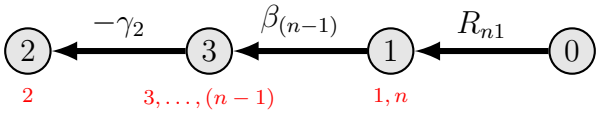


Figure 2.4: Worst-case shortest path tree after moving segments 3,. . . ,n-2 from product 2 to product 3.

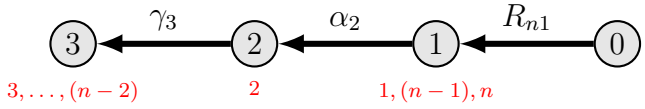


Figure 2.5: Worst-case shortest path tree after moving segment n-1 from product 3 to product 1.

assignment. The new shortest path tree is as in Figure 2.3. Note that $n-3$ products are in a Hamming distance of 3 again. The required moving constraint that implies the new assignment is better than the previous one is:

$$-\alpha_1 + (n-2)(\beta_{n-1} - \gamma_{n-2} - \alpha_1) + \gamma_{n-2} - \gamma_{n-1} \;>\; 0, \tag{2.6}$$

and the necessary inequalities to enforce the corresponding shortest path tree structure are:

$$\begin{aligned}
\beta_i &\geq \beta_{n-1}, \forall i \in \{3,4,\ldots,n-2\}, \\
R_{k3} &\geq \beta_{n-1} + R_{n1}, \forall k \in \{3,4,\ldots,n-1\}, \\
\alpha_2 &\geq \beta_{n-1} - \gamma_j, \forall j \in \{2,3,\ldots,n-2\}, \\
R_{k2} &\geq R_{n1} + \beta_{n-1} - \gamma_j, \forall j \in \{2,3,\ldots,n-2\}, \forall k \in \{2,3,\ldots,j\}.
\end{aligned} \tag{2.7}$$

Since $\gamma_j > 0, \forall j \in \{2,3,\ldots,n-2\}$ and they are in ascending order, segments $3,\ldots,n-2$ move from product 2 to product 3 in reverse order. Then the tree looks like the one given in Figure 2.4.

Afterward we move segment $n-1$ from product 3 to product 1 and the shortest path tree structure changes as in Figure 2.5. The necessary condition for this movement is:

$$-\beta_{n-1} + (\alpha_2 - \beta_{n-1} + \gamma_2) + (n-4)(\alpha_2 + \gamma_3 - \beta_{n-1}) \;>\; 0. \tag{2.8}$$

Necessary inequalities to enforce this shortest path tree structure are similar to the system of inequalities (2.4). In fact, if we define the set of system of inequalities (2.2),(2.3), (2.4), (2.5), (2.6),(2.7) and (2.8) as "one block of iterations," we can generalize this to all blocks of equations indexed by $\ell \in \{1,2,\ldots,L\}$ where $L = \lfloor (n-4)/2 \rfloor + 1$. Then the following linear inequalities should hold for every ℓ to impose the desired worst-case behavior of the instance:

$$\begin{aligned}
\alpha_1 &\geq 0, \\
0 &\geq \alpha_n, \\
\beta_i &\geq 0, \forall i \in \{2,3,\ldots,n-1\}, \\
0 &\geq \beta_n, \\
0 &\geq \gamma_1, \\
\gamma_i &> 0, \forall i \in \{2,3,\ldots,n-1\}, \\
R_{13} &:= R_{n2} := R_{n3} = 0, \\
\gamma_{n-1} &\geq \gamma_{n-2} \geq \cdots \geq \gamma_2, \\
\alpha_j &\geq \alpha_\ell, \forall j \in \{\ell, \ell+1, \ldots, n\},
\end{aligned} \tag{2.9}$$

$$
\begin{aligned}
R_{j2} &\geq \alpha_\ell + R_{n1}, \forall j \in \{\ell, (\ell+1), \ldots, (n-\ell-1)\}, \\
R_{k3} &\geq \gamma_j + \alpha_\ell + R_{n1}, \forall j \in \{(\ell+1), (\ell+2), \ldots, (n-\ell)\}, \forall k \in \{j, (j+1), \ldots, (n-\ell)\}, \\
\beta_k &\geq \alpha_\ell + \gamma_j, \forall j \in \{(\ell+1), (\ell+2), \ldots, (n-\ell)\}, \forall k \in \{j, (j+1), \ldots, (n-\ell)\},
\end{aligned}
$$

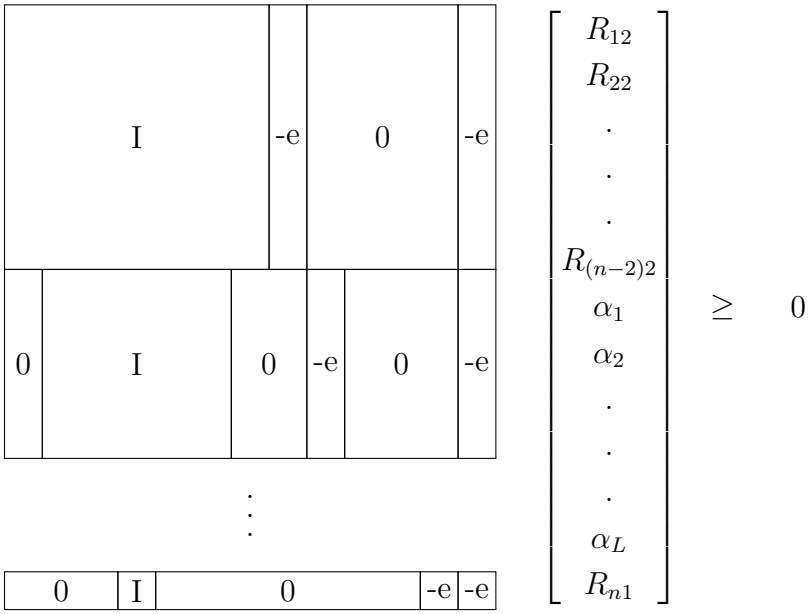
$$
-\gamma_j + (n-j-\ell)(\gamma_{j+1} - \gamma_j) > 0, \forall j \in \{(\ell+1), (\ell+2), \ldots, (n-\ell-1)\},
$$

$$
-\alpha_\ell + (n-2\ell)(\beta_{n-\ell} - \gamma_{n-\ell-1} - \alpha_\ell) + \gamma_{n-\ell-1} - \gamma_{n-\ell} > 0,
$$

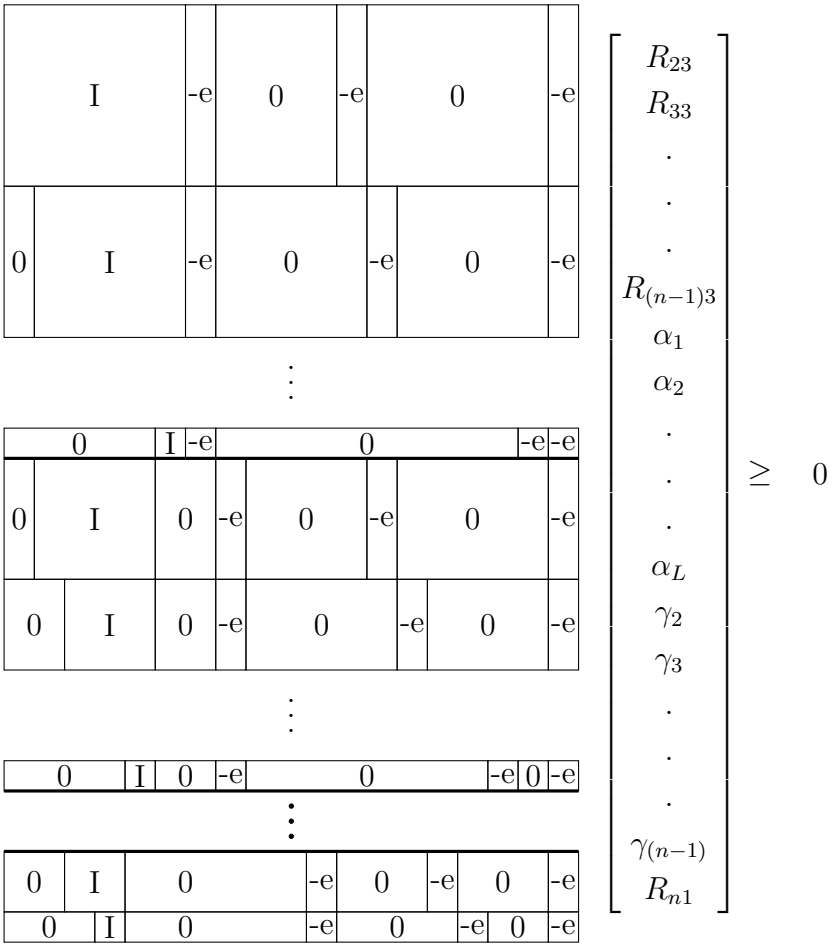
$$
\begin{aligned}
\beta_i &\geq \beta_{n-\ell}, \forall i \in \{(\ell+2), (\ell+3), \ldots, (n-\ell-1)\}, \\
R_{k3} &\geq \beta_{n-\ell} + R_{n1}, \forall k \in \{(\ell+2), (\ell+3), \ldots, (n-\ell)\}, \\
\alpha_{\ell+1} &\geq \beta_{n-\ell} - \gamma_j, \forall j \in \{(\ell+1), (\ell+2), \ldots, (n-\ell-1)\}, \\
R_{k2} &\geq R_{n1} + \beta_{n-\ell} - \gamma_j, \forall j \in \{(\ell+1), (\ell+2), \ldots, (n-\ell-1)\}, \forall k \in \{(\ell+1), (\ell+2), \ldots, j\},
\end{aligned}
$$

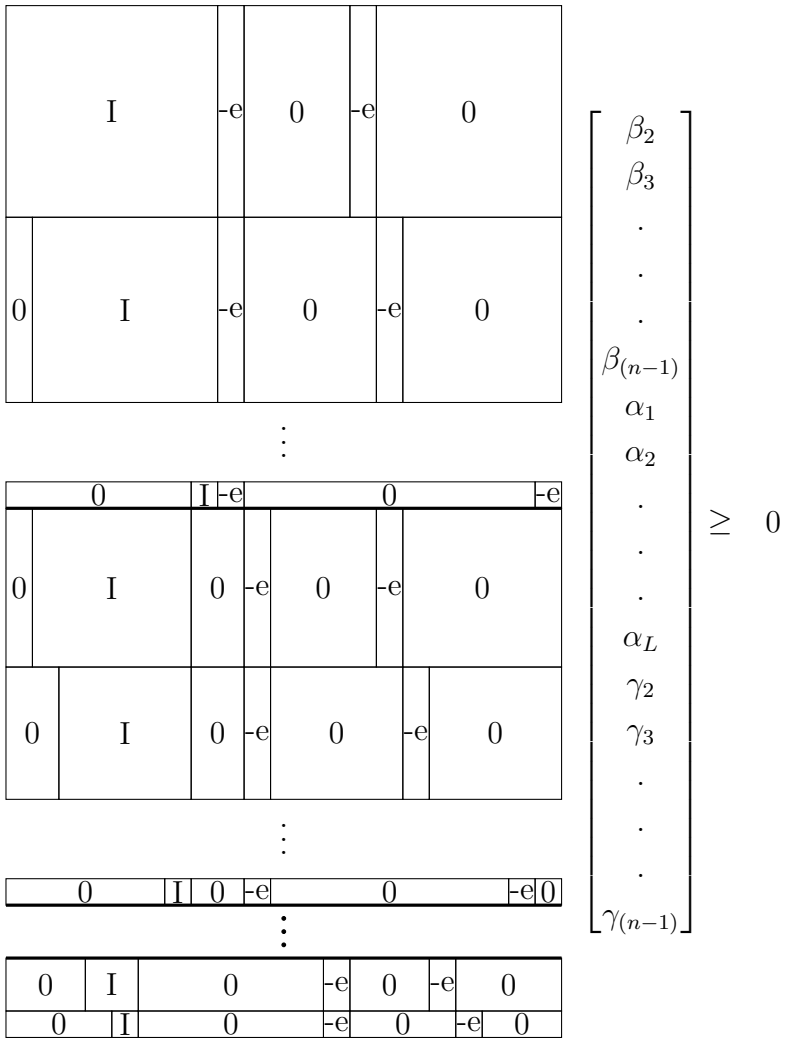
$$
-\beta_{n-\ell} + (\alpha_{\ell+1} - \beta_{n-\ell} + \gamma_{\ell+1}) + (n - 2(\ell+1))(\alpha_{\ell+1} + \gamma_{\ell+2} - \beta_{n-\ell}) > 0.
$$

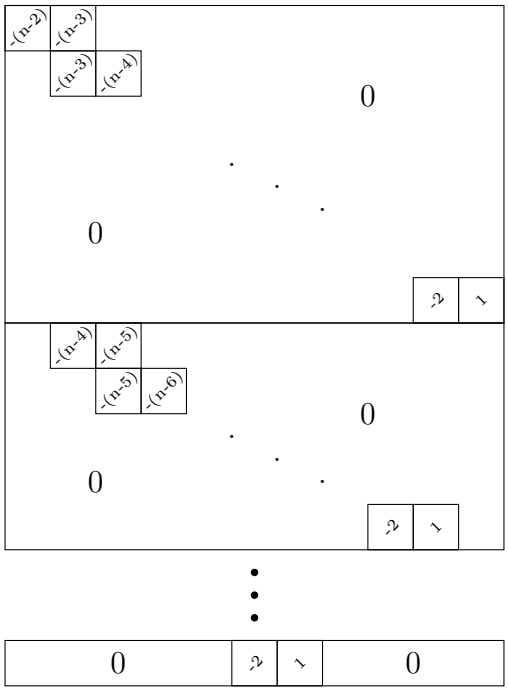
Matrix representation of system (2.9) is the following:



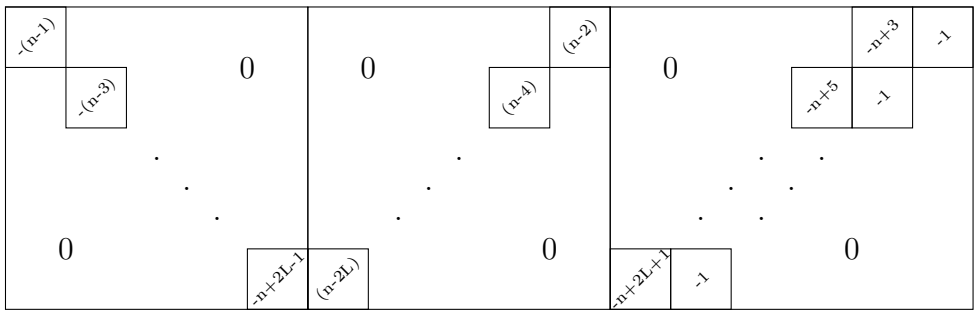
$$
\begin{bmatrix}
R_{12} \\ R_{22} \\ \cdot \\ \cdot \\ \cdot \\ R_{(n-2)2} \\ \alpha_1 \\ \alpha_2 \\ \cdot \\ \cdot \\ \cdot \\ \alpha_L \\ R_{n1}
\end{bmatrix}
\geq 0
$$


I -e 0 -e 0 -e
0 I -e 0 -e 0 -e
0 I -e 0 -e 0 -e
0 I 0 -e 0 -e 0 -e
0 I 0 -e 0 -e 0 -e
0 I 0 -e 0 -e 0 -e
0 I 0 -e 0 -e 0 -e
0 I 0 -e 0 -e 0 -e
0 I 0 -e 0 -e 0 -e
R_{23}
R_{33}
$R_{(n-1)3}$
α_1
α_2
α_L
γ_2
γ_3
$\gamma_{(n-1)}$
R_{n1}
$\geq$ 0

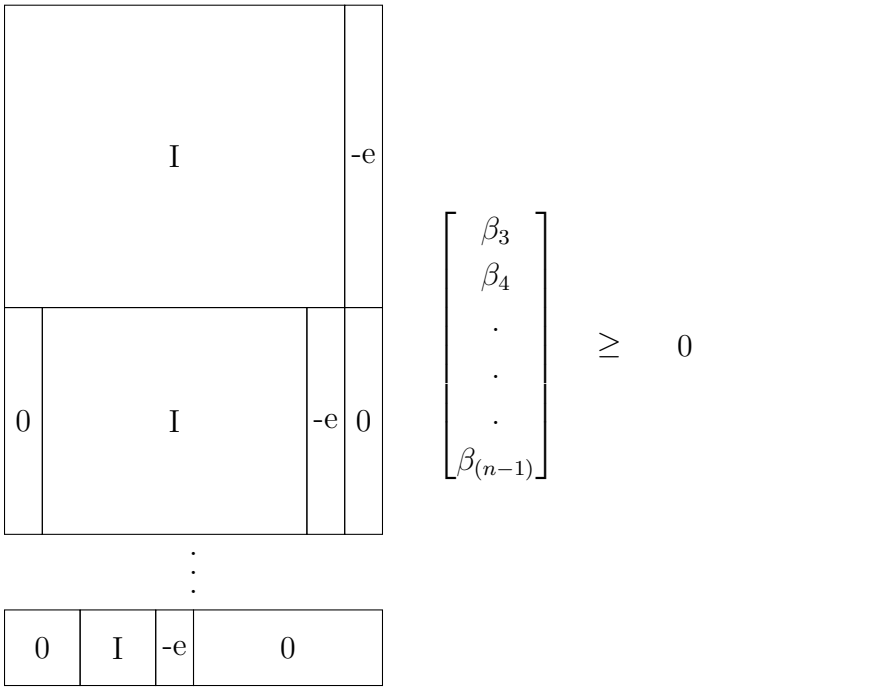

I -e 0 -e 0
0 I -e 0 -e 0
0 I -e 0 -e 0
0 I 0 -e 0 -e 0
0 I 0 -e 0 -e 0
0 I 0 -e 0 -e 0
0 I 0 -e 0 -e 0
0 I 0 -e 0 -e 0
β_2
β_3
$\beta_{(n-1)}$
α_1
α_2
α_L
γ_2
γ_3
$\gamma_{(n-1)}$
$\geq$ 0

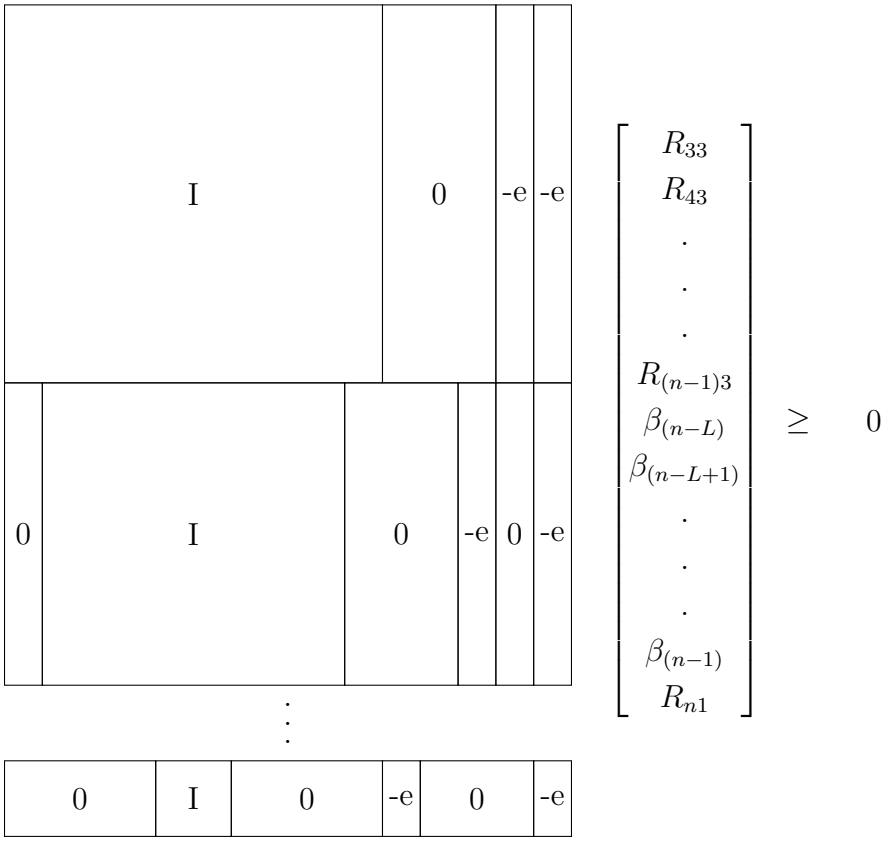


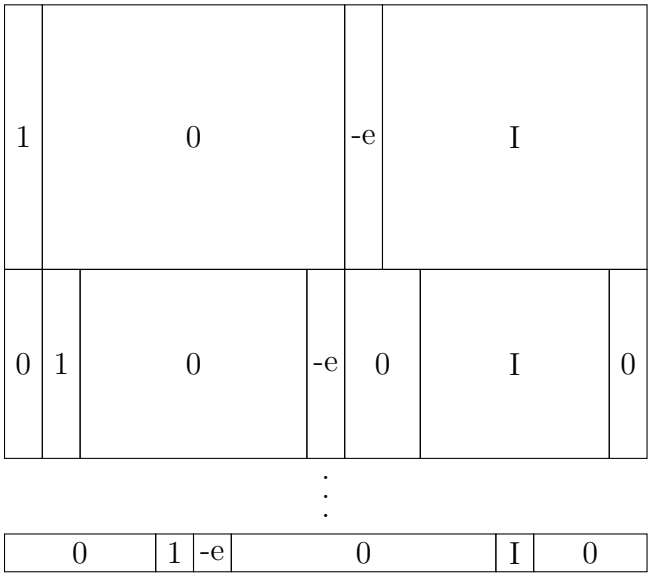
$$\begin{bmatrix} \gamma_2 \\ \gamma_3 \\ . \\ . \\ . \\ \gamma_{(n-1)} \end{bmatrix} > 0$$



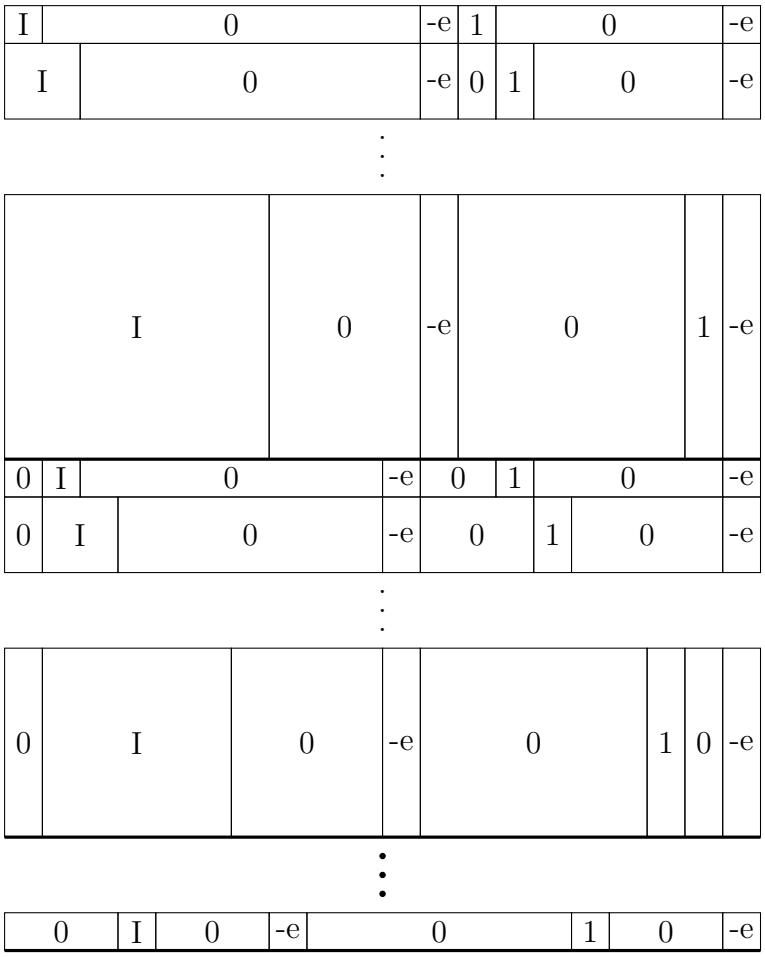
$$\begin{bmatrix} \alpha_1 \\ \alpha_2 \\ . \\ . \\ . \\ \alpha_L \\ \beta_{(n-L)} \\ \beta_{(n-L-1)} \\ . \\ . \\ . \\ \beta_{(n-1)} \\ \gamma_{(n-L-1)} \\ \gamma_{(n-L-2)} \\ . \\ . \\ . \\ \gamma_{(n-1)} \end{bmatrix} > 0$$


I
-e
0
I
-e
0
0
I
-e
0
$\begin{bmatrix} \beta_3 \\ \beta_4 \\ . \\ . \\ . \\ \beta_{(n-1)} \end{bmatrix} \geq 0$

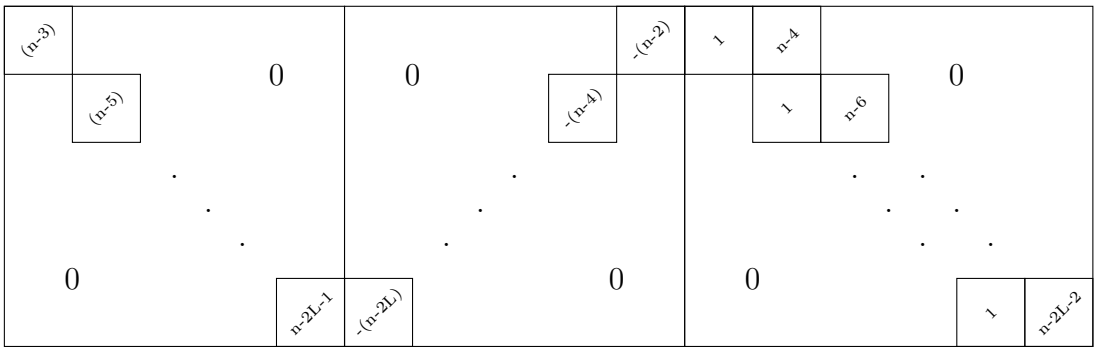

I
0
-e
-e
0
I
0
-e
0
-e
0
I
0
-e
0
-e
$\begin{bmatrix} R_{33} \\ R_{43} \\ . \\ . \\ . \\ R_{(n-1)3} \\ \beta_{(n-L)} \\ \beta_{(n-L+1)} \\ . \\ . \\ . \\ \beta_{(n-1)} \\ R_{n1} \end{bmatrix} \geq 0$



$$\begin{bmatrix} \alpha_2 \\ \alpha_3 \\ \cdot \\ \cdot \\ \cdot \\ \alpha_{(L+1)} \\ \beta_{(n-L)} \\ \beta_{(n-L+1)} \\ \cdot \\ \cdot \\ \cdot \\ \beta_{(n-1)} \\ \gamma_2 \\ \gamma_3 \\ \cdot \\ \cdot \\ \cdot \\ \gamma_{(n-2)} \end{bmatrix} \geq \quad 0$$



$$\begin{bmatrix} R_{22} \\ R_{32} \\ . \\ . \\ . \\ R_{(n-2)2} \\ \beta_{(n-L)} \\ \beta_{(n-L+1)} \\ . \\ . \\ . \\ \beta_{(n-1)} \\ \gamma_2 \\ \gamma_3 \\ . \\ . \\ . \\ \gamma_{(n-2)} \\ R_{n1} \end{bmatrix} \geq 0$$



$$\begin{bmatrix} \alpha_2 \\ \alpha_3 \\ . \\ . \\ . \\ \alpha_{(L+1)} \\ \beta_{(n-L)} \\ \beta_{(n-L-1)} \\ . \\ . \\ . \\ \beta_{(n-1)} \\ \gamma_2 \\ \gamma_3 \\ . \\ . \\ . \\ \gamma_{(L+2)} \end{bmatrix} > 0$$

We showed the following fact.

Proposition 2.1. *Let n and L be a pair of given positive integers such that $L \leq \left\lfloor \frac{(n-4)}{2} \right\rfloor + 1$. Then, the system of inequalities (2.9) is feasible iff there exists an instance of the Optimal Pricing Problem* (1.8) *with number of customer segments n, number of products $m = 3$ and $N_i = 1$, $\forall i \in \{1,2, \ldots, n\}$ with a QGR pattern.*

Next we prove that the underlying family of system of linear inequalities have solutions for every n large enough.

Theorem 2.2. *For every $n \geq 8$ and $L = \left\lfloor \frac{(n-4)}{2} \right\rfloor + 1$, the system of inequalities (2.9) is feasible.*

Proof. Let $n \geq 8$, and set $L := \left\lfloor \frac{(n-4)}{2} \right\rfloor + 1$. We further define

$$
\begin{aligned}
R_{i1} &= R_{12} = R_{13} = R_{(n-1)2} = R_{n2} = R_{n3} = 0, \forall i \in \{1, \ldots, L\} \cup \{n-L, \ldots, n\}, \\
\gamma_2 &= \epsilon, \\
R_{(n-2)2} &= R_{22}, \\
\gamma_3 &= \frac{(n-1)}{(n-3)}\epsilon, \\
\gamma_i &= \left[\frac{(n-1)(n-4)(n-6)\ldots(n-2i+4)}{(n-3)(n-5)\ldots(n-2i+3)} + \frac{(n-6)\ldots(n-2i+4)}{(n-5)\ldots(n-2i+3)} \right. \\
&+ \left. \cdots + \frac{1}{(n-2i+3)} \right] \epsilon, \forall i \in \left\{ 4, \ldots, \left\lceil \frac{n}{2} \right\rceil \right\}, \\
\gamma_i &= 2\gamma_{i-1} + \epsilon, \forall i \in \left\{ \left\lceil \frac{n}{2} \right\rceil + 1, \ldots, n-1 \right\}, \\
R_{22} &= \left[(n-2)\gamma_{n-1} - \frac{(n-1)(n-4)}{(n-3)}\epsilon \right] / (n-3), \\
\beta_{n-2} &= R_{22} + \gamma_{n-2}, \\
\alpha_i &= \left[(n-2i+2)\beta_{n-i+1} - \gamma_i - (n-2i)\gamma_{i+1} + \epsilon \right] / (n-2i+1), \forall i \in \left\{ 3, \ldots, \left\lfloor \frac{n}{2} \right\rfloor \right\}, \\
\beta_{n-i} &= \left[(n-2i+1)\alpha_i + (n-2i-1)\gamma_{n-i-1} + \gamma_{n-i} + \epsilon \right] / (n-2i), \forall i \in \{3, \ldots, L\},
\end{aligned}
$$

where

$$\epsilon > 0.$$

Also,

$$\begin{aligned} R_{(n-L-1),1} &= 0, \\ \alpha_{\lfloor \frac{n}{2} \rfloor} &= \alpha_{\lfloor \frac{n}{2} \rfloor + 1} \end{aligned}$$

if n is odd.

Even though tedious, one can check that there exists $\epsilon > 0$ such that the above described values satisfy every constraint in (2.9). □

Corollary 2.3. *There exist instances of the Optimal Pricing Problem* (1.8) *for which the modified Dobson-Kalish algorithm makes* $\Omega(n^2)$ *reassignments of some customer segments.*

Chapter 3

Dobson-Kalish Analysis (m=3)

In this chapter, we study the original Dobson-Kalish algorithm. As opposed to the modified version, the original algorithm chooses to reassign a customer segment to its parent node only if it leads to the best improvement among all possible reassignments. The algorithm stops when there is no further improvement is possible. Similar to the previous chapter, we focus on the special case of three products. Moreover, we follow the same shortest path tree structure change to investigate the worst-case behavior. However, in this chapter we relax the single customer assumption (that is, N_i is not necessarily equal to 1). Note that in the example of Shioda et al. [7] usage of different values for N_i was critical. Moreover, even though $N_i = 1$, $\forall i$ case admits an LP formulation approach as in Chapter 2, the analysis and LPs get much more complicated due to the formulation of "best improving reassignment." Therefore, in this chapter, we immediately focus on the more general case.

We want the definitions, simplifications and initial assignments in the previous chapter also to hold for this chapter. Then, the required set of inequalities is simply the combination of (2.1),(2.2), and (2.3):

$$\begin{aligned}
\alpha_i &:= R_{i2} - R_{i1}, \forall i \in \{1, 2, \ldots, n\}, \\
\beta_i &:= R_{i3} - R_{i1}, \forall i \in \{1, 2, \ldots, n\}, \\
\gamma_i &:= R_{i3} - R_{i2}, \forall i \in \{1, 2, \ldots, n\}, \\
\alpha_1 &\geq 0, \\
0 &\geq \alpha_n, \\
\beta_i &\geq 0, \forall i \in \{2, 3, \ldots, n-1\}, \\
0 &\geq \beta_n, \\
0 &\geq \gamma_1, \\
\gamma_i &> 0, \forall i \in \{2, 3, \ldots, n-1\},
\end{aligned} \tag{3.1}$$

$$\begin{aligned} R_{13} &:= R_{n2} := R_{n3} = 0, \\ \gamma_2 &\leq \gamma_3 \leq \cdots \leq \gamma_{n-1}, \\ \alpha_\ell &\leq \alpha_j, \forall \ell \in \{1, 2, \ldots, \lfloor (n-4)/2 \rfloor + 1\}, \forall j \in \{\ell, \ell+1, \ldots, n\}. \end{aligned}$$

In addition, we need the following shortest path conditions for initial tree to be as in Figure 2.1:

$$\begin{aligned} R_{j2} &\geq \alpha_1 + R_{n1}, \forall j \in \{1, 2, \ldots, n-2\}, \\ R_{k3} &\geq \gamma_j + \alpha_1 + R_{n1}, \forall j \in \{2, 3, \ldots, n-1\}, \forall k \in \{j, j+1, \ldots, n-1\}, \\ \beta_k &\geq \alpha_1 + \gamma_j, \forall j \in \{2, 3, \ldots, n-1\}, \forall k \in \{j, j+1, \ldots, n-1\}. \end{aligned} \tag{3.2}$$

For the segments 2,3,...,n-2 to move to product 2, the following conditions must hold:

$$\begin{aligned} \min_{i=2,\ldots,(n-1)} \{R_{i3}\} &\geq \min_{i=2,\ldots,(n-1)} \{\beta_i\} + R_{n1}, \\ \min_{i=k,\ldots,(n-1)} \{R_{i3}\} &\geq \min_{j=1,\ldots,(k-1)} \{R_{j2}\} + \gamma_k, \forall k \in \{2, 3, \ldots, n-1\}, \\ R_{j2} &\geq \alpha_2 + R_{n1}, \forall j \in \{2, 3, \ldots, (n-3)\}, \\ R_{k3} &\geq \gamma_j + \alpha_2 + R_{n1}, \forall j \in \{3, 4, \ldots, (n-2)\}, \forall k \in \{j, (j+1), \ldots, (n-1)\}, \\ \beta_k &\geq \alpha_2 + \gamma_j, \forall j \in \{3, 4, \ldots, (n-2)\}, \forall k \in \{j, (j+1), \ldots, (n-1)\}, \end{aligned} \tag{3.3}$$

$$\sum_{i=3}^{n-1} N_i \gamma_3 + \sum_{i=1}^{n-1} N_i \alpha_1 - \sum_{i=2}^{n-1} N_i \min_{k=2,\ldots,(n-1)} \{\beta_k\} > 0, \tag{3.4}$$

for every $k \in \{2, 3, \ldots, n-2\}$,

$$\begin{aligned} \sum_{i=k+1}^{n-1} N_i \gamma_{k+1} - \sum_{i=k}^{n-1} N_i \gamma_k + \sum_{i=1}^{n-1} N_i \alpha_1 + \sum_{i=1}^{n} N_i R_{n1} - \sum_{i=1}^{n-1} N_i \min_{j=1,\ldots,(k-1)} \{R_{j2}\} &> 0, \\ \sum_{i=k+1}^{n-1} N_i \gamma_{k+1} - \sum_{i=k}^{n-1} N_i \gamma_k &> 0, \end{aligned}$$

and for every $k \in \{3, 4, \ldots, n-2\}$,

$$\sum_{i=k+1}^{n-1} N_i \gamma_{k+1} - \sum_{i=k}^{n-1} N_i \gamma_k + \sum_{i=1}^{n-1} N_i \alpha_1 - \sum_{i=2}^{n-1} N_i \alpha_2 > 0.$$

Conditions (3.3) enforce the shortest path tree structure for each possible reassignment while conditions (3.4) guarantee that moving segments 2,3,...,n-2 to product 2 are the most profitable reassignments among all possible. The corresponding shortest path tree is

as in Figure 2.2. Then, we want segment 1 to move from product 2 to product 1 and the new shortest path tree to be as in Figure 2.3. Note that the tree structure changes with this reassignment.

$$
\begin{aligned}
R_{j2} &\geq \alpha_1 + R_{n1}, \forall j \in \{1, 2, \ldots, (n-1)\}, && (3.5)\\
-\sum_{i=2}^{n-2} N_i\gamma_{n-2} + \sum_{i=2}^{n-1} N_i\beta_{n-1} - N_{n-1}\gamma_{n-1} - \sum_{i=1}^{n-1} N_i\alpha_1 &> 0,\\
-\sum_{i=2}^{n-2} N_i\gamma_{n-2} + \sum_{i=2}^{n-1} N_i\beta_{n-1} - \sum_{i=1}^{n-1} N_i\alpha_i &> 0,\\
-\sum_{i=2}^{n-2} N_i\gamma_{n-2} + \sum_{i=2}^{n-1} N_i\beta_{n-1} + \sum_{i=1}^{n} N_iR_{n1} - N_{n-1}\gamma_{n-1} - \sum_{i=1}^{n-1} N_i \min_{j=1,\ldots,(n-2)}\{R_{j2}\} &> 0.
\end{aligned}
$$

Necessary inequalities to enforce the corresponding shortest path tree structure are:

$$
\begin{aligned}
\beta_i &\geq \beta_{n-1}, \forall i \in \{3, 4, \ldots, n-2\}, && (3.6)\\
R_{k3} &\geq \beta_{n-1} + R_{n1}, \forall k \in \{3, 4, \ldots, n-1\},\\
\alpha_2 &\geq \beta_{n-1} - \gamma_j, \forall j \in \{2, 3, \ldots, n-2\},\\
R_{k2} &\geq R_{n1} + \beta_{n-1} - \gamma_j, \forall j \in \{2, 3, \ldots, n-2\}, \forall k \in \{2, 3, \ldots, j\}.
\end{aligned}
$$

Then we want segments $3, \ldots, n-2$ move from product 2 to product 3 in reverse order. Additional inequalities needed are:

$$
\begin{aligned}
\min_{i=2,\ldots,(n-2)}\{R_{i2}\} &\geq \alpha_2 + R_{n1}, && (3.7)\\
-\sum_{i=2}^{n-3} N_i\gamma_{n-3} + \sum_{i=2}^{n-1} N_i\beta_{n-1} - \sum_{i=2}^{n-2} N_i\alpha_2 &> 0,\\
\sum_{i=2}^{k+1} N_i\gamma_{k+1} - \sum_{i=2}^{k} N_i\gamma_k + N_nR_{n1} + \sum_{i=1}^{n-1}(R_{n1} - R_{11}) &> 0, \forall k \in \{2, 3, \ldots, n-3\},\\
-\sum_{i=2}^{k} N_i\gamma_k + \sum_{i=2}^{k+1} N_i\gamma_{k+1} + \sum_{i=2}^{n-1} N_i\beta_{n-1} - \sum_{i=2}^{n-1} N_i\beta_{n-2} &> 0, \forall k \in \{2, 3, \ldots, n-4\},
\end{aligned}
$$

$$
\begin{aligned}
R_{k3} &\geq \beta_{n-2} + R_{n1}, \forall k \in \{4, 5, \ldots, n-2\},\\
\alpha_2 &\geq \beta_{n-2} - \gamma_j, \forall j \in \{3, 4, \ldots, n-3\},\\
R_{k2} &\geq R_{n1} + \beta_{n-2} - \gamma_j, \forall j \in \{3, 4, \ldots, n-3\}, \forall k \in \{2, 3, \ldots, j\},\\
R_{k3} &\geq \beta_{n-1} + R_{11}, \forall k \in \{3, 4, \ldots, n-1\},\\
R_{k2} &\geq R_{11} + \beta_{n-1} - \gamma_j, \forall j \in \{2, 3, \ldots, n-2\}, \forall k \in \{2, 3, \ldots, j\}.
\end{aligned}
$$

Resulting tree looks like the one given in Figure 2.4. Afterward we move segment $n-1$ from product 3 to product 1 and the shortest path tree structure changes as in Figure 2.5. The necessary conditions for this movement are:

$$\sum_{i=3}^{n-2} N_i\gamma_3 + \sum_{i=2}^{n-2} N_i\alpha_2 + N_2\gamma_2 - \sum_{i=2}^{n-1} N_i\beta_{n-1} > 0, \quad (3.8)$$

$$\sum_{i=3}^{n-2} N_i\gamma_3 + \sum_{i=2}^{n-2} N_i\alpha_2 - \sum_{i=2}^{n-1} N_i \min_{k=2,\ldots,(n-1)}\{\beta_k\} > 0,$$

$$\sum_{i=3}^{n-2} N_i\gamma_3 + \sum_{i=2}^{n-2} N_i\alpha_2 + N_2\gamma_2 - \sum_{i=2}^{n-1} N_i\beta_{n-1} + \sum_{i=1}^{n} N_iR_{n1} - \sum_{i=1}^{n-1} N_iR_{11} > 0.$$

The set of system of inequalities (3.1),(3.2), (3.3), (3.4), (3.5),(3.6), (3.7) and (3.8) defines "the first block of iterations." However, the other blocks of iterations are not in a similar structure. Moreover, many of the inequalities in the above formulation is no longer linear or even convex (due to the terms like $N_i\alpha_j$, $N_i\beta_j$, etc.). Hence, a straightforward generalization does not seem possible. Therefore, the analysis of the worst-case complexity of the Dobson-Kalish algorithm only seems manageable through the modified version.

Chapter 4

Minimally Infeasible Systems and Upper Bounds on the Number of Reassignments

In Chapter 2, we show that there exist instances of the optimal pricing problem (1.8) for which the modified Dobson-Kalish algorithm makes $\Omega(n^2)$ reassignments of some customer segments. In this chapter, we work on the upper bound for the worst-case complexity of the algorithm.

After L iterations, i.e after QGR is realized, the Hamming distances of the segments from node 0 are as follows:

Hamming distance from node 0	:	1 2 3
Number of segments at each distance	:	[n-2, 1, 1]

assuming $m = 3$ and $N_i = 1$, $\forall i$. Now, we further force the pattern of the number of customer segments to change as follows:

$$
\begin{array}{l}
[\ 1, \quad 1, \quad n\text{-}3\ ] \\
[\ 1, \quad 2, \quad n\text{-}4\ ] \\
\ldots \\
[1, \quad n\text{-}3, \quad 1] \\
[2, \quad 1, \quad n\text{-}4] \\
[2, \quad 2, \quad n\text{-}5] \\
\ldots \\
[2, \quad n\text{-}4, \quad 1] \\
[3, \quad 1, \quad n\text{-}5]
\end{array}
$$

$$
\begin{array}{lll}
[3, & 2, & n\text{-}6] \\
\ldots & & \\
\ldots & & \\
[n\text{-}3, & 1, & 1] \\
[\ 1, & 1, & n\text{-}4\ ] \\
\ldots & & \\
\ldots & & \\
\ldots & & \\
[1, & 1, & 1]
\end{array}
$$

If this pattern change is possible, then it provides a new lower bound, n^3, on the number of reassignments. If not, we can use the minimally infeasible system to get an upper bound. Let's focus on the special case of $n = 14$ products. The shortest path tree after L iterations looks like the following:

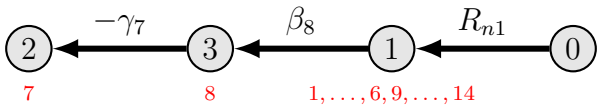


Figure 4.1: Worst-case shortest path tree after QGR is realized.

We kill segment 14, that is delete it completely so that the shortest path tree changes and $n-3$ segments become furthest from node 0. The new shortest path tree is as follows:

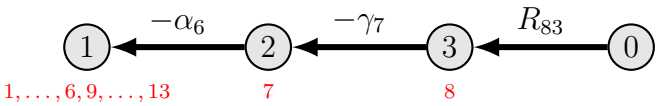


Figure 4.2: Worst-case shortest path tree after killing segment 14.

Conditions for the above shortest path tree to be realized are:

$$
\begin{array}{rcl}
R_{72} & \geq & -\gamma_7 + R_{83}, \\
R_{i1} & \geq & -\alpha_6 - \gamma_7 + R_{83}, \forall i \in \{1, \ldots, 6, 9, \ldots, 13\}, \\
-\beta_i & \geq & -\alpha_6 - \gamma_7, \forall i \in \{1, \ldots, 6, 9, \ldots, 13\},
\end{array}
\tag{4.1}
$$

and the moving constraint implying an increase in the objective function is:

$$-(n-3)\alpha_6 - (n-3)\gamma_7 + (n-1)R_{83} > 2\beta_8 + nR_{n1}. \tag{4.2}$$

Then, we force segment 6 to move from product 1 to product 2 and the tree looks like the following:

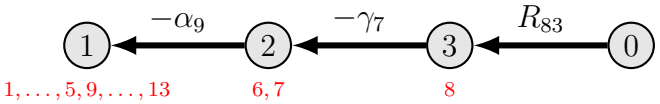


Figure 4.3: Worst-case shortest path tree after moving segment 6 from product 1 to product 2.

The necessary inequalities to enforce this shortest path tree structure are:

$$\begin{aligned} R_{i2} &\geq -\gamma_7 + R_{83}, \forall i \in \{6, 7\}, \\ R_{i1} &\geq -\alpha_9 - \gamma_7 + R_{83}, \forall i \in \{1, \ldots, 5, 9, \ldots, 13\}, \\ -\beta_i &\geq -\alpha_9 - \gamma_7, \forall i \in \{1, \ldots, 5, 9, \ldots, 13\}. \end{aligned} \tag{4.3}$$

Since $\alpha_6 \geq \alpha_9$, we do not need an additional moving constraint. Afterward we attempt to move segment 9 from product 1 to product 2 but the system becomes infeasible when we add the required moving constraint. Here is the minimally infeasible system:

$$\begin{aligned} \gamma_2 &> 0, \\ -\gamma_j + (14 - j - \ell)(\gamma_{j+1} - \gamma_j) &> 0, \forall \ell \in \{1, \ldots, 6\}, \forall j = \ell + 1, \\ \gamma_9 - 2\gamma_8 &> 0, \\ -\beta_9 &\geq -\alpha_9 - \gamma_7. \end{aligned} \tag{4.4}$$

The first constraint is the one that guarantees the movement of segments from product 2 to product 3. The second set of constraints and the third constraint are needed to move segments $2, 3, \ldots, 8$ from product 3 to product 2. The last constraint is needed to move segment 9 from product 1 to product 2. This minimally infeasible system implies that the number of times that segment 9 is reassigned might be bounded by a relatively slow growing function of m and n. In fact, a similar analysis with $m = 4$ results a similar minimally infeasible system:

$$\begin{aligned} \lambda_3 &> 0, \\ -\lambda_j + (14 - j - \ell)(\lambda_{j+1} - \lambda_j) &> 0, \forall \ell \in \{1, \ldots, 5\}, \forall j = \ell + 2, \\ \lambda_6 - \lambda_8 &\geq 0. \end{aligned} \tag{4.5}$$

where

$$\lambda_i = R_{i4} - R_{i3}, \forall i \in \{1, 2, \ldots, n\}.$$

The first inequality motivates the movement of segments from product 3 to product 4. The second set of inequalities ensures that segments $3, 4, \ldots, 7$ move from product 4 to product 3. The last inequality is required to move segment 9 from product 2 to product 4. Similar to the previous system, this one also implies that the number of times that segment 9 is reassigned might be limited.

In conclusion, minimally infeasible systems suggest that the number of reassignments of a fixed customer segment i from j_1 to j_2 may be bounded above by a mild function of m and n. Analyzing such a bound directly yields an upper bound on the complexity of the modified Dobson-Kalish algorithm. In particular, if we prove that for a fixed customer segment i and a fixed pair of products j_1 and j_2, the number of reassignments of customer segment i from product j_1 to product j_2 is at most $k(m, n)$, we can conclude that the modified Dobson-Kalish algorithm performs $O(nm^2k(m, n))$ reassignments. The same bound also applies to the Dobson-Kalish algorithm.

Chapter 5

Conclusion, Current and Future Research

In this study, we have presented mixed integer programming formulations of optimal pricing problem, the Dobson-Kalish Reassignment Heuristic and the modified version of the heuristic. We have created worst-case instances of optimal pricing problem by treating the data of the problem as variables and then solved these instances by using linear programming techniques. In Chapter 2, we have proved that there exist instances of optimal pricing problem for which the modified Dobson-Kalish algorithm makes $\Omega(n^2)$ reassignments of customer segments. In Chapter 4, we have derived the conditions for the existence of an instance having a higher degree pattern change than the QGR pattern and showed that the underlying system of inequalities is not feasible. The resultant minimally infeasible system can be used as a guide to prove an upper bound on the number of reassignments of a given customer segment from a given product to any other product. In the near future, we aim to use this system to find an upper bound for the worst-case complexity of the algorithm [1].

The method that we followed in this study can be applied to find the worst-case complexity of any other algorithms where the classical methods fail to do that. That is to say, for a given optimization problem and a given algorithm for its solution, we can try to express certain patterns on the behaviour of the algorithm as constraints on the data for the problem. Then such a set of constraints defines a family of optimization problems in the data space. This family can be used to generate worst-case instances as well as ideas for proving upper bounds on the complexity of the algorithm at hand.

APPENDICES

Appendix A

Proof that the Optimal Pricing Problem is NP-hard

In graph theory, a vertex cover of a graph is defined as a set of vertices such that each edge of the graph is incident to at least one vertex of the set.

Minimum vertex cover problem is the problem of finding a vertex cover having the minimum size. Decision version of the problem is the following:

Definition A.1. *Vertex Cover Problem: Given a graph G and a positive integer B, does G have a vertex cover of size at most B?*

If one can solve this problem, he can solve the optimization version as well: Let $|V|$ denote the number of vertices in G and B^* denote the size of the minimum vertex cover. B^* can be computed by applying binary search on B and solving the above given decision version of the vertex cover problem for each B. Since B is bounded below by 0 and bounded above by $|V|$, binary search algorithm finds the optimal B in at most $\log_2 |V|$ iterations. Once B^* is determined, a corresponding vertex cover is found by Algorithm A.1.

Vertex cover problem is proven to be a $\mathcal{NP}$-complete problem (see Karp [6]). Therefore, its optimization version is $\mathcal{NP}$-hard. Similarly, if we can show the decision version of the pricing problem is $\mathcal{NP}$-complete, it implies that the optimal pricing problem is $\mathcal{NP}$-hard.

Definition A.2. *Decision version of the Pricing Problem: Given an instance of pricing problem and a lower bound k, is there a feasible solution to the problem with an objective function value at least k?*

We prove $\mathcal{NP}$-completeness of this problem by reduction from the vertex cover problem.

Proposition A.1. *The optimal pricing problem* (1.8) *is $\mathcal{NP}$-hard.*

Algorithm A.1 Minimum Vertex Cover Algorithm

Require: Graph G and B^*: size of the minimum vertex cover in G.

$k := B^*$ and $C := \emptyset$.
repeat
  for all vertices i **do**
    Remove vertex i and all incident edges.
    Solve vertex cover problem with $B = k - 1$.
    if the answer of the problem is yes **then**
      Add vertex i to set C, decrement k by 1 and update network.
    else
      Restore previously updated network.
    end if
  end for
until $k = 0$.
Return C.

Proof. We give a reduction from the problem of finding the vertex cover of cardinality B or less in a graph, which is known to be $\mathcal{NP}$-hard. Let $G = (V, E)$ be the given graph of the vertex cover problem and B be the given positive integer. Construct an instance of the optimal pricing problem as follows. For every vertex in $v \in V$, make a product; for every edge $\{u, v\} \in E$, make a customer segment. Define

$$R_{ij} := \begin{cases} 1, & \text{if edge } i \text{ is incident to vertex } j \text{ in G,} \\ 0, & \text{otherwise.} \end{cases}$$

That is, the customer segment $i = \{u, v\}$ is only interested in two products, namely u and v. Let the fixed cost for each product be 1 and the variable cost be 0. Further let $N_i := 1$, $\forall i$ (i.e. there is only one customer in each segment). Finally, set $k := |E| - B$.

Let $C \subseteq V$ be a vertex cover in G such that $|C| \leq B$. Define $\bar{x}_{ij}$ and $\bar{y}_j$ as

$$\bar{y}_j := \begin{cases} 1, & \text{if } j \in C, \\ 0, & \text{otherwise,} \end{cases}$$

and

$$\bar{x}_{ij} := \begin{cases} 1, & \text{if } j \in C,\ k \notin C \text{ and } i = \{j, k\}\ , \\ 1, & \text{if } j, k \in C, \text{ and } j < k\ , \\ 0, & \text{otherwise.} \end{cases}$$

Further set $\bar{\pi}_{ij} := 1$, $\forall j \in C$. Notice that $(\bar{x}, \bar{y}, \bar{\pi})$ is feasible in (1.8) with objective function value $|E| - |C|$. Therefore, the optimal objective function value of (1.8) $\geq |E| - |C|$ $\geq |E| - B = k$.

For the converse, let (x^*, y^*, π^*) be an optimal solution of (1.8) with objective value at least $k = |E| - B$. Let $C := \left\{j \in V : y_j^* = 1\right\}$. We claim that C is a vertex cover in G. By the second group of constraints in (1.8) and that x_{ij} must be 0 or 1, we have that for every i, there exists $j(i)$ such that $x_{ij(i)}^* = 1$. By the third group of constraints in (1.8) and that y_j is 0 or 1, we deduce $y_{j(i)}^* = 1$, $\forall i$. Therefore, C is a vertex cover in G. By the first group of constraints, $\pi_j^* = 1$, $\forall j \in C$. Therefore, the objective value of (x^*, y^*, π^*) is

$$\sum_i \pi_{j(i)}^* x_{ij(i)}^* - |C| = |E| - |C| \geq k.$$

Thus, we have $|E| - |C| \geq |E| - B \Rightarrow |C| \leq B$. That is G has a vertex cover of size at most B. $\square$

Bibliography

[1] D. Demirtaş, R. Shioda, and L. Tunçel. Worst-Case complexity analyses for the Dobson-Kalish optimal pricing algorithm and its relatives. In preparation. 38

[2] G. Dobson and S. Kalish. Positioning and pricing a product line. *Marketing Science*, 7(2):107–125, 1988. 5, 8, 11, 13

[3] G. Dobson and S. Kalish. Heuristics for pricing and positioning a product-line using conjoint and cost data. *Management Science*, 39(2):160–175, 1993. 7

[4] V. Guruswami, J.D. Hartline, A.R. Karlin, D. Kempe, C. Kenyon, and F. McSherry. On profit-maximizing envy-free pricing. In *Proceedings of the sixteenth annual ACM-SIAM symposium on Discrete algorithms*, page 1173. Society for Industrial and Applied Mathematics, 2005. 7

[5] W. Hanson and R.K. Martin. Optimal bundle pricing. *Management Science*, 36(2):155–174, 1990. 7

[6] Richard M. Karp. Reducibility among combinatorial problems. In *Complexity of computer computations (Proc. Sympos., IBM Thomas J. Watson Res. Center, Yorktown Heights, N.Y., 1972)*, pages 85–103. Plenum, New York, 1972. 40

[7] R. Shioda, L. Tunçel, and T.G.J. Myklebust. Maximum utility product pricing models and algorithms based on reservation prices. Technical Report CORR 2007-08, Combinatorics and Optimization, University of Waterloo, AUGUST 2007. 4, 7, 12, 13, 14, 30

[8] B.C. Smith, J.F. Leimkuhler, and R.M. Darrow. Yield management at American airlines. *Interfaces*, 22:8–31, 1992. 2

[9] K.T. Talluri and G.J. Van Ryzin. *The theory and practice of revenue management*. International Series in Operations Research & Management Science, 68. Kluwer Academic Publishers, Boston, MA, 2004. 2, 14

[10] L. Tunçel. Optimization based approaches to product pricing. In *Proceedings of fourth International Conference on Business, Management and Economics, ICBME2008*, pages 93–102, O. Içöz and C. Pınar (eds.), Yaşar University, Izmir, Turkey, 2008. 15