A bilevel product pricing problem with ranks and utilities: Models and Algorithms

Abdul Moiz Ansari

A Thesis

in

The Department

of

Mechanical, Industrial and Aerospace Engineering

Presented in Partial Fulfillment of the Requirements

for the Degree of

Master of Applied Science (Industrial Engineering) at

Concordia University

Montréal, Québec, Canada

August 2022

CONCORDIA UNIVERSITY

School of Graduate Studies

This is to certify that the thesis prepared

By: **Abdul Moiz Ansari**

Entitled: **A bilevel product pricing problem with ranks and utilities: Models and Algorithms**

and submitted in partial fulfillment of the requirements for the degree of

Master of Applied Science (Industrial Engineering)

complies with the regulations of this University and meets the accepted standards with respect to originality and quality.

Signed by the Final Examining Committee:

_____ Chair
Dr. Dr. Onur Kuzgunkaya, MIE

_____ External Examiner
Dr. Dr. Navneet Vidyarthi, JMSB

_____ Examiner
Dr. Dr. Onur Kuzgunkaya, MIE

_____ Supervisor
Dr. Dr. Ivan Contreras

Approved by _____
Martin D. Pugh, Chair
Department of Mechanical, Industrial and Aerospace Engineering

_____ 2022

_____
Mourad Debbabi, Dean
Faculty of Engineering and Computer Science

Abstract

A bilevel product pricing problem with ranks and utilities: Models and Algorithms

Abdul Moiz Ansari

Product pricing is a revenue management strategy that facilitates the determination of the prices of a series of products by understanding customer's purchase behavior to optimize the firm's revenue. It is worth mentioning that contributions in the current literature mainly focuses on a single purchasing behavior of the customer either by using utility or rank. To address this problem, we propose a product pricing model incorporating both the customer's utility and the rank. We present a bilevel programming formulation to model this problem and present its corresponding single level formulation. We present two algorithms to assess the validity of the single level formulation and obtain high-quality solutions in reasonable CPU times. Results of computational experiments are presented to assess the performance of the proposed algorithms, in comparison with the single level formulation by solving it with a general purpose solver.

Keywords: Product pricing, Revenue management, Bilevel programming, Reservation price, Rank pricing problem.

Acknowledgments

In the name of Allah, the Most Gracious and the Most Merciful.

All praise is due to Allah, who bestowed upon me this opportunity and gave me the patience to accomplish my master's degree. Peace and blessings upon his final messenger and servant, Muhammad.

After that, I am grateful and indebted to my supervisor Dr. Ivan Contreras for his guidance and patience throughout this journey, helping me accomplish my goals on time. His advice has been a great help, assisting me with the best resources for each problem. Thanks to Professor Daria Terekhov for her support and advice in the initial period of my program.

I must express my gratitude to my parents Abdul Rahim and Malka Javeria, and my brother Ansari Osama for their unconditioned support, continuous prayers, and encouragement.

I would also like to thank my friends back in India for their motivation and support. Sincere thanks to my friends and lab-mates for their help in writing the thesis. Jahidul and Safwan have been great mentors to me throughout this journey, their constant advise helped me to stay focus and on-track. I express my gratitude to my friends Akbar Ali and Sara Ahmed for putting up their time to read my thesis and send their valuable feedback.

Thanks to Concordia University for providing me with this wonderful opportunity and the financial support during my study. At last, thanks to many other people whose names are not mentioned here but have contributed to this research.

Contents

List of Tables

Chapter 1

Introduction

Revenue Management (RM) is the practice of demand management decisions to establish pricing and capacity decisions to maximize the firm's revenue. The practice of RM originated from the airline industry, dating back to the Airline Deregulation Act of 1978, by which the United States removed federal control over fares and routes. This deregulation helped innovation and paved the way for rapid changes in the airline industry. Ever since, RM has been extensively used in almost all industries such as e-commerce, healthcare, manufacturing, and financial services.

A common objective for firms is to maximize profit by providing different services/products at a competitive price to compete and survive in today's challenging markets. Selling these services/products to customers involves various challenges and uncertainties, as competitors selling similar products make use of different methods to have an advantage over other firms. This competition forces the firms to better understand the needs and reactions of the customers, the value the customers assign to a product, and the prices they are willing to pay, these are some of the decisions that the firms should consider in order to maximize their revenue. These decisions are very difficult to study as customers' behavior can be different, and each customer has their unique purchasing criteria. Revenue management aims to provide firms with the necessary techniques to make such decisions easier. RM can be considered as a system that anticipates the customer's decisions and effectively reacts to them to maximize the firm's profit. One of the best definitions of RM is by Cross (2011) who formally defines it as "the art and science of predicting real-time customer demand at the micro-market level and optimizing the price and availability of products". Hence, it

can be termed as a practice of managing customer demand through varying capacity or prices to maximize profitability.

There has been tremendous success stories by firms incorporating the practice of RM. James Whitehurst, former chief operating officer of Delta Airlines, in his keynote address summarized by Garrow and Ferguson (2008) explains how Delta Airlines overcame bankruptcy and a loss of about $2.2bn, within a span of about 2 years, Delta went from being one of the least profitable airlines to the second most profitable airline in the United States. This was achieved by identifying the root causes and implementing the practices of RM. This study highlights that the applications of RM can be the difference between bankruptcy and being the most profitable firm. Cross (2011) reported that the practice of RM helped Marriott Hotel to gain an additional revenue of US$100mn.

The practice of RM can be broadly classified into two categories, price-based which is commonly used by the retailing industries, and quantity-based heavily incorporated by the airlines as described by Talluri et al. (2004). The price-based RM practice has seen widespread applications and improvements. For example, Ford Motor Co. developed a new pricing strategy for its products and in 1998, Ford tested out this new strategy in the first five U.S. sales regions, they collectively beat their profit targets by $1 billion. While the 13 regions that used the old strategy, missed their targets by about $250 million Coy (n.d.). Diverse industries have made use of the price-based RM model, Poldrugovac et al. (2019) provides its significance for the camping industry, Ivanov and Zhechev (2012) for the hotel industry and Bitran and Caldentey (2003) gives an overview of the different pricing models used in RM.

Within the class of price-based RM, one of the widely researched models is the price setting problem. The price setting problem is formulated as a bilevel program, which is an optimization problem that has another optimization problem as its constraints. Thus, the price setting problem involves a hierarchical relationship between two levels, a firm belonging to the upper level of hierarchy (leader) aiming to determine taxes or prices for some activities to maximize their revenue. While the consumers/customers belonging to the lower level of the hierarchy (follower) select the activities to minimize their operating costs. This framework fits in many applications and has been extensively used, for example, the toll optimization of highways studied by Heilporn et al. (2011), truck toll systems, passenger transportation systems, pricing schemes for hotel rooms, car rentals,

etc. It also finds its applications in telecommunications as described by Bouhtou et al. (2007). Labbé and Violin (2016) provides a review of price setting problems which can be modeled as bilevel programs.

Another variant of price setting problems is the product pricing problem, which seeks to determine the prices of a series of products of a firm to maximize its revenue. Consider a firm selling a series of products, the firm always finds it challenging to decide the prices of the products it is offering. Selling the right product at the right price is a difficult task. If the firm decides to set low prices, it will attract more customers but would generate low revenue, on the other hand, setting high prices might put off potential customers. These types of managerial decisions are common but difficult to be accounted for. We find applications of the product pricing problem in the manufacturing and retail industries. Companies like Nike, which designs footwear, release a series of products with variations in styles, colors, qualities, and, most importantly, prices. Phone and Car manufacturing companies such as Apple, Samsung, and Toyota, Honda, respectively, also release a series of products with variations in price and functionalities. These companies have a series of products to account for customer diversity. From the perspective of customers, product diversification leads the customer to hold different reservation prices and rank for various products due to their style, colors, quality, and so on.

To address these challenges, different types of product pricing models have been proposed to model the diverse purchasing decisions of the customers. One of the most common methods to model customer's purchase behavior is the reservation price framework. Shioda et al. (2011) provided a review of the different models in the reservation price framework. In this framework, each customer has a reservation price for each product which denotes how much the customer is willing to pay for that product. Once the pricing strategy is known, the customer purchases the product with the largest utility, which is the difference between the customer's reservation price and the final price of the purchased product. Whereas recently, Calvete et al. (2019) proposed a rank pricing model. In this framework each customer has a rank allocated to the products offered by the firm they are interested in buying, without having any ties between any two products, and they have a fixed budget. Once the firm has set the prices of the products, each customer purchases his most preferred product amongst the ones they can purchase, i.e., maximizing their preferences. If the

customer cannot afford anything they will not make a purchase. Domínguez et al. (2021) extended on this model and developed a rank pricing problem with ties.

The rank pricing model solely focuses on maximizing the preferences of the customers. Whereas, the reservation price model focuses on maximizing the customer's utility, and a purchase is made if the product's price is within their budget or reservation prices. Often in the real world, customers make more complex decisions rather than solely relying on a single criterion such as a rank or utility. The customers might be willing to pay more than their reservation price if the product is to their liking or provides much better functionality. This thesis aims to study and address the limitations of the literature.

In this thesis, we propose an improved product pricing model that builds up on the two pricing models described above. Each customer has a fixed budget, a fixed reservation price, and a preference rank for products they are interested in purchasing without ties. Once the pricing strategy is known, each customer aims to maximize their net utility, which is the sum of the utility and the preference rank. Hence, instead of having a single basis for making decisions, customers make decision based on their preferences as well as savings. We also consider the fact that a customer may even pay beyond their reservation price for a product, if the net utility for that product is the highest. This provides a more robust model for the customer's purchasing decision and focuses more on the objective value of the customers. Also, we assume that each customer buys at most one product i.e., unit demand. In case of ties between the net utility, the customer will choose the product with the highest price which is termed as cooperative behavior in bilevel programming. A cooperative behavior leads to optimistic solutions for the leader, when there are multiple options for the follower, the leader assumes that the follower chooses the option which is most favorable to the leader. On the other hand. a pessimistic solution is one where the leader protects himself against the follower's worst possible reaction. We refer the reader to Dempe (2002) for a more complete discussion on this topic.

We present an example to highlight the difference between our model and the two models described above.

Table (1.1) shows the reservation price, rank, utility, and the net utility of a customer for three products. For a customer that has to choose between three products, given that the customer has a

Customer	Budget	Reservation Price	Rank	Utility	Net Utility	Product Price
Product 1	85	80	1	20	4.8	60
Product 2	85	75	2	17	5	58
Product 3	85	65	3	-15	-0.6	80

Table 1.1: Reservation Price

reservation price, rank for each product, and a fixed budget. We observe that in the case of the rank pricing model by Calvete et al. (2019) the customer will select product three, as it is the highest ranked product within their budget, while in the case of the reservation price model by Shioda et al. (2011) the customer will choose product 1 as this product gives the highest utility. Whereas for the model, we propose, the resulting net utility is highest for product 2, thereby the customer will choose this product.

The main contributions of this thesis are:

- We introduce the Product Pricing Problem with utility and ranks (PP-UR), to have a better representation of the customer's purchasing behavior.
- We present the Bilevel formulation (BPP-UR) for the PP-UR.
- We propose a Single Level Mixed Integer Non Linear formulation of the BPP-UR and then linearize it.
- We develop a metaheuristic and a heuristic to obtain high quality solutions and provide an informal proof through computational experiments for the validity of the single level formulation.

The remainder of this thesis is organized as follows. In Chapter 2, a comprehensive literature review on product pricing problems is presented. Chapter 3 contains the problem description, notations, the BPP-UR and the single level formulation. In Chapter 4 we present the solution methodologies and computational results. Finally, we draw our conclusion in Chapter 5.

Chapter 2

Literature Review

In this chapter, we present a literature review on product pricing problems. We start by discussing some of the variations in the product pricing models, then we describe the utility product pricing model and the rank pricing problem. Finally, we review some of the methodologies used to solve bilevel problems.

2.1 Product Pricing Problems

Product Pricing Problems have gained wide attention in business, as varying the prices of the products is the most natural mechanism for revenue management. Firms often use various methods of dynamic pricing such as promotions and sales, to account for market fluctuations. With such wide applications in different industries, different models have been proposed to capture the behaviors of the customers and further improve the firm's revenue, Talluri et al. (2004) provides a multitude of models with regards to the same. Guruswami et al. (2005) presents an envy-free pricing model, wherein the customer's willingness to pay for the bundle is known. The customer purchases a product only if their utility is non-negative, and the prices are assigned in such a way that the allocation must be envy-free i.e., for given prices of the products the customer would not prefer any other product.

Rusmevichientong et al. (2006) formulated product pricing models based on data collected by General Motors (GM), the availability of accurate datasets helped them leverage their formulations.

The data is collected using an Auto Choice Advisor website, which records customers' budgets and requirements and then recommends a ranked list of vehicles based on their input. The recommended vehicles' price can be beyond their budget provided other requirements are satisfied. They propose two models to capture the customer's behavior, namely a Rank-Pricing and a Min-Pricing model. A Rank-Pricing model is where a customer buys the vehicle under his budget from the recommended list if and only if the higher-ranked vehicles are beyond his budget. For the Min-Pricing model, a customer purchases the cheapest vehicle in their list that is within their budget without considering the ranks. Often, firms segment the customers based on certain conditions, Bucarey et al. (2021) proposed a pricing model for single-minded customer segment who purchase a subset of products i.e., bundle. Each customer has a budget that determines the maximum price the customer is willing to pay for that bundle. The customer purchases their bundle if the total price of the bundle set by the firm is within their budget.

Zhao et al. (2021) categorize the customers into three categories based on the difference between their reference and selling price namely loss-averse, gain-seeking, and loss-neutral, thus studying the different customer segments jointly. Another method of modeling customer behavior is using stochastic choice models, estimating the customer preferences. The most common approach to determine the customer's purchase probability is by using the multinomial logit model (MNL). Hanson and Martin (1996) and Aksoy-Pierson et al. (2013) provide an overview of product pricing models using the MNL approach. Bialas and Karwan (1984) proposed the first bilevel pricing problem with linear problems at both levels.

2.2 Maximum utility product pricing model

In this section, we present the mathematical formulation proposed by Shioda et al. (2011) based on the reservation price framework. As described in Chapter 1, once the pricing strategy is known, the customer purchases the product with the largest utility, which is the difference between the customer's reservation price and the price of the product purchased. Let K and J denote the set of customers and products. Each customer $k \in K$ has a fixed reservation price r_j^k for each product j and a demand of η_k. Decision variables x_j^k are equal to 1 if customer k purchases product j and 0

otherwise. The price set by the firm for product j is denoted by π_j .

$$\max \quad \sum_{k \in K} \sum_{j \in J} \eta_k \pi_j x_j^k \tag{1}$$

$$\text{s.t.} \quad (r_j^k - \pi_j) x_j^k \geq r_i^k x_i^k - \pi_i \qquad \forall k \in K, \forall i, j \in J, \forall i, \neq j \tag{2}$$

$$(r_j^k - \pi_j) x_j^k \geq 0 \qquad \forall k \in K, \forall j \in J \tag{3}$$

$$\sum_{j \in J \cup 0} x_j^k \leq 1 \qquad \forall k \in K \tag{4}$$

$$x_j^k \in \{0, 1\} \qquad \forall k \in K, \forall j \in J \tag{5}$$

$$\pi_j \geq 0 \qquad \forall j \in J \tag{6}$$

In this mathematical model, the objective function (1) maximizes the firm's revenue. Constraints (2) ensure that each customer buys the product with the greatest difference between the reservation price and the price set by the company. Constraints (4) ensure that each customer chooses at most one product $j \in J$ or the dummy product 0, which denotes the no-purchase option. Finally, constraints (5) and (6) make sure that the variables for products' prices are non-negative and the variables for customers' selection choices are binary.

2.2.1 Rank Pricing Problem

Here we present the mathematical formulation proposed by Calvete et al. (2019). As stated in Chapter 1 after the firm has set the prices of the products, each customer purchases their most preferred product amongst the ones they can purchase, i.e., maximizing their preferences. If the customer cannot afford anything they will not make a purchase. Let K and J denote the set of customers and products. Each customer k has a subset of products $S^k \in J$ which the customer is interested to purchase, a preference value s_j^k for each product $j \in S^k$, where $s_j^k > s_i^k$ if customer k prefers product j over product i, the preference of the customers are assumed to be without ties, i.e., the customer do not have the same preference value for any two products. As budgets can be equal for different customers, let $B = \{b^1, ..., b^M\}, M \leq |K|$, the set of different budgets. To represent

the budget of customers a function σ: $K \rightarrow \{1, .., M\}$ is used, such that $\sigma(k) = l$, if budget of customer k is b^l then we can say customer k_1 is richer than customer k_2 if $\sigma(k_1) > \sigma(k_2)$ and the richest customers will be with budget b^M. The price set by the firm for product $j$is denoted by π_j.

$$\max \quad \sum_{k \in K} \sum_{j \in S^k} \pi_j x_j^k \tag{7}$$

$$\text{s.t.} \quad \pi_j \geq 0 \qquad \forall j \in J \tag{8}$$

where $\forall k \in K, x^k$ is an optimal solution of

$$\max_{x^k} \quad \sum_{j \in S^k} s_j^k x_j^k \tag{9}$$

$$\sum_{j \in S^k} x_j^k \leq 1 \tag{10}$$

$$\sum_{j \in S^k} \pi_j x_j^k \leq b^{\sigma(k)} \tag{11}$$

$$x_j^k \in \{0, 1\} \qquad \forall j \in S^k \tag{12}$$

The objective (7) maximizes the revenue of the firm while constraints (8) ensure the prices of products set by the firm are non-negative. The follower's objective which is their preference value is maximized by constraints (9). Constraints (10) ensure that each customer purchases only one product or none, the constraints (11) establishes that the customer only purchases those products which they can afford. Finally, constraints (12) ensure that the product selection variables of the customers are binary.

Rusmevichientong et al. (2006) observed that the optimal solution of the bilevel problem exists such that π_j will always be in the budget set B. Calvete et al. (2019) hence defined a new variable $v_j^l, j \in J, l \in \{1, .., M\}$ representing the prices of products, such that $v_j^l = 1$ if the product j has price b^l .The price of the product j can be replaced using $\pi_j = \sum_{l \in M} b^l v_j^l$. The bilevel problem can be reformulated by replacing the π_j variables by v_j^l. Constraints (8) can be replaced by the

following set of constraints:

$$\sum_{l \in M} v_j^l \leq 1 \qquad \forall j \in J \tag{13}$$

$$v_j^l \in \{0,1\} \qquad \forall j \in J, l \in \{1,..,M\} \tag{14}$$

Constraints (13) ensure that each product j has only one price, and only one binary variable v_j^l can take value 1 in a feasible solution $\forall j \in J$. Constraints (11) can be replaced by the following constraints.

$$x_j^k \leq \sum_{l=1}^{\sigma(k)} v_j^l \qquad \forall k \in K, j \in S^k \tag{15}$$

After replacing it with v_j^l variables , the matrix corresponding to feasible set of each lower level problem of the bilevel problem is totally unimodular, enabling us to relax the integrality constraints (12), according to Wolsey (2020).

The decision variables π_j are parameters for the lower level problems, thus the lower level is a linear program and thus convex. As a result, the Karush–Kuhn–Tucker (KKT) conditions are necessary and sufficient for optimality and the bilevel problem can be formulated into a single level optimization problem. Also, the strong duality theorem can be applied to the lower level problem. Calvete et al. (2019) uses duality theory and transforms the bilevel problem into a single level non-linear optimization problem and further linearizes it to obtain the following single level linear formulation.

$$\max_{v,x,z} \quad \sum_{k \in K} z^k \tag{16}$$

$$\text{s.t.} \quad \sum_{j \in S^k} x_j^k \leq 1 \qquad \forall k \in K \tag{17}$$

$$\sum_{l=1}^{M} v_j^l \leq 1 \qquad \forall j \in I \tag{18}$$

$$\sum_{i \in \overline{B(k,j)}} x_i^k + \sum_{l=1}^{\sigma(k)} v_j^l \leq 1 \qquad \forall k \in K, j \in S^k : \overline{B(k,j)} \neq 0 \qquad (19)$$

$$x_j^k + \sum_{l=\sigma(k)+1}^{M} v_j^l \leq 1 \qquad \forall k \in K, j \in S^k \qquad (20)$$

$$z^k \leq \sum_{l=1}^{\sigma(k)} b^l v_j^l + b^\sigma(k) \sum_{i \in S^k : i \neq j} x_i^k \qquad \forall k \in K, j \in S^k \qquad (21)$$

$$z^k \leq b^\sigma(k) \sum_{j \in S^k} x_j^k \qquad \forall k \in K \qquad (22)$$

$$v_j^l, x_j^k \in \{0,1\}, z^k \geq 0 \qquad \forall k \in K, j \in S^k, l \in \{1,..,M\} \qquad (23)$$

The variables z^k for $k \in K$ represents the profit obtained from customer K, it is obtained by means of constraints (21) and (22). For a detailed description of the single level linearization procedure, we refer the reader to Calvete et al. (2019).

2.3 Solution Methodologies for Bilevel Programs

Bilevel programming problems are a challenging class of optimization problems as they contain two optimization problems hierarchically. The leader at the upper level and the follower at the lower level both aim to optimize their payoff functions. Bilevel problems having linear functions at both levels are in general shown to be NP-hard (Hansen et al. (1992)). Depending on the structure of the bilevel problems, various exact solution methodologies have been proposed. If the lower level is convex and satisfies a suitable constraint qualification, then it can be reformulated into a single level optimization problem using KKT conditions of the lower level problem, or a strong duality theorem can be applied to the lower level as described by Kleinert et al. (2021) in their survey paper. These are some of the most common methods used to solve bilevel problems provided necessary conditions are met. Sinha et al. (2017) provide a review of the different solution methodologies implemented to solve bilevel problems ranging from classical to evolutionary methods.

However, these traditional methods are restricted to convexity conditions and cannot be applied in the case of non-convex and non-linear lower levels. To overcome this complexity, the use of

heuristics is widely used, because they help us to overcome the various challenges of bilevel problems such as non-convexity and non-differentiability. Various metaheuristics have been used to solve such bilevel problems.

Nested Heuristics have been widely practiced to solve bilevel problems. Ma (2016) proposed nested Genetic Algorithms at both upper and lower levels to solve a non-linear bilevel problem, Miao et al. (2016) also made use of a Genetic Algorithm to solve a Mixed Integer Non-Linear Bilevel Program. Angelo and Barbosa (2015) used an Ant Colony and Differential Evolution at the upper and lower level respectively, to solve a bilevel transportation routing problem, Gao et al. (2011) used a particle swarm optimization based algorithm for both levels in the context of a pricing problem in the supply chain. Balakrishnan et al. (2013) and Rajesh et al. (2003) made use of a tabu search. The successful implementations of metaheuristics to solve bilevel problems with non-convex lower levels motivates this work.

2.4 Complexity of the Proposed Model

We propose a bilevel program with discrete variables at the lower level, i.e., non-convex lower level to model the customer behavior considering both rank and utility. In the model we propose, as we consider the reservation prices of each customer for each product, whereas in the case of Calvete et al. (2019) rank-pricing model, the model consisted of only prices, thus they were able to exploit the property where the prices of the products will always be equal to the budget of the customers, exploiting this property led to a totally unimodular constraint matrix in the lower level enabling them to relax the integrality constraints. Thereby, the problem was reformulated to a single level using strong duality as described in (2.2.1). In our case, the prices of the products are not necessarily equal to the budget of customers, hence the lower level is non-convex. Thus, the KKT conditions or strong duality are not applicable. It is interesting to observe the combination of reservation prices and ranks makes the problem difficult to solve. Bilevel programming problems with discrete variables in the lower level and continuous upper-level problems have been studied only by a few articles as highlighted by Fanghänel and Dempe (2009). Hence, we make use of metaheuristics to solve the bilevel problem.

Chapter 3

Problem Definition and Formulation

In this chapter we formally define the bilevel product pricing with utility and ranks (BPP-UR) and propose a singe level linear formulation.

3.1 Problem Description

Let K and J denote the set of customers and products. Each customer $k \in K$ has a positive budget b^k, a subset of products $S^k \in J$ which the customer has interest to purchase, a preference value s_j^k for each product $j \in S^k$, where $s_j^k > s_i^k$ if customer k prefers product j over product i. The preference of the customers are assumed to be without ties, i.e., the customer do not have the same preference value for any two product and it also has a fixed reservation price r_j^k for each product $j \in S^k$ which denotes how much the customer k is willing to spend for product j.

We assume that the customers are interested in purchasing at least one product from the company, i.e., $S^k \neq 0$ for $k \in K$, also each product is included in the list of preferences of at least one customer, i.e., for any product $j \in J$ there exists $k \in K$ such that $j \in S^k$. Else, the customer or the product can be removed from the optimization process. We also assume the customers have unit demand. The reservation price r_j^k of each customer $k \in K$ for product $j \in S^k$ is less than the budget b^k of the customer. We do not account for competitors in this model. We make use of the bilevel model commonly known as the single leader with multiple independent followers.

The BPP-UR aims at establishing prices for a set of products sold by a manufacturer/firm to

maximize its revenue by taking into account the customer's reaction to the prices determined by the firm. The objective function of the leader is to maximize the revenue, whereas the follower i.e., customers seeks to maximize their net utility. The net utility of each customer is defined as the weighted sum of the utility and the preference rank.

3.2 Bilevel product pricing with utility and ranks (BPP-UR)

In this section, we provide the mathematical formulation of BPP-UR. To formulate the problem we use a set of binary decision variables and a set of continuous decision variables.

For each customer $k \in K$ and product $j \in J$ we define:

$$x_j^k = \begin{cases} 1 & \text{if customer } k \text{ purchases product } j \text{ ,} \\ 0 & \text{otherwise.} \end{cases}$$

For each product $j \in J$ we define, π_j which represents the price set by the firm for product j.

The Bilevel Formulation is as follows:

$$\max \sum_{k \in K} \sum_{j \in S^k} \pi_j x_j^k \tag{24}$$

$$\text{s.t. } \pi_j \geq 0 \qquad \forall j \in J \tag{25}$$

where x_j^k is an optimal solution of:

$$\max \left\{ \sum_{k \in K} \sum_{j \in S^k} s_j^k x_j^k, \sum_{k \in K} \sum_{j \in S^k} \left(r_j^k - \pi_j \right) x_j^k \right\} \tag{26}$$

$$\text{s.t. } \sum_{j \in S^k} \pi_j x_j^k \leq b^k \qquad \forall k \in K \tag{27}$$

$$\sum_{j \in S^k} x_j^k \leq 1 \qquad \forall k \in K \tag{28}$$

$$x_j^k \in \{0, 1\} \qquad \forall j \in S^k \tag{29}$$

For the leader (firm), the objective (24) aims to maximize the revenue. Constraints (25) ensure the prices of the products set by the firm are non-negative. For the follower (customer's) problem,

the objective (26) maximizes the net utility. Constraints (27) ensure that the customers will only buy the products within their budget. Constraints (28), enforce the customers to buy one product or none.

The proposed BPP-UR is a bilevel optimization problem with bi-objective functions at the lower level, which is defined as a semivectorial bilevel optimization problem by Bonnel and Morgan (2006). The most common approach in solving multi-objective optimization problems is the weighted-sum method. If all the weights are positive, this approach provides sufficient conditions for Pareto optimality, as shown by Marler and Arora (2010). We introduce a parameter w to scalarize the lower level. Where w is the weights assigned by the decision maker. Thus the follower's objective (24) can be represented as:

$$\max \left(\sum_{k \in K} \sum_{j \in S^k} w s_j^k x_j^k + \sum_{k \in K} \sum_{j \in S^k} (1-w) \left(r_j^k - \pi_j \right) x_j^k \right) \tag{30}$$

3.3 Single Level Non-Linear Reformulation

We present the single level reformulation of the BPP-UR. The objective function of the follower can be represented with the help of additional constraints. We follow the work by Shioda et al. (2011) who represented the objective function of the follower as constraints for the maximum utility product pricing model (MPP) as following:

$$\left(r_j^k - \pi_j \right) x_j^k \geq r_i^k x_j^k - \pi_i, \qquad \forall k \in K, i, j \in J : i \neq j \tag{31}$$

Constraints (31) ensure that if the customer purchases a product, i.e. $x_j^k = 1$, the utility is greater than all the other products. If $x_j^k = 0$ the constraints always holds.

The MPP does not have any budget and ranks. In the case of BPP-UP we can make a similar comparison with certain additional conditions. We can compare the net utility of each customer for each product, with the condition that the comparison must be done only when the products are within the customer's preferred set S^k and budget b^k. Thus, the objective function of the follower

(30) can be represented by using the following constraint:

$$\left(ws_j^k + (1-w)\left(r_j^k - \pi_j\right)\right) x_j^k \geq \left(ws_i^k + (1-w)r_i^k\right) x_j^k - (1-w)\,\pi_i, \qquad \forall k \in K, i, j \in S^k, i : \pi_i \leq b^k, i \neq j \tag{32}$$

The set $\{i \in S^k : \pi_i \leq b^k\}$ can be represented by using disjunctive inequalities. We define auxiliary variable q_i^k as:

$$q_i^k = \begin{cases} 0, & \text{if } \pi_i \leq b^k, \\ 1, & \text{otherwise} \end{cases}$$

The constraint 32 can be represented as:

$$\left(ws_j^k + (1-w)\left(r_j^k - \pi_j\right)\right) x_j^k \geq \left(ws_i^k + (1-w)r_i^k\right) x_j^k - (1-w)\,\pi_i - Mq_i^k, \qquad \forall k \in K, i, j \in S^k : i \neq j \tag{33}$$

The auxiliary variable q_i^k works as a switch when the price of product π_i is less than equal to the budget b^k of the customer , the switch is off and comparison between net utility of each customer for each product within the customer's preferred set S^k is made. But, when budget is more than the price, as the customer cannot purchase those products, the big M ensures that the constraint holds.

Using constraints (33) the single level reformulation of BPP-UP can be presented as following:

$$\text{max} \quad \sum_{k \in K} \sum_{j \in S^k} \pi_j x_j^k \tag{34}$$

$$\text{s.t :} \quad \pi_j \geq 0, \qquad \forall j \in J \tag{35}$$

$$\left(ws_j^k + (1-w)\left(r_j^k - \pi_j\right)\right) x_j^k \geq \left(ws_i^k + (1-w)r_i^k\right) x_j^k - (1-w)\,\pi_i - Mq_i^k, \qquad \forall k \in K, i, j \in S^k, i \neq j \tag{36}$$

$$b^k \geq \pi_i \left(1 - q_i^k\right) + Lq_i^k, \qquad \forall k \in K, i \in S^k \tag{37}$$

$$b^k \leq (\pi_i - \epsilon)\, q_i^k + U\left(1 - q_i^k\right), \qquad \forall k \in K, i \in S^k \tag{38}$$

$$\sum_{j \in S^k} \pi_j x_j^k \leq b^k, \qquad \forall k \in K \tag{39}$$

$$\sum_{j \in S^k} x_j^k \leq 1, \qquad \forall k \in K \qquad (40)$$

$$x_j^k \in (0,1), \qquad \forall k \in K, j \in J \qquad (41)$$

The objective function of the follower is replaced by constraints (36). Constraints (37) and (38) ensure q_i^k takes value of 1 when budget of customer b^k is strictly less than the product price π_k , else 0 otherwise. L and U are the lower and upper bound for the budgets. ϵ is a small value. Without the usage of ϵ, q_i^k variable would not be enforced to 0 when the budget of customer b^k is equal to the price of the product π_k. The rest of the constraints are the same.

3.4 Single Level Linear Reformulation (SLL)

Formulation SLNL is non-linear because of the objective function (34), the constraints (36, (37), (38) and 39). In order to linearize the constraints, we introduce continuous variables p_j^k and t_i^k, which are defined as:

$$p_j^k = \begin{cases} \pi_j, & \text{if } x_j^k = 1, \\ 0, & \text{otherwise} \end{cases}$$

$$t_i^k = \begin{cases} \pi_i, & \text{if } q_i^k = 1, \\ 0, & \text{otherwise} \end{cases}$$

Constraints (36) can be linearized by the following set of constraints which is similar to the linearization technique used by Shioda et al. (2011).

$$p_j^k \geq 0 \qquad \forall k \in K, j \in S^k \qquad (42)$$

$$p_j^k \leq b^k x_j^k \qquad \forall k \in K, j \in S^k \qquad (43)$$

$$p_j^k \leq \pi_j \qquad \forall k \in K, j \in S^k \qquad (44)$$

$$p_j^k \geq \pi_j - b_{\max}\left(1 - x_j^k\right) \qquad \forall k \in K, j \in S^k \qquad (45)$$

Where $b_{max} = \max_{k \in K}\left(b^k\right)$, constraints (42) ensure price of product is positive, constraints (43)

ensure that the price of the selected product $j \in J$ for customer $k \in K$ is less than their budget. Constraints (45) forces p_j^k to take value of π_j when the customer purchases the product, i.e. $x_j^k = 1$. Note that by adding constraints (43), the constraints (39) are not necessary.

Constraint (37) and (38) can be linearized by the following set of constraints:

$$\pi_{\min} q_i^k \leq t_i^k \leq \pi_{\max} q_i^k \qquad \forall k \in K, i \in S^k \qquad (46)$$

$$-\pi_{\max}\left(1 - q_i^k\right) \leq t_i^k - \pi_i \leq -\pi_{\min}\left(1 - q_i^k\right) \qquad \forall k \in K, i \in S^k \qquad (47)$$

Constraints (46) ensure t_i^k takes value of 0, when q_i^k is 0, and constraints (47) enforce t_i^k to take value of 1, when q_i^k is 1. Thus, the corresponding Singe level linear formulation is presented below.

$$\max \quad \sum_{k \in K} \sum_{j \in S^k} p_j^k \qquad (48)$$

$$\text{s.t.} \quad \sum_{j \in S^k} x_j^k \leq 1 \qquad \forall k \in K \qquad (49)$$

$$x_j^k \in (0,1) \qquad \forall k \in K, j \in S^k \qquad (50)$$

$$\left(w s_j^k + (1-w) r_j^k\right) x_j^k - (1-w) p_j^k \geq \left(w s_i^k + (1-w) r_i^k\right) x_j^k - (1-w)\pi_i - M q_i^k \qquad \forall k \in K, i, j \in S^k, i \neq j \qquad (51)$$

$$p_j^k \leq \pi_j \qquad \forall k \in K, j \in S^k \qquad (52)$$

$$p_j^k \leq b^k x_j^k \qquad \forall k \in K, j \in S^k \qquad (53)$$

$$p_j^k \geq \pi_j - b_{\max}\left(1 - x_j^k\right) \qquad \forall k \in K, j \in S^k \qquad (54)$$

$$b^k \geq \pi_i - t_i^k + L q_i^k \qquad \forall k \in K, i \in S^k \qquad (55)$$

$$b^k \leq t_i^k - \epsilon q_i^k + U\left(1 - q_i^k\right) \qquad \forall k \in K, i \in S^k \qquad (56)$$

$$\pi_{\min} q_i^k \leq t_i^k \leq \pi_{\max} q_i^k \qquad \forall k \in K, i \in S^k \qquad (57)$$

$$-\pi_{\max}\left(1 - q_i^k\right) \leq t_i^k - \pi_i \leq -\pi_{\min}\left(1 - q_i^k\right) \qquad \forall k \in K, i \in S^k \qquad (58)$$

$$p_j^k, \pi_j \geq 0 \qquad \forall k \in K, j \in S^k \qquad (59)$$

The objective function of the leader (34) is replaced by (48), the rest of the constraints are same.

Chapter 4

Solution Algorithms

In this chapter, we present a metaheuristic and a nested heuristic to solve the BPP-UR. As previously established in Chapter 2, bilevel problems are a challenging class of optimization problems, the difficulty that arises in solving bilevel problems is that unless a solution is optimal for the lower level problem, it cannot be feasible for the overall problem as experimented by Angelo and Barbosa (2015). This means that an approximate solution method cannot be used to solve lower-level problems as they do not guarantee optimality. Hence, to solve the lower-level problem to optimality, we use an assignment heuristic, which assigns the products to the customers depending on their net utility. The metaheuristic uses a scatter search algorithm to solve the upper level and the assignment heuristic to solve the lower level. Whereas, the heuristic perturbs the prices by using the budgets of the customers and uses the same assignment heuristic to solve the lower level.

4.1 Scatter Search Metaheurestic

Scatter search is an evolutionary metaheuristic which has been successfully applied to solve nonlinear and hard optimization problems. It contrasts with other evolutionary procedures such as genetic algorithms by providing strategic principles for joining solutions, whereas other evolutionary procedures resort to randomisation. It is based on formulations proposed in the 1960s for combining decision rules, Glover (1997) described the scatter search frame work, Glover et al. (2003) presented a simple scatter search tutorial. Scatter Search has been successfully implemented in a

bilevel problem by González Velarde et al. (2015) for determining highway tolls. The methodology includes the following basic elements: Generation of a Population $PSet$, Extraction of Reference Set $RefSet$ from the Population, Combination of subsets from $RefSet$, and update of $RefSet$.

Scatter search differs widely from the population-based metaheuristics as it does not handle the entire population, it only requires a subset of 10 to 20 solutions from the population, this subset is called a reference set. Hence, unlike populations, the reference set of scatter search is relatively small. Evolutionary metaheuristics work in a randomized way to generate new solutions, whereas scatter search chooses two or more elements of a reference set in a systematic way to create new solutions. How the reference set is generated initially and later updated, significantly affects the performance of the scatter search algorithm. The construction of the initial $RefSet$ can be based on quality or diversity, the updating of $RefSet$ can also be performed on these two factors. Generally, a mix of both is selected to initialize the solution to have a balanced $RefSet$.

To solve the BPP-UR, we analyzed the implementation described in Glover et al. (2003). Scatter search was taken as a basis to solve the upper level of the BPP-UR, where the leader(firm) determines the product prices to maximize their revenue. An assignment heuristic is used to solve the problem at the lower level, which is described in Section 3.2. The scatter search algorithm consists of five main steps. The pseudo-code for the scatter search metaheuristic is depicted in Algorithm 1. We denote the solution with the worst upper-level objective in $RefSet1$ as $worstobjref$.

Step 1: The algorithm starts by generating $PSet$, an initial set of diverse solutions for the price variables to be determined by the leader. This initial set is generated randomly to ensure diversity as the improvement solutions are built on it, but at the same time, the initialization is also done in a controlled manner, to guarantee that the generated initiated solutions are within the budget of the customers, hence feasible. The lower bound for each product $j \in J$ is set to the least reservation price r_j^k offered by the customers $k \in K$ for that product and the upper bound is set to the maximum budget of the customers. The algorithm starts by dividing the range of price variables into sub-ranges of equal size with a width of five. Then, a solution is constructed in two steps. First, a subrange is randomly selected and then a value is randomly generated within that subrange. The process is repeated to obtain a population of size $|P|$ denoted as $PSet$.

Step 2: After the generation of the initial set of solutions, the improvement method is applied.

Algorithm 1: Scatter Search Metaheuristic Pseudo-code

Step 0: *Initiate Data*
Read the bilevel instance
Step 1,2: *Generate and Improve Initial Solution*
Use the diversification generator to generate an initial solution set ($PSet$) of size $|P|$.
Apply the Improvement method.
Step 3: *Create Reference Set*
Build $RefSet$ of size b consisting of $b1$ high quality solutions and $b2$ diverse solutions.
Step 4: *Create Subsets*
while terminating condition **do**
  NewElements ← True
  while NewElements **do**
    for Each subset of two in $RefSet$ **do**
      Step 5: *Combine Solutions and Check for Improvement*
      Apply the solution combination method to obtain new solutions x_n and then apply improvement method to obtain x_n^*.
      if x_n^* is not in $RefSet1$ and upper-level objective is greater than equal to upper level objective of $worstobjref$ **then**
        if upper-level objective of x_n^* is equal to upper level objective of $worstobjref$ **then**
          if lower-level objective of x_n^* is better than lower-level objective of $worstobjref$ **then**
            Add x_n^* to $RefSet1$ and delete the worst solution currently in $RefSet1$.
            Make NewElements ← True
          end
        else
          Add x_n^* to $RefSet1$ and delete the worst solution currently in RefSet1.
          Make NewElements ← True
        end
      end
    end
  end
  if terminating condition is not reached **then**
    Build a new $RefSet$, initialize with the solutions currently in $RefSet1$, fill the remaining set using the diversification generator as described above
  end
end

As the solutions generated are feasible, the improvement method always works on feasible solutions. The improvement method used is the Nelder-Mead Simplex Method as proposed in Nelder and Mead (1965), it is a classical local optimizer for unconstrained non-linear optimization problems. It runs for a fixed number of iterations to improve the evaluation of the leader's objective. We make use of the well-known scipy python library provided by Virtanen et al. (2020) to implement the

Nelder-Mead Simplex Method.

Step 3: The next step is to build the $RefSet$, which consists of a combination of high quality and diverse solutions, generally, the high quality and diverse solutions each constitute 50% of the $RefSet$ denoted by sizes $b1$ and $b2$ respectively. The size of $RefSet$ is $b = b1 + b2$. This $RefSet$ is used to generate new solutions by way of Solution Combination methods. The $RefSet$ is constructed by selecting the best $b1$ solutions from the $PSet$ concerning the leader's objective function, the leader's objective for each solution in the $RefSet$ is evaluated by solving the lower level problem described in 3.2 with the help of an assignment heuristic, the decision variables of the upper level i.e., prices are parameters for the lower level, the $PSet$ is sorted in decreasing order of the leader's objective and the desired size of $b1$ are selected. The selected solutions are then deleted from $PSet$. Next, for each solution in $PSet$, the Euclidean distance with each solution $RefSet$ is computed, and the minimum Euclidean distance with $RefSet$ is selected. After computing the minimum Euclidean distance for each solution in $PSet$ with the solutions in $RefSet$, the solution which has the maximum of these minimum Euclidean distances is selected, this solution is then added to the $RefSet$ and deleted from $PSet$. This process is repeated $b2$ times. The resulting $RefSet$ consists of $b1$ high-quality solutions termed as Reference Set 1 ($RefSet1$) and $b2$ diverse solutions termed as Reference Set 2 ($RefSet2$).

The pseudo-code for the assignment heuristic is depicted in Algorithm 2. The assignment heuristic starts by computing the lower-level objective value which is described by: $\left(ws_j^k x_j^k + (1-w)\left(r_j^k - \pi_j\right)x_j^k\right)$ for each customer $k \in K$, for each product $j \in J$ which lies in their preferred set S_k and is within their budget b^k. The customer selects the product which gives them the highest objective value i.e., their net utility, in case of ties the selection will be of the product which has the highest price, as we have assumed a cooperative behavior. Such a selection will be beneficial to the leader.

Algorithm 2: Assignment Heuristic

```
for k=1; k ≤ K; k++ do
    maxobj ← 0
    selectedprod ← 0
    for j=1; j ≤ J; j++ do
        if s_k^j ≠ 0 and b_k ≥ π_j then
            obj = w s_j^k x_j^k + (1 − w) (r_j^k − π_j) x_j^k
            if obj ≥ maxobj then
                if obj =maxobj then
                    if π_j > π_selectedprod then
                        maxobj ← obj
                        selectedprod ← j
                    end
                end
            else
                maxobj ← obj
                selectedprod ← j
            end
        end
    end
end
```

Step 4: After this, for every pair of solutions in the $RefSet$, a subset is generated. These subsets are used to form a linear combination of $RefSet$. The resulting process creates three new combined solutions, which are computed based on the following equations:

C1: $x = x' - d$

C2: $x = x' + d$

C3: $x = x'' + d$

where x' and x'' are the $RefSet$ solutions , $d = r(x'' - x')/2$ and r is a random number between $(0, 1)$.

Step 5: After the solutions are combined, they are processed through the improvement method. If the combined solution is different than the solutions present in $RefSet1$ and gives a better upper-level objective value than the solution with the worst upper-level objective in $RefSet1$, a replacement is done. A caveat here is as this is a bilevel problem if the combined solution's upper-level objective is equal to that of the solution with the worst upper-level objective in the $RefSet1$, these

solutions are also of interest to us, it could be possible for such combined solutions that the lower-level objective is improved, thus those solutions are better bilevel solutions. Hence, we compare the lower-level objectives to check for improvement and if there is an improvement the worst solution with regards to the upper-level objective in $RefSet1$ is replaced by the combined solution. The process is repeated until the combined solutions provide better quality solutions or all the subsets in the $RefSet$ are exhausted to combine solutions.

At this stage, the diversification method is used to generate a new $RefSet2$, the $RefSet1$ remains the same as it contains high-quality solutions. The new $RefSet$ is obtained by combining these two and the process is repeated for a fixed number of iterations.

4.2 Price Perturbation Heuristic

In this section, we present a solution algorithm that provides feasible solutions in a short amount of time. We propose a nested heuristic to solve the upper and lower-level problems. Indeed, optimality of solutions generated from the heuristics is not guaranteed, but good quality solutions are obtained in a very short amount as compared to the scatter search metaheuristic described in the previous section. The upper and lower level of the BPP-UR are relatively simple problems to solve, provided the decision variables of the other level are known. If the leader determines the prices of the products then the lower level becomes an integer optimization problem, where each customer $k \in K$ aims to optimize their net utility, and the purchase decision x_j^k of each customer for a product can be determined using a heuristic. Similarly, if the purchase decisions of the customers are known, then the upper level becomes an unconstrained optimization problem, which is to maximize the price variables within its bound.

The pseudo-code for the price perturbation heuristic is depicted in Algorithm 3. The Heuristic starts by initiating the products' prices to be determined by the leader. We set the price of each product $j \in J$ to the least reservation price r_j^k offered by the customers $k \in K$ for that product. After the price is set, the prices become parameters for the lower level problem, and thus it is a linear integer problem. We make use of the same assignment heuristic described previously at 2, to determine the optimal product selection of each customer x_j^k with regards to their net utility. After

solving the lower level, we perturb the product's prices randomly to one of the customer's budgets. The process is repeated for a fixed number of iterations.

Algorithm 3: Price Perturbation Heuristic

// Initiate Data
Read bilevel instance
Initialize: π_j
bestobjective $\leftarrow 0$
while terminating condition **do**
  Solve the lower level problem using Assignment Heuristic 2
  if objective value of leader is better than the bestobjective obtained **then**
    modify the bestobjective
  end
  Perturb the product prices
end

Chapter 5

Computational Experiments

In this chapter, we present the results of computational experiments. We compare the performance of the scatter search metaheuristic, and price perturbation heuristic with the solutions obtained by solving the SLL of the BPP-UR as described in Chapter 3 using CPLEX. We present four computational sessions. The aim of these sessions are:

(1) Provide a validity of the SLL formulation.

(2) Provide a metaheuristic and a heuristic that are computationally faster than the SLL formulation.

(3) Assess the performance of the heuristics.

In this research, all solution procedures and algorithms are coded in python. CPLEX 12.10 is used as the MIP solver. All computational experiments are conducted on an Intel(R) Core(TM) i7-7700HQ CPU @ 2.80GHz with 24 GB of RAM running on a windows environment. The solver can use up to 8 threads and a total time limit of 4h of CPU time (14400s) is imposed. CPLEX is used in its default settings.

5.1 Comparing the Performances

We use product pricing instances proposed by Shioda et al. (2011), We assume the list of preferred products for each customer be 75% of $|J|$, rounded up, and the ranks are generated randomly.

The size of the instance is determined by the number of customers $|K|$ and the number of products $|J|$. We use four sets of instances of sizes 10 * 10, 20 * 10, 20 * 20, and 90 * 20 to demonstrate our computational results, each Set of instances consists of 10 instances. The scatter search parameters are, Population Size (P) is 100, size of RefSet is 10, with 5 each for RefSet1 and RefSet2, and the maximum number of iterations is set to 3. We run the price perturbation heuristic for 5000 iterations. To assess the metaheuristic and heuristic's performance we run them for fifteen independent times for each instance.

The computational results for Set 1 are summarized in Table 5.1. The first column is the name of the instance. The next 4 columns indicate the results obtained from CPLEX by solving the SLL, the columns Leader Obj and Follower Obj denote the total objective value of the leader and follower respectively. The next column %GAP displays the relative gap between the best integer feasible solution and the best bound found by CPLEX at termination. A %GAP of zero indicates that CPLEX has solved the instance to optimality, and the column CPU(s) denotes the time taken by CPLEX to find the optimal solution in seconds. Columns 6-11 represent the results obtained from solving the BPP-UR using the scatter search metaheuristic. The Leader Obj avg and Follower Obj avg represent the average obtained by running the metaheuristic for fifteen independent times, the next 3 columns denote the minimum, maximum, and average relative optimality gap between the leader's objective obtained by CPLEX and that of the metaheuristic, finally, the average CPU time in seconds is presented. Columns 12-17 represent the results obtained by solving the BPP-UR using the price perturbation heuristic in a similar way.

Table 5.1: Algorithms' Performance: Set 1

	Single Level Linear Formulation				Scatter Search Metaheuristic						Price Perturbation Heuristic					
Instance	LeaderObj	Follower Obj	%GAP	CPU(s)	Leader Obj	Follower Obj	opt gap			CPU(s)	Leader Obj	Follower Obj	opt gap			CPU(s)
					avg	avg	min	max	avg	avg	avg	avg	min	max	avg	avg
test 10k10ta1	203.00	56.00	0..00	0.56	195.93	59.38	2.46	6.90	3.48	2.09	191.80	62.08	4.43	6.40	5.52	0.25
test 10k10ta2	221.00	58.60	0.00	0.16	213.00	58.13	0.90	6.79	3.62	2.38	216.73	60.59	0.90	2.26	1.93	0.24
test 10k10ta3	228.00	55.80	0.00	0.55	222.66	56.96	1.32	3.51	2.34	2.37	223.07	57.97	0.88	3.07	2.16	0.26
test 10k10ta4	227.00	60.80	0.00	0.53	220.27	62.35	0.44	6.17	2.97	2.37	220.93	65.29	0.88	4.41	2.67	0.26
test 10k10ta5	214.00	56.20	0.0.00	0.61	201.20	58.29	2.80	9.81	5.98	2.22	203.20	61.95	3.74	6.07	5.05	0.28
test 10k10ta6	210.00	66.20	0.00	0.83	201.40	66.16	1.43	8.57	4.10	2.27	201.53	66.97	1.43	6.19	4.03	0.25
test 10k10ta7	228.00	53.20	0.00	0.25	224.40	55.87	0.44	4.39	1.67	2.18	225.13	58.69	0.44	1.75	1.26	0.29
test 10k10ta8	213.00	40.60	0.00	2.55	206.53	47.72	1.41	5.63	3.04	2.32	206.60	50.19	2.35	4.23	3.00	0.29
test 10k10ta9	231.00	60.40	0.00	0.63	222.20	63.27	1.73	8.66	3.81	2.51	224.80	62.36	0.87	3.90	2.68	0.27
test 10k10ta10	228.00	51.20	0.00	0.47	217.4	54.31	1.32	9.65	4.65	2.15	219.53	52.40	1.32	4.82	3.71	0.28
Average			0.00	0.71			1.43	7.01	3.56	2.29			1.72	4.31	3.20	0.27

For Set 1 Instances, we observe that CPLEX solves all 10 instances to optimality with an average time of under 1 second. The scatter search metaheuristic also solves within an average time of 3 seconds with an average relative optimality gap of less than 4% for the 10 instances, the average minimum and maximum relative optimality gap for the instances are 1.43% and 7% respectively, and the worst deviation from optimality is within 10%. Whereas, the price perturbation heuristic was able to solve with an average time of fewer than 0.3 seconds, which is even lower than the time taken by the scatter search metaheuristic, the average relative minimum, maximum, and average optimality gap for the 10 instances are 1.72%, 4.31% and 3.20% respectively. The results indicate that both the metaheuristic and heuristic reach high-quality solutions for Set 1 within a short period of time.

The next instance Set 2, consists of 20 Customers $|K|$ and 10 Products $|J|$, computational results are summarized in 5.2.

Table 5.2: Algorithms' Performance: Set 2

Instance	Single Level Linear Formulation				Scatter Search Metaheuristic						Price Perturbation Heuristic					
	LeaderObj	Follower Obj	%GAP	CPU(s)	Leader Obj	Follower Obj	opt gap			CPU(s)	Leader Obj	Follower Obj	opt gap			CPU(s)
					avg	avg	min	max	avg	avg	avg	avg	min	max	avg	avg
test 20k10ta1	432.00	112.80	0.00	34.2	411.60	122.83	1.16	7.41	4.72	6.96	410.93	120.44	4.40	5.32	4.88	0.44
test 20k10ta2	424.00	116.60	0.00	17.70	413.67	108.97	1.42	4.95	2.44	8.36	380.80	102.88	9.20	11.08	10.19	0.47
test 20k10ta3	415.00	106.40	0.00	23.06	402.13	109.43	0.48	4.58	3.10	6.70	375.60	95.15	9.16	9.88	9.49	0.44
test 20k10ta4	422.00	119.00	0.00	26.09	400.33	118.08	2.61	7.58	5.14	8.77	391.87	116.20	5.69	8.06	7.14	0.57
test 20k10ta5	392.00	115.20	0.00	55.27	369.53	110.00	3.32	8.42	5.73	6.72	367.73	114.56	5.36	7.14	6.19	0.45
test 20k10ta6	434.00	110.60	0.00	14.78	408.93	117.60	3.92	8.06	5.78	7.33	414.27	120.96	3.00	5.30	4.55	0.47
test 20k10ta7	411.00	118.40	0.00	64.45	392.47	115.23	2.43	6.33	4.51	7.99	393.53	122.20	3.65	4.87	4.25	0.45
test 20k10ta8	434.00	117.20	0.00	22.84	415.60	117.44	1.84	8.76	4.24	7.10	415.20	118.32	2.76	5.07	4.33	0.48
test 20k10ta9	418.00	111.80	0.00	29.70	402.13	109.65	1.91	5.98	3.80	6.57	397.87	111.03	4.07	5.50	4.82	0.45
test 20k10ta10	435.00	114.60	0.00	60.33	423.47	116.77	0.23	6.90	2.65	7.60	424.73	118.75	1.84	3.22	2.36	0.43
Average			0.00	34.84			1.93	6.90	4.21	7.41			4.91	6.54	5.82	0.47

For the Set 2 Instances, we observe that CPLEX is again able to solve all the 10 instances to optimality with an average time of about 35 seconds. The results from the metaheuristic are obtained within an average time of below 8 seconds, with an average relative optimality gap of about 4% for the 10 instances, the average minimum and maximum relative optimality gap for the instance set are 1.93% and 6.90% respectively. For all the fifteen iterations for the instances no solution's relative optimality gap has gone beyond 9% which shows the metaheuristic can achieve high-quality results consistently. The price perturbation heuristic was able to solve with an average time of fewer than 0.5 seconds, the average relative minimum, maximum and average optimality gap for the instance set is 4.91%, 6.54%, and 5.82% respectively. We observe that the deviations are

clustered around 5%, indicating the consistency of the algorithm. The heuristic was able to obtain high-quality solutions within a fraction of a second, The metaheuristic was also able to obtain high-quality solutions within 1/5th of the time utilized by CPLEX to solve to optimality.

The next Instance Set consists of 20 Customers $|K|$ and 20 Products $|J|$, computational results are summarized in 5.3.

Table 5.3: Algorithms' Performance: Set 3

Instance	Single Level Linear Formulation				Scatter Search Metaheuristic						Price Perturbation Heurestic					
	LeaderObj	Follower Obj	%GAP	CPU(s)	Leader Obj	Follower Obj	opt gap			CPU(s)	Leader Obj	Follower Obj	opt gap			CPU(s)
					avg	avg	min	max	avg	avg	avg	avg	min	max	avg	avg
test 20k20ta1	469.00	289.40	0.00	37.55	433.27	256.72	2.77	8.74	6.38	11.48	441.47	273.17	3.84	6.82	5.87	0.67
test 20k20ta2	452.00	259.4	0.00	27.58	433.27	261.64	2.21	5.97	4.14	11.61	433.27	262.71	2.43	5.09	4.14	0.67
test 20k20ta3	442.00	263.20	0.00	168.03	418.60	260.25	2.26	9.05	5.29	13.35	421.73	266.63	3.85	5.66	4.59	0.66
test 20k20ta4	458.00	263.00	0.00	0.39	435.33	252.75	1.75	7.21	4.91	10.88	440.13	248.87	3.28	4.37	3.90	0.68
test 20k20ta5	457.00	264.00	0.00	6.80	433.33	262.68	2.63	8.32	5.18	12.94	434.20	256.07	3.94	6.13	4.99	0.75
test 20k20ta6	454.00	274.60	0.00	200.08	428.53	254.01	3.08	8.15	5.61	11.85	430.93	279.21	4.19	5.95	5.08	0.72
test 20k20ta7	464.00	271.00	0.00	1123.30	445.07	259.67	2.59	5.82	4.08	12.51	443.67	266.25	3.23	5.17	4.38	0.68
test 20k20ta8	443.00	267.00	0.00	0.47	428.13	258.83	2.03	5.19	3.36	11.97	428.20	260.27	2.48	3.84	3.34	0.70
test 20k20ta9	450.00	247.00	0.00	54.64	428.93	265.63	2.00	6.44	4.68	12.48	424.47	267.23	4	6.67	5.67	0.69
test 20k20ta10	452.00	259.80	0.00	109.7	431.00	252.43	1.77	6.44	4.65	13.86	425.07	257.12	4.87	6.64	5.96	0.68
Average			0.00	172.85			2.31	7.15	4.83	12.29			3.61	5.63	4.79	0.69

For the Set 3 Instances, CPLEX solves to optimality all the 10 instances in the set with an average time of about 3 minutes. On the other hand, the metaheuristic was able to obtain solutions with an average relative optimality gap of about 5% within 13 seconds, its average worst and best deviations are 7.15% and 2.15% respectively. The heuristic achieves a solution much quicker as compared to CPLEX or the metaheuristic. achieving solutions within an average relative optimality gap of less than 5% within a second. Its worst deviations fare better than the metaheuristic with an average value of 5.63%, the average minimum deviation is 3.61%. Both the metaheuristic and heuristic consistently achieve high-quality solutions quickly within small deviations from the optimal solution obtained by CPLEX.

Lastly, we run our algorithms on a much larger instance consisting of 90 Customers $|K|$ and 20 Products $|J|$, the results for instance Set 4 are presented in two tables, in a slightly different format as CPLEX was not able to solve these instances to optimality. Table 5.4 provides the results obtained from CPLEX by solving the SLL, Columns 2 and 3 show the Lower and Upper Bound found by CPLEX. The next column %GAP represents the gap between the upper and the lower bound. Finally, the time taken by CPLEX in seconds is shown.

Table 5.4: SLL Performance: Set 4

	Single Level Linear Formulation				
Instance	LeaderObj LB	LeaderObj UB	Follower Obj	%GAP	CPU(s)
test 90k20ta1	1935.00	2094.00	1169.00	8.22	time
test 90k20ta2	1968.00	2091.00	1173.60	6.25	time
test 90k20ta3	1961.00	2077.95	1176.80	5.96	time
test 90k20ta4	1970.00	2083.00	1084.40 0	5.74	time
test 90k20ta5	1941.00	2072.00	1126.60	6.75	time
test 90k20ta6	1925.00	2073.00	1196.00	7.69	time
test 90k20ta7	1933.00	2053.89	1179.00	6.25	time
test 90k20ta8	1945.00	2061.00	1175.20	5.96	time
test 90k20ta9	1971.00	2070.00	1163.40	5.02	time
test 90k20ta10	1949.00	2089.99	1106.40	7.23	time
Average			0	6.51	

Table 5.5 provides a comparison between the scatter search metaheuristic, and the price perturbation heuristic with the results obtained from CPLEX. The table shows the best objective value of leader $(BestOFL)$, the average objective value of the leader $(AverageOFL)$, and the average objective value of the follower $(AverageOFF)$ obtained from the fifteen runs of the metaheuristic and the heuristic. We calculate the %GAP between the metaheuristic's solutions and the lower bound found by CPLEX as $\frac{LB-OFL}{LB} \times 100$, the GAP between the metaheuristic's best and average OFLs and CPLEX's LB are calculated in the respective %GAP LB columns. Likewise, we also calculate the gap between the metaheuristic's solutions and the upper bound found by CPLEX as $\frac{UB-OFL}{UB} \times 100$ which are presented in the % GAP UB columns for the best and average solutions. Finally, we do the same calculations for the price perturbation heuristic.

Table 5.5: Algorithms' Performance: Set 4

	Scatter Search Metaheuristic								Price Perturbation Heuristic							
Instance	Best			Average					Best			Average				
	Best OFL	%GAP LB	%GAP UB	Average OFL	Average OFF	%GAP UB	%GAP LB	CPU(s)	Best OFL	%GAP LB	%GAP UB	Average OFL	Average OFF	%GAP LB	%GAP UB	CPU(s)
test 90k20ta1	1870.00	3.36	10.70	1856.53	1090.87	4.06	11.36	292.89	1881.00	2.79	10.17	1868.27	1113.92	3.45	10.78	3.00
test 90k20ta2	1881.00	4.42	10.04	1853.47	1146.27	5.82	11.34	243.86	1889.00	4.01	9.66	1860.67	1146.91	5.45	11.02	2.58
test 90k20ta3	1901.00	3.06	8.52	1878.27	1138.01	4.22	9.61	252.86	1906.00	2.80	8.28	1895.93	1194.33	3.32	8.76	2.85
test 90k20ta4	1913.00	2.89	8.16	1894.73	1137.37	3.82	9.04	287.00	1903.00	3.40	8.64	1891.60	1891.60	3.98	9.19	2.88
test 90k20ta5	1895.00	2.37	8.54	1866.60	1128.19	3.83	9.91	252.63	1871.00	3.61	9.70	1862.60	1179.52	4.04	10.11	3.00
test 90k20ta6	1899.00	1.35	8.39	1867.67	1155.05	2.98	9.91	281.48	1875.00	2.60	9.55	1864.13	1189.23	3.16	10.08	2.93
test 90k20ta7	1887.00	2.38	8.13	1863.47	1123.04	3.60	9.27	251.44	1886.00	2.43	8.17	1877.00	1168.17	2.90	8.61	2.86
test 90k20ta8	1896.00	2.52	8.01	1838.53	1146.72	5.47	10.79	331.22	1848.00	4.99	10.33	1830.93	1120.96	5.86	11.16	2.95
test 90k20ta9	1908.00	3.20	7.83	1873.60	1169.23	4.94	9.49	307.28	1894.00	3.91	8.50	1882.13	1193.39	2.77	9.08	2.77
test 90k20ta10	1902.00	2.41	8.99	1872.00	1134.80	3.95	10.43	301.10	1859.00	4.62	11.05	1846.73	1127.96	2.77	11.64	3.40
Average		2.80	8.73			4.27	10.12	280.17		3.52	9.41			3.77	10.04	2.92

The computational results from Table 5.4 show that CPLEX is not able to solve the instances to optimality within the time limit of 14400s. CPLEX achieves an average gap of 6.51% between its

lower bound and the best upper bound it achieves. The metaheuristic and the heuristic were able to find solutions within a 5% deviation from CPLEX's lower bound for all the instances. Although it doesn't outperform CPLEX results obtained were very consistent from both the algorithms with average deviations from CPLEX's lower bound around 4%. For all the instances the algorithm's results are within 11% deviation from CPLEX's upper bound, with the average deviation lying near 10%. If we observe the deviations of best solutions obtained by the algorithms with the LB of CPLEX's solution, the average is within 4%. The metaheuristic takes an average time of 280s while the heuristic runs for just 3 seconds.

We observe from the results of the four sets of instances, that the algorithms show good performance against CPLEX in terms of speed, generating high-quality solutions in a short amount of time consistently, thereby providing a cheaper computational option.

Chapter 6

Conclusion and Future Research

In this thesis we studied a product pricing problem with ranks and utilities, extending the existing formulations on the maximum product pricing problem and the rank pricing problem. The problem aims at determining optimal product prices for a firm offering a series of products to unit demand customer segments. We modeled the problem as a bilevel problem and presented its single level linear formulation. We then developed a scatter search metaheuristic and a price perturbation heuristic.

We presented a computational study of the bilevel and single level formulation. We observe that the solutions from the bilevel formulation were not outperforming the single-level formulation, providing an informal validity for the SLL. CPLEX was able to solve the small and medium-sized instances to optimality. For the larger-size instances, CPLEX could not solve it to optimality in the time limit we specified. Furthermore, the algorithms performed much faster and were able to achieve high-quality results within a short amount of time. The metaheuristic and the heuristic obtained near-optimal solutions for the small and medium-sized instances, for larger-size instances, they achieved an average optimality gap of 4.27% and 3.77% respectively.

For future research, several directions are possible. Additional work can be done on modeling customer purchase behaviors. One could consider multiple leaders with multiple followers, modeling the firm's competitors to make it even more realistic. Mathematical proof for transforming such non-linear bilevel problems into a single level formulation can be studied.

References

Aksoy-Pierson, M., Allon, G., & Federgruen, A. (2013). Price competition under mixed multinomial logit demand functions. *Management Science*, *59*(8), 1817–1835.

Angelo, J. S., & Barbosa, H. J. (2015). A study on the use of heuristics to solve a bilevel programming problem. *International Transactions in Operational Research*, *22*(5), 861–882.

Balakrishnan, A., Banciu, M., Glowacka, K., & Mirchandani, P. (2013). Hierarchical approach for survivable network design. *European Journal of Operational Research*, *225*(2), 223–235.

Bialas, W. F., & Karwan, M. H. (1984). Two-level linear programming. *Management science*, *30*(8), 1004–1020.

Bitran, G., & Caldentey, R. (2003). An overview of pricing models for revenue management. *Manufacturing & Service Operations Management*, *5*(3), 203–229.

Bonnel, H., & Morgan, J. (2006). Semivectorial bilevel optimization problem: penalty approach. *Journal of Optimization Theory and Applications*, *131*(3), 365–382.

Bouhtou, M., van Hoesel, S., van der Kraaij, A. F., & Lutton, J.-L. (2007). Tariff optimization in networks. *INFORMS journal on computing*, *19*(3), 458–469.

Bucarey, V., Elloumi, S., Labbé, M., & Plein, F. (2021). Models and algorithms for the product pricing with single-minded customers requesting bundles. *Computers & Operations Research*, *127*, 105139.

Calvete, H. I., Domínguez, C., Galé, C., Labbé, M., & Marin, A. (2019). The rank pricing problem: models and branch-and-cut algorithms. *Computers & operations research*, *105*, 12–31.

Coy, P. (n.d.). The power of smart pricing companies are fine-tuning their price strategies–and it's paying off. *Bloomberg*. Retrieved 2010-04-10, from https://www.bloomberg.com/

news/articles/2000-04-09/the-power-of-smart-pricing

Cross, R. G. (2011). *Revenue management: Hard-core tactics for market domination*. Currency.

Dempe, S. (2002). *Foundations of bilevel programming*. Springer Science & Business Media.

Domínguez, C., Labbé, M., & Marín, A. (2021). The rank pricing problem with ties. *European Journal of Operational Research*, *294*(2), 492–506.

Fanghänel, D., & Dempe, S. (2009). Bilevel programming with discrete lower level problems. *Optimization*, *58*(8), 1029–1047.

Gao, Y., Zhang, G., Lu, J., & Wee, H.-M. (2011). Particle swarm optimization for bi-level pricing problems in supply chains. *Journal of Global Optimization*, *51*(2), 245–254.

Garrow, L., & Ferguson, M. (2008). Revenue management and the analytics explosion: Perspectives from industry experts. *Journal of Revenue and Pricing Management*, *7*(2), 219–229.

Glover, F. (1997). A template for scatter search and path relinking. In *European conference on artificial evolution* (pp. 1–51).

Glover, F., Laguna, M., & Martí, R. (2003). Scatter search. In *Advances in evolutionary computing* (pp. 519–537). Springer.

González Velarde, J. L., Camacho-Vallejo, J.-F., & Pinto Serrano, G. (2015). A scatter search algorithm for solving a bilevel optimization model for determining highway tolls. *Computación y Sistemas*, *19*(1), 05–16.

Guruswami, V., Hartline, J. D., Karlin, A. R., Kempe, D., Kenyon, C., & McSherry, F. (2005). On profit-maximizing envy-free pricing. In *Soda* (Vol. 5, pp. 1164–1173).

Hansen, P., Jaumard, B., & Savard, G. (1992). New branch-and-bound rules for linear bilevel programming. *SIAM Journal on scientific and Statistical Computing*, *13*(5), 1194–1217.

Hanson, W., & Martin, K. (1996). Optimizing multinomial logit profit functions. *Management Science*, *42*(7), 992–1003.

Heilporn, G., Labbé, M., Marcotte, P., & Savard, G. (2011). Valid inequalities and branch-and-cut for the clique pricing problem. *Discrete Optimization*, *8*(3), 393–410.

Ivanov, S., & Zhechev, V. (2012). Hotel revenue management–a critical literature review. *Tourism: an international interdisciplinary journal*, *60*(2), 175–197.

Kleinert, T., Labbé, M., Ljubić, I., & Schmidt, M. (2021). A survey on mixed-integer programming techniques in bilevel optimization. *EURO Journal on Computational Optimization*, *9*, 100007.

Labbé, M., & Violin, A. (2016). Bilevel programming and price setting problems. *Annals of operations research*, *240*(1), 141–169.

Ma, S. (2016). A nonlinear bi-level programming approach for product portfolio management. *SpringerPlus*, *5*(1), 1–18.

Marler, R. T., & Arora, J. S. (2010). The weighted sum method for multi-objective optimization: new insights. *Structural and multidisciplinary optimization*, *41*(6), 853–862.

Miao, C., Du, G., Xia, Y., & Wang, D. (2016). Genetic algorithm for mixed integer nonlinear bilevel programming and applications in product family design. *Mathematical Problems in Engineering*, *2016*.

Nelder, J. A., & Mead, R. (1965). A simplex method for function minimization. *The computer journal*, *7*(4), 308–313.

Poldrugovac, K., Janković, S., & Peršić, M. (2019). The significance of competitive pricing and revenue management in the camping industry. *International Journal of Revenue Management*, *11*(1-2), 76–88.

Rajesh, J., Gupta, K., Kusumakar, H. S., Jayaraman, V. K., & Kulkarni, B. D. (2003). A tabu search based approach for solving a class of bilevel programming problems in chemical engineering. *Journal of Heuristics*, *9*(4), 307–319.

Rusmevichientong, P., Van Roy, B., & Glynn, P. W. (2006). A nonparametric approach to multiproduct pricing. *Operations Research*, *54*(1), 82–98.

Shioda, R., Tunçel, L., & Myklebust, T. G. (2011). Maximum utility product pricing models and algorithms based on reservation price. *Computational Optimization and Applications*, *48*(2), 157–198.

Sinha, A., Malo, P., & Deb, K. (2017). A review on bilevel optimization: from classical to evolutionary approaches and applications. *IEEE Transactions on Evolutionary Computation*, *22*(2), 276–295.

Talluri, K. T., Van Ryzin, G., & Van Ryzin, G. (2004). *The theory and practice of revenue management* (Vol. 1). Springer.

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., … SciPy 1.0 Contributors (2020). SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, *17*, 261–272. doi: 10.1038/s41592-019-0686-2

Wolsey, L. A. (2020). *Integer programming*. John Wiley & Sons.

Zhao, N., Wang, Q., Cao, P., & Wu, J. (2021). Pricing decisions with reference price effect and risk preference customers. *International Transactions in Operational Research*, *28*(4), 2081–2109.