Assortment Optimization under the Multinomial Logit Model with Product Synergies

Venus Lo[a,1,*], Huseyin Topaloglu[b]

[a]School of Operations Research and Information Engineering, Cornell University, Ithaca, New York, United States 14853
[b]School of Operations Research and Information Engineering, Cornell Tech, New York, New York, United States 10044

Abstract

In synergistic assortment optimization, a product's attractiveness changes as a function of which other products are offered. We represent synergy structure graphically. Vertices denote products. An edge denotes synergy between two products, which increases their attractiveness when both are offered. Finding an assortment to maximize retailer's expected profit is NP-hard in general. We present efficient algorithms when the graph is a path, a tree, or has low treewidth. We give a linear program to recover the optimal assortment for paths.

Keywords: Assortment optimization, synergy, multinomial logit model

1. Introduction

Modeling customers' purchasing behaviours is critical in retail operations because it affects the retailer's decisions on which products to offer in order to maximize his profit. Early work in inventory management assumed that demands are independent of the offered assortment. Subsequent work in revenue management recognized that demand for a product might decrease if customers have more options. However, most choice models do not allow for synergy effects. Synergy can increase the demand for a product when it is seen with some other products.

We study the assortment optimization problem under a synergistic version of the multinomial logit model (MNL). Customers associate a preference weight with each product, and the preference weight can increase via synergy. Synergy occurs between a pair of products when the retailer offers both of them in his assortment, even when customers purchase at most one product. The purchase probabilities of either product may increase or decrease, depending on the increase in preference weight and the weights of the other product. Marketing research shows that retailers can increase demand by offering a less attractive product to highlight the target product. The assortment optimization problem is to select a subset of products to offer, in order to maximize the retailer's expected profit.

Our Contributions: We consider a retailer who has access to n products. Each product has a base preference weight, which describes a customer's preference for the product when it is seen alone. Each pair of products, i and j, also have a pair of synergy weights, which describes how synergy from product i acts on j and vice versa. The purchase probability of a product is proportional to its preference weight in the offered assortment.

We use a graph to depict pairs of products with positive synergy when they are seen together. The assortment optimization problem under synergistic MNL is NP-hard for general synergy graphs. We restrict the product pairs with synergy and study the special cases of a synergy path and a synergy tree. We present algorithms that find the optimal assortment via dynamic programming, with runtime polynomial in the number of products. We extend our dynamic program to consider synergy graphs with low treewidth. In the case of the synergy path, we present a linear program that can recover the optimal assortment.

Literature Review: Our paper is inspired by the welfare-based choice models of [9]. The authors introduced a welfare function, which defined the welfare of an assortment as a function of its products' utilities. Purchase probabilities are given by the gradients of the welfare functions with respect to each product. The welfare function has three properties. Monotonicity ensures that welfare increases when all the utilities increase. Translation invariance states that purchase probabilities remain the same if all utilities increased by the same amount. Convexity ensures that a high-utility product improves welfare more than multiple low-utility products.

The authors defined operations on welfare-based choice models which create new welfare-based choice models [9]. One resulting model is the basis for our synergistic MNL. In their model, if two products create synergy, then the sum of their synergy weights is equal to a weighted geometric mean of their base preference weights. We allow

*Corresponding author
Email addresses: venus.hl.lo@cityu.edu.hk (Venus Lo), topaloglu@orie.cornell.edu (Huseyin Topaloglu)
[1]Present address - Department of Management Sciences, City University of Hong Kong, Kowloon Tong, Hong Kong SAR

synergy weights to be any non-negative values. They focused on defining a new class of choice models, and did not consider assortment optimization. We focus on the latter problem.

Evidence of synergy exists in marketing literature. Product sales can increase when an inferior product is introduced [23]. In one study, a retailer selling a bread maker introduced a second, over-priced bread maker to his store. The demand for the original bread maker increased even though the assortment became larger. Some products can trigger a change in preference; [11] found that cookies sales increased in a cafeteria if applesauce was offered but not if green beans were offered.

A traditional approach to choice modeling is via utility-maximization. In the class of random utility models, a customer's utility for a product is the sum of its mean utility plus a random noise. MNL is the most famous model in this class, and the random noise for each product follows an independent standard Gumbel distribution [15, 18]. A product's preference weight is the exponential of its mean utility, and its purchase probability is proportional to its preference weight in the assortment.

MNL has the independence of irrelevant alternative (IIA) property: the ratio of two products' purchase probabilities is unchanged regardless of what other products are offered. Our model follows MNL such that purchase probabilities are proportional to preference weights, but it is not subject to IIA because synergy can increase the preference weights of products. We describe extensions to MNL; details of these choice models can be found in [26].

In the nested logit model, products are partitioned into nests by similarities. A customer chooses a nest, and then chooses a product within her chosen nest. The retailer can increase the probability that a customer chooses a nest by adding products to the nest, but that might not increase the purchase probability of individual products. In fact, incorporating synergy into the nested logit model might sacrifice its utility-maximizing property. The nested logit model has been extended to the d-level nested logit model with $d > 2$ levels [14], and to the generalized extreme value model where products belong to multiple nests [26].

In the mixed logit model (MMNL), different customer types have different sets of preference weights [17, 5, 21]. MMNL can approximate any random utility choice model [17], and [6] gave a fully-polynomial time approximation scheme (FPTAS) to compute an assortment that guarantees $(1-\epsilon)$-fraction of the optimal expected profit.

The random noises of utilities do not have to follow the Gumbel distribution. In the probit model, the noises follow a multivariate normal distribution and the model does not exhibit the IIA property [25]. Choice models outside the class of random utility models include the Markov Chain choice model [1, 8] and the non-parametric choice model [16, 12]. [7] studied a special case of the non-parametric choice model, where customers consider at most k products for some small, fixed k.

Our underlying parametric problem requires maximizing a quadratic function subject to binary variables. Unconstrained quadratic binary programming is NP-hard, and can be rewritten as an integer program. This problem can be classified by its underlying graph, where vertex i represents variable x_i and edge ij exists if the coefficient of the quadratic term $x_i x_j$ is non-zero. [19] studied the graph and the related integer program, and identified cases where the convex hull of the feasible region is integral. We use their approach to classify cases of the assortment optimization problem under synergistic MNL that can be solved efficiently. A survey of quadratic binary programming can be found in [13].

Our linear program for finding the optimal assortment under a synergy path is similar to the sales-based linear program (SBLP) described by [10]. The authors studied the network revenue management problem under a generalization of MNL, where the retailer offers assortments over time subject to resource constraints. When there is one sales period, then their SBLP recovers the optimal assortment for their model. We construct our version of the SBLP by taking the dual of the linear program formulation of our dynamic program.

Organization: In Section 2, we describe the synergistic MNL model and its corresponding parametric problem, as well as the synergy graph. In Section 3, we focus on the synergy path and synergy tree, and use dynamic programming to solve the parametric problem. In Section 4, we present a linear program which recovers the optimal assortment when we have a synergy path. In Section 5, we consider other forms of the synergy weights and incorporate synergy into MMNL. We conclude in Section 6.

2. General Model and NP-hardness

We describe the general model and the parametric problem, and show that the assortment optimization problem under general synergy effects is NP-hard.

2.1. The Model

The retailer has access to n products: $\{1, \ldots, n\}$. Product i generates a profit of r_i when it is purchased by a customer. We do not require $r_i > 0$ for all products, only that at least one product has positive profit. Note that the retailer can offer products with negative profit if they increase the demand of highly profitable products. The retailer chooses an assortment from $\{1, \ldots, n\}$ to offer, which we denote by a binary vector $x \in \{0,1\}^n$ such that $x_i = 1$ if product i is in the assortment and 0 otherwise.

We describe customers' preferences by preference weights, and purchase probabilities are proportional to these weights. The preference weight of product i is a function of the assortment x because synergy from other products in the assortment can increase its preference weight. Product i has a base preference weight of $u_i \geq 0$ when it is the only product in the assortment. When product j is added to the assortment, synergy between products i and j increases

the preference weight of product i additively. We denote the synergy effect of product j on i by $v_i^j \geq 0$. Simultaneously, product i increases the preference weight of product j by v_j^i. Given an assortment x, the preference weight of product i when it is offered is $u_i + \sum_{j \neq i} v_i^j x_j$.

We discuss the possibility of negative synergy weights in Section 5, but we point out that it is possible for product j to create synergy for product i without synergy in the reverse direction by setting $v_i^j > 0$ and $v_j^i = 0$. Then the probability of selling product i may increase when product j is added into the assortment, but the probability of selling product j cannot increase when product i is added into the assortment.

Upon seeing assortment x, a customer's probability of purchasing product i is proportional to the preference weight of product i in the assortment, along with the no-purchase option. The no-purchase option is her ability to leave the store without a purchase. We scale the preference weight of the no-purchase option to 1 without loss of generality. In response to assortment x, the probability that the customer purchases a product i is:

$$P_i(x) = \frac{\left(u_i + \sum_{j \neq i} v_i^j x_j\right) x_i}{1 + \sum_{k=1}^n \left(u_k + \sum_{j \neq k} v_k^j x_j\right) x_k}.$$

The customer purchases at most one product, and synergy simply makes products look more attractive when they are offered concurrently. The retailer's problem is to find an assortment x that maximizes his expected profit: $\Pi(x) = \sum_{i=1}^n r_i P_i(x)$.

For cleaner notation, we define $\bar{r}_{i,j}$ as the weighted average of the profits of products i and j, and $v_{i,j}$ as the total increase to the preference weight of the assortment due to synergy when both products i and j are offered, with the convention that $i < j$. When $v_j^i = v_i^j = 0$ so that there is no synergy in either directions, then $\bar{r}_{i,j} = 0$ and $v_{i,j} = 0$. Otherwise, define $\bar{r}_{i,j} = (r_i v_i^j + r_j v_j^i)/(v_i^j + v_j^i)$ and $v_{i,j} = v_i^j + v_j^i$.

Using this notation, our assortment optimization problem $\max_{x \in \{0,1\}^n} \Pi(x)$ expands out to:

$$\max_{x \in \{0,1\}^n} \frac{\sum_{i=1}^n r_i u_i x_i + \sum_{i=1}^{n-1} \sum_{j=i+1}^n \bar{r}_{i,j} v_{i,j} x_i x_j}{1 + \sum_{i=1}^n u_i x_i + \sum_{i=1}^{n-1} \sum_{j=i+1}^n v_{i,j} x_i x_j}. \quad (1)$$

2.2. The Parametric Problem

We apply a standard parametrization technique for fractional combinatorial problems [20]. Suppose there exists an assortment x with expected profit greater or equal to δ. By rearranging $\Pi(x) \geq \delta$, we observe:

$$\sum_{i=1}^n (r_i - \delta) u_i x_i + \sum_{i=1}^{n-1} \sum_{j=i+1}^n (\bar{r}_{i,j} - \delta) v_{i,j} x_i x_j \geq \delta.$$

We can maximize the left side over all assortments and the inequality would still hold. The parametric problem

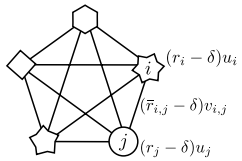

Figure 1: Synergy graph on five products - Vertices i and j represent products and the edge ij represents the synergy between these products. In the parametric problem, the vertices contribute $(r_i - \delta)u_i$ and $(r_j - \delta)u_j$ respectively, and the edge contributes $(\bar{r}_{i,j} - \delta)v_{i,j}$.

corresponding to Problem (1) is:

$$h(\delta) = \max_{x \in \{0,1\}^n} \sum_{i=1}^n (r_i - \delta) u_i x_i + \sum_{i=1}^{n-1} \sum_{j=i+1}^n (\bar{r}_{i,j} - \delta) v_{i,j} x_i x_j. \quad (2)$$

Claim 1. *Given $h(\delta)$ as defined above, let δ^* be the optimal expected profit of our assortment optimization problem. Then the following are true: i) $h(\delta) > \delta$ if $\delta < \delta^*$, ii) $h(\delta) < \delta$ if $\delta > \delta^*$, and iii) $h(\delta) = \delta$ if $\delta = \delta^*$.*

Suppose we can solve Problem (2) with corresponding optimal objective value $h(\delta)$ for any $\delta \geq 0$. Since $h(0) > 0$ and $h(\delta)$ is monotone decreasing to 0, one method to find δ^* is Newton's method. Let m denote the number of pairs i, j such that $v_{i,j} \neq 0$. We can rewrite the quadratic functions in the numerator and denominator of $\Pi(x)$ with linear functions by introducing $O(m)$ new binary variables and constraints [19]. Newton's method finds δ^* in $O(m^2 \log^2 m)$ iterations of computing $h(\delta)$ when the feasible region is a subset of $\{0,1\}^{O(m)}$. Hence, for a general synergy graph, we can solve Problem (1) in polynomial time if we can solve Problem (2) in polynomial time.

Unfortunately, Problem (2) is a special case of quadratic binary programming, which is NP-hard in general [13]. We prove that Problem (1) is NP-hard in the next theorem via a reduction from the maximum independent set problem. All proofs are in the online supplement.

Theorem 2. *The assortment optimization problem under synergistic MNL is NP-hard.*

Instead, one way to classify quadratic binary programs and identify cases which are solvable in polynomial-time is to consider the underlying graph [19]. Construct a graph $G = (V, E)$ such that vertex i represents variable x_i and an edge goes from vertex i to j if the coefficient of the quadratic term $x_i x_j$ is non-zero. In Problem (2), the coefficient $(\bar{r}_{i,j} - \delta)v_{i,j}$ changes as we vary δ and we want to consider the quadratic binary program for all possible values of δ, so we add an edge ij whenever $v_{i,j} \neq 0$. Then $V = \{1, \ldots, n\}$ and $E = \{ij : \ v_{i,j} \neq 0\}$, with $n = |V|$ and $m = |E|$. We call this graph the synergy graph (e.g. Figure 1). We write Problems (1) and (2) using graph notation from hereon, unless G is a path so that there is a clear ordering of the products.

If our synergy graph has at least two components, then Problem (2) can be broken up into smaller sub-problems

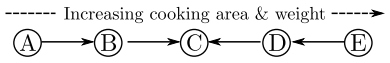

Figure 2: Synergy graph for portable grills which are vertically differentiated - The arrows point in the direction that synergy is created.

containing only the products in the component. Without loss of generality, we assume that our synergy graph is connected. In practice, synergy does not exist between arbitrary pairs of products. We focus on the cases where the synergy graph is a path or a tree and extend our results to consider synergy graphs with low treewidth.

3. Optimal Assortments on Synergy Paths and Trees

We give dynamic programs to solve Problem (2) when the synergy graph is a path, a tree, or has low treewidth.

3.1. A Synergy Path

The synergy path models products that are vertically differentiated. Consider the portable grills in Figure 2, which increase in weight as their cooking areas become larger [22, 24]. A mid-weight, mid-size grill becomes more attractive when it is seen with an extreme alternative, as it offers a compromise between the two features. In the presence of A, B may feel bigger because it is not the smallest option [24]. In the presence of E, D may feel lighter because it is not the heaviest grill. In this example, the presence of A exerts positive synergy on B, and the presence of E exerts positive synergy on D.

Since the synergy graph is a path, we can order the products so that product i creates synergy with products $i-1$ and $i+1$ only. Let $\bar{r}_i = \bar{r}_{i,i+1}$ and $v_i = v_{i,i+1}$ for $i = 1, \ldots, n-1$ when we consider the synergy path. Problem (2) simplifies to:

$$h(\delta) = \max_{x \in \{0,1\}^n} \sum_{i=1}^{n} (r_i - \delta)\, u_i x_i + \sum_{i=1}^{n-1} (\bar{r}_i - \delta)\, v_i x_i x_{i+1}.$$

We use a dynamic program to compute the value of $h(\delta)$. For $i \geq 2$, define the value function $V_i(x_{i-1})$ to be the maximum value that products i to n can contribute to the objective function of Problem (2), given the state x_{i-1} for product $i-1$. At product 1, there is no preceding product whose state we have to consider. We attribute the synergy between products $i-1$ and i to product i, given the decision x_{i-1}. Hence, our value function can be written as the following, with $h(\delta) = V_1$:

$$V_1 = \max_{\substack{x_i \in \{0,1\}: \\ i=1,\ldots,n}} \sum_{i=1}^{n} (r_i - \delta)\, u_i x_i + \sum_{i=1}^{n-1} (\bar{r}_i - \delta)\, v_i x_i x_{i+1},$$

$$V_i(x_{i-1}) = \max_{\substack{x_j \in \{0,1\}: \\ j=i,\ldots,n}} (\bar{r}_{i-1} - \delta) v_{i-1} x_{i-1} x_i + \sum_{j=i}^{n} (r_j - \delta) u_j x_j + \sum_{j=i}^{n-1} (\bar{r}_j - \delta) v_j x_j x_{j+1}.$$

The presence or absence of product $i-1$ does not affect products $i+1$ to n once we have decided whether or not to offer product i. If product i is offered, then its contribution is $R_i^\delta(x_{i-1}) := (\bar{r}_{i-1} - \delta) v_{i-1} x_{i-1} + (r_i - \delta) u_i$. The first term is the synergy between products $i-1$ and i, given the decision of offering product $i-1$. The second term is the base contribution of product i. We can rewrite our value functions using the definition of $R_i^\delta(x_{i-1})$ and $V_{i+1}(x_i)$ to get our dynamic program:

$$V_1 = \max_{x_1 \in \{0,1\}} (r_1 - \delta) x_1 + V_2(x_1), \tag{3}$$

$$V_i(x_{i-1}) = \max_{x_i \in \{0,1\}} R_i^\delta(x_{i-1}) \cdot x_i + V_{i+1}(x_i), \quad \forall i = 2, \ldots, n,$$

$$V_{n+1}(x_n) = 0.$$

Our base cases are $V_{n+1}(0) = V_{n+1}(1) = 0$, and we compute the dynamic program backwards from product n to 1. If we decide $x_i = 0$, then $V_i(x_{i-1}) = V_{i+1}(0)$, and we immediately lose the synergy with both products $i-1$ and $i+1$, as well as the base value $(r_i - \delta)u_i$ from product i. If we decide $x_i = 1$, then $V_i(x_{i-1}) = R^\delta(x_{i-1}) + V_{i+1}(1)$. We get the base value from product i, and we may get the synergy with product $i-1$, depending on the value of x_{i-1}. Furthermore, we have the opportunity to create synergy with product $i+1$ at the next value function.

We conclude with the runtime analysis. For a fixed δ, we can compute $h(\delta)$ in $O(n)$ operations because there are n products, and $O(1)$ states and $O(1)$ decisions at each state per product. Since the synergy graph is a path, we have $m = n-1$, so Newton's method finds δ^* in $O(n^2 \log^2 n)$ iterations of computing $h(\delta)$. The optimal assortment can be computed in $O(n^3 \log^2 n)$ operations. Alternatively, we can find δ^* by solving a linear program in Section 4, which has $O(n)$ variables and constraints. This allows us to compute the optimal assortment with $O(n)$ operations plus the solution of one linear program.

3.2. A Synergy Tree

Suppose the synergy graph is a tree. For example, a generic-brand product creates synergy with the entry-level products of national brands. In turn, the entry-level product of each national brand creates synergy with the higher-end products of its brand [23].

Given a product i, let p_i denote its parent and C_i denote the set of its children. We index the products so that $p_i < i < c$ for $c \in C_i$. If product i is a leaf of the synergy tree, then $C_i = \emptyset$. Let T_i denote the set of products in the subtree rooted at product i. Since each non-root product only has one parent, we let $\bar{r}_i = \bar{r}_{p_i,i}$ and $v_i = v_{p_i,i}$ for cleaner notation. We can rewrite Problem (2) as:

$$h(\delta) = \max_{x \in \{0,1\}^n} \sum_{i \in V} (r_i - \delta)\, u_i x_i + \sum_{i \in V} \sum_{c \in C_i} (\bar{r}_c - \delta)\, v_c x_i x_c.$$

Suppose we are considering whether or not to offer product i, and we already know whether its parent is offered (i.e. x_{p_i}). Then the decision of whether or not to

offer product i is independent of the decisions for the ancestors of p_i, as well as the other children of p_i. We can focus on the subtree rooted at i. When product i is not the root, we define our value function $V_i(x_{p_i})$ as the maximum contribution from products in T_i to the objective function of Problem (2), given our decision on offering product p_i. We attribute the synergy between products p_i and i to product i and the value function $V_i(x_{p_i})$. The value function at the root, V_{root}, has no synergy from a parent and is by definition equal to $h(\delta)$. Our value functions are:

$$V_{\text{root}} = \max_{\substack{x_i \in \{0,1\}: \\ i \in V}} \sum_{i \in V} (r_i - \delta)\, u_i x_i + \sum_{i \in V} \sum_{c \in C_i} (\bar{r}_c - \delta)\, v_c x_i x_c,$$

$$V_i(x_{p_i}) = \max_{\substack{x_j \in \{0,1\}: \\ j \in T_i}} (\bar{r}_i - \delta) v_i x_{p_i} x_i + \sum_{j \in T_i} (r_j - \delta) u_j x_j + \sum_{j \in T_i} \sum_{c \in C_j} (\bar{r}_c - \delta)\, v_c x_j x_c.$$

We apply the strategy from the case of the synergy path. Suppose product i is offered in our assortment and it is not the root. Its contribution to the objective function of Problem (2) is the synergy with its parent given the decision x_{p_i} and its base contribution: $R_i^\delta(x_{p_i}) := (\bar{r}_i - \delta) v_i x_{p_i} + (r_i - \delta) u_i$. If product i is the root, then its contribution is $(r_{\text{root}} - \delta) u_{\text{root}}$. We can rewrite the value functions as a dynamic program, using the definition of $R_i^\delta(x_{p_i})$ and $V_c(x_i)$ for $c \in C_i$:

$$V_{\text{root}} = \max_{x_{\text{root}} \in \{0,1\}} (r_{\text{root}} - \delta) u_{\text{root}} x_{\text{root}} + \sum_{c \in C_{\text{root}}} V_c(x_{\text{root}}),$$

$$V_i(x_{p_i}) = \max_{x_i \in \{0,1\}} R_i^\delta(x_{p_i}) \cdot x_i + \sum_{c \in C_i} V_c(x_i), \quad \forall i \neq \text{root}. \tag{4}$$

We solve the dynamic program starting from the leaves until we reach the root. If product i is a leaf, then $C_i = \emptyset$ and $V_i(x_{p_i}) = \max_{x_i \in \{0,1\}} R_i^\delta(x_{p_i}) \cdot x_i$, so the base case looks similar to the synergy path's base case. If product i is not a leaf, then $x_i = 0$ implies that $V_i(x_{p_i}) = \sum_{c \in C_i} V_c(0)$. This means we lose the base contribution from product i and the synergy with its parent, as well as any chance of synergy with its children. If $x_i = 1$, then we keep the base contribution of product i and possibly synergy with its parent. The decision of whether we create synergy with each child is deferred to the corresponding child.

We need to compute the value function at each vertex of our synergy tree, and there are $O(1)$ states and $O(1)$ decisions at each state per product. Hence, we can solve the dynamic problem in Problem (4) and compute $h(\delta) = V_{\text{root}}$ in $O(n)$ operations for any fixed δ. Similar to the case of the synergy path, we have $m = n - 1$, so Newton's method can find δ^* in $O(n^2 \log^2 n)$ iterations of computing $h(\delta)$. Alternatively, we can find δ^* via a linear program similar to the one in Section 4. The number of operations to compute the optimal assortment when we have a synergy tree or a synergy path is on the same order.

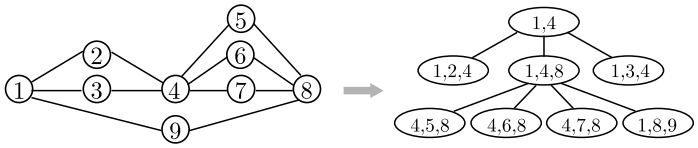

Figure 3: Tree decomposition - On the left, we have a synergy graph on nine products. On the right, we have a tree decomposition of the synergy graph. A vertex on the tree represents a subset of vertices on the original graph. The treewidth is 2.

The dynamic program for the synergy tree suggests that we can use tree decompositions to consider synergy graphs with low treewidth. Tree decompositions are a method to represent a graph with a tree such that each tree vertex represents a subset of vertices on the original graph, and each graph vertex is associated with a subtree on the tree. A graph can have many tree decompositions, and each decomposition is associated with a measure called width. The treewidth of a graph is the minimum width over all of its tree decompositions. Tree-based dynamic programs can be modified into efficient algorithms when the graph has low treewidth [28]. Details of this extension are deferred to Appendix A, and an example of a tree decomposition is depicted in Figure 3.

Theorem 3. *For a synergy graph G, suppose we are given a tree decomposition with width t and $O(n)$ vertices. Then it takes $O(2^{2t}n)$ operations to solve Problem (2). If we use Newton's algorithm to find δ^*, then the total operations needed to compute the optimal assortment is $O(2^{2t}n^5 \log^2 n)$.*

For example, if the synergy graph has no K_4 minor so that it is a series-parallel graph, then it has treewidth $t^* = 2$ [27, 4]. Theorem 3 assumes that a tree decomposition of low width is given. If a graph has treewidth t^*, algorithms exist to find a tree decomposition with width $t = t^*$ and at most $O(n)$ vertices, with runtime polynomial in n but exponential in t^* [2]. On the other hand, there exists algorithms to find a tree decomposition with width $t = O(t^* \log n)$ in polynomial time [3]. This increases the runtime of the dynamic program by a factor of n^2 compared to using a tree decomposition with minimal width.

4. Sales-Based Linear Program for Synergy Path

We revisit our synergy path, and present a linear program that reveals the optimal assortment. This allows a practitioner to take advantage of linear program solvers and avoids solving Problem (2) repeatedly. Proofs and details for this section are deferred to Appendix B.

Suppose the value of δ is fixed. The values of V_1 and $V_i(x_{i-1})$ are not constrained to be integers. If we simply want to compute the value of $h(\delta) = V_1$, then we can transform our dynamic program into a linear program by replacing the decision process at each state with two constraints: the value of $V_i(x_{i-1})$ is greater or equal to the outcomes of both decisions $x_i = 0$ and $x_i = 1$. We transform our

dynamic program in Problem (3) into the following linear program with variables V_1 and V_i^x, where $x \in \{0,1\}$ represents the state x_{i-1} for $i \geq 2$. By construction, the optimal objective value of this linear program is $h(\delta)$.

$$\begin{aligned}
\min\ & V_1 && (5)\\
\text{s.t. } & V_1 \geq V_2^0 \\
& V_1 \geq (r_1 - \delta)u_1 + V_2^1 \\
& V_i^0 \geq V_{i+1}^0 && \forall i = 2,\dots,n \\
& V_i^0 \geq (r_i - \delta)u_i + V_{i+1}^1 && \forall i = 2,\dots,n \\
& V_i^1 \geq V_{i+1}^0 && \forall i = 2,\dots,n \\
& V_i^1 \geq (\bar{r}_{i-1} - \delta)v_{i-1} + (r_i - \delta)u_i + V_{i+1}^1 && \forall i = 2,\dots,n \\
& V_{n+1}^0 = V_{n+1}^1 = 0.
\end{aligned}$$

Let $\mathcal{V}$ denote the set of constraints above. Claim 1 tells us that the linear program has optimal objective value δ^* if and only if we had created this linear program with $\delta = \delta^*$. Furthermore, the constraints in $\mathcal{V}$ are linear in δ, so we can treat δ as a variable. Specifically, we add an extra constraint to the linear program, $\delta = V_1$, to obtain:

$$\min_{V \in \mathcal{V}, \delta \in \mathbb{R}} \{V_1 : \delta = V_1\}. \tag{6}$$

Lemma 4. *The optimal objective value of Problem (6) is equal to the optimal expected profit of Problem (1), δ^*.*

Lemma 4 lets us find δ^* by solving a linear program with $O(n)$ constraints and variables, and immediately proceed to computing $h(\delta^*)$. This avoids iterating over different values of δ in Newton's method and repeatedly solving the dynamic program. These techniques also work for finding δ^* for the synergy tree, and the linear program for a synergy tree also has $O(n)$ constraints and variables.

We take one more step to find the optimal assortment directly via the solution of a linear program when we have a synergy path. Since Problem (6) is a feasible linear program with finite optimal objective value, strong duality states that its dual is feasible and has optimal objective value δ^*. By taking the dual of Problem (6) and making the appropriate transformation of variables, we obtain:

$$\begin{aligned}
\max_{y,w \geq 0}\ & \sum_{i=1}^{n} r_i y_i + \sum_{i=1}^{n-1} \bar{r}_i w_i && (7)\\
\text{s.t. } & y_i/u_i \geq w_i/v_i && \forall i = 1,\dots,n-1 \\
& y_{i+1}/u_{i+1} \geq w_i/v_i && \forall i = 1,\dots,n-1 \\
& y_0 \geq y_i/u_i + y_{i+1}/u_{i+1} - w_i/v_i && \forall i = 1,\dots,n-1 \\
& y_0 + \sum_{i=1}^{n} y_i + \sum_{i=1}^{n-1} w_i = 1.
\end{aligned}$$

Problem (7) looks similar to the SBLP in [10], except for the terms related to synergy: w_i. If we can construct assortments from solutions to Problem (7), then we can interpret y_i as the probability that a customer purchases product i due to its base preference weight, and w_i as the additional probability that a customer purchases either products i or $i+1$ due to the synergy created when both are present. We would like to show that there is a one-to-one correspondence between extreme points of the feasible region in Problem (7) and assortments.

Lemma 5. *If (y,w) is an extreme point of the feasible region in Problem (7), then (y,w) satisfies these conditions:*

1. *If $w_i = 0$, then $y_i = 0$ or $y_{i+1} = 0$.*
2. *If $w_i > 0$, then $y_i/u_i = y_{i+1}/u_{i+1} = w_i/v_i$.*
3. *If $y_i > 0$, then $y_i/u_i = y_0$.*

We show that each assortment maps to a feasible solution of Problem (7).

Lemma 6. *Given an assortment x, there exists a solution (y,w) which is feasible to Problem (7) with objective value equal to $\Pi(x)$.*

We now consider the reverse direction, and show that an extreme point solution (y,w) maps to an assortment.

Lemma 7. *Suppose (y,w) is an extreme point solution to Problem (7). If $x_i = \mathbb{1}[y_i > 0]$, then assortment x has expected profit equal to the objective value of (y,w).*

Theorem 8. *Under synergistic MNL with a synergy path, we can recover the optimal assortment x^* by solving the linear program in Problem (7) for an optimal extreme point solution (y^*,w^*) and setting $x_i^* = \mathbb{1}[y_i^* > 0]$.*

In summary, the linear program allows us to compute the optimal assortment without running the dynamic program for multiple iterations.

5. Extensions

We look at three extensions of our problem. First, we remove the assumption that $v_i^j \geq 0$ for all $i,j \in V$. Second, we incorporate synergy into another choice model: MMNL. Third, we use multiplicative synergy weights so that preference weights change by multiplicative factors.

5.1. Negative Synergy Weights

Suppose product j cannibalizes the demand of product i, to the extent that product j looks more attractive but product i looks less attractive when both products are in the assortment. We can represent this scenario with a negative synergy weight on product i: $v_i^j < 0$. The parametrization techniques in Section 2 are valid if $|\sum_{j \neq i} \min\{v_i^j, 0\}| < u_i$, so that a product ultimately has a positive overall preference weight irrespective of what other products are offered. Note that MNL with a negative overall preference weight does not make sense. To ensure that all notations are well-defined and that the analyses in Sections 3 and 4 continue to hold, if $v_j^i, v_i^j \neq 0$, then we also require that $v_j^i + v_i^j \neq 0$, which is the condition for an edge to exist in our synergy graph. In Problem (7), we would replace $w_i \geq 0$ for all $i = 1,\dots n-1$ with $w_i \geq 0$ if $v_i > 0$ and $w_i \leq 0$ if $v_i < 0$, so that $w_i/v_i \geq 0$.

5.2. Incorporating Synergy into MMNL

Suppose there are K customer types. With probability λ_k, the customer is of type k. A customer of type k has base preference weight $u_i^k \geq 0$ for $i \in V$, and synergy weight from product j on product i of $v_i^{j,k} \geq 0$. In this extension, we assume that $r_i \geq 0$ for all $i \in V$. Then $\bar{r}_{i,j}^k \geq 0$ and $v_{i,j}^k \geq 0$ when we follow the definitions from Section 2 for $k = 1, \ldots, K$. We choose an assortment x to offer to any customer, regardless of her type:

$$\max_{x\in\{0,1\}^n} \sum_{k=1}^{K} \lambda_k \frac{\sum_{i\in V} r_i u_i^k x_i + \sum_{ij\in E} \bar{r}_{i,j}^k v_{i,j}^k x_i x_j}{1 + \sum_{i\in V} u_i^k x_i + \sum_{ij\in E} v_{i,j}^k x_i x_j}. \quad (8)$$

We approach Problem (8) by incorporating the expected profit from customer types 2 to K into the constraints and maximizing the expected profit from type 1 customers. Suppose we are given $(N_2, D_2, \ldots, N_K, D_K) \in \mathbb{R}_+^{2(K-1)}$ and we construct the following problem:

$$\max_{x\in\{0,1\}^n} \frac{\sum_{i\in V} r_i u_i^1 x_i + \sum_{ij\in E} \bar{r}_{i,j}^1 v_{i,j}^1 x_i x_j}{1 + \sum_{i\in V} u_i^1 x_i + \sum_{ij\in E} v_{i,j}^1 x_i x_j} \quad (9)$$

$$\text{s.t.} \sum_{i\in V} r_i u_i^k x_i + \sum_{ij\in E} \bar{r}_{i,j}^k v_{i,j}^k x_i x_j \geq N_k \quad \forall k = 2, \ldots, K$$

$$1 + \sum_{i\in V} u_i^k x_i + \sum_{ij\in E} v_{i,j}^k x_i x_j \leq D_k \quad \forall k = 2, \ldots, K.$$

Any feasible solution to this integer program would ensure that the expected profit from a type k customer is at least N_k/D_k. An assortment x with objective value δ_1 to Problem (9) would guarantee $\Pi(x) \geq \lambda_1\delta_1 + \sum_{k=2}^{K} \lambda_k \frac{N_k}{D_k}$.

Taking the approach of [6], we use a geometric grid with accuracy level ϵ to construct tuples $(N_2, D_2, \ldots, N_K, D_K)$. If we solve Problem (9) for every tuple on our grid, then the assortment with the largest expected profit would guarantee $(1-\epsilon)$-fraction of the optimal expected profit. Instead of solving Problem (9) exactly, we round the coefficients of the constraints in the manner of the knapsack problem to obtain an approximate solution, yielding an FPTAS for a fixed K. We refer to Problem (9) with rounded coefficients as Problem (9'). The details are deferred to Appendix C.

The above discussion for constructing an FPTAS applies for a general synergy graph. The last hurdle is to solve Problem (9') efficiently, and it is unclear how we can achieve this goal even when the synergy graph is a tree. In fact, we can only construct a dynamic program to solve the parametric problem corresponding to Problem (9') when the synergy graph is a path. Hence, we only give a result pertaining to an FPTAS for assortment optimization under synergistic MMNL with a synergy path.

Lemma 9. *Suppose x^* is the optimal assortment under synergistic MMNL, and $\epsilon \in (0,1)$. Let $R = \max_i r_i$ and $U = \max\{1, \max_{k,i} u_i^k, \max_{k,i} v_i^k\}$, and r, u be defined similarly as the minimum parameters. If the synergy graph is a path, then there exists an algorithm that finds an assortment achieving expected profit of $(1-\epsilon)\Pi(x^*)$ in $O\left(\frac{\log^{K-1}(nRU/r)\log^{K-1}((n+1)U/u)\; n^{2K+1}\log^2 n}{\epsilon^{4(K-1)}}\right)$ operations.*

5.3. Multiplicative Synergy Effects

Inspired by [9], we assumed that the synergy weights are additive. If we take the approach that synergy invokes an additive change in the utility of products, then this translates to multiplicative synergy weights. The synergy effect of product j on i is $v_i^j > 0$ and the preference weight of product i is $u_i \cdot \prod_{j\neq i}(v_i^j)^{x_j}$. Note that multiplicative updates easily incorporate a reduction in utility by letting $v_i^j < 1$. To construct our synergy graph, we let $V = N$ and $E = \{ij : v_j^i \neq 1 \text{ or } v_i^j \neq 1\}$. Our problem becomes:

$$\max_{x\in\{0,1\}^n} \frac{\sum_{i\in V} r_i u_i x_i \cdot \prod_{ij\in E}(v_i^j)^{x_j}}{1 + \sum_{i\in V} u_i x_i \cdot \prod_{ij\in E}(v_i^j)^{x_j}}. \quad (10)$$

We cannot consider the marginal increase in the assortment's total preference weight from synergy between products i and j. Hence, we do not combine the terms v_j^i and v_i^j. We focus on synergy paths and trees to show how multiplicative synergy weights affect our algorithms.

Let $h(\delta)$ be the optimal objective value if we parametrize Problem (10) with δ. To understand the difficulty of using multiplicative weights, consider our value function for the synergy path in Subsection 3.1. To simplify notation, let $\overline{v_i} = v_i^{i+1}$ and $\underline{v_i} = v_i^{i-1}$. We cannot consider the marginal effect on product i from $\overline{v_i}$ and $\underline{v_i}$ separately. Instead, we need the values of x_{i-1}, x_i, and x_{i+1} to determine the total synergy weight in the value function. We modify our value function to remember the decisions for two products. Let $V_i(x_{i-1}, x_i)$ be the maximum value that products i to n can contribute to the objective function of the parametric problem, given the decisions of x_{i-1} and x_i. Using the convention that $\underline{v_1} = \overline{v_{n+1}} = 1$ and $x_0 = x_{n+1} = 1$, our value functions are:

$$V_1(x_1) = \max_{\substack{x_i\in\{0,1\}:\\ i=2,\ldots n}} \sum_{i=1}^{n} (r_i - \delta) u_i x_i \cdot (\underline{v_i})^{x_{i-1}} \cdot (\overline{v_i})^{x_{i+1}},$$

$$V_i(x_{i-1}, x_i) = \max_{\substack{x_j\in\{0,1\}:\\ j=i+1,\ldots,n}} \sum_{j=i}^{n} (r_j - \delta) u_j x_j \cdot (\underline{v_j})^{x_{j-1}} \cdot (\overline{v_j})^{x_{j+1}},$$

$$V_n(x_{n-1}, x_n) = (r_n - \delta) u_n x_n \cdot (\underline{v_n})^{x_{n-1}}. \quad (11)$$

Based on the value functions, we have $h(\delta) = \max\{V_1(0), V_1(1)\}$. In Appendix D, we use dynamic programming to compute $V_1(0)$ and $V_1(1)$ efficiently for any value of $\delta > 0$.

Lemma 10. *Suppose we have a synergy path and multiplicative synergy weights. We can compute an optimal assortment in $O(n^3 \log^2 n)$ operations.*

The issue of having to remember the decisions for more than one product carries over to the synergy tree. If there is synergy going from a vertex to its parent and vice versa, then we need the values of x_{p_i}, x_i, and x_c for all $c \in C_i$ in order to determine the total preference weight of product i. We cannot consider the decision of each child vertex independently to measure their marginal contribution to

the preference weight of product i. We present results on two special cases of synergy trees. First, we consider an out-tree, which we define as a synergy tree with $v_i^{p_i} \neq 1$ and $v_{p_i}^i = 1$ for all $i \neq$ root. Returning to our earlier example, a generic-brand product creates synergy towards the entry-level products of national brands, which in turn create synergy for higher-end products in their respective brands. There is no synergy in the reverse direction. Second, we consider a tree that has at most d children per vertex. In this case, each product only creates synergy with a small number of other products.

Lemma 11. *Suppose G is a synergy tree with multiplicative synergy weights. If G is an out-tree, then we can compute an optimal assortment in $O(n^3 \log^2 n)$ operations. If G has at most d children per vertex, then we can compute an optimal assortment in $O(2^{3d} n^3 (d^2 + \log^2 n))$ operations.*

In the above lemma, if $d = O(\log n)$, then our algorithm has runtime polynomial in n, specifically $O(n^6 \log^2 n)$. If $d = O(1)$, then our algorithm computes the optimal assortment in $O(n^3 \log^2 n)$ operations and has the same runtime as synergy trees with additive synergy weights.

6. Conclusion

One future direction is to study parameter estimation and validity of this model. The log-likelihood function of our model is not concave, so parameter estimation would have to rely on local optimal solutions if the maximum likelihood estimator is used. Under this limitation, it would be interesting to obtain real-world data and test whether synergistic MNL performs well in predicting customer purchases and computing near-optimal assortments.

We extended synergy to MMNL, but it would be interesting to see how synergy can be incorporated in other widely-used choice models, such as the nested logit model. Synergy could be created between nests or between products, and each type of synergy could have different interpretations. The incorporation of synergy into other choice models and the construction of algorithms for the underlying assortment optimization problem could lead to choice models that are more useful in applications.

Acknowledgement: We thank Jacob Feldman for his help in developing the proof of Theorem 2.

Funding: This work was supported by NSERC-PGS-D and Cornell University.

References

[1] Blanchet, J., Gallego, G. and Goyal, V. [2016], 'A markov chain approximation to choice modeling', *Oper. Res.* **64**(4), 886–905.

[2] Bodlaender, H. L. [1996], 'A linear-time algorithm for finding tree-decompositions of small treewidth', *SIAM J. on Comput.* **25**(6), 1305–1317.

[3] Bodlaender, H. L., Gilbert, J. R., Hafsteinsson, H. and Kloks, T. [1995], 'Approximating treewidth, pathwidth, frontsize, and shortest elimination tree', *J. Algorithms* **18**(2), 238–255.

[4] Brandstadt, A., Spinrad, J. P. et al. [1999], *Graph classes: a survey*, Vol. 3, SIAM.

[5] Bront, J. J. M., Méndez-Díaz, I. and Vulcano, G. [2009], 'A column generation algorithm for choice-based network revenue management', *Oper. Res.* **57**(3), 769–784.

[6] Désir, A., Goyal, V. and Zhang, J. [2014], 'Near-optimal algorithms for capacity constrained assortment optimization', *Available at SSRN 2543309* .

[7] Feldman, J., Paul, A. and Topaloglu, H. [2019], 'Technical note—assortment optimization with small consideration sets', *Oper. Res.* .

[8] Feldman, J. and Topaloglu, H. [2017], 'Revenue management under the markov chain choice model', *Oper. Res.* **65**(5), 1322–1342.

[9] Feng, G., Li, X. and Wang, Z. [2015], 'Analysis of discrete choice models: A welfare-based framework', *arXiv preprint arXiv:1503.01854* .

[10] Gallego, G., Ratliff, R. and Shebalov, S. [2014], 'A general attraction model and sales-based linear program for network revenue management under customer choice', *Oper. Res.* **63**(1), 212–232.

[11] Hanks, A. S., Just, D. R. and Wansink, B. [2012], 'Trigger foods: The influence of "irrelevant" alternatives in school lunchrooms', *Agric. and Resour. Econ. Rev.* **41**(1), 114–123.

[12] Honhon, D., Jonnalagedda, S. and Pan, X. A. [2012], 'Optimal algorithms for assortment selection under ranking-based consumer choice models', *Manuf. Serv. Oper. Manag.* **14**(2), 279–289.

[13] Kochenberger, G., Hao, J.-K., Glover, F., Lewis, M., Lü, Z., Wang, H. and Wang, Y. [2014], 'The unconstrained binary quadratic programming problem: a survey', *J. Comb. Optim.* **28**(1), 58–81.

[14] Li, H. and Huh, W. T. [2011], 'Pricing multiple products with the multinomial logit and nested logit models: Concavity and implications', *Manuf. Serv. Oper. Manag.* **13**(4), 549–563.

[15] Luce, R. D. [1959], *Individual Choice Behavior a Theoretical Analysis*, John Wiley and sons.

[16] Mahajan, S. and Van Ryzin, G. [2001], 'Stocking retail assortments under dynamic consumer substitution', *Oper. Res.* **49**(3), 334–351.

[17] McFadden, D. and Train, K. [2000], 'Mixed MNL models for discrete response', *J. Appl. Économ.* **15**(5), 447–470.

[18] McFadden, D. et al. [1973], 'Conditional logit analysis of qualitative choice behavior'.

[19] Padberg, M. [1989], 'The boolean quadric polytope: some characteristics, facets and relatives', *Math. Program.* **45**(1), 139–172.

[20] Radzik, T. [1998], Fractional combinatorial optimization, *in* 'Handbook of Combinatorial Optimization', Springer, pp. 429–478.

[21] Rusmevichientong, P., Shmoys, D. B., Tong, C. and Topaloglu, H. [2014], 'Assortment optimization under the multinomial logit model with random choice parameters', *Prod. Oper. Manag.* **23**(11), 2023–2039.

[22] Simonson, I. [1989], 'Choice based on reasons: The case of attraction and compromise effects', *J. Consum. Res.* **16**(2), 158–174.

[23] Simonson, I. [1999], 'The effect of product assortment on buyer preferences', *J. Retail.* **75**(3), 347–370.

[24] Simonson, I. and Tversky, A. [1992], 'Choice in context: Tradeoff contrast and extremeness aversion', *J. Mark. Res.* **29**(3), 281–295.

[25] Thurstone, L. L. [1927], 'A law of comparative judgment.', *Psychol. Rev.* **34**(4), 273.

[26] Train, K. [2003], *Discrete choice methods with simulation*, Cambridge university press.

[27] Wald, J. A. and Colbourn, C. J. [1983], 'Steiner trees, partial 2–trees, and minimum IFI networks', *Networks* **13**(2), 159–167.

[28] Williamson, D. P. and Shmoys, D. B. [2011], *The design of approximation algorithms*, Cambridge university press.

A. Graphs with Low Treewidth

We present a dynamic program to solve the parametric problem, Problem (2), when we are given a tree decomposition of the synergy graph with width t. For a synergy graph $G = (V, E)$, let (X, T) denote a tree decomposition of G, where $T = (V', E')$ is a tree. Furthermore, $X = \{X_1, \ldots, X_K\}$ is a collection of subsets of vertices in V, such that each vertex $k \in V'$ is associated with $X_k \subseteq V$, and X satisfies three properties:

1. Every vertex of G is in some X_k: $\cup_{k \in V'} X_k = V$.
2. For every $ij \in E$, there exists $k \in V'$ such that $i, j \in X_k$.
3. Suppose $m \in V'$ is on the unique k-ℓ path in T. Then $X_k \cap X_\ell \subseteq X_m$.

The width of (X, T) is defined as $t = \max_{k \in V'} |X_k| - 1$, and the treewidth t^* of G is the minimum width of all its tree decompositions. See Figure 3 in Subsection 3.2 for an example of a tree decomposition with width 2.

Select a vertex of V' to be the root of T. Since we work with subsets of vertices, we use set notation rather than vector notation in this appendix. We use p_k to denote the parent of k on T and C'_k to denote the set of children of k. Let T_k denote the set of products which are in some X_ℓ, where ℓ is a vertex in the subtree rooted at k. Recursively, $T_k = X_k$ if k is a leaf and $T_k = (\cup_{c \in C'_k} T_c) \cup X_k$ if k is not a leaf. Notice that $C'_k \subseteq V'$ and $X_k \subseteq T_k \subseteq V$, so they are subsets of vertices on different graphs.

Define $R^\delta(S)$ as the contribution to Problem (2) from the vertices and edges in the subgraph of G induced by products in S. In other words,

$$R^\delta(S) = \sum_{i \in V} (r_i - \delta) u_i \cdot \mathbb{1}[i \in S] + \sum_{ij \in E} (\bar{r}_{i,j} - \delta) v_{i,j} \cdot \mathbb{1}[i, j \in S].$$

Our value function is defined on the tree rather than the original graph. At a vertex $k \in V'$, our value function considers the maximum contribution to Problem (2) that we can achieve by offering products within $S \subseteq T_k$, given that we include $S_{p_k} \subseteq X_{p_k}$ in the assortment. A tree decomposition allows product i to be in X_{p_k} and T_k, so we must ensure that such a product is included in S if and only if it is included in S_{p_k}. If so, we say that $S \subseteq T_k$ is feasible with respect to S_{p_k}. More precisely, given $S_{p_k} \subseteq X_{p_k}$, the feasible subsets of T_k are:

$$F_k^T(S_{p_k}) = \{S \subseteq T_k \ : \ S \cap X_{p_k} = S_{p_k} \cap T_k\}.$$

The value function at $k \in V'$ is:

$$V_k(S_{p_k}) = \max_{S \in F_k^T(S_{p_k})} R^\delta(S).$$

To formulate a dynamic program, we want to restrict our current decision to $S \subseteq X_k$ rather than $S \subseteq T_k$. Similar to the definition of $F_k^T(S_{p_k})$, we want to define the feasible subsets that we may offer from X_k when we are given S_{p_k}. Let $F_k^X(S_{p_k})$ be the collection of feasible subsets that we may offer:

$$F_k^X(S_{p_k}) = \{S \subseteq X_k : \ S \cap X_{p_k} = S_{p_k} \cap X_k\}.$$

The next lemma proves we can compute our value function via dynamic programming using the definition of $F_k^X(S_{p_k})$ and $R^\delta(S)$.

Lemma A.1. *The value function $V_k(S_{p_k})$ can be written as:*

$$V_k(S_{p_k}) = \max_{S_k \in F_k^X(S_{p_k})} R^\delta(S_k) + \sum_{c \in C'_k} \left(V_c(S_k) - R^\delta(S_k \cap X_c)\right).$$

The intuition behind the dynamic program is that the decision made at vertex $\ell \in V'$, where ℓ is in the subtree rooted at k, is feasible with respect to S_{p_k}. This is because a product $i \in X_{p_k} \cap X_\ell$ must also be in X_k, as well as all X_m if m is on the k-ℓ path in the tree. Hence we would make the same decision regarding product i at each tree vertex from k to ℓ. To prove the correctness of the dynamic program, we check that we do not double-count a product or the synergy between products since we may observe a graph vertex or edge in multiple X_ℓ. Furthermore, if $ij \in E$ and $i, j \in T_k$, we check that both i, j are contained in some X_ℓ to ensure that their synergy is recorded. We first use Lemma A.1 to prove the runtime analysis in Theorem 3, and then revisit Lemma A.1 to prove the correctness of the dynamic program.

Proof of Theorem 3. For each vertex on the tree, the state space is the collection of subsets of X_{p_k} and has size $O(2^t)$ since $|X_{p_k}| \leq t - 1$. We consider including $S \in F_k^X(S_{p_k})$ in the assortment, so we consider at most $O(2^t)$ decisions. Our dynamic program is computed from the leaves of T up to the root, and the tree has $O(n)$ vertices. Hence Problem (2) can be solved in $O(2^{2t} n)$ operations.

If we use Newton's algorithm to find the fixed point δ^*, then $m = O(n^2)$ and we need to solve Problem (2) at most $O(n^4 \log^2 n)$ times. Hence the total number of operations to find the optimal assortment is $O(2^{2t} n^5 \log^2 n)$. □

In order to prove Lemma A.1, we use a series of claims to show that the decisions for products in different subtrees T_c are independent when we are given the decision to include $S_k \subseteq X_k$, where c is a child of k. We first show that a product in T_k must either be in X_k or in exactly one T_c. Similarly, a pair of products with positive synergy must either both be in X_k or both be in exactly one T_c.

Claim A.2. *Consider vertex $k \in V'$ and two of its children $c, c' \in C'_k$. Then no products are in both T_c and $T_{c'}$ unless they are also in X_k: $(T_c \backslash X_k) \cap (T_{c'} \backslash X_k) = \emptyset$.*

Proof. Suppose by contradiction that there exists some product i such that $i \in (T_c \backslash X_k) \cap (T_{c'} \backslash X_k)$. Then there exists some $\ell \in V'$ in the subtree rooted at c, such that $i \in X_\ell$. Similarly, there exists $\ell' \in V'$ in the subtree rooted at c' such that $i \in X_{\ell'}$. The path from ℓ to ℓ' contains k, so $X_\ell \cap X_{\ell'} \subseteq X_k$. This implies $i \in X_k$ and contradicts $i \in (T_c \backslash X_k) \cap (T_{c'} \backslash X_k)$. □

Claim A.3. *Suppose there exists an edge $ij \in E$ such that $i, j \in T_k$. Then either $i, j \in X_k$, or at least one of products i, j is not in X_k and there exists an unique $c \in C'_k$ such that $i, j \in T_c$.*

Proof. By definition of tree decomposition, there exists some $\ell \in V'$ such that $i, j \in X_\ell$. We consider three cases: *i)* $\ell = k$, *ii)* $\ell \neq k$ and ℓ is a vertex in the subtree rooted at k, and *iii)* ℓ is not a vertex in the subtree rooted at k. In case i, the first condition is satisfied. In case ii, ℓ must be in the subtree rooted at a vertex $c \in C'_k$, and $i, j \in T_c$. Furthermore, the uniqueness of c follows from observing that at least one of i, j is not in X_k and applying Claim A.2. This satisfies the second condition.

In case iii, there exists vertices ℓ', ℓ'' in the subtree rooted at k, $\ell' \neq \ell''$, such that $i \in X_{\ell'} \backslash X_{\ell''}$ and $j \in X_{\ell''} \backslash X_{\ell'}$ because $i, j \in T_k$. The unique path from ℓ to ℓ' on T contains k, and similarly the path from ℓ to ℓ'' contains k. This implies that $X_\ell \cap X_{\ell'} \subseteq X_k$, so $i \in X_k$. Similarly, $X_\ell \cap X_{\ell''} \subseteq X_k$, so $j \in X_k$. This contradicts our assumption that no vertex ℓ exists in the subtree rooted at k such that $i, j \in X_\ell$. □

The next claim breaks up $R^\delta(S)$ over X_k and T_c for $c \in C'_k$, so that we can count the contributions from X_k and T_c separately.

Claim A.4. *At vertex $k \in V'$, suppose we have $S \subseteq T_k$. Then*

$$R^\delta(S) = R^\delta(S \cap X_k) + \sum_{c \in C'_k} \left(R^\delta(S \cap T_c) - R^\delta(S \cap T_c \cap X_k) \right).$$

Proof. By definition of $R^\delta(S)$, we have

$$R^\delta(S) = \sum_{i \in V} (r_i - \delta) u_i \cdot \mathbb{1}[i \in S] + \sum_{ij \in E} (\bar{r}_{i,j} - \delta) v_{i,j} \cdot \mathbb{1}[i, j \in S].$$

First, we consider the sum over the vertices of G. Since $S \subseteq T_k$, for product $i \in V$, we have,

$$\mathbb{1}[i \in S] = \mathbb{1}[i \in S \cap X_k] + \sum_{c \in C'_k} \left(\mathbb{1}[i \in S \cap T_c] - \mathbb{1}[i \in S \cap T_c \cap X_k] \right). \tag{A.1}$$

If $i \in X_k$, then $\mathbb{1}[i \in S \cap T_c] = \mathbb{1}[i \in S \cap T_c \cap X_k]$, so we count the contribution of product i when it appears in X_k. If $i \notin X_k$, then Claim A.2 tells us that there exists an unique $c \in C'_k$ such that $i \in T_c$.

Next, we consider the sum over the edges of G. Suppose $ij \in E$, then the following is true:

$$\mathbb{1}[i, j \in S] = \mathbb{1}[i, j \in S \cap X_k] + \sum_{c \in C'_k} \left(\mathbb{1}[i, j \in S \cap T_c] - \mathbb{1}[i, j \in S \cap T_c \cap X_k] \right). \tag{A.2}$$

If $i, j \in X_k$, then $\mathbb{1}[i, j \in S \cap T_c] = \mathbb{1}[i, j \in S \cap T_c \cap X_k]$ and we count the synergy between the two products in X_k. Otherwise, Claim A.3 implies that there is an unique c such that $i, j \in T_c$.

We can expand $R^\delta(S)$ and apply equalities (A.1) and (A.2). Equality (A.2) only needs to hold true for $ij \in E$ because $v_{i,j} = 0$ otherwise. Grouping the terms by their indicator functions, we see that $R^\delta(S) = R^\delta(S \cap X_k) + \sum_{c \in C'_k} (R^\delta(S \cap T_c) - R(S \cap T_c \cap X_k))$. □

Finally, we prove that every subset in $F_k^T(S_{p_k})$ can be written as the union of $S_k \in F_k^X(S_{p_k})$ and $S_c \in F_c^T(S_k)$ for $c \in C'_k$, and vice versa.

Claim A.5. *Suppose we are given the decision $S_{p_k} \subseteq X_{p_k}$ and consider a subset of products $S \subseteq F_k^T(S_{p_k})$. Define $S_k = S \cap X_k$ and $S_c = S \cap T_c$ for $c \in C'_k$. Then $S_k \in F_k^X(S_{p_k})$ and $S_c \in F_c^T(S_k)$.*

Proof. Since $S \in F_k^T(S_{p_k})$, we know that

$$S \cap X_{p_k} = S_{p_k} \cap T_k.$$

We intersect both sides with X_k to obtain:

$$S \cap X_{p_k} \cap X_k = S_{p_k} \cap T_k \cap X_k.$$

Since $S_k = S \cap X_k$ and $X_k \subseteq T_k$, we have $S_k \cap X_{p_k} = S_{p_k} \cap X_k$. Hence $S_k \in F_k^X(S_{p_k})$. Finally, to show that $S_c \in F_c^T(S_k)$, observe:

$$S_c \cap X_k = S \cap T_c \cap X_k = S_k \cap T_c.$$

□

Claim A.6. *Given $S_{p_k} \subseteq X_{p_k}$, suppose we have $S_k \in F_k^X(S_{p_k})$ and $S_c \in F_c^T(S_k)$ for all $c \in C'_k$. Let S be the union of these sets such that $S = (\cup_{c \in C'_k} S_c) \cup S_k$, then $S \in F_k^T(S_{p_k})$.*

Proof. We consider:

$$S \cap X_{p_k} = \left(S_k \cap X_{p_k} \right) \cup \left(\cup_{c \in C'_k} \left(S_c \cap X_{p_k} \right) \right). \tag{A.3}$$

First we show that $X_{p_k} \cap T_k = X_{p_k} \cap X_k$. The "$\supseteq$" direction is due to $T_k \supseteq X_k$. For the "$\subseteq$" direction, observe that for every $\ell \in V'$ in the subtree rooted at k, including $\ell = k$, we have $X_{p_k} \cap X_\ell \subseteq X_k$ by the third property of (X, T) being a tree decomposition. Since T_k is the union of all X_ℓ, we have $X_{p_k} \cap T_k \subseteq X_k$. Hence $X_{p_k} \cap T_k = X_{p_k} \cap X_k$, which also implies that $S_{p_k} \cap T_k = S_{p_k} \cap X_k$.

For the first term in the right side of equality (A.3), we use $S_k \in F_k^X(S_{p_k})$ to obtain:

$$\begin{aligned} S_k \cap X_{p_k} &= S_{p_k} \cap X_k \\ &= S_{p_k} \cap T_k. \end{aligned}$$

For the second term, consider some $c \in C'_k$. We have:

$$\begin{aligned} S_c \cap X_{p_k} &= S_c \cap X_{p_k} \cap X_k \\ &= S_k \cap T_c \cap X_{p_k} \\ &= S_{p_k} \cap X_k \cap T_c \\ &\subseteq S_{p_k} \cap T_k. \end{aligned}$$

The first line is true since $S_c \cap X_{p_k} \subseteq T_k \cap X_{p_k} \subseteq X_k$. The second line uses $S_c \in F_c^T(S_k)$, and the third line uses $S_k \in F_k^X(S_{p_k})$. Taking the union of the above terms, we obtain:

$$S \cap X_{p_k} = S_{p_k} \cap T_k,$$

and hence $S \in F_k^T(S_{p_k})$ as required. □

We are now ready to prove Lemma A.1.

Proof of Lemma A.1. We prove the correctness of our dynamic program via induction. In the base case of a leaf, the dynamic program is correct because a leaf vertex does not have children.

At a non-leaf vertex $k \in V'$, assume that the dynamic program holds for all descendants in the subtree rooted at k on T. The following is true by Claim A.4:

$$\begin{aligned} V_k(S_{p_k}) &= \max_{S \in F_k^T(S_{p_k})} R^\delta(S) \\ &= \max_{S \in F_k^T(S_{p_k})} R^\delta(S \cap X_k) \\ &\quad + \sum_{c \in C'_k} (R^\delta(S \cap T_c) - R^\delta(S \cap T_c \cap X_k)). \end{aligned}$$

Claims A.5 and A.6 state that we can choose S_k from $F_k^X(S_{p_k})$ and then S_c from $F_c^T(S_k)$ such that $S = (\cup_{c \in C'_k} S_c) \cup S_k$, rather than S from $F_k^T(S_{p_k})$. This gives us:

$$\begin{aligned} V_k(S_{p_k}) &= \max_{S_k \in F_k^X(S_{p_k})} \max_{\substack{S_c \in F_c^T(S_k): \\ c \in C'_k}} R^\delta(S_k) \\ &\quad + \sum_{c \in C'_k} \left(R^\delta(S_c) - R^\delta(S_c \cap X_k)\right) \\ &= \max_{S_k \in F_k^X(S_{p_k})} R^\delta(S_k) \\ &\quad + \sum_{c \in C'_k} \left(\max_{S_c \in F_c^T(S_k)} R^\delta(S_c) - R^\delta(S_k \cap X_c) \right) \\ &= \max_{S_k \in F_k^X(S_{p_k})} R^\delta(S_k) + \sum_{c \in C'_k} \left(V_c(S_k) - R^\delta(S_k \cap X_c)\right). \end{aligned}$$

The second line is true because $S_c \in F_c^T(S_k)$, so $S_c \cap X_k = S_k \cap X_c$ in the third term. As a result, only the second term is dependent on the inner maximization. We apply the induction hypothesis to get $V_c(S_k) = \max_{S_c \in F_c^T(S_k)} R^\delta(S_c)$. □

B. Constructing the Sales-based Linear Program

We construct the SBLP in Problem (7), and show that the extreme points of its feasible region satisfy the conditions in Lemma 5. As a result, we can compute the optimal assortment simply by solving a linear program, rather than performing multiple iterations of a dynamic program.

Problem (6) expands out to the following linear program when we implicitly set $\delta = V_1$:

$$\begin{aligned} \min \quad & V_1 \\ \text{s.t.} \quad & V_1 \geq V_2^0 \\ & V_1 \geq (r_1 - V_1)u_1 + V_2^1 \\ & V_i^0 \geq V_{i+1}^0 && \forall i = 2, \ldots, n \\ & V_i^0 \geq (r_i - V_1)u_i + V_{i+1}^1 && \forall i = 2, \ldots, n \\ & V_i^1 \geq V_{i+1}^0 && \forall i = 2, \ldots, n \\ & V_i^1 \geq (\bar{r}_{i-1} - V_1)v_{i-1} + (r_i - V_1)u_i + V_{i+1}^1 && \forall i = 2, \ldots, n \\ & V_{n+1}^0 = V_{n+1}^1 = 0. \end{aligned}$$

We first prove Lemma 4, to see that the above linear program has optimal objective value δ^*.

Proof of Lemma 4. Suppose we know the value of δ^* and solved Problem (5) with $\delta = \delta^*$. By Claim 1, Problem (5) has optimal solution V^* with $V_1^* = h(\delta^*) = \delta^*$. Hence (V^*, δ^*) is feasible to Problem (6) with objective value δ^*.

Now suppose by contradiction that Problem (6) has optimal solution $\bar{V}$ with optimal objective value $\bar{V}_1 = \bar{\delta} < \delta^*$. Then $\bar{V}$ is a feasible solution to Problem (5) when $\delta = \bar{\delta}$, with objective value equal to $\bar{\delta}$. But the optimal objective value of Problem (5) at $\delta = \bar{\delta}$ is $h(\bar{\delta})$, which implies that $h(\bar{\delta}) \leq \bar{\delta} < \delta^*$. This contradicts part (*iii*) of Claim 1. □

Our next step is to take the dual of the linear program. We associate the following dual variables with the constraints relating products $i-1$ and i:

- z_i^{00} when we do not offer $i-1$ and i,
- z_i^{01} when we do not offer $i-1$ but offer i,
- z_i^{10} when we offer $i-1$ but do not offer i, and
- z_i^{11} when we offer both $i-1$ and i.

We define the above for $i = 1, \ldots, n$ with $z_1^{10} = z_1^{11} = 0$, because there is no profit nor synergy associated with the

no-purchase option. The dual is:

$$\max \sum_{i=1}^{n} r_i u_i \left(z_i^{01} + z_i^{11}\right) + \sum_{i=1}^{n-1} \bar{r}_i v_i z_{i+1}^{11}$$

s.t.

$$\begin{aligned}
&z_1^{00} + z_1^{01} + \sum_{i=1}^{n} u_i(z_i^{01} + z_i^{11}) + \sum_{i=1}^{n-1} v_i z_{i+1}^{11} = 1 \\
&z_i^{00} - z_{i-1}^{00} + z_i^{01} - z_{i-1}^{10} = 0 && \forall i = 2, \ldots, n \\
&z_i^{10} - z_{i-1}^{01} + z_i^{11} - z_{i-1}^{11} = 0 && \forall i = 2, \ldots, n \\
&z_i^{00}, z_i^{01}, z_i^{10}, z_i^{11} \geq 0 && \forall i = 1, \ldots, n \\
&z_1^{10} = z_1^{11} = 0.
\end{aligned}$$

Lemma B.1. *The dual to Problem (6) is equivalent to Problem (7).*

Proof. Suppose we have a solution z to the dual of Problem (6), then we can construct our solution (y, w) as:

$$\begin{aligned}
&y_0 = z_1^{00} + z_1^{01}, \\
&y_i = u_i(z_i^{01} + z_i^{11}), && \forall i = 1, \ldots, n, \\
&w_i = v_i z_{i+1}^{11}, && \forall i = 1, \ldots, n-1.
\end{aligned}$$

It is simple to verify that (y, w) is feasible to Problem (7) and has the same objective value as z.

In the reverse direction, if we have a solution (y, w) to Problem (7), we can construct a solution z as follows:

$$\begin{aligned}
&z_1^{00} = y_0 - \frac{y_1}{u_1}, \\
&z_i^{00} = y_0 - \frac{y_i}{u_i} - \frac{y_{i-1}}{u_{i-1}} + \frac{w_{i-1}}{v_{i-1}}, && \forall i = 2, \ldots, n, \\
&z_1^{01} = \frac{y_1}{u_1}, \\
&z_i^{01} = \frac{y_i}{u_i} - \frac{w_{i-1}}{v_{i-1}}, && \forall i = 2, \ldots, n, \\
&z_i^{10} = \frac{y_{i-1}}{u_{i-1}} - \frac{w_{i-1}}{v_{i-1}}, && \forall i = 2, \ldots, n, \\
&z_i^{11} = \frac{w_{i-1}}{v_{i-1}}, && \forall i = 2, \ldots, n, \\
&z_1^{10} = z_1^{11} = 0.
\end{aligned}$$

This solution z is feasible to the dual and has the same objective value as (y, w) in Problem (7). □

The key to proving Lemma 5 is to show that every feasible point with some y_i such that $0 < y_i/u_i < y_0$ can be written as a convex combination of two other feasible points, and hence it is not an extreme point.

Lemma B.2. *Every extreme point to the feasible region in Problem (7) satisfies $y_i = 0$ or $y_i/u_i = y_0$ for all $i = 1, \ldots, n$.*

Proof. Suppose we have a solution (y, w) with at least one variable y_i such that $0 < y_i/u_i < y_0$. Define the set of non-tight variables as $J = \{y_i : 0 < y_i/u_i < y_0\} \cup \{w_i : 0 < w_i/v_i < y_0\}$.

Order the variables as $y_1, w_1, y_2, \ldots, y_{n-1}, w_{n-1}, y_n$. Define the set of indices $B = \{b_1, \ldots, b_{L+1}\}$ based on the tightness of the third constraint of Problem (7). Specifically, $b_1 = 0$, $b_{L+1} = n$, $b_1 < b_2 < \cdots < b_{L+1}$, and $i \in B$ if and only if $y_0 = y_i/u_i + y_{i+1}/u_{i+1} - w_i/v_i$.

We partition the variables into $L+1$ sets, $A_0, A_1, \ldots, A_L$. For A_ℓ such that $1 \leq \ell \leq L$, we define:

$$\begin{aligned}
A_\ell = &\{y_i : \ b_\ell < i \leq b_{\ell+1}\} \cup \{w_i : \ b_\ell < i < b_{\ell+1}\} \\
&\cup \{w_i : \ i = b_{\ell+1} \neq n, \ w_i/v_i = y_i/u_i < y_{i+1}/u_{i+1}\} \\
&\cup \{w_i : \ i = b_\ell \neq 0, \ w_i/v_i = y_{i+1}/u_{i+1} < y_i/u_i\}.
\end{aligned}$$

Set A_0 contains the remaining indices of w:

$$\begin{aligned}
A_0 = &\{w_i : \ i \in B, \ w_i/v_i = y_i/u_i = y_{i+1}/u_{i+1}\} \\
&\cup \{w_i : \ i \in B, \ w_i/v_i < \min\{y_i/u_i, y_{i+1}/u_{i+1}\}\}
\end{aligned}$$

To create a new solution, we first define a scaling factor β as:

$$\beta = \sum_{\substack{i:\ y_i \in J \cap A_\ell \\ \ell \text{ odd}}} u_i + \sum_{\substack{i:\ w_i \in J \cap A_\ell \\ \ell \text{ odd}}} v_i - \sum_{\substack{i:\ y_i \in J \cap A_\ell \\ \ell \neq 0 \text{ even}}} u_i - \sum_{\substack{i:\ w_i \in J \cap A_\ell \\ \ell \neq 0 \text{ even}}} v_i.$$

The value of β can any real number. For some small ϵ satisfying $0 < \epsilon < 1/\beta$, the new solution $(\bar{y}, \bar{w})$ is:

$$\bar{y}_i = \begin{cases} \frac{1}{1+\epsilon\beta} y_i & \text{if } y_i \notin J \\ \frac{1}{1+\epsilon\beta}(y_i + \epsilon u_i) & \text{if } y_i \in J \cap A_\ell \text{ for odd } \ell \\ \frac{1}{1+\epsilon\beta}(y_i - \epsilon u_i) & \text{if } y_i \in J \cap A_\ell \text{ for even } \ell \neq 0 \end{cases},$$

$$\bar{w}_i = \begin{cases} \frac{1}{1+\epsilon\beta} w_i & \text{if } w_i \notin J \text{ or } w_i \in A_0 \\ \frac{1}{1+\epsilon\beta}(w_i + \epsilon v_i) & \text{if } w_i \in J \cap A_\ell \text{ for odd } \ell \\ \frac{1}{1+\epsilon\beta}(w_i - \epsilon v_i) & \text{if } w_i \in J \cap A_\ell \text{ for even } \ell \neq 0 \end{cases},$$

$$\bar{y}_0 = y_0/(1 + \epsilon\beta).$$

It is clear that $\bar{y}_0 + \sum_{i=1}^{n} \bar{y}_i + \sum_{i=1}^{n-1} \bar{w}_i = 1$, so we need to check the other constraints.

Case 1 - $\bar{y}_i, \bar{w}_i, \bar{y}_{i+1} \in A_\ell$ for odd ℓ: If $w_i \notin J$, then $\bar{w}_i = w_i/(1 + \epsilon\beta)$ and $\bar{y}_i \geq y_i/(1 + \epsilon\beta)$, so we have $\bar{y}_i/u_i \geq \bar{w}_i/v_i$. If $w_i \in J$ and $y_i \in J$, then

$$\begin{aligned}
\bar{y}_i/u_i &= (y_i/u_i + \epsilon)/(1 + \epsilon\beta) \\
&\geq (w_i/v_i + \epsilon)/(1 + \epsilon\beta) = \bar{w}_i/v_i.
\end{aligned}$$

If $w_i \in J$ and $y_i \notin J$, then $y_i/u_i = y_0$, which implies $y_i/u_i > w_i/v_i$. There exists small ϵ such that:

$$\begin{aligned}
\bar{y}_i/u_i &= y_i/u_i/(1 + \epsilon\beta) \\
&\geq (w_i/v_i + \epsilon)/(1 + \epsilon\beta) = \bar{w}_i/v_i.
\end{aligned}$$

Using the same argument, we can show that $\bar{y}_{i+1}/\bar{u}_{i+1} \geq \bar{w}_i/\bar{v}_i$. Finally, observe:

$$\begin{aligned}
&\frac{\bar{y}_i}{u_i} + \frac{\bar{y}_{i+1}}{u_{i+1}} - \frac{\bar{w}_i}{v_i} \\
\leq &\frac{1}{1 + \epsilon\beta} \left(\frac{y_i + \epsilon u_i}{u_i} + \frac{y_{i+1} + \epsilon u_{i+1}}{u_{i+1}} - \frac{\bar{w}_i}{v_i} \right) \\
\leq &\frac{y_0}{1 + \epsilon\beta} = \bar{y}_0,
\end{aligned}$$

where the last inequality is true for small enough ϵ because the third constraint of Problem (7) is not tight, by definition of B.

Case 2 - $\bar{y}_i, \bar{w}_i, \bar{y}_{i+1} \in A_\ell$ for even $\ell \neq 0$: If $w_i \in J$, then

$$\begin{aligned}\bar{y}_i/u_i &\geq (y_i/u_i - \epsilon)/(1+\epsilon\beta) \\ &\geq (w_i/v_i - \epsilon)/(1+\epsilon\beta) = \bar{w}_i/v_i.\end{aligned}$$

If $w_i \notin J$ and $y_i \notin J$, then $\bar{w}_i = w_i/(1+\epsilon\beta)$ and $\bar{y}_i = y_i/(1+\epsilon\beta)$, so we have $\bar{y}_i/u_i \geq \bar{w}_i/v_i$. If $w_i \notin J$ and $y_i \in J$, we must have $w_i = 0$ and $y_i > 0$, so there exists small ϵ such that

$$\bar{y}_i/u_i = (y_i/u_i - \epsilon)/(1+\epsilon\beta) \geq 0 = \bar{w}_i/v_i.$$

Finally,

$$\begin{aligned}&\frac{\bar{y}_i}{u_i} + \frac{\bar{y}_{i+1}}{u_{i+1}} - \frac{\bar{w}_i}{u_i} \\ \leq &\frac{1}{1+\epsilon\beta}\left(\frac{y_i}{u_i} + \frac{y_{i+1}}{u_{i+1}} - \frac{w_i - \epsilon v_i}{v_i}\right) \\ \leq &\frac{y_0}{1+\epsilon\beta} = \bar{y}_0,\end{aligned}$$

where the last equality again holds for small enough ϵ because the third constraint of Problem (7) is not tight,by definition of B.

Case 3 - $\bar{y}_i \in A_\ell$ and $\bar{y}_{i+1} \in A_{\ell+1}$ for odd ℓ: First suppose $w_i \in A_0$. If $w_i/v_i = y_i/u_i = y_{i+1}/u_{i+1}$, then they are equal to y_0 because the third constraint of Problem (7) is tight when y_i, y_{i+1} are in different sets. Hence they are all not in J and only scaled, so $\bar{w}_i/v_i = \bar{y}_i/u_i = \bar{y}_{i+1}/u_{i+1}$ and all the constraints still hold with equality. Otherwise, $w_i/v_i < \min\{y_i/u_i, y_{i+1}/u_{i+1}\}$, so there is some small ϵ such that $w_i/v_i \leq \min\{y_i/u_i, y_{i+1}/u_{i+1} - \epsilon\}$, which ensures $\bar{w}_i/v_i \leq \min\{\bar{y}_i/u_i, \bar{y}_{i+1}/u_{i+1}\}$. For the third constraint, notice that $y_0 - y_i/u_i = y_{i+1}/u_{i+1} - w_i/v_i > 0$ implies $y_i \in J$. Similarly, $y_{i+1} \in J$. Hence the following equalities hold:

$$\begin{aligned}\bar{y}_0 = \frac{y_0}{1+\epsilon\beta} &= \frac{1}{1+\epsilon\beta}\left(\frac{y_i}{u_i} + \frac{y_{i+1}}{u_{i+1}} - \frac{w_i}{u_i}\right) \\ &= \frac{1}{1+\epsilon\beta}\left(\frac{y_i + \epsilon u_i}{u_i} + \frac{y_{i+1} - \epsilon u_{i+1}}{u_{i+1}} - \frac{w_i}{u_i}\right) \\ &= \frac{\bar{y}_i}{u_i} + \frac{\bar{y}_{i+1}}{u_{i+1}} - \frac{\bar{w}_i}{u_i}.\end{aligned}$$

Next, suppose $w_i \in A_\ell$. Then $y_i/u_i = w_i/v_i < y_{i+1}/u_{i+1}$, so $y_0 = y_{i+1}/u_{i+1}$ by tightness of the third constraint and $y_{i+1} \notin J$. Both w_i and y_i increase in the same direction, so $\bar{w}_i/v_i = \bar{y}_i/u_i$, and there exists small ϵ such that $\bar{w}_i/v_i \leq \bar{y}_{i+1}/u_{i+1}$. For the third constraint, $\bar{y}_i$ and $\bar{w}_i$ negate each other's changes, and $\bar{y}_{i+1}$ is only scaled by $1/(1+\epsilon\beta)$, so equality must still hold.

Finally, suppose $w_i \in A_{\ell+1}$.Then $y_i/u_i > w_i/v_i = y_{i+1}/u_{i+1}$, so $y_0 = y_i/u_i$ by tightness of the third constraint and $y_i \notin J$. Both w_i and y_{i+1} decrease in the same direction, so $\bar{w}_i/v_i = \bar{y}_{i+1}/u_{i+1}$ and are smaller than $\bar{y}_i/u_i$. For the third constraint, $\bar{y}_{i+1}$ and $\bar{w}_i$ negate each other's changes, and $\bar{y}_i$ is only scaled by $1/(1+\epsilon\beta)$, so equality must still hold.

Case 4 - $\bar{y}_i \in A_\ell$ and $\bar{y}_{i+1} \in A_{\ell+1}$ for even $\ell \neq 0$: A similar analysis as case 3 holds.

Hence $(\bar{y}, \bar{w})$ is feasible to Problem (7). We can define a symmetric solution (y', w') as:

$$y'_i = \begin{cases} \frac{1}{1-\epsilon\beta} y_i & \text{if } y_i \notin J \\ \frac{1}{1-\epsilon\beta}(y_i - \epsilon u_i) & \text{if } y_i \in J \cap A_\ell \text{ for odd } \ell \\ \frac{1}{1-\epsilon\beta}(y_i + \epsilon u_i) & \text{if } y_i \in J \cap A_\ell \text{ for even } \ell \neq 0 \end{cases},$$

$$w'_i = \begin{cases} \frac{1}{1-\epsilon\beta} w_i & \text{if } w_i \notin J \text{ or } w_i \in A_0 \\ \frac{1}{1-\epsilon\beta}(w_i - \epsilon v_i) & \text{if } w_i \in J \cap A_\ell \text{ for odd } \ell \\ \frac{1}{1-\epsilon\beta}(w_i + \epsilon v_i) & \text{if } w_i \in J \cap A_\ell \text{ for even } \ell \neq 0 \end{cases},$$

$$\bar{y}_0 = y_0/(1-\epsilon\beta).$$

The same case analysis shows that (y', w') is feasible to Problem (7).

Finally, we have $(y, w) = \lambda(\bar{y}, \bar{w}) + (1-\lambda)(y', w')$ by taking $\lambda = (1+\epsilon\beta)/2$. Hence, (y, w) is not an extreme point, and any extreme point cannot have y_i such that $0 < y_i/u_i < y_0$. □

Using Lemma B.2, we can now prove Lemma 5.

Proof of Lemma 5. The third condition is satisfied by Lemma B.2, so we focus on the first and second conditions. If $y_i = 0$ or $y_{i+1} = 0$, then it is clear that $w_i = 0$. Otherwise, $y_i/u_i = y_{i+1}/u_{i+1} = y_0$. Feasibility to the first three constraints of Problem (7) ensures that $w_i/v_i = y_i/u_i = y_{i+1}/v_{i+1}$. □

Proof of Lemma 6. Construct a solution (y, w) from assortment x as follows:

$$\begin{aligned}y_0 &= \frac{1}{1 + \sum_{j=1}^n u_j x_j + \sum_{j=1}^{n-1} v_j x_j x_{j+1}}, \\ y_i &= \frac{u_i x_i}{1 + \sum_{j=1}^n u_j x_j + \sum_{j=1}^{n-1} v_j x_j x_{j+1}}, \qquad i = 1, \ldots, n, \\ w_i &= \frac{v_i x_i x_{i+1}}{1 + \sum_{j=1}^n u_j x_j + \sum_{j=1}^{n-1} v_j x_j x_{j+1}}, \quad i = 1, \ldots, n-1.\end{aligned}$$

Then (y, w) is feasible to Problem (7) and has objective value equal to $\Pi(x)$. □

Proof of Lemma 7. Since (y, w) is an extreme point, we can rewrite the fourth constraint of Problem (7) as:

$$1 = y_0 + \sum_{i=1}^n y_0 u_i \cdot \mathbb{1}[y_i > 0] + \sum_{i=1}^{n-1} y_0 v_i \cdot \mathbb{1}[w_i > 0].$$

Furthermore, we have:

$$\mathbb{1}[w_i > 0] = \mathbb{1}[y_i > 0] \cdot \mathbb{1}[y_{i+1} > 0].$$

Since $x_i = \mathbb{1}[y_i > 0]$, we can solve for y_0 in terms of x:

$$y_0 = \frac{1}{1 + \sum_{i=1}^{n} u_i x_i + \sum_{i=1}^{n-1} v_i x_i x_{i+1}}.$$

Substituting in the value of y_0, the objective value of (y, w) can be rewritten in terms of x:

$$\begin{aligned}
&\sum_{i=1}^{n} r_i y_i + \sum_{i=1}^{n-1} \bar{r}_i w_i \\
=&\sum_{i=1}^{n} r_i y_0 u_i \cdot \mathbb{1}[y_i > 0] + \sum_{i=1}^{n-1} \bar{r}_i y_0 v_i \cdot \mathbb{1}[w_i > 0] \\
=&\frac{\sum_{i=1}^{n} r_i u_i x_i + \sum_{i=1}^{n-1} \bar{r}_i v_i x_i x_{i+1}}{1 + \sum_{i=1}^{n} u_i x_i + \sum_{i=1}^{n-1} v_i x_i x_{i+1}} = \Pi(x).
\end{aligned}$$

□

Proof of Theorem 8. Suppose (y^*, w^*) is an optimal extreme point to Problem (7), with objective value OPT. By Lemma 6, we know that $\text{OPT} \geq \Pi(x)$ for any assortment x. By Lemma 7, we can construct assortment x^* from (y^*, w^*) such that $\Pi(x^*) = \text{OPT}$. Hence x^* is an optimal assortment. □

C. MMNL with Synergy Path

In this appendix, we require $\bar{r}^k_{i,j} \geq 0$ and $v^k_{i,j} \geq 0$, where $\bar{r}^k_{i,j}$ is the weighted average profit on products i and j for customer type k, and $v^k_{i,j}$ is the total synergy weight between products i and j for customer type k. To satisfy this requirement, we assume that $r_i \geq 0$ for all $i \in V$ and $v^{j,k}_i \geq 0$ for all $i, j \in V$ and $k = 1, \ldots, K$, where $v^{j,k}_i$ is the synergy from product j on product i for customer type k. For cleaner presentation, we present the case with $K = 2$ customer types. The results are easy to extend for $K > 2$ customer types. Let $R = \max_i r_i$ and $U = \max\{1, \max_{i,k} u^k_i, \max_{ij,k} v^k_{i,j}\}$. Similarly, let $r = \min_i r_i$ and $u = \min\{1, \min_{i,k} u^k_i, \min_{ij,k} v^k_{i,j}\}$.

Suppose x^* is the optimal assortment. Let

$$\begin{aligned}
N^* &= \sum_{i \in V} r_i u_i^2 x_i^* + \sum_{ij \in E} \bar{r}^2_{i,j} v^2_{i,j} x_i^* x_j^*, \text{ and} \\
D^* &= 1 + \sum_{i \in V} u_i^2 x_i^* + \sum_{ij \in E} v^2_{i,j} x_i^* x_j^*.
\end{aligned}$$

Observe that if an oracle revealed N^* and D^* and we write Problem (9) with $N_2 = N^*$, $D_2 = D^*$, then Problem (8) has the same optimal solution as Problem (9). Since we do not know the values of N^* and D^*, we construct a geometric grid with accuracy parameter $\epsilon \in (0, 1)$, so that there exists grid points that are close to (N^*, D^*).

By using the problem inputs and accuracy parameter $\epsilon \in (0, 1)$, we define our geometric grid by letting:

$$\begin{aligned}
\mathcal{N} &= \{ru(1+\epsilon)^d : ru(1+\epsilon)^d \leq (1+\epsilon)nRU, d \in \mathbb{Z}_+\} \cup \{0\}, \\
\mathcal{D} &= \{u(1+\epsilon)^d : u(1+\epsilon)^d \leq (1+\epsilon)(n+1)U, d \in \mathbb{Z}_+\} \cup \{1\}.
\end{aligned}$$

Then our grid is $\mathcal{G} = \mathcal{N} \times \mathcal{D}$. We construct Problem (9) for all $(N, D) \in \mathcal{G}$. Let $|G| = |V| + |E|$ be the size of the synergy graph. Instead of solving Problem (9) directly, we apply a strategy from the knapsack problem. We consider a relaxation of Problem (9) where we round the coefficients and constants to integers. Denoting the no-purchase option as product 0 so that $u^2_0 = 1$, let:

$$\begin{aligned}
r'_i &= \left\lfloor \frac{r_i u_i^2}{\epsilon N/|G|} \right\rfloor, && \forall i \in V, \\
r''_{i,j} &= \left\lfloor \frac{\bar{r}^2_{i,j} v^2_{i,j}}{\epsilon N/|G|} \right\rfloor, && \forall ij \in E, \\
u'_i &= \left\lceil \frac{u_i^2}{\epsilon D/(|G|+1)} \right\rceil, && \forall i \in V \cup \{0\}, \\
v'_{i,j} &= \left\lceil \frac{v^2_{i,j}}{\epsilon D/(|G|+1)} \right\rceil, && \forall ij \in E.
\end{aligned}$$

Our bounds on right side become $Y_N = \left\lfloor \frac{|G|}{\epsilon} \right\rfloor - |G|$ and $Y_D = \left\lceil \frac{|G|+1}{\epsilon} \right\rceil + |G| + 1$. We obtain the following relaxation of Problem (9):

$$\begin{aligned}
\max_{x \in \{0,1\}^n} \ & \frac{\sum_{i \in V} r_i u_i^1 x_i + \sum_{ij \in E} \bar{r}^1_{i,j} v^1_{i,j} x_i x_j}{1 + \sum_{i \in V} u_i^1 x_i + \sum_{ij \in E} v^1_{i,j} x_i x_j} \qquad \text{(C.1)} \\
\text{s.t.} \ & \sum_{i \in V} r'_i x_i + \sum_{ij \in E} r''_{i,j} x_i x_j \geq Y_N \\
& u'_0 + \sum_{i \in V} u'_i x_i + \sum_{ij \in E} v'_{i,j} x_i x_j \leq Y_D.
\end{aligned}$$

Lemma C.1. *Given $(N, D) \in \mathcal{G}$, suppose Problem (9) is feasible. Then Problem (C.1) is also feasible. Furthermore, any feasible solution to Problem (C.1) satisfies*

$$\begin{aligned}
&\sum_{i \in V} r_i u_i^2 x_i + \sum_{ij \in E} \bar{r}^2_{i,j} v^2_{i,j} x_i x_j \geq (1 - 2\epsilon)N, \\
&1 + \sum_{i \in V} u_i^2 x_i + \sum_{ij \in E} v^2_{i,j} x_i x_j \leq (1 + 2\epsilon)D.
\end{aligned}$$

Proof. If x is a feasible solution to Problem (9), then:

$$\begin{aligned}
&\sum_{i\in V} r'_i x_i + \sum_{ij\in E} r''_{i,j} x_i x_j \\
&\quad \geq \left(\sum_{i\in V} \frac{r_i u_i^2 x_i}{\epsilon N/|G|} - \sum_{i\in V} x_i\right) + \\
&\quad\quad \left(\sum_{ij\in E} \frac{\bar{r}_{i,j}^2 v_{i,j}^2 x_i x_j}{\epsilon N/|G|} - \sum_{ij\in E} x_i x_j\right) \\
&\quad \geq \frac{N}{\epsilon N/|G|} - |V| - |E| \geq \left\lfloor \frac{|G|}{\epsilon} \right\rfloor - |G|, \text{ and}
\end{aligned}$$

$$\begin{aligned}
&u'_0 + \sum_{i\in V} u'_i x_i + \sum_{ij\in E} v'_{i,j} x_i x_j \\
&\quad \leq \left(\frac{1}{\epsilon D/(|G|+1)} + 1\right) + \left(\sum_{i\in V} \frac{u_i^2 x_i}{\epsilon D/(|G|+1)} + \sum_{i\in V} x_i\right) \\
&\quad\quad + \left(\sum_{ij\in E} \frac{v_{i,j}^2 x_i x_j}{\epsilon D/(|G|+1)} + \sum_{ij\in E} x_i x_j\right) \\
&\quad \leq \frac{D}{\epsilon D/(|G|+1)} + 1 + |V| + |E| \leq \left\lceil \frac{|G|+1}{\epsilon} \right\rceil + |G| + 1.
\end{aligned}$$

Hence x is feasible to Problem (C.1).

In the reverse direction, suppose x is feasible to Problem (C.1). Plugging x into the constraints of Problem (9) results in:

$$\begin{aligned}
&\sum_{i\in V} r_i u_i^2 x_i + \sum_{ij\in E} \bar{r}_{i,j}^2 v_{i,j}^2 x_i x_j \\
&\quad \geq \frac{\epsilon N}{|G|} \cdot \left(\sum_{i\in V} r'_i x_i + \sum_{ij\in E} r''_{i,j} x_i x_j\right) \\
&\quad \geq \frac{\epsilon N}{|G|} \cdot \left(\left\lfloor \frac{|G|}{\epsilon} \right\rfloor - |G|\right) \geq (1-2\epsilon)N, \text{ and}
\end{aligned}$$

$$\begin{aligned}
&1 + \sum_{i\in V} u_i^2 x_i + \sum_{ij\in E} v_{i,j}^2 x_i x_j \\
&\quad \leq \frac{\epsilon D}{|G|+1} \left(u'_0 + \sum_{i\in V} u'_i x_i + \sum_{ij\in E} v'_{i,j} x_i x_j\right) \\
&\quad \leq \frac{\epsilon D}{|G|+1} \cdot \left(\left\lceil \frac{|G|+1}{\epsilon} \right\rceil + |G| + 1\right) \leq (1+2\epsilon)D.
\end{aligned}$$

□

Let $\mathcal{G}_{\text{feas}} \subseteq \mathcal{G}$ be the set of (N, D) for which Problem (C.1) is feasible. We use Lemma C.1 to achieve the following approximation guarantee:

Lemma C.2. *For all $(N, D) \in \mathcal{G}_{\text{feas}}$, let $x^{N,D}$ be the optimal solution to Problem (C.1) at (N, D) with expected profit $\Pi(x^{N,D})$. Then $\max_{(N,D)\in\mathcal{G}_{\text{feas}}} \Pi(x^{N,D}) \geq (1-\epsilon)\Pi(x^*)$.*

Proof. There exists $(N', D') \in \mathcal{G}$ such that $N' \leq N^* \leq (1+\epsilon)N'$ and $D' \geq D^* \geq D'/(1+\epsilon)$. Problem (9) with input (N', D') is a relaxation of the same problem constructed at (N^*, D^*). By Lemma C.1, we have $(N', D') \in \mathcal{G}_{\text{feas}}$. Let x' be the optimal solution to Problem (C.1) with input (N', D'), and let δ' be the optimal objective value. We can lower-bound $\Pi(x')$ by:

$$\begin{aligned}
\Pi(x') &= \sum_{k=1}^{2} \lambda_k \frac{\sum_{i\in V} r_i u_i^k x'_i + \sum_{ij\in E} \bar{r}_{i,j}^k v_{i,j}^k x'_i x'_j}{1 + \sum_{i\in V} u_i^k x'_i + \sum_{ij\in E} v_{i,j}^k x'_i x'_j} \\
&\geq \lambda_1 \delta' + \lambda_2 \frac{(1-2\epsilon)N'}{(1+2\epsilon)D'} \\
&\geq \frac{1-2\epsilon}{1+2\epsilon} \cdot \frac{1}{(1+\epsilon)^2} \cdot \left(\lambda_1 \delta' + \lambda_2 \frac{N^*}{D^*}\right) \\
&\geq (1-6\epsilon) \cdot \sum_{k=1}^{2} \lambda_k \frac{\sum_{i\in V} r_i u_i^k x_i^* + \sum_{ij\in E} \bar{r}_{i,j}^k v_{i,j}^k x_i^* x_j^*}{1 + \sum_{i\in V} u_i^k x_i^* + \sum_{ij\in E} v_{i,j}^k x_i^* x_j^*} \\
&= (1-6\epsilon)\Pi(x^*).
\end{aligned}$$

The first inequality is true by Lemma C.1. The second inequality is due to the choice of N', D'. The last inequality is true because Problem (C.1) is a relaxation of Problem (9) at (N', D'), which in turn is a relaxation of Problem (9) at (N^*, D^*).

Finally, for any accuracy parameter ϵ, we can create the geometric grid and Problem (C.1) with $\epsilon' = \epsilon/6$. Then $\max_{(N,D)\in\mathcal{G}_{\text{feas}}} \Pi(x^{N,D}) \geq \Pi(x') \geq (1-\epsilon)\Pi(x^*)$. □

Until this point, the analysis applies to any general synergy graph. The last challenge is to construct an efficient method to solve the parametric problem corresponding to Problem (C.1). Unfortunately, it is unclear whether we can solve the parametric problem efficiently even for a synergy tree. We are only able to present a dynamic program to solve the parametric problem for a synergy path. After presenting the dynamic program, we explain why our techniques for the synergy path do not work for the synergy tree.

Similar to Subsection 3.1, we use $r''_i = r''_{i,i+1}$ and $v'_i = v'_{i,i+1}$ for $i = 1, \ldots, n-1$. We parametrize Problem (C.1) by δ for the synergy path and denote its optimal objective value by $h(\delta)$:

$$\begin{aligned}
h(\delta) = \max_{x\in\{0,1\}^n} &\sum_{i=1}^{n} (r_i - \delta) u_i^1 x_i + \sum_{i=1}^{n-1} (\bar{r}_i^1 - \delta) v_i^1 x_i x_{i+1} \qquad \text{(C.2)} \\
\text{s.t.} &\sum_{i=1}^{n} r'_i x_i + \sum_{i=1}^{n-1} r''_i x_i x_{i+1} \geq Y_N \\
&u'_0 + \sum_{i=1}^{n} u'_i x_i + \sum_{i=1}^{n-1} v'_i x_i x_{i+1} \leq Y_D.
\end{aligned}$$

For $i \geq 2$, we define the value function $V_i(x_i, y_N, y_D)$ to be the maximum contribution to the objective function of Problem (C.2) from products i to n, subject to y_N being

the lower-bound on the numerator and y_D being the upper-bound on the denominator of the expected profit function from type 2 customers:

$$\begin{aligned} V_i(x_{i-1},y_N,y_D) = &\max_{\substack{x_j\in\{0,1\}:\\ j=i,\dots,n}} (\bar{r}^1_{i-1}-\delta)v^1_{i-1}x_{i-1}x_i \\ &+\sum_{j=i}^{n}(r_j-\delta)u^1_j x_j + \sum_{j=i}^{n-1}(\bar{r}^1_j-\delta)v^1_j x_j x_{j+1} \\ \text{s.t. } & r''_{i-1}x_{i-1}x_i + \sum_{j=i}^{n} r'_j x_j + \sum_{j=i}^{n-1} r''_j x_j x_{j+1} \geq y_N \\ & v'_{i-1}x_{i-1}x_i + \sum_{j=i}^{n} u'_j x_j + \sum_{j=i}^{n-1} v'_j x_j x_{j+1} \leq y_D. \end{aligned}$$

For product 1, we define the value function $V_1(y_N, y_D)$ similarly, but without synergy from a preceding product. Hence, $h(\delta) = V_1(Y_N, Y_D - u'_0)$ where u'_0 is moved to the right side of the second constraint in Problem (C.2) because the no-purchase option is always available.

We compute $V_1(Y_N, Y_D - u'_0)$ via dynamic programming. For cleaner presentation, define:

$$\begin{aligned} R^\delta_i(x_{i-1}) &:= (\bar{r}^1_{i-1}-\delta)v^1_{i-1}x_{i-1} + (r_i-\delta)u^1_i, \\ f_i(x_{i-1}) &:= (r''_{i-1}x_{i-1} + r'_i), \\ g_i(x_{i-1}) &:= (v'_{i-1}x_{i-1} + u'_i). \end{aligned}$$

For $y_N \in \{0,\dots,Y_N\}$ and $y_D \in \{0,\dots,Y_D - u'_0\}$, our dynamic program is as follows:

$$\begin{aligned} V_1(y_N, y_D) = &\max_{x_1\in\{0,1\}} (r_1-\delta)u^1_1 x_1 \\ &+ V_2(x_1,\ y_N - r'_1 x_1,\ y_D - u'_1 x_1), \end{aligned}$$

$$\begin{aligned} V_i(x_{i-1}, y_N, y_D) = &\max_{x_i\in\{0,1\}} R^\delta_i(x_{i-1})\cdot x_i \\ &+ V_{i+1}(x_i,\ y_N - f_i(x_{i-1})\cdot x_i,\ y_D - g_i(x_{i-1})\cdot x_i), \end{aligned}$$

$$V_{n+1}(x_n, y_N, y_D) = \begin{cases} 0 & : y_N \leq 0, y_D \geq 0 \\ -\infty & : \text{otherwise} \end{cases}.$$

To reduce the state space to $y_N \in \{0,\dots,Y_N\}$ and $y_D \in \{0,\dots,Y_D - u'_0\}$, we can set $V_i(x_{i-1}, y_N, y_D) = V_i(x_{i-1}, 0, y_D)$ if $y_N < 0$ and $y_D \geq 0$. Under these conditions, the first constraint of Problem (C.2) is always satisfied and we do not need to track its state. We can also set $V_i(x_{i-1}, y_N, y_D) = -\infty$ if $y_D < 0$. Under this condition, the second constraint of Problem (C.2) is never satisfied. The assumptions that $r_i \geq 0$ and $u^2_i, v^{j,2}_i \geq 0$ are necessary to achieve this reduction in state space. These assumptions ensure that $\bar{r}^2_{i,j}, v^2_{i,j} \geq 0$ so that $r'_i, r''_i \geq 0$ and $u'_i, v'_i \geq 0$, and hence the second and third states are monotone decreasing. We prove the running time of our FPTAS, as stated in Lemma 9.

Proof of Lemma 9. We present the proof for the case of $K = 2$, and discuss how it can be extended for $K > 2$. We construct Problem (C.1) for every $(N, D) \in \mathcal{G}$, and there are $O\left(\frac{\log(nRU/r)\log((n+1)U/u)}{\epsilon^2}\right)$ such points.

To solve Problem (C.1) for a synergy path, we use the dynamic program above to solve Problem (C.2). For a fixed δ, there are n products and $O(n^2/\epsilon^2)$ states per product, with $O(1)$ decisions per state. The dynamic program can be solved in $O(n^3/\epsilon^2)$ operations. Finally, since $m = n-1$, we can find δ^* in $O(n^2 \log^2 n)$ iterations of computing $h(\delta)$ when we use Newton's method. The total runtime is $O\left(\frac{\log(nRU/r)\log((n+1)U/u)n^5\log^2 n}{\epsilon^4}\right)$ operations.

With $K > 2$ types of customers, the grid size increases to $O\left(\frac{\log^{K-1}(nRU/r)\log^{K-1}((n+1)U/u)}{\epsilon^{2(K-1)}}\right)$. We also have to consider the tuple $(N_2, D_2, \dots, N_K, D_K)$, so the state space of the dynamic program has size $O(n\cdot(n/\epsilon)^{2(K-1)})$. □

To conclude this appendix, we discuss why our techniques do not work for a synergy tree. Following Subsection 3.2, let $\bar{r}^1_i = \bar{r}^1_{p_i,i}$, $v^1_i = v^1_{p_i,i}$, $r''_i = r''_{p_i,i}$, and $v'_i = v'_{p_i,i}$ so that we drop the index on the parent. After parametrizing Problem (C.1), we can set up the value function at each vertex. If $i \neq$ root, then:

$$\begin{aligned} V_i(x_{p_i}, y_N, y_D) = &\max_{\substack{x_j\in\{0,1\}:\\ j\in T_i}} (\bar{r}^1_i-\delta)v^1_i x_{p_i}x_i + \\ &\sum_{j\in T_i}(r_j-\delta)u^1_j x_j + \sum_{j\in T_i}\sum_{c\in C_j}\left(\bar{r}^1_c-\delta\right)v^1_c x_j x_c \\ \text{s.t. } & r''_i x_{p_i}x_i + \sum_{j\in T_i} r'_j x_j + \sum_{j\in T_i}\sum_{c\in C_j} r''_c x_j x_c \geq y_N \\ & v'_i x_{p_i}x_i + \sum_{j\in T_i} u'_j x_j + \sum_{j\in T_i}\sum_{c\in C_j} v'_c x_j x_c \leq y_D. \end{aligned}$$

The value function at the root does not consider a parent.

The issue with the synergy tree is that we cannot set up an efficient dynamic program to compute the value functions above. Suppose we combine the dynamic programs for the synergy tree under MNL (Subsection 3.2) and for the synergy path under MMNL. We want to write the value function at vertex i as a sum of product i's contribution and the value functions at its children. The decision space for the dynamic program includes not only the decision of whether or not to offer product i, but also all the possible ways of dividing y_N, y_D over the children of product i. Hence, the decision space becomes extremely large.

D. Multiplicative Synergy Effects

In the case of multiplicative updates, we cannot denote $v_{i,j} = v^i_j + v^j_i$ as the marginal increase in the assortment's total preference weight when both products i and j are offered. Instead, we need to consider the effect of v^i_j and v^j_i separately. For cleaner presentation in the case of synergy paths, let $\overline{v_i} = v^{i+1}_i$ and $\underline{v_i} = v^{i-1}_i$. In the case of synergy

trees, we always use the child vertex as the reference index, so that $\overline{v_i} = v_i^{p_i}$ and $\underline{v_i} = v_{p_i}^i$.

We first discuss the case of synergy paths. Recall the value functions of Problem (11). We can use dynamic programming to compute these value functions, because the presence or absence of product $i-1$ does not affect products $i+1$ to n. We attribute to product i its total contribution from its base preference weight and possible synergy, which implies that we need to know the values of x_{i-1}, x_i, and x_{i+1}. Since we are given the decisions for x_{i-1} and x_i, we only need to decide in advance whether or not to offer product $i+1$. Let $R_i^\delta(x_{i-1}, x_i) = (r_i - \delta)u_i x_i \cdot (\underline{v_i})^{x_{i-1}}$. Our dynamic program for computing the value functions of Problem (11) is:

$$V_1(x_1) = \max_{x_2 \in \{0,1\}} (r_1 - \delta)u_1 x_1 \cdot (\overline{v_1})^{x_2} + V_2(x_1, x_2),$$
$$V_i(x_{i-1}, x_i) = \max_{x_{i+1} \in \{0,1\}} R_i^\delta(x_{i-1}, x_i) \cdot (\overline{v_i})^{x_{i+1}} + V_{i+1}(x_i, x_{i+1}),$$
$$V_n(x_{n-1}, x_n) = R_n^\delta(x_{n-1}, x_n).$$

We have $h(\delta) = \max\{V_1(0), V_1(1)\}$. To determine the number of iterations that we need to compute $h(\delta)$ in order to find δ^*, observe that we can write the numerator and denominator of $\Pi(x)$ as linear functions by introducing binary variables that indicate whether product i is offered under each outcome of (x_{i-1}, x_{i+1}). Define variable y_i^{ab} for $a, b \in [0,1]$ to be 1 if product i is offered, $x_{i-1} = a$, and $x_{i+1} = b$. If $i = 1$, then $y_1^{10} = y_1^{11} = 0$. If $i = n$, then $y_n^{01} = y_n^{11} = 0$. We use the following constraints to obtain a one-to-one mapping between x and y:

$$y_i^{00} + y_i^{01} + y_i^{10} + y_i^{11} \le 1, \qquad \forall i = 1, \dots, n,$$
$$y_i^{01} + y_i^{11} \le y_{i+1}^{00} + y_{i+1}^{01} + y_{i+1}^{10} + y_{i+1}^{11}, \quad \forall i = 1, \dots, n-1,$$
$$y_i^{10} + y_i^{11} \le y_{i-1}^{00} + y_{i-1}^{01} + y_{i-1}^{10} + y_{i-1}^{11}, \quad \forall i = 2, \dots, n,$$
$$y_1^{10} = y_1^{11} = y_n^{01} = y_n^{11} = 0.$$

The first constraint says that we offer product i under at most one outcome of (x_{i-1}, x_{i+1}). The second and third constraints ensure consistency of our decisions for products $i-1, i$, and $i+1$. In the objective function, we replace $u_i x_i \cdot (\underline{v_i})^{x_{i-1}} \cdot (\overline{v_i})^{x_{i+1}}$ with $u_i(y_i^{00} + \underline{v_i} y_i^{10} + \overline{v_i} y_i^{01} + \underline{v_i}\overline{v_i} y_i^{11})$. Then our one-to-one mapping is to take $y_i^{ab} = \mathbb{1}[x_{i-1} = a, x_{i+1} = b] \cdot x_i$ and $x_i = y_i^{00} + y_i^{01} + y_i^{10} + y_i^{11}$. Note that the mapping is only used to demonstrate that the numerator and denominator of $\Pi(x)$ can be written as linear functions. The one-to-one mapping continues to hold for the parametric problem, and we can keep our dynamic program in terms of x.

Proof of Lemma 10. For each value of δ and each vertex, our dynamic program has $O(1)$ states and $O(1)$ decisions per state. Hence, we can compute $h(\delta)$ in $O(n)$ operations. If we use the above transformation to write the numerator and denominator of $\Pi(x)$ as linear functions, then the number of variables is $O(n)$. We compute $h(\delta)$ at most $O(n^2 \log^2 n)$ times to obtain δ^*. Hence, the total number of operations is $O(n^3 \log^2 n)$. □

Next, we move onto the case of a synergy out-tree, where $\underline{v_i} = 1$ for all i. In this case, the total preference weight of product i depends only on the decision of x_i and its parent x_{p_i}. Using the convention that $\overline{v_{\text{root}}} = 1$ and $x_{p(\text{root})} = 1$, the parametric problem is:

$$h(\delta) = \max_{x \in \{0,1\}^n} \sum_{i \in V} (r_i - \delta)u_i x_i \cdot (\overline{v_i})^{x_{p_i}}.$$

The value function for the out-tree is similar to the value function for synergy trees with additive synergy weights. Given the decision x_{p_i}, $V_i(x_{p_i})$ is the maximum contribution from vertices in T_i to the objective function of the parametric problem. We attribute to product i its total contribution: $R_i^\delta(x_{p_i}) = (r_i - \delta)u_i \cdot (\overline{v_i})^{x_{p_i}}$. The dynamic program then decides whether or not to offer product i and can be written as:

$$V_{\text{root}} = \max_{x_{\text{root}} \in \{0,1\}} (r_{\text{root}} - \delta)u_{\text{root}} x_{\text{root}} + \sum_{c \in C_{\text{root}}} V_c(x_{\text{root}}),$$
$$V_i(x_{p_i}) = \max_{x_i \in \{0,1\}} R_i^\delta(x_{p_i}) \cdot x_i + \sum_{c \in C_i} V_c(x_i).$$

For each vertex, this dynamic program has $O(1)$ states and $O(1)$ decisions per state, so the value of $h(\delta)$ can be computed in $O(n)$ operations. We consider the number of iterations that we need to compute $h(\delta)$ to find δ^* later, after we present the dynamic program for a synergy tree with d children.

If we have a synergy tree with at most d children, then the parametric problem is:

$$h(\delta) = \max_{x \in \{0,1\}^n} \sum_{i \in V} (r_i - \delta)u_i x_i \cdot (\overline{v_i})^{x_{p_i}} \cdot \prod_{c \in C_i} (\underline{v_c})^{x_c}.$$

If i is a leaf, then $C_i = \emptyset$ and the product is defined to be 1. Synergy goes in two directions and we need to know the decisions of x_{p_i}, x_i, and x_c for all $c \in C_i$ to determine the preference weight of product i. We cannot consider the marginal contribution from the parent and each child of i separately. We modify the value function for a synergy path with multiplicative weights, so that we know the decisions of x_{p_i} and x_i at the value function for product i. Our value function is:

$$V_{\text{root}}(x_{\text{root}}) = \max_{\substack{x_i \in \{0,1\}: \\ i \in V \setminus \{\text{root}\}}} (r_{\text{root}} - \delta)u_{\text{root}} x_{\text{root}} \cdot \prod_{c \in C_{\text{root}}} (\underline{v_c})^{x_c} + \sum_{i \in V \setminus \{\text{root}\}} (r_i - \delta)u_i x_i \cdot (\overline{v_i})^{x_{p_i}} \cdot \prod_{c \in C_i} (\underline{v_c})^{x_c},$$
$$V_i(x_{p_i}, x_i) = \max_{\substack{x_j \in \{0,1\}: \\ j \in T_i \setminus \{\text{i}\}}} \sum_{j \in T_i} (r_j - \delta)u_j x_j \cdot (\overline{v_j})^{x_{p_j}} \cdot \prod_{c \in C_j} (\underline{v_c})^{x_c}.$$

To formulate our dynamic program, we compute the contribution from product i by deciding which combination

of its d children to include in the assortment. Define $R_i^\delta(x_{p_i}, x_i) = (r_i - \delta)u_i x_i \cdot (\overline{v_i})^{x_{p_i}}$ for $i \neq$ root. Then we have the following dynamic program:

$$V_{\text{root}}(x_{\text{root}}) = \max_{\substack{x_c \in \{0,1\}: \\ c \in C_{\text{root}}}} (r_{\text{root}} - \delta)u_{\text{root}} x_{\text{root}} \cdot \prod_{c \in C_{\text{root}}} (\underline{v_c})^{x_c} + \sum_{c \in C_{\text{root}}} V_c(x_{\text{root}}, x_c),$$

$$V_i(x_{p_i}, x_i) = \max_{\substack{x_c \in \{0,1\}: \\ c \in C_i}} R_i^\delta(x_{p_i}, x_i) \cdot \prod_{c \in C_i} (\underline{v_c})^{x_c} + \sum_{c \in C_i} V_c(x_i, x_c).$$

We compute the value of the parametric problem as $h(\delta) = \max\{V_{\text{root}}(0), V_{\text{root}}(1)\}$. Notice that this dynamic program has $O(1)$ states per vertex but $O(2^d)$ decisions per state. Hence the number of operations to compute $h(\delta)$ is polynomial in n for a fixed d: $O(2^d n)$.

Proof of Lemma 11. We have already stated the runtime for computing $h(\delta)$ using dynamic programming. We need to determine the number of iterations that we compute $h(\delta)$ when we use Newton's method to find δ^*.

For both types of trees, we can rewrite the numerator and denominator of $\Pi(x)$ as linear functions by creating one binary variable for each outcome of x_{p_i} and x_c for all $c \in C_i$. This follows from the argument for a synergy path.

When we have an out-tree, we need a binary variable for each outcome of x_{p_i}, so we need $O(n)$ binary variables. The number of iterations for which we need to compute $h(\delta)$ is $O(n^2 \log^2 n)$, and we can compute the optimal assortment in $O(n^3 \log^2 n)$ operations.

When we have a synergy tree with at most d children, we need $O(2^d n)$ binary variables to represent each tuple of outcomes for the product and its neighbours. The number of iterations for which we need to compute $h(\delta)$ is $O(2^{2d} n^2 (d^2 + \log^2 n))$, and we can compute the optimal assortment in $O(2^{3d} n^3 (d^2 + \log^2 n))$ operations. □

E. Other Proofs

Proof of Claim 1. We only prove the first case because the other two cases are similar. Let x^* be an optimal assortment with expected profit strictly greater than δ:

$$\frac{\sum_{i \in V} r_i u_i x_i^* + \sum_{ij \in E} \bar{r}_{i,j} v_{i,j} x_i^* x_j^*}{1 + \sum_{i \in V} u_i x_i^* + \sum_{ij \in E} v_{i,j} x_i^* x_j^*} > \delta.$$

We can multiply both sides by the denominator because it is strictly positive, and rearrange terms to obtain:

$$\sum_{i \in V} (r_i - \delta) u_i x_i^* + \sum_{ij \in E} (\bar{r}_{i,j} - \delta) v_{i,j} x_i^* x_j^* > \delta.$$

The left side of the above inequality is upper-bounded by $h(\delta)$. □

Proof of Theorem 2. Assortment optimization under the synergistic MNL is in NP: given an assortment x and some K, it is easy to check whether $\Pi(x) \geq K$. We prove that the problem is NP-hard via a reduction from the maximum independent set problem.

Given a graph $G = (V, E)$, we need to determine if there exists a subset of vertices $S \subseteq V$ of size K, such that $ij \notin E$ if $i, j \in S$. Index the vertices by $\{1, \ldots, n\}$. We can create an instance of the synergistic MNL such that the expected profit is greater or equal to $K/(1+K)$ if and only if there exists an independent set of size K.

Create a product i for each vertex $i \in V$. We slightly abuse notation and use the same name for a product and its corresponding vertex. Product i has profit $r_i = 0$ and base preference weight $u_i = 0$. For the synergy terms, let $v_j^i = v_i^j = n/2$ if $ij \in E$ and $v_j^i = v_i^j = 0$ otherwise. Then $\bar{r}_{i,j} = 0$, $v_{i,j} = n$ if $ij \in E$, and $\bar{r}_{i,j} = 0$, $v_{i,j} = 0$ if $ij \notin E$.

Introduce an auxiliary product $n+1$, which has profit $r_{n+1} = 1$ and base preference weight $u_{n+1} = 0$. Product $n+1$ creates synergy with all the other products. Specifically, $v_{n+1}^i = 1$ and $v_i^{n+1} = 0$ for all $i = 1, \ldots, n$, so that the preference weight of product $n+1$ increases by 1 for every additional product offered. Then $\bar{r}_{i,n+1} = 1$ and $v_{i,n+1} = 1$. Since this is the only product with non-zero profit, it must be included in an optimal assortment.

We may disregard $S = \emptyset$ as a single vertex is always an independent set. For a set $S \subseteq V$, create an assortment x where $x_i = \mathbb{1}[i \in S]$ for $i = 1 \leq 1 \leq n$ and $x_{n+1} = 1$. Then the expected profit is:

$$\Pi(x) = \frac{\sum_{i \in V} \bar{r}_{i,n+1} x_i}{1 + \sum_{ij \in E} v_{i,j} x_i x_j + \sum_{i \in V} v_{i,n+1} x_i} = \frac{\sum_{i \in V} x_i}{1 + n \sum_{ij \in E} x_i x_j + \sum_{i \in V} x_i}.$$

First we show that any optimal assortment must correspond to an independent set in G. If $S \neq \emptyset$ is an independent set, then

$$\Pi(x) = \frac{|S|}{1 + |S|} \geq \frac{1}{2}.$$

If S is not an independent set, then there is at least one edge ij where $x_i = x_j = 1$

$$\Pi(x) \leq \frac{|S|}{1 + n + |S|} < \frac{1}{2}.$$

The optimal assortment must be an independent set S along with product $n+1$, with expected profit $|S|/(|S|+1)$. As the expected profit is increasing in the size of the independent set, there exists an independent set of size K if and only if the optimal expected profit is at least $K/(K+1)$. □