TU Delft

Delft University of Technology

Exact and heuristic algorithms for cardinality-constrained assortment optimization problem under the cross-nested logit model

Zhang, Le; Azadeh, Shadi Sharif; Jiang, Hai

DOI
10.1016/j.ejor.2024.12.037

Publication date
2025

Document Version
Final published version

Published in
European Journal of Operational Research

Citation (APA)
Zhang, L., Azadeh, S. S., & Jiang, H. (2025). Exact and heuristic algorithms for cardinality-constrained assortment optimization problem under the cross-nested logit model. *European Journal of Operational Research*, *324*(1), 183-199. https://doi.org/10.1016/j.ejor.2024.12.037

Important note
To cite this publication, please use the final published version (if applicable).
Please check the document version above.

Copyright
Other than for strictly personal use, it is not permitted to download, forward or distribute the text or part of it, without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license such as Creative Commons.

Takedown policy
Please contact us and provide details if you believe this document breaches copyrights.
We will remove access to the work immediately and investigate your claim.

This work is downloaded from Delft University of Technology.
For technical reasons the number of authors shown on this cover page is limited to a maximum of 10.

Green Open Access added to TU Delft Institutional Repository

'You share, we take care!' - Taverne project

https://www.openaccess.nl/en/you-share-we-take-care

Otherwise as indicated in the copyright section: the publisher is the copyright holder of this work and the author uses the Dutch legislation to make this work public.

European Journal of Operational Research 324 (2025) 183–199

Contents lists available at ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/eor

Decision Support

Exact and heuristic algorithms for cardinality-constrained assortment optimization problem under the cross-nested logit model

Le Zhang [a], Shadi Sharif Azadeh [b], Hai Jiang [a,*]

[a] Department of Industrial Engineering, Tsinghua University, Beijing 100084, China
[b] Department of Transport & Planning, Delft University of Technology, Netherlands

ARTICLE INFO

Keywords:
Assortment optimization
Cross-nested logit
Revenue management
Branch-and-Bound
Cardinality constraint

ABSTRACT

We study a class of assortment optimization problems where customers choose products according to the cross-nested logit (CNL) model and the number of products offered in the assortment cannot exceed a fixed number. Currently, no exact method exists for this NP-hard problem that can efficiently solve even small instances (e.g., 50 products with a cardinality limit of 10). In this paper, we propose an exact solution method that addresses this problem by finding the fixed point of a function through binary search. The parameterized problem at each iteration corresponds to a nonlinear binary integer programming problem, which we solve using a tailored Branch-and-Bound algorithm incorporating a novel variable-fixing mechanism, branching rule and upper bound generation strategy. Given that the computation time of the exact method can grow exponentially, we also introduce two polynomial-time heuristic algorithms with different solution strategies to handle larger instances. Numerical results demonstrate that our exact algorithm can optimally solve all test instances with up to 150 products and more than 90% of instances with up to 300 products within a one-hour time limit. Using the exact method as a benchmark, we find that the best-performing heuristic achieves optimal solutions for the majority of test instances, with an average optimality gap of 0.2%.

1. Introduction

Assortment optimization is a significant operational problem faced by numerous retailers and has been extensively studied in the field of revenue management (Feldman & Topaloglu, 2015; Gallego & Topaloglu, 2014; Zhang, Rusmevichientong, & Topaloglu, 2020). Under this problem, the retailers need to determine an optimal subset of potential products for sale subject to various business constraints, such as budget limitations for product procurement and restricted shelf space for product display, with the objective of maximizing expected revenue from customers (Kok, Fisher, & Vaidyanathan, 2008). A pivotal aspect of this problem lies in accurately predicting customers' product choices among the available assortment. Since the work of Van Ryzin and Mahajan (1999), discrete choice models have been widely employed to characterize customer purchasing behavior in assortment optimization, for their capability to accommodate flexible substitution patterns among products. Early researchers focused on the assortment optimization problems under the multinomial logit (MNL) model (McFadden, 1974) and developed efficient algorithms to obtain optimal or high-quality solutions under various side constraints (Rusmevichientong, Shen, & Shmoys, 2010; Talluri & Van Ryzin, 2004). However, the MNL model assumes that the random utility terms of all alternatives are independent, leading to the well-known Independence of Irrelevant Alternatives (IIA) property (Ben-Akiva, Lerman, Lerman, et al., 1985; McFadden, 1974). This implies that including or excluding a product will lead to a proportional shift in the choice probabilities of all other products. Such a property may not hold in many practical scenarios, especially when certain pairs of products exhibit a significant level of substitutability.[1]

To address the unrealistic IIA property, numerous studies opt to employ the nested logit (NL) model (Williams, 1977) in assortment optimization as a viable alternative (Feldman & Topaloglu, 2015; Gallego & Topaloglu, 2014). In the NL model, products are partitioned into disjoint groups, commonly referred to as nests, where items within the same nest are generally close substitutes in a specific aspect. When a

* Corresponding author.
E-mail address: haijiang@tsinghua.edu.cn (H. Jiang).

[1] Consider a scenario in which products A and B exhibit high substitutability, with product A priced lower than product B. When both products are available from the retailer, customers tend to predominantly choose product A due to its lower price. However, if product A becomes unavailable, a significant portion of the initial demand for product A is likely to shift to product B, rather than to other products, due to their substitutable nature.

[2] The value of a nest's dissimilarity parameter reflects the extent of substitutability among the products within the nest. From 1 to 0, a value closer to 0 means higher substitution effects, and vice versa.

https://doi.org/10.1016/j.ejor.2024.12.037
Received 14 February 2024; Accepted 21 December 2024
Available online 31 December 2024
0377-2217/© 2025 Elsevier B.V. All rights are reserved, including those for text and data mining, AI training, and similar technologies.

product becomes unavailable under the NL model, its demand tends to shift more towards other products within the same nest than those in different nests if the dissimilarity parameter of the unavailable product's nest is less than 1,[2] thereby alleviating the IIA property observed in the MNL model. Nevertheless, the NL model has limitations in capturing substitution patterns among products. Consider the example of traffic mode choice in Train (2009), where commuters can select from three modes: car alone (i.e., a car owned and used by an individual commuter), carpooling (i.e., a car shared with several commuters) or public bus. When specifying the nesting structure, we can group the modes "car alone" and "carpooling" together because both involve the use of a private car. It is also reasonable to group "carpooling" and "public bus" together since both allow several commuters to share the ride. However, these two groupings cannot simultaneously exist in the NL model because each product can only be assigned to exactly one nest. In other words, the non-overlapping nesting structure in the NL model limits its ability to capture the correlation between products along multiple dimensions.

A natural extension of the NL model to overcome the limitations above is to incorporate nest overlapping, leading to the cross-nested logit (CNL) model (Vovsha, 1997; Wen & Koppelman, 2001). The CNL model also involves a nesting structure akin to the NL model but allows products to belong to multiple nests with varying degrees of membership. Such flexibility enables the CNL model to capture correlations among products across multiple dimensions simultaneously, thus facilitating a wider range of substitution relationships between products than the NL model. The challenge encountered in the aforementioned traffic mode choice example can be readily addressed by assigning "carpooling" to two nests, each associated with a specific allocation parameter. According to Fosgerau, McFadden, and Bierlaire (2013), the choice probabilities of any random utility model may be approximated by a CNL model.

The flexibility of the CNL model has encouraged scholars to explore assortment optimization problems under more realistic scenarios. For instance, Bernstein and Guo (2023) studied an assortment planning problem within a subscription box service. In this emerging business scenario, customers interested in purchasing a product face a choice between engaging in active search (i.e., visiting physical stores) or subscribing to a box delivery service. Since a product may be available both in stores and in the subscription box, the assortment included in the box influences customers' valuation of their active search choices and, consequently, their subscription decisions. The CNL model can easily capture such interactions by assigning overlapping products to both the "active search" and "subscription box" nests. The degree of membership of product i to nest m, represented by the allocation parameter α_{im}, reflects the extent to which the utility of product i contributes to the attractiveness of channel m. For a given assortment, different allocation parameter values yield varying access probabilities for channels and distinct purchase probabilities for products within each channel. This complexity, however, is challenging to address using the standard MNL or NL models. By implementing the CNL model, the authors theoretically analyzed how the subscription box company manages customer search behavior by adjusting the box contents, ultimately determining the optimal product assortment structure within the box.

Apart from Bernstein and Guo (2023), which is closely tied to specific scenarios and derives managerial insights by analyzing the mathematical properties of the stylized model, to the best of our knowledge, only Le and Mai (2024) have investigated algorithm design for the assortment optimization problem under the CNL model. They proposed non-polynomial time approximation algorithms for the constrained assortment optimization problem under the CNL model. However, the development of non-trivial exact algorithms or efficient heuristics for this problem remains an open question. Our study aims to fill this gap.

1.1. Main contribution

In this study, we develop a non-trivial exact algorithm and efficient heuristics for solving the cardinality-constrained assortment optimization problem when customers choose according to the CNL model (we abbreviate this problem as CAOP-CNL). In the exact solution approach, we initially transform the CAOP-CNL into a fixed point finding problem that is solvable using a binary search framework. In each iteration of the binary search, we need to address an NP-hard parameterized problem which can be formulated as a nonlinear binary integer programming problem. A tailored Branch-and-Bound (B&B) algorithm is developed to obtain the optimal solution for this parameterized problem. The B&B algorithm incorporates a novel variable-fixing mechanism, branching rule and upper bound generation strategy, all capitalizing on the unique structure of the parameterized problem's objective function. Since the computation time of exact algorithm may increase exponentially, we also introduce two polynomial-time heuristic algorithms employing different solution strategies. Numerical results indicate that our exact method can efficiently handle moderately large instances within a reasonable time limit. Specifically, the exact algorithm can optimally solve all test instances with up to 150 products and more than 90% of instances with up to 300 products within a one-hour time limit. Using the exact method as a benchmark, we observe that the top-performing heuristic can attain optimal solutions for the majority of test instances, with a average optimality gap below 0.2%.

We summarize the main contributions of our study as follows:

- To the best of our knowledge, we propose the first non-trivial exact method for solving the cardinality-constraint assortment optimization problem under the CNL model, capable of addressing moderately large instances within a reasonable time limit;
- In designing the exact algorithm, we introduce a novel variable-fixing mechanism, branching rule and upper bound generation strategy to expedite the solving process. Numerical experiments demonstrate the effectiveness of these components across various scenarios;
- We develop efficient heuristic algorithms that can rapidly solve the CAOP-CNL with near-optimal performance. The best-performing heuristic attains optimal solutions for the majority of test instances, with an average optimality gap no larger than 0.2%.

1.2. Literature review

In this subsection, we primarily review the literature on assortment optimization using discrete choice models to characterize consumer choice behavior, with a particular emphasis on those employing models from the generalized extreme value (GEV) family (McFadden, 1978). When discussing the CNL model, we include a concise review of the literature that focuses on its empirical applications. Additionally, we discuss several studies that also develop exact solution approaches for addressing complex assortment optimization problems.

Given the simplicity and practicality of the MNL model, assortment optimization problems associated with it are among the first to be studied by scholars (Van Ryzin & Mahajan, 1999). The unconstrained and cardinality-constrained versions of these problems are shown to be polynomially solvable by Rusmevichientong et al. (2010) and Talluri and Van Ryzin (2004), respectively. Addressing uncertainty in the parameters of the logit model, Rusmevichientong and Topaloglu (2012) explore robust assortment optimization under the MNL model. To account for the heterogeneity in consumer choice behavior, several studies have also investigated assortment optimization under the latent class logit model and the mixed logit model (Bront, Méndez-Díaz, & Vulcano, 2009; Méndez-Díaz, Miranda-Bront, Vulcano, & Zabala, 2014; Rusmevichientong, Shmoys, Tong, & Topaloglu, 2014).

To address the limitations of the IIA property inherent in the MNL model, numerous studies have explored assortment optimization problems under the NL model. Within the existing literature, two prevalent specifications of the NL model are observed: the "standard form" and the "general form". In the standard form, the dissimilarity parameters associated with each nest must fall within the range of $(0, 1]$, and there is a single no-purchase option that forms its own nest. In the general form, the dissimilarity parameters are permitted to exceed 1, and each nest may include a no-purchase option.

The standard form of the NL model has been widely adopted in previous studies. Li and Rusmevichientong (2014) propose a greedy algorithm capable of solving the unconstrained problem in polynomial time. For the cardinality-constrained version, Feldman and Topaloglu (2015) and Gallego and Topaloglu (2014) demonstrate that the problem can be optimally solved in polynomial time when cardinality constraints are applied either to each nest or to the overall assortment. However, when the problem involves space constraints – where each product occupies a certain amount of space, and the total or nest-specific space is limited – it becomes NP-hard. Several studies have proposed polynomial-time approximation algorithms for this problem (Chen & Jiang, 2020; Feldman & Topaloglu, 2015; Gallego & Topaloglu, 2014; Rusmevichientong, Shen, & Shmoys, 2009).

Davis, Gallego, and Topaloglu (2014) is the first to investigate assortment optimization problems under the general form of the NL model. They demonstrate that even the unconstrained assortment optimization problem under a non-standard NL model is NP-hard. In recent years, there has been an increasing focus on assortment optimization under general NL models. For instance, Alfandari, Hassanzadeh, and Ljubić (2021), a study closely related to ours, develop a tailored Branch-and-Bound algorithm to optimally solve the unconstrained assortment optimization problem under general NL models. Additionally, Kunnumkal (2023) and Segev (2022) design approximation algorithms for the cardinality-constrained and space-constrained versions, respectively.

By reviewing studies on the assortment optimization problem under the NL model (AOP-NL), we note a significant advantage in solving the AOP-NL: the problem can be decomposed by nest after a specific transformation or approximation. This property renders the sub-problem more manageable and allows for an approximate solution using classical approaches like dynamic programming (Chen & Jiang, 2020) or others.

As previously discussed, the NL model falls short in simultaneously capturing correlations among products across multiple dimensions. To address these limitations in the context of assortment optimization, Ghuge, Kwon, Nagarajan, and Sharma (2022) and Zhang et al. (2020) employ the paired combinatorial logit (PCL) model to characterize customer choice behavior. The authors establish the NP-hardness of the assortment optimization problem under the PCL model (AOP-PCL), even in the absence of constraints. They propose polynomial-time approximation algorithms for the AOP-PCL under various side constraints.

In the PCL model, each pair of products forms a nest with its own dissimilarity parameter, allowing pairs of products to exhibit varying degrees of substitutability. However, as noted by Le and Mai (2024), in a PCL model with n products, allocating a product uniformly to $n-1$ nests restricts its correlation with others to $1/(n-1)$, thereby constraining the magnitude of cross-elasticities among product pairs. In contrast, the CNL model permits assigning varying proportions of each product to multiple nests, offering greater flexibility in modeling cross-elasticities and the covariance structure (Marzano, Papola, Simonelli, & Vitillo, 2013). Nevertheless, the fixed nesting structure of the PCL model has its advantages. It enables the linearization of the AOP-PCL formulation by introducing a set of new variables related to each product pair, thereby facilitating the development of approximation algorithms.

The CNL model is among the most recently introduced choice models within the GEV family (Small, 1987; Vovsha, 1997), and it encompasses various formulations, including the generalized nested logit (GNL) model as discussed in Train (2009) and Wen and Koppelman (2001). The specification of the CNL model used in this paper follows the GNL model outlined in Train (2009). Ever since Vovsha (1997), the CNL model has been extensively applied in the field of transportation demand modeling, including travel mode choice (Ermagun & Levinson, 2017; Fan, Ding, Long, & Wu, 2024; Huan, Hess, Yamamoto, & Yao, 2024), route choice (Lai & Bierlaire, 2015; Yang & Wang, 2017), departure time choice (Lemp, Kockelman, & Damien, 2010), and airline product choice (Drabas & Wu, 2013; Zhang, Duan, & Jiang, 2024). Given its capability to deal with multi-dimensional choice, the CNL model is also widely employed to analyze joint travel mode and departure time choice (Ding, Mishra, Lin, & Xie, 2015), joint mode and transit route choice (Mepparambath, Soh, Jayaraman, Tan, & Ramli, 2023; Zhang, Yao, & Pan, 2019), joint location and travel mode choice (Vega & Reynolds-Feighan, 2009), as well as joint choice of residential location, travel mode, and departure time (Yang, Zheng, & Zhu, 2013). Moreover, the CNL model is also adopted in topics gaining emerging attention, like alternative fuel vehicles demand analysis (Domarchi & Cherchi, 2024; Hess, Fowler, Adler, & Bahreinian, 2012), residence planning under climate change (Lu, Zhang, Wu, & Rahman, 2016), and location choice in international migration (Beine, Bierlaire, & Docquier, 2021).

As mentioned in the previous section, only Le and Mai (2024) have investigated algorithm design for the assortment optimization problem under the CNL model (AOP-CNL). They propose non-polynomial time approximation algorithms that can derive solutions to the AOP-CNL with a performance guarantee of $\frac{1-\epsilon}{1+\epsilon}$ for any accuracy level $\epsilon > 0$. However, a smaller value of ϵ would lead to a longer computation time. Currently, there is a lack of non-trivial exact solution methods and efficient heuristic algorithms for solving the AOP-CNL, and our study aims to address this gap.

In addition to Alfandari et al. (2021) mentioned above, several studies have focused on developing exact methods to solve challenging assortment optimization problems. For instance, Chung, Ahn, and Jasin (2019) investigate the assortment optimization problem under a re-scaled multi-attempt model. They formulate this problem as a mixed-integer linear fractional program (MILFP) and apply the Dinkelbach algorithm (Dinkelbach, 1967) to solve it exactly. Although their solution framework is similar to that of Alfandari et al. (2021) and our own, the inner problem in our study is more complex due to the non-linearity and nest overlapping effects in the objective function. Bertsimas and Mišić (2019) address the assortment optimization problem under a ranking-based choice model. They propose a novel MILP formulation whose inherent structure allows the authors to effectively apply Benders decomposition to achieve optimal solutions at a large scale. Akchen and Mišić (2023) adopt a similar approach to Bertsimas and Mišić (2019) by replacing the ranking-based choice model with a more generalized decision forest model, which is claimed to be capable of representing any discrete choice model (Chen & Mišić, 2022). Furthermore, Chen, He, Rong, and Wang (2024) introduce a quick-commerce assortment optimization problem (QAP), where retailers must determine assortments for both offline and online consumer segments. Each consumer segment follows a distinct MNL model, and the personalized online assortment is constrained by the offline assortment due to the need for prompt delivery. The authors identify favorable properties of the convex hull of the online choice probability sets and develop a cutting-plane approach for solving their formulation to provable optimality.

1.3. Organization

The remainder of this paper is structured as follows: Section 2 formally defines the assortment optimization problem addressed in this

study. We then describe the design of our exact method in Section 3 and heuristic algorithms in Section 4. The numerical results of both exact and heuristic algorithms, applied to synthetic datasets, are presented in Section 5. Finally, Section 6 concludes the study and outlines potential future directions.

2. Problem definition

2.1. Formulation

In this section, we formulate the assortment optimization problem when customers' choice is modeled by the CNL model and a total cardinality constraint is imposed on the assortment. There is a set of potential products $N = \{1, 2, \dots, n\}$ that we can offer to the customers. For each product j, let r_j and v_j denote its revenue and preference weight, where $r_j, v_j > 0, \forall j \in N$. Without loss of generality, we assume that $r_1 \geq r_2 \geq \cdots \geq r_n > 0$. The preference weight of no-purchase option is $v_0 > 0$. We also define a collection of nests $M = \{1, 2, \dots, m\}$, where each nest represents a grouping of alternatives that share similarity along a certain dimension. Under the CNL model, each product can be assigned to multiple nests in order to capture the products' similarities across various dimensions. Hence, for every pair of a product $j \in N$ and a nest $i \in M$, an allocation parameter α_{ij} is introduced to quantify the extent of product j belonging to nest i, and a value of zero indicates that the product has no membership in that particular nest. For normalization, there are two additional conditions for the allocation parameters, namely $0 \leq \alpha_{ij} \leq 1, \forall i, j$, and $\sum_{i=1}^{m} \alpha_{ij} = 1, \forall j$. Moreover, each nest i is associated with a dissimilarity parameter γ_i, representing the degree of dissimilarity among products within that nest. A lower value of γ_i reflects a higher level of similarity. We assume that the value of each γ_i lies in $(0, 1]$ to ensure the CNL model always consistent with the random utility maximization framework (Bierlaire, 2006), regardless of the preference weights' values.

Under the CNL model, the choice process of an arriving customer can be conceptualized as occurring in two stages. At the first stage, the customer decides whether to make a purchase within one of the nests or to leave without purchasing anything. If the decision is to purchase in a specific nest, the process advances to the second stage. At this stage, the customer must select one of the products available within the chosen nest and is not permitted to exit the nest without making a selection. This process is similar to that under the NL model (Feldman & Topaloglu, 2015; Gallego & Topaloglu, 2014), with a crucial distinction being that a product can belong to multiple nests in the CNL model. More specifically, under the NL model, a product can be purchased at the second stage only if its corresponding nest is selected at the first stage. However, under the CNL model, even if a customer selects different nests during the first stage, she may still end up purchasing the same product at the second stage, provided that this product belongs to each selected nest to a certain extent larger than 0. This characteristic also renders the assortment optimization problem under the CNL model more complex to solve compared to the NL model, as will be explicitly discussed in the subsequent sections.

Let $\boldsymbol{x} = (x_1, x_2, \dots, x_n) \in \{0, 1\}^n$ be a vector of n binary variables representing an assortment decision where $x_j = 1$ if and only if product j is provided. As discussed in the literature review, this paper specifies the CNL model according to the formulation of the GNL model as outlined in Train (2009). Therefore, the probability of a customer selecting nest i at the first stage can be expressed as $P_i(\boldsymbol{x}) = V_i(\boldsymbol{x})^{\gamma_i} / (v_0 + \sum_{l \in M} V_l(\boldsymbol{x})^{\gamma_l})$, where $V_i(\boldsymbol{x}) = \sum_{j \in N} (\alpha_{ij} v_j)^{1/\gamma_i} x_j$ denotes the total preference weight of nest i under assortment $\boldsymbol{x}$. If the customer decides to choose nest i, the conditional probability of choosing product j at the second stage is given by $P_{j|i}(\boldsymbol{x}) = (\alpha_{ij} v_j)^{1/\gamma_i} x_j / V_i(\boldsymbol{x})$. For ease of presentation, we introduce an auxiliary term "nest-specific preference weight" v_{ij} to measure the contribution of product j to the total preference weight of nest i. Formally, this weight is defined as $v_{ij} = (\alpha_{ij} v_j)^{1/\gamma_i} \geq 0$. A similar notation can also be seen in Bernstein and Guo (2023). Utilizing this term, we can simplify several formulae mentioned above. For instance, $V_i(\boldsymbol{x})$ and $P_{j|i}(\boldsymbol{x})$ can be rewritten as $\sum_{j \in N} v_{ij} x_j$ and $v_{ij} x_j / V_i(\boldsymbol{x})$, respectively. Recall that under the specific variation of the CNL model studied in the manuscript, customers are not allowed to leave the nest without a purchase, thus we have $v_{i0} = 0, \forall i \in M$. The probability function of a customer choosing product j under the CNL model can then be derived as

$$P_j(\boldsymbol{x}) = \sum_{i \in M} P_i(\boldsymbol{x}) P_{j|i}(\boldsymbol{x}) = \sum_{i \in M} \left(\frac{V_i(\boldsymbol{x})^{\gamma_i}}{v_0 + \sum_{l \in M} V_l(\boldsymbol{x})^{\gamma_l}} \cdot \frac{v_{ij} x_j}{V_i(\boldsymbol{x})} \right). \tag{1}$$

Denote $R_i(\boldsymbol{x}) = (\sum_{j \in N} r_j v_{ij} x_j) / V_i(\boldsymbol{x})$ as the expected revenue if the customer has decided to choose nest i at the first stage. The expected revenue per customer can be expressed as

$$R(\boldsymbol{x}) = \sum_{j \in N} r_j P_j(\boldsymbol{x}) = \frac{\sum_{i \in M} V_i(\boldsymbol{x})^{\gamma_i} R_i(\boldsymbol{x})}{v_0 + \sum_{l \in M} V_l(\boldsymbol{x})^{\gamma_l}}. \tag{2}$$

We observe that if $V_i(\boldsymbol{x}) = 0$, the expressions for $P_{j|i}(\boldsymbol{x})$ and $R_i(\boldsymbol{x})$ may result in an indeterminate form of $0/0$. However, this does not present a challenge. If an assortment yields a total preference weight of zero for nest i, then customers would never choose nest i during the first stage, rendering the values of $P_{j|i}(\boldsymbol{x})$ and $R_i(\boldsymbol{x})$ inconsequential. Moreover, it is worth mentioning that our treatment of the no-purchase option in the CNL model diverges from that in Le and Mai (2024). In our model, there exists only one no-purchase option, which forms a nest on its own. In contrast, Le and Mai (2024) postulate that each nest contains a respective no-purchase option, with no standalone no-purchase option forming its own nest. Such differences may influence the feasibility of directly implementing their approach under the problem in our specification, and vice versa. Our treatment aligns with many of previous works on assortment optimization based on the NL model (Feldman & Topaloglu, 2015; Gallego & Topaloglu, 2014; Rusmevichientong et al., 2009), as well as the PCL model (Ghuge et al., 2022; Zhang et al., 2020).

The objective is to identify an assortment of products that maximizes the expected revenue per customer, subject to a total cardinality constraint that limits the number of products offered in the assortment. This constraint is prevalent in retailing context. For instance, when customers browse an e-commerce website, they typically focus only on the products displayed at the top of the page. Thus, only a limited number of products can be effectively showcased to customers. Suppose that the maximum cardinality is set as c, then we can formally model the cardinality-constrained assortment optimization problem under the Cross-Nested Logit model (CAOP-CNL) as

$$z^* = \max_{\boldsymbol{x} \in \{0,1\}^n : \|\boldsymbol{x}\|_1 \leq c} R(\boldsymbol{x}) = \max_{\boldsymbol{x} \in \{0,1\}^n : \|\boldsymbol{x}\|_1 \leq c} \frac{\sum_{i \in M} V_i(\boldsymbol{x})^{\gamma_i} R_i(\boldsymbol{x})}{v_0 + \sum_{l \in M} V_l(\boldsymbol{x})^{\gamma_l}}, \tag{3}$$

where $\|\boldsymbol{x}\|_1$ denotes the L1-norm of vector $\boldsymbol{x}$. Given that all elements of $\boldsymbol{x}$ are binary variables, $\|\boldsymbol{x}\|_1$ also represents the number of elements equal to 1 in vector $\boldsymbol{x}$. This notation is useful for expressing the cardinality constraints. The optimal solution $\boldsymbol{x}^*$ corresponds to the assortment that yields the maximum expected revenue, denoted by z^*.

2.2. Complexity analysis

The assortment optimization problem under the Paired Combinatorial Logit (PCL) model has been proved to be strongly NP-hard in Zhang et al. (2020), even in its uncapacitated version. Therefore, it is straightforward to infer that the uncapacitated assortment optimization problem under the CNL model is also NP-hard, since any PCL model can also be regarded as a CNL model. In short, the formulation of PCL model in Zhang et al. (2020) is equivalent with ours of CNL model if we regard the preference weight v_j in Zhang et al. (2020) as a multiplication of α and the preference weight v_j in our CNL model, where $\alpha = \frac{1}{2(n-1)}$ is the value of all non-zero allocation parameters associated with each product. Specifically, for each pair of a product $j \in N$ and a nest (i, j) where $i \in N, i \neq j$, the corresponding allocation parameter $\alpha_{j,(i,j)} = \frac{1}{2(n-1)}$. Moreover, we have $\alpha_{j,(j,i)} = \frac{1}{2(n-1)}, \forall i \in N, i \neq j$. Since the uncapacitated assortment optimization problem is a special case of cardinality-constrained version when $c \geq n$, the CAOP-CNL is NP-hard.

3. Exact solution algorithm

To the best of our knowledge, other than fully enumerating all possible assortments, there is currently no exact method for solving the CAOP-CNL optimally. Therefore, we first aim to develop a non-trivial exact solution algorithm for this problem. This exact method can also serve as a benchmark to evaluate the performance of heuristics introduced in the subsequent section.

3.1. Solution framework

In accordance with several prior studies (Alfandari et al., 2021; Feldman & Topaloglu, 2015; Gallego & Topaloglu, 2014), we convert the CAOP-CNL into a problem of finding the fixed point of a function. Define $F(z) = \max_{\boldsymbol{x}\in\{0,1\}^n:\|\boldsymbol{x}\|_1\le c}\{\sum_{i\in M} V_i(\boldsymbol{x})^{\gamma_i}[R_i(\boldsymbol{x}) - z]\}$ as a function of $z \in R^+$, we establish the following lemma.

Lemma 1. *If we let z^* be the value of z that satisfies the following fixed point problem:*

$$v_0 z = F(z) = \max_{\boldsymbol{x}\in\{0,1\}^n:\|\boldsymbol{x}\|_1\le c}\left\{\sum_{i\in M} V_i(\boldsymbol{x})^{\gamma_i}[R_i(\boldsymbol{x}) - z]\right\}, \tag{4}$$

then z^* corresponds to the optimal expected revenue in problem (3). And the optimal assortment $\boldsymbol{x}^*$ of problem (3) can be found by solving the following maximization problem:

$$F(z^*) = \max_{\boldsymbol{x}\in\{0,1\}^n:\|\boldsymbol{x}\|_1\le c}\left\{\sum_{i\in M} V_i(\boldsymbol{x})^{\gamma_i}[R_i(\boldsymbol{x}) - z^*]\right\}. \tag{5}$$

To save space, we omit the proof of this lemma, which can be found in Lemma 1 of Gallego and Topaloglu (2014). Since $v_0 z$ is a strictly increasing function in z, and $F(z)$ is a continuous decreasing function of z,[3] as long as we can determine the value and corresponding solution of $F(z)$ for any $z \in R^+$, it becomes feasible to solve problem (4) utilizing a binary search framework proposed by Alfandari et al. (2021), as delineated in Algorithm 2.

Within the binary search framework, we have to first compute the initial upper and lower bounds of the optimal expected revenue through two algorithms Z_UPPER and Z_LOWER, respectively. An intuitive version of Z_UPPER is shown in Algorithm 1. It is a two-step relaxation: initially setting the price of all products to r_1, then establishing an upper bound of purchasing probability. That is, $z^* = \frac{\sum_{i\in M} V_i(\boldsymbol{x}^*)^{\gamma_i} R_i(\boldsymbol{x}^*)}{v_0+\sum_{l\in M} V_l(\boldsymbol{x}^*)^{\gamma_l}} \le r_1 \cdot \frac{\sum_{i\in M} V_i(\boldsymbol{x}^*)^{\gamma_i}}{v_0+\sum_{l\in M} V_l(\boldsymbol{x}^*)^{\gamma_l}} \le r_1 \cdot \frac{\sum_{i\in M} \overline{V}_i(\boldsymbol{x}^*)^{\gamma_i}}{v_0+\sum_{l\in M} \overline{V}_l(\boldsymbol{x}^*)^{\gamma_l}}$, where $\overline{V}_i(\boldsymbol{x}^*)$ is an upper bound of $V_i(\boldsymbol{x}^*)$.

Algorithm 1 Heuristic for finding the upper bound of z^*

Require: Instance of the UAOP-CNL u.
1: **for** $i = 1$ to m **do**
2: Set $\bar{V}_i \leftarrow$ the sum of the top c largest elements of $\{v_{i1}, v_{i2}, \ldots, v_{in}\}$.
3: **end for**
4: **return** $r_1 \cdot \frac{\sum_{i\in M} \bar{V}_i^{\gamma_i}}{v_0+\sum_{l\in M} \bar{V}_l^{\gamma_l}}$.

Z_LOWER corresponds to a heuristic algorithm that generates a sufficiently good feasible solution for the CAOP-CNL. Since developing effective heuristic algorithm is an important contribution of this study, we will explicitly introduce two candidate heuristic algorithms with different solution strategies in Section 4. After generating the lower bound $\underline{z}$ and the upper bound $\bar{z}$, we successively solve the parameterized problems $F(\underline{z})$ and $F(\bar{z})$, updating the best-found assortment $\boldsymbol{x}_{best}$ according to its expected revenue. We then proceed to the while loop of the binary search process. In line with Alfandari et al. (2021), aside from the convergence of z and the reach of a time limit, we present an additional stopping criterion for the while loop that expedites the binary search process, as demonstrated in the following proposition.

Proposition 1 (*Alfandari et al., 2021*). *At any iteration of the binary search process, let $\boldsymbol{x}_l$ and $\boldsymbol{x}_u$ be the optimal solutions to $F(\underline{z})$ and $F(\bar{z})$, respectively. If $\boldsymbol{x}_l = \boldsymbol{x}_u$, then $\boldsymbol{x}^* = \boldsymbol{x}_l = \boldsymbol{x}_u$ is the optimal assortment for the original problem.*

Proof. First, observe that $F(z)$ is a piecewise linear function of z, where each linear segment corresponds to a specific feasible assortment $\boldsymbol{x}$. If $\boldsymbol{x}_l = \boldsymbol{x}_u$, then $\underline{z}$ and $\bar{z}$ must lie on the same linear segment. Since z^* must reside within the interval $[\underline{z}, \bar{z}]$, the optimal assortment $\boldsymbol{x}^*$ must coincide with $\boldsymbol{x}_l$ and $\boldsymbol{x}_u$. □

In our pretests, this additional stopping criterion was demonstrated to be an effective measure for reducing the computation time of the exact method.

Algorithm 2 (Alfandari et al., 2021) Binary search algorithm for solving the CAOP-CNL

Require: Instance of the CAOP-CNL u, tolerance ϵ, time limit TL.
1: $\underline{z} \leftarrow$ Z_LOWER(u).
2: $\bar{z} \leftarrow$ Z_UPPER(u).
3: Obtain $\boldsymbol{x}_l$ by solving $F(\underline{z})$.
4: Set $\boldsymbol{x}_{best} \leftarrow \boldsymbol{x}_l$.
5: Obtain $\boldsymbol{x}_u$ by solving $F(\bar{z})$.
6: **if** $R(\boldsymbol{x}_u) > R(\boldsymbol{x}_{best})$ **then**
7: $\boldsymbol{x}_{best} \leftarrow \boldsymbol{x}_u$.
8: **end if**
9: **while** $v_0\bar{z} > v_0\underline{z} + \epsilon$ and $\boldsymbol{x}_l \neq \boldsymbol{x}_u$ and TL is not exceeded **do**
10: Set $z \leftarrow (\bar{z} + \underline{z})/2$.
11: Obtain $\boldsymbol{x}^*$ by solving $F(z)$.
12: **if** $R(\boldsymbol{x}^*) > R(\boldsymbol{x}_{best})$ **then**
13: $\boldsymbol{x}_{best} \leftarrow \boldsymbol{x}^*$.
14: **end if**
15: **if** $v_0 z < F(z)$ **then**
16: $\underline{z} \leftarrow z$.
17: **else**
18: $\bar{z} \leftarrow z$.
19: **end if**
20: **end while**
21: **return** $(R(\boldsymbol{x}_{best}), \boldsymbol{x}_{best})$.

3.2. Solution approach to the parameterized problem (CNLAPP)

In this subsection, we provide a detailed discussion on how to solve $F(z)$ for $\forall z \in R^+$. Following the approach of Alfandari et al. (2021), we refer to the maximization problem $F(z)$ as the *Cross-Nested Logit Assortment Parameterized Problem (CNLAPP)*, which is dependent on the parameter z:

$$\begin{aligned}\text{(CNLAPP)} &\max_{\boldsymbol{x}\in\{0,1\}^n:\|\boldsymbol{x}\|_1\le c}\left\{\sum_{i\in M} V_i(\boldsymbol{x})^{\gamma_i}[R_i(\boldsymbol{x}) - z]\right\} \\ &= \max_{\boldsymbol{x}\in\{0,1\}^n:\|\boldsymbol{x}\|_1\le c}\left\{\sum_{i\in M} \frac{\sum_{j\in N}(r_j - z)v_{ij}x_j}{(\sum_{j\in N} v_{ij}x_j)^{1-\gamma_i}}\right\}.\end{aligned} \tag{6}$$

It is important to note that the fractional form $\frac{\sum_{j\in N}(r_j-z)v_{ij}x_j}{(\sum_{j\in N} v_{ij}x_j)^{1-\gamma_i}}$ for $V_i(\boldsymbol{x})^{\gamma_i}[R_i(\boldsymbol{x}) - z]$ is valid only if $V_i(\boldsymbol{x}) > 0$. Otherwise, this term equals 0 and should be excluded from the summation.

Given the NP-hardness of the CAOP-CNL, developing a polynomial-time algorithm capable of optimally solving the CNLAPP remains challenging, unless the CAOP-CNL itself is proven to be polynomially solvable. Furthermore, the decision variables within the CNLAPP are inherently binary. Therefore, we develop a customized Branch-and-Bound

[3] Because $F(z)$ is a point-wise maximum of finite linear decreasing functions of z (Zhang et al., 2020).

algorithm to obtain the optimal solution for the CNLAPP.

For each node t in the B&B search tree, let S_0^t and S_1^t represent the sets of products whose binary variables have already been fixed to 0 and 1, respectively. The values of these variables cannot be altered in the successors of node t. We denote $\bar{S}^t$ as the set of products for which inclusion in the assortment has not yet been determined. The remaining cardinality at node t is given by $c_t = c - |S_1^t|$. It should be noted that S_0^t, S_1^t, $\bar{S}^t$ and c_t are initially inherited from the predecessor node of t and subsequently updated based on the preprocessing and branching operations. Let $a_{it} = \sum_{j\in S_1^t} v_{ij}$ and $b_{it} = \sum_{j\in S_1^t}(r_j - z)v_{ij}$. The parameterized problem (CNLAPP) at node t can then be reformulated as follows:

$$(\text{CNLAPP} - t) \quad \max_{\boldsymbol{x}\in\{0,1\}^{|\bar{S}^t|}:\|\boldsymbol{x}\|_1\le c_t} \left\{ \sum_{i\in M} \frac{b_{it} + \sum_{j\in\bar{S}^t}(r_j - z)v_{ij}x_j}{(a_{it} + \sum_{j\in\bar{S}^t} v_{ij}x_j)^{1-\gamma_i}} \right\}. \tag{7}$$

According to the proposition below, we can include all products with revenue no larger than z into the set S_0^0 at root node (node 0). This inclusion ensures that b_{it} remains non-negative in the CNLAPP-t problem at any node t.

Proposition 2 (*Alfandari et al., 2021*). *Given $z \in R^+$, the optimal solution $\boldsymbol{x}^*$ to the parameterized problem (CNLAPP) must satisfy $x_j^* = 0, \forall j \in \{k \in N | r_k \le z\}$.*

This proposition represents a specific case of Proposition 4 (Alfandari et al., 2021) with $\gamma_i \le 1$, and its rationale is straightforward: adding a product k with $r_k \le z$ only decreases the numerator in (7). Simultaneously, since $\gamma_i \in (0, 1]$, it increases the denominator, thereby reducing the value of the objective function under consideration.

Before computing the upper bound at node t and performing pruning or branching operations, it is worthwhile to leverage the unique structure of the objective function in CNLAPP-t to accelerate the search process. Specifically, we attempt to fix certain variables in $\bar{S}^t$ to 1 without compromising the optimality of the solution. Effective variable-fixing can significantly reduce the search space complexity in subsequent nodes, thereby improving computational efficiency. We now present a series of lemmas that facilitate the variable-fixing operations. The proofs of these lemmas are provided in Appendices A.1, A.2 and A.3.

Lemma 2. *Given $z = \frac{b}{a^{1-\gamma}} > 0$, where $\gamma \in (0, 1], a, b > 0$. Then for $\forall r, v$ satisfying $r \ge \frac{b}{a}$ and $v > 0$, we have $z' = \frac{b+rv}{(a+v)^{1-\gamma}} > z$.*

Lemma 3. *Given $\frac{b+r_1v_1}{(a+v_1)^{1-\gamma}} \ge \frac{b}{a^{1-\gamma}} > 0$, where $\gamma \in (0, 1]$, $a, b, r_1, v_1 > 0$. Then for $\forall r_2, v_2$ satisfying $r_2 \ge r_1$ and $v_2 > 0$, we have $z' = \frac{b+r_1v_1+r_2v_2}{(a+v_1+v_2)^{1-\gamma}} \ge z = \frac{b+r_1v_1}{(a+v_1)^{1-\gamma}}$.*

Lemma 4. *Given $\frac{b+r_1v_1}{(a+v_1)^{1-\gamma}} \ge \frac{b}{a^{1-\gamma}} > 0$, where $\gamma \in (0, 1], a, b, r_1, v_1 > 0$. Then for $\forall r_2, v_2$ satisfying $0 < r_2 \le r_1$ and $v_2 > 0$, we have $z' = \frac{b+r_1v_1+r_2v_2}{(a+v_1+v_2)^{1-\gamma}} \ge z = \frac{b+r_2v_2}{(a+v_2)^{1-\gamma}}$.*

Based on the lemmas provided above, when the number of undetermined products does not exceed the remaining cardinality, we can attempt to iteratively fix the remaining undetermined binary variables to 1, as outlined in the following proposition:

Proposition 3. *If node t satisfies $|\bar{S}^t| \le c_t$. Denote s as the product with the largest revenue in $\bar{S}^t$, if $\frac{b_{it}+(r_s-z)v_{is}}{(a_{it}+v_{is})^{1-\gamma_i}} \ge \frac{b_{it}}{a_{it}^{1-\gamma_i}}$, for $\forall i \in \{k \in M | v_{ks} > 0, a_{kt} > 0\}$. Then there exits an optimal solution $\boldsymbol{x}^*$ to CNLAPP-t satisfying $x_s^* = 1$.*

The proof of this proposition is provided in Appendix A.4. Here we refer to the first inequality $|\bar{S}^t| \le c_t$ as the "precondition", and the second inequality $\frac{b_{it}+(r_s-z)v_{is}}{(a_{it}+v_{is})^{1-\gamma_i}} \ge \frac{b_{it}}{a_{it}^{1-\gamma_i}}$ as the "variable-fixing condition". By examining the structure of the inequality in the variable-fixing condition, we can analytically identify several scenarios under which product s is more likely to be included in the optimal assortment for CNLAPP-t, provided the precondition is met.

First, since an increase in r_s only raises the value of the numerator, a product s with higher revenue is more likely to be included in the optimal assortment. Particularly, if product s has a revenue $r_s > b_{it}/a_{it} + z$, it must be included in the optimal assortment, as indicated by Lemma 2. Second, even if product s has a revenue lower than $b_{it}/a_{it} + z$, it is more likely that the variable-fixing condition will hold if s has a relatively high nest-specific preference weight for each nest to which it is positively allocated. This is because the denominator term is a concave function with respect to v_{is}, given that $\gamma_i \in (0, 1]$. If v_{is} is sufficiently large, the increase in the numerator will significantly exceed that in the denominator, thereby making the left-hand side of the inequality larger than the right-hand side. Moreover, if product s belongs to nests with relatively higher dissimilarity parameters, the variable-fixing condition is also more likely to be satisfied. This is because the denominator term increases more slowly when $1-\gamma_i$ is close to 0 compared to when $1 - \gamma_i$ is close to 1. Consequently, the inclusion of product s would raise the left-hand side of the inequality without requiring a large value of v_{is}.

It is important to note that when $z = z^*$, the optimal solution at node t may correspond to the optimal solution of the original problem. Therefore, the properties discussed above also shed light on the optimal assortment for the CAOP-CNL. In summary, the final optimal assortment typically includes products with relatively higher revenue and higher nest-specific preference weights, A product with higher revenue can enhance the expected revenue generated by the nest, while a higher nest-specific preference weight can increase the nest's attractiveness, thereby boosting the purchase probability. When the cardinality capacity is limited, a trade-off may arise between high revenue and large preference weights. A product with relatively lower revenue may be included in the optimal assortment if it significantly enhances the attractiveness of one or more nests, while a higher-revenue product with low preference weight may be excluded. Additionally, the optimal assortment is more likely to include products associated with nests that have relatively higher dissimilarity parameters. This is because, with a large dissimilarity parameter, the substitution effects between product within the same nest are weak, necessitating the inclusion of more products to satisfy the diverse preferences of customers.

Regarding the implementation of Proposition 3, we first verify whether node t satisfies the precondition. If the precondition is met and product s satisfies the variable-fixing condition, we fix the value of x_s to 1 in all successor nodes of node t and update several terms as follows: $a_{it} = a_{it} + v_{is}, b_{it} = b_{it} + (r_s - z)v_{is}, \forall i \in M, c_t = c_t - 1, S_1^t = S_1^t \cup \{s\}, \bar{S}^t = \bar{S}^t \backslash \{s\}$. The variable-fixing operation is performed iteratively until either the precondition or the variable-fixing condition no longer holds. Due to the existence of the precondition, the variable-fixing operation is particularly effective when the total cardinality capacity is large.

Now we proceed to develop an efficient upper bound of CNLAPP-t when no variables can be fixed to 1. Note that the main difficulty of solving CNLAPP-t arises from the nest overlapping. More specifically, for a given $j \in N$, v_{ij} may take values greater than 0 for multiple $i \in M$. This implies that the decision to include product j (i.e. the value of x_j) affects the objective function values across multiple nests, instead of only one nest under the NL model. However, if we could eliminate such overlapping, then the problem becomes decomposable by nests, and the sub-problem can be readily solved based on the extensive results of assortment optimization problems under the NL models. Therefore, our central idea is to reformulate CNLAPP-t so that the overlapping between nests is represented by newly introduced constraints rather than decision variables in the objective function. We then solve the Lagrangian dual of the reformulated problem to obtain an upper bound for CNLAPP-t. Specifically, problem (7) is reformulated as follows:

$$\max \quad \sum_{i\in M} \frac{b_{it}+\sum_{j\in\bar{S}^t}(r_j-z)v_{ij}x_{ij}}{(a_{it}+\sum_{j\in\bar{S}^t}v_{ij}x_{ij})^{1-\gamma_i}} \tag{8a}$$

$$\text{s.t.} \quad \sum_{j\in\bar{S}^t} x_{ij} \le c_t, \quad i\in M \tag{8b}$$

$$\sum_{i\in M} x_{ij} = m y_j, \quad j\in\bar{S}^t \tag{8c}$$

$$x_{ij}\in\{0,1\}, \quad i\in M, j\in\bar{S}^t \tag{8d}$$

$$y_j\in\{0,1\}, \quad j\in\bar{S}^t. \tag{8e}$$

In the reformulation, we expand each nest-independent decision variable $x_j \in \bar{S}^t$ in problem (7) into m nest-specific binary decision variables $\{x_{1j}, x_{2j}, \dots, x_{mj}\}$. This approach allows us to eliminate the effect of nest overlapping on the objective function by replacing each x_j with x_{ij} in the fractional function specific to nest i. Instead, we capture the nest overlapping effect within the constraint space. Specifically, we add $|\bar{S}^t|$ auxiliary binary variables $y_j, j\in\bar{S}^t$ and impose constraints (8c) to restrict that $x_{ij}, \forall i\in M$ should take the same value for each $j\in\bar{S}^t$. The cardinality constraint is also replicated for each nest as shown in (8b). It is evident that problems (7) and (8) are equivalent. Therefore, we omit the proof of equivalency here.

For problem (8), we introduce a vector of Lagrangian multipliers $\boldsymbol{\mu}\in R^{|\bar{S}^t|}$ for constraints (8c) and formulate the corresponding Lagrangian dual problem as follows:

$$\min_{\boldsymbol{\mu}} g(\boldsymbol{\mu}), \tag{9a}$$

$$\text{where } g(\boldsymbol{\mu}) = \max_{\boldsymbol{x}_1,\dots,\boldsymbol{x}_m,\boldsymbol{y}} \sum_{i\in M} \frac{b_{it}+\sum_{j\in\bar{S}^t}(r_j-z)v_{ij}x_{ij}}{(a_{it}+\sum_{j\in\bar{S}^t}v_{ij}x_{ij})^{1-\gamma_i}} + \sum_{j\in\bar{S}^t}\mu_j\Big(\sum_{i\in M}x_{ij} - m y_j\Big) \tag{9b}$$

$$\text{s.t.} \quad \sum_{j\in\bar{S}^t} x_{ij} \le c_t, \quad i\in M \tag{9c}$$

$$\boldsymbol{x}_i = (x_{ij})_{j\in\bar{S}^t}\in\{0,1\}^{|\bar{S}^t|}, \quad i\in M \tag{9d}$$

$$\boldsymbol{y} = (y_j)_{j\in\bar{S}^t}\in\{0,1\}^{|\bar{S}^t|}. \tag{9e}$$

Here, we encapsulate all decision variables $x_{ij}, j\in\bar{S}^t$ related to nest i into the vector $\boldsymbol{x}_i$, which captures all decisions associated with the objective function of nest i in the reformulated problem. The optimal value of dual problem (9) serves as an upper bound for CNLAPP-t. Fortunately, the optimal Lagrangian multipliers for this dual problem can be readily determined, as demonstrated in the following proposition. The proof of this proposition is provided in Appendix A.5.

Proposition 4. *There are a set of optimal Lagrangian multipliers $\boldsymbol{\mu}^*$ for problem (9) satisfying that $\boldsymbol{\mu}^* = \mathbf{0} = (0, 0, \dots, 0)$.*

Therefore, $g(\mathbf{0})$ provides an efficient upper bound to CNLAPP-t. We can define $g(\mathbf{0})$ as a *decoupling problem* of CNLAPP-t as follows:

$$(d-\text{CNLAPP}-t) \max_{\substack{\boldsymbol{X}\in\{0,1\}^{m\times|\bar{S}^t|}:\\ \|\boldsymbol{x}_i\|_1\le c_t,\forall i\in M}} \left\{\sum_{i\in M}\frac{b_{it}+\sum_{j\in\bar{S}^t}(r_j-z)v_{ij}x_{ij}}{(a_{it}+\sum_{j\in\bar{S}^t}v_{ij}x_{ij})^{1-\gamma_i}}\right\}, \tag{10}$$

where $\boldsymbol{X}$ is an $m\times|\bar{S}^t|$ matrix with each element being a binary variable. Obviously, the decoupling problem (10) is separable by nest and can be rewritten as:

$$(d-\text{CNLAPP}-t)\sum_{i\in M}\max_{\substack{\boldsymbol{x}_i\in\{0,1\}^{|\bar{S}^t|}:\\ \|\boldsymbol{x}_i\|_1\le c_t}} \left\{\frac{b_{it}+\sum_{j\in\bar{S}^t}(r_j-z)v_{ij}x_{ij}}{(a_{it}+\sum_{j\in\bar{S}^t}v_{ij}x_{ij})^{1-\gamma_i}}\right\} \tag{11}$$

Finally, we observe that the decoupling problem can be optimally solved in polynomial time.

Lemma 5. *The decoupling problem d-CNLAPP-t can be solved optimally in a polynomial time complexity of $O(m|\bar{S}^t|^3)$.*

To save space, we provide the proof of this lemma in Appendix A.6. It is worth noting that the initial upper bound computed by Algorithm 1 can be further refined by decoupling the original problem into a CAOP under the NL model, as discussed explicitly in Appendix B.3.

Once the upper bound for node t is obtained, it is essential to compare it with the current best-known objective value among all traversed nodes. If the upper bound is not greater than the best-known objective value, further branching on this node is unnecessary.

Regarding the branching rule at node t, in addition to the rule proposed by Alfandari et al. (2021), which branches on the variable with the highest revenue, we introduce an alternative rule that branches on the variable contributing the most to the current objective function at node t. Specifically, we iteratively set each variable in $\bar{S}^t$ to 1 and compute the resulting increment in the objective function value of problem (7) at node t. After evaluating all undetermined variables, we select the one that yields the largest increase in the objective value for branching at this node. The effects of these two branching rules are compared in our numerical experiments.

For the traversal strategy in the Branch-and-Bound tree, based on our preliminary tests, we determine to traverse the branching tree using a best-first search approach, prioritizing the node with the current largest objective value.[4]

The main operations at node t can be summarized as follows:

- Inherit c_t, S_0^t, S_1^t and $\bar{S}^t$ from the predecessor node;
- variable-fixing: Attempt to iteratively fix variables in $\bar{S}^t$ to 1 using Proposition 3, and update c_t, S_1^t and $\bar{S}^t$ after each variable-fixing operation;
- Deriving upper bound: Calculate the upper bound of node t by solving the decoupling problem d-CNLAPP-t. Compare this upper bound with the best-known objective value to determine whether further branching at node t;
- Branching: Applying one of the branching rule mentioned above to select a binary variable in $\bar{S}^t$ for branching, thereby generating successor nodes with updated c_t, S_0^t, S_1^t and $\bar{S}^t$.

The pseudo-codes for all procedures performed at the root node and all other nodes are provided in Appendix B.1 and Appendix B.2, respectively.

4. Heuristic algorithms

Since the exact method embeds a Branch-and-Bound algorithm, its computation time may increase dramatically with the growth of instance size. Therefore, developing a heuristic algorithm that can efficiently solve large instances while delivering high-quality solutions is of considerable value. In this section, we introduce two heuristic algorithms designed to address the CAOP-CNL, and their performance will be compared in subsequent numerical experiments.

The primary concept of the first heuristic algorithm is to identify a sorted-by-revenue assortment that generates the highest expected revenue. Drawing on the definitions presented in Alfandari et al. (2021) and Davis et al. (2014), we represent each sorted-by-revenue assortment in the form $S_{lk} = \{l, l+1, \dots, k\}$, where $1\le l\le k\le n$ and $k-l+1\le c$. This approach can be viewed as an enumeration-based method, and Algorithm 3 provides a detailed description.

The second heuristic integrates concepts from both our exact method and the greedy algorithm. Specifically, the exact algorithm

[4] In our preliminary experiments, we assessed the computational efficiency of three prevalent tree traversal algorithms: depth-first search, best-first search, and breadth-first search. Our findings indicate that the best-first search algorithm exhibits superior computational speed relative to the other two methods. Consequently, we have decided to employ best-first search for tree traversal in our subsequent analyses.

Algorithm 3 Heuristic algorithm 1 for solving the CAOP-CNL

Require: Instance of the CAOP-CNL u.

$R_{max} \leftarrow 0$.
for $k = 1$ to c **do**
  for $i = 1$ to $n - k + 1$ **do**
    Set $\boldsymbol{x}^{ik}$ such that $x_j^{ik} = 1, \forall j \in S_{i(i+k-1)}$ and $x_j^{ik} = 0, \forall j \notin S_{i(i+k-1)}$.
    $R_{ik} \leftarrow R(\boldsymbol{x}^{ik})$.
    if $R_{ik} > R_{max}$ **then**
      $R_{max} \leftarrow R_{ik}$.
    end if
  end for
end for
return R_{max} and its corresponding solution $\boldsymbol{x}^{ik}$.

derives the optimal solution by solving the fixed-point problem $v_0 z = F(z)$. Given the monotonically decreasing nature of $F(z)$, identifying a lower bound function $G(z)$ that closely approximates $F(z)$, and determining the intersection point between $v_0 z$ and $G(z)$, enable us to establish a lower bound for the original problem. To derive this lower bound, a straightforward greedy algorithm can be employed. The algorithm starts with an empty assortment and iteratively adds products that maximize the increment in the metric $Q(S, z) = \sum_{i \in M} \frac{\sum_{j \in S}(r_j - z)v_{ij}}{(\sum_{j \in S} v_{ij})^{1-\gamma_i}}$.

The inclusion process is halted either upon reaching the cardinality limit or when no further increment in this metric can be observed. The output from this greedy algorithm, denoted by $G(z)$, serves as a lower bound of $F(z)$ because $F(z)$ represents the maximum of $Q(S, z)$ among all feasible assortments. A detailed description of the second heuristic is provided in Algorithm 4.

Algorithm 4 Heuristic algorithm 2 for solving the CAOP-CNL

Require: Instance of the CAOP-CNL u, tolerance ε, time limit TL.

$\underline{z} \leftarrow 0$.
$\bar{z} \leftarrow$ Z_UPPER(u) using Algorithm 1.
while $v_0 \bar{z} > v_0 \underline{z} + \varepsilon$ and TL is not exceeded **do**
  Set $z \leftarrow (\bar{z} + \underline{z})/2$.
  $S \leftarrow \emptyset, k \leftarrow 0$.
  while $k < c$ **do**
    Find $j \in N \setminus S$ that maximizes $Q(S \cup \{j\}, z) - Q(S, z)$.
    if $Q(S \cup \{j\}, z) > Q(S, z)$ **then**
      $S \leftarrow S \cup \{j\}$.
      $k \leftarrow k + 1$.
    else
      Stop the while loop.
    end if
  end while
  $G(z) \leftarrow Q(S, z)$.
  if $v_0 z < G(z)$ **then**
    $\underline{z} \leftarrow z$.
  else
    $\bar{z} \leftarrow z$.
  end if
end while
return $G(z)$ and its corresponding greedy solution $\boldsymbol{x}_g$.

Note that $G(z)$ may not be continuous with respect to z, but it is upper-bounded by the continuous decreasing function $F(z)$. This guarantees that $v_0 z \geq G(z)$ when z is equal to the initial upper bound. Additionally, since each product has positive revenue and preference weight, $G(0)$ must be greater than 0, indicating that $v_0 z < G(z)$ when $z = 0$. As a result, the binary search process will consistently converge, ensuring the second heuristic always feasible.

5. Numerical experiments

5.1. Experimental settings

In this section, we evaluate the performance of our exact solution algorithm and two heuristic approaches for solving the CAOP-CNL through extensive numerical experiments. Specifically, we test these algorithms on manually generated instances with the number of nests $m \in \{5, 10\}$ and the number of products $n \in \{25, 50, 100, 150, 200, 300, 500\}$. Following the manner of Alfandari et al. (2021), we generate 20 instances for each combination of (m, n), resulting in a total of 280 instances to be solved. As for the generation of revenue r_j and preference weight v_j for product j in each instance, we adopt the approach outlined in Gallego and Topaloglu (2014). Specifically, u_j is first generated from a uniform distribution over $[0, 1]$, while x_j and y_j are drawn from a uniform distribution over $[0.75, 1.25]$. The revenue is then calculated as $r_j = 10 \times u_j^2 \times x_j$ and the preference weight is computed as $v_j = 10 \times (1 - u_j) \times y_j$. Due to the inclusion of u_j, products with higher revenues are more likely to have lower preference weights. However, since x_j and y_j are generated independently, it is not always the case that a more expensive product has a lower preference weight. Additionally, the quadratic term u_j^2 skews the revenue distribution, resulting in a large number of products with low revenues and a small number of products with high revenues. In line with Le and Mai (2024), we set the preference weight of the no-purchase option as a baseline in the function computing the preference weight of each product. This ensures that the preference weight of the no-purchase option is of the same magnitude as that of any other product. Specifically, we set $v_0 = 10$.

Regarding the nesting specification, we first sample the dissimilarity parameter γ_i for each nest i from a uniform distribution over $[0.25, 0.75]$ (Gallego & Topaloglu, 2014; Le & Mai, 2024). Since a product may be assigned to multiple nests under the CNL model, we introduce a parameter $\delta > 1$ to represent the average number of nests to which a product belongs, thereby controlling the nest overlapping rate. Following the approach of Le and Mai (2024), the value of δ is set to 1.2. We also conduct a sensitivity analysis to examine the impact of the overlapping rate on the performance of our algorithms, as detailed in Appendix C. Once the overlapping rate δ is determined, the nesting structure under the CNL model, specifically the values of α_{ij} for all $i \in M$ and $j \in N$, is randomly constructed as described in Algorithm 5.

Algorithm 5 Procedures of generating $\alpha_{ij}, \forall i \in M, j \in N$

Require: δ, n, m.

Set $\alpha_{ij} \leftarrow 0, \forall i \in M, j \in N$.
Set $k \leftarrow 1$.
while $k \leq \lceil \delta \times n \rceil$ **do**
  if $k \leq n$ **then**
    Randomly and uniformly generate i from M.
    Generate α from the uniform distribution $[1, 10]$.
    Set $\alpha_{ik} \leftarrow \alpha$.
    $k \leftarrow k + 1$.
  else
    Randomly and uniformly generate i from M and j from N.
    if $\alpha_{ij} = 0$ **then**
      Generate α from the uniform distribution $[1, 10]$.
      Set $\alpha_{ij} \leftarrow \alpha$.
      $k \leftarrow k + 1$.
    end if
  end if
end while
for $j = 1$ to n **do**
  $\alpha_j \leftarrow \sum_{i \in M} \alpha_{ij}$.
  Set $\alpha_{ij} \leftarrow \frac{\alpha_{ij}}{\alpha_j}, \forall i \in M$.
end for

When addressing each instance, we impose a total cardinality constraint, i.e. $\|\boldsymbol{x}\|_1 \leq c$, to limit the total number of products included in the assortment. To assess the impact of this cardinality constraint, each instance is solved thrice with c successively set to values from the set $\{\lceil 0.1n \rceil, \lceil 0.2n \rceil, \lceil 0.3n \rceil\}$. For clarity in interpretation, we define $c = \lceil 0.1n \rceil$ as a relatively strict cardinality constraint, while $c = \lceil 0.3n \rceil$ is considered a relatively relaxed constraint. Regarding the parameters of the binary search algorithm, we set the tolerance $\epsilon = 10^{-5}$ and the time limit TL to 3600 s. If the binary search algorithm does not converge to optimality within the time limit, we report the best-found assortment ($\boldsymbol{x}_{best}$ in Algorithm 2) and its corresponding objective value. The initial upper bound is computed using the modified method described in Appendix B.3, while the initial lower bound is obtained via the second heuristic (Algorithm 4).

We also attempted to solve CAOP-CNL instances optimally using alternative exact solution methods for benchmarking purposes. However, in our preliminary tests, we encountered significant challenges in optimally solving the non-linear integer parameterized problem (CNLAPP) using either open-source tools (e.g. Couenne) or commercial software (e.g. Cplex, Gurobi, GAMS) due to the complexity of the objective function.[5] As a result, we employ a full enumeration approach to calculate the optimal solution for comparison of computation times. However, this method is only feasible for relatively small problem sizes.

All the experiments are implemented using C++ and conducted on a desktop computer equipped with an Intel(R) Core(TM) i5-7300U CPU @ 2.60 GHz.[6]

5.2. Numerical results

The primary numerical results are reported in Tables 1 to 7, with each table corresponding to a specific value of n. All tables display the values of parameters n, m and c in the first two rows. For brevity, we denote the first heuristic algorithm as "SBR" (sorted-by-revenue) and the second as "BS+GA" (binary search + greedy algorithm). The exact algorithm is abbreviated as "BS+BB" (binary search + Branch-and-Bound).

Columns three through eight, each referred to as a "scenario", represent distinct combinations of the parameters n, m, and c. Under each scenario, we solve 20 instances and report the mean objective value (Obj.) achieved and the CPU time in seconds (Time) consumed by each algorithm. For the exact algorithm, we solve the 20 instances in two rounds, each round implementing a different branching rule. The branching rule used in Alfandari et al. (2021) is denoted as Branching Rule 1 (BR1), while the newly introduced rule at the end of Section 3 is referred to as Branching Rule 2 (BR2). For the round with the lower computation time, we also record the average number of iterations (# Iter.)[7] and the count of instances that achieve convergence (# Conv.) within the time limit. To compare the performance of the exact method with the heuristics, we record the number of instances where the exact algorithm outperforms each heuristic (# Imp.) and its average percentage improvement in expected revenue over each heuristic. The percentage improvement is calculated as follows:

$$\text{Impr.}(\%) = \frac{1}{20}\sum_{k=1}^{20} 100 \times \left(\frac{\text{BEST}^k - \text{HEUR}^k}{\text{HEUR}^k}\right), \tag{12}$$

where k indexes the instances, HEUR^k represents the heuristic's objective value for instance k, and BEST^k corresponds to the best objective value $R(\boldsymbol{x}_{best})$ found by the exact method.

It can be observed that the enumeration method is feasible only in limited scenarios (i.e., $n \leq 50$ and $c \leq 8$). In contrast, the exact algorithm (BS+BB) can handle instances with larger n and c within a reasonable time limit. Specifically, our exact algorithm converges to optimality in all test instances with $n \leq 150$ and in the majority of instances ($\geq$90%) with $n \leq 300$ within a one-hour time limit. These results highlight the significantly greater capability of our exact method in solving the CAOP-CNL compared to the full enumeration method.

Regarding the performance of the two heuristics, we observe that the second heuristic algorithm (BS+GA) consistently outperforms the first heuristic (SBR) across all scenarios. The values of #Imp. and Impr.(%) further demonstrate that BS+GA achieves optimal solutions for a large proportion of the test instances, with an average optimality gap within 0.2%.[8] These findings suggest that for time-sensitive assortment optimization needs (e.g., immediate product recommendation requirements on e-commerce platforms), the second heuristic can provide prompt solutions while maintaining high solution quality.

The first heuristic exhibits the shortest computation time among all the algorithms. However, its objective value is considerably inferior to that of the exact algorithm under the relatively strict cardinality constraint, indicating significant differences between the optimal assortment structure and the sorted-by-revenue assortment when the number of available products is limited. Nevertheless, as the cardinality constraint is relaxed, the optimality gap of SBR narrows substantially. This suggests that the optimal assortment structure increasingly aligns with the sorted-by-revenue assortment as the carnality limit c approaches the total number of products n.

We now examine the impact of branching rules on the performance of the exact algorithm. For each scenario, we highlight in bold the CPU time of the exact method with the lower value (if the two CPU times are identical, both are bolded). The results indicate that the exact algorithm implementing our newly introduced branching rule (BR2) consumes significantly shorter computation time than the one using BR1 in the majority of scenarios (specifically, 32 out of 42 scenarios). However, when the number of products reaches a certain level, the exact method using BR1 can outperform BR2 under relatively relaxed cardinality constraints (in 5 cases when $c = \lceil 0.3n \rceil$ and in 2 cases when $c = \lceil 0.2n \rceil$).

This phenomenon underscores the significance of selecting appropriate branching rules in the Branch-and-Bound algorithm under different scenarios. In our problem, prioritizing branching on the variables that are likely to contribute more significantly to the objective value proves beneficial when the cardinality constraint is relatively strict. We attribute this to the fact that, under a relatively strict cardinality constraint, the structure of the optimal assortment deviates considerably from the sorted-by-revenue assortment, as previously discussed. When using BR1, which branches on variables in a revenue-decreasing order, the algorithm may initially branch on many variables that are ultimately set to 0 in the optimal solution. This can lead to the generation of redundant nodes in the Branch-and-Bound tree. In contrast, by implementing BR2, which follows a greedy mechanism for branching, the search process can quickly approach nodes with relatively high objective values early on, facilitating more effective pruning and reducing the search space in subsequent iterations. Additionally, it is noteworthy that the greedy mechanism employed in BR2 is identical to that used in the second heuristic (BS+GA). As a result, even in instances where the exact algorithm does not fully converge, it can still generate a best-found solution with an objective value no less than those produced by the heuristics. This is because BR2 can guide the search process to

[5] Couenne requires the objective function to be factorable, a condition not satisfied by our formulation. Cplex and Gurobi are limited to solving quadratic integer programming problems. While GAMS packages can generate a solution for our problem, the solution is not optimal in most cases.

[6] The source code and test instances used in this study are available at the following GitHub repository: https://github.com/zlnewplayer/CAOP-CNL.

[7] This value equals 1 if the exact algorithm only solves $F(\underline{z})$ within the time limit. Otherwise, it is calculated as 2 plus the number of iterations the exact method goes through the while loop.

[8] When the exact algorithm converges to optimality, the optimality gap can be calculated using the Impr.(%) value: Optimality gap $= 1 - \frac{1}{1+\text{Impr.}(\%)}$.

Table 1
Results on instances with $n = 25$.

$n = 25$		$m = 5$			$m = 10$		
		$c = \lceil 0.1n \rceil$	$c = \lceil 0.2n \rceil$	$c = \lceil 0.3n \rceil$	$c = \lceil 0.1n \rceil$	$c = \lceil 0.2n \rceil$	$c = \lceil 0.3n \rceil$
H1: SBR	Obj.	2.358	2.775	3.010	2.374	2.955	3.298
	Time	0.00	0.00	0.00	0.00	0.00	0.00
H2: BS+GA	Obj.	2.499	2.916	3.040	2.591	3.126	3.356
	Time	0.00	0.00	0.00	0.00	0.00	0.00
BS+BB	Obj.	2.499	2.916	3.040	2.591	3.128	3.356
	Time (BR1)	0.02	0.02	0.02	0.02	0.02	**0.02**
	Time (BR2)	**0.01**	**0.01**	**0.01**	**0.01**	**0.01**	**0.02**
	#Conv.	20	20	20	20	20	20
	#Imp. vs. H1	17	17	11	18	18	12
	#Imp. vs. H2	0	0	0	0	1	0
	Impr. (%) vs. H1	6.34	5.31	1.01	9.31	6.04	1.78
	Impr. (%) vs. H2	0.00	0.00	0.00	0.00	0.07	0.00
	#Iter.	2.7	2.1	2.0	2.8	2.5	2.1
Enum.	Time	0.01	0.08	2.19	0.01	0.14	4.13

Table 2
Results on instances with $n = 50$.

$n = 50$		$m = 5$			$m = 10$		
		$c = \lceil 0.1n \rceil$	$c = \lceil 0.2n \rceil$	$c = \lceil 0.3n \rceil$	$c = \lceil 0.1n \rceil$	$c = \lceil 0.2n \rceil$	$c = \lceil 0.3n \rceil$
H1: SBR	Obj.	3.035	3.476	3.608	3.082	3.741	3.899
	Time	0.00	0.00	0.00	0.00	0.00	0.00
H2: BS+GA	Obj.	3.275	3.581	3.638	3.318	3.852	3.919
	Time	0.00	0.00	0.01	0.00	0.01	0.01
BS+BB	Obj.	3.282	3.582	3.639	3.318	3.852	3.919
	Time (BR1)	0.17	0.47	0.05	0.15	0.06	**0.03**
	Time (BR2)	**0.03**	**0.05**	**0.03**	**0.06**	**0.02**	**0.03**
	#Conv.	20	20	20	20	20	20
	#Imp. vs. H1	20	20	13	19	18	13
	#Imp. vs. H2	4	2	2	1	1	1
	Impr. (%) vs. H1	8.34	3.09	0.89	7.83	3.01	0.54
	Impr. (%) vs. H2	0.20	0.04	0.01	0.00	0.00	0.00
	#Iter.	2.9	2.5	2.0	3.6	2.0	2.1
Enum.	Time	3.99	–	–	7.81	–	–

Table 3
Results on instances with $n = 100$.

$n = 100$		$m = 5$			$m = 10$		
		$c = \lceil 0.1n \rceil$	$c = \lceil 0.2n \rceil$	$c = \lceil 0.3n \rceil$	$c = \lceil 0.1n \rceil$	$c = \lceil 0.2n \rceil$	$c = \lceil 0.3n \rceil$
H1: SBR	Obj.	3.694	4.140	4.229	4.023	4.542	4.608
	Time	0.00	0.00	0.00	0.00	0.00	0.00
H2: BS+GA	Obj.	3.996	4.223	4.246	4.368	4.617	4.632
	Time	0.01	0.02	0.02	0.01	0.03	0.04
BS+BB	Obj.	3.999	4.223	4.246	4.368	4.618	4.633
	Time (BR1)	40.49	3.37	0.13	31.86	2.21	0.04
	Time (BR2)	**0.94**	**0.12**	**0.07**	**0.95**	**0.07**	**0.03**
	#Conv.	20	20	20	20	20	20
	#Imp. vs. H1	20	20	16	20	18	16
	#Imp. vs. H2	5	1	4	1	4	4
	Impr. (%) vs. H1	8.34	2.02	0.41	8.62	1.66	0.54
	Impr. (%) vs. H2	0.09	0.01	0.01	0.00	0.01	0.02
	#Iter.	2.9	2.2	2.0	2.8	2.2	2.0

quickly reach nodes that are close to the BS+GA solution.

Before discussing why BR1 outperforms BR2 in certain scenarios, we first examine the impact of variable-fixing operations on the performance of the exact algorithm. Specifically, we run the exact algorithm on all test instances without applying any variable-fixing operation. To enhance conciseness in presenting the results, for each scenario, we rerun the exact method using only the branching rule that achieved lower computation time in the original numerical experiments. The term "variable-fixing Operation" is abbreviated as "VFO". Additionally, results for $n = 25$ and 50 are omitted, as the computation times are negligible. The detailed results are provided in Table 8.

We highlight in bold the CPU times without VFO that show a significant increase compared to those with VFO (specifically, an increase exceeding 30%). It can be observed that under the relatively strict cardinality constraint, the computation times with and without VFO are nearly identical, suggesting that variable-fixing operations are rarely applied in such scenarios. We attribute this to the difficulty of satisfying the precondition in Proposition 3 under strict cardinality constraints. In contrast, when the cardinality constraint is relatively relaxed, neglecting VFO can result in a substantial increase in computation time, especially as the total number of products n grows. This is because the precondition in Proposition 3 is more likely to be satisfied under relaxed cardinality constraints, leading to more frequent application of variable-fixing. The increased number of variable-fixing operations can significantly reduce the search space in the Branch-and-Bound algorithm, thereby considerably decreasing the computation time. With

Table 4
Results on instances with $n = 150$.

$n = 150$		$m = 5$			$m = 10$		
		$c = \lceil 0.1n \rceil$	$c = \lceil 0.2n \rceil$	$c = \lceil 0.3n \rceil$	$c = \lceil 0.1n \rceil$	$c = \lceil 0.2n \rceil$	$c = \lceil 0.3n \rceil$
H1: SBR	Obj.	4.093	4.524	4.558	4.475	4.983	5.017
	Time	0.00	0.00	0.01	0.00	0.01	0.01
H2: BS+GA	Obj.	4.403	4.567	4.573	4.870	5.033	5.035
	Time	0.02	0.04	0.05	0.04	0.08	0.09
BS+BB	Obj.	4.404	4.567	4.573	4.872	5.033	5.035
	Time (BR1)	461.56	1.41	0.07	296.40	0.29	**0.03**
	Time (BR2)	**5.64**	**0.09**	**0.06**	**5.13**	**0.08**	**0.03**
	#Conv.	20	20	20	20	20	20
	#Imp. vs. H1	20	20	16	20	20	19
	#Imp. vs. H2	1	1	1	4	0	2
	Impr. (%) vs. H1	7.65	0.95	0.33	8.89	0.99	0.38
	Impr. (%) vs. H2	0.01	0.00	0.00	0.04	0.00	0.00
	#Iter.	2.5	2.2	2.0	3.0	2.1	2.1

Table 5
Results on instances with $n = 200$.

$n = 200$		$m = 5$			$m = 10$		
		$c = \lceil 0.1n \rceil$	$c = \lceil 0.2n \rceil$	$c = \lceil 0.3n \rceil$	$c = \lceil 0.1n \rceil$	$c = \lceil 0.2n \rceil$	$c = \lceil 0.3n \rceil$
H1: SBR	Obj.	4.359	4.752	4.773	4.857	5.286	5.308
	Time	0.00	0.01	0.01	0.01	0.01	0.02
H2: BS+GA	Obj.	4.636	4.781	4.784	5.207	5.326	5.327
	Time	0.04	0.08	0.08	0.07	0.14	0.15
BS+BB	Obj.	4.636	4.782	4.785	5.207	5.326	5.327
	Time (BR1)	2439.13	4.32	0.40	1719.63	1.21	**0.05**
	Time (BR2)	**289.25**	**0.24**	**0.16**	**155.20**	**0.42**	0.07
	#Conv.	19	20	20	20	20	20
	#Imp. vs. H1	20	19	16	20	20	18
	#Imp. vs. H2	1	1	2	2	4	4
	Impr. (%) vs. H1	6.42	0.63	0.27	7.21	0.77	0.37
	Impr. (%) vs. H2	0.00	0.01	0.02	0.01	0.01	0.01
	#Iter.	2.8	2.3	2.0	2.9	2.0	2.0

Table 6
Results on instances with $n = 300$.

$n = 300$		$m = 5$			$m = 10$		
		$c = \lceil 0.1n \rceil$	$c = \lceil 0.2n \rceil$	$c = \lceil 0.3n \rceil$	$c = \lceil 0.1n \rceil$	$c = \lceil 0.2n \rceil$	$c = \lceil 0.3n \rceil$
H1: SBR	Obj.	4.799	5.095	5.100	5.377	5.648	5.649
	Time	0.01	0.02	0.03	0.02	0.04	0.07
H2: BS+GA	Obj.	5.019	5.111	5.111	5.604	5.670	5.670
	Time	0.10	0.17	0.18	0.18	0.31	0.32
BS+BB	Obj.	5.019	5.113	5.113	5.605	5.672	5.672
	Time (BR1)	3526.36	229.23	**1.36**	2802.56	**13.65**	**0.08**
	Time (BR2)	**864.56**	**185.25**	4.48	**422.98**	42.24	0.92
	#Conv.	18	19	20	18	20	20
	#Imp. vs. H1	20	20	20	20	20	20
	#Imp. vs. H2	2	4	4	3	3	3
	Impr. (%) vs. H1	4.58	0.36	0.26	4.26	0.44	0.41
	Impr. (%) vs. H2	0.01	0.05	0.05	0.01	0.03	0.03
	#Iter.	2.5	2.0	2.0	2.5	2.0	2.1

the benefit of VFO, we are able to optimally solve instances with $n \leq 500$ within 10 s under the relatively relaxed cardinality constraint.

We now turn to discussing why BR1 outperforms BR2 in certain scenarios. To facilitate this explanation, we first present Table 9, which displays the CPU times of the exact method using both branching rules with and without VFO for scenarios where BR1 outperforms BR2. It can be observed that, in some scenarios, BR2 may still achieve lower CPU times than BR1 when VFO is not implemented. However, this advantage is less pronounced compared to the cases where the cardinality constraint is relatively strict and the total number of products is relatively large. We hypothesize that the effectiveness of the greedy mechanism employed in BR2 diminishes as the size of the optimal assortment increases. Furthermore, with the introduction of VFO, the reduction in CPU time is more substantial for BR1 than for BR2 across all listed scenarios. This suggests that BR1 may be more compatible with VFO, particularly in cases involving a large product set and relaxed cardinality constraints. These factors collectively contribute to the superior performance of BR1 in these scenarios.

Moreover, an analysis of the average optimal objective values across scenarios reveals that the expected revenue gains from increasing c from $\lceil 0.1n \rceil$ to $\lceil 0.2n \rceil$ generally surpass those achieved by further increasing c from $\lceil 0.2n \rceil$ to $\lceil 0.3n \rceil$. This indicates that the revenue improvement resulting from an equivalent increase in capacity may vary depending on the initial cardinality constraint. Retailers should carefully weigh the costs and benefits of expanding their assortment capacity based on their specific circumstances before making such decisions.

Table 7
Results on instances with $n = 500$.

$n = 500$		$m = 5$			$m = 10$		
		$c = \lceil 0.1n \rceil$	$c = \lceil 0.2n \rceil$	$c = \lceil 0.3n \rceil$	$c = \lceil 0.1n \rceil$	$c = \lceil 0.2n \rceil$	$c = \lceil 0.3n \rceil$
H1: SBR	Obj.	5.267	5.527	5.528	5.868	6.041	6.041
	Time	0.03	0.07	0.12	0.05	0.15	0.26
H2: BS+GA	Obj.	5.490	5.544	5.544	6.028	6.062	6.062
	Time	0.29	0.47	0.48	0.51	0.78	0.76
BS+BB	Obj.	5.490	5.545	5.545	6.030	6.064	6.064
	Time (BR1)	3600.00	1035.81	**10.28**	3545.03	**225.08**	**0.34**
	Time (BR2)	**2485.52**	**740.66**	200.17	**1942.94**	317.70	181.70
	#Conv.	8	16	20	11	19	20
	#Imp. vs. H1	20	20	20	20	20	20
	#Imp. vs. H2	3	5	5	6	10	10
	Impr. (%) vs. H1	4.25	0.33	0.32	2.77	0.37	0.37
	Impr. (%) vs. H2	0.00	0.01	0.01	0.03	0.03	0.03
	#Iter.	1.4	1.9	2.0	1.8	2.1	2.4

Table 8
Comparison of computation times for the exact method with and without variable-fixing operations.

		$m = 5$			$m = 10$		
		$c = \lceil 0.1n \rceil$	$c = \lceil 0.2n \rceil$	$c = \lceil 0.3n \rceil$	$c = \lceil 0.1n \rceil$	$c = \lceil 0.2n \rceil$	$c = \lceil 0.3n \rceil$
$n = 100$	Time (with VFO)	0.94	0.12	0.07	0.95	0.07	0.03
	Time (without VFO)	0.93	0.12	0.08	0.95	0.07	**0.05**
$n = 150$	Time (with VFO)	5.64	0.09	0.06	5.13	0.08	0.03
	Time (without VFO)	5.65	0.09	0.08	5.08	0.08	**0.08**
$n = 200$	Time (with VFO)	289.25	0.24	0.16	155.20	0.42	0.05
	Time (without VFO)	287.52	0.24	**0.23**	153.08	0.42	**0.79**
$n = 300$	Time (with VFO)	864.56	185.25	1.36	422.98	13.65	0.08
	Time (without VFO)	872.71	185.12	**187.02**	422.44	**42.52**	**43.26**
$n = 500$	Time (with VFO)	2485.52	740.66	10.28	1942.94	225.08	0.34
	Time (without VFO)	2487.89	741.24	**1043.03**	1947.18	**301.54**	**304.56**

Table 9
Computation times of the exact method under different settings of VFO and branching rules.

		Without VFO	With VFO	Decreasing ratio
$n = 200, m = 10, c = \lceil 0.3n \rceil$	Time (BR1)	0.79	0.05	**93.7%**
	Time (BR2)	0.40	0.07	82.5%
$n = 300, m = 5, c = \lceil 0.3n \rceil$	Time (BR1)	187.02	1.36	**99.3%**
	Time (BR2)	186.70	4.48	97.6%
$n = 300, m = 10, c = \lceil 0.2n \rceil$	Time (BR1)	42.52	13.65	**67.9%**
	Time (BR2)	42.01	42.24	−0.5%
$n = 300, m = 10, c = \lceil 0.3n \rceil$	Time (BR1)	43.26	0.08	**99.8%**
	Time (BR2)	41.28	0.92	97.8%
$n = 500, m = 5, c = \lceil 0.3n \rceil$	Time (BR1)	1043.03	10.28	**99.0%**
	Time (BR2)	740.93	200.17	73.0%
$n = 500, m = 10, c = \lceil 0.2n \rceil$	Time (BR1)	301.54	225.08	**25.4%**
	Time (BR2)	344.47	317.70	7.8%
$n = 500, m = 10, c = \lceil 0.3n \rceil$	Time (BR1)	304.56	0.34	**99.9%**
	Time (BR2)	346.96	181.70	47.6%

The algorithms proposed in this study can assist retailers in making more informed decisions by providing optimal or near-optimal expected revenue estimates under various possible cardinality constraints.

Finally, we summarize the key findings from the numerical experiments as follows:

- The exact algorithm, BS+BB, is capable of optimally solving all test instances with $n \leq 150$ and the majority of instances ($\geq$90%) with $n \leq 300$ within a one-hour time limit. This demonstrates the exact method's significantly greater efficiency in solving the CAOP-CNL compared to the full enumeration method.
- The choice of branching rules and the application of variable-fixing operations can have a considerable impact on the computation time of the exact algorithm. When the cardinality constraint is relatively strict or when the number of products is moderate, employing BR2 results in significantly lower CPU time compared to BR1. However, in cases where the product set is large and the cardinality constraint is relatively relaxed, the combined use of BR1 and VFO can markedly reduce computation time compared to using BR2.
- The second heuristic algorithm, BS+GA, which integrates the binary search framework with a greedy algorithm, consistently outperforms the other heuristic across all scenarios. It achieves optimal solutions in most instances, with an average optimality gap no larger than 0.2%. This suggest that for time-sensitive assortment optimization needs, BS+GA can provide prompt solutions while maintaining high solution quality.

- As the cardinality constraint becomes more relaxed, the performance difference between the exact method and the first heuristic diminishes, indicating that the structure of the optimal assortment increasingly aligns with the sorted-by-revenue assortment.

6. Conclusions and future directions

In this study, we developed a non-trivial exact algorithm and efficient heuristics for solving the CAOP-CNL. The exact solution method integrates a binary search framework with a customized B&B algorithm tailored for the NP-hard parameterized problem. The B&B algorithm incorporates a novel variable-fixing mechanism, branching rule and upper bound generation strategy, all leveraging the unique structure of the parameterized problem's objective function. To address extremely large instances beyond the capacity of the exact method, we introduced two polynomial-time heuristic algorithms employing different solution strategies. Numerical results indicate that our exact method can efficiently handle moderately large instances within a reasonable time limit. Specifically, the exact algorithm optimally solved all test instances with up to 150 products and more than 90% of instances with up to 300 products within a one-hour time limit. Under relatively relaxed cardinality constraints, the variable-fixing mechanism enabled the exact method to manage larger instances within approximately 10 s. Using the exact method as a benchmark, we observed that the most effective heuristic achieved optimal solutions for the majority of test instances, with an average optimality gap not exceeding 0.2%.

This study provides retailers with an effective tool for determining the optimal assortment when the number of offered products is constrained. For scenarios with a small dataset or lenient time constraints, retailers can use the exact method to obtain the optimal solution. Alternatively, for larger datasets or more time-sensitive situations, heuristics can be employed to derive a nearly optimal assortment decision in polynomial time. The algorithms presented in this study also aid retailers in making decisions about expanding or reducing current display space by evaluating the costs and benefits under various cardinality constraints.

There are several future directions worth further investigation based on this study. First, this study may be modified to solve the joint price and assortment optimization under the CNL model following the manner in Gallego and Topaloglu (2014). Second, it is worth considering to develop exact or approximation solution methods for assortment optimization problem under the CNL model with more general constraints (e.g. capacity, partition constraints). Third, when the products' preference weights are not known in advance, the dynamic assortment optimization problem under the CNL model worth further exploration.

CRediT authorship contribution statement

Le Zhang: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Shadi Sharif Azadeh:** Writing – review & editing, Supervision, Investigation, Formal analysis. **Hai Jiang:** Writing – review & editing, Supervision, Resources, Project administration, Methodology, Investigation, Funding acquisition, Formal analysis, Conceptualization.

Acknowledgments

This research is supported in part by the National Natural Science Foundation of China (grant numbers 72361137005 and 72431002) and by Project SINERGI (project code - C61A61) from JPI-ERANET and NWO.

Appendix A. Proof of Lemmas and Propositions

A.1. Proof of Lemma 2

Since $\frac{b}{a^{1-\gamma}} / \frac{b+rv}{(a+v)^{1-\gamma}} = \frac{b}{b+rv} \cdot (\frac{a+v}{a})^{1-\gamma} = (\frac{b}{b+rv})^{\gamma} \cdot (\frac{ab+bv}{ab+arv})^{1-\gamma} < 1$ as $ar \geq b$, we have $z' = \frac{b+rv}{(a+v)^{1-\gamma}} > z = \frac{b}{a^{1-\gamma}}$. □

A.2. Proof of Lemma 3

We show Lemma 3 under two possible scenarios based on the value of r_1:

(*i*) $r_1 > \frac{b}{a}$ (Scenario 1). Since $r_2 \geq r_1 > \frac{b+r_1v_1}{a+v_1}$, $z' = \frac{b+r_1v_1+r_2v_2}{(a+v_1+v_2)^{1-\gamma}} > \frac{b+r_1v_1}{a+v_1} = z$ can be easily derived through Lemma 2.

(*ii*) $r_1 \leq \frac{b}{a}$ (Scenario 2). Define $f(x) = \frac{b+r_1x}{(a+x)^{1-\gamma}}$, then $f'(x) = (a+x)^{\gamma-1}\left[r_1 - (1-\gamma)\frac{b+r_1x}{a+x}\right]$. As $r_1 \leq \frac{b}{a}$, if we define $h(x) = r_1 - (1-\gamma)\frac{b+r_1x}{a+x}$, then $h(x)$ is a non-decreasing function with respect to x. Since $f(v_1) \geq f(0)$, we must have $f'(v_1) \geq 0$, which is equivalent to $h(v_1) \geq 0$. If $h(v_1) < 0$, given its non-decreasing property, we have $h(x) < 0$ for $x \in [0, v_1]$, thereby $f'(x) < 0$ for $x \in [0, v_1]$, which leads to a contradiction. This implies that $r_1 \geq (1-\gamma)\frac{b+r_1v_1}{a+v_1}$.

Now define $g(x) = \frac{b+r_1v_1+r_2x}{(a+v_1+x)^{1-\gamma}}$, $g'(x) = (a+v_1+x)^{\gamma-1}\left[r_2 - (1-\gamma)\frac{b+r_1v_1+r_2x}{a+v_1+x}\right]$.

If $r_2 \leq \frac{b+r_1v_1}{a+v_1}$, as $r_2 \geq r_1$, when $x \geq 0$, we have $r_2 - (1-\gamma)\frac{b+r_1v_1+r_2x}{a+v_1+x} \geq r_1 - (1-\gamma)\frac{b+r_1v_1+r_2x}{a+v_1+x} \geq r_1 - (1-\gamma)\frac{b+r_1v_1}{a+v_1} \geq 0$. This implies that $g'(x) \geq 0$ for $\forall x \geq 0$, thus $z' = g(v_2) \geq g(0) = z$.

If $r_2 > \frac{b+r_1v_1}{a+v_1}$, then according to Lemma 2, we can directly obtain $z' = \frac{b+r_1v_1+r_2v_2}{(a+v_1+v_2)^{1-\gamma}} > z$. □

A.3. Proof of Lemma 4

First we have $\frac{b+r_1v_1}{(a+v_1)^{1-\gamma}} - \frac{b}{a^{1-\gamma}} = \frac{a^{1-\gamma}(b+r_1v_1) - b(a+v_1)^{1-\gamma}}{(a+v_1)^{1-\gamma}a^{1-\gamma}} \geq 0$. As the denominator is always positive, after rearranging terms in the numerator, we can obtain $r_1v_1 \geq \frac{b}{a^{1-\gamma}}\left[(a+v_1)^{1-\gamma} - a^{1-\gamma}\right]$. As $\gamma \in (0,1]$, we have $(a+v_1)^{1-\gamma} - a^{1-\gamma} \geq (a+v_1+v_2)^{1-\gamma} - (a+v_2)^{1-\gamma}$, which leads to $r_1v_1 \geq \frac{b}{a^{1-\gamma}}\left[(a+v_1+v_2)^{1-\gamma} - (a+v_2)^{1-\gamma}\right]$.

If $\frac{b+r_2v_2}{(a+v_2)^{1-\gamma}} \leq \frac{b}{a^{1-\gamma}}$, then $r_1v_1 \geq \frac{b+r_2v_2}{(a+v_2)^{1-\gamma}}\left[(a+v_1+v_2)^{1-\gamma} - (a+v_2)^{1-\gamma}\right]$. After rearranging terms, we can obtain that $z' = \frac{b+r_1v_1+r_2v_2}{(a+v_1+v_2)^{1-\gamma}} \geq \frac{b+r_2v_2}{(a+v_2)^{1-\gamma}} = z$.

If $\frac{b+r_2v_2}{(a+v_2)^{1-\gamma}} > \frac{b}{a^{1-\gamma}}$, as $r_2 \leq r_1$, we can directly obtain $z' = \frac{b+r_1v_1+r_2v_2}{(a+v_1+v_2)^{1-\gamma}} \geq \frac{b+r_2v_2}{(a+v_2)^{1-\gamma}} = z$ by Lemma 3. □

A.4. Proof of Proposition 3

Suppose $\frac{b_{it}+(r_s-z)v_{is}}{(a_{it}+v_{is})^{1-\gamma_i}} \geq \frac{b_{it}}{a_{it}^{1-\gamma_i}}$, for $\forall i \in \{k \in M | v_{ks} > 0, a_{kt} > 0\}$ and there is an optimal solution $\boldsymbol{x}^*$ to CNLAPP-t where $x_s^* = 0$. We use $L = \{l \in \bar{S}^t | x_l^* = 1\}$ to denote the set of products provided in the optimal assortment $\boldsymbol{x}^*$. Since $|\bar{S}^t| \leq c_t$, we must have $\|\boldsymbol{x}^*\|_1 = |L| \leq c_t - 1$. And the optimal objective value can be written as $\sum_{i\in M} \frac{b_{it}+\sum_{j\in L}(r_j-z)v_{ij}}{(a_{it}+\sum_{j\in L} v_{ij})^{1-\gamma_i}}$. If we set another solution $\boldsymbol{x}'$ such that $x_s' = 1$ whereas the values of other elements are the same as $\boldsymbol{x}^*$. Then its corresponding objective value is $\sum_{i\in M} \frac{b_{it}+\sum_{j\in L}(r_j-z)v_{ij}+(r_s-z)v_{is}}{(a_{it}+\sum_{j\in L} v_{ij}+v_{is})^{1-\gamma_i}}$. We can also guarantee the feasibility of $\boldsymbol{x}'$ as $\|\boldsymbol{x}'\|_1 = \|\boldsymbol{x}^*\|_1 + 1 \leq c_t$.

For $\forall i \in \{k \in M | v_{ks} = 0\}$, $\frac{b_{it}+\sum_{j\in L}(r_j-z)v_{ij}+(r_s-z)v_{is}}{(a_{it}+\sum_{j\in L} v_{ij}+v_{is})^{1-\gamma_i}} = \frac{b_{it}+\sum_{j\in L}(r_j-z)v_{ij}}{(a_{it}+\sum_{j\in L} v_{ij})^{1-\gamma_i}}$.

For $\forall i \in \{k \in M | v_{ks} > 0, a_{kt} = 0\}$, denote $v = \sum_{j\in L} v_{ij} \geq 0$, if $v = 0$, we have $\frac{(r_s-z)v_{is}}{v_{is}^{1-\gamma_i}} \geq 0$. If $v > 0$, set $r = \frac{\sum_{j\in L}(r_j-z)v_{ij}}{v} + z$. Since s is the product with the largest revenue in $\bar{S}^t$, we have $r_s - z \geq r - z > 0$. According to Lemma 2, we can obtain $\frac{\sum_{j\in L}(r_j-z)v_{ij}+(r_s-z)v_{is}}{(\sum_{j\in L} v_{ij}+v_{is})^{1-\gamma_i}} \geq$

$\frac{\sum_{j\in L}(r_j-z)v_{ij}}{(\sum_{j\in L} v_{ij})^{1-\gamma_i}}$.

For $\forall i \in \{k \in M | v_{ks} > 0, a_{kt} > 0\}$, denote $v = \sum_{j\in L} v_{ij} \geq 0$, if $v = 0$, we can directly obtain $\frac{b_{it}+(r_s-z)v_{is}}{(a_{it}+v_{is})^{1-\gamma_i}} \geq \frac{b_{it}}{a_{it}^{1-\gamma_i}}$ from the supposition. If $v > 0$, set $r = \frac{\sum_{j\in L}(r_j-z)v_{ij}}{v} + z$. Since s is the product with the largest revenue in $\bar{S}^t$, we have $r_s - z \geq r - z > 0$. According to Lemma 4, we can obtain $\frac{b_{it}+\sum_{j\in L}(r_j-z)v_{ij}+(r_s-z)v_{is}}{(a_{it}+\sum_{j\in L} v_{ij}+v_{is})^{1-\gamma_i}} \geq \frac{b_{it}+\sum_{j\in L}(r_j-z)v_{ij}}{(a_{it}+\sum_{j\in L} v_{ij})^{1-\gamma_i}}$.

To summarize, the objective value of $\boldsymbol{x}'$ is no less than that of $\boldsymbol{x}^*$, so $\boldsymbol{x}'$ must also be an optimal solution to CNLAPP-t. □

A.5. Proof of Proposition 4

If the dual optimal $\boldsymbol{\mu}^* \neq \mathbf{0}$, denote a subset $S = \{j \in \bar{S}^t | \mu_j^* \neq 0\} \subseteq N$. Let $\{\boldsymbol{x}_1^*, \boldsymbol{x}_2^*, \ldots, \boldsymbol{x}_m^*, \boldsymbol{y}^*\}$ be the optimal solution to $g(\boldsymbol{\mu}^*)$, we have $g(\boldsymbol{\mu}^*) = \sum_{i\in M} V_i(\boldsymbol{x}_i^*)^{\gamma_i}[R_i(\boldsymbol{x}_i^*) - z] + \sum_{j\in S} \mu_j^*(\sum_{i\in M} x_{ij}^* - m y_j^*)$. It can be readily observed that the second part of $g(\boldsymbol{\mu}^*)$, written as $\sum_{j\in S} \mu_j^*(\sum_{i\in M} x_{ij}^* - m y_j^*)$, is always non-negative. This follows from the fact that y_j^* is certainly set to 1 when $\mu_j^* < 0$ and must be assigned to 0 when $\mu_j^* > 0$.

Let $(\boldsymbol{x}_1', \boldsymbol{x}_2', \ldots, \boldsymbol{x}_n')$ be an optimal solution to $g(\mathbf{0})$ (the value of $\boldsymbol{y}'$ need not to be determined). Since $(\boldsymbol{x}_1', \boldsymbol{x}_2', \ldots, \boldsymbol{x}_m', \boldsymbol{y}^*)$ is also feasible to $g(\boldsymbol{\mu}^*)$, we have $g(\boldsymbol{\mu}^*) \geq \sum_{i\in M} V_i(\boldsymbol{x}_i')^{\gamma_i}[R_i(\boldsymbol{x}_i') - z] + \sum_{j\in S} \mu_j^*(\sum_{i\in M} x_{ij}' - m y_j^*) \geq g(\mathbf{0}) = \sum_{i\in M} V_i(\boldsymbol{x}_i')^{\gamma_i}[R_i(\boldsymbol{x}_i') - z] \geq g(\boldsymbol{\mu}^*)$. There must be $g(\mathbf{0}) = g(\boldsymbol{\mu}^*)$, or else there is a contradiction. In either case, the proposition holds. □

A.6. Proof of Lemma 5

We can solve d-CNLAPP-t by tackling m nest-specific maximization problem. For each nest i, the corresponding sub-problem is

$$\max_{\boldsymbol{x}_i\in\{0,1\}^{|\bar{S}^t|}:\|\boldsymbol{x}_i\|_1\leq c_t} \left\{ \frac{b_{it} + \sum_{j\in\bar{S}^t}(r_j - z)v_{ij}x_{ij}}{(a_{it} + \sum_{j\in\bar{S}^t} v_{ij}x_{ij})^{1-\gamma_i}} \right\}. \tag{A.1}$$

We prove that problem (A.1) can be solve optimally in polynomial time. Now we first rewrite problem (A.1) into its compact form. That is $\max_{\boldsymbol{x}_i\in\{0,1\}^{|\bar{S}^t|}:\|\boldsymbol{x}_i\|_1\leq c_t} \left\{V_i(\boldsymbol{x}_i)^{\gamma_i}\left[R_i(\boldsymbol{x}_i) - z\right]\right\}$, where $V_i(\boldsymbol{x}_i) = a_{it} + \sum_{j\in\bar{S}^t} v_{ij}x_{ij}$ and $R_i(\boldsymbol{x}_i) = \frac{b_{it}+za_{it}+\sum_{j\in\bar{S}^t} r_j v_{ij}x_{ij}}{a_{it}+\sum_{j\in\bar{S}^t} v_{ij}x_{ij}}$. Refer to Gallego and Topaloglu (2014), we can obtain the following lemma:

Lemma 6. *let $\boldsymbol{x}_i^*$ be an optimal solution to problem (A.1) and set $u_i = \max\{z, \gamma_i z + (1-\gamma_i)R_i(\boldsymbol{x}_i^*)\}$, if there is an optimal solution $\hat{\boldsymbol{x}}_i$ to the maximization problem*

$$\max_{\substack{\boldsymbol{x}_i\in\{0,1\}^{|\bar{S}^t|}:\\ \|\boldsymbol{x}_i\|_1\leq c_t}} \left\{V_i(\boldsymbol{x}_i)\left[R_i(\boldsymbol{x}_i) - u_i\right]\right\}$$

$$= \max_{\substack{\boldsymbol{x}_i\in\{0,1\}^{|\bar{S}^t|}:\\ \|\boldsymbol{x}_i\|_1\leq c_t}} \left\{ b_{it} + (z-u_i)a_{it} + \sum_{j\in\bar{S}^t}(r_j - u_i)v_{ij}x_{ij} \right\}, \tag{A.2}$$

then $\hat{\boldsymbol{x}}_i$ is also an optimal solution to problem (A.1).

Proof. For notational brevity, we denote $\hat{V}_i = V_i(\hat{\boldsymbol{x}}_i)$, $V_i^* = V_i(\boldsymbol{x}_i^*)$, $\hat{R}_i = R_i(\hat{\boldsymbol{x}}_i)$ and $R_i^* = R_i(\boldsymbol{x}_i^*)$. If $R_i^* > z$, then $u_i = \gamma_i z + (1-\gamma_i)R_i^*$. Since $\boldsymbol{x}_i^*$ is a feasible but not necessarily optimal solution to problem (A.2), we have $\hat{V}_i(\hat{R}_i - \gamma_i z - (1-\gamma_i)R_i^*) \geq V_i^*(R_i^* - \gamma_i z - (1-\gamma_i)R_i^*) = \gamma_i V_i^*(R_i^* - z)$. After rearranging terms, we can obtain

$$\hat{V}_i(\hat{R}_i - z) \geq (\gamma_i V_i^* + (1-\gamma_i)\hat{V}_i)(R_i^* - z). \tag{A.3}$$

Since $\gamma_i \in (0,1]$, u^{γ_i} is a concave function of $u \in R^+$ and satisfies the subgradient inequality $u^{\gamma_i} \leq \hat{u}^{\gamma_i} + \gamma_i\hat{u}^{\gamma_i-1}(u-\hat{u}) = \hat{u}^{\gamma_i-1}(\gamma_i u + (1-\gamma_i)\hat{u}), \forall u, \hat{u} \in R^+$. Set $u = V_i^*$, $\hat{u} = \hat{V}_i$, we have $(V_i^*)^{\gamma_i} \leq \hat{V}_i^{\gamma_i-1}(\gamma_i V_i^* + (1-\gamma_i)\hat{V}_i)$. As $R_i^* > z \geq 0$, it follows that $V_i^* > 0$, and thus $\hat{V}_i > 0$ according to the inequality above. If we multiply both sides of (A.3) by $\hat{V}_i^{\gamma_i-1}$, then we obtain

$$\hat{V}_i^{\gamma_i}(\hat{R}_i - z) \geq \hat{V}_i^{\gamma_i-1}(\gamma_i V_i^* + (1-\gamma_i)\hat{V}_i)(R_i^* - z) \geq (V_i^*)^{\gamma_i}(R_i^* - z). \tag{A.4}$$

Then $\hat{\boldsymbol{x}}_i$ must also be an optimal solution to problem (A.1).

If $R_i^* \leq z$, then $u_i = z$. It is easy to verify that $\hat{V}_i(\hat{R}_i - z) \geq V_i(\mathbf{0})(R_i(\mathbf{0}) - z) = 0$. This inequality implies that either $\hat{V}_i = 0$ or $\hat{V}_i > 0$ and $\hat{R}_i - z \geq 0$. In either case, since $R_i^* \leq z$, we can obtain that $\hat{V}_i^{\gamma_i}(\hat{R}_i - z) \geq 0 \geq (V_i^*)^{\gamma_i}(R_i^* - z)$. This also indicates that $\hat{\boldsymbol{x}}_i$ must also be an optimal solution to problem (A.1). □

According to Lemma 6, as long as we know the value of u_i, we can determine the optimal solution to problem (A.1) by solving problem (A.2) easily through a greedy algorithm. However, directly computing u_i requires knowing $\boldsymbol{x}_i^*$ in advance, which is impracticable since our objective is to find $\boldsymbol{x}_i^*$. To circumvent this challenge, we can first derive the range of all possible values of u_i, that is $[z, \max\{b_{it}/a_{it} + z, \max_{j\in\bar{S}^t} r_j\}]$ when $a_{it} > 0$ or $[z, \max_{j\in\bar{S}^t} r_j]$ when $a_{it} = 0$. Then we can come up with a collection of assortments S_i that obtains optimal solutions to each of the possible u_i. It can be shown that S_i includes $O(|\bar{S}^t|^2)$ solutions and must contain an optimal solution to problem (A.1). Since deriving a candidate assortment and evaluate its performance in problem (A.1) consumes a time complexity of $O(|\bar{S}^t|)$. Then the total time complexity analysis of solving m problem (A.1) is $O(m|\bar{S}^t|^3)$. □

Appendix B. Pseudo-codes of procedures at B&B tree nodes

B.1. Procedures at root node 0

Algorithm 6 Procedures at node 0

Require: z, instance of the CAOP-CNL u.

$obj_{max} \leftarrow 0$.
$S_0^0 \leftarrow \{j \in N | r_j \leq z\}$.
$c_0 \leftarrow c$.
$x_j^* \leftarrow 0$ for $\forall j \in S_0^0$.
if $S_0^0 = N$ **then**
  $S_1^0 \leftarrow \emptyset$, $\bar{S}^0 \leftarrow \emptyset$.
  $\boldsymbol{x}^* \leftarrow \mathbf{0}$.
  Exit (the parameterized problem is solved).
else
  $S_1^0 \leftarrow \emptyset, \bar{S}^0 \leftarrow N \setminus S_0^0$.
  $a_{i0} \leftarrow 0, b_{i0} \leftarrow 0, \forall i \in M$.
  $s \leftarrow \arg\max_{j\in\bar{S}^0} r_j$.
  while $0 < |\bar{S}^0| \leq c_0$ and $\frac{b_{i0}+(r_s-z)}{(a_{i0}+v_{is})^{1-\gamma_i}} \geq \frac{b_{i0}}{a_{i0}^{1-\gamma_i}}$ for $\forall i \in \{k \in M | v_{ks} > 0, a_{k0} > 0\}$ **do**
    $S_1^0 \leftarrow S_1^0 \cup \{s\}$, $\bar{S}^0 \leftarrow \bar{S}^0 \setminus \{s\}$.
    $a_{i0} \leftarrow a_{i0} + v_{is}$, $b_{i0} \leftarrow b_{i0} + (r_s - z)v_{is}, \forall i \in \{k \in M | v_{ks} > 0, a_{k0} > 0\}$.
    $x_s^* \leftarrow 1, c_0 \leftarrow c_0 - 1, s \leftarrow \arg\max_{j\in\bar{S}^0} r_j$.
  end while
  $obj_{max} \leftarrow \sum_{i\in M} \frac{b_{i0}}{a_{i0}^{1-\gamma_i}}$, record current node.
  if $|\bar{S}^0| > 0$ and $c_0 > 0$ **then**
    Continue with branching following the selected branching rule.
  else
    Stop without branching.
  end if
end if

Table C.10
Results on instances with $n = 50$ under varying overlapping rates.

$m = 5$, $n = 50$, $c = \lceil 0.1n \rceil$		$\delta = 1.0$	$\delta = 1.2$	$\delta = 1.5$	$\delta = 2.0$	$\delta = 2.5$	$\delta = 3.0$
H1: SBR	Obj.	3.028	3.035	2.918	2.745	2.698	2.671
	Time	0.00	0.00	0.00	0.00	0.00	0.00
H2: BS+GA	Obj.	3.302	3.275	3.238	2.978	2.946	2.843
	Time	0.00	0.00	0.00	0.00	0.00	0.00
BS+BB	Obj.	3.302	3.282	3.245	2.984	2.951	2.847
	Time	0.03	0.03	0.03	0.06	0.03	0.04
	#Opt.	20	20	20	20	20	20
	#Imp. vs. H1	19	20	20	19	20	20
	#Imp. vs. H2	0	4	3	5	5	5
	Impr. (%) vs. H1	9.20	8.34	11.43	8.78	9.53	6.72
	Impr. (%) vs. H2	0.00	0.20	0.21	0.20	0.14	0.14
	#Iter.	3.1	2.9	3.1	3.4	2.7	2.9

Table C.11
Results on instances with $n = 100$ under varying overlapping rates.

$m = 5$, $n = 100$, $c = \lceil 0.1n \rceil$		$\delta = 1.0$	$\delta = 1.2$	$\delta = 1.5$	$\delta = 2.0$	$\delta = 2.5$	$\delta = 3.0$
H1: SBR	Obj.	3.832	3.694	3.572	3.484	3.306	3.092
	Time	0.00	0.00	0.00	0.00	0.00	0.00
H2: BS+GA	Obj.	4.107	3.996	3.880	3.738	3.570	3.313
	Time	0.01	0.01	0.01	0.01	0.01	0.01
BS+BB	Obj.	4.107	3.999	3.882	3.739	3.574	3.313
	Time	0.47	0.94	1.28	2.35	1.34	4.12
	#Opt.	20	20	20	20	20	20
	#Imp. vs. H1	20	20	20	20	20	20
	#Imp. vs. H2	0	5	3	4	5	1
	Impr. (%) vs. H1	7.31	8.34	8.72	7.33	8.16	7.22
	Impr. (%) vs. H2	0.00	0.09	0.05	0.02	0.14	0.00
	#Iter.	3.2	2.9	3.4	3.1	2.7	2.6

Table C.12
Results on instances with $n = 150$ under varying overlapping rates.

$m = 5$, $n = 150$, $c = \lceil 0.1n \rceil$		$\delta = 1.0$	$\delta = 1.2$	$\delta = 1.5$	$\delta = 2.0$	$\delta = 2.5$	$\delta = 3.0$
H1: SBR	Obj.	4.168	4.093	3.996	3.874	3.617	3.513
	Time	0.00	0.00	0.00	0.00	0.00	0.00
H2: BS+GA	Obj.	4.463	4.403	4.317	4.149	3.915	3.747
	Time	0.02	0.02	0.02	0.02	0.03	0.03
BS+BB	Obj.	4.463	4.404	4.318	4.152	3.916	3.747
	Time	8.82	5.64	14.56	26.12	43.28	32.17
	#Opt.	20	20	20	20	20	20
	#Imp. vs. H1	20	20	20	20	20	20
	#Imp. vs. H2	0	1	4	3	3	0
	Impr. (%) vs. H1	7.17	7.65	8.12	7.25	8.26	6.67
	Impr. (%) vs. H2	0.00	0.01	0.04	0.08	0.02	0.00
	#Iter.	2.8	2.5	3.6	3.0	2.9	3.1

Table C.13
Results on instances with $n = 200$ under varying overlapping rates.

$m = 5$, $n = 200$, $c = \lceil 0.1n \rceil$		$\delta = 1.0$	$\delta = 1.2$	$\delta = 1.5$	$\delta = 2.0$	$\delta = 2.5$	$\delta = 3.0$
H1: SBR	Obj.	4.432	4.359	4.204	4.123	3.969	3.807
	Time	0.00	0.00	0.00	0.00	0.00	0.00
H2: BS+GA	Obj.	4.720	4.636	4.522	4.380	4.266	4.094
	Time	0.04	0.04	0.04	0.04	0.04	0.04
BS+BB	Obj.	4.722	4.636	4.523	4.381	4.266	4.095
	Time	275.13	289.25	306.30	491.13	1103.94	996.89
	#Opt.	19	19	19	18	17	18
	#Imp. vs. H1	20	20	20	20	20	20
	#Imp. vs. H2	5	1	4	4	3	2
	Impr. (%) vs. H1	6.56	6.42	7.64	6.30	7.53	7.52
	Impr. (%) vs. H2	0.04	0.00	0.01	0.03	0.02	0.03
	#Iter.	3.1	2.8	3.0	3.1	3.2	2.9

Algorithm 7 Procedures at node $t \neq 0$

Require: z, obj_{max}, instance of the CAOP-CNL u.
1: Inherit a_{it}, b_{it}, c_t, S_0^t, S_1^t and $\bar{S}^t$ from the predecessor node.
2: $s \leftarrow \arg\max_{j \in \bar{S}^t} r_j$.
3: **while** $0 < |\bar{S}^t| \leq c_t$ and $\frac{b_{it}+(r_s-z)}{(a_{it}+v_{is})^{1-\gamma_i}} \geq \frac{b_{it}}{a_{it}^{1-\gamma_i}}$ for $\forall i \in \{k \in M | v_{ks} > 0, a_{kt} > 0\}$ **do**
4: $S_1^t \leftarrow S_1^t \cup \{s\}$, $\bar{S}^t \leftarrow \bar{S}^t \setminus \{s\}$.
5: $a_{it} \leftarrow a_{it} + v_{is}$, $b_{it} \leftarrow b_{it} + (r_s - z)v_{is}, \forall i \in \{k \in M | v_{ks} > 0, a_{kt} > 0\}$.
6: $x_s^* \leftarrow 1, c_t \leftarrow c_t - 1, s \leftarrow \arg\max_{j \in \bar{S}^t} r_j$.
7: **end while**
8: $obj_{cur} \leftarrow \sum_{i \in M} \frac{b_{it}}{a_{it}^{1-\gamma_i}}$.
9: **if** $obj_{cur} > obj_{max}$ **then**
10: $obj_{max} \leftarrow obj_{cur}$, record current node.
11: **end if**
12: $bound \leftarrow$ solve d-CNLAPP-t.
13: **if** $bound > obj_{max}$ and $|\bar{S}^t| > 0$ and $c_t > 0$ **then**
14: Continue with branching following the selected branching rule.
15: **else**
16: Stop without branching.
17: **end if**

B.2. Procedures at non-root node t

B.3. Improve the initial upper bound of expected revenue

Following the same idea of generating the upper bound of CNLAPP-t, we can relaxed the original problem (3) into the following problem

$$\max \quad \frac{\sum_{i \in M} V_i(\boldsymbol{x}_i)^{\gamma_i} R_i(\boldsymbol{x}_i)}{v_0 + \sum_{l \in M} V_l(\boldsymbol{x}_l)^{\gamma_l}} \tag{B.1a}$$

$$\text{s.t.} \quad \sum_{j \in N} x_{ij} \leq c, \quad i \in M \tag{B.1b}$$

$$\boldsymbol{x}_i = (x_{i1}, x_{i2}, \ldots, x_{in}) \in \{0,1\}^n, \quad i \in M. \tag{B.1c}$$

Here the original decision variable vector $\boldsymbol{x}$ is expanded into a series of nest-specific variable vectors $\boldsymbol{x}_i$, which eliminate the nest-overlapping effect. This problem is known to be the assortment optimization problem under the NL model with cardinality constraint imposed on each nest, which is shown to be polynomial-time solvable by Gallego and Topaloglu (2014). In this paper, we solve it in a slightly different manner from the literature. Specifically, we employ a similar binary search framework as shown in Section 3.1 to find the fixed point of a function $D(z) = \sum_{i \in M} \max_{\boldsymbol{x}_i \in \{0,1\}^n : |\boldsymbol{x}_i| \leq c} \left\{ V_i(\boldsymbol{x}_i)^{\gamma_i} \left[R_i(\boldsymbol{x}_i) - z \right] \right\}$. The initial upper bound is computed by Algorithm 1 and the lower bound is obtained from the heuristic shown in Section 4. Since $D(z)$ can be optimally solved in polynomial times as shown in Appendix A.6, we can easily obtain the optimal solution of problem (B.1) to derive a tighter upper bound of the original problem.

Appendix C. Impact of nest overlapping rate

A significant feature that enhances the flexibility of the CNL model, despite increasing estimation complexity compared to the MNL and NL models, is its ability to assign each product to multiple nests. In our numerical experiments, we control this feature using the nest overlapping rate δ, which dictates the average number of nests to which a product belongs. This section aims to assess the impact of this rate on the performance of our algorithms. To achieve this, we generate test instances with δ varying in $\{1.0, 1.5, 2.0, 2.5, 3.0\}$. The numerical results of our algorithms on instances with $m = 5$ and $n \in \{50, 100, 150, 200\}$ under a cardinality constraint $c = \lceil 0.1n \rceil$, a cardinality level where the exact algorithm exhibits the highest computation time in our main numerical experiments, are reported in Tables C.10 to C.13.

The results indicate that the computation time of heuristic algorithms is stable across varying values of overlapping rate. The objective value improvement of the exact method over each heuristic also remains consistent within a specific range. Conversely, the computation time of the exact algorithm increases significantly with the growth of the overlapping rate δ, yet it remains within a manageable magnitude. For instance, when the overlapping rate increases from 1.2 to 3.0, the optimal solution for all instances with $n \leq 150$ can still be obtained within a one-hour time limit.

References

Akchen, Y.-C., & Mišić, V. V. (2023). Assortment optimization under the decision forest model. arXiv preprint arXiv:2103.14067.

Alfandari, L., Hassanzadeh, A., & Ljubić, I. (2021). An exact method for assortment optimization under the nested logit model. *European Journal of Operational Research, 291*(3), 830–845.

Beine, M. A., Bierlaire, M., & Docquier, F. (2021). *New York, Abu Dhabi, London or stay at home? using a cross-nested logit model to identify complex substitution patterns in migration: IZA discussion paper.*

Ben-Akiva, M. E., Lerman, S. R., Lerman, S. R., et al. (1985). *Vol. 9, Discrete choice analysis: theory and application to travel demand*. MIT Press.

Bernstein, F., & Guo, Y. (2023). Managing customer search: Assortment planning for a subscription box service. *Manufacturing & Service Operations Management, 25*(5), 1623–1642.

Bertsimas, D., & Mišić, V. V. (2019). Exact first-choice product line optimization. *Operations Research, 67*(3), 651–670.

Bierlaire, M. (2006). A theoretical analysis of the cross-nested logit model. *Annals of Operations Research, 144*, 287–300.

Bront, J. J. M., Méndez-Díaz, I., & Vulcano, G. (2009). A column generation algorithm for choice-based network revenue management. *Operations Research, 57*(3), 769–784.

Chen, Y., He, T., Rong, Y., & Wang, Y. (2024). An integer programming approach for quick-commerce assortment planning. arXiv preprint arXiv:2405.02553.

Chen, R., & Jiang, H. (2020). Capacitated assortment and price optimization under the nested logit model. *Journal of Global Optimization, 77*(4), 895–918.

Chen, Y.-C., & Mišić, V. V. (2022). Decision forest: A nonparametric approach to modeling irrational choice. *Management Science, 68*(10), 7090–7111.

Chung, H., Ahn, H.-S., & Jasin, S. (2019). (Rescaled) multi-attempt approximation of choice model and its application to assortment optimization. *Production and Operations Management, 28*(2), 341–353.

Davis, J. M., Gallego, G., & Topaloglu, H. (2014). Assortment optimization under variants of the nested logit model. *Operations Research, 62*(2), 250–273.

Ding, C., Mishra, S., Lin, Y., & Xie, B. (2015). Cross-nested joint model of travel mode and departure time choice for urban commuting trips: Case study in Maryland–Washington, DC region. *Journal of Urban Planning and Development, 141*(4), Article 04014036.

Dinkelbach, W. (1967). On nonlinear fractional programming. *Management Science, 13*(7), 492–498.

Domarchi, C., & Cherchi, E. (2024). Role of car segment and fuel type in the choice of alternative fuel vehicles: A cross-nested logit model for the english market. *Applied Energy, 357*, Article 122451.

Drabas, T., & Wu, C.-L. (2013). Modelling air carrier choices with a segment specific cross nested logit model. *Journal of Air Transport Management, 32*, 8–16.

Ermagun, A., & Levinson, D. (2017). Public transit, active travel, and the journey to school: a cross-nested logit analysis. *Transportmetrica A: Transport Science, 13*(1), 24–37.

Fan, Y., Ding, J., Long, J., & Wu, J. (2024). Modeling and evaluating the travel behaviour in multimodal networks: A path-based unified equilibrium model and a tailored greedy solution algorithm. *Transportation Research Part A: Policy and Practice, 182*, Article 104032.

Feldman, J. B., & Topaloglu, H. (2015). Capacity constraints across nests in assortment optimization under the nested logit model. *Operations Research, 63*(4), 812–822.

Fosgerau, M., McFadden, D., & Bierlaire, M. (2013). Choice probability generating functions. *Journal of Choice Modelling, 8*, 1–18.

Gallego, G., & Topaloglu, H. (2014). Constrained assortment optimization for the nested logit model. *Management Science, 60*(10), 2583–2601.

Ghuge, R., Kwon, J., Nagarajan, V., & Sharma, A. (2022). Constrained assortment optimization under the paired combinatorial logit model. *Operations Research, 70*(2), 786–804.

Hess, S., Fowler, M., Adler, T., & Bahreinian, A. (2012). A joint model for vehicle type and fuel type choice: evidence from a cross-nested logit study. *Transportation, 39*(3), 593–625.

Huan, N., Hess, S., Yamamoto, T., & Yao, E. (2024). Modelling intermodal traveller behaviour in mega-city regions: simultaneous versus sequential estimation frameworks. *Transportation*, 1–36.

Kok, A. G., Fisher, M. L., & Vaidyanathan, R. (2008). Assortment planning: Review of literature and industry practice. *Retail Supply Chain Management*, *122*(1), 99–153.

Kunnumkal, S. (2023). New bounds for cardinality-constrained assortment optimization under the nested logit model. *Operations Research*.

Lai, X., & Bierlaire, M. (2015). Specification of the cross-nested logit model with sampling of alternatives for route choice models. *Transportation Research, Part B (Methodological)*, *80*, 220–234.

Le, C., & Mai, T. (2024). Constrained assortment optimization under the cross-nested logit model. *Production and Operations Management*, *33*(10), 2073–2090.

Lemp, J. D., Kockelman, K. M., & Damien, P. (2010). The continuous cross-nested logit model: Formulation and application for departure time choice. *Transportation Research, Part B (Methodological)*, *44*(5), 646–661.

Li, G., & Rusmevichientong, P. (2014). A greedy algorithm for the two-level nested logit model. *Operations Research Letters*, *42*(5), 319–324.

Lu, Q.-C., Zhang, J., Wu, L., & Rahman, A. S. (2016). Job and residential location changes responding to floods and cyclones: an analysis based on a cross-nested logit model. *Climatic Change*, *138*, 453–469.

Marzano, V., Papola, A., Simonelli, F., & Vitillo, R. (2013). A practically tractable expression of the covariances of the cross-nested logit model. *Transportation Research, Part B (Methodological)*, *57*, 1–11.

McFadden, D. (1974). Conditional logit analysis of qualitative choice behavior. *Frontiers in Econometrics*, 105–142.

McFadden, D. (1978). Modeling the choice of residential location. *Transportation Research Record*, (673), 72–77.

Méndez-Díaz, I., Miranda-Bront, J. J., Vulcano, G., & Zabala, P. (2014). A branch-and-cut algorithm for the latent-class logit assortment problem. *Discrete Applied Mathematics*, *164*, 246–263.

Mepparambath, R. M., Soh, Y. S., Jayaraman, V., Tan, H. E., & Ramli, M. A. (2023). A novel modelling approach of integrated taxi and transit mode and route choice using city-scale emerging mobility data. *Transportation Research Part A: Policy and Practice*, *170*, Article 103615.

Rusmevichientong, P., Shen, Z.-J. M., & Shmoys, D. B. (2009). A PTAS for capacitated sum-of-ratios optimization. *Operations Research Letters*, *37*(4), 230–238.

Rusmevichientong, P., Shen, Z.-J. M., & Shmoys, D. B. (2010). Dynamic assortment optimization with a multinomial logit choice model and capacity constraint. *Operations Research*, *58*(6), 1666–1680.

Rusmevichientong, P., Shmoys, D., Tong, C., & Topaloglu, H. (2014). Assortment optimization under the multinomial logit model with random choice parameters. *Production and Operations Management*, *23*(11), 2023–2039.

Rusmevichientong, P., & Topaloglu, H. (2012). Robust assortment optimization in revenue management under the multinomial logit choice model. *Operations Research*, *60*(4), 865–882.

Segev, D. (2022). Approximation schemes for capacity-constrained assortment optimization under the nested logit model. *Operations Research*, *70*(5), 2820–2836.

Small, K. A. (1987). A discrete choice model for ordered alternatives. *Econometrica*, 409–424.

Talluri, K., & Van Ryzin, G. (2004). Revenue management under a general discrete choice model of consumer behavior. *Management Science*, *50*(1), 15–33.

Train, K. E. (2009). *Discrete choice methods with simulation*. Cambridge University Press.

Van Ryzin, G., & Mahajan, S. (1999). On the relationship between inventory costs and variety benefits in retail assortments. *Management Science*, *45*(11), 1496–1509.

Vega, A., & Reynolds-Feighan, A. (2009). A methodological framework for the study of residential location and travel-to-work mode choice under central and suburban employment destination patterns. *Transportation Research Part A: Policy and Practice*, *43*(4), 401–419.

Vovsha, P. (1997). Application of cross-nested logit model to mode choice in Tel Aviv, Israel, metropolitan area. *Transportation Research Record*, *1607*(1), 6–15.

Wen, C.-H., & Koppelman, F. S. (2001). The generalized nested logit model. *Transportation Research, Part B (Methodological)*, *35*(7), 627–641.

Williams, H. C. (1977). On the formation of travel demand models and economic evaluation measures of user benefit. *Environment and Planning A*, *9*(3), 285–344.

Yang, C.-W., & Wang, H.-C. (2017). A comparison of flight routes in a dual-airport region using overlapping error components and a cross-nested structure in GEV models. *Transportation Research Part A: Policy and Practice*, *95*, 85–95.

Yang, L., Zheng, G., & Zhu, X. (2013). Cross-nested logit model for the joint choice of residential location, travel mode, and departure time. *Habitat International*, *38*, 157–166.

Zhang, L., Duan, P., & Jiang, H. (2024). Modeling joint row-and column-wise correlation in air passenger seat selection: A cross-nested logit approach. *Journal of Air Transport Management*, *114*, Article 102485.

Zhang, H., Rusmevichientong, P., & Topaloglu, H. (2020). Assortment optimization under the paired combinatorial logit model. *Operations Research*, *68*(3), 741–761.

Zhang, R., Yao, E., & Pan, L. (2019). Optimizing EV-based P&R subsidy policies for commuting corridor based on cross-nested logit model. *International Journal of Sustainable Transportation*, *13*(7), 461–478.