ARTICLE IN PRESS

Discrete Applied Mathematics ▮ (▮▮▮▮) ▮▮▮–▮▮▮

Contents lists available at SciVerse ScienceDirect

Discrete Applied Mathematics

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

A branch-and-cut algorithm for the latent-class logit assortment problem

Isabel Méndez-Díaz [a], Juan José Miranda-Bront [a], Gustavo Vulcano [b], Paula Zabala [a,c,*]

[a] Departamento de Computación, FCEyN, Universidad de Buenos Aires, Argentina
[b] Leonard N. Stern School of Business, New York University, New York, NY, USA
[c] Consejo Nacional de Investigaciones Científicas y Técnicas, Argentina

ARTICLE INFO

Article history:
Received 15 October 2010
Received in revised form 23 February 2012
Accepted 3 March 2012
Available online xxxx

Keywords:
Retail operations
Revenue management
Choice behavior
Multinomial logit
Integer programming
Fractional programming

ABSTRACT

We study the product assortment problem of a retail operation that faces a stream of customers who are heterogeneous with respect to preferences. Each customer belongs to a market segment characterized by a *consideration set* that includes the alternatives viewed as options, and by the preference weights that the segment assigns to each of those alternatives. Upon arrival, he checks the *offer set* displayed by the firm, and either chooses one of those products or quits without purchasing according to a multinomial-logit (MNL) criterion. The firm's goal is to maximize the expected revenue extracted during a fixed time horizon. This problem also arises in the growing area of choice-based, network revenue management, where computational speed is a critical factor for the practical viability of a solution approach.

This so-called latent-class, logit assortment problem is known to be NP-Hard. In this paper, we analyze unconstrained and constrained (i.e., with a limited number of products to display) versions of it, and propose a branch-and-cut algorithm that is computationally fast and leads to (nearly) optimal solutions.

1. Introduction

Assortment planning is one of the critical tasks in retail operations. It consists of specifying the mix of products a retailer should carry in each store at each point in time so as to maximize sales or gross margin subject to several possible constraints that vary by context. Examples of these constraints are: limited space for displaying products on a shelf (or on a website for the case of e-tailers), limited budget for the procurement of products, and strategic decisions of having only one or more than one vendor for a particular type of product.

The building block for retailers to make merchandising management decisions is the *category*. A *merchandise category* is a group of stock keeping units (SKUs) that customers may see as substitutes. This group can sometimes be segmented into *subcategories* or *classes*. The retailer's planning problem starts from the long-term, strategic decision of defining the merchandise *variety* or *breadth*: how many categories should she carry? The next decision is still strategic: how many and which SKUs should she carry in each category? (i.e., the merchandise *depth*). The answers to these two questions define the retailer's positioning in the market and its brand image. For instance, a full-line discount store chain may offer a wide variety of merchandise categories ranging from consumer electronics to kid's apparel, but only a limited number of SKUs in each category. In contrast, a consumer electronic retailer provides a narrower breadth, but more SKUs per (sub)category. The

* Corresponding author at: Departamento de Computación, FCEyN, Universidad de Buenos Aires, Argentina. Fax: +54 1145763359.
E-mail addresses: imendez@dc.uba.ar (I. Méndez-Díaz), jmiranda@dc.uba.ar (J.J. Miranda-Bront), gvulcano@stern.nyu.edu (G. Vulcano), pzabala@dc.uba.ar (P. Zabala).

0166-218X/$ – see front matter © 2012 Elsevier B.V. All rights reserved.
doi:10.1016/j.dam.2012.03.003

Please cite this article in press as: I. Méndez-Díaz, et al., A branch-and-cut algorithm for the latent-class logit assortment problem, Discrete Applied Mathematics (2012), doi:10.1016/j.dam.2012.03.003

third relevant question is more tactical: how much inventory should she stock of each SKU, and at what price? In addition, retailers need to periodically revise their assortment because of seasonality effects, introduction of new products, and shifts in consumer tastes.

When consumers visit the store, they may know in advance which specific product they are looking for (at the SKU level), they may just know the kind of product they want to purchase, or they may become interested in a specific product while browsing different categories. In any case, it is clear that the retailer can influence their decisions and induce substitution behavior by displaying (or not) a particular SKU. In the end, the retailer's sales volume is a function of volume of demand, customers' preferences, and answers to the three questions above.

According to Fisher [7], despite the huge amount of consumer transaction data that retailers have been collecting through point-of-sales (POS) scanners, radio frequency identification (RFID) tags, customer loyalty cards, and website click-through streams, their decision making process is still highly qualitative and judgment based. Moreover, he reports evidence on the significant suboptimality of decisions primarily based on qualitative assessments. Nevertheless, the retailing industry seems to be undergoing a transition to a balanced blend of art and science. Indeed, assortment planning is a relatively new and active area of research that is receiving high attention from retailers, consultants and software providers.

The retailer's solution to the assortment planning problem implies answering the three questions posed before. Here, we focus on the depth problem, i.e., the definition of the product mix within a category. The optimal assortment that answers this question solves the tradeoff between: (a) including a new product and increasing the demand to the category, and (b) cannibalizing the demand of other products' sales from the same category. In this paper, we do not address the question on inventory level decisions. Our approach is suitable for settings where the retailer has the ability to change the product mix upon arrival of each consumer so that she only shows items in stock (e.g., online retailers who can check in real time their stock levels and show a customized menu of products), or for slow moving SKUs of which she carries just one unit or a small amount of inventory (e.g., jewelry, auto parts, books and DVDs).[1] We also assume that prices have already been set.

This problem also arises in other settings. For example, in online advertising, the demand for each product may be defined by the number of customers who click on the ad, and the probability that a customer will click on a particular ad will likely depend on the mix of ads shown. In addition, the number of ads displayed is limited by the number of slots in the web page.

Another area of application is revenue management (RM). In this context (e.g., when an airline sells tickets), the central problem is how to ration the amount of capacity sold to various products in order to maximize revenue. The airline may implement the rationing by dynamically controlling the availability of products (with different restrictions and fares) as a function of the remaining capacity and time prior to flight departure. This rationing occurs at the leg or network levels. The book by Talluri and van Ryzin [27] provides a complete overview of both single-resource and network RM problems. As opposed to traditional RM models where demand was unaffected by the availability of other products, recently there has been a growing interest in modeling customer choice behavior. One of the most promising proposals to account for choice behavior in RM is the *choice-based deterministic linear programming* (CDLP) model. This method has shown an excellent revenue performance in exhaustive simulation studies [20]. The output of CDLP are the *offer sets* (i.e., the subsets of the full assortment) to make available to the customers at different points in time during the booking horizon. The decision variables are the length of time during which these offer sets must be exhibited, and therefore there is an exponential number of them. Indeed, the cases known so far where the CDLP model can be solved relatively efficiently are restrictive in terms of the supported demand model. The first attempt to overcome this limitation seeking a more general demand model was our column generation algorithm in [18], where we proposed a heuristic embedded in a standard mixed-integer programming (MIP) solver (CPLEX in our case) to approximately solve the column generation subproblem. The solution to the column generation subproblem identifies the product mix that should become basic in the master linear program. It turns out that this subproblem is equivalent to the retailer's assortment depth problem stated before.

In this paper, we present a branch-and-cut (B&C) algorithm to solve exactly the product assortment problem of maximizing the expected revenue rate. A distinctive feature of our formulation is the generality of the demand model. Specifically, we consider a market model where each customer belongs to a segment characterized by a *consideration set* that includes the alternatives that the customer views as options. Upon arrival, a customer checks the offer set displayed by the firm, and either chooses one of the products in his choice set (i.e., in the intersection of his consideration set and the offer set) or quits without purchasing, according to a multinomial-logit (MNL) criterion. In our demand model, segments are allowed to overlap; i.e., a product can belong simultaneously to the consideration sets of two or more segments. The firm knows the market segmentation, but *ex ante* does not know which class each arriving customer belongs to. Yet, she needs to provide an optimal offer set. This *latent class* multinomial-logit (LC-MNL) demand model scales the complexity of the assortment problem from polynomial when segments are disjoint (e.g., see [13] or [21]) to NP-Hard (as proved in our former paper [18]), but brings enough flexibility into the model so that more realistic preferences can be captured. For example, in retailing, one segment could be defined by price-sensitive customers for a class of products, and another segment by quality-sensitive customers for the same class of products. Clearly, the preference weight that a customer from each segment gives to a particular product (e.g. a high-end, high price product) is different.

We study two versions of the latent class logit assortment problem. In the *capacitated LC-MNL* version the number of products to display is limited by shelf space in a brick-and-mortar retailer, or by number of slots in a website for an e-tailer, as

[1] Fisher and Vaidyanathan [8] report empirical results of a study for snack foods and car tires under this assumption.

Please cite this article in press as: I. Méndez-Díaz, et al., A branch-and-cut algorithm for the latent-class logit assortment problem, Discrete Applied Mathematics (2012), doi:10.1016/j.dam.2012.03.003

opposed to the unconstrained number in the *uncapacitated LC-MNL*. For both versions, the problem formulation corresponds to a hyperbolic (or fractional) 0–1 programming model. Following Wu [30], the model can be linearized into a mixed 0–1 formulation. In our former paper [18], where we solved the uncapacitated version of this MIP using a combination of standard built-in methods provided by CPLEX and a greedy heuristic, the computational time to solve the CDLP to optimality was significant for some of the reported cases, reaching almost an hour for moderate-size networks. We circumvented this drawback by solving the CDLP approximately. The revenue losses for those difficult cases averaged 0.75% and scaled up to 1.4%, which were yet very favorable when compared to the revenues obtained under the traditional independent demand model assumption, but that could hopefully be recovered within a feasible computational budget.

A capacitated, latent class, MNL model has been recently considered by Smith [25]. He points out that under two extreme cases the model can be (approximately) solved via a simple ranking procedure. These extreme cases correspond to the retailer being either a monopolist or a marginal player in the market. He also conducts an empirical analysis over the optimization of an assortment of DVD players, and argues that when customers are indeed segmented according to consideration sets and the retailer uses this information correctly, a substantial increase in expected profit may be achieved. In his reported numerical cases, the additional revenues over the case where the seller ignores the heterogeneity in the market scale by factors of 2 or 3.

Smith's promising results confirm the need for a computationally feasible methodology, able to solve or at least approximately solve assortment planning problems for segmented demand models under different retailers' market shares. Our B&C methodology contributes in this regard. It relies on five families of valid inequalities that we managed to derive to tighten the MIP formulation. These inequalities were analyzed under different primal heuristics, cutting plane strategies and branching strategies, so that in the end we identified a combination of decisions that notoriously speeds up the work of the solver. We test our procedure in the context of capacitated and uncapacitated retail assortment problems, and in the context of the CDLP for choice-based RM.

The contribution of this piece of research is two-fold. Revenue-wise, we are able to recover an average of 1% left behind by the greedy heuristic presented in our earlier paper [18]. Although this 1% might sound not relevant enough, it may indeed be quite significant for retail operations that typically operate with high gross margins and low net profits, where small changes in revenue can have a profound impact on the financial performance. For illustration, according to the US Census Bureau, the estimated average annual gross margin (as a percentage of sales) of US retail firms for the period 1993 through 2008 was 27.8%. Just for the year 2008, it was 27.3%, with peak values of 47.9% for shoe stores, 43.3% for clothing stores, 45.3% for furniture and home furnishings stores. The aggregate net profit for large retailers in 2008 was 1.37%. Taking a reference gross margin of 27%, a 1% increase in revenues like the one achieved by our B&C will turn into a 20% increase in profits.[2] Time-wise, the computational study shows that our B&C is faster than the standard CPLEX, and that it converges to higher-revenue solutions within a given computational budget. It is even competitive with the heuristic in many of the cases. When evaluating the fulfillment of these two desirable features, our approach dominates on both for many of the instances and provides an interesting tradeoff in others, becoming an excellent candidate to be pursued in practice.

The remainder of this paper is organized as follows: In Section 2, we review the related literature. In Section 3, we formulate our optimization problem for both capacitated and uncapacitated LC-MNL settings. Section 4 presents our approach for solving the MIP. Our numerical results are reported in Section 5, and we conclude in Section 6.

2. Literature review

We refer the reader to the excellent survey of Kök et al. [12] for a comprehensive review of the assortment planning literature. In this paper, we consider a utility-based model of substitution, where each consumer associates a random utility with each product in his consideration set and with the no-purchase option, and chooses the available alternative with highest utility. One of the random utility-based models most commonly used in the economics and marketing literature is the MNL (e.g., see the books by Ben-Akiva and Lerman [1] and Train [28]).

Kök et al. [12] identify two types of substitution in view of its supply side causes. In *assortment-based* or *static* substitution, a consumer identifies a favorite product in a catalog or based on what he has seen in other stores, and switches to an available variant when his favorite product is not carried in store. Consumers know the assortment but not the inventory levels. If a consumer selects a variant that is out of stock, he does not undertake a second choice and the sale is lost. This is the type of substitution studied in [26,29]. As a result, demand is independent of inventory levels, though it does depend on the initial (or advertised) product mix offered. Cachon et al. [2] extend the results in [29] to account for consumer search costs.

In *stockout-based* substitution, a consumer switches to an available variant when his favorite product carried in the store catalog is stocked-out at the time of his shopping. In a follow-up paper, Mahajan and van Ryzin [14] develop a stochastic gradient algorithm for a newsvendor-type, stochastic inventory model under the so-called *dynamic substitution*, where arriving consumers substitute among product variants while inventory is depleted as long as there is an available variant which is more valuable than the no-purchase option. Here, consumers observe the set of products in-stock and then make their product choice. In this regard, dynamic substitution accounts for both assortment-based and stockout-based substitutions. In our work we do not focus on inventory considerations, so that the distinction between assortment-based and stockout-based substitution blurs.

[2] This fact comes from the simple math: change in profit = revenue increase × gross margin/net profit, where values are in percentages.

Please cite this article in press as: I. Méndez-Díaz, et al., A branch-and-cut algorithm for the latent-class logit assortment problem, Discrete Applied Mathematics (2012), doi:10.1016/j.dam.2012.03.003

More recent developments include the assortment optimization studied by Rusmevichientong et al. [21] for a capacitated, single-class, MNL choice model. Our capacitated and uncapacitated LC-MNL problems are generalizations of the *static* settings described in Section 2 of their paper. By *static* those authors mean that the preference weights that consumers have for the alternatives are known in advance. The uncapacitated problem is indeed equivalent to the column generation subproblem derived from the CDLP formulation introduced by Gallego et al. [9] and further studied by Liu and van Ryzin [13]. The main computational feature of this subproblem is that when the stream of arrivals is homogeneous with respect to preferences, a simple greedy procedure solves the problem to optimality in polynomial time. Rusmevichientong et al. [21] showed that the single-class capacitated MNL cannot be solved through a greedy procedure, but developed an alternative polynomial time algorithm that builds upon the ideas introduced by Meggido [16] for optimizing a rational objective function. In Rusmevichientong and Topaloglu [23], the authors analyze a single-period, robust assortment optimization problem under MNL choice, where they incorporate uncertainty in the model parameters through an uncertainty set.

An interesting operational extension is when the seller learns consumer preferences in real time, and adjusts the assortment accordingly. This dynamic assortment planning problem was pioneered by the work of Caro and Gallien [4]. In their formulation, demands for products are independent, and also irrespective of product availability. They also extend the model to account for substitution among products, but not following the MNL model. Rusmevichientong et al. [21] also study preference learning in the *dynamic* version of their model. Saure and Zeevi [24] tackle a related problem. In their paper, given limited display capacity, the seller implements dynamic policies that balance exploration and exploitation phases in order to infer the customer mean utilities for the different products and maximize revenues within a fixed time horizon. The random utility model that they consider is more general than the single-class MNL of Rusmevichientong et al. [21].

All these models share a limitation: They either assume that consumers are homogeneous with respect to preferences or that their consideration sets are non-overlapping, so that the problem can be decomposed into a collection of homogeneous consumer subproblems. As mentioned earlier in this paper, price-sensitive and quality-sensitive consumers have different priorities for a given set of products, and hence partitioning the product space between both sensitive types is by all means a very strong assumption.

Our demand model is, in fact, an example of the so-called *latent class MNL model*, a particular case of the *mixed multinomial logit* (MMNL) model, first introduced by Cardell and Dunbar [3]. In the general MMNL model choice probabilities are defined by overlapping MNL models over a mixing distribution. For the latent class model, which has been common in psychology and marketing (e.g., [11,5]), and more recently in RM, this mixing distribution is discrete. One of the features of the MMNL models is that they do not exhibit *independence from irrelevant alternatives* (IIA), which precludes inaccuracies in cases where similar options could drive towards pathological substitution patterns (e.g., in RM, adjacent flights whereby demand spill from one flight flows disproportionately to flights with nearby departure times). For further discussion, e.g. see [28, Chapter 6]. McFadden and Train [15] establish the important result that, in theory, any random utility model can be captured by a correct specification of the mixing distribution in a MMNL, also providing support to the flexibility of our model in terms of its capability to represent more general customer choice behavior.

As mentioned in Section 1, we proved in our previous paper [18] that the latent class logit assortment problem is NP-Hard,[3] and we solved the CDLP using a greedy heuristic to speed up the solution of the column generation subproblem. Meissner and Straus [17] yet explored alternative heuristics based on approximate dynamic programming that exhibit some level of revenue improvement, but that are considerably more computationally expensive. The branch-and-cut approach that we describe in the next sections is able to solve the CDLP to optimality noticeably fast.

Finally, a possible concern about this complex demand model is how to estimate its parameters. Smith [25, Section 3] describes a simple procedure to define the consideration sets and associated preference weights for the products involved. The number of customer classes in his empirical study is 2213. His work is a clear indicator of the importance of capturing a segmented demand model, and of its feasibility. More recently, Farias et al. [6] propose an estimation procedure for a related choice model.

3. Model

3.1. Basic formulation

We consider a single period optimization problem where a risk neutral firm seeks to maximize expected revenues. There is a set of in-stock products denoted by $N = \{1, \ldots, n\}$, and the firm must decide the offer set $S \subset N$ to exhibit at the beginning of the period. The reward obtained from a consumer's acceptance of one unit of product j is w_j. A dynamic version of this problem could be obtained by partitioning the selling horizon into small enough pieces (e.g., by looking at the interarrival times), and then solving the single period problem within each time slot.

An arriving customer belongs to one of L market segments denoted $l = 1, \ldots, L$. Each segment is characterized by a *consideration set* $C_l \subset N$. This is different from [9,21,23], who just allow for a unique segment $C_1 = N$. In addition, the distinguishing feature of our model with respect to Liu and van Ryzin [13] – who consider multiple segments – is that we

[3] Goyal et al. [10] later proved a hardness result for a different choice model, where customers are represented by preference lists.

Please cite this article in press as: I. Méndez-Díaz, et al., A branch-and-cut algorithm for the latent-class logit assortment problem, Discrete Applied Mathematics (2012), doi:10.1016/j.dam.2012.03.003

allow them to overlap. That is, we admit that $C_l \bigcap C_{l'} \neq \emptyset$ for $l \neq l'$. From the firm's perspective, the arriving customer belongs to segment l with probability p_l, with $\sum_{l=1}^{L} p_l = 1$. Customers arrive in accordance with a Poisson process with rate λ, so that the arriving stream of segment-l customers is a Poisson process with rate $\lambda_l = \lambda p_l$.

Given an offer set S, the arriving customer chooses product $j \in S$ with probability $P_j(S)$, where $P_j(S) = 0$ if $j \notin S$. We will denote the no-purchase probability by $P_0(S)$, and by total probability, we have that $\sum_{j \in S} P_j(S) + P_0(S) = 1$. As it is generally the case in the choice behavior literature under consumer driven substitution, these probabilities are based on the multinomial logit model (MNL). Under the MNL choice model, the choice probability of a segment-l customer is defined by a preference vector $\vec{v}_l \geq 0$, $\vec{v}_l \neq 0$, that indicates the "attractiveness" of each product contained in C_l. This vector, together with the no-purchase preference v_{l0}, determine a customer's choice probabilities as follows: If we let $P_{lj}(S)$ denote the probability that a customer from segment l chooses product $j \in C_l \bigcap S$ when S is offered, then,

$$P_{lj}(S) = \frac{v_{lj}}{\sum\limits_{h \in C_l \bigcap S} v_{lh} + v_{l0}}.$$

If $j \notin C_l \bigcap S$ or $j \notin C_l$, then $v_{lj} = 0$ (and hence $P_{lj}(S) = 0$). Noting that from the seller's perspective the segment of a customer is not identifiable, the probability that an arriving customer chooses product $j \in S$ is given by

$$P_j(S) = \sum_{l=1}^{L} p_l P_{lj}(S). \tag{1}$$

The value $P_j(S)$ can be interpreted as the deterministic quantity of product j sold when set S is offered.

We assume that every arrival has a positive probability $P_0(S)$ of not purchasing any product, i.e., $v_{l0} > 0$ for $l = 1, \ldots, L$. The probability of purchase, $1 - P_0(S)$, can be considered as a proxy for the market share of the firm.

Our decision variables are represented by the binary vector $\vec{y} \in \{0, 1\}^n$. It defines the characteristic vector of set S: If S is the set currently offered, $y_j = 1$ if $j \in S$, and $y_j = 0$ otherwise. Thus, our revenue maximization problem can be expressed as:

$$\max_{\vec{y} \in \{0,1\}^n} \left\{ \sum_{l=1}^{L} \lambda_l \frac{\sum\limits_{j \in C_l} w_j v_{lj} y_j}{\sum\limits_{i \in C_l} v_{li} y_i + v_{l0}} \right\}$$

or equivalently,

$$\max_{\vec{y} \in \{0,1\}^n} \left\{ \sum_{j=1}^{n} w_j y_j \left(\sum_{l=1}^{L} \frac{\lambda_l v_{lj}}{\sum\limits_{i \in C_l} v_{li} y_i + v_{l0}} \right) \right\}. \tag{2}$$

Without loss of generality, we will assume that $w_j > 0$ (otherwise, $y_j^* = 0$ in the optimal solution). In Theorem 1 of our predecessor paper [18] we proved that problem (2) is NP-Hard when $L \geq n - 1$. More recently, Rusmevichientong et al. [22] generalize this result for the case when $L \geq 2$.

This hyperbolic problem can be reformulated as a nonlinear MIP problem (e.g. see [19]). By defining the variables

$$x_l = \frac{1}{\sum\limits_{i \in C_l} v_{li} y_i + v_{l0}}, \quad l = 1, \ldots, L, \tag{3}$$

problem (2) can be rewritten as:

$$\begin{aligned}
&\max \sum_{l=1}^{L} \sum_{j \in C_l} \lambda_l w_j v_{lj} y_j x_l \\
&\text{s.t.: } x_l v_{l0} + \sum_{i \in C_l} v_{li} y_i x_l = 1, \quad l = 1, \ldots, L \\
&y_j \in \{0, 1\}, \quad j \in N, \\
&x_l \geq 0 \quad l = 1, \ldots, L.
\end{aligned} \tag{4}$$

In this formulation, the objective is to maximize the revenue rate. The first set of constraints is enforcing that the total probability for choosing the available products from each consideration set C_l adds up to 1. More specifically, the multiplication $x_l v_{l0}$ stands for the no-purchase probability, and $v_{li} y_i x_l$ stands for the probability that a segment-l arrival chooses product i.

Please cite this article in press as: I. Méndez-Díaz, et al., A branch-and-cut algorithm for the latent-class logit assortment problem, Discrete Applied Mathematics (2012), doi:10.1016/j.dam.2012.03.003

3.2. Linear formulation

The nonlinear terms $y_i x_l$ can be linearized (see Wu [30]), and a linear mixed 0–1 formulation can be obtained. The linearization is based on the following observation: A polynomial mixed 0–1 term $z = xy$, where x is a continuous variable and y is a 0–1 variable, can be represented by the following linear inequalities: (1) $x - z \leq K - Ky$; (2) $z \leq x$; (3) $z \leq Ky$; and (4) $z \geq 0$, where K is a *large number*.

In the context of our problem, if we define variables $z_{li} = x_l y_i$ and apply this result, it is possible to obtain the formulation:

$$\max \sum_{l=1}^{L} \sum_{j \in C_l} \lambda_l w_j v_{lj} z_{lj}$$

$$\text{s.t.: } x_l v_{l0} + \sum_{i \in C_l} v_{li} z_{li} = 1, \quad \forall l$$

$$x_l - z_{li} \leq K - K y_i, \quad \forall l,\ i \in C_l \tag{5}$$

$$z_{li} \leq x_l, \quad \forall l,\ i \in C_l$$

$$z_{li} \leq K y_i, \quad \forall l,\ i \in C_l \tag{6}$$

$$y_j \in \{0, 1\}, \quad x_l \geq 0,\ z_{li} \geq 0.$$

Since $x_l = \frac{1}{\sum_{i \in C_l} v_{li} y_i + v_{l0}}$, we can tighten (5) by noting that $x_l - z_{li} = x_l(1 - y_i)$. For a given segment l, it would be enough to take $K = 1/v_{l0}$, and therefore $v_{l0} x_l - v_{l0} z_{li} \leq 1 - y_i,\ \forall l,\ i \in C_l$. We can also tighten (6) by replacing it with $(v_{l0} + v_{li}) z_{li} \leq y_i,\ \forall l,\ i \in C_l$, leading to the following MIP for the latent-class MNL problem:

$$\text{(LC-MNL)} \quad \max \sum_{l=1}^{L} \sum_{j \in C_l} \lambda_l w_j v_{lj} z_{lj} \tag{7}$$

$$\text{s.t.: } x_l v_{l0} + \sum_{i \in C_l} v_{li} z_{li} = 1, \quad \forall l$$

$$v_{l0} x_l - v_{l0} z_{li} \leq 1 - y_i, \quad \forall l,\ i \in C_l \tag{8}$$

$$z_{li} \leq x_l, \quad \forall l,\ i \in C_l$$

$$(v_{l0} + v_{li}) z_{li} \leq y_i, \quad \forall l,\ i \in C_l \tag{9}$$

$$y_j \in \{0, 1\}, \quad x_l \geq 0,\ z_{li} \geq 0.$$

What we have described so far is the unconstrained version of the problem. For the capacitated LC-MNL problem there is a limit c on the number of products to exhibit, and we simply add the constraint: $\sum_{j=1}^{n} y_j \leq c$.

Under this setting, we may wonder if the solutions are incremental with c in the sense that if for a given instance of the problem we consider two capacity levels c_1 and c_2 with $c_1 < c_2$, and with respective optimal solutions S_1^* and S_2^*, then it should hold that $S_1^* \subset S_2^*$. van Ryzin and Mahajan [29] showed that this is the case for a single-class MNL model where profit margins of all products are the same. If that were the case here, we would have an incremental construction of the optimal solution. Rusmevichientong et al. [21, Example 2.1] show that, unfortunately, this is no longer true when there is a capacity constraint even under a single-class demand model. Independently, Smith [25, Section 2.3] provides a counterexample for a multi-class demand setting.

3.3. Polyhedral results

Our main focus is on developing a B&C algorithm which is a refinement of a branch-and-bound where LP relaxations for each subproblem are strengthened with globally valid inequalities (cutting planes). There are two types of cuts that can be added in a B&C scheme. The first type are general purpose cuts that are applicable to any integer programming problem, such us cover and Gomory cuts. Most commercial packages have options for including them in the LP relaxation. The second type takes advantage of the specific polyhedral structure of the model. From our investigations of the underlying polytope associated with the LC-MNL formulation, we were able to infer the five families of valid inequalities that follow.

Proposition 1. *For a segment l and a product $j \in C_l$,*

$$\left(v_{l0} + \sum_{i \in C_l} v_{li} \right) z_{lj} \geq y_j \tag{10}$$

is a valid inequality.

Please cite this article in press as: I. Méndez-Díaz, et al., A branch-and-cut algorithm for the latent-class logit assortment problem, Discrete Applied Mathematics (2012), doi:10.1016/j.dam.2012.03.003

Proof. From the linearization applied in Section 3, we know that $z_{lj} = x_l y_j$, and substituting this expression in (10) we get

$$\left(v_{l0} + \sum_{i \in C_l} v_{li}\right) x_l y_j \geq y_j.$$

If a feasible solution has $y_j = 0$, this expression is trivially satisfied. Otherwise, if $y_j = 1$, the inequality reads

$$\left(v_{l0} + \sum_{i \in C_l} v_{li}\right) x_l \geq 1,$$

which is satisfied given the definition of variable x_l in (3). □

The previous proposition is enforcing a necessary condition for feasible MNL probabilities within each segment: $x_l \geq 1/(v_{l0} + \sum_{i \in C_l} v_{li})$. Equivalently, it is guaranteeing that

$$v_{l0} + v_{lj} + \sum_{i \in C_l, i \neq j} v_{li} \geq v_{l0} + v_{lj} + \sum_{i \in C_l, i \neq j} v_{li} y_i.$$

For a given consideration set C_l, the LHS represents the total attractiveness of the products there, while the RHS represents some partial aggregated attractiveness of a subset of products.

Proposition 2. *Consider a segment l and a product $k \in C_l$ such that $\sum_{\substack{j \neq k, \\ j \in C_l}} v_{lj} \geq v_{lk}$. Then, for all $i \neq k$, $i \in C_l$,*

$$\left(v_{l0} + \sum_{\substack{j \neq k, \\ j \in C_l}} v_{lj}\right) z_{li} - (v_{l0} + v_{lk}) z_{lk} \geq y_i - y_k \tag{11}$$

is a valid inequality.

Proof. First, similarly to the previous proposition, we replace variables z_{li} and z_{lk} in inequality (11) by their definitions, which leads to the expression

$$\left(v_{l0} + \sum_{\substack{j \neq k, \\ j \in C_l}} v_{lj}\right) x_l y_i - (v_{l0} + v_{lk}) x_l y_k \geq y_i - y_k. \tag{12}$$

We now verify that this inequality is satisfied by all feasible solutions, considering the following three cases:

- Case 1: $y_i = y_k = 0$. The inequality is trivially satisfied, given that both sides equal zero.
- Case 2: Either $y_k = 1$ or $y_i = 1$. When $y_k = 1$, expression (12) reduces to inequality (9). On the contrary, when $y_i = 1$, given that $y_k = 0$ and that v_{lk} will not be present in the expression defining x_l, the inequality (12) reduces to the expression

  $$v_{l0} + \sum_{\substack{j \neq k, \\ j \in C_l}} v_{lj} \geq v_{l0} + \sum_{\substack{h \neq k, \\ h \in C_l}} v_{lh} y_h,$$

  which clearly holds.
- Case 3: $y_i = y_k = 1$. In this case, expression (12) can be rewritten as

  $$x_l \left(\sum_{\substack{j \neq k, \\ j \in C_l}} v_{lj} - v_{lk}\right) \geq 0.$$

  By definition, we know that $x_l > 0$. In addition, from the hypothesis of the proposition, $\sum_{j \neq k} v_{lj} \geq v_{lk}$, which proves that the inequality is satisfied.

These three cases cover all possible situations and prove the validity of inequality (11). □

The meaningful instance for Proposition 2 is case 2 when $y_k = 0, y_i = 1$. Like Proposition 1, it is stating that the total attractiveness of the products other than k in C_l is bigger than the partial attractiveness of a subset of them. Case 3 follows from the assumption in the statement of the proposition: the aggregated attractiveness of the products other than k in C_l is higher than the single attractiveness of k.

The next three propositions are the outcome of algebraic manipulations and do not seem to have a clean interpretation in terms of the choice problem. They rely on the fact that $v_{lj} \geq 0$, for all l, j.

Please cite this article in press as: I. Méndez-Díaz, et al., A branch-and-cut algorithm for the latent-class logit assortment problem, Discrete Applied Mathematics (2012), doi:10.1016/j.dam.2012.03.003

Proposition 3. *For a segment l and a product $k \in C_l$,*

$$\sum_{j\neq k, j\in C_l} (v_{lj}(v_{l0}+v_{lj})z_{lj}) - v_{lk}(v_{l0}+v_{lk})z_{lk} \leq \sum_{j\neq k, j\in C_l} v_{lj}y_j - v_{lk}y_k \tag{13}$$

is a valid inequality.

Proof. We verify the validity considering the following four cases:

- Case 1: $y_i = 0\ \forall i \in C_l$. The inequality is trivially satisfied, given that both sides equal zero.
- Case 2: Consider $\bar{Z} = \{j : y_j = 1, j \in C_l, j \neq k\}$ and $y_k = 0$. By definition, $z_{lj} = \frac{1}{v_{l0}+\sum_{j\in\bar{Z}} v_{lj}}$ for $j \in \bar{Z}$, then the inequality expression can be rewritten as

  $$\sum_{j\in\bar{Z}} v_{lj}(v_{l0}+v_{lj}) \leq \sum_{j\in\bar{Z}} v_{lj}\left(v_{l0}+\sum_{h\in\bar{Z}} v_{lh}\right)$$

  which is valid since $v_{lj} \geq 0$, and $v_{lj} \leq \sum_{k\in\bar{Z}} v_{lk}$, for $j \in \bar{Z}$.
- Case 3: $y_k = 1$ and $y_j = 0\ \forall j \in C_l, j \neq k$. By definition, $z_{lk} = \frac{1}{v_{l0}+v_{lk}}$. Then, the inequality expression can be rewritten as

  $$-v_{lk}(v_{l0}+v_{lk}) \leq -v_{lk}(v_{l0}+v_{lk})$$

  which trivially holds.
- Case 4: Again, consider $\bar{Z} = \{j : y_j = 1, j \in C_l, j \neq k\}$, but now with $y_k = 1$. By definition, $z_{lk} = z_{lj} = \frac{1}{v_{l0}+v_{lk}+\sum_{j\in\bar{Z}} v_{lj}}$ for $j \in \bar{Z}$. Then, the inequality reads

  $$\sum_{j\in\bar{Z}} v_{lj}(v_{l0}+v_{lj}) - v_{lk}(v_{l0}+v_{lk}) \leq \sum_{j\in\bar{Z}} v_{lj}\left(v_{l0}+v_{lk}+\sum_{h\in\bar{Z}} v_{lh}\right) - v_{lk}\left(v_{l0}+v_{lk}+\sum_{h\in\bar{Z}} v_{lh}\right).$$

  Canceling terms we get, as in case 2,

  $$\sum_{j\in\bar{Z}} v_{lj}(v_{l0}+v_{lj}) \leq \sum_{j\in\bar{Z}} v_{lj}\left(v_{l0}+\sum_{h\in\bar{Z}} v_{lh}\right).$$

  These four cases prove the validity of inequality (13). □

Proposition 4. *Take a segment l with $|C_l| = n_l$, and let $\{j_1, \ldots, j_{n_l}\}$ be an index permutation of the products in C_l. Then,*

$$-v_{lj_1}(v_{l0}+v_{lj_1})z_{lj_1} + v_{lj_2}(v_{l0}+v_{lj_2})z_{lj_2} + \sum_{i=3}^{n_l} v_{lj_i}\left(v_{l0}+v_{lj_i}+2v_{lj_2}+2\sum_{h=3}^{i-1} v_{lj_h}\right)z_{lj_i} \geq -v_{lj_1}y_{j_1} + \sum_{i=2}^{n_l} v_{lj_i}y_{j_i} \tag{14}$$

is a valid inequality.

Proof. For this proof, we define $\bar{Z} = \{i : y_{j_i} = 1, j_i \in C_l, i \neq 1, 2\}$. We verify the validity of (14) considering the following five cases:

- Case 1: $y_{j_i} = 0\ \forall i = 1, \ldots, n_l$. The inequality is trivially satisfied, given that both sides equal zero.
- Case 2: $y_{j_1} = y_{j_2} = 0$. By definition, $z_{lj_i} = \frac{1}{v_{l0}+\sum_{k\in\bar{Z}} v_{lj_k}}$ for $i \in \bar{Z}$, then the inequality expression can be rewritten as

  $$\sum_{i\in\bar{Z}} v_{lj_i}\left(v_{l0}+v_{lj_i}+2v_{lj_2}+2\sum_{h=3}^{i-1} v_{lj_h}\right) \geq \sum_{i\in\bar{Z}} v_{lj_i}\left(v_{l0}+\sum_{k\in\bar{Z}} v_{lj_k}\right).$$

  Considering that

  $$\sum_{i\in\bar{Z}} v_{lj_i}\sum_{k\in\bar{Z}} v_{lj_k} = \sum_{i\in\bar{Z}} v_{lj_i}^2 + \sum_{i\in\bar{Z}} 2v_{lj_i}\sum_{h=3, h\in\bar{Z}}^{i-1} v_{lj_h}$$

  the validity of the inequality follows.
- Case 3: $y_{j_1} = 0$ and $y_{j_2} = 1$. By definition, $z_{lj_2} = z_{lj_i} = \frac{1}{v_{l0}+v_{lj_2}+\sum_{k\in\bar{Z}} v_{lj_k}}$ for $i \in \bar{Z}$, then the inequality expression can be rewritten as

  $$v_{lj_2}(v_{l0}+v_{lj_2}) + \sum_{i\in\bar{Z}} v_{lj_i}\left(v_{l0}+v_{lj_i}+2v_{lj_2}+2\sum_{h=3}^{i-1} v_{lj_h}\right)$$
  $$\geq v_{lj_2}\left(v_{l0}+v_{lj_2}+\sum_{k\in\bar{Z}} v_{lj_k}\right) + \sum_{i\in\bar{Z}} v_{lj_i}\left(v_{l0}+v_{lj_2}+\sum_{k\in\bar{Z}} v_{lj_k}\right),$$

Please cite this article in press as: I. Méndez-Díaz, et al., A branch-and-cut algorithm for the latent-class logit assortment problem, Discrete Applied Mathematics (2012), doi:10.1016/j.dam.2012.03.003

or equivalently,

$$\sum_{i\in\bar{Z}} v_{lj_i}\left(v_{l0}+v_{lj_i}+2v_{lj_2}+2\sum_{h=3}^{i-1}v_{lj_h}\right)\geq 2v_{lj_2}\sum_{i\in\bar{Z}}v_{lj_i}+\sum_{i\in\bar{Z}}v_{lj_i}\left(v_{l0}+\sum_{k\in\bar{Z}}v_{lj_k}\right),$$

leading to expression

$$\sum_{i\in\bar{Z}} v_{lj_i}\left(v_{lj_i}+2\sum_{h=3}^{i-1}v_{lj_h}\right)\geq \sum_{i\in\bar{Z}}v_{lj_i}\sum_{k\in\bar{Z}}v_{lj_k}.$$

From previous case 2, the validity follows.

- Case 4: $y_{j_1}=1, y_{j_2}=0$. By definition, $z_{lj_1}=z_{lj_i}=\frac{1}{v_{l0}+v_{lj_1}+\sum_{k\in\bar{Z}}v_{lj_k}}$. Replacing the values in inequality (14),

$$\begin{aligned}&-v_{lj_1}(v_{l0}+v_{lj_1})+\sum_{i\in\bar{Z}}v_{lj_i}\left(v_{l0}+v_{lj_i}+2v_{lj_2}+2\sum_{h=3}^{i-1}v_{lj_h}\right)\\&\quad\geq -v_{lj_1}\left(v_{l0}+v_{lj_1}+\sum_{k\in\bar{Z}}v_{lj_k}\right)+\sum_{i\in\bar{Z}}v_{lj_i}\left(v_{l0}+v_{lj_1}+\sum_{k\in\bar{Z}}v_{lj_k}\right),\end{aligned}$$

which is equivalent to

$$\sum_{i\in\bar{Z}} v_{lj_i}\left(v_{lj_i}+2v_{lj_2}+2\sum_{h=3}^{i-1}v_{lj_h}\right)\geq \sum_{i\in\bar{Z}}v_{lj_i}\sum_{k\in\bar{Z}}v_{lj_k}.$$

From case 2, the validity follows.

- Case 5: $y_{j_1}=1, y_{j_2}=1$. By definition, $z_{lj_1}=z_{lj_2}=z_{lj_i}=\frac{1}{v_{l0}+v_{lj_1}+v_{lj_2}+\sum_{k\in\bar{Z}}v_{lj_k}}$, replacing the values in the inequality

$$\begin{aligned}&-v_{lj_1}(v_{l0}+v_{lj_1})+v_{lj_2}(v_{l0}+v_{lj_2})+\sum_{i\in\bar{Z}}v_{lj_i}\left(v_{l0}+v_{lj_i}+2v_{lj_2}+2\sum_{h=3}^{i-1}v_{lj_h}\right)\\&\quad\geq -v_{lj_1}\left(v_{l0}+v_{lj_1}+v_{lj_2}+\sum_{k\in\bar{Z}}v_{lj_k}\right)+v_{lj_2}\left(v_{l0}+v_{lj_1}+v_{lj_2}+\sum_{k\in\bar{Z}}v_{lj_k}\right)\\&\qquad+\sum_{i\in\bar{Z}}v_{lj_i}\left(v_{l0}+v_{lj_1}+v_{lj_2}+\sum_{k\in\bar{Z}}v_{lj_k}\right),\end{aligned}$$

which is equivalent to

$$\sum_{i\in\bar{Z}} v_{lj_i}\left(v_{l0}+v_{lj_i}+2v_{lj_2}+2\sum_{h=3}^{i-1}v_{lj_h}\right)\geq v_{lj_2}\sum_{k\in\bar{Z}}v_{lj_k}+\sum_{i\in\bar{Z}}v_{lj_i}\left(v_{l0}+v_{lj_2}+\sum_{k\in\bar{Z}}v_{lj_k}\right).$$

From case 2, the validity follows.

These five cases cover all possible situations, proving the validity of inequality (14). □

Proposition 5. *Take a segment l with $|C_l|=n_l$, and let $\{j_1,\ldots,j_{n_l}\}$ be an index permutation of the products in C_l, with $j_1<j_2$. Then,*

$$\begin{aligned}&v_{lj_1}(v_{l0}+v_{lj_1})(v_{l0}+2v_{lj_2})z_{lj_1}+v_{lj_2}(v_{l0}+v_{lj_2})(v_{l0}+2v_{lj_1})z_{lj_2}\\&\quad-\sum_{i=3}^{n_l}\left(v_{lj_i}v_{l0}(v_{l0}+v_{lj_i})-2v_{lj_i}v_{lj_1}v_{lj_2}+2v_{lj_i}v_{l0}\sum_{h=3}^{i-1}v_{lj_h}\right)z_{lj_i}\\&\leq 2v_{lj_1}v_{lj_2}+v_{l0}v_{lj_1}y_{j_1}+v_{l0}v_{lj_2}y_{j_2}-\sum_{i=2}^{n_l}v_{l0}v_{lj_i}y_{j_i}\end{aligned}\tag{15}$$

is a valid inequality.

Proof. For this proof, we define $\bar{Z}=\{i: y_{j_i}=1, j_i\in C_l, i\neq 1,2\}$. We verify the validity of (15) considering the following five cases:

Please cite this article in press as: I. Méndez-Díaz, et al., A branch-and-cut algorithm for the latent-class logit assortment problem, Discrete Applied Mathematics (2012), doi:10.1016/j.dam.2012.03.003

- Case 1: $y_{j_i} = 0\ \forall i = 1, \ldots, n_l$. The inequality is trivially satisfied, given that both sides equals zero.
- Case 2: $y_{j_1} = y_{j_2} = 0$. By definition, $z_{lj_i} = \frac{1}{v_{l0}+\sum_{k\in\bar{Z}} v_{lj_k}}$ for $i \in \bar{Z}$, then the inequality expression can be rewritten as

$$\sum_{i\in\bar{Z}} -v_{lj_i}v_{l0}(v_{l0}+v_{lj_i}) + 2v_{lj_i}v_{lj_1}v_{lj_2} - 2v_{lj_i}v_{l0}\sum_{h=3}^{i-1} v_{lj_h}$$
$$\leq 2v_{lj_1}v_{lj_2}\left(v_{l0}+\sum_{k\in\bar{Z}} v_{lj_k}\right) - \sum_{i\in\bar{Z}} v_{l0}v_{lj_i}\left(v_{l0}+\sum_{k\in\bar{Z}} v_{lj_k}\right),$$

  or equivalently,

$$\sum_{i\in\bar{Z}}\left(-v_{lj_i}v_{l0}v_{lj_i} - 2v_{lj_i}v_{l0}\sum_{h=3}^{i-1} v_{lj_h}\right) \leq 2v_{lj_1}v_{lj_2}v_{l0} - \sum_{i\in\bar{Z}} v_{l0}v_{lj_i}\left(\sum_{k\in\bar{Z}} v_{lj_k}\right).$$

  Since

$$\sum_{i\in\bar{Z}} v_{l0}v_{lj_i}\left(\sum_{k\in\bar{Z}} v_{lj_k}\right) = \sum_{i\in\bar{Z}} v_{l0}v_{lj_i}^2 + 2v_{l0}\sum_{k\in\bar{Z}} v_{lj_k}\sum_{h=3,h\in\bar{Z}}^{i-1} v_{lj_h}$$

  and $v_{l0}, v_{lj_1}, v_{lj_2} \geq 0$, the validity follows.
- Case 3: $y_{j_1} = 1, y_{j_2} = 0$. By definition, $z_{lj_1} = z_{lj_i} = \frac{1}{v_{l0}+v_{lj_1}+\sum_{k\in\bar{Z}} v_{lj_k}}$, then the inequality expression can be rewritten as

$$v_{lj_1}(v_{l0}+v_{lj_1})(v_{l0}+2v_{lj_2}) + \sum_{i\in\bar{Z}}\left(-v_{lj_i}v_{l0}(v_{l0}+v_{lj_i}) + 2v_{lj_i}v_{lj_1}v_{lj_2} - 2v_{lj_i}v_{l0}\sum_{h=3}^{i-1} v_{lj_h}\right)$$
$$\leq 2v_{lj_1}v_{lj_2}\left(v_{l0}+v_{lj_1}+\sum_{k\in\bar{Z}} v_{lj_k}\right) + v_{l0}v_{lj_1}\left(v_{l0}+v_{lj_1}+\sum_{k\in\bar{Z}} v_{lj_k}\right) - \sum_{i\in\bar{Z}} v_{l0}v_{lj_i}\left(v_{l0}+v_{lj_1}+\sum_{k\in\bar{Z}} v_{lj_k}\right),$$

  or equivalently,

$$\sum_{i\in\bar{Z}}\left(-v_{lj_i}v_{l0}v_{lj_i} - 2v_{lj_i}v_{l0}\sum_{h=3}^{i-1} v_{lj_h}\right) \leq -\sum_{i\in\bar{Z}} v_{l0}v_{lj_i}\sum_{k\in\bar{Z}} v_{lj_k}.$$

  With the same arguments as in case 1, the validity follows.
- Case 4: $y_{j_1} = 0, y_{j_2} = 1$. This is symmetric to the previous case.
- Case 5: $y_{j_1} = 1, y_{j_2} = 1$. By definition, $z_{lj_1} = z_{lj_2} = z_{lj_i} = \frac{1}{v_{l0}+v_{lj_1}+v_{lj_2}+\sum_{k\in\bar{Z}} v_{lj_k}}$, then the inequality expression can be rewritten as

$$v_{lj_1}(v_{l0}+v_{lj_1})(v_{l0}+2v_{lj_2}) + v_{lj_2}(v_{l0}+v_{lj_2})(v_{l0}+2v_{lj_1})$$
$$+ \sum_{i\in\bar{Z}}\left(-v_{lj_i}v_{l0}(v_{l0}+v_{lj_i}) + 2v_{lj_i}v_{lj_1}v_{lj_2} - 2v_{lj_i}v_{l0}\sum_{h=3}^{i-1} v_{lj_h}\right)$$
$$\leq 2v_{lj_1}v_{lj_2}\left(v_{l0}+v_{lj_1}+v_{lj_2}+\sum_{k\in\bar{Z}} v_{lj_k}\right) + v_{l0}v_{lj_1}\left(v_{l0}+v_{lj_1}+v_{lj_2}+\sum_{k\in\bar{Z}} v_{lj_k}\right)$$
$$+ v_{l0}v_{lj_2}\left(v_{l0}+v_{lj_1}+v_{lj_2}+\sum_{k\in\bar{Z}} v_{lj_k}\right) - \sum_{i\in\bar{Z}} v_{l0}v_{lj_i}\left(v_{l0}+v_{lj_1}+v_{lj_2}+\sum_{k\in\bar{Z}} v_{lj_k}\right),$$

  which is equivalent to

$$\sum_{i\in\bar{Z}}\left(-v_{lj_i}v_{l0}(v_{l0}+v_{lj_i}) - 2v_{lj_i}v_{l0}\sum_{h=3}^{i-1} v_{lj_h}\right) \leq -\sum_{i\in\bar{Z}} v_{l0}v_{lj_i}\left(v_{l0}+\sum_{k\in\bar{Z}} v_{lj_k}\right).$$

  With the same arguments as in case 1, the validity follows.

  These five cases cover all possible situations, proving the validity of inequality (15). □

For the last two families of inequalities, which are based on permutation of the indices, we verified no redundancy. That is, for each family, we were able to find a point that violates one of them, but not the rest. In addition, we also confirmed the no redundancy between both families; i.e. we verified that there are points that violate one of the inequalities in family 4 but satisfies all the inequalities in family 5, and vice versa.

Please cite this article in press as: I. Méndez-Díaz, et al., A branch-and-cut algorithm for the latent-class logit assortment problem, Discrete Applied Mathematics (2012), doi:10.1016/j.dam.2012.03.003

4. Solution methodology

LP-based B&C algorithms are currently the most important tool to deal computationally with (mixed) linear integer programming problems.

The approach of a B&C is to recursively split the feasible set into subsets and solve the problem over each part. This procedure generates an enumeration tree where offsprings of a node correspond to the partition of the set associated with the parent node. In each node of the tree, a linear relaxation of the problem is considered by dropping integrality constraints and adding valid inequalities which cut off the fractional solution (i.e., the separation phase).

The performance of a B&C algorithm relies on a combination of several factors. To reduce the number of nodes in the tree, it is important to have good lower and upper bounds of the optimal objective function value, good rules to partition the feasible set, good strategies to explore the tree, and good separation procedures.

Our solution approach is implemented in C++ using the CPLEX package. In what follows we describe the details of the different factors that we consider.

4.1. Primal heuristic

Constructing a feasible solution is the first step in a B&C algorithm. We start by solving the LP relaxation of the root node of the B&C tree, and proceed by setting an integer solution based on the relaxed LP. This solution, once included into the B&C tree, establishes a lower bound for the optimal objective function value from the very beginning, with the objective of reducing the number of nodes.

Our approach to construct a feasible solution follows a greedy argument. First, for each product j, we calculate $Fun_j = w_j \sum_{l=1, j\in C_l}^{L} \lambda_l v_{lj} z^*_{lj}$, where z^*_{lj} is the relaxation optimal solution, and sort the products in decreasing order according to this value. Note that these values correspond to the terms of the objective function in (7) at optimality. The rationale is that in order to guarantee a positive value z_{lj}, we should set the corresponding $y_j = 1$. Based on preliminary computational results, we propose two different approaches depending on the value of the capacity c. In the first case, we consider instances where the capacity satisfies $c > \frac{n}{2}$, which also includes the uncapacitated version of the problem. For this situation, we follow the usual rounding heuristic procedure for the fractional value of y_j.

For the capacitated version with $c \leq \frac{n}{2}$, according to our computational experience, optimal solutions show a tendency to offer at least as many products as $2/3$ of the bounding capacity c. Taking into account this characteristic, the greedy heuristic selects the first $\lfloor 2c/3 \rfloor$ products from the ordered list of Fun_j, and for the remaining third of the capacity it requires that $y_j \geq 0.3$ in order to add product j to the solution (i.e., in order to set $y_j = 1$). We next show the pseudocode of our primal heuristic.

1. For each product j, calculate $Fun_j = w_j \sum_{l=1, j\in C_l}^{L} \lambda_l v_{lj} z^*_{lj}$ and order them decreasingly so that $Fun_{j_1} \geq Fun_{j_2} \geq \cdots \geq Fun_{j_n}$.
2. If $c \leq \frac{n}{2}$ then
   For $k := 1$ to $\lfloor \frac{2}{3} c \rfloor$
   Set $y_{j_k} := 1$;
   For $k := \lfloor \frac{2}{3} c \rfloor + 1$ to c
   If $y_{j_k} \geq 0.3$ then set $y_{j_k} := 1$
   else set $y_{j_k} := 0$;
   else
   For $k := 1$ to c
   If $y_{j_k} \geq 0.5$ then set $y_{j_k} := 1$
   else set $y_{j_k} := 0$;

4.2. Cutting plane generation

Efficient (time-wise) and effective (solution-wise) separation procedures are crucial for the success of a B&C algorithm. Given a fractional LP relaxation solution, one should look for inequalities that are violated at the current solution. Once these valid inequalities are added, the LP-relaxation is resolved. After several computational tests, we choose a strategy that applies 15 rounds of cutting plane generation in the root node, and two rounds in each other node. Each round is followed by a re-solving of the linear relaxation.

Each round starts by applying our inequalities in Proposition 1, in every node of the B&C tree, and without a quantity limit (note that there is a polynomial number of them). Next, we apply the inequalities in Propositions 4, 2 and 3, in this order, with a limit of 300 cuts per tree node per round. The quota of 300 is filled in that order. Starting from inequalities in Proposition 4, since there is a combinatorial number of them, we use a greedy heuristic as the separation procedure. Let (x^*, y^*, z^*) be the current LP relaxation solution. For each consideration set C_l, we construct a list of the elements z^*_{lj} in decreasing order. We choose as $j_1, \ldots, j_{n_l}$ in Proposition 4, the indices of the elements in the ordered list, where $n_l = |C_l|$. If there is a violated inequality (14), we add it to the formulation. Then, we shuffle the indices j_k and look for another violated inequality. We perform several trials limited by an input parameter that we adjusted empirically. After many tests, we set

Please cite this article in press as: I. Méndez-Díaz, et al., A branch-and-cut algorithm for the latent-class logit assortment problem, Discrete Applied Mathematics (2012), doi:10.1016/j.dam.2012.03.003

this parameter at 50, which seems to provide a good compromise between time and effectiveness. Once we reach the bound for inequalities in Proposition 4, we proceed with inequalities in Propositions 2 and 3 (there is a polynomial number of each of them) until we fulfill the 300 quota.

Inequalities in Proposition 5 are only applied in the root node and just in case the quota of 300 is not filled with inequalities in Propositions 1–4.

4.3. Branching rule

The choice of the branching variable has a large effect on the performance of the algorithm. This choice is made even before solving the root node of the B&C tree, and hence cannot be based on values of the decision variables. In our initial computational experiments, we tested various branching strategies that the CPLEX package offers. The results we got with them were quite disappointing. We finally decided to control the order in which CPLEX branches on variables by issuing a priority order. This order assigns a branching priority to all binary variables and the algorithm performs branches on variables with a higher assigned priority number before variables with a lower priority. We have tried with three priorities based on: (a) l_j= number of segments that include product j, (b) the weight $q_j = w_j \sum_{l=1, j\in C_l}^{L} \lambda_l v_{lj}$, and (c) the weight $l_j q_j$. Note that l_j is an indicator of the importance of product j in the market, and q_j is a measure of the *a priori* contribution of product j to the objective function (7) in (LC-MNL). According to our computational tests, the last priority dominates the other ones in terms of the number of subproblems explored and total CPU time. We allow CPLEX to choose automatically branching directions (up or down) as well as node selection strategy.

5. Computational results

In this section, we evaluate the performance of the derived inequalities in a B&C algorithm both from revenue and computational time perspectives. First, we introduce a simple heuristic that we use as benchmark. Next, we report results for retailing applications of the problem, where we choose a single offer set for the whole selling period. Then, we report results of embedding our algorithm in the CDLP for choice-based RM.

In the previous section we described all the design decisions that we made in order to come up with our B&C, including primal heuristic, cutting plane generation, and branching rule. Those decisions were the outcome of trials on an exhaustive collection of problem instances. Once we tuned our algorithm as explained, we ran the set of experiments that follow, all of them under the same algorithm design. The experiments were conducted on a PC with processor Intel(R)Core(TM) 2 Quad CPU of 2.39 Ghz, RAM of 3.24 GB and operating system Windows XP. The algorithms were coded in C++ and linked to CPLEX 12.2 optimization routines.

5.1. Benchmark procedure: a greedy heuristic (GH)

Based on the promising results presented in our former paper [18], we consider the greedy heuristic there as a benchmark to assess the tradeoff between quality of the solutions obtained and computational requirements. The main idea of the heuristic is to iteratively add products to the solution based on the marginal benefit that each product brings into the objective function, until no further improvement can be achieved. This heuristic requires small computational effort (its worst-case complexity is $O(Ln^2)$, with L typically smaller than n) and in general produces good quality solutions. We show below the sketch of the heuristic. Here, Value(S) stands for the outcome of the objective function in (2) calculated over set S.

1. Let $S' \subseteq N$ be the set of products j with no assigned value for y_j.
2. Compute $j_1^* := \arg\max_{j\in S'}\{\sum_{l=1}^{L} \lambda_l \frac{w_j v_{lj}}{v_{lj}+v_{l0}}\}$. Set $S := \{j_1^*\}$ and $S' := S' - \{j_1^*\}$.
3. **Repeat**

   Compute $j^* := \arg\max_{j\in S'} \left\{ \sum_{l=1}^{L} \lambda_l \frac{\sum_{i\in C_l\cap(S\cup\{j\})} w_j v_{li}}{\sum_{i\in C_l\cap(S\cup\{j\})} v_{li}+v_{l0}} \right\}$

   If Value($S \cup \{j^*\}$) > Value(S), then $S := S \cup \{j^*\}$ and $S' := S' - \{j^*\}$.

   until S is not modified and $|S| \le c$
4. For all $j \in S$, set $y_j := 1$. For $j \notin S$, set $y_j := 0$.

5.2. Retail examples

We have randomly generated instances for three families of problems for this setting.

5.2.1. Type 1 problems

For type 1, we have $n = 500$ products and $L = 200$ segments. The arrival rates λ_l are Unif[0,1], the preference weights $v_{lj}, j = 1, \ldots, n$ are discrete Unif[0,10], and the attractiveness of the no purchase alternative v_{l0} is Unif[0,4]. Revenues

Please cite this article in press as: I. Méndez-Díaz, et al., A branch-and-cut algorithm for the latent-class logit assortment problem, Discrete Applied Mathematics (2012), doi:10.1016/j.dam.2012.03.003

Table 1
Computational times (in seconds) and percentage suboptimality gap of GH for the uncapacitated and capacitated with 50%, type 1 problem.

Price range	Uncapacitated				Capacitated 50%			
	CPLEX	B&C	GH		CPLEX	B&C	GH	
	Time	Time	Time	% Gap	Time	Time	Time	% Gap
100–150	4.75	1.90	38.38	0.13	694.03 (2)	866.26 (3)	29.56	0.18
100–200	187.90	3.54	33.66	0.52	****	53.89	28.03	0.57
100–250	97.50 (8)	5.17	30.05	0.52	1183.96 (3)	17.39	27.35	0.54
100–300	260.02 (9)	5.37	27.49	0.74	302.45 (8)	5.76	26.32	0.75
100–350	284.72 (9)	5.32	27.13	0.69	474.05 (9)	6.82	26.19	0.69

Table 2
Computational times (in seconds) and percentage gain gaps for the type 1 problem, with $c = 100$. The gaps are relative to the GH results and refer to the best lower bounds found by B&C and CPLEX in 15 s.

Price range	GH		
	Time	% Gap B&C	% Gap CPLEX
100–150	7.64	0.32	0.33
100–200	7.24	1.12	0.93
100–250	7.01	0.96	0.74
100–300	6.81	1.16	0.93
100–350	6.79	1.19	0.88

$w_j, j = 1, \ldots, n$, vary uniformly within a range that changes for different instances (to be specified below). The overlapping between the segments occur according to: $C_2 \subset C_1$, $C_4 \subset C_3, \ldots, C_L \subset C_{L-1}$. Each consideration set C_l, $l = 1, 3, 5, \ldots, L-1$, consists of 6 products chosen randomly among the 500 and hence the intersection between them may be nonempty. For each of those C_l, the 3 lowest generated elements w_j define C_{l+1}. This kind of overlapping is representative of substitution patterns where C_l, l odd, includes products for price insensitive customers, and C_{l+1} includes the three cheapest elements considered by price sensitive customers.

Table 1 (left) shows results for the uncapacitated, type 1 problem. We generate 10 instances for each price range. We report the average computational times (in seconds) for the standard branch-and-cut built in CPLEX (that only uses general purpose cuts), for our B&C, and for the plain GH. Stars indicate that none of the instances could be solved within a half hour of CPU time, and values in parenthesis show the number of instances solved to optimality within that computational burden.[4] The computational times are averaged over the time of instances that finished in less than a half hour. We also report the average percentage suboptimality gap of GH. This average is computed by taking for each instance the revenue gap between the optimal solution (when B&C finds it) and GH, and between the best lower bound obtained by B&C and GH (when B&C fails to finish in half hour of CPU time). We see that B&C significantly reduces both CPLEX and GH times. Recall that the small suboptimality gap of GH may be significant from a profit potential perspective for retail operations that typically work with very high gross margins and narrow net profits. Table 1 (right) shows results for a capacitated version of type 1 problems, with $c = 250$ (i.e., with capacity for 50% of the products). We also verify a better revenue performance of B&C. Here, CPLEX cannot even obtain an optimal solution for any of the instances in half hour. Except for the first case, our B&C requires a computational budget similar to or lower than the one of GH, yet recovering the average 0.54% left on the table by the latter.

When considering the type 1 problems with $c = 100$ (i.e., with capacity for 20% of the products), neither CPLEX nor our B&C were able to solve a single instance to optimality in half hour. Hence, we tried a different test here. Table 2 shows the computational times of GH, and the gain gaps of the best bounds reached by B&C and CPLEX in 15 s of CPU time. The baseline is the revenue obtained by GH. We observe that even in 15 s of B&C, we can recover around 1% of revenues left on the table by GH. Note also that the best feasible solution found by our B&C within the 15 s is better than the one found by CPLEX.

As a general observation for type 1 problems, note that as they become more capacity constrained (i.e., as we reduce c), the instances tend to be easier to solve when the price dispersion is big; the intuition being that when we have a small number of products to display, we would certainly tend to offer high price products (for similar demand segment intensities), which are easier to choose when the dispersion of prices is wider. This is particularly true for CPLEX and our B&C.

5.2.2. Type 2 problems

The second family of problems has a more general overlapping structure, of the form $C_l \cap C_{l+1} \neq \emptyset$ for $l = 1, \ldots, L-1$, over $n = 500$ products. We vary the number of segments $L \in \{30, 50, 70, 100\}$. There is a parameter $k \in \{3, 6, 9, 12\}$ that defines the minimum cardinality of the intersection between C_l and C_{l+1}. The random generation of the instances proceeds as follows: We start picking 15 products to build C_1. We take the last k chosen products, and pick another $15 - k$ to complete

[4] No values in parenthesis mean that all 10 instances were solved to optimality.

Please cite this article in press as: I. Méndez-Díaz, et al., A branch-and-cut algorithm for the latent-class logit assortment problem, Discrete Applied Mathematics (2012), doi:10.1016/j.dam.2012.03.003

Table 3
Computational times (in seconds) and percentage suboptimality gap for the uncapacitated type 2 problem.

Instance		CPLEX	B&C	GH	
L	k	Time	Time	Time	% Gap
30	3	1.36	0.44	3.82	1.07
30	6	3.17	1.08	2.83	0.85
30	9	1.68	0.39	1.91	0.96
30	12	0.55	0.17	0.96	0.23
50	3	6.09	1.95	7.72	1.16
50	6	16.06	4.73	6.76	0.90
50	9	11.97	2.25	5.01	0.67
50	12	1.62	0.48	2.22	0.43
70	3	212.04 (9)	61.97	12.77	0.74
70	6	126.01	14.37	11.27	0.83
70	9	43.57	3.84	7.95	0.75
70	12	1.75	0.66	4.22	0.52
100	3	823.37 (8)	95.04 (9)	19.77	0.87
100	6	554.51 (6)	185.88 (9)	18.10	0.73
100	9	397.65 (9)	145.75	14.07	0.70
100	12	2.87	1.21	7.89	0.54

Table 4
Computational times (in seconds) and percentage suboptimality gap for the capacitated type 2 problem, with $c = 250$.

Instance		CPLEX	B&C	GH	
L	k	Time	Time	Time	% Gap
30	3	1.00	0.37	3.82	1.07
30	6	1.90	0.85	2.83	0.85
30	9	0.97	0.32	1.91	0.96
30	12	0.30	0.13	0.96	0.23
50	3	4.53	1.68	7.73	1.16
50	6	11.52	4.32	6.76	0.90
50	9	7.26	1.82	5.01	0.67
50	12	1.14	0.37	2.22	0.43
70	3	312.19 (9)	55.51	12.78	0.74
70	6	102.61	13.76	11.27	0.83
70	9	41.97	3.78	7.96	0.75
70	12	1.79	0.65	4.22	0.52
100	3	795.75 (6)	104.55 (9)	19.76	0.87
100	6	494.42 (7)	212.33 (9)	18.11	0.73
100	9	418.63 (9)	149.40	14.07	0.70
100	12	2.86	1.22	7.89	0.54

C_2. Next, we choose the last k of C_2, and add $15 - k$ to build C_3, and so on. Arrival rates λ_l are Unif[0,1], preference weights $v_{lj}, j = 1, \ldots, n$, are discrete Unif[0,10], no-purchase attractiveness v_{l0} is discrete Unif[0,4], and revenues w_j are Unif[100, 200]. For each combination (L, k), we generate 10 instances and report the average computational times (in seconds) and percentage suboptimality gap of GH. In Table 3 we report results for the uncapacitated version of the problem. For $L = 30$ and $L = 50$, we see that the time performance of B&C is better than the other two methods under consideration. For the bigger instances (i.e., $L = 70$ and $L = 100$), GH is usually faster than CPLEX and B&C, but revenue-wise it is off by 0.71%. Here, B&C also seems to provide a good compromise time- and revenue-wise between CPLEX and GH. Observe that when the number of overlapping products is small, the instances are harder: They take longer for all methods (particularly for the exact methods), and the revenue performance of the heuristic deteriorates.

For the capacitated version of type 2 problems when $c = 250$ (Table 4) (i.e., $c = 50\%n$), we observe similar results to the uncapacitated version. This is due to the fact that in most of the cases, the number of products offered in the solutions to the uncapacitated problems of Table 3 does not exceed 50% of the capacity.

Different from type 1 problems, here the capacitated version of the problem at $c = 20\%n$ could be solved to optimality within 30 min in most of the cases (see Table 5). The performance of the exact methods diminishes while the number of segments increases. Even when optimality is not reached, the best solution got within 30 min of computational time recovers 1% of the revenues left behind by GH. It is still true that when the number of overlapping products increases, it is easier to solve the instances exactly, and the gap with respect to GH reduces.

Given the difficulty of type 2 instances when capacity is tight (i.e., $c = 20\%n$), in Table 6 we show the computational times of GH, and the gain gaps of the best bounds reached by B&C and CPLEX in 15 s of CPU time. The baseline is the revenue obtained by GH. We observe that even in 15 s of B&C, we can recover around 0.8% of revenues left on the table by GH.

Please cite this article in press as: I. Méndez-Díaz, et al., A branch-and-cut algorithm for the latent-class logit assortment problem, Discrete Applied Mathematics (2012), doi:10.1016/j.dam.2012.03.003

Table 5
Computational times (in seconds) and percentage suboptimality gap for the capacitated type 2 problem, with $c = 100$.

Instance		CPLEX	B&C	GH	
L	k	Time	Time	Time	% Gap
30	3	1.25	0.38	3.67	1.08
30	6	2.12	0.84	2.83	0.85
30	9	0.93	0.32	1.91	0.96
30	12	0.29	0.13	0.95	0.23
50	3	371.53 (7)	272.63 (7)	5.43	1.26
50	6	163.63	54.24	5.89	0.93
50	9	8.73	1.82	4.92	0.67
50	12	1.12	0.37	2.22	0.43
70	3	***	***	7.11	0.90
70	6	514.45 (2)	84.41 (2)	7.57	0.94
70	9	456.01 (8)	65.54 (9)	6.47	0.81
70	12	2.04	0.82	4.13	0.52
100	3	***	***	9.41	1.15
100	6	***	***	10.00	0.93
100	9	***	***	8.31	0.94
100	12	24.33	19.35	5.32	0.68

Table 6
Computational times of GH (in seconds) and percentage gain gaps for the type 2 problem, with $c = 100$. The gaps are relative to the GH results and refer to the best lower bounds found by B&C and CPLEX in 15 s.

Number of segments L	GH		
	Time	% Gap B&C	% Gap CPLEX
50	5.43	1.26	1.26
50	5.89	0.93	0.93
50	4.92	0.67	0.67
50	2.22	0.43	0.43
70	7.11	0.86	0.89
70	7.57	0.93	0.92
70	6.47	0.81	0.81
70	4.13	0.52	0.52
100	9.41	1.08	0.52
100	10.00	0.81	0.33
100	8.31	0.84	0.71
100	5.32	0.67	0.68

Note also that the best feasible solution found by our B&C within the 15 s is generally better than the one found by CPLEX, especially when the problem is big (i.e., $L = 100$).

5.2.3. Type 3 problems

Finally, the set of instances type 3 has a less restrictive overlapping pattern. All but three segments are guaranteed to have intersection with at least another three segments. There are $n = 500$ products. The number of segments is $L \in \{50, 100, 200\}$. There is a parameter $n_l \in \{5, 10, 15\}$ that defines the cardinality of each segment C_l. The arrival rates λ_l are Unif[0,1], the preference weights $v_{lj}, j = 1, \ldots, n$ are discrete Unif[0,10], and the attractiveness of the no purchase alternative v_{l0} is Unif[0,4]. Revenues $w_j, j = 1, \ldots, n$, vary uniformly within 100 and 150.

The generation process of random type 3 instances proceeds as follows: For the first segment C_1, n_1 products are picked among the 500 products. For C_2, we pick 3 random elements from C_1, and complete with $n_2 - 3$ products (without replacement). For C_3, we pick 3 random elements from $C_1 \cup C_2$, and complete with $n_3 - 3$ products (without replacement). For C_l, $l = 4, \ldots, L$, we choose randomly 3 segments among $\{1, \ldots, l-1\}$, and randomly take one product from each of them. The remaining $n_l - 3$ products for C_l are generated randomly without replacement. For each combination (L, k), we generate 10 instances and report the average time (in seconds) and the suboptimality gap of GH.

In Table 7 we fix the number of segments in $L = 100$, and vary the number of products per segment between 5, 10 and 15. We notice that the uncapacitated and capacitated instances (with $c = 50\%n$ and $c = 20\%n$) do not show differences since the optimum of the uncapacitated version includes just few products in the assortment. In general, less than 25% of the products were included in the optimal uncapacitated solution, which explains that in only few instances there was a difference in the optimum of the three cases (and hence, the average suboptimality gap of GH is the same for the three cases). In general, computational times are increasing in the size of the segments, but our B&C cuts the CPLEX times by half when $n_l = 10$ and $n_l = 15$, and are roughly comparable to the ones of the heuristic, yet recovering the revenue left behind

Please cite this article in press as: I. Méndez-Díaz, et al., A branch-and-cut algorithm for the latent-class logit assortment problem, Discrete Applied Mathematics (2012), doi:10.1016/j.dam.2012.03.003

Table 7
Computational times (in seconds) and percentage suboptimality gap of GH for type 3 instances: uncapacitated and capacitated with $c = 250$ and $c = 100$.

Segment cardinality	Uncapacitated				Capacitated 50%				Capacitated 20%			
	CPLEX	B&C	GH		CPLEX	B&C	GH		CPLEX	B&C	GH	
n_l	Time	Time	Time	% Gap	Time	Time	Time	% Gap	Time	Time	Time	%Gap
5	0.25	0.32	2.98	0.37	0.25	0.33	2.99	0.37	0.30	0.39	2.98	0.37
10	4.75	2.21	13.13	1.34	4.74	2.20	13.05	1.34	4.87	2.21	13.09	1.34
15	106.45	38.21	21.10	0.88	81.50	43.02	21.09	0.88	133.75	46.93	19.25	0.88

Table 8
Computational times (in seconds) and percentage suboptimality gap of GH for the uncapacitated and capacitated with $c = 250$ and $c = 100$.

Number of segments L	Uncapacitated				Capacitated 50%				Capacitated 20%			
	CPLEX	B&C	GH		CPLEX	B&C	GH		CPLEX	B&C	GH	
50	3.15	1.08	10.51	1.41	3.29	1.07	10.5	1.41	3.56	2.13	9.80	1.42
100	143.22	42.25	21.89	1.34	154.12	43.44	21.89	1.34	204.39	56.05	19.66	1.35
200	465.1 (8)	144.8 (9)	46.07	0.79	405.7 (9)	160.2 (9)	45.63	0.79	599.8 (7)	211.5 (9)	40.93	0.80

by GH, which exceeds 1.3% in some of the cases. The number of products in the optimal offer set is also increasing in the size of the segments.

Table 8 shows how the solution changes when we increase the number of segments. We can notice that the uncapacitated version of the problem offers just few products at optimality, in fact less than 50% (and in few instances even less than 20%) of the 500 available. The computational times augment with the number of segments, and exceed 30 min for the exact methods in few of the instances. In general, our B&C takes one third of the computational time of CPLEX, and is able to solve all but three of the instances within the 30 min (better than CPLEX, that fails time-wise in 6 of the instances).

5.3. CDLP examples

The choice-based, deterministic, LP model (CDLP) is an approximation for the dynamic programming formulation of the network RM problem under customer choice behavior. We refer the reader to [13,18] for details. We include here a quick overview: Each product is an itinerary-fare-class combination. The fare for product j is r_j, and we denote by $R(S)$ the expected revenue generated from an arriving customer when set S is displayed, i.e., $R(S) = \sum_{j=1}^{n} r_j P_j(S)$. Given an arrival, let $Q_i(S)$ denote the conditional probability of using a unit of capacity on leg i when set S is offered. The vector of capacity consumption (conditional) probabilities is given by $Q(S) = AP(S)$, where $P(S) = (P_1(S), \ldots, P_n(S))$ is the vector of purchase probabilities as in (1) and A is the incidence matrix that describes the topology of the flight network. The decision variables $t(S)$ represent the length of time during which to offer set S. Let c be the vector of initial capacities for the flight legs, and T be the length of the booking horizon. The CDLP formulation is given by

$$\begin{aligned} V^{\text{CDLP}} = \max & \sum_{S\subset N} \lambda R(S) t(S) \\ \text{s.t.} & \sum_{S\subset N} \lambda Q(S) t(S) \le c, \\ & \sum_{S\subset N} t(S) \le T, \\ & t(S) \ge 0, \quad \forall S \subset N. \end{aligned} \tag{16}$$

Given the exponential number of variables, the CDLP is solved via column generation. As it was mentioned in Section 1, under a latent class, MNL model, the column generation subproblem turns out to be equivalent to our problem (2):

$$\max_{\bar{y}\in\{0,1\}^n} \left\{ \sum_{j=1}^{n} (r_j - A_j^{\mathsf{T}}\pi) y_j \left(\sum_{l=1}^{L} \frac{\lambda_l v_{lj}}{\sum_{i\in C_l} v_{li} y_i + v_{l0}} \right) \right\} - \sigma,$$

where π and σ are the dual variables of the capacity and time constraints in (16), respectively.

In this section, we focus on the difficult instances identified in Miranda Bront et al. [18, Section 6.3]. The approach proposed there to solve CDLP by column generation was first trying the greedy heuristic to identify an entering column to the base. If GH does not find an entering column, then we use the exact CPLEX procedure. This approach is labeled GH+CPLEX in Table 9 here. Because of the long computational times, in our previous paper we suggested to solve the problem approximately. Specifically, we stopped the column generation algorithm when GH could no longer find an entering column. This approximation reduced the times to less than a second, but meant a revenue loss between 0.55% and 1.41%.

Please cite this article in press as: I. Méndez-Díaz, et al., A branch-and-cut algorithm for the latent-class logit assortment problem, Discrete Applied Mathematics (2012), doi:10.1016/j.dam.2012.03.003

Table 9
Computational times (in seconds) for the Hub-and-Spoke instance reported in [18, Section 6.3].

Instance		GH + CPLEX	B&C	GH + B&C
α	v_0			
0.6	(1.5)	75.82	11.23	3.22
0.8	(1.5)	876.63	8.57	0.86
1.0	(1.5)	3184.25	76.32	41.99
1.2	(1.5)	1211.33	26.25	18.14
1.4	(1.5)	109.62	4.41	2.27

Table 9 provides strong support in favor of our new computational proposal. All the instances were solved to optimality using our B&C (second column of results), reducing the previous computational times by factors between 7 and 100. Moreover, combining GH+B&C to find an entering column (i.e., trying with GH first, and when it fails, applying our B&C), we can solve CDLP to optimality within times between 0.86 and 42 s and yet recover the revenue loss.[5]

6. Conclusions

While assortment planning under consumer substitution effects has been a prolific area of research in the last few years, the more general setting where customers belong to unobservable segments (i.e., latent classes) and choose according to a MNL choice model remains a challenging problem to solve in practice. This difficulty stems from two facts: (1) The estimation procedure is more complex. Since there are usually many parameters involved, the success of the methods (e.g., maximum likelihood) will strongly depend on counting with high volumes of data which is indeed feasible in e-commerce; and (2) The difficulty of the optimization procedure which has to deal with an NP-Hard problem.

In this paper, we focused on the optimization part and developed a Branch-and-Cut (B&C) algorithm that relies on five families of specific valid inequalities that we derived for the problem. The algorithm was tested over hundreds of randomly generated instances, and over instances previously reported in the literature.

Our B&C was shown to dominate the standard CPLEX algorithm both from a computational and revenue perspectives. When compared to a simple greedy heuristic, our B&C recovered around 1% of the revenues left behind even when constraining its computational burden to 15 s. Given the goodness of its behavior, we think that it is a very promising approach to be pursued in practice, specially for online retailing and airline revenue management where computational speed is a critical factor to assess the viability of an optimization procedure.

As for additional work, it would be interesting to characterize the facets of the MIP polyhedra for the LC-MNL formulation. In particular, the five families of inequalities that we develop in this paper are defined at the intra-customer-segment level. Inferring inequalities that span two or more segments, and analyzing its computational performance is worthy of further study.

Acknowledgments

This paper was completed while the third author was visiting Universidad Torcuato di Tella, Buenos Aires, Argentina. It was partially funded by FONCyT grant PICT 2006-01070 (Proyecto Raíces) from the Ministerio de Ciencia, Tecnología e Innovación Productiva, República Argentina, and by project UBACyT X143, from Universidad de Buenos Aires, Argentina.

References

[1] M. Ben-Akiva, S. Lerman, Discrete Choice Analysis: Theory and Applications to Travel Demand, sixth ed., The MIT Press, Cambridge, MA, 1994.
[2] G. Cachon, C. Terwiesch, Y. Xu, Retail assortment planning in the presence of consumer search, Manufacturing & Service Operations Management 7 (2005) 330–346.
[3] N. Cardell, F. Dunbar, Measuring the societal impacts of automobile downsizing, Transportation Research Part A 14 (1980) 423–434.
[4] F. Caro, J. Gallien, Dynamic assortment with demand learning for seasonal consumer Goods, Management Science 53 (2007) 276–292.
[5] P. Chintagunta, D. Jain, N. Vilcassim, Investigating heterogeneity in brand preference in logit models for panel data, Journal of Marketing Research 28 (1991) 417–428.
[6] V. Farias, S. Jagabathula, D. Shah, A new approach to modeling choice with limited data, Working paper, MIT Sloan School of Management, Cambridge, MA, 2009.
[7] M. Fisher, Rocket science retailing: the 2006 Philip McCord Morse lecture, Operations Research 57 (2009) 527–540.
[8] M. Fisher, R. Vaidyanathan, An algorithm and demand estimation procedure for retail assortment optimization, Working paper, OPIM Department, The Wharton School, University of Pennsylvania, 2009.
[9] G. Gallego, G. Iyengar, R. Phillips, A. Dubey, Managing flexible products on a network, Technical Report CORC TR-2004-01, Department of Industrial Engineering and Operations Research, Columbia University, 2004.

[5] These instances were run on the same equipment used in our former paper [18]: A SUN UltraSparc III server (CPU of 1 GHz, RAM of 2 GB, operating system SunOS 5.9). The algorithms were coded in C++, and compiled using gcc version 3.4.6 (GNU compiler). The code was linked to the CPLEX 8.1 optimization routines used in [18].

[10] V. Goyal, R. Levi, D. Segev, Near-optimal algorithms for the assortment planning problem under dynamic substitution and stochastic demand, Working paper, Operations Research Center, MIT, Cambridge, MA, 2009.
[11] W. Kamakura, G. Russell, A probabilistic choice model for market segmentation and elasticity structure, Journal of Marketing Research 26 (1989) 379–390.
[12] G. Kök, M. Fisher, R. Vaidyanathan, Assortment planning: review of literature and industry practice, in: N. Agrawal, S. Smith (Eds.), Retail Supply Chain Management: Quantitative Models and Empirical Studies, in: International Series in Operations Research & Management Science, Springer, 2009, pp. 99–154 (Chapter 6).
[13] Q. Liu, G. van Ryzin, On the choice-based linear programming model for network revenue management, Manufacturing & Service Operations Management 10 (2008) 288–310.
[14] S. Mahajan, G. van Ryzin, Stocking retail assortments under dynamic consumer substitution, Operations Research 49 (2001) 334–351.
[15] D. McFadden, K. Train, Mixed MNL models for discrete response, Journal of Applied Econometrics 15 (2000) 447–470.
[16] N. Meggido, Combinatorial optimization with rational objective functions, Mathematics of Operations Research 4 (1979) 414–424.
[17] J. Meissner, A. Strauss, Choice-based network revenue management under weak market segmentation, Working paper, Lancaster University Management School, Bailrigg, Lancaster, UK, 2009.
[18] J. Miranda Bront, I. Méndez-Díaz, G. Vulcano, A column generation algorithm for choice-based network revenue management, Operations Research 57 (2009) 769–784.
[19] O. Prokopyev, C. Meneses, C. Oliveira, P. Pardalos, On multiple-ratio hyperbolic 0–1 programming problems, Pacific Journal of Optimization 1 (2) (2005) 327–345.
[20] R. Ratliff, L. Weatherford, A review of revenue management methods with dependent demands, in: AGIFORS Cargo and RM Study Group Meeting, Amsterdam, Netherlands, May 2009.
[21] P. Rusmevichientong, M. Shen, D. Shmoys, Dynamic assortment optimization with multinomial logit choice model and capacity constraint, Operations Research 58 (2010) 1666–1680.
[22] P. Rusmevichientong, D. Shmoys, H. Topaloglu, Assortment optimization with mixtures of logits, Working paper, School of Operations Research and Information Engineering, Cornell University, Ithaca, NY, 2010.
[23] P. Rusmevichientong, H. Topaloglu, Robust assortment optimization under the multinomial logit choice model, Working paper, School of Operations Research and Information Engineering, Cornell University, Ithaca, NY, 2010.
[24] D. Saure, A. Zeevi, Optimal dynamic assortment planning, Working paper, Graduate School of Business, Columbia University, New York, NY, 2009.
[25] S. Smith, Optimizing retail assortments for diverse customer preferences, in: N. Agrawal, S. Smith (Eds.), Retail Supply Chain Management: Quantitative Models and Empirical Studies, in: International Series in Operations Research & Management Science, Springer, 2009, pp. 183–205 (Chapter 8).
[26] S. Smith, N. Agrawal, Management of multi-item retail inventory systems with demand substitution, Operations Research 48 (2000) 50–64.
[27] K. Talluri, G.J. van Ryzin, The Theory and Practice of Revenue Management, Kluwer Academic Publishers, New York, NY, 2004.
[28] K. Train, Discrete Choice Methods with Simulation, Cambridge University Press, New York, NY, 2003.
[29] G. van Ryzin, S. Mahajan, On the relationship between inventory costs and variety benefits in retail assortments, Management Science 45 (1999) 1496–1509.
[30] T. Wu, A note on a global approach for general 0–1 fractional programming, European Journal of Operational Research 101 (1997) 220–223.

Please cite this article in press as: I. Méndez-Díaz, et al., A branch-and-cut algorithm for the latent-class logit assortment problem, Discrete Applied Mathematics (2012), doi:10.1016/j.dam.2012.03.003