An Enhanced Concave Program Relaxation for Choice Network Revenue Management

Joern Meissner
Arne Strauss
Kalyan Tallury

January 2011

Barcelona GSE Working Paper Series

Working Paper n° 534

An Enhanced Concave Program Relaxation for Choice Network Revenue Management

Joern Meissner, Arne Strauss[*], Kalyan Talluri[†]

20 January 2011

Abstract

The network choice revenue management problem models customers as choosing from an offer-set, and the firm decides the best subset to offer at any given moment to maximize expected revenue. The resulting dynamic program for the firm is intractable and approximated by a deterministic linear program called the $CDLP$ which has an exponential number of columns. However, under the choice-set paradigm when the segment consideration sets overlap, the $CDLP$ is difficult to solve. Column generation has been proposed but finding an entering column has been shown to be NP-hard. In this paper, starting with a concave program formulation based on segment-level consideration sets called $SDCP$, we add a class of valid inequalities called product cuts, that project onto subsets of intersections. In addition we propose a natural direct tightening of the $SDCP$ called $\kappa SDCP$, and compare the performance of both methods on the benchmark data sets in the literature. Both the product cuts and the $\kappa SDCP$ method are very simple and easy to implement, work with general discrete choice models and are applicable to the case of overlapping segment consideration sets. In our computational testing $SDCP$ with product cuts achieves the $CDLP$ value at a fraction of the CPU time taken by column generation and hence has the potential to be scalable to industrial-size problems.

Keywords: bid prices, yield management, heuristics, discrete-choice, network revenue management

1 Introduction and literature review

Revenue management is the control of the sale of a limited quantity of a resource (hotel rooms for a night, airline seats, advertising slots etc.) to a heterogenous population with different valuations for a unit of the resource. The resource is perishable, and for simplicity's sake, we assume that it perishes at a fixed point of time in the future. Customers are independent of each other and arrive randomly during a sales period, and demand one unit of resource each. The sales process is online, so the firm has to decide if it wishes to sell (at a specific price) or not, the tradeoff being selling too much at a too low price early and running out of capacity, or, rejecting too many low valuation customers and ending up with excess unsold inventory.

[*]Lancaster University Management School, Lancaster LA1 4YX, United Kingdom.
[†]ICREA & Universitat Pompeu Fabra, Ramon Trias Fargas 25-27, 08005 Barcelona, Spain.

In industries such as hotels and airlines the products consume bundles of different resources (multi-night stays, multi-leg itineraries) and the decision to accept or reject a particular product at a certain price depends on the future demands and revenues for all the resources used by the product (and indirectly potentially on all resources in the network, depending on the network and product structure). Network revenue management (network RM) controls acceptance/reject decisions for demand requests for the entire network. Chapter 3 of Talluri and van Ryzin (2004b) contains all the necessary background on network RM.

Revenue management incorporating more realistic models of customer behavior, such as customers choosing products from an offer set, have recently become popular, initiated by work of Talluri and van Ryzin (2004a) on the single-resource problem. Many network RM extensions of such models have recently been proposed: Zhang and Cooper (2005), Gallego et al. (2004), Liu and van Ryzin (2008) propose a deterministic linear programming approximation ($CDLP$), Kunnumkal and Topaloglu (2010) a Lagrangian relaxation, Zhang and Adelman (2009) and Meissner and Strauss (2008) affine relaxation approximations. In many cases these are modifications of older methods proposed for network RM with the so-called independent class assumption. The incorporation of customer choice models, however, makes the approximations considerably more difficult to solve. The $CDLP$ formulation has an exponential number of variables and the solution strategy is to use column generation, but finding an entering column is computationally easy only in a limited number of cases. While the Lagrangian and affine relaxation approximations are closer to the dynamic program than the $CDLP$, their solution is no easier; for instance the affine relaxation approximation involves generating columns by solving a non-linear mixed-integer program. The objective of this paper is to find new methods to solve or approximate $CDLP$ quickly when the segment consideration sets overlap.

Given the difficulties in solving these models, Talluri (2010) proposed a concave programming formulation that is weaker than the upper bound resulting from the ($CDLP$) from Gallego et al. (2004) and Liu and van Ryzin (2008), but coincides for non-overlapping segments. The advantage is that the method is tractable for any choice model whenever the number of elements in a segment's consideration set is not too large. In this paper we extend the method to obtain progressively tighter relaxations of ($CDLP$). We add valid inequalities derived from a projection onto subsets of intersections, and refer to them as *product cuts*. These cuts are easy to generate and work for general discrete choice models—in fact this is the only approach that we know of that can handle general discrete choice models and overlapping segment consideration sets. We report extensive computational results showing their performance on various types of networks. We contrast the results with an extension of $SDCP$ called $\kappa SDCP$. In our numerical testing, $SDCP$ with product cuts achieves the $CDLP$ value at a fraction of the CPU time taken by column generation (Table 13) making $CDLP$ tractable even for industrial-size problems.

The remainder of the paper is organized as follows: In §2 we introduce the notation, the demand model and the basic dynamic program. In §3 we state the ($CDLP$) and ($SDCP$) approximations of the dynamic program, followed by the presentation of the main computational approaches that we propose in this paper in §4. §5 contains our numerical results using the new methods, and we present our conclusions in §6.

2 Model and Notation

A product is a specification of a price and a combination of resources to be consumed. For example, a product could be an itinerary-fare class combination for an airline network, where an itinerary

is a combination of flight legs; in a hotel network, a product would be a multi-night stay for a particular room type at a certain price point. Time is discrete and assumed to consist of T intervals, indexed by t. We assume that the booking horizon begins at time 0 and that all the resources perish instantaneously at time T. We make the standard assumption that the time intervals are fine enough so that the probability of more than one customer arriving in any single time period is negligible. The underlying network has m resources (indexed by i) and n products (indexed by j) of resources, and we refer to the set of all resources as I and the set of all products as J. Product j uses a subset of resources, and is identified (possibly) with a set of sale restrictions or features and a revenue of r_j. A resource i is said to be in product j ($i \in j$) if j uses resource i. The resources used by j are represented by $a_{ij} = 1$ if $i \in j$, and $a_{ij} = 0$ if $i \notin j$, or alternately with the 0-1 incidence vector A_j of product j. Let A denote the resource-product incidence matrix; columns of A are then A_j. We denote capacity on resource i at time t as $c_{i,t}$ and the vector of capacities as $\vec{c}_t$, so the initial set of capacities at time 0 is $\vec{c}_0$. The vector $\vec{1}$ is a vector of all ones, and $\vec{0}$ is a vector of all zeroes (dimension appropriate to the context).

We represent a mathematical program or a dynamic program by a label that also serves as the optimal value of the program. For example, $(CDLP)$ represents the deterministic linear program (described below) and $CDLP$ represents the objective function value of the optimal solution.

2.1 Demand model

The demand model is a latent finite segment-mixture model with overlapping segment consideration sets. We assume there are $L > 1$ underlying segments, each with distinct purchase behavior. In each period, there is a customer arrival with probability λ. A customer belongs to segment l with probability p_l. We denote $\lambda_l = p_l\lambda$ and assume $\sum_l p_l = 1$, so $\lambda = \sum_l \lambda_l$. Define $\vec{\lambda} = [\lambda_1, \ldots, \lambda_L]$. We are assuming time-homogenous arrivals (homogenous in rates and segment mix), but the model and all solution methods in this paper can be transparently extended to the case when rates and mix change by period. Each segment l has a *consideration set*, a subset of products $C_l \subseteq J$ that it considers for purchase. We assume this consideration set is known to the firm (by a previous process of estimation and analysis) and the consideration sets for different segments can overlap.

In each period the firm offers a subset S of its products for sale, called the *offer set*. Given an offer set S, an arriving customer purchases a product j in the set S or decides not to purchase. The no-purchase option is indexed by 0 and is always present for the customer.

A segment-l customer is indifferent to a product outside his consideration set; i.e., his choice probabilities are not affected by products offered outside the consideration set. A segment-l customer purchases $j \in S$ with given probability $P_j^l(S)$. This is a set-function defined on all subsets of J. For the moment we assume these set functions are given by an oracle; it could conceivably be given by a simple formula such as the multinomial-logit model. Whenever we specify probabilities for a segment l for a given offer-set S, we just write it with respect to $S_l := C_l \cap S$ (note that $P_j^l(S) = P_j^l(S_l)$). We define the vector $\vec{P}^l(S) = [P_1^l(S_l), \ldots, P_n^l(S_l)]$.

Given a customer arrival, and an offer set S, the probability that the firm sells $j \in S$ is then given by $P_j(S) = \sum_l p_l P_j^l(S_l)$ and makes no sale with probability $P_0(S) = 1 - \sum_{j \in S} P_j(S)$. We define the vector $\vec{P}(S) = [P_1(S), \ldots, P_n(S)]$. Notice that $\vec{P}(S) = \sum_l p_l \vec{P}^l(S)$. We define the vectors $\vec{Q}^l(S) = A\vec{P}^l(S)$ and $\vec{Q}(S) = A\vec{P}(S)$. The revenue functions can be written as $R^l(S) = \sum_{j \in S_l} r_j P_j^l(S_l)$ and $R(S) = \sum_{j \in S} r_j P_j(S)$. Define a subset incidence matrix B with rows for all $S_l \subset C_l, l = 1, 2, \ldots, L$ and columns $S \subseteq J$, and $B_{S_l S} = 1$ if subset $S_l = S \cap C_l$ and 0 otherwise.

In our notation and demand model we broadly follow Bront et al. (2009) and Liu and van Ryzin (2008). The motivation for the design of our solution procedures comes from the following premise: The number of elements in a segment's consideration set is usually small. It sounds unlikely that a customer can process hundreds of choices in making a decision. So the problem for a single segment might be tractable by just brute-force enumeration, i.e., the number of subsets of C_l for a segment l can be enumerated explicitly as if say, $|C_l| \sim 10$, we can easily compute all the $2^{10} = 1024$ subsets of C_l.

2.2 Dynamic program

The dynamic program (DP) to determine optimal controls can be written down as follows: Let $V_t(\vec{c}_t)$ denote the maximum expected revenue to go, given remaining capacity $\vec{c}_t$ in period t. Then $V_t(\vec{c}_t)$ must satisfy the well-known Bellman equation

$$V_t(\vec{c}_t) = \max_{S \subseteq J} \left\{ \sum_{j \in S} \lambda P_j(S)(r_j + V_{t+1}(\vec{c}_t - A_j)) + (\lambda P_0(S) + 1 - \lambda) V_{t+1}(\vec{c}_t) \right\} \tag{1}$$

with the boundary condition $V_T(\vec{c}_T) = V_t(\vec{0}) = 0$ for all $\vec{c}_T$. Let $V^{DP} = V_0(\vec{c}_0)$ denote the optimal value of this dynamic program from 0 to T, for the given initial capacity vector $\vec{c}_0$.

3 Approximations and upper bounds

We give now the $CDLP$ and $SDCP$ approximations of (1).

3.1 Choice Deterministic Linear Program ($CDLP$)

The choice deterministic linear program ($CDLP$) defined in Gallego et al. (2004) and Liu and van Ryzin (2008) is as follows:

$$(CDLP) \qquad \begin{aligned} \max \quad & \sum_{S \subseteq J} \lambda R(S) w(S) \\ \text{s.t.} \quad & \sum_{S \subseteq J} \lambda w(S) \vec{Q}(S) \leq \vec{c}_0 \\ & \sum_{S \subseteq J} w(S) = T \\ & 0 \leq w(S), \forall S \subseteq J. \end{aligned} \tag{2}$$

The formulation has 2^n variables $w(S)$ that can be interpreted as the time each set is offered (including $w(\emptyset)$). Liu and van Ryzin (2008) show that the optimal objective value is an upper bound on V^{DP}. They also show that the problem can be solved efficiently by column-generation for the multinomial-logit choice model and non-overlapping segments. Bront et al. (2009) investigate this further and show that column generation is NP-hard whenever the consideration sets for the segments overlap.

3.2 Segment-based Deterministic Concave Program ($SDCP$)

Talluri (2010) proposed the following formulation that coincides with the ($CDLP$) when the segments do not overlap. For segment l, define a capacity vector $\vec{0} \leq \vec{y}_{lt} \leq \vec{1}$ that we reserve for sale to segment l in period t (even if we cannot identify this segment at the time of purchase). Given $\vec{y}_{lt}$, let $R_l^*(\vec{y}_{lt})$ represent the optimal revenue we can obtain offering some convex combination of product sets to segment l. $R_l^*(\vec{y}_{lt})$ can be obtained by solving the following linear program (we index sets in $C_l \cap J$ by S_l):

$$
\begin{aligned}
R_l^*(\vec{y}_t^{\,l}) = \quad & \max \quad \sum_{S_l} \lambda_l R^l(S_l) w^l_{S_l} \\
& \text{s.t.} \quad \sum_{S_l} \lambda_l w^l_{S_l} \vec{Q}^l(S_l) \leq \vec{y}_t^{\,l} \\
(Rgen) \qquad & \quad \sum_{S_l} w^l_{S_l} \leq 1 \\
& \quad w^l_{S_l} \geq 0, \ \forall S_l \subseteq C_l \cap J.
\end{aligned}
\tag{3}
$$

Note that $R_l^*(\vec{y}_{lt})$ is a concave function of $\vec{y}_{lt}$. The linear program ($Rgen$) has an exponential number of columns but can be solved by column generation and the column generation is often easier than that of ($CDLP$) as it is segment specific and ($Rgen$) considers only subsets of the consideration set of a single segment at a time. In fact, we can easily generate all columns if the number of considered products $|C_l|$ is small.

Now, define the following concave programming problem over the capacity vectors:

$$
\begin{aligned}
& \max \quad \sum_{t=1}^{T} \sum_{l=1}^{L} R_l^*(\vec{y}_t^{\,l}) \\
& \text{s.t.} \quad \sum_{t=1}^{T} \sum_{l=1}^{L} \vec{y}_t^{\,l} \leq \vec{c}_0 \\
(SDCP) \qquad & \quad \vec{y}_t^{\,l} \leq \lambda_l \vec{1}, \quad \forall\, l, t \\
& \quad \vec{y}_t^{\,l} \geq \vec{0}.
\end{aligned}
\tag{4}
$$

The above formulation of ($SDCP$) assumes uniform arrival rates and segment mix for simplicity, but can be modified transparently by using λ's indexed by t. The formulation is a discrete-time formulation but can be made compact by merging periods with the same arrival rates.

($SDCP$) is a compact formulation, and can be solved by any number of standard concave-programming methods generating the objective function values by solving ($Rgen$). So the critical computation lies in the calculation of $R_l^*(\vec{y}_t^{\,l})$.

Clearly ($SDCP$) overestimates revenue compared to ($CDLP$) and therefore $CDLP \leq SDCP$ but the objective values of both formulations coincide for the case of non-overlapping segments (Talluri, 2010).

4 Tightening $SDCP$

Solving ($CDLP$) when the segments overlap has been found to be exceptionally difficult. In the most general setting, the segments' consideration sets can overlap in a variety of ways, and the choice

probabilities depend on the offer set, and need not follow any structure. Indeed, Bront et al. (2009) show that generating the columns of ($CDLP$) even in a very restrictive setting (multinomial-logit model of probabilities, two segments) is NP-hard.

In this section we describe the two computational approaches that we propose in this paper.

4.1 Product cuts

The first method is based on consistency of projections onto the intersections of the considerations sets, that we call *product cuts*. This method works in a general discrete-choice setting and makes no assumptions on the overlap structure of the consideration sets or the choice probabilities. Throughout we assume that choice probabilities are given by an oracle for every segment l and offer set S.

We first describe the intuition behind our cuts. The w_S's can be interpreted as a distribution over subsets of J, and can be considered a randomization rule—at each point choose a subset based on this distribution. The distribution in turn induces a distribution for each one of the segments l, via the matrix B, $w^l_{S_l} = \sum_S B_{S_l S} w_S$.

Now the space of w_S's is prohibitively large, and the matrix B has almost no structure as the considerations sets are arbitrary. So we choose to work in the smaller space of $w^l_{S_l}$ and impose consistency conditions that arise from the fact that the segment-level distributions are generated by a common set of w_S's for each segment l. We would like these consistency conditions to be linear and to be easily generated.

Let X_j be a Bernoulli random variable which takes the value $X_j = 1$ if $j \in S$ for an offer set S sampled from the w_S distribution, and $X_j = 0$ otherwise. The expectation $E[X_j]$ is then the probability that product j is offered under this randomized rule; moreover this expected value has to be the same if we follow a similar randomized rule for segment-level distributions w^l with $j \in C_l$. So $E[X_j]$ should be the same for all segments that contain product j in their consideration sets. One can extend this to subsets of products. As X_j is a Bernoulli random variable, for any pair of segments l and k that contain two products j_1 and j_2 the following equation holds: $E[X_{j_1} X_{j_2}] = \sum_{S \ni \{j_1, j_2\}} w_S = \sum_{S_l \ni \{j_1, j_2\}} w^l_{S_l} = \sum_{S_k \ni \{j_1, j_2\}} w^k_{S_k}$. So we obtain linear valid cuts to ($SDCP$) of the form $\sum_{S_l \ni \{j_1, j_2\}} w^l_{S_l} = \sum_{S_k \ni \{j_1, j_2\}} w^k_{S_k}$ for all segments l, k such that $C_l, C_k \ni \{j_1, j_2\}$. This extends to triples of products $\{j_1, j_2, j_3\}$ via $\sum_{S_l \ni \{j_1, j_2, j_3\}} w^l_{S_l} = \sum_{S_k \ni \{j_1, j_2, j_3\}} w^k_{S_k}$, and so on.

An alternate way of viewing this idea is that the distributions w_S's and $w^l_{S_l}$'s have to be consistent once we project them onto the subsets of the intersection of the consideration sets. Since our premise is that consideration sets are relatively small, intersections of consideration sets are small also, and we can enumerate all subsets of the intersections and obtain a large number of valid cuts.

The relation between $CDLP$ and $SDCP$ is shown in (Talluri, 2010) and we repeat the connection here to show the validity of the product cuts. First formulate $CDLP$ as follows (as we are assuming uniform arrival rates, we forego subscripting the variables by t):

$$\max \quad \sum_{t=1}^{T} \sum_l \lambda_l \sum_{S_l \subset C_l} R^l(S_l) w^l_{S_l} \tag{5}$$

$$(CDLP') \quad \sum_{t=1}^{T} \sum_l \lambda_l \sum_{S_l \subset C_l} \vec{Q}^l(S_l) w^l_{S_l} \leq \vec{c}_0 \tag{6}$$

$$w^l_{S_l} \in \text{Proj}(\mathcal{W}),$$

where $\mathcal{W}$ is the space of probability distributions w over all S and $\text{Proj}(\mathcal{W})$ is the projection of $\mathcal{W}$

onto the space of $w^l_{S_l}$'s, that means it is the set of all probability distributions w^l over S_l, for all l, such that there exists a feasible solution to the following system:

$$\begin{aligned} & \sum_S B_{S_l S} w_S = w^l_{S_l} \quad \forall l \\ (\mathcal{W}([w^l])) \quad & \sum w_S = 1 \\ & w_S \geq 0. \end{aligned}$$

The $w^l_{S_l}$'s in the above formulation can be thought of as the marginal distribution on subsets of C_l for a distribution of w on $S \subset C$.

Proposition 1. *$CDLP' = CDLP$.*

Proof
Notice that $\sum_l \lambda_l \sum_{S_l \subset C_l} \vec{Q}_l(S_l) \sum_S B_{S_l S} w_S = \sum_{S \subseteq J} \lambda w(S) \vec{Q}(S)$.
□

The difficulty of solving $(CDLP)$ is in solving $(\mathcal{W}([w^l]))$ as its columns are indexed by all subsets S and the matrix B has almost no structure when the segment consideration sets overlap. Consider now $SDCP$ written as follows,

$$\begin{aligned} & \max_{\vec{y}_t} \quad \sum_{t=1}^{T} \sum_{l=1}^{L} z_{lt} \\ & \text{s.t.} \quad \sum_{t=1}^{T} \sum_{l=1}^{L} \vec{y}_{lt} \leq \vec{c}_0 \\ (SDCP') \quad & \vec{y}_{lt} \leq \lambda_l \vec{1} \quad \forall\, l, t \\ & \sum_{l \in \mathcal{L}} z_{lt} \leq R^*_{\mathcal{L}}(\vec{y}_t) \quad \forall \mathcal{L} \subset \{1, \ldots, L\} \qquad (7) \\ & \vec{y}_{lt} \geq \vec{0}, \end{aligned}$$

and

$$\begin{aligned} R^*_{\mathcal{L}}(\vec{y}_t) = \quad & \max \quad \sum_{l \in \mathcal{L}} \sum_{S_l} \lambda_l R^l(S_l) w^l_{S_l} \\ & \text{s.t.} \quad \sum_{S_l} \lambda_l \vec{Q}^l(S_l) w^l_{S_l} \leq \vec{y}_{lt} \quad \forall l \in \mathcal{L}, \\ (Rgen_{\mathcal{L}}) \quad & \sum_{S_l} w^l_{S_l} \leq 1 \quad \forall l \in \mathcal{L}, \\ & w^l_{S_l} \geq 0,\ \forall l \in \mathcal{L}, \forall S_l \subseteq C_l \cap J. \end{aligned}$$

If we consider only subsets of the form $\mathcal{L} := \{l\}$, we recover $(SDCP)$. It should be clear that equations (7) in $SDCP'$ are redundant for all non-singleton subsets of $\{1, \ldots, L\}$. However, if we define $\mathcal{L}$ to contain two overlapping segments, we can tighten the formulation by adding the following constraints to $(Rgen_{\mathcal{L}})$:

$$\sum_{S_l \supseteq S_{lk}} w^l_{S_l} = \sum_{S_k \supseteq S_{lk}} w^k_{S_k} \quad \forall S_{lk} \subseteq C_l \cap C_k, \forall \{l, k\} \subset \mathcal{L}, \qquad (8)$$

that we call *product cuts* (if we restrict $|S_{lk}| = \kappa$, we refer to them as κPC cuts). We implement this by obtaining the dual solution $(\vec{\pi}, \vec{\sigma})$ to $(Rgen_{\mathcal{L}})$ with the additional cuts (8) for the current $\vec{y}_t$, and adding the cut $\sum_{l\in\mathcal{L}} z_{lt} \leq \sum_{l\in\mathcal{L}}[\langle\vec{\pi}_l, \vec{y}_{lt}\rangle + \sigma_l]$ to $(SDCP')$ iteratively.

Proposition 2. *Suppose we add product cuts to $(Rgen_{\mathcal{L}})$ and solve $SDCP'$ as described above, then the value of the resulting linear program is greater than or equal to $CDLP$.*

Proof
Suppose $w^l_{S_l}$ and $w^k_{S_k}$'s are feasible solutions of $(CDLP')$, then there exists a set of w_S's such that $w^l_{S_l} = \sum_S B_{S_l S} w_S$ as $w^l_{S_l} \in \text{Proj}(\mathcal{W})$. Then, for any fixed $S_{lk} \subseteq C_l \cap C_k$, we have:

$$\sum_{S_l \supseteq S_{lk}} w^l_{S_l} = \sum_{S_l \supseteq S_{lk}} \sum_S B_{S_l S} w_S = \sum_{S|S\cap C_l \supseteq S_{lk}} w_S = \sum_{S|S\cap C_k \supseteq S_{lk}} w_S = \sum_{S_k \supseteq S_{lk}} w^k_{S_k}.$$

So the $w^l_{S_l}$'s should satisfy the product cuts (8) and a solution of $SDCP'$ leads to a feasible solution of $(CDLP')$.

□

Thus we obtain a tightening of $SDCP$ by adding product cuts and in our computational testing on the benchmark data sets, we obtain the $CDLP$ value rapidly just by adding cuts with small values of $|\mathcal{L}|$. Indeed we were unable to come up with an example where the values are different. Our conjecture (that we have been unable to prove) is that the product cuts obtain $CDLP$ value when there is some structure in the way the segment consideration sets intersect—for instance when the intersection graph is a tree.

4.2 κ-Segment Deterministic Concave Program ($\kappa SDCP$)

In this section we describe our second method that is a natural tightening of $(SDCP)$. Notice that we can write $(SDCP)$ as

$$\begin{aligned}
& & \max \quad & \sum_{t=1}^{T}\sum_{l=1}^{L} z_{lt} & & (9)\\
& & \text{s.t.} \quad & \sum_{t=1}^{T}\sum_{l=1}^{L} \vec{y}_{lt} \leq \vec{c}_0 & & \\
& (SDCP'') & & \vec{y}_{lt} \leq \lambda_l \vec{1} \quad \forall l & & \\
& & & z_{lt} - R^*_l(\vec{y}_{lt}) \leq 0 \quad \forall l & & (10)\\
& & & \vec{y}_{lt} \geq \vec{0} & &
\end{aligned}$$

Our idea now is to progressively tighten this formulation by adding constraints for each subset of size κ of the segments. We call this level-κ formulation $(\kappa SDCP)$. This way we obtain a way of fine-tuning the formulation—so that we always maintain an upper bound on the dynamic program, in fact on the $CDLP$—and can choose the level to suit the network and computational resources. In practice we expect this to be solved for levels 2 or 3 at most. If a large proportion of the products belong to at most two or three segments, then it is reasonable to assume that we come close to $CDLP$.

To reduce notation, we describe the formulation for segment pairs, i.e., $\kappa = 2$; the general case should be transparent from the description. For every pair of segments (k_1, k_2) where $k_1 < k_2$ and

a set of vectors of assigned capacities $\vec{y}_{k_1}, \vec{y}_{k_2}$, define

$$
\begin{aligned}
R^*_{k_1,k_2}(\vec{y}_{k_1t}, \vec{y}_{k_2t}) = \quad & \max \quad \sum_{S \subseteq C_{k_1} \cup C_{k_2}} (\lambda_{k_1} R_{k_1}(S_{k_1}) + \lambda_{k_2} R_{k_2}(S_{k_2}))w(S) \qquad (11)\\
& \text{s.t.} \quad \sum_{S \subseteq C_{k_1} \cup C_{k_2}} \Big[\lambda_{k_1} \vec{Q}_{k_1}(S_{k_1}) + \lambda_{k_2} \vec{Q}_{k_2}(S_{k_2})\Big] w(S) \leq \vec{y}_{k_1t} + \vec{y}_{k_2t}\\
(Rgen_{(k_1,k_2)}) \quad & \sum_{S \subseteq C_{k_1} \cup C_{k_2}} w(S) \leq 1\\
& w(S) \geq 0, \ \forall S \subseteq C_{k_1} \cup C_{k_2}
\end{aligned}
$$

Notice that $R^*_{k,l}(\vec{y}_{kt}, \vec{y}_{lt})$ is a concave function of the variables $\vec{y}_{kt}, \vec{y}_{lt}$. We add the following constraint to $(SDCP'')$ for all pairs (k_1, k_2) of segments:

$$z_{k_1t} + z_{k_2t} - R^*_{k_1,k_2}(\vec{y}_{k_1t}, \vec{y}_{k_2t}) \leq 0 \qquad (12)$$

and call the resulting formulation $(\kappa SDCP)$. We solve the generating concave program $(Rgen_{(k_1,k_2)})$ on the fly (and in parallel) and replace the constraints (12) by linear constraints

$$z_{k_1t} + z_{k_2t} - \langle \vec{\pi}^k_{tk_1k_2}, \vec{y}_{k_1} + \vec{y}_{k_2} \rangle \leq \sigma^k_{tk_1k_2}, \qquad (13)$$

where $(\vec{\pi}^k_{tk_1k_2}, \sigma^k_{tk_1k_2})$ is the dual solution to $R^*_{k_1,k_2}(\vec{y}_{k_1t}, \vec{y}_{k_2t})$.

To motivate $(\kappa SDCP)$, consider a simple situation where there are exactly two segments $(L = 2)$ with consideration sets C_1 and C_2. $(\kappa SDCP)$ is then $(CDLP)$, just written slightly differently. Now if the network naturally has a partition of the segments so that the consideration sets of segments in two different elements of the partition do not overlap (or have scarce overlap), then our formulation would exploit it as follows: We assign a capacity vector to each element of the partition and then, for a fixed set of capacity vectors, try to determine the optimal revenue from the assignment. For instance, if each element of the partition has exactly two segments, then we do recover the $CDLP$ as pointed out earlier.

For the general case, where each element of the partition has multiple segments, we can still solve the $2SDCP$ as an approximation. Of course, we can strengthen the formulation by defining generating concave programs for triplets of segments and so on. For a κ-tuple of segments $\{k_1, \ldots, k_\kappa\} \subseteq \{1, \ldots, L\}$ we define generating concave programs $R^*_{\{k_1,\ldots,k_\kappa\}}(\vec{y}_{k_1t}, \ldots, \vec{y}_{k_\kappa t})$ analogous to (11) and incorporate the corresponding inequalities as in (12) for the κ-tuple of variables. If we do not have an idea of the partition, we just solve it for all pairs of segments or all κ-tuples in general. No matter to what depth we solve the problem, at every stage, we are assured of an upper bound on the dynamic program. In general this upper bound is weaker than the $CDLP$ bound.

Proposition 3. $(SDCP'')$ *with the constraints (12) has an objective value greater than or equal to* $CDLP$.

Proof
Let $w(S)$ be a solution to $(CDLP)$. For every segment l, define

$$\vec{y}_{lt} = \lambda_l \sum_S \vec{Q}_l(S_l) \frac{w(S)}{T}.$$

We verify $(\kappa SDCP)$ with these vectors has an objective value same as $(CDLP)$. The vectors $\vec{y}_{lt}$ satisfy $\sum_{t=1}^T \sum_{l=1}^L \vec{y}_{lt} \leq \vec{c}_0$ as these are the same constraints as that of $(CDLP)$. Next, notice

that by construction $w(S) = \sum_{\{S' | S' \cap (C_{k_1} \cup C_{k_2}) = S\}} w(S'), S \subseteq C_{k_1} \cup C_{k_2}$ is a feasible solution to $(Rgen_{(k_1,k_2)})$, so we conclude $\kappa SDCP \geq CDLP$.

□

4.2.1 κ-Segment Randomized Concave Program (κRCP)

The $(SDCP)$ can be tightened by means of randomization, similar to the randomized linear program for the deterministic linear program as proposed by Talluri and van Ryzin (1999). This approach is called randomized concave program (RCP) and was developed by Talluri (2010). We generate K sample paths of segment demands based on the Bernoulli random variables with parameters λ_l drawn T times. We represent the realization of segment l demand in period t for the kth sample path by the indicator function $\mathbf{1}^k_{[lt]}$, which shall be equal to 1 if there is a segment l arrival and 0 otherwise. We consider a time-aggregated version of (RCP):

$$
\begin{aligned}
& \max && \sum_{l=1}^{L} R_l^*(\vec{y}_l) \\
& \text{s.t.} && \sum_{l=1}^{L} \vec{y}_l \leq \vec{c}_0 \\
(RCP^k) \qquad & && \vec{y}_l \leq \sum_{t=1}^{T} \mathbf{1}^k_{[lt]} \vec{1}, \quad \forall\, l, \\
& && \vec{y}_l \geq \vec{0}
\end{aligned}
\tag{14}
$$

We define the value of $RCP(K)$ as the average of the K concave programs:

$$RCP(K) = \frac{\sum_{k=1}^{K} RCP^k}{K}.$$

We can randomize $(\kappa SDCP)$ in the same way as we obtained (RCP) from $(SDCP)$, that is, we substitute λ_l with $\mathbf{1}^k_{[lt]}$ in equation $\vec{y}_{lt} \leq \lambda_l \vec{1}$ and in $R^*_{k_1,k_2}(\vec{y}_{k_1 t}, \vec{y}_{k_2 t})$ as defined in (11) for all segments l and time steps t of $(\kappa SDCP)$ for each sample path k. In our numerical experiments, we consider the time-aggregated version and refer to it as (κRCP). We omit further details as they should be obvious from the discussion above.

5 Numerical Results

The $(CDLP)$ is usually implemented to produce an estimate of the marginal value of capacity for each resource, and subsequently to decompose the network problem into a collection of single-resource problems. There are numerous studies that analyze the revenue performance of this decomposition process, see for example Zhang and Adelman (2009). We do not add anything new to network decomposition methods and therefore do not carry out related experiments. Instead, we focus on the tightening of upper bounds achieved by the $(SDCP)$. All experiments were programmed in Matlab R2009b with Tomlab R7 /CPLEX 11.2 on a desktop PC running MS Windows XP with a Pentium 4 CPU 2GHz and 1GB RAM.

5.1 Overview Tested Methods

We conduct a numerical study on various test networks where we compare the values resulting from the following (time-aggregated) approaches:

- $CDLP$: Choice-based deterministic linear program as defined in §3.1. As proposed by Bront et al. (2009), we use their pricing heuristic to identify new columns; if it does not find any more columns, then we use their mixed integer programming formulation until optimality is reached.
- $SDCP$: Segment-based deterministic concave program as defined in §3.2.
- κPC: $SDCP$ with Product Cuts as defined in §4.1. We define $\mathcal{L} := \{1, \ldots, L\}$. In method κPC we add product cut constraints of the form (8) only for subsets $|S_{lk}| \leq \kappa$. We use $\kappa \in \{1, 2\}$.
- $R2PC$: This is $2PC$ as defined above with randomization over λ.
- $\kappa SDCP$: The $SDCP$ with segment-pair cuts as specified in .
- RCP: Randomized Concave Program, which is a randomized version of $SDCP$ as defined in (Talluri, 2010),
- κRCP: Randomized version of κSDCP, see §4.2.1.

All randomized approaches use 500 sample paths for the first network (parallel flights) and 300 for the others.

We add product cuts for just pairs of products in the intersections of the considerations sets ($2PC$) as in all but one case (where $3PC$ gets $CDLP$ value) we obtain the $CDLP$ value and need not consider larger subsets. Likewise we test $\kappa SDCP$ with κ at most 2.

5.2 Test Networks

We use the same test networks as in Liu and van Ryzin (2008) and Bront et al. (2009), where different scenarios were obtained by scaling the capacities by a factor $\alpha \in \{0.4, 0.6, 1, 1.2, 1.4\}$. For each of these scenarios, different no-purchase weights v_0 are applied to vary demand. The probabilities are derived from the weights by using the multinomial-logit model for each of the segments exactly as in Bront et al. (2009).

Parallel Flights Example

The first network example consists of three parallel flight legs as depicted in Figure 1 with initial leg capacity 30, 50 and 40, respectively. On each flight there is a low and a high fare class L and H, respectively, with fares as specified in Table 1. We define four customer segments in Table 2; note that we do not give the preference values for the no-purchase option at this point. This is because we consider various scenarios of this network by varying both the vector of no-purchase preferences and the network capacity. The sales horizon consists of 300 time periods. In Table 3 we report upper bounds on the optimal expected revenue obtained from our various approaches. Randomization over λ does not gain much for these network instances (see Table 4 for standard deviations). The product

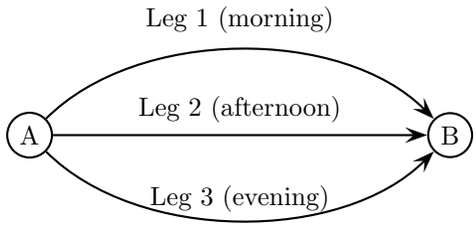

Figure 1: Parallel Flights Example.

Product	Leg	Class	Fare
1	1	L	400
2	1	H	800
3	2	L	500
4	2	H	1,000
5	3	L	300
6	3	H	600

Table 1: Product definitions for Parallel Flights Example.

Segment	Consideration set	Pref. vector	λ_l	Description
1	{2,4,6}	[5,10,1]	0.1	Price insensitive, afternoon preference
2	{1,3,5}	[5,1,10]	0.15	Price sensitive, evening preference
3	{1,2,3,4,5,6}	[10,8,6,4,3,1]	0.2	Early preference, price sensitive
4	{1,2,3,4,5,6}	[8,10,4,6,1,3]	0.05	Price insensitive, early preference

Table 2: Segment definitions for Parallel Flights Example.

cuts are very successful; 2PC obtains the $CDLP$ objective value in all instances, and even 1PC is already close to $CDLP$.

α	v_0	$CDLP$	$R2PC$	$2PC$	$1PC$	$2RCP$	$2SDCP$	RCP	$SDCP$
0.6	[1,5,5,1]	56,884	56,832	56,884	57,338	57,408	57,556	58,193	58,755
	[1,10,5,1]	56,848	56,796	56,848	57,316	57,398	57,546	58,193	58,755
	[5,20,10,5]	53,820	53,352	53,820	53,839	53,721	54,047	54,381	54,684
0.8	[1,5,5,1]	71,936	71,240	71,936	72,025	72,349	72,650	73,607	73,870
	[1,10,5,1]	71,795	70,832	71,795	71,865	72,072	72,608	73,465	73,870
	[5,20,10,5]	61,868	61,366	61,868	61,898	61,859	62,302	63,139	63,440
1.0	[1,5,5,1]	79,156	78,553	79,156	79,373	81,355	82,188	84,439	85,424
	[1,10,5,1]	76,866	76,547	76,866	77,069	79,339	79,938	82,688	83,377
	[5,20,10,5]	63,256	63,037	63,256	63,256	63,707	64,036	65,528	65,848
1.2	[1,5,5,1]	80,371	80,045	80,371	80,371	82,771	83,130	87,691	88,332
	[1,10,5,1]	78,045	77,741	78,045	78,045	80,513	80,880	85,635	86,333
	[5,20,10,5]	63,296	63,171	63,296	63,296	64,170	64,339	66,370	66,648
1.4	[1,5,5,1]	81,067	80,627	81,067	81,067	82,924	83,130	88,338	88,621
	[1,10,5,1]	78,817	78,365	78,817	78,817	80,660	80,880	86,100	86,355
	[5,20,10,5]	63,337	63,231	63,337	63,337	64,449	64,642	66,639	66,841

Table 3: Upper bounds for Parallel Flights Example.

α	v_0	$R2PC$	$2RCP$	RCP
0.6	[1,5,5,1]	1,147	1,076	839
	[1,10,5,1]	1,166	1,084	839
	[5,20,10,5]	1,424	1,262	1,143
0.8	[1,5,5,1]	2,414	1,837	1,755
	[1,10,5,1]	2,840	2,289	2,058
	[5,20,10,5]	4,351	4,413	4,416
1.0	[1,5,5,1]	5,842	5,942	5,015
	[1,10,5,1]	6,130	6,203	5,289
	[5,20,10,5]	5,572	5,479	5,352
1.2	[1,5,5,1]	6,683	7,223	7,056
	[1,10,5,1]	6,603	7,176	7,097
	[5,20,10,5]	5,670	5,664	5,687
1.4	[1,5,5,1]	6,978	7,433	7,687
	[1,10,5,1]	6,904	7,383	7,653
	[5,20,10,5]	5,654	5,681	5,933

Table 4: Standard deviations of objectives values of R2PC, 2RCP and RCP based on $K = 500$ for Parallel Flights example

Small Network Example

Next, we test the policies on a network with seven flight legs as depicted in Figure 2. In total, 22 products are defined in Table 5 and the network capacity is $\vec{c}_0 = [100, 150, 150, 150, 150, 80, 80]$, where c_{0i} is the initial seat capacity of flight leg i. In Table 6, we summarize the segment definitions

according to desired origin-destination (O-D), price sensitivity and preference for earlier flights. The booking horizon has $\tau = 1000$ time periods.

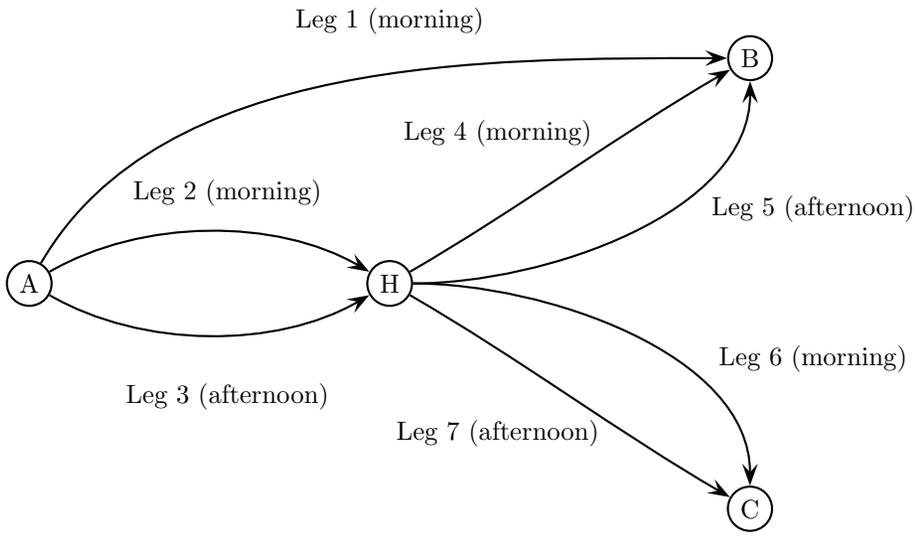

Figure 2: Small Network example.

Class = H			Class = L		
Product	Legs	Fare	Product	Legs	Fare
1	1	1,000	12	1	500
2	2	400	13	2	200
3	3	400	14	3	200
4	4	300	15	4	150
5	5	300	16	5	150
6	6	500	17	6	250
7	7	500	18	7	250
8	2,4	600	19	2,4	300
9	3,5	600	20	3,5	300
10	2,6	700	21	2,6	350
11	3,7	700	22	3,7	350

Table 5: Product definitions for Small Network Example

The upper bound results for the Small Network example in Table 7 look a bit different from the Parallel Flights case in that $2SDCP$ achieves the $CDLP$ value in all instances. This is due to the fact that each product is being considered by exactly two customer segments, and from the definition of $2SDCP$ it follows that this approach is equivalent to $CDLP$ in this situation. The product cuts perform again quite well: $2PC$ equals $CDLP$ in all except one instance. This instance $\alpha = 0.8$, $v_0 = [15 \ldots]$ is the only one where $2PC$ does not equal $CDLP$; however, $3PC$ does deliver the $CDLP$ solution 266,934. Randomization over λ lowers the respective bounds only marginally; the associated standard deviations are reported in Table 8.

Segment	O-D	Consideration set	Pref. vector	λ_l	Description
1	A→B	{1,8,9,12,19,20}	(10,8,8,6,4,4)	0.08	less price sensitive, early pref.
2	A→B	{1,8,9,12,19,20}	(1,2,2,8,10,10)	0.2	price sensitive
3	A→H	{2,3,13,14}	(10,10,5,5)	0.05	less price sensitive
4	A→H	{2,3,13,14}	(2,2,10,10)	0.2	price sensitive
5	H→B	{4,5,15,16}	(10,10,5,5)	0.1	less price sensitive
6	H→B	{4,5,15,16}	(2,2,10,8)	0.15	price sensitive, slight early pref.
7	H→C	{6,7,17,18}	(10,8,5,5)	0.02	less price sensitive, slight early pref.
8	H→C	{6,7,17,18}	(2,2,10,8)	0.05	price sensitive
9	A→C	{10,11,21,22}	(10,8,5,5)	0.02	less price sensitive, slight early pref.
10	A→C	{10,11,21,22}	(2,2,10,10)	0.04	price sensitive

Table 6: Segment definitions for Small Network Example

α	v_0	$CDLP$	$R2PC$	$2PC$	$1PC$	$2RCP$	$2SDCP$	RCP	$SDCP$
0.6	[1,5]	215,793	215,694	215,793	215,793	215,694	215,793	216,412	216,649
	[5,10]	200,515	199,521	200,515	201,294	199,521	200,515	205,312	206,392
	[10,20]	170,137	169,948	170,137	170,265	169,948	170,137	173,763	173,948
0.8	[1,5]	266,934	265,616	266,949	268,842	265,597	266,934	272,335	272,719
	[5,10]	223,173	222,934	223,173	223,536	222,934	223,173	230,000	230,393
	[10,20]	188,574	187,956	188,574	188,657	187,956	188,574	192,912	193,464
1.0	[1,5]	281,967	281,573	281,967	282,078	281,573	281,967	295,826	296,513
	[5,10]	235,284	234,866	235,284	235,446	234,866	235,284	244,635	245,226
	[10,20]	192,038	191,687	192,038	192,094	191,687	192,038	198,025	198,636
1.2	[1,5]	284,772	284,422	284,772	285,052	284,422	284,772	301,280	301,773
	[5,10]	238,562	238,113	238,562	238,562	238,113	238,562	248,304	248,728
	[10,20]	192,373	192,043	192,373	192,373	192,043	192,373	198,553	198,914
1.4	[1,5]	287,076	286,406	287,076	287,357	286,406	287,076	304,843	305,329
	[5,10]	238,562	238,160	238,562	238,562	238,160	238,562	248,909	249,372
	[10,20]	192,373	192,043	192,373	192,373	192,043	192,373	198,553	198,914

Table 7: Upper bounds for Small Network example

α	v_0	$R2PC$	$2RCP$	RCP
0.6	[1,5]	3,222	3,222	3,099
	[5,10]	4,256	4,256	3,124
	[10,20]	4,267	4,267	4,080
0.8	[1,5]	5,981	5,995	3,587
	[5,10]	6,096	6,096	6,081
	[10,20]	5,380	5,380	6,200
1.0	[1,5]	9,335	9,335	9,355
	[5,10]	6,731	6,731	6,929
	[10,20]	6,653	6,653	6,759
1.2	[1,5]	9,477	9,477	10,035
	[5,10]	8,311	8,311	8,619
	[10,20]	6,851	6,851	7,209
1.4	[1,5]	9,848	9,848	10,044
	[5,10]	8,389	8,389	8,891
	[10,20]	6,851	6,851	7,209

Table 8: Standard deviations of objectives values of R2PC, 2RCP and RCP based on $K = 300$ for Small Network example

Hub & Spoke Example

Consider the Hub & Spoke network in Figure 3. It has eight flight legs, one hub and four spokes. Each flight i has initial capacity $c_i = 200$ and the booking horizon is divided into $\tau = 2000$ time periods. There are 80 products in total which we define in Table 9 in the following way: product 1 corresponds to the trip ATL-BOS using leg 3 in class Y, product 4 is ATL-BOS in class Q, product 5 is BOS-ATL using leg 4 in class Y and so on. Definitions of the 40 customer segments for this example can be found in Table 10. We report upper bounds on the optimal expected revenue

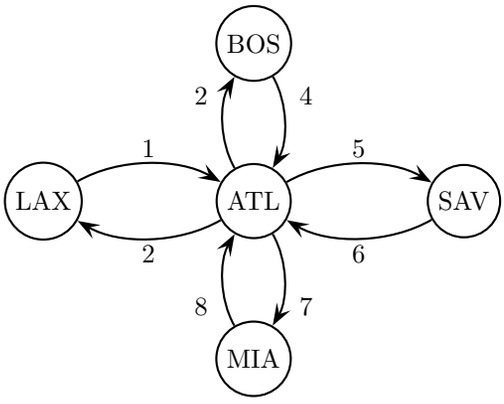

Figure 3: Hub & Spoke Network example.

in Table 11. As for the Small Network example, each product is being considered by at most two segments. Therefore, $2SDCP$ is equivalent to $CDLP$. The product cuts $2PC$ obtain $CDLP$ value in all instances. We point out that for both (κPC) and ($\kappa SDCP$) it is not possible to apply the standard proof of tightening the upper bound by randomization over the random variable λ because

O-D Market	Legs	Revenue			
		Y	M	B	Q
ATLBOS/BOSATL	3/4	310	290	95	69
ATLLAX/LAXATL	2/1	455	391	142	122
ATLMIA/MIAATL	7/8	280	209	94	59
ATLSAV/SAVATL	5/6	159	140	64	49
BOSLAX/LAXBOS	4,2/1,3	575	380	159	139
BOSMIA/MIABOS	4,7/8,3	403	314	124	89
BOSSAV/SAVBOS	4,5/6,3	319	250	109	69
LAXMIA/MIALAX	1,7/8,2	477	239	139	119
LAXSAV/SAVLAX	1,5/6,2	502	450	154	134
MIASAV/SAVMIA	8,5/6,7	226	168	84	59

Table 9: Product definitions for Hub and Spoke Example.

Segment	C_l	v_l	λ_l	Segment	C_l	v_l	λ_l
ATL/BOS H	{1,2,3,4}	{6,7,9,10}	0.015	BOS/MIA H	{41,42,43,44}	{6,7,10,10}	0.008
ATL/BOS L	{3,4}	{8,10}	0.035	BOS/MIA L	{43,44}	{8,10}	0.03
BOS/AT H	{5,6,7,8}	{6,7,9,10}	0.015	MIA/BOS H	{45,46,47,48}	{6,7,10,10}	0.008
BOS/ATL L	{7,8}	{8,10}	0.035	MIA/BOS L	{47,48}	{8,10}	0.03
ATL/LAX H	{9,10,11,12}	{5,6,9,10}	0.01	BOS/SAV H	{49,50,51,52}	{5,6,9,10}	0.01
ATL/LAX L	{11,12}	{10,10}	0.04	BOS/SAV L	{51,52}	{8,10}	0.035
LAX/ATL H	{13,14,15,16}	{5,6,9,10}	0.01	SAV/BOS H	{53,54,55,56}	{5,6,9,10}	0.01
LAX/ATL L	{15,16}	{10,10}	0.04	SAV/BOS L	{55,56}	{8,10}	0.035
ATL/MIA H	{17,18,19,20}	{5,5,10,10}	0.012	LAX/MIA H	{57,58,59,60}	{5,6,10,10}	0.012
ATL/MIA L	{19,20}	{8,10}	0.035	LAX/MIA L	{59,60}	{9,10}	0.028
MIA/ATL H	{21,22,23,24}	{5,5,10,10}	0.012	MIA/LAX H	{61,62,63,64}	{5,6,10,10}	0.012
MIA/ATL L	{23,24}	{8,10}	0.035	MIA/LAX L	{63,64}	{9,10}	0.028
ATL/SAV H	{25,26,27,28}	{4,5,8,9}	0.01	LAX/SAV H	{65,66,67,68}	{6,7,10,10}	0.016
ATL/SAV L	{27,28}	{7,10}	0.03	LAX/SAV L	{67,68}	{10,10}	0.03
SAV/ATL H	{29,30,31,32}	{4,5,8,9}	0.01	SAV/LAX H	{69,70,71,72}	{6,7,10,10}	0.016
SAV/ATL L	{31,32}	{7,10}	0.03	SAV/LAX L	{71,72}	{10,10}	0.03
BOS/LAX H	{33,34,35,36}	{5,5,7,10}	0.01	MIA/SAV H	{73,74,75,76}	{6,7,8,10}	0.01
BOS/LAX L	{35,36}	{9,10}	0.032	MIA/SAV L	{75,76}	{9,10}	0.025
LAX/BOS H	{37,38,39,40}	{5,5,7,10}	0.01	MIA/SAV H	{77,78,79,80}	{6,7,8,10}	0.01
LAX/BOS L	{39,40}	{9,10}	0.032	MIA/SAV L	{79,80}	{9,10}	0.025

Table 10: Segment definitions for Hub and Spoke Example.

λ appears in the coefficient matrix. Indeed, in Table 11 we observe instances where $R2PC$ does not tighten the $2PC$ bound, and likewise for $2RCP$ and $2SDCP$. We even tested some of those instances using 3,000 sample paths, but still no improvement was achieved, suggesting that these two methods indeed do not necessarily provide tighter bounds.

α	v_0	$CDLP$	$R2PC$	$2PC$	$1PC$	$2RCP$	$2SDCP$	RCP	$SDCP$
0.6	[1,5]	163,897	163,796	163,897	163,952	163,796	163,897	176,065	176,808
	[5,10]	132,674	132,772	132,674	132,674	132,772	132,674	143,888	144,249
	[10,20]	111,897	111,800	111,897	111,897	111,800	111,897	122,655	122,932
0.8	[1,5]	177,384	177,907	177,384	177,978	177,907	177,384	199,059	199,682
	[5,10]	146,338	146,336	146,338	146,641	146,336	146,338	163,740	164,037
	[10,20]	122,464	122,318	122,464	122,575	122,318	122,464	138,239	138,752
1.0	[1,5]	187,270	187,919	187,270	189,294	187,919	187,270	219,059	219,671
	[5,10]	156,243	155,875	156,243	157,082	155,875	156,243	180,217	180,880
	[10,20]	128,386	127,796	128,386	128,389	127,796	128,386	143,402	143,723
1.2	[1,5]	195,269	194,984	195,269	198,923	194,984	195,269	235,920	236,739
	[5,10]	160,206	159,889	160,206	160,674	159,889	160,206	188,996	189,955
	[10,20]	128,448	128,170	128,448	128,448	128,170	128,448	143,408	143,723
1.4	[1,5]	197,113	196,974	197,113	201,894	196,974	197,113	245,864	246,768
	[5,10]	160,453	160,422	160,453	160,818	160,422	160,453	189,536	189,955
	[10,20]	128,448	128,170	128,448	128,448	128,170	128,448	143,408	143,723

Table 11: Upper bounds for Hub & Spoke Example

α	v_0	$R2PC$	$2RCP$	RCP
0.6	[1,5]	4,808	4,808	4,472
	[5,10]	3,513	3,513	3,435
	[10,20]	2,702	2,702	2,752
0.8	[1,5]	5,092	5,092	4,896
	[5,10]	3,514	3,514	3,600
	[10,20]	2,880	2,880	3,204
1.0	[1,5]	4,997	4,997	5,105
	[5,10]	3,820	3,820	4,109
	[10,20]	3,433	3,433	4,293
1.2	[1,5]	5,062	5,062	5,687
	[5,10]	4,261	4,261	5,291
	[10,20]	3,648	3,648	4,303
1.4	[1,5]	5,610	5,610	6,737
	[5,10]	4,493	4,493	5,631
	[10,20]	3,648	3,648	4,303

Table 12: Standard deviations of objectives values of $R2PC$, $2RCP$ and RCP based on $K = 300$ for Hub & Spoke example

5.3 Run Times

The main motivation for the methods discussed in this article is to overcome the numerical difficulties inherent to the $(CDLP)$ formulation. While $(CDLP)$ optimizes over all possible offer sets, our

alternative formulations require much less variables. In fact, based on the premise that there are only few subsets $S_l \subset C_l$ for each segment, we can even solve the $(SDCP)$ (with or without product cuts) as a single linear program. In Table 13, we compare the run times in CPU seconds of the different approaches. We emphasize that—as always in such experiments—the measured run times will depend on the programming language used, the hardware and the skill of the programmer. However, they still may serve as an indication to the benefits obtainable from the new methods that we proposed. We implemented the methods $(2PC)$, $(1PC)$ and $(SDCP)$ as a single linear program, while $(2SDCP)$ uses the dynamic generation of cuts as described above. The run times of $(SDCP)$ with and without product cuts are significantly shorter than $(CDLP)$ using column-generation.

α	v_0	$CDLP$	$2PC$	$1PC$	$2SDCP$	$SDCP$
0.6	[1,5]	15.92	0.28	0.28	2.43	0.16
	[5,10]	11.79	0.28	0.27	2.80	0.16
	[10,20]	16.14	0.28	0.27	2.85	0.15
0.8	[1,5]	23.72	0.29	0.28	2.44	0.16
	[5,10]	18.95	0.36	0.31	2.95	0.19
	[10,20]	17.90	0.28	0.27	3.15	0.15
1.0	[1,5]	24.85	0.29	0.29	2.76	0.15
	[5,10]	7.27	0.33	0.27	3.12	0.16
	[10,20]	4.32	0.28	0.27	2.87	0.15
1.2	[1,5]	9.40	0.34	0.27	2.18	0.15
	[5,10]	4.55	0.28	0.27	2.84	0.15
	[10,20]	1.52	0.28	0.27	2.81	0.16
1.4	[1,5]	1.71	0.28	0.27	2.19	0.15
	[5,10]	1.69	0.28	0.27	2.90	0.16
	[10,20]	1.53	0.28	0.27	2.81	0.15

Table 13: Run times in CPU seconds. $CDLP$ run times are for solution by column-generation. $2PC$ and $2SDCP$ obtain the same values as $CDLP$ for all cases except one where we need to go to $3PC$.

6 Conclusions

In this paper, we have developed a computationally attractive method for solving the $(CDLP)$ approximation to the choice network revenue management problem. Using a formulation based on segments and their consideration sets, we add product cuts that are easy to generate and highly effective—they obtain the same value as $CDLP$ in all the benchmark test instances, usually in a fraction of CPU time (Table 13). Moreover, the formulation and the cuts operate at a high level of generality being applicable to a general discrete-choice model of demand, and of course for overlapping customer segments. Along with the product cuts we develop the $\kappa SDCP$ method to generalize $(SDCP)$. Finally, we perform extensive numerical simulations to test the methods. Our results indicate that it is possible to solve $(CDLP)$ effectively for industrial-sized problems when customer consideration sets are not too large, under any choice model, by adding product cuts to $(SDCP)$.

References

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

Gallego, G., G. Iyengar, R. Phillips, A. Dubey. 2004. Managing flexible products on a network. Tech. Rep. TR-2004-01, Dept of Industrial Engineering, Columbia University, NY, NY.

Kunnumkal, S., H. Topaloglu. 2010. A new dynamic programming decomposition method for the network revenue management problem with customer choice behavior. *Production and Operations Management* **19** 575–590.

Liu, Q., G. van Ryzin. 2008. On the choice-based linear programming model for network revenue management. *Manufacturing and Service Operations Management* **10**(2) 288–310.

Meissner, J., A. K. Strauss. 2008. Network revenue management with inventory-sensitive bid prices and customer choice. Tech. rep., Lancaster University, Department of Management Science.

Talluri, K. T., G. J. van Ryzin. 1999. A randomized linear programming method for computing network bid prices. *Transportation Science* **33** 207–216.

Talluri, K. T., G. J. van Ryzin. 2004a. Revenue management under a general discrete choice model of consumer behavior. *Management Science* .

Talluri, K. T., G. J. van Ryzin. 2004b. *The Theory and Practice of Revenue Management*. Kluwer, New York, NY.

Talluri, K.T. 2010. A randomized concave programming method for choice network revenue management. Tech. Rep. 1215, Department of Economics, Universitat Pompeu Fabra, Barcelona, Spain.

Zhang, D., D. Adelman. 2009. An approximate dynamic programming approach to network revenue management with customer choice. *Transportation Science* **43**(3) 381–394.

Zhang, D., W. L. Cooper. 2005. Revenue management for parallel flights with customer choice. *Operations Research* **53** 415–431.