Math. Prog. Comp. (2013) 5:75–112
DOI 10.1007/s12532-012-0049-9

FULL LENGTH PAPER

Branch-and-cut for separable piecewise linear optimization and intersection with semi-continuous constraints

I. R. de Farias Jr. · E. Kozyreff · R. Gupta · M. Zhao

Received: 13 November 2011 / Accepted: 21 October 2012 / Published online: 8 November 2012
© Springer-Verlag Berlin Heidelberg and Mathematical Optimization Society 2012

Abstract We report and analyze the results of our computational testing of branch-and-cut for piecewise linear optimization using the cutting planes given recently by Zhao and de Farias. Besides evaluating the performance of the cuts, we evaluate the effect of formulation on the performance of branch-and-cut. Finally, we report and analyze results on piecewise linear optimization problems with semi-continuous constraints.

Keywords Piecewise linear optimization · Mixed-integer programming · Knapsack problem · Special ordered set · Semi-continuous variable · Polyhedral method · Branch-and-cut

Mathematics Subject Classification 90

1 Introduction

Piecewise linear optimization (PLO) stands as one of the most frequently used optimization models in practice. It arises in such diverse applications as transportation

I. R. de Farias Jr. (✉) · E. Kozyreff · R. Gupta
Department of Industrial Engineering, Texas Tech University, Lubbock, USA
e-mail: ismael.de-farias@ttu.edu

E. Kozyreff
e-mail: ernee.kozyreff@ttu.edu

R. Gupta
e-mail: rajat.gupta@ttu.edu

M. Zhao
SAS, Cary, USA
e-mail: ming.zhao@sas.com

[1], finance [3,20,24], and supply-chain management [5,6]. It is also often used to approximate difficult nonlinear programming (NLP) models (see for example [22,25–28,31]), and to optimize functions that are not defined analytically, e.g. functions whose values are known only for a number of sample points [23]. In applications, more often than not, PLO is continuous and *separable* (for noncontinuous PLO, see de Farias et al. [14]).

Let n be a positive integer and $N = \{1, \ldots, n\}$. The continuous and separable PLO is:

$$\begin{aligned} minimize \quad & \sum_{j \in N} g_j(x_j) \\ s.t. \quad & Ax \leq b \\ & x_j \in [0, \omega_j], \; j \in N, \end{aligned} \tag{1}$$

where g_j is a continuous piecewise linear function $\forall j \in N$. This is the model studied in this work. Thus, henceforth, unless noted explicitly, PLO will mean continuous and separable PLO. When all functions g_j are convex, PLO can be solved in polynomial time [15], otherwise it is NP-hard [19].

Typically, in the literature as well as in practice, PLO is formulated using *special ordered sets of type 2 (SOS2)*.

Definition 1 (Beale and Tomlin [2]) A set of variables $\{\lambda_0, \ldots, \lambda_T\}$ is SOS2 when:

$$\text{at most two variables can be nonzero, and two nonzero variables must be adjacent.} \tag{2}$$

□

Specifically, consider the variable x_j, $j \in N$, and let $d_j^0 = 0, d_j^1, \ldots, d_j^T = \omega_j$ be the *breakpoints* of function g_j (i.e. the points where the slope of g_j changes plus the endpoints 0 and ω_j), where $d_j^0 < \cdots < d_j^T$. For notational simplicity, we assume without loss of generality (WLOG) that all functions g_j have the same number $T + 1$ of breakpoints (in this case we may have to add breakpoints at places where the slope does not change). We write:

$$x_j = \sum_{k=0}^{T} d_j^k \lambda_j^k, \tag{3}$$

$$\sum_{k=0}^{T} \lambda_j^k = 1, \tag{4}$$

and

$$\lambda_j^k \geq 0 \quad \forall k \in \{0, \ldots, T\}. \tag{5}$$

Now, let

$$\sum_{j \in N} \alpha_j x_j \leq b \tag{6}$$

be one of the inequalities in (1). Plugging (3) into (6), we obtain:

$$\sum_{j \in N^+} \sum_{k=0}^{T} a_j^k \lambda_j^k - \sum_{j \in N^-} \sum_{k=0}^{T} a_j^k \lambda_j^k \leq b, \tag{7}$$

where $a_j^k = |\alpha_j| d_j^k \ \forall j \in N, k \in \{0, \ldots, T\}$, $N^+ = \{j \in N : \alpha_j \geq 0\}$, and $N^- = \{j \in N : \alpha_j < 0\}$. Note that $N^+ \cup N^- = N$, and since $d_j^0 < \cdots < d_j^T, 0 < a_j^1 < \cdots < a_j^T \ \forall j \in N$ when $\alpha_j \neq 0$. By repeating this with all variables x_j and all constraints in (1), we obtain a model on variables λ_j^k. And by adding the constraint

$$\{\lambda_j^0, \ldots, \lambda_j^T\} \text{ is SOS2} \quad \forall j \in N, \tag{8}$$

we obtain the *SOS2 model*.

Constraint (8) can be dealt with by introducing binary variables as in the "usual" mixed-integer programming (MIP) approach [7] or the more recent "logarithmic" (LOG) approach [29,30]; or alternatively through the *SOS2 approach* [2]. We note that PLO can also be formulated as in the *incremental cost model* of Markowitz and Manne [21]. Theoretically, the incremental cost and the "usual" MIP models are equivalent with respect to their initial linear programming bounds [18]. Whether, in practice, they perform with similar efficiency, remains to be tested.

Due to its importance, all main commercial MIP software contain special facilities for SOS2. Yet, they lack the capability of solving within acceptable computational time PLO instances of medium to large sizes. One possible reason is their lacking of separation heuristics for cutting planes that are valid for general PLO, but that are specific for SOS2 models. Recently, Zhao and de Farias [32] gave several new families of cutting planes for PLO, some of which define facets under mild conditions (see also Keha et al. [19] for a previous study. Some of the inequalities in [19] are now special cases of inequalities in [32]). Here we present a computational study of branch-and-cut for PLO in which we evaluate the impact on the performance of branch-and-cut of the:

1. inequalities of Zhao and de Farias [32]
2. choice of the "usual" MIP approach, LOG formulation, or SOS2 approach
3. default branch-and-cut facilities (i.e. preprocessing, primal heuristics, and MIP cutting planes) when an MIP approach is adopted.

Zhao and de Farias [32] also strengthened their inequalities for the case in which PLO contains *semi-continuous variables*. A variable x is semi-continuous [8,13] when $x \in [0, p] \cup [l, \omega]$, where $0 \leq p < l \leq \omega$. Semi-continuous variables abound in applications, including NLP applications that are approximated as PLO, see for example [26,27,31]. Additionally, semi-continuous variables are present in models

where PLO arises "naturally" (i.e. not as an approximation), see for example [24]. Here, in addition to our analysis of branch-and-cut for PLO, we analyze PLO with semi-continuous variables.

The remainder of the paper is organized as follows. In Sect. 2 we present assumptions, notation, and the inequalities of Zhao and de Farias [32] that are analyzed in this work. In Sect. 3 we give their separation heuristics. In Sect. 4 we describe the platform used for computation, the instances tested, and the specific tests we conducted. In Sect. 5 we give results on instances of transshipment with concave piecewise linear cost functions. In Sect. 6 we give results on instances of transportation with concave piecewise linear cost functions. In Sect. 7 we present results for PLO with functions that contain many breakpoints, i.e. with large SOS2's. In Sect. 8 we give results on PLO with semi-continuous variables. Finally, in Sect. 9 we give directions for continued research.

2 Valid inequalities

Our cutting plane strategy follows that pioneered by Crowder et al. [4] for 0–1 programming, whose success carries over to PLO, as it was first demonstrated by Keha et al. [19]. Specifically, we use as cuts inequalities valid for a *knapsack PLO relaxation* of PLO, defined by (4), (5), (7), and (8), or equivalently by a single inequality (6).

Because we focus on a single inequality (6), we assume $\alpha_j \neq 0 \, \forall j \in N$. Note that in this case, N^+ becomes $\{j \in N : \alpha_j > 0\}$. Also, because it is easier to study the inequalities valid for a polyhedron if it is full-dimensional, we proceed as in Keha et al. [19]: denoting $K = \{1, \ldots, T\}$, we project out variables λ_j^0 from the formulation $\forall j \in N$, and we replace constraints (4) and the SOS2 constraints with

$$\sum_{j \in K} \lambda_j^k \leq 1 \tag{9}$$

and SOS2′, respectively, where SOS2′ is defined as:

Definition 2 (Keha et al. [19]) The set $\{\lambda_j^1, \ldots, \lambda_j^T\}$ is SOS2′ if it is SOS2, and $\lambda_j^k > 0$ for $k > 1 \Rightarrow \sum_{k \in K} \lambda_j^k = 1$. □

Now, we rewrite (7) as

$$\sum_{j \in N^+} \sum_{k \in K} a_j^k \lambda_j^k - \sum_{j \in N^-} \sum_{k \in K} a_j^k \lambda_j^k \leq b. \tag{10}$$

Our polyhedron of interest is $P = conv(S)$, where

$$S = \{\lambda \in \Re^{nT} : \lambda \text{ satisfies (5), (9), (10), and } SOS2' \, \forall j \in N\}.$$

Next, we give the inequalities for P derived in Zhao and de Farias [32] that we tested computationally. They are two families of *lifted convexity inequalities* and three

families of *lifted cover inequalities*. Additionally, we tested the extensions of the lifted convexity inequalities and two of the lifted cover inequalities for when there are semi-continuous constraints that were derived in Zhao and de Farias [32]. For the remainder of the paper, we adopt the convention that when $k > l$, $\sum_{j=k}^{l} \chi_j = 0$, where the χ_j's are real numbers.

2.1 Valid inequalities for PLO

2.1.1 Lifted convexity inequalities

Below we give the first family of lifted convexity inequalities.

Theorem 1 (Zhao and de Farias [32]) *Let* $j \in N^+, N_1^- \subseteq N^-$, *and* $b' = b + \sum_{i \in N_1^-} a_i^{m_i}$, *where* $m_i \in K \ \forall i \in N_1^-$. *Let* $s \in K$ *be such that* $a_j^s < b', I = \{i \in N^+ - \{j\} : a_j^s + a_i^T > b'\}$, *and* $k_i = \min\{k \in K : a_j^s + a_i^k > b'\} \ \forall i \in I$. *Suppose that* $I \neq \emptyset$. *Then,*

$$\frac{1}{a_j^s}\sum_{k=1}^{s-1} a_j^k \lambda_j^k + \sum_{k=s}^{T} \lambda_j^k + \sum_{i \in I} \sum_{k=\max\{1,k_i-1\}}^{T} \alpha_i^k \lambda_i^k - \sum_{i \in N_1^-} \sum_{k=m_i+1}^{T} \beta_i^k \lambda_i^k - \sum_{i \in N^- - N_1^-} \sum_{k \in K} \frac{a_i^k}{a_j^s} \lambda_i^k \leq 1 \tag{11}$$

is valid for P, where

$$\left(\alpha_i^{k_i-1}, \alpha_i^{k_i}\right) \in \left\{(0,0), \left(\frac{a_j^s + a_i^{k_i-1} - b'}{a_j^s}, \frac{a_j^s + a_i^{k_i} - b'}{a_j^s}\right)\right\} \quad \forall i \in I$$
$$\text{with } k_i > 1 \text{ and } a_j^s + a_i^{k_i-1} < b', \tag{12}$$
$$\left(\alpha_i^{k_i-1}, \alpha_i^{k_i}\right) = \left(0, \frac{a_j^s + a_i^{k_i} - b'}{a_j^s}\right) \quad \forall i \in I \text{with } k_i > 1 \text{ and } a_j^s + a_i^{k_i-1} = b',$$
$$\alpha_i^{k_i} = 0 \quad \forall i \in I \text{ with } k_i = 1,$$
$$\alpha_i^k = \frac{a_j^s + a_i^k - b'}{a_j^s} \quad \forall i \in I \text{ with } k > k_i,$$

and

$$\beta_i^k = \frac{a_i^k - a_i^{m_i}}{a_j^s}.$$

□

Example 1 Let $|N^+| = |N^-| = 2, T = 3, d_1^1 = 2, d_1^2 = 6, d_1^3 = 8, d_2^1 = 5, d_2^2 = 9, d_2^3 = 20, d_3^1 = 4, d_3^2 = 6, d_3^3 = 8, d_4^1 = 2, d_4^2 = 5, d_4^3 = 8, \alpha_1 = \alpha_2 = 1, \alpha_3 = \alpha_4 = -1$, and (10) be

$$2\lambda_1^1 + 6\lambda_1^2 + 8\lambda_1^3 + 5\lambda_2^1 + 9\lambda_2^2 + 20\lambda_2^3 - 4\lambda_3^1 - 6\lambda_3^2 - 8\lambda_3^3 - 2\lambda_4^1 - 5\lambda_4^2 - 8\lambda_4^3 \leq 10.$$

Let $j = 1, s = 1$ and $N_1^- = \emptyset$. Then $b' = 10, I = \{2\}, k_2 = 2$ and (11) is

$$2\lambda_1^1 + 2\lambda_1^2 + 2\lambda_1^3 - 3\lambda_2^1 + \lambda_2^2 + 12\lambda_2^3 - 4\lambda_3^1 - 6\lambda_3^2 - 8\lambda_3^3 - 2\lambda_4^1 - 5\lambda_4^2 - 8\lambda_4^3 \leq 2.$$

This inequality defines a facet of P. □

We now give the second family of lifted convexity inequalities.

Theorem 2 (Zhao and de Farias [32]) *Let $j \in N^+, N_1^- \subseteq N^-$, and $b' = b + \sum_{i \in N_1^-} a_i^{m_i}$, where $m_i \in K \ \forall i \in N_1^-$. Let $s \in K - \{1\}$ and $I = \{i \in N^+ - \{j\} : a_j^{s-1} + a_i^T \geq b'\}$. Suppose that $I \neq \emptyset$. Let $k_i = \min\{k \in K : a_j^{s-1} + a_i^k \geq b'\}, i \in I$, and $L = \{i \in I : a_j^s + a_i^{k_i - 1} \geq b'$ and $k_i > 1\}$. Suppose that $L \neq \emptyset$, and let $a_L = \min\{a_i^{k_i - 1} : i \in L\}$. Then,*

$$\sum_{k=1}^{s-1} \gamma_j^k \lambda_j^k + \sum_{k=s}^{T} \lambda_j^k + \sum_{i \in L} \sum_{k=k_i}^{T} \alpha_i^k \lambda_i^k - \sum_{i \in N_1^-} \sum_{k=m_i+1}^{T} \beta_i^k \lambda_i^k - \sum_{i \in N^- - N_1^-} \sum_{k \in K} \gamma_i^k \lambda_i^k \leq 1 \tag{13}$$

is valid for P, where

$$\alpha_i^k = \frac{a_i^k - a_L}{b' - a_L} \quad \forall i \in L \text{ with } k \geq k_i,$$
$$\beta_i^k = \frac{a_i^k - a_i^{m_i}}{b' - a_L} \quad \forall i \in N_1^- \text{ with } k \geq m_i + 1,$$

and

$$\gamma_i^k = \frac{a_i^k}{b' - a_L} \quad \forall i \in N^- \cup \{j\} - N_1^- \text{ with } k \in K.$$

□

Example 2 Let $|N| = |N^+| = 3, T = 3, d_1^1 = 2, d_1^2 = 6, d_1^3 = 8, d_2^1 = 3, d_2^2 = 7, d_2^3 = 10, d_3^1 = 5, d_3^2 = 7, d_3^3 = 9, \alpha_1 = \alpha_2 = \alpha_3 = 1$, and (10) be

$$2\lambda_1^1 + 6\lambda_1^2 + 8\lambda_1^3 + 3\lambda_2^1 + 7\lambda_2^2 + 10\lambda_2^3 + 5\lambda_3^1 + 7\lambda_3^2 + 9\lambda_3^3 \leq 10. \tag{14}$$

Let $j = 1$ and $s = 2$. Then $I = \{2, 3\}$, $k_2 = k_3 = 3$, $L = \{2, 3\}$, $b' = 10$, and $a_L = 7$. The following inequality defines a facet of P:

$$\frac{2}{3}\lambda_1^1 + \lambda_1^2 + \lambda_1^3 + \lambda_2^3 + \frac{2}{3}\lambda_3^3 \leq 1.$$

□

2.1.2 Lifted cover inequalities

We now give the three families of lifted cover inequalities derived in Zhao and de Farias [32] that we tested computationally. Given $j \in N$, we denote

$$u_j^k = a_j^k - a_j^{k-1} \; \forall k \in K - \{1\} \quad \text{and} \quad u_j^1 = a_j^1.$$

We now assume that $N = N^+$. Later in this section this assumption will be lifted.

Definition 3 (Keha et al. [19]) Let $C \subseteq N$ and $l_j \in \{2, \ldots, T\} \; \forall j \in C$ be such that

$$\sum_{j \in C} a_j^{l_j} = b + \rho \tag{15}$$

with $\rho > 0$. The set C is a cover. □

We note that the cover is not defined just by the elements of C but also by the l_j's. So, in principle, the notation for a cover should specify jointly C and the l_j's. In what follows, however, in no occasion will two different covers be considered at the same time. Because there is no risk of confusion, to simplify notation, we will denote the cover by C, with the l_j's understood by the particular context. For the remainder of the paper C will denote a cover.

We now give the first family of lifted cover inequalities.

Theorem 3 (Zhao and de Farias [32]) *Let C_1 and C_2 be two disjoint subsets of C such that $C = C_1 \cup C_2$, and $l_j > 2 \; \forall j \in C_1$. Then,*

$$\sum_{j \in C_1} \left(\alpha_j \lambda_j^{l_j - 2} + \beta_j \lambda_j^{l_j - 1} + \sum_{k=l_j}^{T} \lambda_j^k \right) + \sum_{j \in C_2} \left(\gamma_j \lambda_j^{l_j - 1} + \sum_{k=l_j}^{T} \lambda_j^k \right) \leq |C| - 1 \tag{16}$$

is valid for P, where

$$\alpha_j = min \left\{ 0, \frac{\rho - u_j^{l_j} - u_j^{l_j - 1}}{\rho} \right\}, \quad \beta_j = \frac{\rho - u_j^{l_j}}{\rho}, \quad \text{and} \quad \gamma_j = min\{0, \beta_j\}.$$

□

Example 3 Let $N = \{1, 2, 3\}, T = 3, d_1^1 = 1, d_1^2 = 3, d_1^3 = 4, d_2^1 = 1, d_2^2 = 3, d_2^3 = 4, d_3^1 = 2, d_3^2 = 4, d_3^3 = 9, \alpha_1 = \alpha_2 = \alpha_3 = 1$, and (10) be

$$\lambda_1^1 + 3\lambda_1^2 + 4\lambda_1^3 + \lambda_2^1 + 3\lambda_2^2 + 4\lambda_2^3 + 2\lambda_3^1 + 4\lambda_3^2 + 9\lambda_3^3 \leq 10.$$

Let $C = \{1, 2, 3\}, l_1 = l_2 = 3, l_3 = 2, C_1 = \{1, 2\}$, and $C_2 = \{3\}$. We have that

$$-\frac{1}{2}\lambda_1^1 + \frac{1}{2}\lambda_1^2 + \lambda_1^3 - \frac{1}{2}\lambda_2^1 + \frac{1}{2}\lambda_2^2 + \lambda_2^3 + \lambda_3^2 + \lambda_3^3 \leq 2 \tag{17}$$

is valid for P. It cuts off the point $\lambda_1^3 = \frac{1}{2}, \lambda_2^3 = \lambda_3^2 = 1$, and $\lambda_j^k = 0$ otherwise, which is a vertex of the feasible set of the linear programming (LP) relaxation (obtained by dropping the SOS2′ constraints). □

Next, we give the second family of lifted cover inequalities.

Theorem 4 (Zhao and de Farias [32]) *Let* $a_C = max\{a_i^{l_i} : \forall i \in C\}, \bar{C} = \{j \in N - C : a_j^T \geq a_C$ *and* $a_j^{T-1} \geq a_C - \rho\}$, *and* $t_j = min\{k : a_j^k \geq a_C$ *and* $a_j^{k-1} \geq a_C - \rho\} \forall j \in \bar{C}$. *Suppose that* $\bar{C} \neq \emptyset$. *The lifted cover inequality*

$$\sum_{j \in C} \left(\gamma_j \lambda_j^{l_j - 1} + \sum_{k=l_j}^{T} \lambda_j^k \right) + \sum_{j \in \bar{C}} \sum_{k=t_j}^{T} \lambda_j^k \leq |C| - 1 \tag{18}$$

is valid for P, *where the* γ_j*'s are as in Theorem 3.* □

Example 4 Let $|N^+| = 4, N^- = \emptyset, T = 3, d_1^1 = 2, d_1^2 = 6, d_1^3 = 8, d_2^1 = 3, d_2^2 = 7, d_2^3 = 10, d_3^1 = 4, d_3^2 = 8, d_3^3 = 10, d_4^1 = 5, d_4^2 = 7, d_4^3 = 9, \alpha_1 = \alpha_2 = \alpha_3 = \alpha_4 = 1$, and (10) be

$$2\lambda_1^1 + 6\lambda_1^2 + 8\lambda_1^3 + 3\lambda_2^1 + 7\lambda_2^2 + 10\lambda_2^3 + 4\lambda_3^1 + 8\lambda_3^2 + 10\lambda_3^3 + 5\lambda_4^1 + 7\lambda_4^2 + 9\lambda_4^3 \leq 10. \tag{19}$$

We take $C = \{1, 2\}$ and $l_1 = l_2 = 2$. Then, $\bar{C} \in \{3, 4\}$ and $t_3 = t_4 = 2$. The inequality

$$-\frac{1}{3}\lambda_1^1 + \lambda_1^2 + \lambda_1^3 - \frac{1}{3}\lambda_2^1 + \lambda_2^2 + \lambda_2^3 + \lambda_3^2 + \lambda_3^3 + \lambda_4^2 + \lambda_4^3 \leq 1 \tag{20}$$

defines a facet of P. □

Now we drop the assumption that $N^- = \emptyset$.

Definition 4 (Keha et al. [19]) Let $C^+ \subseteq N^+, C^- \subseteq N^-, 2 \leq l_j \leq T \; \forall j \in C^+, 1 \leq l_j \leq T - 1 \; \forall j \in C^-$, and $C = C^+ \cup C^-$. If

$$\sum_{j \in C^+} a_j^{l_j} - \sum_{j \in C^-} a_j^{l_j} = b + \rho$$

with $\rho > 0$, C is a generalized cover. □

We give below the third family of lifted cover inequalities.

Theorem 5 (Zhao and de Farias [32]) *Let C be a generalized cover. The inequality*

$$\sum_{j\in C_1}\left(\alpha_j\lambda_j^{l_j-2}+\beta_j\lambda_j^{l_j-1}+\sum_{k=l_j}^{T}\lambda_j^k\right)+\sum_{j\in C_2}\left(\gamma_j\lambda_j^{l_j-1}+\sum_{k=l_j}^{T}\lambda_j^k\right)$$
$$-\sum_{j\in N^-}\left(\tau_j\lambda_j^{l_j+1}+\sum_{k=l_j+2}^{T}\lambda_j^k\right)\le |C^+|-1 \tag{21}$$

is valid for P, where C_1 and C_2 are disjoint, $C^+=C_1\cup C_2, l_j>2\ \forall j\in C_1, l_j=0\ \forall j\in N^- - C^-, \alpha_j, \beta_j, \gamma_j$ are given as in Theorem 3, and

$$\tau_j=\max\left\{1,\frac{u_j^{l_j+1}}{\rho}\right\}.$$

□

Example 1 (Continued) We let $C=\{1,2,3\}$ with $C^+=\{1,2\}, C^-=\{3\}, l_1=3, l_2=2$, and $l_3=2$. We also let $C_1=\{1\}$ and $C_2=\{2\}$, and we have that $l_4=0$. It then follows that

$$-5\lambda_1^1-\lambda_1^2+\lambda_1^3-3\lambda_2^1+\lambda_2^2+\lambda_2^3-2\lambda_3^3-3\lambda_4^1-\lambda_4^2-\lambda_4^3\le 1$$

is valid for P. It cuts off the point $\lambda_1^3=1, \lambda_2^2=\frac{2}{3}, \lambda_3^1=1$, and $\lambda_j^k=0$ otherwise, which is a vertex of the feasible set of the LP relaxation. □

2.2 Valid inequalities for PLO with semi-continuous variables

Let x be a semi-continuous variable, i.e. $x\in[0,p]\cup[l,\omega]$, where $0\le p<l\le\omega$. For simplicity of notation, and WLOG, we assume that the forbidden interval (p,l) is determined by two adjacent breakpoints of x (to keep the number of breakpoints for each variable x_j the same, as assumed earlier, one may add additional breakpoints in a non-forbidden interval). Specifically, let $i\in N$ and the breakpoints of x_i be $d_i^0(=0),\ldots,d_i^T(=\omega_i)$. Then, $p=d_i^{k-1}$ and $l=d_i^k$ for some $k\in K$. Also for simplicity of notation, and WLOG, we assume that all variables $x_j, j\in N$, are semi-continuous (in this case we may have to allow $p=l$ for some variables, or equivalently that the forbidden interval (p,l) is empty for these variables). We denote the index $k\in K$ that defines the semi-continuous constraint of x_i as k_i^*, i.e. $x_i\in[0,d_i^{k_i^*-1}]\cup[d_i^{k_i^*},\omega_i]$; and we denote the PLO polytope with semi-continuous constraints P_{SC}.

Zhao and de Farias [32] strengthened inequalities (11), (13), (16), and (18) to account for when PLO has semi-continuous constraints. We now give these inequalities, which we tested computationally. We call the inequalities collectively SC-PLO cuts.

Below we give the first SC-PLO cut.

Theorem 6 (Zhao and de Farias [32]) *Let $j \in N^+, N_1^- \subseteq N^-$, and $b' = b + \sum_{i \in N_1^-} a_i^{m_i}$, where $m_i \in K\ \forall i \in N_1^-$. Let $s \in K$ be such that $a_j^s < b'$, $I = \{i \in N^+ - \{j\} : a_j^s + a_i^T > b'\}$, and $k_i = \min\{k \in K : a_j^s + a_i^k > b'\}\ \forall i \in I$. Let $I^* = \{i \in I : k_i^* = k_i\}$. Suppose that $I^* \neq \emptyset$. Then,*

$$\frac{1}{a_j^s}\sum_{k=1}^{s-1} a_j^k \lambda_j^k + \sum_{k=s}^{T} \lambda_j^k + \sum_{i \in I^*}\sum_{k=k_i}^{T} \alpha_i^{*k}\lambda_i^k + \sum_{i \in I - I^*}\sum_{k=\max\{1,k_i-1\}}^{T} \alpha_i^k \lambda_i^k$$
$$- \sum_{i \in N_1^-}\sum_{k=m_i+1}^{T} \beta_i^k \lambda_i^k - \sum_{i \in N^- - N_1^-}\sum_{k \in K} \frac{a_i^k}{a_j^s}\lambda_i^k \leq 1 \tag{22}$$

is valid for P_{SC}, where

$$\alpha_i^{*k} = \frac{a_j^s + a_i^k - b'}{a_j^s} \quad \forall i \in I^*, k \in K,$$

$$\left(\alpha_i^{k_i-1}, \alpha_i^{k_i}\right) \in \left\{(0,0), \left(\frac{a_j^s + a_i^{k_i-1} - b'}{a_j^s}, \frac{a_j^s + a_i^{k_i} - b'}{a_j^s}\right)\right\} \quad \forall i \in I - I^*$$
$$\text{with } k_i > 1 \text{ and } a_j^s + a_i^{k_i-1} < b',$$

$$\left(\alpha_i^{k_i-1}, \alpha_i^{k_i}\right) = \left(0, \frac{a_j^s + a_i^{k_i} - b'}{a_j^s}\right) \quad \forall i \in I - I^* \text{ with } k_i > 1 \text{ and } a_j^s + a_i^{k_i-1} = b',$$

$$\alpha_i^{k_i} = 0 \quad \forall i \in I - I^* \text{ with } k_i = 1,$$

$$\alpha_i^k = \frac{a_j^s + a_i^k - b'}{a_j^s} \quad \forall i \in I - I^* \text{ with } k > k_i,$$

and

$$\beta_i^k = \frac{a_i^k - a_i^{m_i}}{a_j^s}.$$

□

Example 4 (Continued) Suppose that $x_1 \in \{0\} \cup [2,8]$, $x_2 \in [0,3] \cup [7,10]$, $x_3 \in [0,8] \cup \{10\}$, and $x_4 \in \{0\} \cup [5,9]$. We then have that $k_1^* = 1, k_2^* = 2, k_3^* = 3$, and $k_4^* = 1$. Then,

$$\frac{1}{3}\lambda_1^1 + \lambda_1^2 + \lambda_1^3 + \frac{1}{2}\lambda_2^2 + \lambda_2^3 + \frac{2}{3}\lambda_3^2 + \lambda_3^3 + \frac{1}{6}\lambda_4^1 + \frac{1}{2}\lambda_4^2 + \frac{5}{6}\lambda_4^3 \leq 1 \tag{23}$$

is valid for P_{SC}. Inequality (23) cuts off $\lambda_1^2 = 1$, $\lambda_4^1 = \frac{4}{5}$, and $\lambda_j^k = 0$ otherwise, which is a vertex of the feasible set of the LP relaxation that satisfies the SOS2′ constraints, but not semi-continuous, corresponding to $x_1 = 6$, $x_2 = x_3 = 0$, and $x_4 = 4$. □

We now give the second SC-PLO cut.

Theorem 7 (Zhao and de Farias [32]) *Let* $j \in N^+, N_1^- \subseteq N^-$, *and* $b' = b + \sum_{i \in N_1^-} a_i^{m_i}$, *where* $m_i \in K\ \forall i \in N_1^-$. *Let* $s \in K - \{1\}$ *and* $I = \{i \in N^+ - \{j\} : a_j^{s-1} + a_i^T \geq b'\}$. *Suppose that* $I \neq \emptyset$. *Let* $k_i = \min\{k \in K : a_j^{s-1} + a_i^k \geq b'\}, i \in I$. *Denote* $I^* = \{i \in I : k_i = k_i^*\}$ *and* $L = \{i \in I - I^* : a_j^s + a_i^{k_i - 1} \geq b'$ *and* $k_i > 1\}$. *Suppose that* $I^* \neq \emptyset$. *Then,*

$$\sum_{k=1}^{s-1} \gamma_j^k \lambda_j^k + \sum_{k=s}^{T} \lambda_j^k + \sum_{i \in I^* \cup L} \sum_{k=k_i}^{T} \alpha_i^k \lambda_i^k - \sum_{i \in N_1^-} \sum_{k=m_i+1}^{T} \beta_i^k \lambda_i^k - \sum_{i \in N^- - N_1^-} \sum_{k \in K} \gamma_i^k \lambda_i^k \leq 1 \tag{24}$$

is valid for P_{SC}, *where*

$$\alpha_i^k = \frac{a_i^k - a_L}{b' - a_j^s} \quad \forall i \in I^* \cup L \text{ with } k \geq k_i,$$

$$\beta_i^k = \frac{a_i^k - a_i^{m_i}}{b' - a_j^s} \quad \forall i \in N_1^- \text{ with } k \geq m_i + 1,$$

and

$$\gamma_i^k = \frac{a_i^k}{b' - a_j^s} \quad \forall i \in N^- \cup \{j\} - N_1^- \text{ with } k \in K.$$

□

Example 2 (Continued) Suppose that $x_1 \in \{0\} \cup [2, 8]$, $x_2 \in \{0\} \cup [3, 10]$, and $x_3 \in \{0\} \cup [5, 9]$. We then have that $k_1^* = k_2^* = k_3^* = 1$. Then,

$$\frac{1}{4}\lambda_1^1 + \frac{3}{4}\lambda_1^2 + \lambda_1^3 + \frac{3}{8}\lambda_2^1 + \frac{5}{8}\lambda_2^2 + \lambda_2^3 + \frac{3}{8}\lambda_3^1 + \frac{5}{8}\lambda_3^2 + \frac{7}{8}\lambda_3^3 \leq 1 \tag{25}$$

is valid for P_{SC}. Inequality (25) cuts off $\lambda_1^3 = 1$, $\lambda_3^1 = \frac{2}{5}$, and $\lambda_j^k = 0$ otherwise, which is a vertex of the feasible set of the LP relaxation that satisfies the SOS2′ constraints, but not semi-continuous, corresponding to $x_1 = 2$, $x_2 = 0$, and $x_3 = 2$. □

Finally, we give inequalities (16) and (18) strengthened for when there are semi-continuous constraints.

Definition 5 Let $C \subseteq N$ and $l_j \in \{2, \ldots, T\} \cup \{k_j^*\}\ \forall j \in C$ be such that

$$\sum_{j \in C} a_j^{l_j} = b + \rho$$

with $\rho > 0$. The set C is a *semi-continuous cover*. □

We now give the third and fourth SC-PLO cuts.

Theorem 8 (Zhao and de Farias [32]) *Let C be a semi-continuous cover, C_1, C_2, l_j, α_j, β_j, and γ_j be as in Theorem 3. Let $C_2^* = \{j \in C_2 : l_j = k_j^*\}$. Then,*

$$\sum_{j \in C_1} \left(\alpha_j \lambda_j^{l_j - 2} + \beta_j \lambda_j^{l_j - 1} + \sum_{k=l_j}^{T} \lambda_j^k \right) + \sum_{j \in C_2^*} \sum_{k=l_j}^{T} \lambda_j^k + \sum_{j \in C_2 - C_2^*} \left(\gamma_j \lambda_j^{l_j - 1} + \sum_{k=l_j}^{T} \lambda_j^k \right) \leq |C| - 1 \tag{26}$$

is valid for P_{SC}. □

Theorem 9 (Zhao and de Farias [32]) *Let C be a semi-continuous cover, $C^* = \{j \in C : l_j = k_j^*\}$, $a_C = max\{a_i^{l_i} : \forall i \in C\}$, $\bar{C} = \{j \in N - C : a_j^T \geq a_C$ and $a_j^{T-1} \geq a_C - \rho\}$, and $t_j = min\{k : a_j^k \geq a_C$ and $a_j^{k-1} \geq a_C - \rho\}$ $\forall j \in \bar{C}$. Suppose that $\bar{C} \neq \emptyset$. Then,*

$$\sum_{j \in C^*} \sum_{k=l_j}^{T} \lambda_j^k + \sum_{j \in C - C^*} \left(\gamma_j \lambda_j^{l_j - 1} + \sum_{k=l_j}^{T} \lambda_j^k \right) + \sum_{j \in \bar{C}} \sum_{k=t_j}^{T} \lambda_j^k \leq |C| - 1 \tag{27}$$

is valid for P_{SC}. □

Example 4 (Continued) Suppose that $x_1 \in \{0\} \cup [2, 8]$, $x_2 \in \{0\} \cup [3, 10]$, $x_3 \in \{0\} \cup [4, 10]$, and $x_4 \in \{0\} \cup [5, 9]$. We then have that $k_1^* = k_2^* = k_3^* = k_4^* = 1$. From Theorem 8,

$$-2\lambda_1^1 + \lambda_1^3 + \lambda_3^1 + \lambda_3^2 + \lambda_3^3 \leq 1$$

is valid for P_{SC} and cuts off the point $\lambda_1^3 = 1, \lambda_3^1 = \frac{1}{2}, \lambda_j^k = 0$ otherwise, which is a vertex of the feasible set of the LP relaxation that satisfies SOS2′, but not semi-continuous, corresponding to $x_1 = 8, x_2 = x_4 = 0$, and $x_3 = 2$.

From Theorem 9,

$$\lambda_1^1 + \lambda_1^2 + \lambda_1^3 + \lambda_3^1 + \lambda_3^2 + \lambda_3^3 + \lambda_4^1 + \lambda_4^2 + \lambda_4^3 + \lambda_2^3 \leq 2$$

is valid for P_{SC} and cuts off the point $\lambda_1^1 = \lambda_3^1 = 1, \lambda_4^1 = \frac{4}{5}, \lambda_j^k = 0$, which is a vertex of the feasible set of the LP relaxation that satisfies SOS2′, but not semi-continuous, corresponding to $x_1 = 2, x_2 = 0, x_3 = 4$, and $x_4 = 4$. □

3 Separation heuristics

We conjecture that the separation problems for the inequalities given in the previous section are intractable. This is due to the inequalities' "complicated appearance". So we give (rather simple) heuristics for the separation problems. Specifically, we give separation heuristics for inequalities (11), (13), (16), (18), and (21). [Because the separation heuristics for (22), (24), (26), and (27) are virtually the same as those for (11), (13), (16), and (18), respectively, we omit them].

For the remainder of this section we let $\bar{\lambda}$ be the optimal LP relaxation solution just obtained. Let $j \in N$ and suppose that $\bar{\lambda}_j^k > 0$ for some $k \in K$. We let

$$\epsilon_j = min\{k \in K : \bar{\lambda}_j^k > 0\}$$

and

$$\phi_j = max\{k \in K : \bar{\lambda}_j^k > 0\}.$$

The heuristics consist in building inequalities of the respective families that have good chances of being violated by $\bar{\lambda}$. The constraints (10) from which we derive the cuts are the ones that are satisfied at equality by $\bar{\lambda}$, and we repeat the procedure below for each one of these constraints.

Inequality (11) When $N^- \neq \emptyset$, we take, for simplicity, $N_1^- = \emptyset$. So the inequality is determined by choosing j and s, and, when (12) holds, one of the two alternative values for $(\alpha_i^{k_i-1}, \alpha_i^{k_i})$. The choice between the two alternatives is made as follows: for every $i \in I$, if $\alpha_i^{k_i-1}\bar{\lambda}_i^{k_i-1} + \alpha_i^{k_i}\bar{\lambda}_i^{k_i} \leq 0$, we take $(\alpha_i^{k_i-1}, \alpha_i^{k_i}) = (0, 0)$, otherwise we choose the other option.

For every $j \in N$, we test whether $\bar{\lambda}$ violates (11) with $s = 1$ and, in case $\epsilon_j > 1$, we also test whether $\bar{\lambda}$ violates (11) with $s = \epsilon_j$. If it does, we add the inequality to the cut pool.

Inequality (13) Again, we take $N_1^- = \emptyset$ when $N^- \neq \emptyset$, so the inequality is determined by choosing j and s. For every $j \in N$ and $s = 2, \ldots, T-1$, we test whether $\bar{\lambda}$ violates (13) and, if it does, we add the inequality to the cut pool.

Inequality (16) For every $j \in N$, we construct covers as follows: $C = \{j\} \cup \{i \in N : i \neq j \text{ and } \epsilon_i \geq 2\}$, with $2 \leq l_j \leq \epsilon_j, l_i = \epsilon_i$, and provided $\sum_{j \in C} a_j^{l_j} > b$. We let

$$C_1 = \left\{ j \in C : \epsilon_j \geq 3 \text{ and } a_j^{\epsilon_j - 1} > b - \sum_{i \in C - \{j\}} a_i^{\epsilon_i} \right\},$$

and $C_2 = C - C_1$. We then test whether $\bar{\lambda}$ violates (16) and, if it does, we add the inequality to the cut pool.

Inequality (18) We construct covers C as in the previous case, and $\bar{C}$ as in Theorem 4. We then test whether $\bar{\lambda}$ violates (18) and, if it does, we add the inequality to the cut pool.

Inequality (21) For every $j \in N^+$, we construct generalized covers as follows: $C^+ = \{j\} \cup \{i \in N^+ : i \neq j \text{ and } \epsilon_i \geq 2\}$, with $2 \leq l_j \leq \epsilon_j$ and $l_i = \epsilon_i$, $C^- = \{i \in N^- : \phi_i \geq 1\}$, with $l_i = \phi_i$, and provided $\sum_{j \in C^+} a_j^{l_j} - \sum_{j \in C^-} a_j^{l_j} > b$. We let

$$C_1 = \left\{ j \in C^+ : \epsilon_j \geq 3 \text{ and } a_j^{\epsilon_j - 1} > b - \sum_{i \in C^+ - \{j\}} a_i^{\epsilon_i} \right\},$$

and $C_2 = C^+ - C_1$. We then test whether $\bar{\lambda}$ violates (21) and, if it does, we add the inequality to the cut pool.

In all of the above procedures, a *minimum violation* had to be met by $\bar{\lambda}$ in order for an inequality to be added to the cut pool. This absolute value was chosen based on preliminary tests, and differed depending on the type of problem tested. For the *transshipment* instances, the minimum violation was 0.3, and for the *transportation* instances it was 0.05. Starting in Sect. 4, we will see that GUROBI filters the user cuts before adding them to the cutpool. Our cut violation filter is done before that, and is implemented in our separation routine. We performed separation at every node of the enumeration tree in all instances we tested.

4 Platform, problems and instances, and tests conducted

4.1 Platform

We performed our computational tests in the Texas Tech High Performance Computing Center's Intel Xeon E5450 3.0 GHz CPU with 16 GB RAM nodes (two CPUs on a single board for each node) [17]. We used the callable libraries of CPLEX 12.2.0.0 and GUROBI 4.0.0. We ran both on a single thread. The computational times and number of branch-and-bound nodes for CPLEX and GUROBI were different in all instances, but in almost all cases they led to the same conclusions. For this reason, we report the results of GUROBI only. In the small number of cases in which CPLEX and GUROBI disagree we report the results of both.

4.2 Problems

We tested branch-and-cut with the inequalities of Sect. 2.1 on difficult and large instances of the *transshipment problem*:

$$\begin{aligned} minimize \quad & \sum_{i \in M} \sum_{j \in M - \{i\}} g_{ij}(x_{ij}) \\ s.t. \quad & \sum_{j \in M - \{i\}} (x_{ij} - x_{ji}) = b_i, \quad i \in M \\ & x_{ij} \geq 0, \quad i, j \in M, i \neq j, \end{aligned}$$

where $M = \{1, \ldots, m\}$ is the set of nodes of the network and g_{ij} is a continuous concave piecewise linear function $\forall i, j \in M$. We assume WLOG that $\sum_{i \in M} b_i = 0$.

We also tested the cuts on difficult and large instances of the *transportation problem*:

$$\begin{aligned} \textit{minimize} \quad & \sum_{i\in I}\sum_{j\in J} g_{ij}(x_{ij}) \\ s.t. \quad & \sum_{j\in J} x_{ij} = \sigma_i, \quad i \in I \\ & \sum_{i\in I} x_{ij} = \delta_j, \quad j \in J \\ & x_{ij} \geq 0, \quad i \in I, j \in J, \end{aligned}$$

where I is the set of supply nodes, J the set of demand nodes, σ_i the supply of node i, and δ_j the demand of node j. We assume WLOG that $\sum_{i\in I}\sigma_i = \sum_{j\in J}\delta_j$. Additionally, we tested the largest instances of Vielma and Nemhauser [30], which are transportation instances, but much smaller than ours. Finally, we tested the inequalities of Sect. 2.2 on instances of the transportation problem with semi-continuous constraints.

In terms of the λ variables, the transshipment model becomes:

$$\begin{aligned} \textit{minimize} \quad & \sum_{i\in M}\sum_{j\in M-\{i\}}\sum_{k\in K} c_{ij}^k \lambda_{ij}^k \\ s.t. \quad & \sum_{j\in M-\{i\}}\sum_{k\in K} a_{ij}^k \lambda_{ij}^k - \sum_{j\in M-\{i\}}\sum_{k\in K} a_{ji}^k \lambda_{ji}^k = b_i, \quad i \in M \qquad (28) \\ & \sum_{k\in K} \lambda_{ij}^k \leq 1, \quad i, j \in M, i \neq j \\ & \lambda_{ij}^k \geq 0, \quad i, j \in M, i \neq j, k \in K \\ & \{\lambda_{ij}^1, \ldots, \lambda_{ij}^T\} \text{ is } SOS2', \quad i, j \in M, i \neq j \end{aligned}$$

and the transportation model becomes:

$$\begin{aligned} \textit{minimize} \quad & \sum_{i\in I}\sum_{j\in J}\sum_{k\in K} c_{ij}^k \lambda_{ij}^k \\ s.t. \quad & \sum_{j\in J} a_{ij}^k \lambda_{ij}^k = \sigma_i, \quad i \in I \qquad (29) \\ & \sum_{i\in I} a_{ij}^k \lambda_{ij}^k = \delta_j, \quad j \in J \qquad (30) \\ & \sum_{k\in K} \lambda_{ij}^k \leq 1, \quad i \in I, j \in J \\ & \lambda_{ij} \geq 0, \quad i \in I, j \in J, k \in K \\ & \{\lambda_{ij}^1, \ldots, \lambda_{ij}^T\} \text{ is } SOS2', \quad i \in I, j \in J. \end{aligned}$$

For both transshipment and transportation the a_{ij}^k's are the values of the breakpoints and are positive. Because $N^- \neq \emptyset$ for transshipment, the inequalities used for the transshipment instances are (11), (13), and (21). On the other hand, since $N^- = \emptyset$ for transportation, the inequalities used for the transportation instances are (11), (13), (16) and (18). For transportation with semi-continuous constraints we took $k_i^* = 1\ \forall i \in N$ [in this case, of course, the inequalities are (22), (24), (26), and (27)].

Note that because they are equality constraints, all constraints (28) for transshipment, and (29) and (30) for transportation, will be used for cut separation in all attempts to cut off an optimal solution to the LP relaxation that does not satisfy SOS2′.

4.3 Instances

In part we followed [19] for generating our instances.

Transshipment We took $m \in \{15, 20, 25, 30, 40, 50, 60, 70, 80, 90, 100\}$ and $T \in \{3, 4, 5, 6, 10, 15, 20, 25, 30\}$. So the instances we tested are considerably larger than those of [19], and so was the range of values of T.

Each node of the network is chosen to be supply, demand, or transshipment with probability $\frac{1}{3}$, except for node m, which, as we show next, will become supply, demand, or transshipment to balance the network, i.e. to guarantee $\sum_{i \in M} b_i = 0$. For $i \in \{1, \ldots, m-1\}$, we choose $|b_i|$ to be an integer uniformly distributed between 1 and $10T$ if it is a supply or demand node (positive if supply, negative if demand), and 0 if it is a transshipment node. As for node m, $b_m = -\sum_{i=1}^{m-1} b_i$. Node m is supply, demand, or transshipment according to whether b_m is positive, negative, or 0, respectively.

Regarding the breakpoints, a_{ij}^1 is an integer uniformly distributed between 1 and 10. For a_{ij}^k with $k \in \{2, \ldots, T\}$,

$$a_{ij}^k = a_{ij}^{k-1} + u_{ij}^k,$$

where u_{ij}^k is an integer uniformly distributed between 1 and 10.

Finally,

$$c_{ij}^k = c_{ij}^{k-1} + 5(T-k) + C_{ij}^k,$$

where $c_{ij}^1 = 5(T-1) + C_{ij}^k$, and C_{ij}^k is an integer uniformly distributed between 1 and 5. This way, the objective function is nondecreasing concave.

Transportation We took $T \in \{5, 10, 20, 30, 32, 40, 80, 120\}$. So we tested a much wider range of values of T than [19]. For transportation with semi-continuous constraints we took $T \in \{5, 10\}$. We took $|J| \in \{10, 20, 50, 100, 200, 300, 400\}$ and $|I| \in \{10, 25, 50, 100\}$. So the instances we tested are considerably larger than those of [19]. For transportation with semi-continuous constraints we took $|I| \in \{5, 7, 8, 10\}$ and $|J| \in \{14, 16, 20\}$. In either case (with or without semi-continuous constraints) $|I| \leq |J|$.

Let $\sigma_{min} = T + 1$ and

$$\sigma_{max} = \left\lfloor \frac{2|J|(T+1)}{|I|} - T - 1 \right\rfloor.$$

Initially, we take for $\sigma_i, i \in I$, an integer uniformly distributed between σ_{min} and σ_{max}. Likewise, let $\delta_{min} = T + 1$ and $\delta_{max} = T + 20$. Initially, we take for $\delta_j, j \in J$, an integer uniformly distributed between δ_{min} and δ_{max}.

If $\sum_{i \in I} \sigma_i = \sum_{j \in J} \delta_j$, the initial values for σ_i and δ_j will be their final values. Now, suppose that $\sum_{i \in I} \sigma_i > \sum_{j \in J} \delta_j$. We then take the initial values of the σ_i's to be their final values and adjust the values of the δ_j's. The adjustment is made by adding 1 unit to the values of $\delta_1, \delta_2, \ldots$ in this order until $\sum_{i \in I} \sigma_i = \sum_{j \in J} \delta_j$. If, after adding one unit to the value $\delta_{|J|}$ we still have $\sum_{i \in I} \sigma_i > \sum_{j \in J} \delta_j$, we repeat the process until $\sum_{i \in I} \sigma_i = \sum_{j \in J} \delta_j$.

If, on the other hand, $\sum_{i \in I} \sigma_i < \sum_{j \in J} \delta_j$, we take the initial values of the δ_j's to be their final values and adjust the values of the σ_i's as described above.

For each variable x_{ij} we take $a_{ij}^T = min\{\sigma_i, \delta_j\}$. We obtain the other breakpoints $a_{ij}^k, k \in K - \{T\}$, in two steps. First, we break the interval $[1, a_{ij}^T]$ into subintervals $[p_{ij}^0, p_{ij}^1], \ldots, [p_{ij}^{k-1}, p_{ij}^k], \ldots, [p_{ij}^{T-1}, p_{ij}^T]$, with $p_{ij}^0 = 1$ and $p_{ij}^T = a_{ij}^T$. Then, we choose $a_{ij}^k, k \in K - \{T\}$, to be a number between p_{ij}^{k-1} and p_{ij}^k. The reason for this is to space the breakpoints more evenly, which in our initial computational tests produced harder instances. Now, for $k \in K - \{T\}$, $p_{ij}^k = 1 + P_{ij}^k$, where P_{ij}^k is an integer uniformly distributed between p_{ij}^{k-1} and $a_{ij}^T - T + k - 1$; and $a_{ij}^k, k \in K - \{T\}$, is an integer uniformly distributed between p_{ij}^{k-1} and $p_{ij}^k - 1$. Finally, c_{ij}^k is same as in the transshipment instances.

4.4 Tests conducted

We evaluate the performance of GUROBI on the piecewise linear transshipment and transportation instances with and without the addition of PLO cuts. In the latter case we consider, separately, the problem with and without semi-continuous constraints. We examine how much preprocessing, primal heuristics, and cutting planes influence the number of enumeration nodes and computational time. In other words, we evaluate GUROBI with and without PLO cuts, and:

- preprocessing, primal heuristics, and GUROBI cuts off, which we call branch-and-bound setting
- precisely one of preprocessing, primal heuristics, or GUROBI cuts on (and the other two off)
- default setting.

Depending on the results, we performed additional tests. For example, when GUROBI cuts appeared to be relevant, we tried to identify which ones are the most relevant. Also, we minded the GUROBI parameters. The most important example is the addition of user cuts to the cutpool. GUROBI filters the user cuts before adding them to the cutpool, and as a result sometimes only a small fraction of them are actually used. We tuned parameters to see when using more cuts than allowed by default would lead to a better performance. All parameter tuning will be explicitly mentioned. To identify which formulation performs best, we conducted all tests above with the LOG, MIP, and SOS2 formulations.

We note that the difference between the optimal value of the instances and the optimal value of their LP relaxations is extremely small. For this reason, we do not report the optimal values or the integrality gaps of the instances. We note that root node gap improvement is a good indicator of the quality of cuts. However, extremely small root node gap is typical of PLO, see for example [9,10,12,19], and for this reason we rely solely on number of nodes and computational time for assessing branch-and-cut strategies, particularly cut efficiency.

Finally, due to limited space available for publishing tables in the journal, we were not able to include in this report the tables that give the number of cutting planes, both PLO and MIP default cuts of CPLEX and GUROBI. We refer to [11] for such information.

5 Computational results for transshipment instances

We tested 85 piecewise linear transshipment instances. The instances can be grouped into 17 sets with same number m of nodes and same value of T. Because preliminary tests indicated these instances to be extremely difficult, we allowed for a maximum computational time of 7,200 seconds of CPU. Here, in most cases, turning the "Aggressive MIP Cuts" option on led to a better performance than default or the "Very Aggressive MIP Cuts" option. Therefore, we report on this option. For the remainder of this section, default refers to GUROBI's default setting, except for "Aggressive MIP Cuts" on.

In Tables 1, 2, and 3 we compare the average solution time and the number of instances solved to proven optimality, and in Tables 4, 5, and 6 the average number of enumeration nodes, for GUROBI in: default setting; branch-and-bound (henceforth denoted B&B in the tables); default with PLO cuts (11), (13), and (21) added; and branch-and-bound with (11), (13), and (21) added.

In all tables each entry refers to 5 instances. We computed both arithmetic and geometric averages, but because they led to the same conclusions, we report only the arithmetic averages. Tables 1 and 4 give results for the LOG formulation, Tables 2 and 5 for the MIP formulation, and Tables 3 and 6 for the SOS2 approach. In Tables 1, 2, 3, 4, 5 and 6, as well as in the other tables where it appears, "%Red." is percentage reduction relative to the default setting. The GUROBI default MIP cutting planes separated (in the "usual" MIP and LOG formulations) were implied bounds, cover, flow cover, Gomory, and MIR.

The impact of PLO cuts both on computational time and number of enumeration nodes was enormous (cf. Tables 1, 2, 3, 4, 5, 6, respectively). Out of the 85 instances tested, GUROBI in default setting was able to solve to proven optimality only 44 instances (52 %), as opposed to 74 (87 %) with PLO cuts, using the LOG model; 29 (34 %), as opposed to 69 (81 %), using the MIP model; and 35 (41 %), as opposed to 68 (80 %), using the SOS2 approach. The average reduction from default in computational time by adding PLO cuts was 63 % with the LOG model; 60 % with the MIP model; and 59 % with the SOS2 approach (cf. Tables 1, 2, 3). The average reduction from default in the number of enumeration nodes by adding PLO cuts was 98 % with the LOG and MIP models, and 99 % with the SOS2 approach (cf. Tables 4, 5, 6). In summary, the use of PLO cuts improved our ability to solve the instances tremendously, not only

Table 1 Average solution time and # of instances solved for transshipment with the LOG model

m.T	Default		B&B			Def. + PLO cuts			B&B + PLO cuts		
	Time	#Sol.	Time	%Red.	#Sol.	Time	%Red.	#Sol.	Time	%Red.	#Sol.
15.20	19	5	16	15.8	5	22	−15.8	5	34	−78.9	5
15.25	49	5	26	46.9	5	23	53.1	5	32	34.7	5
15.30	26	5	17	34.6	5	46	−76.9	5	43	−65.4	5
20.20	208	5	117	43.8	5	44	78.8	5	39	81.3	5
25.15	1,617	4	853	47.2	5	134	91.7	5	142	91.2	5
25.20	2,377	4	2,426	−2.1	4	194	91.8	5	207	91.3	5
30.6	1,854	5	1,074	42.1	5	81	95.6	5	59	96.8	5
30.10	3,287	3	2,844	13.5	4	119	96.4	5	134	95.9	5
30.15	3,325	3	3,142	5.5	3	299	91.0	5	358	89.2	5
40.10	5,089	2	5,384	−5.8	2	525	89.7	5	675	86.7	5
50.6	7,200	0	7,078	1.7	1	872	87.9	5	1,733	75.9	4
60.4	6,685	1	7,200	−7.7	0	708	89.4	5	1,134	83.0	5
70.3	5,772	2	7,200	−24.7	0	289	95.0	5	292	94.9	5
70.5	7,200	0	7,200	0.0	0	3,133	56.5	4	3,420	52.5	3
80.5	7,200	0	7,200	0.0	0	4,949	31.3	4	6,079	15.6	1
90.5	7,200	0	7,200	0.0	0	6,160	14.4	1	6,766	6.0	1
100.5	7,200	0	7,200	0.0	0	7,200	0.0	0	7,200	0.0	0
Average	3,900	2.6	3,893	0.2	2.6	1,459	62.6	4.4	1,667	57.2	4.1
Total	66,308	44	66,177	0.2	44	24,798	62.6	74	28,347	57.2	69

by reducing their computational time and number of enumeration nodes, but also by solving many instances that GUROBI could not solve without them.

Even on "Aggressive MIP Cuts" mode, GUROBI added to the cutpool only 8 % of the PLO cuts separated for LOG in default setting and 6 % in branch-and-bound; 8 % for MIP in default setting and 7 % in branch-and-bound; 9 % for SOS2 in default setting and 7 % in branch-and-bound. CPLEX, on the other hand, added the great majority of the PLO cuts separated. As expected, the percentage reduction in the number of enumeration nodes due to PLO cuts was greater with CPLEX than GUROBI. However, the percentage reduction in computational time was greater with GUROBI. GUROBI filters out cuts that are "relatively similar" to cuts already in the cutpool (e.g. cutting planes for which the angle between their normal vectors is smaller than a certain amount [16]). Obviously several factors may be involved here. But because the cuts were so efficient in reducing computational time and number of enumeration nodes, we conjecture that the positive impact of the filter is due to the elimination of "similar" cuts, rather than inefficient cuts.

We now turn our attention to the impact of the branch-and-cut features present in GUROBI default (preprocessing, primal heuristics, and MIP cutting planes) on the number of enumeration nodes and computational time for the instances. First we note that of these features, the only ones present in the SOS2 approach of GUROBI are preprocessing and primal heuristics.

Table 2 Average solution time and # of instances solved for transshipment with the MIP model

$m.T$	Default		B&B			Def. + PLO cuts			B&B + PLO cuts		
	Time	#Sol.	Time	%Red.	#Sol.	Time	%Red.	#Sol.	Time	%Red.	#Sol.
15.20	126	5	139	−10.3	5	65	48.4	5	39	69.0	5
15.25	257	5	261	−1.6	5	106	58.8	5	59	77.0	5
15.30	1,435	5	864	39.8	5	147	89.8	5	114	92.1	5
20.20	720	5	814	−13.1	5	93	87.1	5	64	91.1	5
25.15	5,651	2	3,933	30.4	3	578	89.8	5	375	93.4	5
25.20	4,977	2	4,921	1.1	2	564	88.7	5	417	91.6	5
30.6	5,107	2	4,539	11.1	2	198	96.1	5	98	98.1	5
30.10	5,611	2	5,770	−2.8	1	335	94.0	5	283	95.0	5
30.15	5,874	1	5,889	−0.3	1	2,121	63.9	5	1,042	82.3	5
40.10	7,200	0	6,887	4.3	1	2,550	64.6	5	1,309	81.8	5
50.6	7,200	0	7,200	0.0	0	2,167	69.9	4	1,978	72.5	4
60.4	7,200	0	7,200	0.0	0	1,530	78.8	5	1,137	84.2	5
70.3	7,200	0	7,200	0.0	0	485	93.3	5	452	93.7	5
70.5	7,200	0	7,200	0.0	0	3,639	49.5	3	3,568	50.4	3
80.5	7,200	0	7,200	0.0	0	6,131	14.8	1	5,908	17.9	2
90.5	7,200	0	7,200	0.0	0	6,877	4.5	1	6,951	3.5	1
100.5	7,200	0	7,200	0.0	0	7,200	0.0	0	7,200	0.0	0
Average	5,139	1.7	4,966	3.4	1.8	2,046	60.2	4.1	1,823	64.5	4.1
Total	87,358	29	84,417	3.4	30	34,786	60.2	69	30,994	64.5	70

The number of nodes and computational time for LOG, MIP, and SOS2 when only preprocessing is on or when only primal heuristics is on is almost the same as branch-and-bound. So, of these alternatives, we only report the results for branch-and-bound. On the other hand, the results for when only the MIP cuts are on (for the LOG and MIP formulations) are almost the same as default. So of these two, we only report the results for default. Thus, we compare the performance of default with branch-and-bound, first without the addition of PLO cuts. We note that in the case of the SOS2 approach, the performance of default and branch-and-bound are not significantly different, which is not surprising in the light of the previous discussion on the efficiency of GUROBI's pre-processing and primal heuristics for these instances.

The default setting reduced the number of enumeration nodes of branch-and-bound by 65 % for LOG and 64 % for MIP (cf. Tables 4 and 5). As mentioned above, this reduction is mostly due to MIP cutting planes. For the LOG model, the vast majority of cuts were Gomory, specifically 90 % of the MIP cuts. For the MIP model, flow cover was the most used, 81 % total, while Gomory was a distant second place, 16 %. This is to be expected, since the MIP formulation exposes the flow cover structure explicitly in the model, which is not the case with the LOG formulation.

Despite the significant reduction in number of nodes, the use of MIP cuts in GUROBI default increased the computational time. At first, the average increase seems

Table 3 Average solution time and # of instances solved for transshipment with the SOS2 approach

m.T	Default		B&B			Def. + PLO cuts			B&B + PLO cuts		
	Time	#Sol.	Time	%Red.	#Sol.	Time	%Red.	#Sol.	Time	%Red.	#Sol.
15.20	171	5	154	9.9	5	38	77.8	5	30	82.5	5
15.25	122	5	130	−6.6	5	44	63.9	5	41	66.4	5
15.30	198	5	174	12.1	5	112	43.4	5	65	67.2	5
20.20	333	5	323	3.0	5	73	78.1	5	43	87.1	5
25.15	2,185	4	2,367	−8.3	4	267	87.8	5	186	91.5	5
25.20	4,529	2	4,511	0.4	2	323	92.9	5	287	93.7	5
30.6	3,220	4	3,202	0.6	4	65	98.0	5	47	98.5	5
30.10	5,162	2	5,227	−1.3	2	131	97.5	5	104	98.0	5
30.15	5,368	2	5,577	−3.9	2	652	87.9	5	562	89.5	5
40.10	6,007	1	5,994	0.2	1	813	86.5	5	765	87.3	5
50.6	7,200	0	7,200	0.0	0	1,855	74.2	4	1,779	75.3	4
60.4	7,200	0	7,200	0.0	0	2,269	68.5	5	1,627	77.4	5
70.3	7,200	0	7,200	0.0	0	623	91.3	5	494	93.1	5
70.5	7,200	0	7,200	0.0	0	3,962	45.0	3	3,652	49.3	3
80.5	7,200	0	7,200	0.0	0	6,072	15.7	1	6,091	15.4	1
90.5	7,200	0	7,200	0.0	0	7,200	0.0	0	7,200	0.0	0
100.5	7,200	0	7,200	0.0	0	7,200	0.0	0	7,200	0.0	0
Average	4,570	2.1	4,592	−0.5	2.1	1,865	59.2	4.0	1,775	61.2	4.0
Total	77,695	35	78,059	−0.5	35	31,699	59.2	68	30,173	61.2	68

modest, 0.2 % for LOG and 3.4 % for MIP (cf. Tables 1, 2). However, if we average the percentage reduction in time for branch-and-bound over default for the instance sizes where either one of default or branch-and-bound was solved to proven optimality (first 13 entries of Table 1 and first 10 entries of Table 2) we obtain different results, 16 % for LOG and 6 % for MIP. This shows that the increase is significant for LOG and higher than MIP's. We also note that for LOG, in a few cases the increase was almost 50 % (entries 15.25, 20.20, 25.15, and 30.6 of Table 1), while for MIP the individual increases were smaller. A possible explanation for this is the large presence of Gomory cuts. Because they are dense, when they are used in large quantities, typically, a reduction of about 60 % in the number of enumeration nodes does not decrease computational time. Remember that many more Gomory cuts were added for LOG than for MIP.

This computational time result is in agreement with tests conducted previously, see for example [10,19], on platforms that are different. There, as well as here, the use of MIP tools for PLO (and other combinatorial problems of similar type), did not improve the computational time required to solve the problem, and in some cases made it worse.

Now we compare the performance of default and branch-and-bound with PLO cuts added to both. As in the previous case, their performances are not too different for the SOS2 approach. On the other hand, the default setting reduced the number of

Table 4 Average number of enumeration nodes for transshipment with the LOG model

$m.T$	Default	B&B		Def. + PLO cuts		B&B + PLO cuts	
	Nodes	Nodes	%Red.	Nodes	%Red.	Nodes	%Red.
15.20	9,972	11,705	−17.4	546	94.5	952	90.5
15.25	12,268	16,834	−37.2	342	97.2	660	94.6
15.30	6,310	8,040	−27.4	652	89.7	741	88.3
20.20	29,023	42,652	−47.0	438	98.5	606	97.9
25.15	168,839	249,014	−47.5	1,204	99.3	1,621	99.0
25.20	144,815	449,428	−210.3	1,222	99.2	1,689	98.8
30.6	241,763	451,226	−86.6	1,592	99.3	2,038	99.2
30.10	222,788	642,184	−188.2	1,032	99.5	1,775	99.2
30.15	230,829	517,038	−124.0	2,171	99.1	3,269	98.6
40.10	197,821	613,385	−210.1	2,790	98.6	4,542	97.7
50.6	268,889	740,425	−175.4	4,360	98.4	12,436	95.4
60.4	321,817	1,253,942	−289.6	5,322	98.3	13,643	95.8
70.3	235,155	753,397	−220.4	1,839	99.2	4,384	98.1
70.5	126,790	407,905	−221.7	6,705	94.7	13,159	89.6
80.5	83,305	296,901	−256.4	6,601	92.1	16,559	80.1
90.5	62,540	223,295	−257.0	5,532	91.2	12,472	80.1
100.5	42,946	161,051	−275.0	2,885	93.3	6,763	84.3
Average	141,522	402,260	−184.2	2,661	98.1	5,724	96.0
Total	2,405,870	6,838,422	−184.2	45,233	98.1	97,309	96.0

enumeration nodes of branch-and-bound by 54 % for LOG and 36 % for MIP (cf. Tables 4, 5), and it reduced the computational time by 13 % for LOG and 11 % for MIP (cf. Tables 1, 2). So even though branch-and-bound was better than default when the PLO cuts were not added, in our tests the best performance overall was obtained when the PLO cuts were added on the top of the default setting.

We now discuss the effect of formulation on the computational results, first without the addition of the PLO cuts. Of the three approaches, LOG was the most efficient, followed by SOS2, followed by MIP. The LOG formulation solved to proven optimality 44 (52 %) instances in default and in branch-and-bound, as opposed to 35 (41 %) in default and in branch-and-bound for SOS2, and 29 (34 %) in default and 30 (35 %) in branch-and-bound for MIP (cf. Tables 1, 2, 3). In number of nodes, LOG reduced the number of SOS2 nodes by 93 % in default and by 81 % in branch-and-bound, and the number of MIP nodes by 46 % in default and 45 % in branch-and-bound (cf. Tables 4, 5, 6). In computational time, LOG reduced the computational time of SOS2 by 15 % in default and in branch-and-bound, and the computational time of MIP by 24 % in default and 22 % in branch-and-bound (cf. Tables 1, 2, 3).

The superiority of SOS2 over MIP (even when equipped with cutting planes, primal heuristic, and pre-processing) had already been noted and analyzed in the literature, see for example [9,10,19]. The superiority of LOG over SOS2 was noted in [29,30] when T (the number of partitions) is large, but it was not analyzed. Here again LOG

Table 5 Average number of enumeration nodes for transshipment with the MIP model

$m.T$	Default	B&B		Def. + PLO cuts		B&B + PLO cuts	
	Nodes	Nodes	%Red.	Nodes	%Red.	Nodes	%Red.
15.20	70,452	89,675	−27.3	1,510	97.9	1,302	98.2
15.25	109,275	127,857	−17.0	1,701	98.4	1,324	98.8
15.30	470,910	360,488	23.4	2,822	99.4	2,385	99.5
20.20	194,841	237,535	−21.9	1,100	99.4	1,135	99.4
25.15	664,683	923,245	−38.9	5,107	99.2	5,346	99.2
25.20	485,813	822,335	−69.3	4,350	99.1	3,652	99.2
30.6	621,309	2,052,059	−230.3	3,920	99.4	3,346	99.5
30.10	715,605	1,312,370	−83.4	3,486	99.5	5,093	99.3
30.15	287,184	859,853	−199.4	17,696	93.8	11,628	96.0
40.10	291,834	754,546	−158.6	14,219	95.1	10,528	96.4
50.6	165,811	1,144,273	−590.1	6,608	96.0	14,357	91.3
60.4	128,332	1,086,020	−746.3	5,219	95.9	14,049	89.1
70.3	106,304	928,584	−773.5	1,657	98.4	6,255	94.1
70.5	64,967	642,132	−888.4	4,536	93.0	14,695	77.4
80.5	38,637	465,171	−1,104.0	4,137	89.3	16,002	58.6
90.5	28,965	341,841	−1,080.2	3,363	88.4	12,302	57.5
100.5	19,351	226,762	−1,071.8	1,292	93.3	6,491	66.5
Average	262,604	727,926	−177.2	4,866	98.1	7,641	97.1
Total	4,464,273	12,374,746	−177.2	82,723	98.1	129,890	97.1

is superior to SOS2. It remains to determine why. The first possible explanation is that LOG allows for the use of all MIP features present in GUROBI, especially MIP cutting planes. But in the present tests, branch-and-bound was superior to default for LOG, so that cannot be the reason. Another possibility is GUROBI's implementation of SOS2 branching being inferior to its 0–1 variable dichotomy branching implementation. As we will see in Sect. 7, different implementations of SOS2 branching can lead to much difference in performance. One other possibility is that the LOG formulation may be breaking some of the symmetry inherent in transshipment with SOS2. We tried to explore further this possibility by repeating all tests with GUROBI in a mode where symmetry is explored aggressively. However, no improvement was detected (same with CPLEX). We note that in case LOG is exploring symmetry, a different branching scheme for SOS2 may eliminate the advantage of LOG and lead to an improved SOS2 approach.

With PLO cuts added, again LOG was superior to SOS2, which was superior to MIP. The LOG formulation solved to proven optimality 74 (87 %) instances in default and 69 (81 %) in branch-and-bound, as opposed to 68 (80 %) in default and branch-and-bound for SOS2, and 69 (81 %) in default and 70 (82 %) in branch-and-bound for MIP (cf. Tables 1, 2, 3). In number of nodes, LOG reduced the number of SOS2 nodes by 71 % in default and by 37 % in branch-and-bound, and the number of MIP nodes

Table 6 Average number of enumeration nodes for transshipment with the SOS2 approach

$m.T$	Default	B&B		Def. + PLO cuts		B&B + PLO cuts	
	Nodes	Nodes	%Red.	Nodes	%Red.	Nodes	%Red.
15.20	352,673	325,453	7.7	1,122	99.7	1,194	99.7
15.25	197,901	216,402	−9.3	925	99.5	1,107	99.4
15.30	280,909	254,896	9.3	2,182	99.2	1,622	99.4
20.20	396,332	394,383	0.5	955	99.8	847	99.8
25.15	2,112,977	2,354,152	−11.4	3,461	99.8	3,009	99.9
25.20	3,248,977	3,346,952	−3.0	2,515	99.9	2,965	99.9
30.6	4,538,896	4,698,361	−3.5	2,854	99.9	2,387	99.9
30.10	4,847,685	5,065,709	−4.5	2,015	100.0	2,228	100.0
30.15	3,324,306	3,618,237	−8.8	6,412	99.8	6,399	99.8
40.10	2,734,485	2,821,903	−3.2	6,327	99.8	6,587	99.8
50.6	3,156,942	3,307,116	−4.8	15,344	99.5	17,387	99.4
60.4	2,837,364	2,914,406	−2.7	31,410	98.9	26,271	99.1
70.3	2,502,456	2,574,268	−2.9	13,892	99.4	11,756	99.5
70.5	1,410,147	1,443,881	−2.4	21,143	98.5	20,721	98.5
80.5	939,498	971,422	−3.4	21,715	97.7	22,887	97.6
90.5	702,865	719,886	−2.4	16,957	97.6	17,726	97.5
100.5	522,391	535,181	−2.4	8,608	98.4	9,125	98.3
Average	2,006,283	2,091,918	−4.3	9,285	99.5	9,072	99.5
Total	34,106,804	35,562,608	−4.3	157,837	99.5	154,218	99.5

by 45 % in default and 25 % in branch-and-bound (cf. Tables 4, 5, 6). In computational time, LOG reduced the computational time of SOS2 by 22 % in default and 6 % in branch-and-bound, and the computational time of MIP by 29 % in default and 9 % in branch-and-bound (cf. Tables 1, 2, 3). Here again, we tested GUROBI in a mode that explores symmetry aggressively, but did not obtain different results (same with CPLEX). Overall, the best setting was LOG in default with PLO cuts.

6 Computational results for transportation instances

We tested 180 piecewise linear transportation instances. The instances can be grouped into 3 sets with same value of T, each containing 12 sets with same number $|I|$ of supply and $|J|$ of demand nodes. The maximum computational time allowed for each instance was 3,600 s of CPU. Here, unlike transshipment, adopting the "Aggressive MIP Cuts" or the "Very Aggressive MIP Cuts" option led to inferior performance in most cases. So we report on GUROBI's default setting.

Again unlike transshipment, default performed better than branch-and-bound for the LOG and MIP models, and almost the same for the SOS2 approach. For the LOG formulation, default solved to proven optimality 94 (52 %) instances, as opposed to 38 (21 %) by branch-and-bound, when no PLO cuts were added, and 113 (63 %), as

opposed to 95 (53 %), when the PLO cuts were added; in average, default reduced the number of nodes of branch-and-bound by 74 % and the computational time by 28 % when no PLO cuts were added, and the number of nodes by 45 % and the computational time by 20 % when the PLO cuts were added. For the MIP formulation, default solved to proven optimality 91 (51 %) instances, as opposed to 30 (17 %) by branch-and-bound, when no PLO cuts were added, and 107 (59 %), as opposed to 80 (44 %) by branch-and-bound, when the PLO cuts were added; in average, default reduced the number of nodes of branch-and-bound by 55 % and the computational time by 32 % when no PLO cuts were added, and the number of nodes by 11 % and the computational time by 26 % when the PLO cuts were added. So we only report results for default. The better performance of default over branch-and-bound is due almost entirely to the MIP cutting planes. In the aforementioned studies [10,19], MIP cuts were not efficient for transportation. Possibly, improvements in the strategies and implementations for MIP cutting planes made them more efficient here. The GUROBI default MIP cutting planes separated for transportation (in the "usual" MIP and LOG formulations) were the same as for transshipment, i.e. implied bounds, cover, flow cover, Gomory, and MIR.

In Tables 7, 8, and 9 we compare the average number of enumeration nodes and solution time, and the number of instances solved to proven optimality for GUROBI in default setting against default with PLO cuts (11), (13), (16) and (18) added. In all tables each entry refers to five instances. As with transshipment, we computed both arithmetic and geometric averages, and they led to the same conclusions. So again we report only the arithmetic averages. Table 7 gives results for the LOG formulation, Table 8 for the MIP formulation, and Table 9 for the SOS2 approach.

In average, GUROBI added to the cutpool 15 % of the PLO cuts separated for LOG, 10 % for MIP, and 14 % for SOS2. As with transshipment, the percentage reduction in the number of enumeration nodes due to PLO cuts was greater with CPLEX than GUROBI, but the percentage reduction in computational time was greater with GUROBI. The majority of GUROBI cutting planes added for LOG and MIP was flow cover, followed by Gomory. For LOG without PLO cuts, flow cover cuts were 55 % and Gomory 43 % of the GUROBI cuts; with PLO cuts, flow cover cuts were 69 % and Gomory 30 % of the GUROBI cuts. For MIP without PLO cuts, flow cover cuts were 54 % and Gomory 32 % of the GUROBI cuts; with PLO cuts, flow cover cuts were 64 % and Gomory 27 % of the GUROBI cuts. Because flow cover cuts are usually not as dense as Gomory, the more balanced use of flow cover and Gomory, with the majority being flow cover, may be one of the reasons MIP cuts were more efficient for transportation than for transshipment.

The impact of PLO cuts both on computational time and number of enumeration nodes was significant, but overall not as great as for transshipment (cf. Tables 7, 8, 9). Out of the 180 instances tested, GUROBI, in default setting, was able to solve to proven optimality 94 (52 %), as opposed to 113 (63 %) with the PLO cuts, using the LOG model; 91 (51 %), as opposed to 107 (59 %), using the MIP model; and 37 (21 %), as opposed to 90 (50 %), using the SOS2 approach. The average reduction from default in the number of enumeration nodes by adding the PLO cuts was 95 % with the LOG model, 92 % with the MIP model, and 99 % with the SOS2 approach. The average reduction from default in computational time by adding the PLO cuts

Table 7 Average solution time and node, and # of instances solved for transportation with the LOG model

$\|I\|.\|J\|.T$	Default			Default + PLO cuts				
	Node	Time	#Sol.	Node	%Red.	Time	%Red.	#Sol.
25.50.5	87,532	936	4	587	99.3	18	98.1	5
25.100.5	54,743	971	5	526	99.0	34	96.5	5
25.200.5	66,092	2,578	2	457	99.3	101	96.1	5
25.300.5	49,010	3,600	0	232	99.5	103	97.1	5
25.400.5	35,873	3,600	0	631	98.2	479	86.7	5
50.100.5	3,703	171	5	175	95.3	37	78.4	5
50.200.5	1,334	272	5	25	98.1	43	84.2	5
50.300.5	2,975	617	5	12	98.4	99	91.0	5
50.400.5	5,356	1,754	4	1	100.0	139	92.1	5
100.200.5	8,743	1,036	4	59	99.3	207	80.0	5
100.300.5	26	322	5	5	80.8	222	31.1	5
100.400.5	393	528	5	1	99.7	260	50.8	5
25.50.10	78,595	659	5	7,139	90.9	398	39.6	5
25.100.10	96,586	2,037	3	10,004	89.6	1,511	25.8	4
25.200.10	60,468	3,466	1	5,097	91.6	2,879	16.9	2
25.300.10	31,651	3,600	0	2,486	92.1	3,172	11.9	1
25.400.10	16,806	3,600	0	1,486	91.2	3,567	0.9	1
50.100.10	21,549	812	5	185	99.1	117	85.6	5
50.200.10	19,027	1,564	3	105	99.4	319	79.6	5
50.300.10	2,773	910	5	105	96.2	473	48.0	5
50.400.10	11,774	3,384	2	115	99.0	817	75.9	5
100.200.10	6,804	986	5	173	97.5	1,033	−4.8	4
100.300.10	150	590	5	2	98.7	595	−0.8	5
100.400.10	92	613	5	6	93.5	736	−20.1	5
25.50.20	206,308	3,150	2	16,025	92.2	3,600	−14.3	0
25.100.20	83,214	3,492	1	5,114	93.9	3,600	−3.1	0
25.200.20	19,508	3,600	0	1,372	93.0	3,600	0.0	0
25.300.20	9,143	3,600	0	625	93.2	3,600	0.0	0
25.400.20	3,677	3,600	0	367	90.0	3,600	0.0	0
50.100.20	27,569	1,962	3	1,783	93.5	3,504	−78.6	1
50.200.20	7,684	3,600	0	599	92.2	3,600	0.0	0
50.300.20	2,522	3,600	0	261	89.7	3,600	0.0	0
50.400.20	1,912	3,600	0	143	92.5	3,600	0.0	0
100.200.20	4,250	3,568	1	190	95.5	3,600	−0.9	0
100.300.20	1,070	3,327	2	66	93.8	3,062	8.0	2
100.400.20	559	2,880	2	13	97.7	2,241	22.2	3
Average	28,596	2,183	2.6	1,560	94.5	1,627	25.5	3.1
Total	1,029,471	78,585	94	56,172	94.5	58,566	25.5	113

Table 8 Average solution time and node, and # of instances solved for transportation with the MIP model

$\|I\|.\|J\|.T$	Default			Default + PLO cuts				
	Node	Time	#Sol.	Node	%Red.	Time	%Red.	#Sol.
25.50.5	146,664	1,381	4	25,343	82.7	356	74.2	5
25.100.5	145,207	1,710	3	2,686	98.2	131	92.3	5
25.200.5	68,845	2,642	2	5,873	91.5	890	66.3	4
25.300.5	44,531	3,300	2	630	98.6	324	90.2	5
25.400.5	39,032	3,600	0	991	97.5	837	76.8	5
50.100.5	5,656	252	5	1,172	79.3	167	33.7	5
50.200.5	660	161	5	80	87.9	59	63.4	5
50.300.5	1,115	318	5	56	95.0	104	67.3	5
50.400.5	1,626	704	5	7	99.6	78	88.9	5
100.200.5	1,107	413	5	99	91.1	175	57.6	5
100.300.5	9	61	5	2	77.8	76	−24.6	5
100.400.5	247	415	5	4	98.4	128	69.2	5
25.50.10	376,311	2,471	2	39,355	89.5	2,480	−0.4	2
25.100.10	507,188	3,600	0	20,001	96.1	3,600	0.0	0
25.200.10	47,534	3,600	0	5,507	88.4	3,600	0.0	0
25.300.10	29,901	3,600	0	2,577	91.4	3,600	0.0	0
25.400.10	17,788	3,600	0	1,424	92.0	3,600	0.0	0
50.100.10	3,714	396	5	308	91.7	218	44.9	5
50.200.10	7,061	1,078	4	209	97.0	497	53.9	5
50.300.10	2,804	1,259	5	390	86.1	1,469	−16.7	5
50.400.10	2,769	2,188	4	165	94.0	1,124	48.6	4
100.200.10	9	68	5	1	88.9	97	−42.6	5
100.300.10	3	96	5	0	100.0	112	−16.7	5
100.400.10	14	122	5	2	85.7	181	−48.4	5
25.50.20	186,226	3,600	0	14,066	92.4	3,600	0.0	0
25.100.20	27,271	3,600	0	4,196	84.6	3,600	0.0	0
25.200.20	8,113	3,600	0	1,379	83.0	3,600	0.0	0
25.300.20	2,739	3,600	0	591	78.4	3,600	0.0	0
25.400.20	2,745	3,600	0	273	90.1	3,600	0.0	0
50.100.20	9,962	3,207	1	2,096	79.0	3,600	−12.3	0
50.200.20	2,629	3,600	0	517	80.3	3,600	0.0	0
50.300.20	2,443	3,600	0	190	92.2	3,600	0.0	0
50.400.20	1,601	3,600	0	65	95.9	3,600	0.0	0
100.200.20	1,551	2,517	2	72	95.4	1,859	26.1	3
100.300.20	390	2,050	4	14	96.4	1,275	37.8	5
100.400.20	150	2,326	3	0	100.0	1,220	47.5	4
Average	47,100	2,109	2.5	3,621	92.3	1,685	20.1	3.0
Total	1,695,615	75,935	91	130,341	92.3	60,657	20.1	107

Table 9 Average solution time and node, and # of instances solved for transportation with the SOS2 model

$\|I\|.\|J\|.T$	Default			Default + PLO cuts				
	Node	Time	#Sol.	Node	%Red.	Time	%Red.	#Sol.
25.50.5	2,110,475	1,815	4	2,314	99.9	28	98.5	5
25.100.5	1,852,388	3,600	0	1,527	99.9	49	98.6	5
25.200.5	619,103	3,600	0	9,993	98.4	942	73.8	5
25.300.5	362,697	3,600	0	2,944	99.2	586	83.7	5
25.400.5	237,542	3,600	0	7,554	96.8	2,748	23.7	2
50.100.5	423,790	1,816	4	1,049	99.8	95	94.8	5
50.200.5	290,946	2,971	2	99	100.0	35	98.8	5
50.300.5	136,919	2,931	1	110	99.9	71	97.6	5
50.400.5	129,503	3,600	0	80	99.9	95	97.4	5
100.200.5	150,929	3,472	1	462	99.7	344	90.1	5
100.300.5	16,894	672	5	141	99.2	229	65.9	5
100.400.5	26,602	1,572	3	48	99.8	147	90.6	5
25.50.10	1,666,720	2,504	2	43,222	97.4	1,846	26.3	3
25.100.10	808,788	3,600	0	25,897	96.8	3,600	0.0	0
25.200.10	325,992	3,600	0	6,661	98.0	3,600	0.0	0
25.300.10	192,535	3,600	0	3,083	98.4	3,600	0.0	0
25.400.10	128,298	3,600	0	1,766	98.6	3,600	0.0	0
50.100.10	271,766	2,298	3	1,924	99.3	678	70.5	5
50.200.10	178,276	3,600	0	1,854	99.0	1,868	48.1	5
50.300.10	106,273	3,600	0	871	99.2	1,808	49.8	4
50.400.10	62,412	3,600	0	853	98.6	3,098	13.9	1
100.200.10	29,714	1,561	3	514	98.3	1,867	−19.6	3
100.300.10	10,105	760	4	177	98.2	1,001	−31.7	5
100.400.10	7,385	752	4	71	99.0	886	−17.8	5
25.50.20	876,301	3,600	0	17,196	98.0	3,600	0.0	0
25.100.20	371,069	3,600	0	5,327	98.6	3,600	0.0	0
25.200.20	152,110	3,600	0	1,576	99.0	3,600	0.0	0
25.300.20	83,838	3,600	0	769	99.1	3,600	0.0	0
25.400.20	52,835	3,600	0	448	99.2	3,600	0.0	0
50.100.20	176,752	3,600	0	1,941	98.9	3,600	0.0	0
50.200.20	53,197	3,600	0	649	98.8	3,600	0.0	0
50.300.20	34,203	3,600	0	325	99.0	3,600	0.0	0
50.400.20	21,153	3,600	0	197	99.1	3,600	0.0	0
100.200.20	25,147	3,600	0	212	99.2	3,600	0.0	0
100.300.20	14,627	3,600	0	106	99.3	3,230	10.3	1
100.400.20	8,044	2,898	1	69	99.1	3,327	−14.8	1
Average	333,759	3,023	1.0	3,945	98.8	2,094	30.7	2.5
Total	12,015,328	108,822	37	142,029	98.8	75,378	30.7	90

was 25 % with the LOG model; 20 % with the MIP model; and 30 % with the SOS2 approach.

We note, however, that the overall performance of the cuts was not uniform across different values of T. For the smaller values of T the PLO cuts were extremely efficient. That was not the case for the larger values of T. That is the reason the overall results for transportation are not as great as for transshipment. For the rest of this section, we refer to Tables 7, 8 and 9.

For the 60 instances with $T = 5$, default with the PLO cuts solved all (100 %) instances to proven optimality as opposed to 44 (73 %) without the PLO cuts, for LOG; 59 (98 %), as opposed to 46 (77 %), for MIP; 57 (95 %), as opposed to 20 (33 %), for SOS2. The average reduction from default in computational time by adding the PLO cuts was 89 %, with the LOG model; 78 %, with the MIP model; and 84 %, with the SOS2 approach. The average reduction from default in the number of enumeration nodes by adding PLO cuts was 99 % with the LOG model, 92 % with the MIP model, and 99.6 % with the SOS2 approach.

For $T = 10$, default with the PLO cuts solved 47 (78 %) instances to proven optimality, as opposed to 39 (65 %) without the PLO cuts, for LOG; 36 (60 %), as opposed to 35 (58 %), for MIP; 31 (52 %), as opposed to 16 (27 %), for SOS2. The average reduction from default in computational time by adding the PLO cuts was 30 %, with the LOG model; 7 %, with the MIP model; and 17 %, with the SOS2 approach. The average reduction from default in the number of enumeration nodes by adding the PLO cuts was 92 % with the LOG model, 93 % with the MIP model, and 98 % with the SOS2 approach.

For $T = 20$, default with the PLO cuts solved 6 (10 %) instances to proven optimality, as opposed to 11 (18 %) without the PLO cuts, for LOG; 12 (20 %), as opposed to 10 (17 %), for MIP; 2 (3 %), as opposed to 1 (2 %), for SOS2. The average reduction from default in computational time by adding the PLO cuts was -3 %, with the LOG model; 5 %, with the MIP model; and -0.1 %, with the SOS2 approach. The average reduction from default in the number of enumeration nodes by adding PLO cuts was 93 % with the LOG model; 91 % with the MIP model; and 98 % with the SOS2 approach. In Sect. 7 we will analyze further the dependence on T of the performance of the PLO cuts for transportation.

We now discuss the effect of formulation on the computational results, first without the addition of the PLO cuts. Overall, MIP and LOG were the most efficient approaches, followed by SOS2. In computational time, MIP reduced by 3 % the computational time of LOG, and by 30 % for SOS2. In number of nodes, MIP reduced by -65 % the number of nodes for LOG, and by 86 % for SOS2. The MIP formulation solved to proven optimality 91 (51 %) instances, as opposed to 94 (52 %) for LOG, and 37 (21 %) for SOS2. This rank remains the same for all values of T.

With PLO cuts added, overall LOG was the most efficient, followed by MIP, followed by SOS2. In computational time, LOG reduced by 3 % the computational time of MIP, and by 22 % for SOS2. In number of nodes, LOG reduced by 57 % the number of nodes for MIP, and by 60 % for SOS2. The LOG formulation solved to proven optimality 113 instances, as opposed to 107 for MIP, and 90 for SOS2. This rank remains the same for $T \in \{5, 10\}$. For $T = 20$, MIP is the most efficient, followed by LOG, followed by SOS2.

7 Large special ordered sets

As pointed out in Sect. 6, the computational results for transportation indicated a degradation in performance of the PLO cuts for larger values of T. In this section we elaborate on this issue. We consider two sets of data, both transportation: the ones we generated and the data used in Vielma and Nemhauser [30]. Here, as in Sect. 6, the maximum computational time allowed was 3,600 s of CPU.

In Tables 10, 11, and 12 we give the number of instances solved, the average computational time, and the average number of enumeration nodes with and without PLO cuts for our data type. Each entry in the tables gives an average over five instances. The first two rows of the tables refer to small values of T (specifically, $T \in \{5, 10\}$), and the other rows to large values of T (specifically, $T \in \{20, 30, 40, 80, 120\}$). Table 10 refers to the LOG formulation, Table 11 to the MIP formulation, and Table 12 to the SOS2 approach.

Table 10 Average solution time and # of nodes, and # of instances solved for transportation with the LOG model

$\|I\|.\|J\|.T$	Default			Default + PLO cuts				
	Node	Time	#Sol.	Node	%Red.	Time	%Red.	#Sol.
10.20.5	14,542	18	5	951	93.5	3	83.3	5
10.20.10	22,418	46	5	1,873	91.6	12	73.9	5
10.20.20	35,044	48	5	8,664	75.3	121	−152.1	5
10.20.30	85,893	173	5	32,563	62.1	846	−389.0	5
10.20.40	165,599	440	5	39,757	76.0	2,074	−371.4	3
10.20.80	278,037	1,924	3	13,624	95.1	3,600	−87.2	0
10.20.120	282,304	3,097	2	4,059	98.6	3,600	−16.4	0
Average	126,262	821	4.3	14,499	88.5	1,466	−78.6	3.3
Total	883,837	5,746	30	101,491	88.5	10,262	−78.6	23

Table 11 Average solution time and # of nodes, and # of instances solved for transportation with the MIP model

$\|I\|.\|J\|.T$	Default			Default + PLO cuts				
	Node	Time	#Sol.	Node	%Red.	Time	%Red.	#Sol.
10.20.5	81,933	129	5	1,907	97.7	7	94.6	5
10.20.10	909,081	1,463	3	258,833	71.5	1,452	0.8	3
10.20.20	1,702,314	2,326	2	208,367	87.8	3,177	−36.6	1
10.20.30	2,064,999	3,600	0	103,104	95.0	3,600	0.0	0
10.20.40	1,070,054	3,600	0	52,286	95.1	3,600	0.0	0
10.20.80	337,078	3,600	0	8,943	97.3	3,600	0.0	0
10.20.120	96,223	3,600	0	2,088	97.8	3,600	0.0	0
Average	894,526	2,617	1.4	90,790	89.9	2,719	−3.9	1.3
Total	6,261,682	18,318	10	635,528	89.9	19,036	−3.9	9

Table 12 Average solution time and # of nodes, and # of instances solved for transportation with the SOS2 approach

$\|I\|.\|J\|.T$	Default			Default + PLO cuts				
	Node	Time	#Sol.	Node	%Red.	Time	%Red.	#Sol.
10.20.5	119,886	23	5	1,754	98.5	2	91.3	5
10.20.10	237,369	70	5	39,827	83.2	113	−61.4	5
10.20.20	861,870	433	5	87,644	89.8	855	−97.5	5
10.20.30	2,029,894	1,591	3	109,731	94.6	2,707	−70.1	2
10.20.40	3,442,864	3,600	0	66,138	98.1	3,600	0.0	0
10.20.80	1,618,481	3,600	0	12,243	99.2	3,600	0.0	0
10.20.120	960,401	3,600	0	4,494	99.5	3,600	0.0	0
Average	1,324,395	1,845	2.6	45,976	96.5	2,068	−12.1	2.4
Total	9,270,765	12,917	18	321,831	96.5	14,477	−12.1	17

Of the 10 instances with small values of T, regardless of whether the PLO cuts were used or not, all (100 %) were solved to proven optimality for the LOG formulation, 8 (80 %) for the MIP formulation, and all (100%) for the SOS2 approach. In average, the PLO cuts reduced the computational time of default by 77 % for the LOG formulation, 8 % for the MIP formulation, and −24 % for the SOS2 approach. The number of nodes was reduced by 92 % in the LOG formulation, 74 % in the MIP formulation, and 88 % in the SOS2 approach.

On the other hand, of the 25 instances with large values of T, default solved 20 (80 %), as opposed to 13 (52 %) with the PLO cuts, for the LOG formulation; 2 (8 %), as opposed to 1 (4 %), for the MIP formulation; and 8 (32 %), as opposed to 7 (28 %), for the SOS2 approach. In average, the use of PLO cuts increased the computational time over default by 80 % for the LOG formulation, 5 % for the MIP formulation, and 12 % for the SOS2 approach. In number of nodes, the use of PLO cuts reduced the amount generated by default by 88 % for the LOG formulation; 93 % for the MIP formulation; and 97 % for the SOS2 approach. So the performance of the PLO cuts was considerably inferior to default for the large values of T.

We note that the number of PLO cuts generated for the instances with large value of T was extremely large, and even more importantly, the percentage of PLO cuts added to the formulation was considerably smaller than for the other tests of this section and of the previous ones. In the particular case of the tests with small T of this section, 17 % of the PLO cuts generated were used for the LOG formulation; 37 % for the MIP formulation; and 0.1 % for the SOS2 approach. For the tests with large T, 0.2 % were used for the LOG formulation; 4 % for the MIP formulation; and 0.001 % for the SOS2 approach. So one possible explanation for the poor performance of the PLO cuts in our tests with large T was, in part, the time wasted separating a large number of cuts, which, in the end, did not pass GUROBI's user cut filter. With this in mind, we tried several alternatives for reducing the number of PLO cuts generated. For example, we tried cut-and-branch, or limiting the number of cuts by a fixed amount, or increasing the violation required for separation. In all our attempts, the performance

Table 13 # of Vielma–Nemhauser instances solved

Solver	LOG			MIP			SOS2		
	Def.	B&B	PLO	Def.	B&B	PLO	Def.	B&B	PLO
CPX	100	100	100	91	92	100	96	97	100
GRB	100	100	100	100	100	100	100	100	100

Table 14 Average solution time for the Vielma–Nemhauser instances

Solver	LOG			MIP			SOS2		
	Def.	B&B	PLO	Def.	B&B	PLO	Def.	B&B	PLO
CPX	6	8	4	684	545	21	266	244	10
GRB	6	4	4	56	56	9	15	15	219

Table 15 Average # of nodes for the Vielma–Nemhauser instances

Solver	LOG			MIP			SOS2		
	Def.	B&B	PLO	Def.	B&B	PLO	Def.	B&B	PLO
CPX	5,063	4,446	57	254,910	303,039	258	709,249	710,774	270
GRB	4,403	4,215	71	41,520	49,369	155	37,800	41,254	22,107

of the PLO cuts continued poor. So, it may just be that our PLO cuts, for the type of data we used, are not efficient when T is large. (Note that the results we present are for branch-and-cut, i.e. we performed separation in all nodes of the enumeration tree).

Now we analyze the results obtained from the data of Vielma and Nemhauser [30]. We tested 100 of their instances, specifically, their largest, with $|I| = |J| = 10$, and $T = 32$. Here we give the results of both CPLEX (denoted as CPX) and GUROBI (denoted as GRB). In Table 13 we give the number of instances solved to proven optimality by default, branch-and-bound, and default with PLO cuts (columns labeled PLO). In Tables 14 and 15 we give the average computational time and number of enumeration nodes, respectively, again for default, branch-and-bound, and default with PLO cuts (columns labeled PLO). First, we analyze the results without the PLO cuts.

With CPLEX, LOG was more efficient than SOS2, which was more efficient than MIP. LOG default and branch-and-bound solved all (100 %) instances to proven optimality, while SOS2 solved 96 (96 %) in default and 97 (97 %) in branch-and-bound, and MIP solved 91 (91 %) in default and 92 (92 %) in branch-and-bound. LOG reduced the computational time of SOS2 by 98 % in default and by 97 % in branch-and-bound; it reduced the computational time of MIP by 99 % in default and by 98 % in branch-and-bound. LOG reduced the number of nodes of SOS2 by 99 % in default and branch-and-bound; it reduced the number of nodes of MIP by 98 % in default and branch-and-bound. These results are in agreement with the results of [30]. We note that branch-and-bound was slightly better than default for LOG, SOS2, and MIP. Therefore, CPLEX's facilities, such as MIP cutting planes, did not play a role.

With GUROBI, LOG was slightly more efficient than SOS2, which was more efficient than MIP, but the difference now was much less significant. All of them solved all instances to proven optimality (100 %) in both default and branch-and-bound. LOG reduced the computational time of SOS2 by 60 % in default and by 73 % in branch-and-bound. However, the difference is of just a few seconds. As for MIP, LOG reduced its computational time by 90 % in default and by 93 % in branch-and-bound, but again the absolute difference is considerably smaller. The small difference in absolute value is in disagreement with [30]. Because branch-and-bound was better than default for CPLEX and equivalent for GUROBI, and branch-and-bound GUROBI was considerably more efficient than branch-and-bound CPLEX, we raise the possibility that, in the case of the data of [30], the difference in performance of LOG and SOS2 may be due to a better implementation of variable dichotomy branching than SOS2 branching. LOG reduced the number of nodes of SOS2 by 88 % in default and by 90 % in branch-and-bound; it reduced the number of nodes of MIP by 89 % in default and by 92 % in branch-and-bound.

With the PLO cuts, all instances were solved to proven optimality (100 %). For CPLEX, the PLO cuts reduced the computational time in all cases, and the difference in performance for LOG, MIP, and SOS2 became of just a few seconds. The reduction was particularly significant for SOS2 and MIP. In the first case, the PLO cuts reduced the computational time of default by 96 % and of branch-and-bound by 95 %. In the second case, they reduced the computational time of default and branch-and-bound by 97 %. For GUROBI, the PLO cuts reduced the computational time for MIP, and their difference in performance for LOG and MIP again became of just a few seconds; however, the PLO cuts worsened the performance of SOS2 considerably. So, for the data of [30], the PLO cuts were efficient even with the value of T being large, except in the case of the SOS2 models running GUROBI. For the data of [30], the percentage of PLO cuts that passed GUROBI's user cut filter was much greater, 23 % for LOG, 0.01 % for SOS2, and 26 % for MIP.

8 Performance of SC-PLO cuts for transportation with semi-continuous constraints

We now give the results of our computational testing of the SC-PLO cuts on transportation with semi-continuous constraints. In all tests we used only the LOG formulation and GUROBI in default setting. Table 16 gives the computational time and number of enumeration nodes for default, default with PLO cuts, and default with SC-PLO cuts. The columns "%Red." for default with PLO cuts refer to number of nodes and computational time reduction over default. However, "%Red." for default with SC-PLO cuts refer to number of nodes and computational time reduction over default with PLO cuts. Unlike in the other sections, each entry of Table 16 refers to a single instance. Because these instances turned out to be considerably more difficult than the ones without semi-continuous constraints, we allowed for a maximum computational time of 7,200 s of CPU.

Of the 25 instances tested, default solved to proven optimality 15 (60 %) instances, as opposed to 17 (68 %) with PLO cuts and 18 (72 %) with SC-PLO cuts. The average

Table 16 Solution time and # of nodes for transportation with semi-continuous constraints

$\|I\|.\|J\|.T$	Default		Def. + PLO cuts				Def. + SC-PLO cuts			
	Node	Time	Node	%Red.	Time	%Red.	Node	%Red.	Time	%Red.
5.20.10	26,720	13	1,360	94.9	5	61.5	1,113	18.2	5	0.0
5.20.10	131,228	108	27,992	78.7	64	40.7	6,741	75.9	24	62.5
5.20.10	50,160	27	3,542	92.9	8	70.4	2,077	41.4	8	0.0
5.20.10	42,766	26	1,555	96.4	5	80.8	927	40.4	3	40.0
5.20.10	145,656	134	2,558	98.2	9	93.3	1,933	24.4	9	0.0
7.14.10	67,945	32	5,543	91.8	16	50.0	2,304	58.4	8	50.0
7.14.10	254,140	171	38,223	85.0	83	51.5	12,480	67.3	45	45.8
7.14.10	572,431	386	37,593	93.4	87	77.5	19,276	48.7	54	37.9
7.14.10	204,731	183	79,948	60.9	185	−1.1	40,034	49.9	120	35.1
7.14.10	32,341	19	8,788	72.8	24	−26.3	11,878	−35.2	31	−29.2
8.16.10	2,814,189	2,442	1,330,216	52.7	3,788	−55.1	300,325	77.4	1,111	70.7
8.16.10	156,997	114	107,716	31.4	321	−181.6	14,586	86.5	65	79.8
8.16.10	3,209,494	2,518	725,697	77.4	1,966	21.9	653,064	10.0	2,280	−16.0
8.16.10	613,659	368	53,172	91.3	157	57.3	25,906	51.3	100	36.3
8.16.10	2,051,311	1,468	445,506	78.3	1,197	18.5	154,669	65.3	551	54.0
10.20.5	2,828,338	7,200	2,001,681	29.2	7,200	0.0	1,728,372	13.7	7,200	0.0
10.20.5	6,315,328	7,200	161,549	97.4	587	91.8	63,803	60.5	244	58.4
10.20.5	4,600,864	7,200	2,392,827	48.0	7,200	0.0	2,316,737	3.2	7,200	0.0
10.20.5	2,744,003	7,200	1,971,728	28.1	7,200	0.0	1,969,959	0.1	7,200	0.0
10.20.5	3,229,417	7,200	1,380,925	57.2	7,200	0.0	1,315,588	4.7	7,200	0.0
10.20.10	2,221,766	7,200	1,050,939	52.7	7,200	0.0	989,279	5.9	6,895	4.2
10.20.10	3,350,263	7,200	762,973	77.2	7,200	0.0	740,642	2.9	7,200	0.0
10.20.10	2,607,928	7,200	798,658	69.4	7,200	0.0	772,746	3.2	7,200	0.0

Table 16 continued

$\|I\|.\|J\|.T$	Default		Def. + PLO cuts				Def. + SC-PLO cuts			
	Node	Time	Node	%Red.	Time	%Red.	Node	%Red.	Time	%Red.
10.20.10	2,904,149	7,200	386,021	86.7	2,236	68.9	128,515	66.7	863	61.4
10.20.10	3,003,894	7,200	1,153,694	61.6	7,200	0.0	1,065,999	7.6	7,200	0.0
Average	1,767,189	3,200	597,216	66.2	2,734	14.6	493,558	17.4	2,513	8.1
Total	44,179,718	80,009	14,930,404	66.2	68,338	14.6	12,338,953	17.4	62,816	8.1

reduction in computational time over default for the 17 instances that default with PLO cuts solved to proven optimality was of 52 %. Of these 17 instances, default with PLO cuts was faster for 13 instances (in almost all cases by more than 50 %) and slower for 4 instances. The average reduction in number of nodes was of 83 % (cf. Table 16).

The average reduction in computational time over default with PLO cuts for the 18 instances that default with SC-PLO cuts solved to proven optimality was of 31 %. Of these 18 instances, default with SC-PLO cuts was faster for 13 instances (in most cases by more than 40 %) and slower for 2 instances. The average reduction in number of nodes was of 46 % (cf. Table 16).

So, PLO cuts were considerably useful for solving the instances, and SC-PLO cuts were even more useful. Only 0.04 and 0.02 % of the PLO and SC-PLO cuts, respectively, separated were used. It is then possible that by separating less PLO and SC-PLO cuts, their performance on these instances may be improved.

9 Summary of conclusions and further research

Our results give a strong indication that in order to solve medium to large-scale difficult instances of PLO it is essential to use cuts that consider the SOS2 constraints. Such cuts may be generic (as far as PLO is concerned), e.g. the cuts of Zhao and de Farias [32]. However, without them, solving PLO to proven optimality is a formidable task. So we suggest as a direction of further research investigating new cuts for PLO.

Our results indicate that the cuts of Zhao and de Farias [32] are consistently efficient when T is small. For larger values of T, their performance worsened with increasing T for our data. However, for the data of Vielma and Nemhauser [30], in which T is within our bracket of larger values, they became, in most cases, efficient again. So another topic for further research is when the cuts of Zhao and de Farias [32] are efficient for large T. A more general question is "Which types of PLO cuts are efficient when T is large?"

As already expected from previous studies, e.g. [9,10,19], SOS2 performed better than MIP in our tests. However, LOG performed better than SOS2. The reason for that needs to be clarified. In our experiments LOG performed better even for small T, differently from the results of Vielma and Nemhauser [30], in which LOG performed better for larger T but not for smaller T. One possibility is that the superiority of LOG is simply the result of CPLEX's and GUROBI's better implementation of 0–1 variable dichotomy branching over SOS2 branching. Another possibility is that LOG possesses an inherent property that makes it more efficient than SOS2, at least for some problems. In this case, it would be interesting determining such property and how to take advantage of it. For example, they could possibly be used to improve other branching schemes, even SOS2.

Still on the issue of formulation, another possibility that should be tested computationally is incremental cost [21]. Despite evidences that it is computationally equivalent to the "usual" MIP formulation, see for example [18], there may be situations in which it performs better. We believe that discovering and understanding such situations is an interesting topic for further investigation.

The results on PLO with semi-continuous constraints showed the potential of cutting planes that consider two or more combinatorial structures at the same time (in our case, SOS2 and semi-continuous). Such studies are scarce (actually, we are not aware of any, at least in a setting as general as ours). We believe that this is an interesting topic of further investigation. Specifically, we suggest investigating the inequality description of these sets (e.g. defined by a knapsack, SOS2, and semi-continuous constraints), and the computational possibilities of their valid inequalities.

Finally, we would like to point out two questions on computational PLO that our results indicated to be interesting:

– How should PLO and SC-PLO cuts be filtered?
– When are MIP cuts for the LOG and MIP models efficient?

We suggest their study as topics for further research.

Acknowledgments This research was partially supported by the Office of Naval Research and the National Science Foundation through grants N000140910332 and CMMI-0620755, respectively. Their support is gratefully acknowledged. We are grateful to George Nemhauser and Juan-Pablo Vielma for making available to us the instances of their paper [30]. We are also grateful to Zhonghao Gu and Ed Rothberg for enlightening discussions. Finally, we are grateful to the anonymous referees and the editors, for several valuable suggestions.

References

1. Beale, E.M.L.: Two transportation problems. In: Kreweras, G., Morlat, G. (eds.) Proceedings of the Third International Conference on Operational Research, Dunod, pp. 780–788 (1963)
2. Beale, E.M.L., Tomlin, J.A.: Special facilities in a general mathematical programming system for nonconvex problems using ordered sets of variables. In: Lawrence, J. (ed.) Proceedings of the Fifth International Conference on Operations Research, Tavistock Publications, pp. 447–454 (1970)
3. Bienstock, D.: Computational study of a family of mixed-integer quadratic programming problems. Math. Program. **74**, 121–140 (1996)
4. Crowder, H., Johnson, E.L., Padberg, M.: Solving large-scale zero-one linear programming problems. Oper. Res. **31**, 803–834 (1983)
5. Croxton, K.L., Gendron, B., Magnanti, T.L.: Models and methods for merge-in-transit operations. Transp. Sci. **37**, 1–22 (2003)
6. Croxton, K.L., Gendron, B., Magnanti, T.L.: Variable Disaggregation in Network Flow Problems with Piecewise Linear Costs. Operations Research Center, Massachusetts Institute of Technology, Cambridge (2003)
7. Dantzig, G.B.: On the significance of solving linear programming problems with some integer variables. Econometrica **28**, 30–44 (1960)
8. de Farias, I.R. Jr.: Semi-continuous cuts for mixed-integer programming. In: Bienstock, D., Nemhauser, G.L. (eds.) Integer Programming and Combinatorial Optimization (IPCO). Lecture Notes in Computer Science, vol. 3064, pp. 163–177, Springer (2004)
9. de Farias, I.R. Jr., Johnson, E.L., Nemhauser, G.L.: A generalized assignment problem with special ordered sets: a polyhedral approach. Math. Program. **89**, 187–203 (2000)
10. de Farias, I.R Jr., Johnson, E.L., Nemhauser, G.L.: Branch-and-cut for combinatorial optimization problems without auxiliary binary variables. Knowl. Eng. Rev. **16**, 25–39 (2001)
11. de Farias, I.R. Jr., Kozyreff, E., Gupta, R., Zhao, M.: Branch-and-Cut for Separable Piecewise Linear Optimization and Intersection with Semi-Continuous Constraints. Texas Tech University, USA (2011)
12. de Farias, I.R. Jr., Nemhauser, G.L.: A polyhedral study of the cardinality constrained knapsack problem. Math. Program. **96**, 439–467 (2003)
13. de Farias, I.R. Jr., Zhao, M.: A polyhedral study of the semi-continuous knapsack problem. Math. Programm. (2011, submitted)

14. de Farias, I.R. Jr., Zhao, M., Zhao, H.: A special ordered set approach for optimizing a discontinuous separable piecewise linear function. Oper. Res. Lett. **36**, 234–238 (2008)
15. Fourer, R., Gay, D.M., Kerninghan, B.W.: AMPL: A Modeling Language for Mathematical Programming. The Scientific Press, USA (1993)
16. Gu, Z.: Personal communication
17. http://www.hpcc.ttu.edu/index.php
18. Keha, A.B., de Farias, I.R. Jr., Nemhauser, G.L.: Models for representing piecewise linear cost functions. Oper. Res. Lett. **32**, 44–48 (2004)
19. Keha, A.B., de Farias, I.R. Jr., Nemhauser, G.L.: A branch-and-cut algorithm without binary variables for nonconvex piecewise linear optimization. Oper. Res. **54**, 847–858 (2006)
20. Konno, H., Wijayanayake, A.: Portfolio optimization problem under concave transaction costs and minimal transaction unit constraints. Math. Program. **89**, 233–250 (2001)
21. Markowitz, H.M., Manne, A.S.: On the solution of discrete programming problems. Econometrica **25**, 84–110 (1957)
22. Martin, A., Möller, M., Moritz, S.: Mixed-integer models for the stationary case of gas network optimization. Math. Program. **105**, 563–582 (2006)
23. Nocedal, J., Wright, S.J.: Numerical Optimization. Springer, Berlin (1999)
24. Perold, A.F.: Large-scale portfolio optimization. Manag. Sci. **30**, 1143–1160 (1984)
25. Sioshansi, R., O'Neill, R.O., Oren, S.S.: Economic consequences of alternative solution methods for centralized unit commitment in day-ahead electricity markets. IEEE Trans. Power Syst. **23**, 344–352 (2008)
26. Takriti, S., Birge, J.R., Long, E.: A stochastic model for the unit commitment problem. IEEE Trans. Power Syst. **11**, 1497–1508 (1996)
27. Takriti, S., Krasenbrink, B., Wu, L.S.Y.: Incorporating fuel constraints and electricity spot prices into the stochastic unit commitment problem. Oper. Res. **48**, 268–280 (2000)
28. Tomlin, J.A.: Special ordered sets and an application to gas supply operations planning. Math. Program. **42**, 69–84 (1988)
29. Vielma, J.P., Ahmed, S., Nemhauser, G.L.: Mixed-integer models for nonseparable piecewise linear optimization: unifying framework and extensions. Oper. Res. **58**, 303–315 (2010)
30. Vielma, J.P., Nemhauser, G.L.: Modeling disjunctive constraints with a logarithmic number of binary variables and constraints. Math. Program. **128**, 49–72 (2011)
31. Zhang, M., Guan, Y.: Two-Stage Robust Unit Commitment Problem. University of Florida, USA (2009)
32. Zhao, M., de Farias, I.R. Jr.: The Piecewise Linear Optimization Polytope: New Inequalities and Intersection with Semi-Continuous Constraints. Math. Program. (2012, in press)