Nonconvex, Lower Semicontinuous Piecewise Linear Optimization ⋆

Juan Pablo Vielma [a,*] Ahmet B. Keha [b] George L. Nemhauser [a]

[a] *School of Industrial and Systems Engineering, Georgia Institute of Technology, 765 Ferst Drive, Atlanta, GA 30332-0205, USA*

[b] *Department of Industrial Engineering, Arizona State University, PO Box 875906, Tempe AZ 85287-5906, USA*

Abstract

A branch-and-cut algorithm for solving linear problems with continuous separable piecewise linear cost functions was developed in 2005 by Keha et. al. This algorithm is based on valid inequalities for an SOS2 based formulation of the problem. In this paper we study the extension of the algorithm to the case where the cost function is only lower semicontinuous. We extend the SOS2 based formulation to the lower semicontinuous case and show how the inequalities introduced by Keha et. al. can also be used for this new formulation. We also introduce a simple generalization of one of the inequalities introduced by Keha et. al. Furthermore, we study the discontinuities caused by fixed charge jumps and introduce two new valid inequalities by extending classical results for fixed charge linear problems. Finally, we report computational results showing how the addition of the developed inequalities can significantly improve the performance of CPLEX when solving these kinds of problems.

Key words: Piecewise Linear Optimization, Discontinuous Piecewise Linear Functions, Branch-and-Cut

⋆ This research has been supported by the National Science Foundation awards DMI-0100020 and DMI-0121495.

* Corresponding author.

Email addresses: jvielma@isye.gatech.edu (Juan Pablo Vielma), Ahmet.Keha@asu.edu (Ahmet B. Keha), gnemhaus@isye.gatech.edu (George L. Nemhauser).

1 Introduction

We study the *nonconvex separable lower semicontinuous piecewise linear optimization problem* given by

$$
\begin{aligned}
&\min \sum_{j \in N} f_j(x_j) \\
&s.t. \\
&\qquad \sum_{j \in N} g_{ij} x_j \le b_i \quad \forall\, i \in \{1, \dots, m\} \\
&\qquad\quad 0 \le x_j \le u_j \quad \forall\, j \in N
\end{aligned}
$$

where $N = \{1, \dots, n\}$, $g_{ij} \ge 0$ for all i, j and $f_j(x_j)$ is a lower semicontinuous nonconvex piecewise linear function.

This problem is NP-hard and has several applications [10] including network flow problems with nonconvex objectives [1,2] and with fixed charges [12,13,15,16,19].

Our goal is to extend the results obtained in [10] for the case where $f_j(x_j)$ is continuous to the semicontinuous case. These results include the development of a branch-and-cut algorithm without binary variables for the *nonconvex separable continuous piecewise linear optimization problem* by deriving valid inequalities for an *SOS2* based formulation of the problem.

In Section 2 we describe this *SOS2* model and the valid inequalities developed in [10]. We also derive a simple generalization of one of these inequalities. In Section 3 we extend the *SOS2* formulation to the semicontinuous case, study its relationship to a binary formulation suggested in [3] and [14] and show how to use cuts from the continuous case. Section 4 is devoted to the study of discontinuities caused by fixed charges. In this section two new valid inequalities are developed by extending classical results for fixed charge linear problems. Finally computational results are presented in Section 5.

2 *SOS2* model for the continuous case

In this section we present the classical *SOS2* model for the continuous case and we summarize the polyhedral results presented in [10]. We begin by reviewing the definition of the *SOS2* condition.

An ordered set of variables is said to satisfy SOS2 *if no more than two variables are positive and if two variables are positive, then they must be adjacent in the order.*

Now, suppose that for each $j \in N$, $f_j(x_j)$ is a continuous piecewise linear function which is linear in segments $[d_j^k, d_j^{k+1}]$ for all $k \in \{0, \ldots, T-1\}$, where $d_j^0 = 0$ and $d_j^T = u_j$. Then, using $x_j = \sum_{k=0}^{T} d_j^k \lambda_j^k$ with $\lambda_j^k \geq 0$ and $\sum_{k=0}^{T} \lambda_j^k = 1$ and imposing the *SOS2* condition to get the correct value of $f_j(x_j)$ gives the model

$$\min \quad \sum_{j \in N} \sum_{k=0}^{T} f_j(d_j^k) \lambda_j^k$$

s.t.

$$\sum_{j \in N} \sum_{k=0}^{T} a_{ij}^k \lambda_j^k \leq b_i \qquad \forall\, i \in \{1, \ldots, m\} \tag{1}$$

$$\sum_{k=0}^{T} \lambda_j^k = 1 \qquad \forall\, j \in N \tag{2}$$

$$\lambda_j^k \geq 0 \qquad \forall\, j \in N\, \forall\, k \in \{0, \ldots, T\} \tag{3}$$

$$(\lambda_j^k)_{k=0}^{T} \text{ is } SOS2 \qquad \forall j \in N \tag{4}$$

where $a_{ij}^k = g_{ij} d_j^k$.

The one row relaxation of this model where (1) is replaced by

$$\sum_{j \in N} \sum_{k=0}^{T} a_j^k \lambda_j^k \leq b \tag{5}$$

is the basis of our polyhedral results. Let $S = \{\lambda = (\lambda_j^k)_{k=0,\, j \in N}^{T} \in \mathbb{R}^{n(T+1)} :$ λ satisfies (2)–(5)$\}$ be the set of feasible solutions to this model and let $LS =$ $\{\lambda \in \mathbb{R}^{n(T+1)} : \lambda$ satisfies (2)–(3),(5)$\}$ be the set of feasible solutions to its LP relaxation.

Several valid inequalities for $P = \text{conv}(S)$ are presented in [10]. In the following section we review these valid inequalities and describe the separation procedure for a given $\lambda \in LS \setminus P$. We also develop a small extension of one of these inequalities.

2.1 Lifted Convexity Constraints

Lifted convexity constraints are obtained by lifting a natural relaxation of (2). For $j \in N$, let $I = \{i \in N \setminus \{j\} \,:\, b - a_j^1 \leq a_i^T\}$ and for $i \in I$ let $k_i = \min\{k \,:\, b - a_j^1 \leq a_i^k\}$. Then, for $i \in I$

$$\sum_{k=1}^{T} \lambda_j^k + \sum_{k=k_i-1}^{T} \alpha_i^k \lambda_i^k \leq 1 \tag{6}$$

is a valid inequality, where

$$(\alpha_i^{k_i-1}, \alpha_i^{k_i}) = \begin{cases} (1 - \frac{(b-a_i^{k_i-1})}{a_j^1}, 1 - \frac{(b-a_i^{k_i})}{a_j^1}) & \text{if } b - a_j^1 < a_i^{k_i} \\ (0,0) & \text{if } b - a_j^1 = a_i^{k_i} \end{cases} \tag{7}$$

$$\alpha_i^k = 1 - \frac{(b - a_i^k)}{a_j^1} \qquad k > k_i. \tag{8}$$

Inequality (6) gives two possibilities for separation. Let $\tilde{\lambda} \in LS \setminus P$ be such that $\tilde{\lambda}_i$ violates *SOS2* and let $\tilde{k}_i = \max\{k \,:\, \tilde{\lambda}_i^k > 0\}$. Then, if $b - a_j^1 \leq a_i^{\tilde{k}_i-1}$ and $\sum_{k=1}^{T} \tilde{\lambda}_j^k = 1$

$$\sum_{k=1}^{T} \lambda_j^k + \sum_{k=\tilde{k}_i}^{T} \alpha_i^k \lambda_i^k \leq 1 \tag{9}$$

cuts off $\tilde{\lambda}$, where all α_i^k are positive and given by (8). We denote this cut as a *Lifted Convexity Cut type I*.

On the other hand, if $a_i^{\tilde{k}_i-1} < b - a_j^1 < a_i^{\tilde{k}_i}$ and $\sum_{k=1}^{T} \tilde{\lambda}_j^k = 1$ then

$$\sum_{k=1}^{T} \lambda_j^k + \alpha_i^{\tilde{k}_i-1} \lambda_i^{\tilde{k}_i-1} + \alpha_i^{\tilde{k}_i} \lambda_i^{\tilde{k}_i} \leq 1 \tag{10}$$

where $\alpha_i^{\tilde{k}_i-1}$ and $\alpha_i^{\tilde{k}_i}$ are given by (7) may cut off $\tilde{\lambda}$. In particular, it will cut the infeasible point if, for example, $\tilde{\lambda}_i^{\tilde{k}_i-1} = 0$. We denote this cut as a *Lifted Convexity Cut type II*.

2.2 Lifted Cover Constraints

Lifted cover constraints extend the concept of a cover to continuous variables with *SOS2* constraints. Consider a set $C \subseteq N$ and $k_j \in \{2, \ldots, T\}$ for $j \in C$

such that $\sum_{j\in C} a_j^{k_j} = b + \Delta$ for $\Delta > 0$. Then

$$\sum_{j\in C}(\alpha_j \lambda_j^{k_j-1} + \sum_{k=k_j}^{T} \lambda_j^k) \leq |C| - 1, \tag{11}$$

is a valid inequality, where $\alpha_j = \min\{0, (\Delta - a_j^{k_j} + a_j^{k_j-1})/\Delta\}$. More generally, requirement $2 \leq k_j$ can be relaxed to

$$2 \leq k_j \text{ or } (1 \leq k_j \wedge \Delta \geq a_j^1).$$

Separation can be done as follows. Let $\tilde{\lambda} \in LS \setminus P$ be such that $\tilde{\lambda}_i$ violates *SOS2*. Let $L = \{l > 1 : \tilde{\lambda}_i^l > 0\}$ and for each $j \neq i$ let $k_j = \max\{k : \sum_{l=k}^{T} \tilde{\lambda}_j^l = 1\}$. Also let $D = \{j \in N \setminus \{i\} : k_j > 0\}$. Then, for each $l \in L$ and for each $C' \subseteq D$ such that $\sum_{j\in C'} a_j^{k_j} + a_i^l > b$, we have that for $C = C' \cup \{i\}$ and $k_i = l$ (11) may separate $\tilde{\lambda}$. In particular, it will cut off $\tilde{\lambda}$ if, for example, $\tilde{\lambda}_i^{l-1} = 0$ or $\alpha_i = 0$.

2.3 Aggregated Lifted Convexity Constraints

In this section we develop a small extension of the lifted convexity constraints that sometimes allows cutting off infeasible points that lifted convexity constraints cannot.

For any $I \subseteq N$ we can aggregate the relaxed convexity constraints to get the valid inequality

$$\sum_{i\in I}\sum_{k=1}^{T} \lambda_i^k \leq |I|, \tag{12}$$

which can be lifted in a manner similar to the convexity constraints if $I \neq N$.

Let $\tilde{\lambda} \in LS \setminus P$ and suppose $\tilde{\lambda}_l$ is *SOS2* infeasible and $k_l = \max\{k : \tilde{\lambda}_l^k > 0\}$. It may happen that $a_i^1 + a_l^{k_l} < b$ for all $i \in N \setminus \{l\}$ but

$$\sum_{i\in I} a_i^1 + a_l^{k_l-1} > b \tag{13}$$

for some $I \subseteq N \setminus \{l\}$. In this case, neither lifted convexity cuts of type I or II will separate $\tilde{\lambda}$, but (13) suggests that we may be able to lift (12) to get a separating inequality.

If (13) is satisfied, inequality

$$\sum_{i \in I} \sum_{k=1}^{T} \lambda_i^k + \alpha_l^{k_l} \lambda_l^{k_l} \leq |I| \tag{14}$$

is valid where $\alpha_l^{k_l} = |I| - z^*$ and

$$z^* = \max \left\{ \sum_{i \in I} \lambda_i^1 \, : \, \sum_{i \in I} a_i^1 \lambda_i^1 \leq b - a_l^{k_l - 1}, \quad 0 \leq \lambda_i^1 \leq 1 \quad \forall \, i \in I \right\}. \tag{15}$$

By condition (13), this yields $\alpha_l^{k_l} > 0$. The validity proof for (14) is similar to the one in [10] for lifted convexity constraints type I, with the difference that for (14) the lifting of (12) with respect to $\lambda_l^{k_l}$ is only done approximatedly.

The separation procedure for this inequality is a simple generalization of the procedure for the separation of Lifted Convexity cuts type I. Let $\tilde{\lambda} \in LS \setminus P$ be such that $\tilde{\lambda}_l$ violates *SOS2* and let $\tilde{k}_l = \max\{k \, : \, \tilde{\lambda}_l^k > 0\}$. We look for a set of indices I such that (13) is satisfied for $k_l = \tilde{k}_l$ and $\sum_{i \in I} \sum_{k=1}^{T} \lambda_i^k = |I|$. We can then solve (15) greedily to get $\alpha_l^{k_l}$ and add (14) to cut off the infeasible point.

3 Extensions of the *SOS2* model to the semicontinuous case

In this section we extend the *SOS2* model to the semicontinuous case and show how the cuts from the continuous case can be used in this extension.

Let $f_j(x_j)$ be a piecewise linear lower semicontinuous function which is linear in the segments (d_j^k, d_j^{k+1}) for $k \in \{0, \dots, T-1\}$. Specifically,

$$\begin{aligned}
f_j(d_j^0) &= \underline{c}_j^0 & & \\
\lim_{x_j \to d_j^{0+}} f_j(x_j) &= \overline{c}_j^0 \geq \underline{c}_j^0 & & \\
\lim_{x_j \to d_j^{k-}} f_j(x_j) &= \underline{c}_j^k & & k \in \{1, \dots, T-1\} \\
\lim_{x_j \to d_j^{k+}} f_j(x_j) &= \overline{c}_j^k & & k \in \{1, \dots, T-1\} \\
f_j(d_j^k) &= \min\{\underline{c}_j^k, \overline{c}_j^k\} & & k \in \{1, \dots, T-1\} \\
f_j(d_j^T) &= \lim_{x_j \to d_j^{T-}} f_j(x_j) = c_j^T. & &
\end{aligned}$$

An example of this type of function is shown in figure 1. When $\overline{c}_j^0 > \underline{c}_j^0$ we say there is a *fixed charge* type jump at 0.

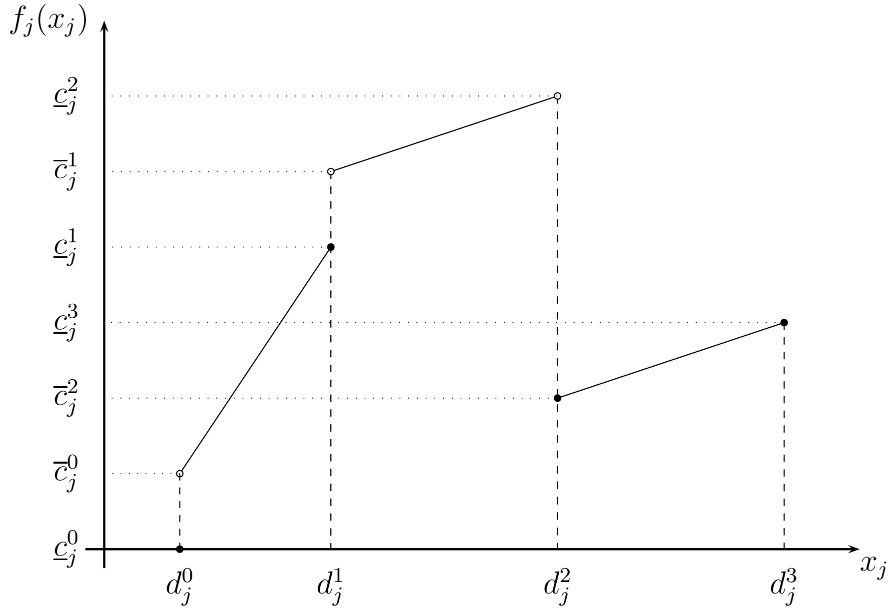

Fig. 1. A piecewise linear lower semicontinuous function.

To treat the discontinuous case we duplicate all break points except the upper bound of the x_j variable and make a distinction between the λ variable associated with the segment below and above d_j^k. We can then write

$$x_j = \sum_{k=0}^{T-1} [\underline{\lambda}_j^k + \overline{\lambda}_j^k] d_j^k + \underline{\lambda}_j^T d_j^T \tag{16}$$

where

$$\sum_{k=0}^{T-1} [\underline{\lambda}_j^k + \overline{\lambda}_j^k] + \underline{\lambda}_j^T = 1, \qquad \underline{\lambda}_j^T, \underline{\lambda}_j^k, \overline{\lambda}_j^k \geq 0 \quad \forall k \in \{0, \ldots, T-1\}. \tag{17}$$

Our intent is that if $x_j \in (d_j^k, d_j^{k+1})$ for some $k \in \{0, \ldots, T-1\}$ then

$$x_j = \overline{\lambda}_j^k d_j^k + \underline{\lambda}_j^{k+1} d_j^{k+1} \quad \text{and} \quad \overline{\lambda}_j^k + \underline{\lambda}_j^{k+1} = 1 \tag{18}$$

and that if $x_j = d_j^k$ for some $k \in \{0, \ldots, T-1\}$ then

$$(\underline{\lambda}_j^k, \overline{\lambda}_j^k) = \begin{cases} (1,0) & \text{if } \lim_{x_j \to d_j^{k-}} f_j(x_j) = f_j(d_j^k) \\ (0,1) & \text{if } \lim_{x_j \to d_j^{k+}} f_j(x_j) = f_j(d_j^k). \end{cases} \tag{19}$$

To assure (18) we only need to force

$$(\underline{\lambda}_j^0, \overline{\lambda}_j^0, \ldots, \underline{\lambda}_j^{T-1}, \overline{\lambda}_j^{T-1}, \underline{\lambda}_j^T) \text{ is } SOS2. \tag{20}$$

Let

$$f_j(x_j) = \sum_{k=0}^{T-1} [\underline{\lambda}_j^k \underline{c}_j^k + \overline{\lambda}_j^k \overline{c}_j^k] + \underline{\lambda}_j^T c_j^T. \tag{21}$$

Then, for x_j satisfying (16), (17) and (20), (21) is a correct expression for the piecewise linear function since (19) will be satisfied automatically by the minimization of $f(x)$ as $f_j(x_j)$ is lower semicontinuous for all $j \in N$.

Note that, if we do not have a fixed charge jump, this model is essentially the same as the *disaggregated* convex-combination binary model proposed in [3] and [14], but with the necessary combinatorial requirements enforced directly by *SOS2* constraints instead of adding binary variables. More specifically, when no fixed charge jump at 0 is present the *SOS2* model is

$$\begin{aligned}
\min \quad & \sum_{j \in N} \left(\overline{\lambda}_j^0 \overline{c}_j^0 + \sum_{k=1}^{T-1} [\underline{\lambda}_j^k \underline{c}_j^k + \overline{\lambda}_j^k \overline{c}_j^k] + \underline{\lambda}_j^T c_j^T \right) \\
s.t. \quad & \\
& \sum_{j \in N} \left(a_{ij}^0 \overline{\lambda}_j^0 + \sum_{k=1}^{T-1} a_{ij}^k [\underline{\lambda}_j^k + \overline{\lambda}_j^k] + a_{ij}^T \underline{\lambda}_j^T \right) \le b_i \qquad \forall\, i \in \{1, \ldots, m\} \\
& \overline{\lambda}_j^0 + \sum_{k=1}^{T-1} [\underline{\lambda}_j^k + \overline{\lambda}_j^k] + \underline{\lambda}_j^T = 1 \qquad \forall\, j \in N \\
& \overline{\lambda}_j^0, \underline{\lambda}_j^T, \underline{\lambda}_j^k, \overline{\lambda}_j^k \ge 0 \qquad \forall\, j \in N \\
& \qquad \forall\, k \in \{1, \ldots, T-1\} \\
& (\overline{\lambda}_j^0, \underline{\lambda}_j^1, \overline{\lambda}_j^1, \ldots, \underline{\lambda}_j^{T-1}, \overline{\lambda}_j^{T-1}, \underline{\lambda}_j^T) \text{ is } SOS2 \qquad \forall j \in N.
\end{aligned} \tag{22}$$

The *disaggregated* convex-combination binary model proposed in [3] and [14] is the same model with extra binary variables y_j^k and (22) replaced by

$$\begin{aligned}
\overline{\lambda}_j^{k-1} + \underline{\lambda}_j^k &= y_j^k && \forall\, j \in N,\, k \in \{1, \ldots, T\} \\
\sum_{k=1}^{T} y_j^k &\le 1 && \forall\, j \in N \\
y_j^k &\in \{0, 1\} && \forall\, j \in N,\, k \in \{1, \ldots, T\}.
\end{aligned}$$

As a direct extension of [9], we have that both models are equivalent in the sense that their LP relaxations have the same optimal objective value and that the convex hulls of their feasible sets are equal in the space of the λ variables. As it has been shown in [3] and [14] that the LP relaxation of the *disaggregated* convex-combination binary model produces a bound at least as tight as any of the other known models for piecewise linear optimization, this also holds for our *SOS2* model. On the other hand, the *SOS2* model is theoretically preferable as it has fewer variables and constraints.

Using the binary variable model to derive cuts could appear to be advantageous at first sight, as lifting binary variables is usually simpler than lifting continuous variables. We could lift variable y_j^k and use the obtained coefficient for variables $\overline{\lambda}_j^{k-1}$ and $\underline{\lambda}_j^k$, but we would then always have the same lifting coefficients for these two λ variables. This procedure would then fail to generate many valid inequalities. For example, we could not generate a lifted convexity cut (6) with $(\alpha_i^{k_i-1}, \alpha_i^{k_i}) \neq (0,0)$.

Finally, we note that a fixed charge jump can be added to both models.

3.1 Using cuts from the continuous model in the semicontinuous model

We now show how cuts derived for the continuous model can be used in the semicontinuous model by using a natural identification between the λ variables. We rename the λ variables in the discontinuous case as

$$(\underline{\lambda}_j^1, \overline{\lambda}_j^1, \ldots, \underline{\lambda}_j^{T-1}, \overline{\lambda}_j^{T-1}, \underline{\lambda}_j^T) = (\lambda_j^k)_{k=1}^{2T-1}.$$

If the fixed charge jump is not present, we eliminate $\underline{\lambda}_j^0$ from the formulation and rename $\overline{\lambda}_j^0$ to λ_j^0. On the other hand, if the fixed charge jump *is* present we keep $\underline{\lambda}_j^0$ and $\overline{\lambda}_j^0$ and add a new variable λ_j^0 plus the additional constraints $\underline{\lambda}_j^0 + \overline{\lambda}_j^0 = \lambda_j^0$ and $\underline{\lambda}_j^0 \in \{0,1\}$. Note that this binary requirement is not artificial and in fact the *SOS2* requirements plus the minimization of the lower semicontinuous function $f(x_j)$ will automatically enforce it. With these identifications and by renaming T as $T = 2T-1$ we recover the continuous model over variables $(\lambda_j^k)_{k=0}^T$ given by (2)–(5). The only difference is that instead of having $a_j^k < a_j^{k+1}$, we now have $a_j^k \leq a_j^{k+1}$. Thus all cuts derived from the continuous case can be used in the semicontinuous case so long as they were not deduced assuming the strict inequality. Fortunately, the loss of the strict inequality between breakpoints only seems to require some extra care when separating.

For lifted convexity cuts note that because $k_i = \min\{k : b - a_j^1 \leq a_i^k\}$ we have that $a_i^{k_i} = a_i^{k_i+1}$ and $a_i^{k_i-1} < a_i^{k_i}$. When the strict inequality assumption applies, if $b - a_j^1 = a_i^{k_i}$ then $\alpha_i^{k_i} = 0$ and $\alpha_i^{k_i+1} > 0$. On the other hand when the strict inequality assumption is dropped we still get $\alpha_i^{k_i} = 0$, but we also get $\alpha_i^{k_i+1} = 0$. This changes the condition for separation with a lifted convexity cuts type I from $b - a_j^1 \leq a_i^{\tilde{k}_i-1}$ to

$$b - a_j^1 < a_i^{\tilde{k}_i-1} \qquad \text{or} \qquad \{b - a_j^1 \leq a_i^{\tilde{k}_i-1} \text{ and } a_i^{\tilde{k}_i} \neq a_i^{\tilde{k}_i-1}\}. \tag{23}$$

In contrast, the conditions for the lifted convexity cuts type II and the aggregated lifted convexity cuts are not changed.

Finally, for lifted cover inequalities validity is preserved when the strict inequality assumption is dropped. The only difference is that $\alpha_j = 0$ whenever $a_j^{l_j-1} = a_j^{l_j}$.

4 Inequalities Using The Fixed Charge Jump

None of the previous valid inequalities include the fixed charge binary variable $\underline{\lambda}_j^0$. One approach to including these binary variables would be to lift them in the inequalities we have already studied. Unfortunately, this approach does not yield very good results. For the lifted convexity and aggregated lifted convexity cuts only the binary variables associated with the original convexity constraints may give non-zero lifted coefficients and even this rarely happens. For lifted cover cuts the results are not good either since if the cover C is chosen to be minimal the lifted coefficients for all $\underline{\lambda}_j^0$ for all $j \in C$ will be zero.

On the other hand, there are many cuts available for fixed charge *linear* problems, so we decided to study the possibility of extending these cuts to the piecewise linear case. One of the most studied fixed charge linear problems is the fixed charge network flow problem, see for example [11] section II.6.4, [12],[13],[15],[16] and [19]. Because of this, we will concentrate our study on two classical cuts for the fixed charge transportation problem: cover and flow cover cuts. We refer the reader to [11] sections II.2.2 and II.2.4 for an in depth treatment of these cuts and to [11] section II.6.4 for a description of their use in fixed charge network and transportation problems.

When a fixed charge jump is included for each variable x_j, our *SOS2* model is (2)–(5) and

$$\underline{\lambda}_j^0 + \overline{\lambda}_j^0 = \lambda_j^0 \qquad \forall j \in N \tag{24}$$

$$\underline{\lambda}_j^0 \in \{0, 1\} \qquad \forall j \in N \tag{25}$$

$$\overline{\lambda}_j^0 \geq 0 \qquad \forall j \in N. \tag{26}$$

As we will be extending cuts for the transportation problem we will also study the case when inequality (5) is replaced by

$$\sum_{j \in N} \sum_{k=0}^{T} a_j^k \lambda_j^k \geq b. \tag{27}$$

The feasible set for the problem with the $\leq$ inequality is still denoted by

$$S = \{\Lambda = (\lambda, (\underline{\lambda}_j^0, \overline{\lambda}_j^0)_{j\in N}) \in \mathbb{R}^{n(T+1)} \times (\{0,1\}\times\mathbb{R})^n \;:\; \Lambda \text{ satisfies } (2)\text{–}(5),(24)\text{–}(26)\;\}$$

and the feasible set for the problem with the $\geq$ inequality is denoted by

$$S^{\geq} = \{\Lambda \in \mathbb{R}^{n(T+1)} \times (\{0,1\} \times \mathbb{R})^n \;:\; \Lambda \text{ satisfies } (2)\text{–}(4),(27),(24)\text{–}(26)\}.$$

Similarly the feasible set for the problem with an equality constraint is denoted by $S^{=} = S \cap S^{\geq}$.

By setting $x_j = \sum_{k=0}^{T} a_j^k \lambda_j^k$ and $y_j = (1 - \underline{\lambda}_j^0)$ we obtain the relaxation of $S^{=}$ given by

$$\begin{aligned} x_j &\leq a_j^T y_j && \forall j \in N \\ \sum_{j\in N} x_j &= b && \qquad (28) \\ y_j &\in \{0,1\} && \forall j \in N \\ x_j &\geq 0 && \forall j \in N \end{aligned}$$

which is exactly the one row relaxation of a fixed charge linear transportation problem, from which classical cover and flow cover cuts can be derived.

Replacing (28) by $\sum_{j\in N} x_j \leq b$ we obtain a variable upper bound flow model from which we can derive flow cover inequalities. Similarly, replacing (28) by $\sum_{j\in N} x_j \geq b$ we obtain the binary knapsack model

$$\begin{aligned} \sum_{j\in N} a_j^T \underline{\lambda}_j^0 &\leq \sum_{j\in N} a_j^T - b \\ \underline{\lambda}_j^0 &\in \{0,1\} && \forall j \in N \end{aligned}$$

from which we can derive, for example, cover inequalities.

This approach can be extended to take into account the structure of the piecewise-linear problem by using the variables x_j in different ways.

Theorem 1 *Let $C \subseteq N$ and $k_j \geq 1$ for all $j \in C$ be such that $\sum_{j\in C} a_j^{k_j} = b + \Delta$ with $\Delta > 0$ then*

$$\sum_{j\in C} \sum_{k=1}^{k_j-1} a_j^k \lambda_j^k + \sum_{j\in C} a_j^{k_j} \sum_{k=k_j}^{T} \lambda_j^k + \sum_{j\in C} (a_j^{k_j} - \Delta)^+ \underline{\lambda}_j^0 \leq b \qquad (29)$$

is valid for $\mathrm{conv}(S)$.

PROOF. For each $j \in N$ we fix $k_j \geq 1$ and let

$$z_j = \sum_{k=1}^{k_j-1} a_j^k \lambda_j^k + a_j^{k_j} \sum_{k=k_j}^{T} \lambda_j^k. \tag{30}$$

Again using $y_j = (1 - \underline{\lambda}_j^0)$ we get a variable upper bound relaxation of S given by

$$z_j \leq a_j^{k_j} y_j \qquad \forall j \in N \tag{31}$$
$$\sum_{j \in N} z_j \leq b \tag{32}$$
$$y_j \in \{0,1\} \qquad \forall j \in N \tag{33}$$
$$z_j \geq 0 \qquad \forall j \in N \tag{34}$$

from which again we can derive flow cover cuts. For example, if $C \subseteq N$ is such that $\sum_{j \in C} a_j^{k_j} = b + \Delta$ with $\Delta > 0$ we get the flow cover inequality

$$\sum_{j \in C} z_j \leq b - \sum_{j \in C} (a_j^{k_j} - \Delta)^+ (1 - y_j) \tag{35}$$

which translates in the original variables to (29).

Once k_j has been chosen for each $j \in N$, the usual separation procedures for flow cover inequalities can be applied to choose C in (29). A reasonable choice of k_j's could be $k_j = \max\{k : \tilde{\lambda}_j^k > 0\}$ for a given $\tilde{\Lambda} \in LS \setminus P$ we wish to separate, but the choice of k_j will affect the coefficient of $\underline{\lambda}_j^0$, so including this choice in the separation procedure might give better results.

Inequality (29) could be improved by lifting variables in $N \setminus C$. Furthermore a possibly stronger inequality could be obtained by lifting the inequality

$$\sum_{j \in C} \sum_{k=1}^{k_j} a_j^k \lambda_j^k \leq b \tag{36}$$

which is clearly valid for $\operatorname{conv}(\{\Lambda \in S : \lambda_i^k = 0 \quad \underline{\lambda}_i^0 = 0 \quad \forall i \in C \quad \forall k \geq k_i + 1,$ $\lambda_i^k = 0 \quad \forall i \in N \setminus C, \forall k \geq 1\})$. In fact, inequality (36) can be lifted with respect to variables $\underline{\lambda}_i^0$ for each $i \in C$ to yield

$$\sum_{j \in C} \sum_{k=1}^{k_j} a_j^k \lambda_j^k + \sum_{j \in C} (a_j^{k_j} - \Delta)^+ \underline{\lambda}_j^0 \leq b,$$

which could presumably be lifted with respect to variables λ_i^k for $i \in C$ and $k \geq k_i + 1$ to get a valid inequality that dominates (29). Unfortunately, this last lifting and the lifting of (29) with respect to variables in $N \setminus C$ does not seem to be easy to compute.

On the other hand, the procedure used to prove validity of (29) can also be used to obtain valid inequalities similar to (29) that also include variables in $N \setminus C$. This can be done by simply replacing (35) by other valid inequalities for (31)–(34) like lifted flow cover inequalities [6]. Furthermore this procedure can be easily extended to the case where negative a_j's are allowed by using extensions to flow cover inequalities that allow negative coefficients like simple and extended generalized flow cover inequalities [11] section II.2.4, [13],[17],[20] and lifted flow cover inequalities [6].

We will now do a similar extension for cover cuts for $\mathrm{conv}(S^{\geq})$, but this time we will be forced to use lifting to obtain a valid inequality. During the lifting procedure we will use the following proposition, whose proof is analogous to the proof of Proposition 1. in [10].

Proposition 1 *Let Λ be an extreme point of $\mathrm{conv}(S^{\geq})$. Then Λ has at most two fractional components, and in case it has a fractional component it must satisfy (27) at equality. Furthermore, if $\lambda_{j_1}^{k_1}, \lambda_{j_2}^{k_2} \in (0,1)$, then $j_1 = j_2$, $k_2 = k_1 + 1$ or $k_2 = k_1 - 1$, and $\lambda_{j_1}^{k_1} + \lambda_{j_2}^{k_2} = 1$.*

Theorem 2 *Let $C \subseteq N$ and $k_j \geq 1$ for all $j \in N \setminus C$ be such that*

$$\rho = b - \sum_{i \in N \setminus C} a_i^{k_i} > 0 \tag{37}$$

$$\sum_{i \in N \setminus C} a_i^{k_i} + a_j^T \geq b \quad \forall j \in C \tag{38}$$

$$\sum_{i \in N \setminus (C \cup \{j\})} a_i^{k_i} + a_j^{k_j+1} \geq b \quad \forall j \in N \setminus C \tag{39}$$

then

$$\sum_{j \in C} \underline{\lambda}_j^0 + \sum_{i \in N \setminus C} \left[\left(\frac{a_i^{k_i} - a_i^{k_i+1}}{\rho} \right) \lambda_i^{k_i+1} - \sum_{k=k_i+2}^{T} \lambda_i^k \right] \leq |C| - 1 \tag{40}$$

is valid for $\mathrm{conv}(S^{\geq})$.

PROOF. Let $S_C^{\geq} = \{\Lambda \in S^{\geq} : \lambda_i^k = 0 \quad \forall i \in N \setminus C \quad \forall k \geq k_i + 1\}$. By letting

$$z_j = \begin{cases} \sum_{k=1}^{k_j} a_j^k \lambda_j^k & j \in N \setminus C \\ \sum_{k=1}^{T} a_j^k \lambda_j^k & j \in C \end{cases}$$

we get the knapsack relaxation of $S^{\geq}_C$ given by

$$\sum_{j\in N\setminus C} a_j^{k_j}\underline{\lambda}_j^0+\sum_{j\in C}a_j^T\underline{\lambda}_j^0\leq \sum_{j\in N\setminus C}a_j^{k_j}+\sum_{j\in C}a_j^T-b$$
$$\underline{\lambda}_j^0\in\{0,1\}\qquad\qquad \forall j\in N$$

from which we can deduce that the cover inequality given by

$$\sum_{j\in C}\underline{\lambda}_j^0\leq |C|-1 \tag{41}$$

is valid for $\operatorname{conv} S^{\geq}_C$. Inequality (40) will be obtained by lifting this cover inequality.

For a fixed $i\in N\setminus C$ we lift (41) with respect to λ_i^k for $k\geq k_i+1$ in increasing order. Let

$$PS^{\geq}_C(i,l)=\operatorname{conv}(\{\Lambda\in S^{\geq}\,:\,\lambda_j^k=0\quad\forall\, j\in N\setminus(C\cup\{i\})\quad\forall\, k\geq k_j+1,$$
$$\lambda_i^k=0\quad\forall\, k\geq l+1\}).$$

Suppose that for $l\geq k_i+1$

$$\sum_{j\in C}\underline{\lambda}_j^0+\sum_{k=k_i+1}^{l-1}\alpha_i^k\lambda_i^k\leq |C|-1 \tag{42}$$

has already been proven valid for $PS^{\geq}_C(i,l-1)$ and was obtained by maximum lifting. Then the maximum lifting coefficient for (42) with respect to λ_i^l is

$$\alpha_i^l=\min\frac{|C|-1-\sum_{j\in C}\underline{\lambda}_j^0-\sum_{k=k_i+1}^{l-1}\alpha_i^k\lambda_i^k}{\lambda_i^l}$$
$$s.t.\quad \Lambda\in V(PS^{\geq}_C(i,l)),\quad \lambda_i^l>0$$

where $V(P)$ is the set of extreme points of P [18]. To simplify this minimization problem we will study the cases $\lambda_i^l=1$ and $0<\lambda_i^l<1$ separately. Then if we let

$$\beta_i^l=\min|C|-1-\sum_{j\in C}\underline{\lambda}_j^0-\sum_{k=k_i+1}^{l-1}\alpha_i^k\lambda_i^k$$
$$s.t.\quad \Lambda\in V(PS^{\geq}_C(i,l)),\quad \lambda_i^l=1$$

and

$$\gamma_i^l=\min\frac{|C|-1-\sum\limits_{j\in C}\underline{\lambda}_j^0-\sum\limits_{k=k_i+1}^{l-1}\alpha_i^k\lambda_i^k}{\lambda_i^l}$$
$$s.t.\quad \Lambda\in V(PS^{\geq}_C(i,l)),\quad 0<\lambda_i^l<1$$

we have $\alpha_i^l = \min\{\beta_i^l, \gamma_i^l\}$. Note that by minimality condition (38) we have $\beta_i^l, \gamma_i^l \leq 0$. It is easy to see that

$$\begin{aligned}
\beta_i^l = \quad & \min |C| - 1 - \sum_{j\in C} \underline{\lambda}_j^0 \\
& s.t. \\
& \quad \sum_{j\in C}(1-\underline{\lambda}_j^0)a_j^T \geq b - a_i^l - \sum_{j\in N\setminus(C\cup\{i\})} a_j^{k_j} \\
& \quad \underline{\lambda}_j^0 \in \{0,1\}
\end{aligned}$$

and as $l \geq k_i + 1$ minimality condition (39) implies that $\beta_i^l = -1$ and hence $\alpha_i^l \leq -1$. Similarly and by using Proposition 1 and $\beta_i^l, \gamma_i^l \leq 0$, it is easy to see that

$$\begin{aligned}
\gamma_i^l = \quad & \min \frac{|C| - 1 - \sum_{j\in C}\underline{\lambda}_j^0 - \alpha_i^{l-1}(1-\lambda_i^l)}{\lambda_i^l} \\
& s.t. \\
& \quad \sum_{j\in C}(1-\underline{\lambda}_j^0)a_j^T = b - (1-\lambda_i^l)a_i^{l-1} - \lambda_i^l a_i^l - \sum_{j\in N\setminus(C\cup\{i\})} a_j^{k_j} \qquad (43) \\
& \quad \underline{\lambda}_j^0 \in \{0,1\} \\
& \quad 0 < \lambda_i^l < 1.
\end{aligned}$$

In particular for $l = k_i + 1$ we have

$$\begin{aligned}
\gamma_i^{k_i+1} = \quad & \min \frac{|C| - 1 - \sum_{j\in C}\underline{\lambda}_j^0}{\lambda_i^{k_i+1}} \\
& s.t. \\
& \quad \sum_{j\in C}(1-\underline{\lambda}_j^0)a_j^T = b - (1-\lambda_i^{k_i+1})a_i^{k_i} \\
& \qquad\qquad - \lambda_i^{k_i+1}a_i^{k_i+1} - \sum_{j\in N\setminus(C\cup\{i\})} a_j^{k_j} \\
& \quad \underline{\lambda}_j^0 \in \{0,1\} \\
& \quad 0 < \lambda_i^{k_i+1} < 1.
\end{aligned}$$

Any Λ feasible for this problem, such that $\sum_{j\in C}\underline{\lambda}_j^0 \leq |C| - 1$ has nonnegative objective value. On the other hand, the only feasible Λ with $\sum_{j\in C}\underline{\lambda}_j^0 = |C|$ is such that

$$\lambda_i^{k_i+1} = \frac{b - \sum_{j\in N\setminus C} a_j^{k_j}}{a_i^{k_i+1} - a_i^{k_i}} = \frac{\rho}{a_i^{k_i+1} - a_i^{k_i}}.$$

The value of $\gamma_i^{k_i+1}$ given by this solution is $(a_i^{k_i} - a_i^{k_i+1})/\rho$ which is less than or equal to -1 because of (39). Hence

$$\gamma_i^{k_i+1} = \frac{a_i^{k_i} - a_i^{k_i+1}}{\rho}.$$

Together with $\alpha_i^{k_i+1} = \min\{\beta_i^{k_i+1}, \gamma_i^{k_i+1}\}$ and $\beta_i^{k_i+1} = -1$ this yields

$$\alpha_i^{k_i+1} = \frac{a_i^{k_i} - a_i^{k_i+1}}{\rho}.$$

Similarly for $l \geq k_i + 2$ we have that the minimum in (43) is again attained by the unique Λ with $\sum_{j\in C} \underline{\lambda}_j^0 = |C|$, but now

$$\gamma_i^l = \frac{-(1 + \alpha_i^{l-1}(1 - \lambda_i^l))}{\lambda_i^l} \geq -1,$$

where the last inequality comes from $\alpha_i^l \leq -1$. So for $l \geq k_i + 2$ we have $\alpha_i^l = \beta_i^l = -1$. Now we see how the lifting can be done independently for each $i \in N \setminus C$. For $H \subset N \setminus C$ let

$$\begin{aligned} PS_C^{\geq}(i,l,H) = \mathrm{conv}(\{\Lambda \in S^{\geq} \,:\, \lambda_j^k = 0 \quad & \forall\, j \in N \setminus (C \cup H \cup \{i\}) \\ & \forall\, k \geq k_j + 1, \quad \lambda_i^k = 0 \quad \forall\, k \geq l+1\}). \end{aligned}$$

Suppose that we have already maximally lifted with respect to λ_j for all $j \in H$ and after that with respect to λ_i^k for all $k \in \{k_i + 1, \ldots, l-1\}$. Let $\hat{\alpha}_i^l$ be the maximum lifting coefficient for λ_i^l. We will prove by induction on $|H|$ that $\hat{\alpha}_i^l$ is equal to the coefficient α_i^l already calculated. The base case $|H| = 0$ follows from the definition of α_i^l. Now, for $|H| \geq 1$ by the induction hypothesis we have that

$$\begin{aligned} \hat{\alpha}_i^l = \min\ & \frac{|C| - 1 - \sum\limits_{j\in C} \underline{\lambda}_j^0 + \sum\limits_{j\in H} \sum\limits_{k \geq k_j+1} (-\alpha_j^k)\lambda_j^k - \sum\limits_{k=k_i+1}^{l-1} \alpha_i^k \lambda_i^k}{\lambda_i^l} \\ s.t.\ & \Lambda \in V(PS_C^{\geq}(i,l,H)), \quad \lambda_i^l > 0. \end{aligned}$$

As in the previous argument we can define $\hat{\beta}_i^l$ and $\hat{\gamma}_i^l$ such that $\hat{\alpha}_i^l = \min\{\hat{\beta}_i^l, \hat{\gamma}_i^l\}$.

Noting that $(-\alpha_j^k) > 0$ for all $j \in H$ and $k \geq k_j + 1$, it is easy to see that

$\hat{\beta}_i^l = \beta_i^l$. Also, by arguments similar to the previous part we have

$$
\begin{aligned}
\hat{\gamma}_i^l = \quad & \min \frac{|C| - 1 - \sum_{j \in C} \underline{\lambda}_j^0 + \sum_{j \in H} \sum_{k \geq k_j+1} (-\alpha_j^k)\lambda_j^k - \alpha_i^{l-1}(1-\lambda_i^l)}{\lambda_i^l} \qquad (44)\\
& s.t.\\
& \quad \sum_{j \in H} \sum_{k \geq k_j} a_j^k \lambda_j^k + \sum_{j \in C} (1 - \underline{\lambda}_j^0) a_j^T = b - (1-\lambda_i^l) a_i^{l-1} - \lambda_i^l a_i^l \\
& \qquad\qquad\qquad\qquad\qquad\qquad\qquad - \sum_{j \in N \setminus (C \cup H \cup \{i\})} a_j^{k_j}\\
& \quad \sum_{k \geq k_j} \lambda_j^k = 1 \qquad \forall\, j \in H\\
& \quad 0 \leq \lambda_j^k \leq 1 \qquad \forall\, j \in H\\
& \quad \underline{\lambda}_j^0 \in \{0,1\} \qquad \forall\, j \in C\\
& \quad 0 < \lambda_i^l < 1.
\end{aligned}
$$

Noting that $(-\alpha_j^k) \geq 1$ for all $j \in H$ and $k \geq k_i + 1$, it is easy to see that the minimum of (44) is attained at a Λ such that $\sum_{j \in C} \underline{\lambda}_j^0 = |C|$ and $\sum_{j \in H} \sum_{k \geq k_j+1} \lambda_j^k = 0$. Under these conditions the problem reverts to the one defining γ_i^l so we have $\hat{\gamma}_i^l = \gamma_i^l$ and hence $\hat{\alpha}_i^l = \alpha_i^l$.

Because of ρ, an exact separation problem for (40) will not have a linear objective function, but there is a simple heuristic way of separating a given $\tilde{\Lambda} \in LS \setminus P$ by starting with $C = \{i \in N : \tilde{\underline{\lambda}}_i^0 = 1\}$ and $k_i = \max\{k : \tilde{\lambda}_i^k > 0\}$ for $i \in N \setminus C$. If necessary we can then add to C indexes $i \in N \setminus C$ with large $\tilde{\underline{\lambda}}_i^0$ to comply with the cover condition (37). Finally, if needed, we can easily correct our choices of C and k_i's to comply with the minimality conditions (38) and (39).

Unfortunately, inequality (40) cannot be directly extended to other inequalities for the knapsack problem. If we start the lifting with other inequalities instead of (41), such as lifted cover inequalities, the lifting problem with respect to continuous variables becomes much harder. The lifting of (40) with respect to binary variables $\underline{\lambda}_j^0$ for $j \in N \setminus C$ seems like a better alternative, but it is still not clear how to give a closed form expression for the lifting coefficients.

5 Computational Experience

In [10] it was shown that adding cuts could significantly improve the performance of an *SOS2* based branch and bound procedure for solving linear problems with piecewise linear separable objective functions. It was also shown that using an *SOS2* model was faster than using a binary model with or without the use of *SOS2* cuts. Advocates of the binary model could argue that this last statement is no longer valid for practical applications as commercial solvers are now so efficient at solving mixed integer problems that the benefit of being able to use their features outweighs the drawbacks of adding extra binary variables. For this reason we decided to use a state of the art commercial solver to evaluate the current practical applicability of the *SOS2* branch and cut procedure. We chose CPLEX 9.0 [8] as a MIP solver using Concert 2.0 [8] as the modeling language because it has built in *SOS2* support.

We modeled the problem using Concert's built in *SOS2* support and for the binary model we chose the disaggregated convex combination model introduced in [3] and [14]. Initial testing showed that the benefit of using *SOS2* sets were not significant when using CPLEX and in fact many times the binary model solved faster. CPLEX's does not generate any cuts in solving a model without binary or integer variables, so we also compared the results of solving the *SOS2* model with CPLEX against solving the binary model with CPLEX's cuts turned off. In this case the advantage of the binary model was diminished but it was still faster to solve than the *SOS2* model. One reason for this behavior is that CPLEX 9.0's branching, preprocessing and primal heuristics for binary variables are much more advanced than those for *SOS2* sets [7]. In theory, most of these binary preprocessing and variable branching schemes translate to *SOS2* preprocessing and branching schemes that could be implemented without binary variables giving even better performance, but it remains to be seen if they are actually worth the programming effort.

The disaggregated convex combination model is a way to implement *SOS2* requirements for the piecewise linear model. The advantages and disadvantages of this approach and the direct implementation of *SOS2* requirements are summarized in table 1.

From our preliminary computational results it seems that currently the best *practical* implementation of *SOS2* requirements is the disaggregated convex combination model. For this reason we decided to implement *SOS2* sets using this approach to test our cuts. Our aim was to study the change in performance when using our *SOS2* based cuts by themselves and also in conjunction with CPLEX's cuts.

Table 1
Qualitative Comparison of Binary and *SOS2* models.

Attribute	**Disaggregated convex combination binary model**	**Concert *SOS2* direct implementation**
# of Variables	More variables, slower LP solve.	Fewer variables, faster LP solve.
# of Constraints	More constraints, slower LP solve.	Fewer constraints, faster LP solve.
Advanced Preprocessing	Currently available.	Theoretically it can be implemented. No current implementation.
Advanced Branching and node selection	Currently available. Constraint branching can also be used	Theoretically it can be implemented. No current implementation. Constraint branching can be used.
Advanced heuristics and RINS [4]	Currently available.	Theoretically it can be implemented. No current implementation.

5.1 Test Instances

Our test instances were based on the same randomly generated transportation problems used in [10], but we modified the objective functions to make the problems harder to solve. We also included a relaxation of the transportation problems in our tests.

The transportation problems consist of the minimization of a nonconvex separable piecewise linear function. As shown in figure 2, functions $f_{ij}(x_{ij})$, for each arc x_{ij} in the underlying transportation graph, were randomly generated by first generating a strictly increasing concave piecewise linear function with $f(0) = 0$. Discontinuities for each break point were then generated by randomly decreasing $\lim_{x_{ij}\to d_{ij}^{k}{}^-} f_{ij}(x_{ij})$ for each $k \geq 1$ and fixed charges were generated by randomly increasing $\lim_{x_{ij}\to 0^+} f_{ij}(x_{ij})$. Finally the value of $f_{ij}(d_{ij}^k)$ was defined so that $f_{ij}(x_{ij})$ would end up being lower semicontinuous. We refer to these instances as the *continuous/discontinuous transportation problems with/without fixed charge.*

The relaxation of the transportation problem only includes the constraints at the supply nodes which were further relaxed to inequality constraints. These problems involve the maximization of a nonconcave separable piecewise linear function. Functions $f_{ij}(x_{ij})$ for these problems were generated in a way analogous to the transportation problem. We refer to these instances as the *con-*

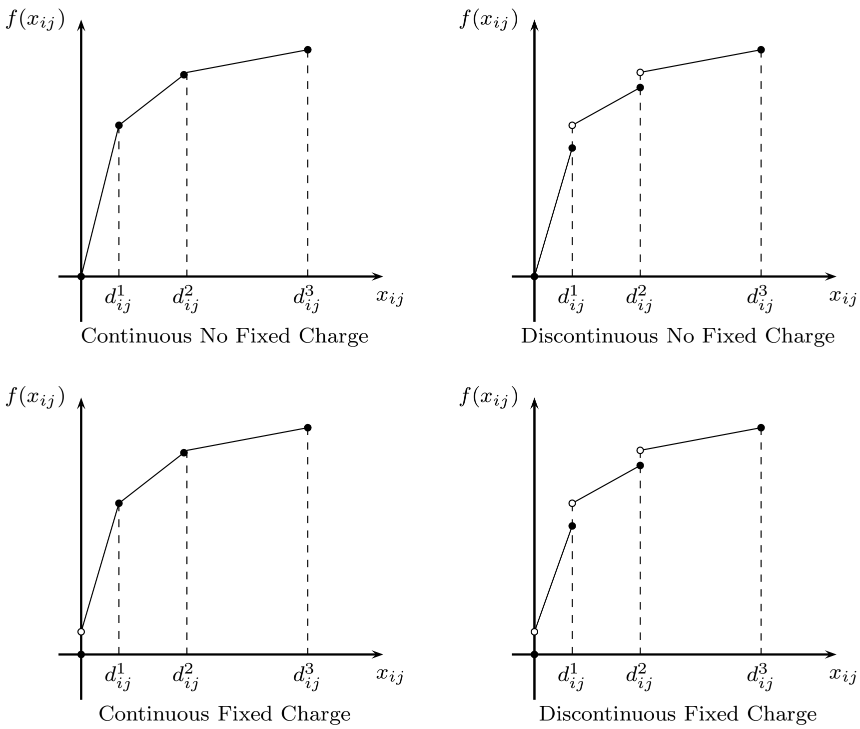

Fig. 2. Construction of piecewise linear function for transportation problem

tinuous/discontinuous maximization problems with/without fixed charge. We included these instances as they only have less than or equal to constraints with positive coefficients and most of the valid inequalities considered in this paper are based on a one row relaxation that has a constraint of this kind. Thus these instances allow us to test the performance of our valid inequalities independently of the effects of other one row relaxations for which we can not generate valid inequalities.

For both types of problems we considered instances with different numbers of supply and demand nodes. We use 4 and 5 segments for the piecewise linear functions as was done in [10].

5.2 Computational Results

To perform computational tests we used a PC with dual 2.40GHz Xeon CPU's and 2 GB of RAM running Linux with kernel 2.4.20.

Tables 2 to 9 summarize results for all problem types. Each problem type is identified as *xxx-yyy-zzz*, where *xxx* is *max* if it is a maximization problem or *transp* if it is a transportation problem, *yyy* is *FC* if the problem's objective function includes a fixed charge or *noFC* if it does not and *zzz* is *cont* if the problem's objective function is continuous besides a possible fixed charge or *disc* if it is not.

In each table a particular instance is identified as $a \times b \times c.d$ where a,b,c and d correspond to the number of supply nodes, number of demand nodes, number of segments of the objective function and the particular seed used for the generation of the problem respectively.

For each instance we present results when solving it using CPLEX 9.0 with its default settings, with CPLEX's cuts turned off and our *SOS2* based cuts and with CPLEX's cuts turned on and our *SOS2* based cuts. In the case where we use our *SOS2* based cuts, we aggressively generated cuts at the root node and we then kept generating cuts every 1000 nodes in a more conservative manner. The exception for this are fixed charge cover cuts which we generated every 5000 nodes. For all cases we present the number of nodes required to solve the instance and the CPU time in seconds. Each run was terminated after at most $10,000$ CPU seconds. Instances which were not solved to optimality in this time frame are marked with a $*$ in the CPU time column followed by the optimality gap at the time of termination. For each problem type we also include the total number of nodes processed and the total CPU time for each method. We also include the number of times each method obtained the best gap, by either solving to optimality when one of the other methods could not or by obtaining the smallest gap when none of the methods reached optimality. Finally, for each instance we use bold font to denote the method that obtained the best number of nodes, CPU time or gap.

We also give in table 10 the total number of *SOS2* based cuts that were generated for each problem type. We consider separately the number of cuts generated when only *SOS2* based cuts were generated and when they were generated in conjunction with CPLEX's default cuts. Columns labeled (A) correspond to lifted convexity cuts (6), columns (B) correspond to lifted cover cuts (11), columns (C) correspond to aggregated lifted convexity cuts (14), columns (D) corresponds to fixed charge flow cover cuts (29) and columns (E) correspond to fixed charge cover cuts (40).

For the maximization problem we can see that using only *SOS2* based cuts instead of CPLEX's default cuts gives significantly better results when the number of nodes processed is considered. CPLEX took almost 13 and 16 times more nodes to solve both the continuous and discontinuous instances with no fixed charges and over 23 and 8 times more nodes to solve the fixed charge ones. Furthermore, two instances which could not be solved to optimality by CPLEX were solved by using only *SOS2* based cuts. When CPU time is considered instead, *SOS2* based cuts still give better results, but the difference is not so significant as CPLEX only takes over 8 and almost 10 times more CPU time to solve instances with no fixed charges and almost 10 and 4 times more CPU time to solve the fixed charge ones. When the *SOS2* based cuts are used in conjunction with CPLEX's default cuts the results are even better. In this case the speed up is 19, 16, 38 and 18 times with respect to number of nodes and 12, 14, 14 and almost 6 times with respect to CPU time.

For the transportation problems *SOS2* based cuts still improve performance with respect to number of nodes, but the speed up is smaller. Using *SOS2* based cuts in conjunction with CPLEX's default cuts is still the fastest approach, but compared with only using *SOS2* cuts the difference is small. When using only *SOS2* based cuts the speed up is 10, almost 8, almost 14 and 15 times with respect to number of nodes and when using *SOS2* based cuts in conjunction with CPLEX's default cuts the speed up is 10, 8, 17 and 17 times. There is very little difference between the approaches with respect to CPU time although the approaches that use *SOS2* based cuts are slightly faster. Using only *SOS2* based cuts does allow us to get better gaps in 30 instances and using *SOS2* in conjunction with CPLEX's default cuts allows us to get better gaps in 37 instances. Using only CPLEX's default cuts got better gaps in just 4 instances. We believe that the reason for the lack of significant speed up in CPU time for these instances is that the current implementation of the separation procedures for fixed charge flow cover cuts and fixed charge cover cuts are too slow. The significant speed up in number of nodes and the number of cuts generated suggest that these cuts are useful though.

6 Conclusions

This paper extends the branch-and-cut algorithm for linear programs with piecewise linear continuous costs developed in [10] to the lower semicontinuous case. We extend the classical *SOS2* formulation for linear programs with piecewise linear continuous costs to the lower semicontinuous case in the same way the classical binary model was extended in [3] and [14]. We note that additional work in this direction has been developed in [5] where the *SOS2* formulation has been extended to the non-lower semicontinuous case by introducing a specialized branching scheme for this case. We then bring valid inequalities developed in [10] to the new model and make a simple generalization of one of these inequalities. Finally we study in detail the discontinuity caused by a fixed charge at 0 and we develop two new valid inequalities by extending classical cuts for fixed charge linear models.

Computationally, we compare the branch-and-cut algorithm without binary variables to solving the binary model with a commercial solver. Computational results show that, although the binary model works better with commercial solvers, adding *SOS2* based cuts can significantly increase performance of the branch and cut procedure for one class of problems. For the other class of problems, adding *SOS2* based cuts can significantly increase performance regarding the number of nodes and best gaps obtained and can provide a small increment in performance regarding CPU time.

References

[1] E. H. Aghezzaf, L. A. Wolsey, Modeling piecewise linear concave costs in a tree partitioning problem, Discrete Appl. Math. 50(1994) 101–109.

[2] A. Balakrishnan, S. Graves, A composite algorithm for a concave-cost network flow problem, Networks 19(1989) 175–202.

[3] K. L. Croxton, B. Gendron, T. L. Magnanti, A Comparison of mixed-integer programming models for nonconvex piecewise linear cost minimization problems, Manage. Sci. 49 (2003) 1268–1273.

[4] E. Danna, E. Rothberg, C. Le Pape, Exploring relaxation induced neighborhoods to improve MIP solutions, Math. Program. 102(2005) 71–90.

[5] I. R. de Farias Jr. M. Zhao, H. Zhao, A special ordered set approach to discontinuous piecewise linear optimization, To appear Oper. Res. Lett.

[6] Z. Gu, G. L. Nemhauser, W. P. Savelsbergh, Lifted flow cover inequalities for mixed 0-1 integer programs, Math. Program. 85(1999) 439–467.

[7] Z. Gu, Personal Comunications.

[8] ILOG Cplex 9.0: user's manual and reference manual, ILOG, S.A., http://www.ilog.com/, 2003.

[9] A. B. Keha, I. R. de Farias Jr., G. L. Nemhauser, Models for representing piecewise linear cost functions, Oper. Res. Lett. 32(2004) 44–48.

[10] A. B. Keha, I. R. de Farias Jr., G. L. Nemhauser, A branch-and-cut algorithm without binary variables for nonconvex piecewise linear optimization, Oper. Res. 54(2006) 847–858.

[11] G. L. Nemhauser, L. A. Wolsey, Integer and combinatorial optimization, Wiley-Interscience, New York, 1988

[12] F. Ortega, L. A. Wolsey, A branch-and-cut algorithm for the single-commodity, uncapacitated, fixed-charge network problem, Networks 41(2003) 143–158.

[13] M. W. Padberg, T. J. Van Roy, L. A. Wolsey, Valid inequalities for fixed charge problems, Oper. Res 33(1985) 842–861.

[14] H. D. Sherali, On mixed-integer zero-one representations for separable lower-semicontinuous piecewise linear functions, Oper. Res. Lett. 28(2001) 155–160.

[15] J. Stallaert, Valid inequalities and separation for capacitated fixed charge flow problems, Discrete Appl. Math. 98(2000) 265–274.

[16] T. J. Van Roy, L. A. Wolsey, Valid inequalities and separation for uncapacitated fixed charge networks, Oper. Res. Lett. 4(1985) 105–112.

[17] T. J. Van Roy, L. A. Wolsey, Valid inequalities for mixed 0-1 programs, Discrete Appl. Math. 14(1986) 199–213.

[18] L. A. Wolsey, Facets and strong valid inequalities for integer programs, Oper. Res 24(1976) 367–372.

[19] L. A. Wolsey, Submodularity and valid inequalities in capacitated fixed charge networks, Oper. Res. Lett. 8(1989) 119–124.

[20] L. A. Wolsey, Strong formulations for mixed integer programming: a survey, Math. Program. 45(1989), 173–191.

	CPLEX cuts		**SOS2 based cuts**		**Both cuts**	
Instance	Nodes	Time	Nodes	Time	Nodes	Time
10×10×4.1	77	**0.27**	109	0.29	**47**	0.37
10×10×4.2	736	0.92	**149**	**0.41**	221	0.72
10×10×4.3	72	**0.33**	121	0.52	**40**	0.69
10×10×4.4	238	0.43	136	**0.34**	**79**	0.41
10×10×4.5	47	0.19	25	**0.14**	**11**	0.21
10×10×5.1	56	**0.24**	19	0.25	**2**	0.39
10×10×5.2	587	**1.10**	605	1.55	**150**	1.39
10×10×5.3	44	**0.28**	**32**	0.37	34	0.66
10×10×5.4	63	**0.32**	48	0.36	**37**	0.40
10×10×5.5	482	**0.89**	430	1.16	**242**	1.31
12×18×4.1	12241	20	12448	23	**2683**	**6.20**
12×18×4.3	72458	112	16064	29	**7912**	**20**
12×18×4.4	5290	8.97	**1088**	**3.33**	1466	4.40
12×18×4.5	17024	27	10448	21	**5190**	**12**
12×18×5.1	307534	581	**22939**	**74**	25579	79
12×18×5.2	57570	110	**18786**	**57**	24738	65
12×18×5.3	320535	633	43491	133	**33642**	**110**
12×18×5.4	2544223	5154	83020	219	**48628**	**143**
12×18×5.5	16728	32	6118	20	**3171**	**12**
15×15×4.1	335	**1.18**	470	1.77	**273**	2.06
15×15×4.2	1526	3.40	1418	**3.23**	**791**	3.48
15×15×4.3	55721	89	11036	23	**6820**	**16**
15×15×4.4	11001	19	3598	7.80	**2886**	**7.00**
15×15×4.5	91	**0.63**	85	0.78	**38**	1.18
15×15×5.1	2858	7.30	2674	7.81	**1223**	**5.41**
15×15×5.2	154	**1.08**	199	2.18	**12**	3.78
15×15×5.3	7413	16	5897	21	**2574**	**11**
15×15×5.4	2167	5.64	1225	4.49	**500**	**4.03**
15×15×5.5	3498	7.66	1701	5.64	**1133**	**4.89**
20×20×4.1	185414	518	47240	160	**16240**	**64**
20×20×4.2	1362	4.97	868	5.19	**218**	**4.84**
20×20×4.3	33735	87	14676	52	**5718**	**25**
20×20×4.4	19648	55	8530	37	**5695**	**27**
20×20×4.5	35850	98	6536	25	**3425**	**14**
20×20×5.1	88827	293	12996	130	**10426**	**80**
20×20×5.2	5811	21	7128	68	**3990**	**20**
20×20×5.3	100451	315	18349	121	**10123**	**66**
20×20×5.4	1373919	4731	**34247**	208	38790	**193**
20×20×5.5	71607	246	19248	100	**10694**	**58**
Total	5357393	13203.43	414197	1568.97	275441	1070.01

Table 2
Cplex cuts v/s SOS2 based cuts v/s both cuts for max-noFC-cont.

	CPLEX cuts		SOS2 based cuts		Both cuts	
Instance	Nodes	Time	Nodes	Time	Nodes	Time
10×10×4.1	77	**0.22**	111	0.29	**53**	0.35
10×10×4.2	1538	1.74	**150**	**0.43**	465	0.97
10×10×4.3	68	**0.31**	95	0.47	**34**	0.64
10×10×4.4	161	**0.35**	121	0.36	**76**	0.38
10×10×4.5	32	0.15	25	**0.13**	**11**	0.24
10×10×5.1	59	0.25	19	**0.24**	**2**	0.34
10×10×5.2	471	**0.97**	575	1.57	**120**	1.37
10×10×5.3	54	**0.30**	34	0.38	**32**	0.61
10×10×5.4	56	**0.29**	**46**	0.36	47	0.39
10×10×5.5	525	**0.91**	349	1.08	**236**	1.34
12×18×4.1	8541	15	7016	14	**4468**	**8.56**
12×18×4.3	69581	107	20290	38	**11677**	**26**
12×18×4.4	7723	12	**1590**	**4.25**	1837	5.26
12×18×4.5	14333	25	10311	21	**3380**	**9.49**
12×18×5.1	468604	940	**24666**	**85**	30656	88
12×18×5.2	94469	182	35956	90	**16019**	**47**
12×18×5.3	1255000	2701	33187	102	**23762**	**84**
12×18×5.4	3870138	8294	61176	171	**37196**	**110**
12×18×5.5	12339	25	6300	17	**3071**	**12**
15×15×4.1	321	**1.17**	448	1.73	**262**	1.66
15×15×4.2	1607	3.68	1379	**2.08**	**859**	3.71
15×15×4.3	28540	48	13223	25	**8398**	**22**
15×15×4.4	15222	28	3549	6.93	**2415**	**6.85**
15×15×4.5	96	**0.71**	87	0.78	**40**	1.22
15×15×5.1	2618	6.83	3134	9.77	**1467**	**5.68**
15×15×5.2	146	**1.01**	187	2.22	**12**	3.59
15×15×5.3	6814	16	4980	23	**2229**	**12**
15×15×5.4	2195	6.06	1251	4.48	**542**	**4.19**
15×15×5.5	3426	7.69	1641	**5.31**	**1472**	5.95
20×20×4.1	246788	709	79214	254	**38084**	**146**
20×20×4.2	1070	**4.61**	650	4.92	**283**	5.44
20×20×4.3	30860	82	16605	62	**7309**	**34**
20×20×4.4	22151	61	7779	37	**6889**	**28**
20×20×4.5	34841	100	6198	22	**3635**	**17**
20×20×5.1	100078	331	14407	109	**9064**	**72**
20×20×5.2	8182	30	8245	49	**4469**	**26**
20×20×5.3	76203	245	19558	154	**8174**	**65**
20×20×5.4	325904	1102	32243	173	**22207**	**132**
20×20×5.5	106206	350	16260	91	**10350**	**55**
Total	6817037	15440.6	433055	1582.33	261302	1041.13

Table 3
Cplex cuts v/s SOS2 based cuts v/s both cuts for max-noFC-disc.

	CPLEX cuts		SOS2 based cuts		Both cuts	
Instance	Nodes	Time	Nodes	Time	Nodes	Time
10×10×4.1	42	**0.19**	97	0.41	**6**	0.53
10×10×4.2	573	**0.85**	485	0.94	**197**	0.91
10×10×4.3	188	**0.43**	114	0.58	**33**	0.82
10×10×4.4	491	**0.78**	463	1.22	**313**	0.97
10×10×4.5	**9**	**0.12**	16	0.18	10	0.25
10×10×5.1	250	**0.72**	302	1.01	**249**	1.00
10×10×5.2	196	**0.63**	721	2.16	**92**	2.81
10×10×5.3	61	**0.33**	85	0.52	**0**	0.86
10×10×5.4	50	**0.28**	**48**	0.36	**48**	0.53
10×10×5.5	779	**1.17**	662	2.10	**275**	3.62
12×18×4.1	9305	16	**4459**	**15**	4604	16
12×18×4.3	4381	8.25	4612	10	**2641**	**6.81**
12×18×4.4	19336	29	11098	40	**7073**	**24**
12×18×4.5	17494	29	8471	27	**2935**	**12**
12×18×5.1	2173353	4016	27143	142	**16918**	**100**
12×18×5.2	225370	419	33537	184	**27581**	**148**
12×18×5.3	865506	1676	25447	149	**20282**	**123**
12×18×5.4	916513	1776	**59030**	316	64791	**303**
12×18×5.5	49118	96	13513	54	**11301**	**36**
15×15×4.1	223	**1.20**	417	2.59	**166**	3.17
15×15×4.2	295000	470	32534	**86**	**29169**	92
15×15×4.3	37056	58	**6159**	**21**	9051	26
15×15×4.4	2923	**5.58**	2799	14	**1906**	9.83
15×15×4.5	2764	**5.67**	3047	7.69	**2210**	7.32
15×15×5.1	23138	45	**5498**	**18**	7960	27
15×15×5.2	52	**0.83**	459	2.90	**19**	11
15×15×5.3	26542	49	**2833**	**19**	4870	29
15×15×5.4	18252	39	4034	20	**1174**	**10**
15×15×5.5	1570	**4.82**	1954	6.10	**781**	6.35
20×20×4.1	3672027	*(0.14)	207834	1074	**49010**	**278**
20×20×4.2	5340	**17**	2267	21	**1033**	20
20×20×4.3	8213	25	3944	36	**1774**	**20**
20×20×4.4	59	**1.05**	67	2.75	**31**	8.30
20×20×4.5	2732	**11**	4413	34	**1542**	34
20×20×5.1	1262240	3937	67869	730	**50544**	**615**
20×20×5.2	9308	**32**	**2735**	38	3613	34
20×20×5.3	11505	39	**2167**	**39**	3923	95
20×20×5.4	155238	515	**11927**	149	13287	**123**
20×20×5.5	3541079	*(0.35)	17126	129	**5341**	**97**
Total	13358276	33324.61	570386	3395.9	346753	2329.16
Best Gap		0		2		2

Table 4
Cplex cuts v/s SOS2 based cuts v/s both cuts for max-FC-cont.

	CPLEX cuts		**SOS2 based cuts**		**Both cuts**	
Instance	Nodes	Time	Nodes	Time	Nodes	Time
10×10×4.1	45	**0.21**	100	0.38	**6**	0.51
10×10×4.2	488	**0.74**	480	0.94	**166**	0.87
10×10×4.3	189	**0.45**	94	0.62	**35**	0.84
10×10×4.4	525	**0.84**	363	1.06	**98**	0.93
10×10×4.5	**9**	**0.14**	15	0.18	10	0.24
10×10×5.1	234	**0.57**	280	0.91	**224**	0.92
10×10×5.2	235	**0.72**	395	1.83	**55**	2.73
10×10×5.3	60	**0.33**	88	0.53	**0**	0.84
10×10×5.4	50	**0.25**	**45**	0.38	48	0.47
10×10×5.5	950	**1.44**	701	2.16	**200**	3.58
12×18×4.1	17409	28	5043	16	**4550**	**13**
12×18×4.3	4131	**7.09**	3092	7.80	**2557**	7.78
12×18×4.4	23563	34	11015	43	**5448**	**25**
12×18×4.5	15206	25	8399	28	**3914**	**14**
12×18×5.1	568511	982	22334	87	**17437**	**86**
12×18×5.2	204526	358	47268	219	**27691**	**176**
12×18×5.3	445306	796	25835	**146**	**23952**	160
12×18×5.4	1409775	2565	65173	318	**42589**	**235**
12×18×5.5	54105	103	16001	58	**10218**	**42**
15×15×4.1	313	**1.39**	402	2.60	**161**	3.10
15×15×4.2	346717	545	35543	92	**24298**	**83**
15×15×4.3	26959	42	**6330**	**20**	7351	28
15×15×4.4	3053	**6.32**	4121	15	**2484**	20
15×15×4.5	2218	**4.63**	**1944**	6.04	3783	9.09
15×15×5.1	24379	45	6778	33	**4286**	**20**
15×15×5.2	52	**0.84**	462	2.95	**19**	10
15×15×5.3	61355	111	**2798**	**14**	14025	43
15×15×5.4	2123	**6.03**	3527	16	**1089**	9.65
15×15×5.5	1167	**3.88**	1394	5.73	**391**	6.29
20×20×4.1	955399	2849	299101	1512	**47948**	**252**
20×20×4.2	4526	**14**	3236	24	**2074**	25
20×20×4.3	6460	**20**	4600	22	**1583**	21
20×20×4.4	58	**1.04**	67	2.85	**31**	7.60
20×20×4.5	3364	**12**	3169	24	**2654**	24
20×20×5.1	1207054	3685	64900	**565**	**49739**	645
20×20×5.2	11246	**39**	**2754**	60	3454	48
20×20×5.3	6295	**23**	2782	53	**1158**	79
20×20×5.4	79722	257	20332	168	**14045**	**126**
20×20×5.5	455790	1412	14721	201	**4617**	**130**
Total	5943567	13984.33	685682	3771.44	324388	2363.1

Table 5
Cplex cuts v/s SOS2 based cuts v/s both cuts for max-FC-disc.

	CPLEX cuts		**SOS2 based cuts**		**Both cuts**	
Instance	Nodes	Time	Nodes	Time	Nodes	Time
10×10×4.1	9718	13	5897	**8.60**	**5326**	11
10×10×4.2	1570	2.59	1398	**2.40**	**1045**	2.59
10×10×4.3	1086	2.21	**712**	**1.70**	815	2.26
10×10×4.4	59	**0.40**	117	0.60	**51**	0.77
10×10×4.5	3763	5.69	1922	4.00	**1845**	**3.93**
10×10×5.1	2246	4.43	**1241**	**3.23**	1387	5.29
10×10×5.2	7395	14	**5061**	**11**	5290	13
10×10×5.3	323	**1.38**	**200**	1.59	212	2.34
10×10×5.4	2072	4.64	923	3.66	**723**	**3.03**
10×10×5.5	8518	15	7184	**13**	**4924**	14
12×18×4.1	3309701	*(1.15)	**280990**	**2608**	388385	4611
12×18×4.3	26969	70	**13370**	66	14044	**63**
12×18×4.4	93024	244	18902	78	**14027**	**76**
12×18×4.5	517778	1567	89530	**590**	**88825**	650
12×18×5.1	250111	874	42788	539	**38267**	**372**
12×18×5.2	3529422	*(4.20)	**212101**	***(3.00)**	218993	*(3.80)
12×18×5.3	3247218	*(0.65)	**136464**	**2344**	152239	3302
12×18×5.4	330796	1090	**34202**	**325**	36994	405
12×18×5.5	1908704	6306	234541	4150	**130504**	**2661**
15×15×4.1	594858	1829	**109770**	**914**	110440	1132
15×15×4.2	370651	1022	108095	**603**	**104843**	659
15×15×4.3	1505726	5173	189003	1837	**152827**	**1723**
15×15×4.4	227764	611	37393	247	**33745**	**208**
15×15×4.5	389622	1161	**52278**	**275**	59051	423
15×15×5.1	1281506	4327	170249	3724	**129501**	**2625**
15×15×5.2	3013948	*(2.43)	**192888**	**5036**	236674	6635
15×15×5.3	3072531	*(2.26)	**238454**	**5252**	251917	6609
15×15×5.4	181290	696	49924	944	**44525**	**596**
15×15×5.5	3519275	*(6.09)	254810	*(5.98)	**208531**	***(4.32)**
20×20×4.1	1983385	*(7.46)	247584	*(7.15)	**223191**	***(5.73)**
20×20×4.2	1861493	*(1.39)	306312	5960	**238358**	**5248**
20×20×4.3	1954141	*(2.93)	**280082**	***(1.51)**	268616	*(1.71)
20×20×4.4	1973907	*(6.21)	**238830**	***(5.37)**	190281	*(7.20)
20×20×4.5	1009948	4767	184346	**3283**	**159676**	3539
20×20×5.1	1901715	*(8.91)	144321	*(8.82)	**155457**	***(8.48)**
20×20×5.2	**1709657**	***(6.61)**	138422	*(8.12)	155049	*(8.90)
20×20×5.3	1482120	*(7.54)	133100	*(7.00)	**119236**	***(6.76)**
20×20×5.4	1921427	*(9.74)	145027	*(10.84)	**126086**	***(9.14)**
20×20×5.5	1364321	*(7.97)	**134001**	***(6.08)**	120274	*(7.61)
Total	44569758	179799.3	4442432	138824.36	4192174	141595.61
Best Gap		1		9		10

Table 6
Cplex cuts v/s SOS2 based cuts v/s both cuts for transp-noFC-cont.

	CPLEX cuts		**SOS2 based cuts**		**Both cuts**	
Instance	Nodes	Time	Nodes	Time	Nodes	Time
10×10×4.1	2829	5.20	1858	3.45	**1234**	**3.01**
10×10×4.2	651	**1.49**	**501**	1.55	511	1.93
10×10×4.3	573	1.30	291	1.13	**249**	**1.12**
10×10×4.4	197	0.73	32	**0.48**	**6**	0.91
10×10×4.5	1591	3.12	1810	**2.32**	**1204**	2.84
10×10×5.1	738	**2.16**	**567**	2.17	690	3.77
10×10×5.2	3074	6.68	2643	**4.73**	**1926**	6.27
10×10×5.3	239	**1.05**	190	1.49	**64**	1.89
10×10×5.4	487	**1.60**	283	1.96	**200**	3.03
10×10×5.5	2574	5.74	2215	4.60	**1488**	**4.49**
12×18×4.1	722168	2199	137598	888	**127110**	**872**
12×18×4.3	5215	16	4879	**16**	**4372**	16
12×18×4.4	32108	91	12920	60	**8317**	**39**
12×18×4.5	213770	656	**43215**	**199**	48744	297
12×18×5.1	72896	259	19684	**175**	**18137**	193
12×18×5.2	2976909	*(2.75)	**316346**	***(0.46)**	256354	*(1.15)
12×18×5.3	442140	1391	**35537**	**312**	52966	735
12×18×5.4	94945	341	23977	182	**23488**	**180**
12×18×5.5	176702	592	47353	522	**36284**	**334**
15×15×4.1	189553	601	**36859**	**241**	58174	490
15×15×4.2	108020	305	37947	175	**29863**	**144**
15×15×4.3	131285	428	28234	**168**	**27064**	179
15×15×4.4	32022	88	**13298**	**62**	13725	82
15×15×4.5	62386	190	**15985**	**68**	16838	83
15×15×5.1	490236	1692	**48396**	**551**	62272	708
15×15×5.2	446102	1643	**56869**	**658**	88086	1514
15×15×5.3	1597445	8915	**62303**	**676**	87221	1365
15×15×5.4	69915	414	24458	316	**21531**	**183**
15×15×5.5	2446533	*(5.31)	267827	*(3.12)	**252785**	***(2.76)**
20×20×4.1	1467191	*(5.47)	276398	*(4.76)	**266335**	***(4.61)**
20×20×4.2	443557	3516	119486	1608	**82886**	**1405**
20×20×4.3	1138180	*(0.91)	**193803**	**4542**	274232	6018
20×20×4.4	1326488	*(5.16)	242747	*(3.71)	**234069**	***(3.16)**
20×20×4.5	565157	4017	96044	1527	**55810**	**797**
20×20×5.1	1270320	*(7.27)	**157087**	***(6.64)**	140360	*(7.17)
20×20×5.2	1578741	*(6.08)	**191001**	***(6.05)**	157241	*(6.94)
20×20×5.3	1592919	*(5.17)	146261	*(3.93)	**134323**	***(3.34)**
20×20×5.4	1848134	*(7.44)	158929	*(6.65)	**155001**	***(5.37)**
20×20×5.5	1652678	*(5.95)	**140109**	***(4.51)**	138528	*(4.93)
Total	23206668	127383.38	2965940	102970.78	2879688	105663.29
Best Gap		0		5		6

Table 7
Cplex cuts v/s SOS2 based cuts v/s both cuts for transp-noFC-disc.

	CPLEX cuts		**SOS2 based cuts**		**Both cuts**	
Instance	Nodes	Time	Nodes	Time	Nodes	Time
10×10×4.1	11822	18	5028	**12**	**4799**	15
10×10×4.2	6989	11	5617	10	**1941**	**4.75**
10×10×4.3	1872	**3.40**	2146	4.04	**1284**	3.82
10×10×4.4	110	**0.60**	155	1.23	**71**	1.70
10×10×4.5	10912	17	6526	**9.94**	**4705**	13
10×10×5.1	8679	15	8710	20	**4229**	**14**
10×10×5.2	91207	160	**15913**	80	20229	**71**
10×10×5.3	**1082**	**2.75**	1116	5.82	1394	5.97
10×10×5.4	3021	**6.50**	3035	12	**2116**	13
10×10×5.5	40217	59	**14625**	**52**	18711	56
12×18×4.1	2968910	*(1.36)	355825	7291	**171793**	**3491**
12×18×4.3	103737	316	**25067**	**287**	26627	303
12×18×4.4	222660	552	19261	**256**	**17985**	279
12×18×4.5	1307114	5112	121665	1759	**118996**	**1620**
12×18×5.1	3200785	*(0.49)	85760	2345	**68108**	**2180**
12×18×5.2	2980359	*(6.24)	152001	*(4.08)	**147556**	***(4.07)**
12×18×5.3	2875942	*(3.54)	**180347**	***(3.05)**	21001	*(8.20)
12×18×5.4	1132544	3521	88118	2467	**63931**	**1569**
12×18×5.5	2394031	*(2.97)	**237784**	**8835**	170530	*(2.65)
15×15×4.1	1072654	3833	**100128**	**1691**	105382	2304
15×15×4.2	687775	2550	**77712**	**697**	101242	1191
15×15×4.3	2620258	*(2.26)	290555	*(1.21)	**188654**	**4057**
15×15×4.4	417251	1089	69255	742	**45153**	**409**
15×15×4.5	870461	3483	**57178**	**676**	69903	888
15×15×5.1	2884161	*(2.42)	**185097**	**7912**	184183	*(0.92)
15×15×5.2	2835407	*(7.32)	156001	*(4.81)	**154000**	***(4.74)**
15×15×5.3	2670548	*(4.38)	**160973**	***(1.54)**	181507	*(1.67)
15×15×5.4	2658701	8909	157852	6979	**117267**	**4685**
15×15×5.5	2908891	*(8.42)	157001	*(6.73)	**132518**	***(5.50)**
20×20×4.1	1610144	*(8.27)	**114001**	***(7.17)**	107001	*(7.33)
20×20×4.2	149246	*(1.82)	166001	*(1.51)	**156065**	***(1.22)**
20×20×4.3	1898997	*(2.26)	**145001**	***(1.93)**	126103	*(2.04)
20×20×4.4	1566417	*(6.14)	125148	*(6.33)	**100482**	***(6.02)**
20×20×4.5	1618928	*(0.51)	204777	*(0.32)	**132625**	**6069**
20×20×5.1	1650041	*(11.16)	51748	*(11.06)	**49908**	***(9.49)**
20×20×5.2	1642505	*(8.01)	60436	*(8.32)	**60506**	***(6.79)**
20×20×5.3	1497741	*(10.66)	44000	*(8.82)	**48903**	***(8.53)**
20×20×5.4	1667211	*(14.41)	45001	*(12.59)	**43900**	***(12.42)**
20×20×5.5	**1644338**	***(9.37)**	46659	*(10.34)	45000	*(10.00)
Total	51933668	229661.21	3743223	202145.61	3016308	189241.55
Best Gap		1		8		13

Table 8
Cplex cuts v/s SOS2 based cuts v/s both cuts for transp-FC-cont.

	CPLEX cuts		**SOS2 based cuts**		**Both cuts**	
Instance	Nodes	Time	Nodes	Time	Nodes	Time
10×10×4.1	4579	7.71	**1485**	**4.41**	1985	7.32
10×10×4.2	3238	5.77	3472	6.23	**1948**	**5.20**
10×10×4.3	1322	**2.75**	**990**	3.09	1339	4.93
10×10×4.4	87	**0.49**	80	1.06	**32**	1.71
10×10×4.5	4514	**7.17**	**4034**	7.32	4508	13
10×10×5.1	3653	**7.32**	**3217**	7.97	3577	16
10×10×5.2	24649	46	**6085**	28	8721	**24**
10×10×5.3	945	**2.50**	1072	4.07	**692**	6.26
10×10×5.4	**489**	**1.87**	1097	6.34	901	7.66
10×10×5.5	10778	18	**6378**	**17**	9874	24
12×18×4.1	2173838	7834	285594	5214	**155958**	**2342**
12×18×4.3	6660	**22**	**6017**	45	7164	78
12×18×4.4	36257	**89**	**11965**	163	12378	139
12×18×4.5	378608	1333	**67287**	**739**	84762	776
12×18×5.1	848909	2736	58861	1396	**31660**	**746**
12×18×5.2	3469023	*(5.85)	201061	*(2.88)	**178376**	***(2.70)**
12×18×5.3	2453048	7335	**91420**	**2577**	131197	5124
12×18×5.4	524181	1570	47099	591	**23337**	**543**
12×18×5.5	2002878	7724	113926	2717	**79638**	**2195**
15×15×4.1	633556	2200	29001	*(3.15)	**50286**	**865**
15×15×4.2	161670	440	**40076**	**235**	45452	406
15×15×4.3	552886	2036	67317	**1067**	**57676**	1078
15×15×4.4	62013	**147**	26041	207	**21032**	171
15×15×4.5	118514	361	**27982**	**278**	32389	298
15×15×5.1	1685290	6617	103147	3480	**90598**	**2909**
15×15×5.2	2849164	*(4.74)	**171082**	***(2.05)**	182884	*(2.98)
15×15×5.3	2495035	*(2.59)	**167754**	**6653**	174064	7327
15×15×5.4	848063	2875	61125	1776	**50907**	**1382**
15×15×5.5	2922026	*(5.98)	164891	*(4.48)	**146382**	***(4.36)**
20×20×4.1	1618321	*(6.98)	**111001**	***(6.06)**	105622	*(6.52)
20×20×4.2	354148	2122	**38797**	**1785**	15000	*(2.14)
20×20×4.3	1896239	*(1.20)	150570	6730	**137917**	**6247**
20×20×4.4	1500046	*(5.89)	**127391**	***(4.79)**	70000	*(5.48)
20×20×4.5	1203688	7713	91196	**2515**	**79513**	2624
20×20×5.1	1791917	*(9.29)	**59001**	***(8.36)**	57397	*(9.20)
20×20×5.2	1711543	*(8.21)	75001	*(8.22)	**70001**	***(7.72)**
20×20×5.3	1455162	*(6.71)	**64000**	***(6.42)**	46969	*(7.37)
20×20×5.4	1603970	*(10.40)	56058	*(9.01)	**59540**	***(8.71)**
20×20×5.5	1756480	*(9.13)	51329	*(6.22)	**59628**	***(6.09)**
Total	39167387	173253.2	2593900	148255.17	2291304	145358.82
Best Gap		2		8		8

Table 9
Cplex cuts v/s SOS2 based cuts v/s both cuts for transp-FC-disc.

	SOS2 based cuts					Both cuts				
Problem Type	(A)	(B)	(C)	(D)	(E)	(A)	(B)	(C)	(D)	(E)
max-noFC-cont	763	4003	44	0	0	610	3077	48	0	0
max-noFC-disc	759	3836	40	0	0	642	3104	41	0	0
max-FC-cont	821	4614	36	245	0	765	4528	53	228	0
max-FC-disc	813	4608	34	284	0	807	4682	51	265	0
transp-noFC-cont	8328	44276	564	0	0	8470	44843	584	0	0
transp-noFC-disc	5821	29938	390	0	0	5732	29449	388	0	0
transp-FC-cont	6504	46884	724	3207	18243	6124	43800	739	35843	17228
transp-FC-disc	4870	49555	520	2636	17138	47222	33395	560	2600	14890

Table 10
Total number of SOS2 based cuts generated.