A note on :"A Superior Representation Method for Piecewise Linear Functions"

Juan Pablo Vielma
Business Analytics and Mathematical Sciences Department, IBM T. J. Watson Research Center, Yorktown Heights, NY 10598, and Department of Industrial Engineering, University of Pittsburgh, Pittsburgh, PA 15261, jvielma@pitt.edu

Shabbir Ahmed, George Nemhauser
H. Milton Stewart School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, Georgia 30332, {shabbir.ahmed@isye.gatech.edu, george.nemhauser@isye.gatech.edu}

This paper studies two Mixed Integer Linear Programming (MILP) formulations for piecewise linear functions considered in Li et al. (2009). Although the ideas used to construct one of these formulations are theoretically interesting and could eventually provide a computational advantage, we show that their use in modeling piecewise linear functions yields a poor MILP formulation. We specifically show that neither of the formulations in Li et al. (2009) have a favorable strength property shared by all standard MILP formulations for piecewise linear functions. We also show that both formulations in Li et al. (2009) are significantly outperformed computationally by standard MILP formulations.

Key words: Mathematics:Piecewise linear; Programming:Integer;
History:

---

1 Introduction

Two Mixed Integer Linear Programming (MILP) formulations for modeling a univariate piecewise linear function f were considered in Li et al. (2009). The first formulation (given by (1)–(3) in Li et al.) uses "Big-M" type constraints, so we denote it by *BigM*. The second formulation (given by (23)–(33) in Li et al.) uses a number of binary variables that is logarithmic in the number of segments in which f is affine, so we denote it by *LiLog*. Based on computational results that show that LiLog outperforms BigM, Li et al. declare LiLog to be superior to other MILP formulations for piecewise linear functions. The ideas used to construct LiLog are very interesting and have some potential practical applications. However, in this paper we show that BigM and LiLog lack a favorable strength property

shared by all standard MILP formulations for piecewise linear functions. We additionally show that both formulations are significantly outperformed computationally by standard MILP formulations.

In Section 2 we describe the formulations considered in Li et al. (2009) and in Section 3 we show that neither of these formulations has a favorable strength property shared by all standard MILP formulations for piecewise linear functions. In Section 4 we present results of computational experiments that compare the Li et al. formulations to two other standard formulations.

2 Piecewise Linear Models Introduced by Li et. al

For $a_0 < a_1 < \ldots < a_K$ let $f : [a_0, a_K] \subset \mathbb{R} \to \mathbb{R}$ be a univariate continuous function that is affine in $[a_{i-1}, a_i]$ for each $i \in \{1, \ldots, K\}$ and is defined by

$$f(x) := \Big\{ m_i x + c_i \quad x \in [d_{i-1}, d_i] \quad \forall i \in \{1, \ldots, K\}. \tag{1}$$

We now describe two formulations of $z = f(x)$ for f defined in (1) that are considered in Li et al. (2009). We refer the reader to Li et al. (2009) for a detailed explanation of these formulations.

The first formulation, denoted by BigM, is

$$a_{i-1} - (a_K - a_0)(1 - y_i) \le x \le a_i + (a_K - a_0)(1 - y_i) \quad \forall i \in \{1, \ldots, K\} \tag{2a}$$

$$m_i x + c_i - M(1 - y_i) \le z \le m_i x + c_i + M(1 - y_i) \quad \forall i \in \{1, \ldots, K\} \tag{2b}$$

$$\sum_{i=1}^{K} y_i = 1, \quad y_i \in \{0, 1\} \qquad \forall i \in \{1, \ldots, K\}, \tag{2c}$$

where

$$M := \max_{i=1}^{K} \max\{m_i a_K + c_i, m_i a_0 + c_i\} - \min_{i=1}^{K} \min\{m_i a_K + c_i, m_i a_0 + c_i\}.$$

The second one, denoted by LiLog is

$$\sum_{j=0}^{K-1} a_j \mu_{1,j} \leq x \tag{3a}$$

$$\sum_{j=0}^{K-1} a_{j+1} \mu_{1,j} \geq x \tag{3b}$$

$$\sum_{j=0}^{K-1} \big(f(a_j) - m_{i+1}(a_j - a_0)\big)\mu_{1,j} + m_{i+1}\mu_{2,j} = z \tag{3c}$$

$$\sum_{j=0}^{K-1} \mu_{1,j} = 1 \tag{3d}$$

$$\sum_{j=0}^{K-1} |\sigma(B(j))|\, \mu_{1,j} + \sum_{l=1}^{\lceil \log_2 K \rceil} z_{1,l} = 0 \tag{3e}$$

$$-u_l \leq z_{1,l} \qquad \forall l \in \{1, \ldots, \lceil \log_2 K \rceil\}, \tag{3f}$$

$$u_l \geq z_{1,l} \qquad \forall l \in \{1, \ldots, \lceil \log_2 K \rceil\}, \tag{3g}$$

$$\sum_{j=0}^{K-1} (-1)^{2-B(j)_l} \mu_{1,j} - (1 - u_l) \leq z_{1,l} \qquad \forall l \in \{1, \ldots, \lceil \log_2 K \rceil\} \tag{3h}$$

$$\sum_{j=0}^{K-1} (-1)^{2-B(j)_l} \mu_{1,j} + (1 - u_l) \geq z_{1,l} \qquad \forall l \in \{1, \ldots, \lceil \log_2 K \rceil\} \tag{3i}$$

$$\sum_{j=0}^{K-1} \mu_{2,j} + a_0 = x \tag{3j}$$

$$\sum_{j=0}^{K-1} |\sigma(B(j))|\, \mu_{2,j} + \sum_{l=1}^{\lceil \log_2 K \rceil} z_{2,l} = 0 \tag{3k}$$

$$-(a_K - a_0)u_l \leq z_{2,l} \qquad \forall l \in \{1, \ldots, \lceil \log_2 K \rceil\}, \tag{3l}$$

$$(a_K - a_0)u_l \geq z_{2,l} \qquad \forall l \in \{1, \ldots, \lceil \log_2 K \rceil\}, \tag{3m}$$

$$\sum_{j=0}^{K-1} (-1)^{2-B(j)_l} \mu_{2,j} - (a_K - a_0)(1 - u_l) \leq z_{2,l} \qquad \forall l \in \{1, \ldots, \lceil \log_2 K \rceil\} \tag{3n}$$

$$\sum_{j=0}^{K-1} (-1)^{2-B(j)_l} \mu_{2,j} + (a_K - a_0)(1 - u_l) \geq z_{2,l} \qquad \forall l \in \{1, \ldots, \lceil \log_2 K \rceil\} \tag{3o}$$

$$\mu_{1,j}, \mu_{2,j} \geq 0 \qquad \forall j \in \{0, \ldots, K-1\} \tag{3p}$$

$$u \in \{0,1\}^{\lceil \log_2 m \rceil} \tag{3q}$$

$$\sum_{l=1}^{\lceil \log_2 K \rceil} 2^{l-1} u_l \leq K - 1, \tag{3r}$$

where $B : \{0, \ldots, m-1\} \to \{0,1\}^{\lceil \log_2 m \rceil}$ is the binary expansion of a number given by

$$B(i) := v \in \{0,1\}^{\lceil \log_2 m \rceil} \text{ s.t. } i = \sum_{l=1}^{\lceil \log_2 m \rceil} 2^{l-1} v_l$$

and $\sigma : \{0,1\}^{\lceil \log_2 m \rceil} \to 2^{\{1,\dots,\lceil \log_2 m \rceil\}}$ is the support of a binary vector given by

$$\sigma(v) = \big\{ l \in \{1, \dots, \lceil \log_2 m \rceil\} \, : \, v_l = 1 \big\}.$$

We note that LiLog is original to Li et al. (2009) and BigM is presented in Li et al. (2009) as the "current method" for modeling piecewise linear functions. However, we are not aware of BigM being used in any other publication. We also note that in Li et al. (2009) the right hand side of (3r) is equal to K, which is most likely the result of a typo as this can be trivially strengthened to $K - 1$.

3 Strength of Formulations

An MILP formulation for a univariate piecewise linear function $f : [a_0, a_K] \to \mathbb{R}$ is $Q := \{(x, y, \lambda, \mu) \in P \, : \, \mu \in \{0,1\}^q\}$ such that $q \in \mathbb{Z}_+$, P is a polyhedron and $\text{proj}_{(x,y)}(Q) = \text{gr}(f) := \{(x, y) \, : \, f(x) = y\}$, where $\text{proj}_{(x,y)}(\cdot)$ is the projection onto the (x, y) variables. An MILP formulation Q is said to be *sharp* (Jeroslow and Lowe, 1984) if its linear programming (LP) relaxation P is such that $\text{proj}_{(x,y)}(P) = \text{conv}(\text{gr}(f))$. Sharp formulations provide the best possible LP relaxation bounds so the sharpness property is crucial for the efficient solution of these problems using branch-and-bound. An MILP formulation is *locally ideal* (Padberg, 2000) if P has integral extreme points. The locally ideal property can provide an additional advantage because it implies the sharpness property (e.g. Vielma et al., 2008). The sharpness property is shared by essentially every standard MILP formulation for piecewise linear functions (This is shown in Vielma et al. (2008) when considering the modeling of $\text{epi}(f) := \{(x, y) \, : \, f(x) \leq y\}$ and the extension to modeling $\text{gr}(f)$ is direct) and most of them are also locally ideal. We now show that neither BigM nor LiLog is sharp. To achieve this we will use the fact that for a sharp formulation we have that $y \geq g(x)$ for all $(x, y) \in \text{proj}_{(x,y)}(P)$, where $g := \text{convenv}_{[a_0,a_K]}(f)$ is the lower convex envelope of f over $[a_0, a_K]$.

For BigM we use the piecewise linear function $f : [0, 4] \to \mathbb{R}$ defined by

$$f(x) := \begin{cases} 3x & x \in [0, 1] \\ 2x + 1 & x \in [1, 4], \end{cases} \tag{4}$$

for which $g(1) = 2.25$ for $g = \text{convenv}_{[0,4]}(f)$. BigM for this function is

$$\begin{aligned}
-4(1-\lambda_0) \le x &\le 1 + 4(1-\lambda_0) \\
1 - 4(1-\lambda_1) \le x &\le 4 + 4(1-\lambda_1) \\
3x - 12(1-\lambda_0) \le y &\le 3x + 12(1-\lambda_0) \\
2x + 1 - 12(1-\lambda_1) \le y &\le 2x + 1 + 12(1-\lambda_1) \\
\lambda_0 + \lambda_1 = 1\,, &\quad \lambda_0, \lambda_1 \in \{0,1\}
\end{aligned}$$

which has $\lambda_0 = \lambda_1 = 0.5$, $x = 1$, and $y = -3 < 2.25$ as a feasible solution to its LP relaxation.

For LiLog we use the piecewise linear function $f : [0,4] \to \mathbb{R}$ defined by

$$f(x) := \begin{cases} 4x & x \in [0,1] \\ 3x+1 & x \in [1,2] \\ 2x+3 & x \in [2,3] \\ x+6 & x \in [3,4], \end{cases}$$

for which $g(1) = 2.5$ for $g = \text{convenv}_{[0,4]}(f)$. LiLog for this function is

$$
\begin{aligned}
r_1 + 2r_2 + 3r_3 \leq x \ &\leq r_0 + 2r_1 + 3r_2 + 4r_3 \\
r_1 + 3r_2 + 6r_3 + 4w_0 + 3w_1 + 2w_2 + w_3 &= y \\
r_0 + r_1 + r_2 + r_3 &= 1 \\
r_1 + r_2 + 2r_3 + z_1 + z_2 &= 0 \\
-u'_1 \leq z_1 &\leq u'_1 \\
-u'_2 \leq z_2 &\leq u'_2 \\
r_0 - r_1 + r_2 - r_3 - (1 - u'_1) \leq z_1 &\leq r_0 - r_1 + r_2 - r_3 + (1 - u'_1) \\
r_0 + r_1 - r_2 - r_3 - (1 - u'_2) \leq z_2 &\leq r_0 + r_1 - r_2 - r_3 + (1 - u'_2) \\
w_0 + w_1 + w_2 + w_3 &= x \\
w_1 + w_2 + 2w_3 + \delta_1 + \delta_2 &= 0 \\
-4u'_1 \leq \delta_1 &\leq 4u'_1 \\
-4u'_2 \leq \delta_2 &\leq 4u'_2 \\
w_0 - w_1 + w_2 - w_3 - 4(1 - u'_1) \leq \delta_1 &\leq w_0 - w_1 + w_2 - w_3 + 4(1 - u'_1) \\
w_0 + w_1 - w_2 - w_3 - 4(1 - u'_2) \leq \delta_2 &\leq w_0 + w_1 - w_2 - w_3 + 4(1 - u'_2) \\
u'_1 + 2u'_2 &\leq 3 \\
r_0, r_1, r_2, r_3, w_0, w_1, w_2, w_3 &\geq 0 \\
u'_1, u'_2 &\in \{0, 1\}
\end{aligned}
$$

which has $x = 1$, $r_0 = r_1 = 0.5$, $r_2 = r_3 = 0$, $u'_1 = 0.5$, $u'_2 = 0$, $w_0 = w_1 = w_2 = 0$, $w_3 = 1$, $z_1 = -0.5$, $z_2 = 0$, $\delta_1 = -2.0$, $\delta_2 = 0$ and $y = 1.5 < 2.5$ as a feasible solution to its LP relaxation.

4 Computational Results

We now present a computational comparison between BigM, LiLog and two standard MILP formulations. Most standard MILP formulations for piecewise linear functions are studied in Vielma et al. (2008). From these formulations we selected the *Convex Combination Model* and *Logarithmic Convex Combination Model* as a representative sample. The Convex

Combination Model appears as early as Dantzig (1960) and is included in many textbooks (Dantzig, 1963; Garfinkel and Nemhauser, 1972; Nemhauser and Wolsey, 1988). Although it is a sharp formulation, it is the only formulation studied in Vielma et al. (2008) that is not locally ideal and it has one of the worst computational performances. Hence it is an example of a classical, but relatively weak formulation. The Logarithmic Convex Combination Model is a sharp and locally ideal formulation introduced in Vielma and Nemhauser (2008a,b) that has a number of binary variables and constraints that is logarithmic in the number of segments in which the modeled function is affine. It has one of the best computational performances in Vielma et al. (2008) and hence is an example of a state of the art formulation. We denote the Convex Combination and Logarithmic Convex Combination Models by *CC* and *Log* respectively.

The first set of instances are from Li et al. and consists of a series of Mixed Integer Nonlinear Programming (MINLP) problems that were obtained by linearizing some Nonlinear Nonconvex Programming (NNP) problems. These MINLPs were obtained by replacing the nonconvex portions of the NNPs by univariate piecewise linear approximations and correspond to Examples 1 and 2 in Li et al. To solve these instances we used Bonmin 1.0.1 (Bonami et al., 2008) with CPLEX 11 (ILOG, 2008) as an MILP subsolver on a 2.4GHz workstation with 2GB of RAM. We selected the Hybrid solver from Bonmin because most of the time it significantly outperforms the other Bonmin solvers. Table 1 shows the solve times in seconds for the different instances, which are identified according to their parameters (e.g. Example 1 in Li et al. has three possible sets of parameter that we identify as 1a, 1b and 1c) and the resolution of the piecewise linear approximations (e.g. Example 2 in Li et al. included approximations with piecewise linear functions with 32 and 64 segments).

	Example 1a		Example 1b			Example 1c			Example 2	
segments	64	256	64	128	256	64	128	256	32	64
Log	1.1	3.3	0.2	0.4	0.7	0.3	0.4	0.7	6.1	11.2
CC	4.8	12.1	1.0	1.6	3.2	1.8	2.1	5.7	24.1	59.5
LiLog	4.3	23.5	1.1	2.2	5.4	1.2	2.5	5.6	116.0	129.2
BigM	1.9	19.2	1.0	2.5	17.5	1.0	4.5	15.8	214.0	306.0

Table 1: Solve times for MINLPs using Bonmin's hybrid algorithm [s].

We see that LiLog is rarely faster than CC and is always at least three times slower than Log. In fact, for the instances from Example 2 the solve times of both LiLog and BigM

are more than twice the time of CC and over an order of magnitude the time of Log. For most instances and formulations, Bonmin solved the problems at the root branch-and-bound node. The exception was Example 2 for which CC required 1406 nodes for 64 segments, LiLog required 4997 and 6362 nodes for 32 and 64 segments respectively and BigM required 43776 and 42852 nodes for 32 and 64 segments respectively.

The second set of instances from Li et al. are MILPs resulting from problems with univariate piecewise linear functions. These instances include Example 4 and a variant of Example 1 from Li et al. constructed by replacing every nonlinearity (convex and nonconvex) of the original NNP with a piecewise linear approximation. We note that for this variant we did not treat the convex nonlinearities specially so that we could assess the performance of the different formulations even in the case in which some of the piecewise linear functions can actually be modeled as LPs. These instances were solved using CPLEX 11. Table 2 shows the results for these instances in the same format as Table 1.

	Example 1a		Example 1b			Example 1c			Example 4		
segments	64	256	64	128	256	64	128	256	32	64	128
Log	0.03	0.07	0.04	0.03	0.10	0.02	0.03	0.11	0.10	0.24	0.47
CC	0.08	0.57	0.38	0.41	1.38	0.30	0.62	1.41	0.58	1.79	3.77
LiLog	0.45	2.86	0.37	1.47	4.60	0.34	1.41	8.96	6.21	34.09	260.68
BigM	1.35	73.34	1.40	8.75	66.90	1.16	7.60	46.78	11.22	56.35	428.06

Table 2: Solve times for MILPs from Li et al. (2009) [s].

For Example 1 the number of branch-and-bound nodes processed ranged from 8 to 21, 10 to 29, 209 to 2499 and 514 to 533 for Log, CC, LiLog and BigM respectively. For Example 4 the number of nodes processed ranged from 156 to 236, 200 to 455, 4732 to 75390 and 2077 to 13912 for Log, CC, LiLog and BigM respectively. The results are very similar to those for the first set and agree with the fact that BigM and LiLog are weaker than standard formulations for piecewise linear functions.

The final set of instances consists of the transportation problems with piecewise linear cost functions studied in Vielma et al. (2008). These instances consider univariate piecewise linear functions that are affine in K segments for $K \in \{4, 8, 16, 32\}$ and include 100 randomly generated instances for each K. We again used CPLEX 11 as an MILP solver and Table 3 shows the minimum, average, maximum and standard deviation of the solve times in seconds. The table also shows the number of times the solves failed because the time limit of 10,000

seconds was reached. We note that BigM was not considered for $K = 16$ and 32 because it had already failed too many times for $K = 8$ and that the statistics for LiLog with $K = 32$ are marked with a dash (-) because it failed in every single instance.

	min	**avg**	max	std	fail
Log	0.2	**2.1**	12	2.3	0
CC	0.3	**4.6**	23	4.3	0
LiLog	10.0	**25.5**	124	15.1	0
BigM	17.4	**652.2**	9951	1245.2	0

(a) 4 segments.

	min	**avg**	max	std	fail
Log	0.6	**12**	84	11	0
CC	2.6	**81**	570	97	0
LiLog	88.9	**2702**	10000	3024	11
BigM	259.8	**6380**	10000	3742	44

(b) 8 segments.

	min	**avg**	max	std	fail
Log	0.5	**24**	96	18	0
CC	3.9	**351**	3691	517	0
LiLog	848.0	**9863**	10000	982	97

(c) 16 segments.

	min	**avg**	max	std	fail
Log	2.5	**43**	194	39	0
CC	67.5	**1938**	10000	2560	4
LiLog	-	**-**	-	-	100

(d) 32 segments.

Table 3: Solve times for univariate continuous functions [s].

For 4/8 segments the average number of branch-and-bound nodes processed were 587/2591, 964/17276, 2376/262575 and 113327/575189 for Log, CC, LiLog and BigM respectively. For 16/32 segments the average number of branch-and-bound nodes processed were 4428/5287, 28895/80224 and 343187/172903 for Log, CC and LiLog respectively. Of course, these numbers are meaningless for the cases in which most instances reached the time limit, such as LiLog for 16/32 segments. We again see that BigM and LiLog are significantly slower than CC and Log. In addition, the results from Table 3 can be compared with the results in Vielma et al. (2008) to see that both BigM and LiLog are significantly slower than each of the six formulations tested in Vielma et al. (2008) for this set of instances.

We finally note that these negative results only concern formulations BigM and LiLog that Li et al. use to model piecewise linear functions. The results in Table 4 from Li et al. suggest that their main ideas could be useful for other problems such as the unique selection over a finite set of choices or other problems with SOS1 type constraints. In particular, these ideas can be used to modify a standard MIP model for piecewise linear functions to obtain a model with a logarithmic number of binary variables that is similar to the *logarithmic disaggregated convex combination model* (DLog) introduced in Vielma et al. (2008). Preliminary computational results show that the performance of the resulting model is significantly better than LiLog and, although it is still much slower than DLog and Log,

it can be significantly faster than standard models with a linear number of binary variables such as CC.

Acknowledgments

This research was supported by NSF grants CMMI-0522485 and CMMI-0758234 and AFOSR grants FA9550-07-1-0177 and FA9550-08-01-0177. The authors thank Prof. John Hooker, an anonymous associate editor and three anonymous referees for their very thoughtful comments.

References

Bonami, P., L. T. Biegler, A. R. Conn, G. Cornuéjols, I. E. Grossmann, C. D. Laird, J. Lee, A. Lodi, F. Margot, N. Sawaya, A. Waechter. 2008. An algorithmic framework for convex mixed integer nonlinear programs. *Discrete Optimization* **5** 186–204.

Dantzig, G. B. 1960. On the significance of solving linear-programming problems with some integer variables. *Econometrica* **28** 30–44.

Dantzig, G. B. 1963. *Linear Programming and Extensions*. Princeton University Press, Princeton.

Garfinkel, R. S., G. L. Nemhauser. 1972. *Integer Programming*. Wiley.

ILOG. 2008. *ILOG CPLEX User's Manual*. ILOG, S.A.

Jeroslow, R. G., J. K. Lowe. 1984. Modeling with integer variables. *Mathematical Programming Study* **22** 167–184.

Li, H.-L., H.-C. Lu, C.-H. Huang, N.-Z. Hu. 2009. A superior representation method for piecewise linear functions. *Informs Journal on Computing* **21** 314–321.

Nemhauser, G. L., L. A. Wolsey. 1988. *Integer and combinatorial optimization*. Wiley-Interscience.

Padberg, M. 2000. Approximating separable nonlinear functions via mixed zero-one programs. *Operations Research Letters* **27** 1–5.

Vielma, J. P., S. Ahmed, G. L. Nemhauser. 2008. Mixed-integer models for nonseparable piecewise linear optimization: unifying framework and extensions. *Operations Research* **To Appear**. DOI: 10.1287/opre.1090.0721.

Vielma, J. P., G. L. Nemhauser. 2008a. Modeling disjunctive constraints with a logarithmic number of binary variables and constraints. A. Lodi, A. Panconesi, G. Rinaldi, eds., *IPCO*, *Lecture Notes in Computer Science*, vol. 5035. Springer, 199–213.

Vielma, J. P., G. L. Nemhauser. 2008b. Modeling disjunctive constraints with a logarithmic number of binary variables and constraints. *Mathematical Programming* **to appear**. DOI:10.1007/s10107-009-0295-4.