Computational testing of exact mixed knapsack separation for MIP problems

Pasquale Avella, Maurizio Boccia
Dipartimento di Ingegneria
Università del Sannio
Viale Traiano, 82100 Benevento, Italy
E-mail: {avella, maboccia}@unisannio.it

Igor Vasilyev
Institute of System Dynamics and Control Theory
Siberian Branch of Russian Academy of Sciences
Lermontov Srt., 134
664033 Irkutsk, Russia
Email: vil@icc.ru

Abstract

In this paper we study an exact separation algorithm for mixed knapsack sets with the aim of evaluating its effectiveness in a cutting plane algorithm for Mixed-Integer Programming. First proposed by Boyd in the 90's, exact knapsack separation has recently found renewed interest.

In this paper we present a "lightweight" exact separation procedure for mixed knapsack sets and perform a computational experience on a wide set of mixed-integer programming instances from MIPLIB 2003 and "Mittleman" sets. Computational experiments confirm that MIR inequalities are the most important class of valid inequalities form a computational viewpoint. Nevertheless there are several difficult instances where exact separation is able to further raise lower bounds.

1 Introduction

Let $P \subset \mathbb{R}^n$ be a polyhedron and let $\bar{x} \in \mathbb{R}^n$. A *separation algorithm* is said to be *exact* if it either guarantees to return a valid inequality for P cutting off $\bar{x}$ or concludes that $\bar{x} \in P$.

In this paper we study the implementation of an exact separation algorithm for the polyhedron $conv(X^{MK})$, where X^{MK} is the *mixed knapsack set*:

$$X^{MK} = \{(\boldsymbol{y}, s) \in \mathbb{Z}^n_+ \times \mathbb{R}_+ : \ \boldsymbol{a}\boldsymbol{y} - s \leq b, \ \boldsymbol{y} \leq \boldsymbol{u}\}$$

with $\boldsymbol{a}, \boldsymbol{u} \in \mathbb{Q}^n_+$ and $b \in \mathbb{Q}_+$.

The polyhedron $conv(X^{MK})$ was investigated by Marchand and Wolsey [8], who characterized several classes of valid inequalities for $conv(X^{MK})$ and showed that Mixed-Integer Rounding (MIR) inequalities

$$\sum_{j=1}^{n} \left(\lfloor a_j \rfloor + \frac{(f_{a_j} - f_b)^+}{1 - f_b} \right) x_j \leq \lfloor b \rfloor + \frac{s}{1 - f_b}$$

(where $f_d = d - \lfloor d \rfloor$) can be easily derived from X^{MK}. Atamturk [1] investigated the more general case: the polyhedron associated with single constraint mixed-integer set

$$X^{MI} = \{(\boldsymbol{y}, \boldsymbol{x}) \in \mathbb{Z}^n_+ \times \mathbb{R}^p_+ : \ \boldsymbol{a}\boldsymbol{y} + \boldsymbol{g}\boldsymbol{x} \leq b, \ \boldsymbol{y} \leq \boldsymbol{u}\}$$

with $\boldsymbol{g} \in \mathbb{Q}^p$.

First proposed by Boyd [3] in the 90's, exact knapsack separation has recently found renewed interest. Letchford and Kaparis [6] proposed exact separation for the 0/1 knapsack polytope, providing computational results for MIPLIB instances. Fukasawa and Goycoolea [5] implemented an exact separation routine for X^{MI}. The core of their separation procedure is a sophisticated Branch-and-Bound algorithm for the single constraint mixed-integer problem. The authors present computational results for MIPLIB instances, showing that MIR inequalities are the most effective family of valid inequalities for $conv(X^{MI})$ from a computational viewpoint. Avella et al. [2] have shown that exact separation for the 0/1 knapsack polytope is effective in solving the Generalized Assignment problem.

Pursuing this research line, in this paper we attempt to answer the question of whether exact separation for $conv(X^{MK})$ is useful in solving general MIP problems. Computational experiments point out the the lower bounds returned by $conv(X^{MK})$ (which is a relaxation fo $conv(X^{MI})$) are pretty good and for several difficult test instances much better than those provided by separation of MIR inequalities. On the other hand we observe in the paper that the exact separation problem for $conv(X^{MK})$ can be reduced to solving two integer knapsack problems by dynamic programming and this turns into quite reasonable computation times for the separation algorithm.

The remainder of the paper is organized as follows. In Section 2 we present the exact separation procedure for $conv(X^{MK})$. In section 3 we give the implementation details that were required to make exact separation effective for general MIP instances. In Section 4 we present computational results on a wide set of benchmark MIP instances.

2 *MK*-SEP: an exact separation procedure for $conv(X^{MK})$

In this section we outline *MK*-SEP: an exact separation algorithm for the $conv(X^{MK})$. We note that $conv(X^{MK})$ is a relaxation of the polyhedron $conv(X^{MI})$ considered by Fukasawa and Goycoolea [5].

Let $(\bar{\boldsymbol{y}}, \bar{s})$ be a point to separate. Any valid inequality for $conv(X^{MK})$ has the form [8]

$$\boldsymbol{\pi y} - \sigma s \leq \pi_0,$$

where $\boldsymbol{\pi}$, σ and π_0 are nonnegative. The exact separation problem amounts to solving the linear program $SEPLP(X^{MK})$:

$$\max_{(\boldsymbol{\pi},\sigma,\pi_0)} \quad \bar{\boldsymbol{y}}\boldsymbol{\pi} - \bar{s}\sigma - \pi_0$$

$$\boldsymbol{w\pi} - t\sigma \leq \pi_0, \quad (\boldsymbol{w}, t) \in X^{MK} \tag{1}$$

$$\boldsymbol{1\pi} + \sigma = 1 \tag{2}$$

$$\boldsymbol{\pi} \geq 0$$

$$\sigma \geq 0$$

$$\pi_0 \geq 0$$

where (2) is a normalization constraint preventing unboundedness. With this normalization we generate a cutting plane maximizing the ratio between the amount of the violation and the L1-norm. In this way we get a cut maximizing an "approximate steepness" (it is approximate because L1-norm replaces the L2-norm):

$$\frac{\bar{\boldsymbol{y}}\boldsymbol{\pi} - \sigma\bar{s} - \pi_0}{\boldsymbol{1\pi} + \sigma}.$$

2.1 Optimization over X^{MK}

To solve the separation problem, we need an algorithm to solve the mixed knapsack problem $MKNAP$:

$$\max_{(\boldsymbol{y},s)} \left\{ \boldsymbol{cy} + \gamma s : \ (\boldsymbol{y}, s) \in X^{MK} \right\}, \tag{3}$$

where $\boldsymbol{c} \in \mathbb{Q}_+^n$ and $\gamma \in \mathbb{Q}_+$.

We can show that this problem can be reduced to two knapsack problems.

Proposition 1. *The optimal solution of $MKNAP$ is the best between the optimal solutions of the two following knapsack problems:*

1. *$KNAP1$:*

$$\max_{\boldsymbol{y}} \left\{ \boldsymbol{c}\boldsymbol{y} : \ \boldsymbol{a}\boldsymbol{y} \le b, \ \boldsymbol{y} \le \boldsymbol{u}, \ \boldsymbol{y} \in \boldsymbol{Z}_+^n \right\}, \ \textit{and } s = 0;$$

2. *$KNAP2$:*

$$\max_{\boldsymbol{y}} \left\{ (\boldsymbol{c} + \gamma \boldsymbol{a})\boldsymbol{y} - \gamma b : \ \boldsymbol{a} \ge b, \ \boldsymbol{y} \le \boldsymbol{u}, \ \boldsymbol{y} \in \boldsymbol{Z}_+^n \right\}, \ \textit{and } s = \boldsymbol{a}\boldsymbol{y} - b;$$

Proof. It follows from the fact that for any optimal solution $(\boldsymbol{y}^*, s^*)$ either $s^* = 0$ or $s^* = \boldsymbol{a}\boldsymbol{y}^* - b > 0$. □

2.2 Row generation

Since $SEPLP(X^{MK})$ includes a huge number of rows, it requires a *row generation approach*, i.e. an iterative approach where, at each iteration, a *partial separation problem* – including only a subset of the constraints (1) – is considered.

Let $(\bar{\boldsymbol{\pi}}, \bar{\sigma}, \bar{\pi}_0)$ be the optimal solution of the partial separation problem. If all the feasible solutions $(\boldsymbol{w}, t) \in X^{MK}$ satisfy the inequality $\bar{\boldsymbol{\pi}}\boldsymbol{y} - \bar{\sigma}s \le \bar{\pi}_0$, then $(\bar{\pi}, \bar{\sigma}, \bar{\pi}_0)$ is the optimal solution of the original separation problem too and the inequality $\bar{\boldsymbol{\pi}}\boldsymbol{y} - \bar{\sigma}s \le \bar{\pi}_0$ maximizes the violation with respect to $(\bar{\boldsymbol{y}}, \bar{s})$. Otherwise a new inequality is added to the partial separation problem and the procedure iterates. The main steps of the row generation procedure are summarized below.

Solve $SEPLP(X^{MK})$

Step 1 Let $S \subset X^{MK}$ be a subset of the feasible solutions of the mixed knapsack set (S can be initialized to $\emptyset$).

Step 2 Solve the *partial separation* problem $SEPLP(S)$:

$$\begin{aligned} \max \quad & \bar{\boldsymbol{y}}\boldsymbol{\pi} - \bar{s}\sigma - \pi_0 \\ & \boldsymbol{w}\boldsymbol{\pi} - t\sigma \le \pi_0, \quad (\boldsymbol{w}, t) \in S \\ & \boldsymbol{\pi} + \sigma = 1 \\ & \boldsymbol{\pi} \ge 0 \\ & \pi_0 \ge 0 \end{aligned}$$

Let $(\bar{\boldsymbol{\pi}}, \bar{\sigma}, \bar{\pi}_0)$ be its optimal solution.

Step 3 Solve the mixed knapsack problem:

$$(\bar{\boldsymbol{w}}, \bar{t}) \in \operatorname*{Argmax}_{(\boldsymbol{w},t)} \left\{ \bar{\boldsymbol{\pi}}\boldsymbol{w} - \bar{\sigma}t : \ (\boldsymbol{w}, t) \in X^{MK} \right\}$$

Step 4 If $\bar{\boldsymbol{\pi}}\bar{\boldsymbol{w}} - \bar{\sigma}\bar{t} > \bar{\pi}_0$ then set $S = S \cup \{(\bar{\boldsymbol{w}}, \bar{t})\}$ and goto **Step 1**.

Step 5 $(\bar{\boldsymbol{\pi}}, \bar{\sigma}, \bar{\pi}_0)$ is the optimal solution of $SEPLP(X^{MK})$ and the inequality $\bar{\boldsymbol{\pi}}\boldsymbol{y} - \bar{\sigma}s \le \bar{\pi}_0$ is valid for $conv(X^{MK})$.

3 Implementation details

When embedded into a cutting plane algorithm to solve MIP problems, the separation procedure is applied to every formulation row (equality constraints are split into two inequalities), which, without loss of generality, form a mixed-integer set:

$$X^{MI} = \{(\boldsymbol{y}, \boldsymbol{x}) \in \mathbb{Z}_+^n \times \mathbb{R}_+^p : \ \boldsymbol{ay} + \boldsymbol{gx} \le b, \ \boldsymbol{y} \le \boldsymbol{u}\}.$$

The overall procedure consists of the following steps:

i) Bound substitution, which takes into account simple and variable bounds on continuous variables.

ii) Preprocessing, which relax the mixed integer set to the mixed knapsack set with integer coefficients.

iii) *MK*-SEP over the variables which are not at their bounds in the current LP solution.

iv) Sequential lifting to get globally valid cuts.

The integer knapsack problems arising in the Step 3 of the row generation procedure $SEPLP(X^{MK})$ and in the lifting phase are both solved by the dynamic programming algorithm of Pisinger [7].

The following subsections will give the implementation details for all the steps of the procedure but *iii*), described above in section 2.

3.1 Bound substitution

Consider the mixed-integer set X^{MI}. The MIP formulation can also contain some additional bounds on continuous variables. Let us assume that for each $j \in \{1, \dots, p\}$ simple $l_j \le x_j \le v_j$ and variable bounds $\tilde{l}_j y_i \le x_j \le \tilde{v}_j y_k$ on the continuous variables are defined. At least one of these bounds is finite since $l_j = 0$ by the definition X^{MI}.

Bound substitution [8, 10] consists of replacing some nonnegativity conditions on continuous variables by their simple or variable bounds. It is done heuristically by performing one of the following substitutions:

$$\begin{aligned} &x_j = l_j + x'_j \\ &x_j = v_j - x'_j \\ &x_j = \tilde{l}_j y_i + x'_j \\ &w_j = \tilde{v}_j y_k - x'_j \end{aligned}$$

To select which bound is used to be replace, Wolter [10], on the base of an extensive set of computational experiments, concluded that the most effective heuristics consists of the following.

Let $(\bar{\boldsymbol{y}}, \bar{\boldsymbol{x}})$ be a current fractional solution. For substitution, the bound with smallest slack is selected. That is, let

$$\mu = \min\{\bar{x}_j - l_j, \ v_j - \bar{x}_j, \ \bar{x}_j - \tilde{l}_j \bar{y}_i, \ \tilde{v}_j \bar{y}_k - \bar{x}_j\}.$$

Then, the substitution is

$$x_j = \begin{cases} l_j + x'_j & \text{if } \mu = x_j - l_j \\ v_j - x'_j & \text{if } \mu = v_j - \bar{x}_j \\ \tilde{l}_j y_i + x'_j & \text{if } \mu = \bar{x}_j - \tilde{l}_j \\ \tilde{v}_j y_k - x'_j & \text{if } \mu = \tilde{v}_j \bar{y}_k - \bar{x}_j \end{cases}$$

In case of multiple choices the preference is given to the lower and variable bounds.

3.2 Preprocessing

Let

$$\sum_{i\in I} a'_i y_i + \sum_{j\in P} g'_j x'_j \le b', \tag{4}$$

with $0 \le y_i \le u_i \ \forall j \in I$ and $x'_j \ge 0 \ \forall j \in P$, be the mixed-integer inequality resulting after bound substitution. The inequality (4) is preprocessed by performing the following operations:

a) As noted in [1], all the continuous variables with nonnegative coefficients can be discarded. All the continuous variables with negative coefficients are aggregated in one continuous variable s:

$$s = -\sum_{j\in P^-} g'_j x'_j,$$

where $P^- = \{j \in P : \ g'_j < 0\}$.

b) All the integer variables with negative coefficients are complemented, i.e. we pose

$$y_j = \begin{cases} u_j - y'_j & \text{if } a'_j < 0 \\ y'_j & \text{otherwise} \end{cases}$$

It results in the mixed knapsack set defined by the inequality:

$$\sum_{i\in I} a''_i y'_i - s \le b''. \tag{5}$$

c) Dynamic programming requires that all the knapsack coefficients are integers, so the coefficients of the integer variables of the base inequality (5) must be converted into suitably small integers before running exact separation. The problem of converting a set of coefficients into integers can be formulated as a mixed-integer programming problem as done in [2] for the generalized assignment problem. But this problem can be very hard to solve in the general case. In order to get faster computing times, we adopt a brute-force approach which consists of enumerating all the $q \in \mathbb{N}$ in the range $1 \div 10^4$, stopping when $qb'' - \lfloor qb'' \rfloor \le \varepsilon$ and $qa''_j - \lfloor qa''_j \rfloor \le \varepsilon$ for each $j \in I$. In our experiments we set $\varepsilon = 10^{-5}$. If the procedure fails, we discard the inequality since too large coefficients may cause numerical problems.

3.3 Lifting

The number of row generation iterations in *MK*-SEP grows exponentially with the size of the mixed knapsack set, so it is crucial to run exact separation on some small subset of variables. Let $(\bar{\boldsymbol{y}}, \bar{s})$ be the current fractional solution of LP relaxation. First we separate this point from $conv(X^{MK}_{\bar{\boldsymbol{y}}})$, where the variables are fixed if they are at their bounds, i.e.

$$X^{MK}_{\bar{\boldsymbol{y}}} = \left\{(\boldsymbol{y}, s) \in X^{MK} : \ y_i = 0 \text{ if } \bar{y}_i = 0, \ y_i = u_i \text{ if } \bar{y}_i = u_i \ \forall i \in I\right\}.$$

Then, the sequential lifting [9] procedure is used to get globally valid cuts.

It is well-known that the resulting inequalities depend on the order in which the variables are lifted, i.e. on the lifting sequence. For a given variable, a better lifting coefficient is obtained if the variable is lifted earlier in the sequence. In our experiments, the variables at upper bound are lifted first (*down-lifting*) and then we lift the variables at the lower bounds (*up-lifting*). To define the lifting sequence inside each group we consider the reduced costs in the current LP solution. Variables with a smaller reduced cost are lifted first.

Technically, the lifting is performed using the following elementary rules. Let the ineqality

$$\sum_{i=2}^{n} \pi_i y_i - \sigma s \leq \pi_0 \tag{6}$$

be given and let us denote

$$\zeta(q_1) = \max_{(\bar{\boldsymbol{y}},s)} \left\{ \sum_{i=2}^{n} \pi_i y_i - \sigma s : \ \boldsymbol{y} \in X^{MK}, \ y_1 = q_1 \right\}.$$

Down-lifting If (6) is valid for $X^{MK} \cap \{\boldsymbol{y} : \ y_1 = u_1\}$, then

$$\sum_{i=1}^{n} \pi_i y_i - \sigma s \leq \pi_0 + \pi_1 u_1$$

is valid for X^{MK}, where

$$\pi_1 = \max_{q_1} \left\{ \frac{\zeta(q_1) - \pi_0}{u_1 - q_1} : \ q_1 = 0, \ldots, u_1 - 1 \right\}.$$

Up-lifting If (6) is valid for $X^{MK} \cap \{\boldsymbol{y} : \ y_1 = 0\}$, then

$$\sum_{i=1}^{n} \pi_i y_i - \sigma s \leq \pi_0$$

is valid for X^{MK}, where

$$\pi_1 = \min_{q_1} \left\{ \frac{\zeta(q_1) - \pi_0}{u_1 - q_1} : \ q_1 = 1, \ldots, u_1 \right\}.$$

Thus, to lift one variable we need to solve a series of mixed knapsack problems, which can be solved as described in subsection 2.1

4 Computational results

Computational experiments were carried out on a 64bit Pentium Quad-core 2.6 GHz processor with 4 Gb RAM. The LP solver was Xpress 2007B [4].

The test bed consists of all the MIPLIB 2003 mixed-integer instances and of the "Mittleman" instances *bc1, bienst1, bienst2, binkar10_1, dano3-4, dano3-5*. We set a limit of 300 CPU secs for the time spent in separation.

To evaluate the usefulness of exact mixed knapsack separation in solving MIP problems, we compare the results of *MK*-SEP with those returned by the MIR separation procedure of Wolter [10], embedded into the mixed-integer programming solver SCIP [11]. We set SCIP parameters to perfom Wolter's procedure on single rows, i.e. to forbid constraint aggregation. Separation of Lifted Cover inequalities is enabled too. For simplicity of comparison, separation routines run on the original (i.e. not preprocessed) instances. *MK*-SEP runs on the formulation returned by SCIP after the addition of MIR inequalities, so the computation time for *MK*-SEP must be added to that required by the separation of the MIR inequalities.

In table 1, LP and BUB show the value of the LP-relaxation and the best known upper bound, respectively. Columns "SCIP LB", "SCIP Gap" and "SCIP Time" report on the lower bound returned by the SCIP separation routine, the % of gap closed and the computation time, respectively. Correspondingly, columns "*MK*-SEP LB", "*MK*-SEP Gap" and "*MK*-SEP Time" report on the lower bound yielded by *MK*-SEP, the % of closed gap and the computation time. The instances where *MK*-SEP significantly outperformed MIR separation in terms of closed gap, are marked in boldface in the first column.

Computational experience confirms Fukasawa and Goycoolea [5] outcomes: MIR inequalities play a crucial role among the valid inequalities for $conv(MK)$ in terms of computational effectiveness.

Nevertheless there are several difficult instances where *MK*-SEP significantly raised the lower bounds returned by MIR separation. Particularly, we observe that for $aflow30a$, $aflow40b$, $bienst1$ and $bienst2$, *MK*-SEP significantly closed the gap whereas MIR separation procedure could not find any violated inequality. Other "successful" instances where *MK*-SEP led to remarkable improvements, are *mkc*, *tr12-30*, *vpm*2 and *binkar10_1*. An increased computation time spent in separation is the price to be paid to get better bounds, so we can conclude that exact separation may be useful for hard instances, when the emphasis is on the quality of the solution more than on computation time.

References

[1] A. Atamtürk. On the facets of the mixed–integer knapsack polyhedron. *Mathematical Programming*, 98:145–175, 2003.

[2] P. Avella, M. Boccia, and I. Vasilyev. A computational study of exact knapsack separation for the generalized assignment problem. *Technical report. Available at* $http://www.optimization-online.org/DB_FILE/2007/04/1648.pdf$, 2007.

[3] E.A. Boyd. Generating fenchel cutting planes for knapsack polyhedra. *SIAM Journal of Optimization*, 3:734–750, 1993.

[4] Dash Optimization. *Xpress-Optimizer Reference Manual, Release 18*, 2007.

[5] R. Fukasawa and M. Goycoolea. On the exact separation of mixed integer knapsack cuts. In Springer, editor, *Proceedings of the 2007 Integer Programming and Combinatorial Optimization conference*, To appear.

[6] K. Kaparis, , and A.N. Letchford. Separation algorithms for 0-1 knapsack polytopes. *In preparation. Available at* $http://www.lancs.ac.uk/staff/letchfoa/publications.htm$, 2007.

[7] H. Kellerer, U. Pferschy, and D. Pisinger. *Knapsack Problems.* Springer, 2004.

[8] H. Marchand and L. A. Wolsey. The 0–1 knapsack problem with a single continuous variable. *Mathematical Programming*, 85:15–33, 1999.

[9] L.A. Wolsey. Facets and strong valid inequalities for integer programs. *Operations Research*, 24:367–372, 1976.

[10] K. Wolter. *Implementation of Cutting Plane Separators for Mixed Integer Programs.* PhD thesis, Technische Universitat Berlin.

[11] ZIB - Berlin. *available at scip.zib.de*, 2007.

Name	LP	BUB	SCIP LB	SCIP %Gap	SCIP Time	*MK*-SEP LB	*MK*-SEP % Gap	*MK*-SEP Time
10teams	917.00	924.00	917.00	0.00	5	917	0.00	1
a1c1s1	997.53	11503.40	997.53	0.0	2	997.53	0.00	2
aflow30a	983.16	1158.00	983.16	0.0	1	1053.29	40.11	9
aflow40b	1005.50	1168.00	1005.50	0.0	4	1058.32	32.50	10
arki001	7579599.81	7580813.05	7579599.81	0.00	4	7579599.81	0.00	1
atlanta-ip	81.24	90.01	81.25	0.11	548	82.46	13.91	300
dano3mip	576.23	687.73	576.23	0.00	1428	576.23	0.00	8
danoint	62.63	65.67	62.63	0.00	10	62.66	0.88	4
fiber	156082.50	405935.18	385094.10	91.66	3	390493.82	93.82	9
fixnet6	1200.88	3983.00	3192.04	71.57	1	3442.60	80.58	195
gesa2	25476489	25777537	25691081	71.28	1	25701859	74.86	4
gesa2-o	25476489	25778001	25476489	0.0	1	25588105	37.02	6
glass4	800002400	1200012600	800002400	0.00	3	800002400	0.00	0
liu	346.00	1138.00	385.00	4.92	14	385.00	4.92	9
markshare1	0.00	1.00	0.00	0.00	1	0.00	0.00	43
markshare2	0.00	1.00	0.00	0.00	1	0.00	0.00	26
mas74	10482.79	11801.19	10482.79	0.00	1	10482.79	0.00	0
mas76	38893.90	40005.05	38901.02	0.64	1	38901.02	0.64	0
misc07	1415.00	2810.00	1415.00	0.00	1	1415	0.00	1
mkc	-611.85	-563.85	-607.18	9.73	56	-605.83	12.54	55
modglob	20430947.60	20740500.00	20430947.60	0.0	1	20431515.90	0.18	9
msc98-ip	19520966.2	19839497.01	19538746.75	5.58	1500	19559084.16	11.97	169
net12	17.25	214.00	31.55	7.27	290	32.08	7.54	106
nsrand-ipx	48880.00	51200.00	49851.43	41.87	123	49877.59	43.00	62
roll3000	11097.13	12890.00	12072.71	54.41	20	12073.49	54.46	25
swath	334.50	467.40	334.50	0.00	6	334.5	0.00	9
timtab1	28694.00	764772.00	195605.34	22.68	1	229628.78	27.30	3
timtab2	83592.00	1096557.00	250004.21	16.43	1	270295.07	18.43	7
tr12-30	14210.43	130676.00	18124.17	3.36	1	84403.46	60.27	8
vpm2	9.89	13.75	10.40	13.21	1	11.21	33.94	2
bienst1	11.72	46.75	11.72	0.0	1	14.01	6.54	2
bienst2	11.72	54.60	11.72	0.0	1	14.88	7.41	3
dano3-4	576.23	576.43	576.23	0.0	297	576.23	0.00	3
dano3-5	576.23	576.92	576.23	0.0	308	576.23	0.00	4
rgn	48.80	82.20	68.00	57.49	0	68.00	57.49	1

Table 1: Computational results on mixed-integer programming problems