SOLVING MIXED INTEGER BILINEAR PROBLEMS USING MILP FORMULATIONS

AKSHAY GUPTE[†§], SHABBIR AHMED[†§], MYUN SEOK CHEON[‡], AND SANTANU DEY[†§]

Abstract. In this paper, we examine a mixed integer linear programming (MILP) reformulation for mixed integer bilinear problems where each bilinear term involves the product of a nonnegative integer variable and a nonnegative continuous variable. This reformulation is obtained by first replacing a general integer variable with its binary expansion and then using McCormick envelopes to linearize the resulting product of continuous and binary variables. We present the convex hull of the underlying mixed integer linear set. The effectiveness of this reformulation and associated facet-defining inequalities are computationally evaluated on five classes of instances.

Key words. bilinear problems, McCormick envelopes, binary expansion, cutting planes, mixed integer programming

AMS subject classifications.

1. Introduction.

Consider the mixed integer bilinear program given as

$$\begin{aligned} \min \quad & \hat{x}^T Q_0 \hat{y} + f_0^T \hat{x} + g_0^T \hat{y} \\ \text{s.t.} \quad & A\hat{x} + G\hat{y} \le h_0 \\ & \hat{x}^T Q_t \hat{y} + f_t^T \hat{x} + g_t^T \hat{y} \le h_t, \qquad t = 1, \dots, p, \\ & 0 \le \hat{x} \le \hat{a} \\ & 0 \le \hat{y} \le \hat{b},\ \hat{y} \in \mathbb{Z}^n, \end{aligned} \tag{BLP1}$$

where $Q_t \in \Re^{m\times n}, f_t \in \Re^m, g_t \in \Re^n$, for $t = 0, \dots, p$, and $\hat{b} \in \Re^n_+, \hat{a} \in \Re^m_+, A \in \Re^{q\times m}, G \in \Re^{q\times n}, h_0 \in \Re^q, h_t \in \Re$ for $t = 1, \dots, p$. In the above formulation, every bilinear term is a product of one continuous and one integer variable.

Although formulation (BLP1) may seem restrictive, it can be used to solve approximations of a general class of bilinear problems. Consider a problem which along with the structure of (BLP1) also has bilinearities between continuous variables. Then, we may think of choosing a suitable subset of continuous variables that appear in bilinear terms with other continuous variables and *discretizing* the variables (cf. [10, 28]) in this chosen subset. This gives us an approximation of the form (BLP1) for the original problem since a subset of the continuous variables are now restricted to take only integer values.

Continuous and mixed integer bilinear problems find many applications [19, 25, 26, 30, 22, 31, 12] and have been fairly well studied in literature. A common solution methodology is to construct polyhedral relaxations using envelopes of each bilinear term [3] within a spatial branch-and-bound framework [16]. Tighter relaxations can be constructed using convex envelopes of the entire bilinear function [34, 33]. There also exist specialized branch-and-bound algorithms that contract the feasible region at each node of the search tree [37]. The reformulation linearization technique (RLT) has been applied to the continuous bilinear problem [32] and extended to the mixed $\{0, 1\}$ problem with a bilinear objective function [1]. The branch-and-cut algorithm in [5] uses four classes of RLT inequalities to solve a pooling problem. Convex relaxations based on semidefinite programming have been studied [4]. Another type of relaxation

[†]School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, GA.
[‡]ExxonMobil Research and Engineering Company, Annandale, NJ.
[§]Research supported by ExxonMobil Research and Engineering Company.

is based on Lagrangian duals [2, 15, 9]. One may also obtain piecewise linear relaxations by dividing the intervals of one or both the variables in a bilinear term into sufficient number of pieces and constructing envelopes in each of these sub-intervals [20]. Branching strategies [8] and heuristics [13] have also been developed.

The main objective of this study is to seek MILP-based solution approaches using polyhedral study of single term bilinear sets. A MILP-based methodology can be particularly advantageous if, besides the nonconvexities of bilinear terms, the integrality constraints on variables are "hard" to satisfy. Since considerable progress has been made in algorithms and state-of-the-art solvers for MILP, these hard constraints can be better dealt with through a MILP solution procedure. The proposed approach differs from previous work in that our focus is on solving (BLP1) as a MILP whereas the existing methods are aimed at obtaining stronger relaxations, branching techniques, and heuristics within a spatial branch-and-bound framework for solving (BLP1). Hence, our first step is to use binary expansion of general integer variables to obtain an extended reformulation. Although the use of binary expansions is known to be inefficient for general MILPs [27], in our case, it gives us an exact MILP reformulation of (BLP1). On the contrary, the use of McCormick envelopes [24] produces a relaxation of the single bilinear term. Binary expansions have been proposed by [19, 18] for reformulating mixed integer bilinear sets. Henry [21] also studied binary reformulations of discrete functions and empirically compared them to other approaches. However, to the best of our knowledge, there has been no study of the polyhedral structure of the sets arising in the context of mixed integer bilinear programs due to such binary reformulations. Our contribution is to obtain complete descriptions of the convex hulls of these reformulated single term bilinear sets and use them in a branch-and-cut algorithm for solving the reformulated MILP.

The rest of the paper is organized as follows. In section 2, we present MILP formulations for (BLP1) and study their relative strengths. In section 3, the single term mixed integer bilinear set is studied and facet-defining inequalities of its convex hull are derived. In section 4, we present some computational results to demonstrate the effectiveness of our cuts. Section 5 concludes the paper with a discussion.

We use the following notation in this paper: $\mathrm{conv}(\cdot)$ denotes the convex hull of a set and $\mathrm{relax}(\cdot)$ denotes the continuous relaxation of a set obtained by dropping the integrality restrictions on its variables. Given a set $\mathcal{X}$ in the (x, y)-space, we define $\mathrm{Proj}_x(\mathcal{X}) := \{x \colon \exists y \text{ s.t. } (x, y) \in \mathcal{X}\}$, as the projection of $\mathcal{X}$ onto the x-space. For ease of notation, we sometimes represent a singleton $\{i\}$ simply as i. $\Re_+$ is the set of nonnegative reals, and $\mathbb{Z}_+$ and $\mathbb{Z}_{++}$ are the set of nonnegative and positive integers, respectively.

2. MILP formulations. Let us linearize the objective function and constraints in (BLP1) by introducing new variables $\hat{w}_{ij} = \hat{x}_l \hat{y}_j$, for all $l \in \{1, ..., m\},\ j \in \{1, ..., n\}$. This gives us the reformulation (BLP) in an extended space. Note that in this reformulation we have reduced all the bilinearities to the constraints $\hat{w}_{lj} = \hat{x}_l \hat{y}_j$, for all $l \in \{1, ..., m\}$, and $j \in \{1, ..., n\}$. In the absence of these bilinearities, the problem is a MILP.

$$
\begin{aligned}
\min \quad & \sum_{l=1}^{m}\sum_{j=1}^{n} Q_{0lj}\hat{w}_{lj} + f_0^T\hat{x} + g_0^T\hat{y} \\
\text{s.t.} \quad & A\hat{x} + G\hat{y} \le h_0 \\
& \sum_{l=1}^{m}\sum_{j=1}^{n} Q_{tlj}\hat{w}_{lj} + f_t^T\hat{x} + g_t^T\hat{y} \le h_t, \qquad t = 1,\dots,p, \\
& \hat{w}_{lj} = \hat{x}_l\hat{y}_j, \quad l = 1,\dots,m,\ j = 1,\dots,n \\
& 0 \le \hat{x} \le \hat{a}, \\
& 0 \le \hat{y} \le \hat{b},\ \hat{y} \in \mathbb{Z}^n.
\end{aligned}
\tag{BLP}
$$

Solving (BLP) using MILP techniques is possible only if we obtain MILP reformulations of the bilinear terms. To do this, it suffices to study reformulations of each bilinear term separately.

2.1. Reformulations of single term mixed integer bilinear set. For bounded continuous and general integer variables x and y, respectively, and a bilinear variable $w = xy$, consider the mixed integer bilinear set:

$$
\mathcal{P} := \{(x, y, w) \in \Re_+ \times \mathbb{Z}_+ \times \Re \colon w = xy, x \le a, y \le b\}. \tag{2.1}
$$

We assume that $b \ge 1$ is a positive integer and $a > 0$ is a positive real. A standard approach adopted for linearizing the bilinear terms is to replace each term by its convex and concave envelopes, also called the McCormick envelopes [24]. Performing this operation on $\mathcal{P}$ gives us the following set

$$
\mathcal{M} := \{(x, y, w) \in \Re \times \mathbb{Z} \times \Re \colon w \ge 0, w \le ay, w \le bx, w \ge bx + ay - ab\}. \tag{2.2}
$$

Another idea is to use a unary or binary expansion of the integer variable y. Let z be the new binary vector used in such an expansion. Using v_i to model the product xz_i for each i, we obtain the sets

$$
\begin{aligned}
\mathcal{U} := \Big\{ & (x, y, w, z, v) \in \Re \times \mathbb{Z} \times \Re \times \{0,1\}^b \times \Re^b \colon y = \sum_{i=1}^{b} iz_i,\ \sum_{i=1}^{b} z_i \le 1, w = \sum_{i=1}^{b} iv_i, \\
& v_i \ge 0, v_i \le az_i, v_i \le x, v_i \ge x + az_i - a, \forall i \in \{1,\dots,b\} \Big\},
\end{aligned}
\tag{2.3}
$$

for unary expansion and

$$
\begin{aligned}
\mathcal{B} := \Big\{ & (x, y, w, z, v) \in \Re \times \mathbb{Z} \times \Re \times \{0,1\}^k \times \Re^k \colon y = \sum_{i=1}^{k} 2^{i-1} z_i \ \le b,\ w = \sum_{i=1}^{k} 2^{i-1} v_i, \\
& v_i \ge 0, v_i \le az_i, v_i \le x, v_i \ge x + az_i - a, \forall i \in \{1,\dots,k\} \Big\},
\end{aligned}
\tag{2.4}
$$

for binary expansion, where $k = \lfloor \log_2 b \rfloor + 1$. The lower and upper bounds on x and y are implied in each of the above three formulations. Note that for $\mathcal{U}$ and $\mathcal{B}$, the linearization of $v_i = xz_i$ is exact because $z_i \in \{0, 1\}$, for all i. We first compare the strengths of these sets in the following result.

PROPOSITION 2.1. *$\mathcal{P} = \mathrm{Proj}_{x,y,w}(\mathcal{U}) = \mathrm{Proj}_{x,y,w}(\mathcal{B})$ and $\mathcal{P} \subseteq \mathcal{M}$. The set $\mathcal{M} \setminus \mathcal{P}$ is nonempty if and only if $b \ge 2$.*

Proof. By construction, it follows that $\mathcal{P} \subseteq \mathcal{M}$, $\mathcal{P} \subseteq \mathrm{Proj}_{x,y,w}(\mathcal{U})$, and $\mathcal{P} \subseteq \mathrm{Proj}_{x,y,w}(\mathcal{B})$. We prove the reverse inclusion only for $\mathcal{U}$. The proof for $\mathcal{B}$ is similar. Consider any feasible point $(x, y, w, z, v) \in \mathcal{U}$. Since $y \in \mathbb{Z}_+$, there are two cases -

1. $y = 0$. Then $z_i = 0$, for all $i \in \{1, \dots, b\}$, which implies that $v_i = 0$, for all $i \in \{1, \dots, b\}$. Therefore $w = yx$.
2. $y > 0$. Then $z_y = 1$ and $z_i = 0, i \in \{1, ..., b\} \setminus \{y\}$. Therefore, $v_i = 0, i \in \{1, ..., b\} \setminus \{y\}$ and $v_y = x$. Hence, $w = yv_y = yx$.

Thus, in both the cases, $(x, y, w) \in \mathcal{P}$.

For $b = 1$, it is straightforward to verify that $\mathcal{M} = \mathcal{P}$. For $b > 1$, observe that $(\frac{a}{b}, 1, a) \in \mathcal{M} \setminus \mathcal{P}$. ∎

The set $\mathcal{M}$ is a strong relaxation of $\mathcal{P}$. In particular, the convex hulls of $\mathcal{M}$ and $\mathcal{P}$ are exactly the same and equal to the linear programming (LP) relaxation of $\mathcal{M}$, i.e. $\mathrm{conv}(\mathcal{P}) = \mathrm{conv}(\mathcal{M}) = \mathrm{relax}(\mathcal{M})$. This follows from the earlier work on McCormick envelopes of a bilinear term [3, 24] and observing that $y \in \{0, b\}$ at extreme points of $\mathrm{relax}(\mathcal{M})$. The sets $\mathcal{U}$ and $\mathcal{B}$ are the two most commonly used reformulations for $\mathcal{P}$. They both add extra, albeit a different number of, binary and continuous variables. The remaining question is how strong the LP relaxations of $\mathcal{U}$ and $\mathcal{B}$ are. Towards this end, we first show that the LP relaxations of $\mathcal{U}$ and $\mathcal{B}$ are generally weaker than that of $\mathcal{M}$.

Proposition 2.2.

1. $\mathrm{relax}(\mathcal{M}) \subseteq \mathrm{Proj}_{x,y,w}(\mathrm{relax}(\mathcal{B}))$ *with strict inclusion if and only if* $b \neq 2^\gamma - 1$, *for any positive integer* γ.
2. $\mathrm{relax}(\mathcal{M}) \subseteq \mathrm{Proj}_{x,y,w}(\mathrm{relax}(\mathcal{U}))$ *and the inclusion is strict if and only if* $b \geq 2$.

Proof. From Proposition 2.1 we have that $\mathcal{P} = \mathrm{Proj}_{x,y,w}(\mathcal{B})$. This implies $\mathcal{P} \subseteq \mathrm{Proj}_{x,y,w}(\mathrm{relax}(\mathcal{B}))$ and since $\mathrm{Proj}_{x,y,w}(\mathrm{relax}(\mathcal{B}))$ is a convex set, we obtain that $\mathrm{conv}(\mathcal{P}) \subseteq \mathrm{Proj}_{x,y,w}(\mathrm{relax}(\mathcal{B}))$. Now, in the above discussion we argued that $\mathrm{relax}(\mathcal{M}) = \mathrm{conv}(\mathcal{P})$ which implies the inclusion $\mathrm{relax}(\mathcal{M}) \subseteq \mathrm{Proj}_{x,y,w}(\mathrm{relax}(\mathcal{B}))$.

Next we verify that the inclusion $\mathrm{relax}(\mathcal{M}) \subseteq \mathrm{Proj}_{x,y,w}(\mathrm{relax}(\mathcal{B}))$ is strict if and only if $b \neq 2^\gamma - 1$, for any $\gamma \in \mathbb{Z}_+$. First suppose that $b \neq 2^\gamma - 1$, for all $\gamma \in \mathbb{Z}_+$. Recall that $k = \lfloor \log_2 b \rfloor + 1$. Take a point (x, y, w, z, v) constructed as follows

$$z_i = \frac{1}{k},\ v_i = az_i, \qquad \forall i \in \{1, \dots, k\}$$
$$y = \frac{2^k - 1}{k},\ w = ay,\ x = \frac{a}{k}.$$

It is easily verified that this point satisfies the linear constraints of $\mathrm{relax}(\mathcal{B})$. Since for $k \geq 2$ we have that $\frac{2^k-1}{k} < 2^{k-1} \leq b$, the upper bound on y is also satisfied. Hence this point belongs to $\mathrm{relax}(\mathcal{B})$. However, because $b \neq 2^\gamma - 1$ and $k = \lfloor \log_2 b \rfloor + 1$, it follows that $b < 2^k - 1$. Therefore $w > bx$ and the chosen point does not belong to $\mathrm{relax}(\mathcal{M})$.

Now suppose that $b = 2^\gamma - 1$, for some $\gamma \in \mathbb{Z}_{++}$. Since $k = \lfloor \log_2 b \rfloor + 1$, we have that $b = 2^k - 1$. Consider any point $(x, y, w, z, v) \in \mathrm{relax}(\mathcal{B})$. Since $v_i \leq x$ for all $i \in \{1, \dots, k\}$,

$$\begin{aligned} w = \sum_{i=1}^{k} 2^{i-1} v_i &\leq \sum_{i=1}^{k} 2^{i-1} x \\ &= (2^k - 1)x \\ &= bx. \end{aligned}$$

Similarly, $v_i \leq az_i$ and $v_i \geq x + az_i - a$ for all $i \in \{1, \ldots, k\}$, imply that $w \leq ay$ and $w \geq bx + ay - ab$, respectively. Hence, the point (x, y, w) belongs to relax($\mathcal{M}$).

The proof for the inclusion relax($\mathcal{M}$) $\subseteq \text{Proj}_{x,y,w}$(relax($\mathcal{U}$)) is similar to that for relax($\mathcal{M}$) $\subseteq \text{Proj}_{x,y,w}$(relax($\mathcal{B}$)). Observe that for $b = 1$, the two sets relax($\mathcal{M}$) and $\text{Proj}_{x,y,w}$(relax($\mathcal{U}$)) are exactly the same because $y = z_1$ and $w = v_1$.

Now suppose that $b \geq 2$ is some positive integer. Construct a point $(x, y, w, z, v) \in$ relax($\mathcal{U}$) as follows

$$z_i = \frac{1}{b},\ v_i = az_i, \qquad \forall i \in \{1, \ldots, b\}$$
$$y = \frac{b+1}{2},\ w = ay,\ x = \frac{a}{b}.$$

Thus, $w = \frac{a(b+1)}{2}$. For $b > 1$, it follows that $w > a = bx$ and hence this point does not belong to relax($\mathcal{M}$). ◻

Now we compare the relaxations of $\mathcal{B}$ and $\mathcal{U}$. We first observe that for $b = 2$, the two sets $\mathcal{B}$ and $\mathcal{U}$ are almost the same except that $\mathcal{U}$ has an additional constraint $z_1 + z_2 \leq 1$, thus giving us relax($\mathcal{U}$) $\subset$ relax($\mathcal{B}$). Proposition 2.2 implies that if $b = 2^\gamma - 1$ for some integer $\gamma \geq 2$, then $\text{Proj}_{x,y,w}$(relax($\mathcal{B}$)) = relax($\mathcal{M}$) = conv($\mathcal{M}$) $\supset$ $\text{Proj}_{x,y,w}$(relax($\mathcal{U}$)). Hence the relaxation of $\mathcal{B}$ is stronger (in the original (x, y, w)-space) than the relaxation of $\mathcal{U}$. However, this dominance does not always hold true.

PROPOSITION 2.3. *Let $b \geq 3$ be an integer such that $b \neq 2^\gamma - 1$, for any $\gamma \in \mathbb{Z}_{++}$. Then in general,*

1. $\text{Proj}_{x,y,w}$(relax($\mathcal{B}$)) $\setminus$ $\text{Proj}_{x,y,w}$(relax($\mathcal{U}$)) $\neq \emptyset$.
2. $\text{Proj}_{x,y,w}$(relax($\mathcal{U}$)) $\setminus$ $\text{Proj}_{x,y,w}$(relax($\mathcal{B}$)) $\neq \emptyset$.

Proof. Consider the point $(\epsilon/B, b, 0)$ where $B = 2^k - 1$ and $\epsilon \in (0, a(B - b)]$. Since $b \neq 2^\gamma - 1$, for any $\gamma \in \mathbb{Z}_{++}$, and $B = 2^k - 1$, it must be that $b < B$ and hence the choice of ϵ is well defined. We will first show that there exists a $z \in [0, 1]^k$ such that $(\epsilon/B, b, 0, z, 0) \in$ relax($\mathcal{B}$). Equivalently, we have to show that there exists a $z \in [0, 1]^k$ such that

$$\sum_{i=1}^{k} 2^{i-1} z_i = b$$
$$0 \leq az_i \leq a - \frac{\epsilon}{B} \qquad \forall i \in \{1, \ldots, k\}.$$

Consider the hypercube $[0, 1 - \epsilon/(aB)]^k$. Then,

$$\begin{aligned}
\max \Big\{ \sum_{i=1}^{k} 2^{i-1} \zeta_i \colon \zeta \in [0, 1 - \epsilon/(aB)]^k \Big\} &= \left(1 - \frac{\epsilon}{aB}\right) \sum_{i=1}^{k} 2^{i-1} \\
&= \left(1 - \frac{\epsilon}{aB}\right) (2^k - 1) \\
&= B - \frac{\epsilon}{a} \\
&\geq b,
\end{aligned}$$

where the last inequality follows from the construction of ϵ. Clearly the minimum of the expression $\sum_{i=1}^{k} 2^{i-1} \zeta_i$ over $[0, 1 - \epsilon/(aB)]^k$ is 0. Then, by continuity of $\sum_{i=1}^{k} 2^{i-1} \zeta_i$, there must exist some $\hat{z} \in [0, 1 - \epsilon/(aB)]^k$ such that $\sum_{i=1}^{k} 2^{i-1} \hat{z}_i = b$.

Hence the point $(\epsilon/B, b, 0, \hat{z}, 0) \in \mathrm{relax}(\mathcal{B})$ and consequently, $(\epsilon/B, b, 0) \in \mathrm{Proj}_{x,y,w}(\mathrm{relax}(\mathcal{B}))$. To show that $(\epsilon/B, b, 0) \notin \mathrm{Proj}_{x,y,w}(\mathrm{relax}(\mathcal{U}))$, suppose for the sake of contradiction that there exist some $(\bar{z}, \bar{v})$ such that $(\epsilon/B, b, 0, \bar{z}, \bar{v}) \in \mathrm{relax}(\mathcal{U})$. Then, $w = 0$ implies that $\bar{v}_i = 0$, $\forall i = 1, \dots, b$, and $y = b$ implies that $\bar{z}_b = 1$. On the other hand,

$$\begin{aligned} x + a\bar{z}_b - a &= \frac{\epsilon}{B} \\ &> 0 \\ &= \bar{v}_b, \end{aligned}$$

a contradiction to the feasibility of $(\epsilon/B, b, 0, \bar{z}, \bar{v})$.

Finally, we construct a point $(x, y, w) \in \mathrm{Proj}_{x,y,w}(\mathrm{relax}(\mathcal{U})) \setminus \mathrm{Proj}_{x,y,w}(\mathrm{relax}(\mathcal{B}))$. Consider a point in $\mathrm{relax}(\mathcal{U})$ such that

$$\begin{aligned} \bar{z}_i = \frac{1}{b},\ \bar{v}_i = az_i, \qquad & \forall i \in \{1, \dots, b\} \\ y = \frac{b+1}{2},\ w = ay,\ x = \frac{a}{b}&. \end{aligned}$$

Suppose, for the purpose of contradiction, there exist z and v such that $(x, y, w, z, v) \in \mathrm{relax}(\mathcal{B})$. Then, $w - ay = 0$ implies

$$\sum_{i=1}^{k} 2^{i-1}(v_i - az_i) = 0.$$

Since $v_i \le az_i, \forall i \in \{1, \dots, k\}$, it follows from the above equality that $v_i = az_i$ and consequently $v_i = az_i \le x$, $\forall i \in \{1, \dots, k\}$. Thus,

$$\begin{aligned} y &= \sum_{i=1}^{k} 2^{i-1} z_i \\ &\le \frac{\sum_{i=1}^{k} 2^{i-1} x}{a} \\ &\le \frac{2^k - 1}{2^{k-1}}, \end{aligned}$$

since $x = a/b$ and $b \ge 2^{k-1}$. One can verify that $(2^k - 1)/2^{k-1} < 2$, which leads to $y < 2$. However this is a contradiction because we chose $y = (b+1)/2$ and assumed $b \ge 3$. Hence we have shown that $\mathrm{Proj}_{x,y,w}(\mathrm{relax}(\mathcal{U})) \setminus \mathrm{Proj}_{x,y,w}(\mathrm{relax}(\mathcal{B}))$ is nonempty. □

In the two MILP reformulations $\mathcal{U}$ and $\mathcal{B}$, the number of additional binary variables is b and $\lfloor \log_2 b \rfloor$, respectively. More binary variables for $\mathcal{U}$ implies more number of branchings to be performed in a branch-and-bound algorithm and thus, possibly a higher computational time. Hence, although the strengths of the LP relaxations of $\mathcal{U}$ and $\mathcal{B}$ are incomparable, we do not consider the reformulation $\mathcal{U}$. Our purpose, as detailed in section 3, is to tighten $\mathrm{relax}(\mathcal{B})$ using valid inequalities.

2.2. Reformulations of (BLP)**.** Suppose that we perform binary expansion of integer variable $\hat{y}_j, \forall j \in \{1, \dots, n\}$, in (BLP) and use the reformulation $\mathcal{B}$ for each bilinear term. For any given j, we use the same binary expansion variable $\hat{z}^j$ for all the bilinear variables $\hat{w}_{lj}, \forall l \in \{1,\dots,m\}$. This gives us the following extended MILP reformulation,

$$\begin{aligned}
\min \quad & \sum_{l=1}^{m}\sum_{j=1}^{n} Q_{0lj}\hat{w}_{lj} + f_0^T\hat{x} + g_0^T\hat{y} \\
\text{s.t.} \quad & A\hat{x} + G\hat{y} \le h_0 \\
& \sum_{l=1}^{m}\sum_{j=1}^{n} Q_{tlj}\hat{w}_{lj} + f_t^T\hat{x} + g_t^T\hat{y} \le h_t, \qquad t = 1,\dots,p, \\
& (\hat{x}_l, \hat{y}_j, \hat{w}_{lj}, \hat{z}^j, \hat{v}^{lj}) \in \mathcal{B}_{lj}, \quad l = 1,\dots,m,\ j = 1,\dots,n.
\end{aligned} \tag{B-BLP}$$

Alternatively, linearizing every bilinear term in (BLP) using the set $\mathcal{M}$ gives us the following MILP relaxation.

$$\begin{aligned}
\min \quad & \sum_{l=1}^{m}\sum_{j=1}^{n} Q_{0lj}\hat{w}_{lj} + f_0^T\hat{x} + g_0^T\hat{y} \\
\text{s.t.} \quad & A\hat{x} + G\hat{y} \le h_0 \\
& \sum_{l=1}^{m}\sum_{j=1}^{n} Q_{tlj}\hat{w}_{lj} + f_t^T\hat{x} + g_t^T\hat{y} \le h_t, \qquad t = 1,\dots,p, \\
& (\hat{x}_l, \hat{y}_j, \hat{w}_{lj}) \in \mathcal{M}_{lj}, \quad l = 1,\dots,m,\ j = 1,\dots,n.
\end{aligned} \tag{M-BLP}$$

On comparing the above two formulations, we note that (M-BLP) has at most $4mn$ more constraints than (BLP) whereas (B-BLP) has at most $(m+1)\sum_{j=1}^{n} k_j$ more variables and $4m\sum_{j=1}^{n} k_j$ more constraints than (BLP), where $k_j = \lfloor \log_2 \hat{b}_j \rfloor + 1$ for $j = 1,\dots,n$.

Let $OPT(\cdot)$ denote the optimum value of a problem. Since $\mathcal{P} = \text{Proj}_{x,y,w}(\mathcal{B})$, it follows that solving (B-BLP) gives us the true optimal value of (BLP), i.e. OPT(B-BLP) $= OPT$(BLP). On the contrary, because $\mathcal{P}$ is a strict subset of $\mathcal{M}$ for $b \ge 2$, (M-BLP) is a relaxation and thus OPT(M-BLP) $\le OPT$(BLP).

2.2.1. Branching strategy for solving (M-BLP)**.** Since formulation (M-BLP) is a relaxation of the original formulation (BLP), one way of obtaining the true optimum value OPT(BLP) using (M-BLP) is to branch on integer feasible solutions. This procedure is explained next. Suppose that we are at a node in the MILP search tree such that the solution at this node $(\hat{x}_l^*, \hat{y}_j^*, \hat{w}_{lj}^*) \in \mathcal{M}_{lj} \setminus \mathcal{P}_{lj}$, for some indices l, j. Thus, $\hat{y}_j^* \in \mathbb{Z}_+$ and this node provides an integer feasible solution to (M-BLP) but $\hat{w}_{lj}^* \ne \hat{x}_l^*\hat{y}_j^*$ implies that this proposed incumbent is infeasible to (BLP). Since the McCormick linearization $\mathcal{M}_{lj}$ of $\mathcal{P}_{lj}$ is exact at the variable bounds, it must be that $\hat{y}_j^* \in (\ell_j, \mu_j)$, where ℓ_j (resp. μ_j) is the lower (resp. upper) bound on $\hat{y}_j$ at this current node. Then, we can branch on the variable $\hat{y}_j$ using the disjunction $\{\hat{y}_j \le \hat{y}_j^*\} \ \vee \ \{\hat{y}_j \ge \hat{y}_j^* + 1\}$. After branching on $\hat{y}_j$, the McCormick envelopes of $w_{lj} = \hat{x}_l\hat{y}_j$ in the two branches are updated using the refined bounds on $\hat{y}_j$. Although $\hat{y}_j = \hat{y}_j^*$ is included in the left ($\hat{y}_j \le \hat{y}_j^*$) branch, one can easily verify that $(\hat{x}_l^*, \hat{y}_j^*, \hat{w}_{lj}^*)$ is cutoff from the left branch by the refined McCormick envelopes. An integer feasible node to (M-BLP) is accepted as an incumbent solution to (BLP) when $|\hat{w}_{lj}^* - \hat{x}_l^*\hat{y}_j^*| \le \epsilon$, $\forall\ l \in \{1,...,m\},\ j \in \{1,...,n\}$, for a small enough positive ϵ. Hence, at termination, we obtain an optimal solution to (BLP). To ensure numerical correctness of the algorithm, the value of ϵ should be chosen equal to the feasibility tolerance in the MILP solver.

It is important to observe that in this proposed branching strategy, we only branch on the integer variables $\hat{y}_j$'s. Thus while solving (M-BLP), we do not branch on the continuous variables $\hat{x}_l$'s as done in the spatial branch-and-bound framework within global optimization solvers.

3. Facets of $\text{conv}(\mathcal{B})$**.** In this section, the focus is on solving reformulation (B-BLP). We conduct a polyhedral study of $\text{conv}(\mathcal{B})$ and describe $\text{conv}(\mathcal{P})$ in the (x, y, w, z, v)-space. The aim is to use these facets as valid inequalities in a branch-and-cut algorithm for solving problem (B-BLP).

We first provide some definitions that will be used in this section. Let

$$\mathcal{K} := \left\{ z \in \{0,1\}^k \colon \sum_{i=1}^{k} 2^{i-1} z_i \leq b \right\} \tag{3.1}$$

be a $\{0,1\}$-knapsack set and let

$$\mathcal{R}^{\mathcal{K}} := \left\{ (x, z, v) \in \Re_+ \times \Re^k \times \Re^k \colon z \in \mathcal{K}, x \leq a, v_i = x z_i, \forall i \in \{1, \ldots, k\} \right\}. \tag{3.2}$$

Note that since $\mathcal{K} \subseteq \{0,1\}^k$, the McCormick linearization of $v_i = x z_i$ is exact for all i, and hence $\mathcal{R}^{\mathcal{K}}$ can be rewritten as

$$\begin{aligned} \mathcal{R}^{\mathcal{K}} = \Big\{ (x, z, v) \in \quad & \Re_+ \times \Re^k \times \Re^k \colon z \in \mathcal{K}, \\ & v_i \geq 0, v_i \leq a z_i, v_i \leq x, v_i \geq x + a z_i - a, \forall i \in \{1, \ldots, k\} \Big\}. \end{aligned}$$

From the definition of $\mathcal{B}$ in (2.4), it follows that the variables y and w are just linear functions of z and v, respectively. Hence

$$\text{conv}(\mathcal{B}) = \left\{ (x, y, w, z, v) \colon y = \sum_{i=1}^{k} 2^{i-1} z_i,\ w = \sum_{i=1}^{k} 2^{i-1} v_i,\ (x, z, v) \in \text{conv}(\mathcal{R}^{\mathcal{K}}) \right\}. \tag{3.3}$$

3.1. Disjunctive result. We now present a general result that helps us determine the convex hull of $\mathcal{P}$. Since $\text{conv}(\mathcal{P}) = \text{Proj}_{x,y,w}(\text{conv}(\mathcal{B}))$, equation (3.3) tells us that an extended representation of $\text{conv}(\mathcal{P})$ can be easily obtained once we know $\text{conv}(\mathcal{R}^{\mathcal{K}})$.

Let $\mathcal{S} \subset \Re^k$ be some nonconvex set (not necessarily discrete) and define $\mathcal{R}^{\mathcal{S}}$ as

$$\mathcal{R}^{\mathcal{S}} := \left\{ (x, z, v) \in \Re_+ \times \Re^k \times \Re^k \colon z \in \mathcal{S}, x \leq a, v_i = x z_i, \forall i \in \{1, \ldots, k\} \right\}. \tag{3.4}$$

The next proposition gives a relationship between the convex hulls of $\mathcal{R}^{\mathcal{S}}$ and $\mathcal{S}$ under certain assumptions. In particular, it obtains a linear inequality description of $\text{conv}(\mathcal{R}^{\mathcal{S}})$ by multiplying each linear inequality describing $\text{conv}(\mathcal{S})$ by x and $a - x$ and replacing the product term $x z_i$ with v_i.

Proposition 3.1. *Assume that the convex hull of $\mathcal{S}$ is a polytope and let it be characterized as $\text{conv}(\mathcal{S}) = \{z \colon \Pi z \leq \pi_0\}$, for some matrix Π and right hand side π_0. Then, the convex hull of $\mathcal{R}^{\mathcal{S}}$ is a polyhedron given by*

$$\begin{aligned} \text{conv}(\mathcal{R}^{\mathcal{S}}) = \Big\{ (x, z, v) \colon x \in [0, a], \qquad & \Pi v \ - \ \pi_0 x \ \leq \ 0, \\ \Pi z - \frac{1}{a} & \Pi v \ + \ \frac{1}{a} \pi_0 x \ \leq \ \pi_0 \Big\}. \end{aligned} \tag{3.5}$$

Proof. We first claim that the convex hull of $\mathcal{R}^{\mathcal{S}}$ can be represented as the convex hull of the union of two polyhedra. To prove this claim, define two sets

$$\mathcal{T}_0 := \{(x,z,v)\colon z \in \operatorname{conv}(\mathcal{S}), x = 0, v = 0\}$$
$$\mathcal{T}_1 := \{(x,z,v)\colon z \in \operatorname{conv}(\mathcal{S}), x = a, v_i - az_i = 0,\ i \in \{1,\dots,k\}\}.$$

By our assumption on $\operatorname{conv}(\mathcal{S})$, it follows that both $\mathcal{T}_0$ and $\mathcal{T}_1$ are polyhedra.

Claim 1. $\operatorname{conv}(\mathcal{R}^{\mathcal{S}}) = \operatorname{conv}(\mathcal{T}_0 \cup \mathcal{T}_1)$. We first verify the reverse inclusion $\operatorname{conv}(\mathcal{T}_0 \cup \mathcal{T}_1) \subseteq \operatorname{conv}(\mathcal{R}^{\mathcal{S}})$. Note that T_0 represents the convex hull of a subset of $\mathcal{R}^{\mathcal{S}}$ obtained by fixing $x = 0$ and $v_i = 0, \forall i$. Similarly T_1 is the convex hull of a subset of $\mathcal{R}^{\mathcal{S}}$ obtained by fixing $x = a$ and $v_i - az_i = 0, \forall i$. Hence, $T_0 \cup T_1 \subseteq \operatorname{conv}(\mathcal{R}^{\mathcal{S}})$ implying that $\operatorname{conv}(\mathcal{T}_0 \cup \mathcal{T}_1) \subseteq \operatorname{conv}(\mathcal{R}^{\mathcal{S}})$.

Now, consider any point $(x,z,v) \in \mathcal{R}^{\mathcal{S}}$. Note that we can rewrite $(x,z,v) = (1-\frac{x}{a})(0,z,0) + \frac{x}{a}(a,z,az)$ where $(0,z,0) \in \mathcal{T}_0$ and $(a,z,az) \in \mathcal{T}_1$. Hence, every point in $\mathcal{R}^{\mathcal{S}}$ can be written as a convex combination of one point from $\mathcal{T}_0$ and another point from $\mathcal{T}_1$. This gives us $\mathcal{R}^{\mathcal{S}} \subseteq \operatorname{conv}(\mathcal{T}_0 \cup \mathcal{T}_1)$ and thus the inclusion $\operatorname{conv}(\mathcal{R}^{\mathcal{S}}) \subseteq \operatorname{conv}(\mathcal{T}_0 \cup \mathcal{T}_1)$. This completes the proof of Claim 1. $\diamond$

Since $\mathcal{T}_0$ and $\mathcal{T}_1$ are both bounded, it follows that the convex hull of $\mathcal{T}_0 \cup \mathcal{T}_1$ is closed. Disjunctive programming [6] provides the following extended formulation.

$$\begin{aligned}\operatorname{conv}(\mathcal{T}_0 \cup \mathcal{T}_1) = \operatorname{Proj}_{x,z,v} \Big\{ &(x^1, z^1, v^1, x^2, z^2, v^2, x, z, v, \lambda)\colon \\ &\Pi z^1 \le \pi_0(1-\lambda), x^1 = 0, v^1 = 0, \\ &\Pi z^2 \le \pi_0 \lambda, x^2 = a\lambda, v^2 - az^2 = 0, \\ &z = z^1 + z^2, x = x^1 + x^2, v = v^1 + v^2, \\ &\lambda \in [0,1] \Big\}.\end{aligned} \tag{3.6}$$

The projection on to the space of (x,z,v) variables is easily obtained by observing that $x = a\lambda, v = az^2$, and $z - \frac{1}{a}v = z^1$. Now Claim 1 and projection of (3.6) give the desired result. $\square$

Using (3.3), we obtain that $\operatorname{conv}(\mathcal{B})$ is given by $\operatorname{conv}(\mathcal{R}^{\mathcal{K}})$ and two linear equalities. Proposition 3.1 helps us obtain $\operatorname{conv}(\mathcal{R}^{\mathcal{K}})$ by multiplying the linear inequalities describing $\operatorname{conv}(\mathcal{K})$ with x and $a - x$.

3.2. Minimal covers of knapsack. It remains to find the convex hull of $\mathcal{K}$. A complete description of the convex hull of a knapsack set with arbitrary weights is unknown. However, note that $\mathcal{K}$ is a special case of the sequential knapsack polytope studied by Pochet and Weismantel [29], who provided a constructive procedure for obtaining all the exponentially many facets of a sequential knapsack polytope with arbitrary upper bounds on variables and divisible coefficients (that are not just powers of some natural number). The set $\mathcal{K}$ is a special case where the weight of each item in the knapsack is a successively increasing power of two. We discuss its properties below.

Consider $\mathcal{K}$ and note that $k = \lfloor \log_2 b \rfloor + 1$. Hence, $2^{k-1} \le b < 2^k$. Now let the binary expansion of b be given by

$$b = 2^{i_1-1} + 2^{i_2-1} + \cdots + 2^{i_r-1} + 2^{k-1}, \tag{3.7}$$

for some $r \ge 0$. Since $2^{k-1} \le b < 2^k$, we can assume w.l.o.g. that the last exponent in the above equation is $k-1$. Note therefore that the convex hull of $\mathcal{K}$ is full dimensional.

We use $r = 0$ to denote that $b = 2^{k-1}$. Let $N := \{1, 2, \ldots, k\}$ and define a function $\sigma\colon 2^N \mapsto \Re_+$ as follows

$$\sigma(C) = \begin{cases} 0 & C = \emptyset, \\ \sum_{i \in C} 2^{i-1} & \text{otherwise.} \end{cases} \tag{3.8}$$

The function $\sigma(\cdot)$ is monotone in the sense that $\sigma(C_1) \le \sigma(C_2)$ for any $C_1 \subseteq C_2 \subseteq N$. A key observation is that

$$\sigma(C) < \sigma(i^*), \quad \text{for any } C \subseteq N \text{ and } i^* > \max\{i\colon i \in C\}. \tag{3.9}$$

Definition 3.2. *A sequence of positive reals $\{a_1, a_2, \ldots\}$ is said to be (weakly) superincreasing if it satisfies $\sum_{\tau=1}^{q} a_\tau < (\le)\ a_{q+1}$, for $q \ge 1$.*

It follows from equation (3.9) that the weights of $\mathcal{K}$ form a superincreasing sequence. Laurent and Sassano [23] used a previous result to construct all the nontrivial facet-defining inequalities for a knapsack with weakly superincreasing weights. We paraphrase their result next.

Proposition 3.3 (Theorem 2.5 and Corollary 2.6 [23]). *Consider a knapsack $\tilde{\mathcal{K}} := \{\tilde{z} \in \{0,1\}^n\colon \sum_{i=1}^{n} \tilde{a}_i \tilde{z}_i \le \tilde{b}\}$ such that $\{\tilde{a}_1, \ldots, \tilde{a}_n\}$ is weakly superincreasing. Construct integers $\tau_1, \ldots, \tau_q$, for some $q \ge 1$, such that $\tau_q = n$ and*

$$\tau_i \ := \ \max \left\{ h < \tau_{i+1}\colon \sum_{j=i+1}^{n} \tilde{a}_{\tau_j} + \tilde{a}_h \ \le \ b \right\}, \qquad 1 \le i \le q-1,$$

and the intervals $\mathcal{A}_i := \{\tau_i + 1, \ldots, \tau_{i+1} - 1\}, 1 \le i \le q-1$. Then,

1. *The minimal covers of $\mathcal{K}$ are the sets*

$$\mathcal{C}_{i,j} = \{j, \tau_{i+1}, \ldots, \tau_q = n\}, \qquad j \in \mathcal{A}_i, 1 \le i \le q-1.$$

2. *The nontrivial facets of $\mathcal{K}$ are given by the minimal covering inequalities*

$$\tilde{z}_j + \tilde{z}_{\tau_{i+1}} + \cdots + \tilde{z}_{\tau_q} \le q - i, \qquad j \in \mathcal{A}_i, 1 \le i \le q-1.$$

Let $\mathcal{C}$ denote the set of minimal covers of $\mathcal{K}$. Proposition 3.3 provides a way of constructing elements of $\mathcal{C}$. For the sake of completeness, we provide an explicit description of $\mathcal{C}$ to establish its dependence on the binary expansion of the right hand side b.

Proposition 3.4. *Define $I := \{i_1, \ldots, i_r, k\}$, where b is given by (3.7) and $\sigma(I) = b$. Assume w.l.o.g. that $i_1 < i_2 < \cdots < i_r < k$. For any $j \in N \setminus I$, let $I_j := \{i \in I\colon i > j\}$. Then,*

$$\mathcal{C} = \bigcup_{j \in N \setminus I} \{j, I_j\}. \tag{3.10}$$

Proof. Note that if $b = 2^k - 1$, then the knapsack inequality in (3.1) is redundant and the set of covers is empty. Henceforth, assume that $b < 2^k - 1$.

We first verify that elements of the form $\{j, I_j\}$ define a minimal cover. Choose a $j \in N \setminus I$ and let $C = \{j, I_j\}$. Then, $\sigma(I) = \sigma(I \setminus I_j) + \sigma(I_j) = \sigma(I \setminus I_j) + \sigma(C) - 2^{j-1}$. Using (3.9), we have that $\sigma(I \setminus I_j) < 2^{j-1}$. Hence, $b = \sigma(I) < \sigma(C)$ and C is a valid cover for the knapsack. Since $2^{j-1} < 2^{i-1}$, for $i \in I_j$, we have that for any $q \in C \setminus j$,

$\sigma(C \setminus q) < \sigma(I_j) \le \sigma(I) = b$. Finally, $\sigma(C \setminus j) \le \sigma(I) \le b$. Hence, C is a minimal cover.

Now, let $C \in \mathcal{C}$ be a minimal cover of the knapsack. Since I is not a cover by definition, we must have $|C \setminus I| \ge 1$. Define $c_1 := \max\{j : j \in C \setminus I\}$ and $T_1 := \{j \in C : j > c_1\}$.

Claim 1. $T_1 = I_{c_1}$. By definition of c_1 and T_1, we obtain that $T_1 \subseteq I_{c_1}$. Now take $j \in I_{c_1}$ and suppose for the sake of contradiction that $j \notin C$. Define $c_q := \max\{i \in C : i < j\}$ and $C^{'} := \{i \in C : i > c_q\}$. By definition of c_q and by the assumption that $j \notin C$, it must be that $C^{'} = \{i \in C : i > j\}$. Also, $C^{'} \subseteq C$ and hence by definition of c_1, we have $C^{'} \subseteq I$. Then $C^{'} \subseteq C \cap I$ and

$$\begin{aligned} b &= \sigma(I) \\ &\ge \sigma(j) + \sigma(C^{'}) && \text{since } j \cup C^{'} \subseteq I \\ &> \sigma(\{i \in C : i < j\}) + \sigma(C^{'}) && \text{due to (3.9)} \\ &= \sigma(C). \end{aligned}$$

Hence C is not a cover, a contradiction. This implies $j \in C$ and since $j > c_1$, it must be that $j \in T_1$. Hence, $I_{c_1} \subseteq T_1$ and finishes the proof of our claim. $\diamond$

By the above claim it follows that $\{c_1, I_{c_1}\} \subseteq C$. Since $\{c_1, I_{c_1}\}$ is a cover, by minimality of C, we obtain that $C = \{c_1, I_{c_1}\}$. $\square$

Example 1. Let $b = 38 = 2^{2-1} + 2^{3-1} + 2^{6-1}$. Hence, $I = \{2, 3, 6\}$. Then, the set of minimal covers is $\{(1, 2, 3, 6), (4, 6), (5, 6)\}$.

From Propositions 3.1, 3.3, and 3.4, and equation (3.3), we obtain the convex hull of $\mathcal{B}$.

COROLLARY 3.5.

$$\begin{aligned} \text{conv}(\mathcal{B}) = \Big\{(x, y, w, z, v) :\ & y = \sum_{i=1}^{k} 2^{i-1} z_i,\ w = \sum_{i=1}^{k} 2^{i-1} v_i, \\ & v_j + \sum_{i \in I_j} v_i - |I_j| x \le 0, \quad j \in N \setminus I \\ & z_j - \frac{1}{a} v_j + \sum_{i \in I_j} \Big(z_i - \frac{1}{a} v_i\Big) + \frac{|I_j|}{a} x \le |I_j|, \quad j \in N \setminus I \\ & v_i \ge 0, v_i \le a z_i, v_i \le x, v_i \ge x + a z_i - a, \ \ i = 1, \dots, k \Big\}. \end{aligned} \tag{3.11}$$

For the sake of completeness, we next address two closely related cases.

Remark 1. Let y be a semi-integer, i.e. $y \in \{0\} \cup \{b^{'}, b^{'} + 1, \dots, b\}$ for some positive integers b', b. Rewriting $y = b^{'} z_0 + \sum_{i=1}^{k} 2^{i-1} z_i$ yields

$$\mathcal{S}^{'} = \left\{ z \in \{0,1\}^{k+1} :\ \sum_{i=1}^{k} 2^{i-1} z_i \le b - b^{'},\ z_i \le z_0, \forall i \in \{1, \dots, k\} \right\}, \tag{3.12}$$

where $y \in \{b^{'}, b^{'}+1, \dots, b\}$ if and only if $z_0 = 1$. Let $\mathcal{K}^{'} = \{z \in \{0,1\}^k :\ \sum_{i=1}^{k} 2^{i-1} z_i \le b - b^{'}\}$ and its convex hull, which can be obtained from Proposition 3.3, be represented as $\text{conv}(\mathcal{K}^{'}) = \{z \in \Re^k : \Pi z \le \pi_0\}$. Observe that $\mathcal{S}^{'} = (\mathcal{K}^{'} \times \{1\}) \cup \{\mathbf{0}\}$. Thus, $\text{conv}(\mathcal{S}^{'}) = \text{conv}(\text{conv}(\mathcal{K}^{'} \times \{1\}) \cup \{\mathbf{0}\}) = \text{conv}(\{z \in \Re^{k+1} : \Pi z \le \pi_0, z_0 = 1\} \cup \{0\})$. Disjunctive programming provides an extended formulation that can be easily projected to obtain the identity $\text{conv}(\mathcal{S}^{'}) = \{z \in \Re^{k+1} : \Pi z \le \pi_0 z_0\}$. Applying

Proposition 3.1 gives the convex hull of the corresponding mixed semi-integer bilinear set.

Remark 2. Suppose that in the binary expansion knapsack defined in equation (3.1), some powers of two are missing. Thus $\mathcal{K}^{'} := \left\{ z \in \{0,1\}^{k'} : \sum_{t=1}^{k'} 2^{i_t - 1} z_t \leq b \right\}$, where $\{i_1, i_2, \ldots, i_{k'}\} =: \mathcal{I}^{'} \subseteq \{1, 2, \ldots, k\}$ such that $i_{k'} = k$. The knapsack weights still form a superincreasing sequence and $\mathcal{K}^{'}$ is a subset of $\mathcal{K}$ obtained by restricting $z_i = 0$ for all $i \notin \mathcal{I}^{'}$. Since for every $i \notin \mathcal{I}^{'}$, $z_i = 0$ is a face of the 0\1 polytope $\mathrm{conv}(\mathcal{K})$, it follows that $\mathrm{conv}(\mathcal{K}^{'})$ is given by fixing $z_i = 0$ for all $i \notin \mathcal{I}^{'}$ in the minimal covering inequalities defining $\mathrm{conv}(\mathcal{K})$. Thus if we add *all the* covering inequalities of Proposition 3.3 as cutting planes at the root node of a branch-and-cut algorithm for solving (B-BLP), then we cannot obtain any nontrivial cover cuts corresponding to knapsack $\mathcal{K}$ at nodes below the root node.

4. Computational results. In this section we report computational results on several test instances. Given a mixed integer bilinear problem, we solved it using the open source nonconvex MINLP solver `Couenne 0.3` [7]. Our goal is to test whether these bilinear problems can be solved efficiently using the MILP formulations (M-BLP) and (B-BLP) from Section 2. We used `CPLEX 12.1` as the MILP solver. Since `CPLEX` is a sophisticated commercial MILP solver whereas `Couenne` is a relatively new open source MINLP solver, we cannot and do not wish to draw conclusions regarding the performance of the spatial branch-and-bound algorithm. Instead, our aim is to show that the proposed MILP approach is a viable alternative on certain classes of problems.

To ensure numerical consistency between `Couenne` and `CPLEX`, we used the following algorithmic parameters: `feasibility tolerance =` 10^{-6}, `integrality tolerance =` 10^{-5}, `relative optimality gap =` 0.01%, `absolute optimality gap =` 10^{-4}. Additionally, for `CPLEX`, we set `Threads = 1` to enforce single threaded computing. All other options were set to default values for the respective solver. Our assumption of nonnegative lower bounds on variables is without any loss of generality since we translated every variable with a nonzero lower bound so that the formulation conforms to (BLP1).

For the MILP relaxation (M-BLP), we employed branching on integer solutions, as discussed towards the end of Section 2. While branching at any fractional or integer node of the branch-and-bound tree, updated McCormick envelopes were added for each bilinear term corresponding to the branching variable, using the local bounds on the variables at this particular node. This is a standard technique used by global optimization solvers. In our preliminary computations, this technique performed better than updating the envelopes only when we branch on integer nodes. The variable selection rule for the branching strategy of §2.2.1 was based on maximum violated bilinear term whereas for fractional nodes we used the branches proposed by `CPLEX`. By solving the relaxation (M-BLP) as a MILP without branching on continuous variables, we have adopted a traditional branch-and-bound solution strategy to test if branching on integer nodes in the original space can outperform the spatial branch-and-bound algorithm of `Couenne` or the extended binary MILP reformulation (B-BLP).

While solving reformulation (B-BLP), the general integer variables $\hat{y}_j$, for all $j \in \{1, ..., n\}$, were substituted out in order to reduce the problem size and to ensure that branching is performed solely on the binary variables. One approach was to solve this reformulation using default branch-and-cut options for `CPLEX`. In the second approach, we added all the inequalities defining $\mathrm{conv}(\mathcal{B}_{lj})$, for all $l \in \{1, ..., m\},\ j \in \{1, ..., n\}$

(see (3.11)), to the user cut pool of CPLEX along with default branch-and-cut options. In our preliminary computations, we also tested the following idea: retaining integer variables $\hat{y}_j, \forall j$, and whenever CPLEX chooses some $\hat{y}_j$ as a branching variable, adding cover inequalities corresponding to the refined bound on $\hat{y}_j$ as local cuts at this node. However, we found no performance gain with this approach.

We test five classes of instances - MINLPLIB, Bundling, MIPLIB, BoxQP, and Disjoint. The experiments were carried out on a Linux machine with kernel 2.6.18 running on a 64-bit x86 processor and 32GB of RAM. The time limit was 1 hour barring a few instances from MINLPLIB and MIPLIB. Tables 4.1–4.11 highlight comparisons between four solution approaches - Couenne, (M-BLP), (B-BLP) + Cuts, and (B-BLP). We report the number of nodes (Nds) processed by the branch-and-bound algorithm and the running time (T) in seconds. A * indicates the instance was not solved to optimality within the time limit. For the binary expansion reformulation, we also report the total number of cover inequalities that were separated by CPLEX (Cuts) and the % root gap closed (Rgp-cl) by adding our cuts with CPLEX cuts over adding only CPLEX cuts.

For an instance $\mathcal{I}$ not solved to optimality within the desired time limit, we solved the formulation (BLP) using Couenne for a long period of time (24hrs). The best feasible solution value after 24hrs.[1] is recorded as $OPT_{\mathcal{I}}$. The performance of the four proposed approaches is compared using two types of % optimality gaps. The first one measures the quality of $\xi_{\mathcal{I}}^{\text{relax}}(\mathcal{A})$, the best relaxation bound due to method $\mathcal{A}$, and is given by

$$\mu_{\mathcal{I}}(\mathcal{A}) = 100 \times \left| 1 - \frac{\xi_{\mathcal{I}}^{\text{relax}}(\mathcal{A})}{OPT_{\mathcal{I}}} \right| \tag{4.1}$$

Thus, $\mu_{\mathcal{I}}(\mathcal{A})$ denotes how close method $\mathcal{A}$ was to solving $\mathcal{I}$ to optimality. The second metric is the % optimality gap of the best feasible solution found by $\mathcal{A}$ (denoted as $\xi_{\mathcal{I}}(\mathcal{A})$) and is given by

$$\omega_{\mathcal{I}}(\mathcal{A}) = 100 \times \left| 1 - \frac{\xi_{\mathcal{I}}(\mathcal{A})}{OPT_{\mathcal{I}}} \right| \tag{4.2}$$

An optimality gap of (–) means no integer feasible solution was found by the algorithm within the time limit. For our test instances, we observed $OPT_{\mathcal{I}} \neq 0$.

Performance profiles. The various performance metrics, such as number of nodes, solution time, optimality gaps, number of user cuts etc., are reported individually for each test instance from MINLPLIB and MIPLIB whereas average values are reported for the remaining three instance classes. We also plot performance profiles [14] of solved and unsolved instances[2] from each of the three classes[3] - Bundling, BoxQP, and Disjoint. For each instance class, we compare the solution times from the four approaches : Couenne, (M-BLP), (B-BLP) + Cuts, and (B-BLP), on the

[1] In particular, $OPT_{\mathcal{I}}$ is the best solution value compared across a) any of the MILPs solved for 1hr. and b) that returned by Couenne after 24hrs.

[2] An instance is marked solved if it was solved within 1hr. by any of the four methods, otherwise it is marked unsolved.

[3] We do not plot performance profiles for MINLPLIB and MIPLIB since individual metrics are reported for these two instance classes.

subset of instances solved within 1hr. Let $T_{\mathcal{I}}(\mathcal{A})$ denote the CPU time in seconds for solving instance $\mathcal{I}$ with method $\mathcal{A}$ and let $\eta_{\mathcal{I}}(\mathcal{A})$ be a relative metric calculated as

$$\eta_{\mathcal{I}}(\mathcal{A}) = \frac{T_{\mathcal{I}}(\mathcal{A}) \; - \; \min_{\mathcal{A}} T_{\mathcal{I}}(\mathcal{A})}{\max_{\mathcal{A}} T_{\mathcal{I}}(\mathcal{A}) \; - \; \min_{\mathcal{A}} T_{\mathcal{I}}(\mathcal{A})} \; \in \; [0,1]$$

Any point (Ω_T, β_T) on the CPU time performance profile for method $\mathcal{A}$ indicates that $\eta_{\mathcal{I}}(\mathcal{A})$ was at most Ω_T for a fraction β_T of the solved instances. Thus, the point $(0, \beta_T)$ implies that a β_T fraction of instances were solved quickest by $\mathcal{A}$. For the subset of unsolved instances after 1hr., we compare the performance profiles of the best feasible solutions obtained with the different methods. Let $\xi_{\mathcal{I}}(\mathcal{A})$ be the best solution value for instance $\mathcal{I}$ using method $\mathcal{A}$. The relative metric $\theta_{\mathcal{I}}(\mathcal{A}) \in [0,1]$ is defined as

$$\theta_{\mathcal{I}}(\mathcal{A}) = \frac{\xi_{\mathcal{I}}(\mathcal{A}) \; - \; \min_{\mathcal{A}} \xi_{\mathcal{I}}(\mathcal{A})}{\max_{\mathcal{A}} \xi_{\mathcal{I}}(\mathcal{A}) \; - \; \min_{\mathcal{A}} \xi_{\mathcal{I}}(\mathcal{A})} \qquad \text{for minimization}$$

$$\theta_{\mathcal{I}}(\mathcal{A}) = \frac{\max_{\mathcal{A}} \xi_{\mathcal{I}}(\mathcal{A}) \; - \; \xi_{\mathcal{I}}(\mathcal{A})}{\max_{\mathcal{A}} \xi_{\mathcal{I}}(\mathcal{A}) \; - \; \min_{\mathcal{A}} \xi_{\mathcal{I}}(\mathcal{A})} \qquad \text{for maximization.}$$

Any point $(\Omega_{\text{val}}, \beta_{\text{val}})$ on the solution value performance profile for method $\mathcal{A}$ implies that $\theta_{\mathcal{I}}(\mathcal{A})$ was at most Ω_{val} for a fraction β_{val} of the unsolved instances. Thus, the point $(0, \beta_{\text{val}})$ implies that $\mathcal{A}$ found the best solution on β_{val} fraction of unsolved instances.

While comparing performance profiles, the most effective method is the one whose profile is the topmost left corner. More details on performance profiling can be found in [14].

4.1. General mixed integer bilinear problems. This set of instances contains problems formulated as (BLP1) where bilinear terms are present in both the objective function and constraints. We divide into two subcategories depending on the source of the test problems.

4.1.1. `MINLPLIB`. We chose 14 instances from this test library [11] that have bilinearities between continuous and integer variables, such as xy, or between two integer variables, such as y_1y_2. Note that for a bilinear term y_1y_2 where $y_i \in \mathbb{Z}_+, i = 1, 2$, the result of Proposition 3.1 carries through. Instances `lop97ic` and `lop97icx` are not considered because of their large size. Instances `tln2` - `tloss` are the bilinear version of the cutting stock problem, where the number of rolls produced by each cutting pattern is also an integer variable.

From Table 4.1 we observe that the bilinear cutting stock instances `tln4` - `tln12` are perhaps the most difficult ones from this set of instances. On these 5 instances, the binary reformulation, with or without our cuts, has done better than both envelope relaxation (M-BLP) and solving with `Couenne`. In particular, for `tln4`, the nodes and time taken by binary MILP was substantially less than for the other two, whereas `tln5` and `tln6` were solved within the time limit by binary MILP (with some help from cuts on `tln6`). The time limit was set to 15min for `tln7` and `tln12`, since we did not observe any notable improvements for longer time periods. Although `tln7` and `tln12` remained unsolved by all four methods, the optimality gap at termination was higher for the first two methods.

TABLE 4.1
Test instances from MINLPLIB.

Instance	Couenne		M-BLP		B-BLP + Cuts				B-BLP	
	Nds	T	Nds	T	Nds	T	Cuts	Rgp-cl	Nds	T
ex1263a	1121	2	2366	1	635	1	0	0	640	1
ex1264a	940	1	2762	1	519	1	0	0	519	1
ex1265a	197	3	995	1	378	1	0	0	378	1
ex1266a	61	1	562	1	10	1	0	0	10	1
prob02	0	0	170	1	12	1	0	0	12	0
prob03	0	0	4	1	0	0	1	5%	0	0
tln2	2	0	12	1	183	0	0	0	183	0
tln4	47384	55	98770	118	4401	4	17	0	4576	4
tln5	496377	*	306394	*	12662	17	0	0	12662	18
tln6	421402	*	242486	*	56130	87	102	0	65514	93
tln7	316152	*	684937	*	1249707	*	36	0	1728487	*
tln12	96500	*	50618	*	134823	*	132	0	180712	*
tloss	537	3	877	1	84	1	0	0	84	1
tltr	371	1	144	1	214	1	11	2%	181	1

TABLE 4.2
% Optimality gaps for test instances from MINLPLIB.

Instance	Couenne		M-BLP		B-BLP + Cuts		B-BLP	
	$\mu_\mathcal{I}$	$\omega_\mathcal{I}$	$\mu_\mathcal{I}$	$\omega_\mathcal{I}$	$\mu_\mathcal{I}$	$\omega_\mathcal{I}$	$\mu_\mathcal{I}$	$\omega_\mathcal{I}$
tln5	42%	2%	43%	3%	0	0	0	0
tln6	54%	0%	69%	1%	0	0	0	0
tln7	75%	17%	75%	6%	34%	0	1%	0
tln12	82%	–	103%	–	80%	2%	80%	4%

4.1.2. Product Bundling. The product bundling problem addressed in [17] can be defined as follows: let P be a set of products and C be a set of customers. The variable $x_p \in \mathbb{Z}_+$ represents the number of units of product p in a bundle and $y_c \in \mathbb{Z}_+$ represents the number of bundles bought by customer c. The objective is to maximize $\sum_{c\in C}\sum_{p\in P} x_p y_c$, which is the total number of products bought, subject to the demand constraint $x_p y_c \le D_{cp}, \forall c \in C, p \in P$. Here, $D_{cp} \in \mathbb{Z}_+$ and not all D_{cp} are zero. Thus, the formulation is

$$\max \left\{ \sum_{c\in C}\sum_{p\in P} x_p y_c \colon x_p y_c \le D_{cp},\ x_p, y_c \in \mathbb{Z}_+ \forall c, p \right\}. \qquad \text{(Bundling)}$$

We first obtain valid upper bounds on the variables.

PROPOSITION 4.1. *The variables x and y in* (Bundling) *can be upper bounded as*

$$\begin{aligned} x_p &\le \mu_p^x := \max\{D_{cp} \colon c \in C\}, \quad p \in P \\ y_c &\le \mu_c^y := \max\{D_{cp} \colon p \in P\}, \quad c \in C. \end{aligned}$$

Proof. First consider the following observation.

Claim 1. $OPT(\text{Bundling}) \ge 1$. Since not all D_{cp} are zero and $D_{cp} \in \mathbb{Z}_+, \forall c, p$, there must be exist some $c \in C, p \in P$ such that $D_{cp} \ge 1$. Set $x_p = y_c = 1$ and all other variables zero. This is a feasible solution with objective value 1. $\diamond$

We now show that any optimal solution (x^*, y^*) to (Bundling) must satisfy $x^* \le \mu^x$ and $y^* \le \mu^y$. Suppose that $x_p^* > \mu_p^x$ for some $p \in P$. This implies that $y_c^* = 0$, for all $c \in C$, since every feasible point must satisfy $x_p y_c \leq D_{cp}, \forall c$. Hence, the optimal value must be zero, which is a contradiction to our first claim. Similarly for y^*. Hence, μ^x and μ^y are valid upper bounds that are not violated by any point from the set of optimal solutions. ☐

Our first problem set of this type consists of 54 instances, created using the random generator of [17]. Half of these are for $|C| = 10, |P| = 30$ and the other half for $|C| = 20, |P| = 50$. For each problem size, we considered $\rho \in \{0.2, 0.5, 0.8\}$ and $\lambda \in \{30, 100, 200\}$, where $D_{cp} = 0$ with probability ρ and if $D_{cp} > 0$, then $D_{cp} \sim \text{Poisson}(\lambda)$. For each combination of ρ and λ, 3 instances were created. Note that a bilinear term $w_{cp} = x_p y_c$ exists only if $D_{cp} > 0$. Otherwise $x_p = 0 \vee y_c = 0$. This disjunction is modeled as a bigM constraint using extra binary variables for (M-BLP) whereas for (B-BLP), the condition $w_{cp} = 0$ is incorporated in the McCormick linearization. As λ increases, the set of integer feasible solutions increases and as ρ decreases, the demand matrix becomes more dense giving rise to more bilinear terms.

In Tables 4.3 and 4.4, we present average values over the 27 random instances for each problem size. We report the average values for our metrics - number of nodes, time taken (sec.), number of user cuts added, and % root gap closed, where the averages are taken over instances in each subgroup. For each method $\mathcal{A}$, we also provide

1. number of instances (# solved) solved to optimality by $\mathcal{A}$,
2. number of instances (# fastest optimal) for which $\mathcal{A}$ found an optimality certificate in the shortest amount of time.

Since there exist some instances that are not solved to optimality by any of the formulations, we also report the following metrics calculated over the instances that were not solved with any of the four methods:

3. number of instances (# best feasible) on which the best feasible solution was found
4. average value of $\mu_{\mathcal{I}}$ over unsolved instances
5. average value of $\omega_{\mathcal{I}}$ over unsolved instances

TABLE 4.3
Product `Bundling` for $|C| = 10, |P| = 30$. 27 random instances.

	Couenne	M-BLP	B-BLP + Cuts	B-BLP
Average Nds	388824	735621	349335	383832
Average T (sec.)	2103	2409	2787	2736
Average Cuts	–	–	517	–
Average % Rgp-cl	–	–	0.5%	–
# solved	13	9	9	8
# fastest optimal	10	4	0	0
# best feasible	1	10	2	4
Average $\mu_{\mathcal{I}}$	63%	12%	203%	204%
Average $\omega_{\mathcal{I}}$	14%	5%	2.6%	2.9%

From both the tables we observe that `Couenne` solved the most number of instances in 1hr. However, amongst the unsolved problems, the best feasible solutions obtained from binary reformulation helped produce strong lower bounds (since its maximization) on the problem. This can be concluded by comparing the optimality gaps $\omega_{\mathcal{I}}$ for the four different methods. In Table 4.3, (M-BLP) was able to produce the best feasible solution on the most number of instances (10). However, the relative quality of these solutions, denoted by $\omega_{\mathcal{I}}$, was smaller than that for (B-BLP) (with and without cuts), implying that the solutions obtained with the binary reformulation model were either the best or almost always very close to being the best. For the larger problem sizes in Table 4.4, a similar reasoning holds for the $\omega_{\mathcal{I}}$ values along with the fact that now the best feasible solutions were obtained solely by one of the two binary models. On the relaxation side, it seems that although a large number of our cover cuts were separated, they were not effective in closing the root gap. In fact, most of our user cuts were separated deeper in the branch-and-cut tree suggesting that the default cuts added by `CPLEX` at root node were itself quite strong on these instances. (M-BLP) has the lowest average termination gap $\mu_{\mathcal{I}}$ and for this set of instances, our proposed branching strategy performed fairly well, possibly due to not too large interval width of the general integers.

TABLE 4.4
Product `Bundling` for $|C| = 20, |P| = 50$. 27 random instances.

	Couenne	M-BLP	B-BLP + Cuts	B-BLP
Average Nds	99790	241450	221272	181680
Average T (sec.)	2776	3600	3600	3600
Average Cuts	–	–	1245	–
Average % Rgp-cl	–	–	0.1%	–
# solved	8	0	0	0
# fastest optimal	8	0	0	0
# best feasible	0	0	10	9
Average $\mu_{\mathcal{I}}$	243%	225%	545%	536%
Average $\omega_{\mathcal{I}}$	55%	26%	9.8%	10.3%

TABLE 4.5
The `watts` instances for Product `Bundling`.

Instance	Couenne		M-BLP		B-BLP + Cuts				B-BLP	
	Nds	T	Nds	T	Nds	T	Cuts	Rgp-cl	Nds	T
`5x41`	305800	*	1217175	*	25893	141	18	0	33762	176
`5x41m`	1044023	*	1301411	*	23323	166	23	0	16426	226
`9x60`	120260	*	319347	*	55277	*	745	0	86562	*
`10x60`	97180	*	239071	*	74913	*	805	0	69911	*
`10x60d`	112030	*	291302	*	31753	808	234	0	60945	1902

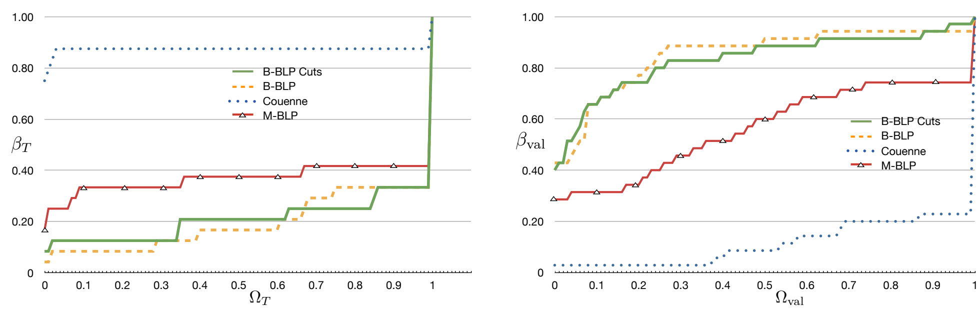
The second set of product bundling problems consists of 5 instances from a real food company, as used in [17]. These are referred to as the `watts` instances, reported in Tables 4.5 and 4.6. For these five instances, we clearly see that the binary reformulation is superior, both in terms of $\mu_{\mathcal{I}}$ and $\omega_{\mathcal{I}}$ and the solved instances. Although our cuts were not effective at the root node, they were helpful on expediting the solve of 3 out of the 5 instances, especially `10x60d` whose solution time was more than halved. On the contrary, for `9x60` and `10x60`, a lot of user cuts were separated below the root node, which potentially led to slow down of `CPLEX` and hence a higher termination gap than (B-BLP) without cuts.

The performance profiles are plotted in Figure 4.1. Maximum number of instances were solved to optimality within 1hr by `Couenne`. The profile of `Couenne` is most dominant in Figure 4.1(a) implying that our MILP formulations are not efficient in solving these instances. However on the unsolved instances, we observe that (B-BLP) with

Table 4.6
% Optimality gaps for `watts` instances

Instance	Couenne		M-BLP		B-BLP + Cuts		B-BLP	
	$\mu_{\mathcal{I}}$	$\omega_{\mathcal{I}}$	$\mu_{\mathcal{I}}$	$\omega_{\mathcal{I}}$	$\mu_{\mathcal{I}}$	$\omega_{\mathcal{I}}$	$\mu_{\mathcal{I}}$	$\omega_{\mathcal{I}}$
`5x41`	31%	7%	101%	24%	0	0	0	0
`5x41m`	6%	0	194%	25%	0	0	0	0
`9x60`	339%	21%	57%	58%	111%	0	65%	0
`10x60`	231%	24%	39%	60%	77%	0	59%	0
`10x60d`	142%	17%	38%	46%	0	0	0	0

and without cuts provided the best quality solutions. The two profiles corresponding to (B-BLP) seem to be evenly matched in terms of the feasible solution qualities.



(a) *CPU times for 24 solved instances.* (b) *Feasible solutions for 35 unsolved instances.*

Fig. 4.1. *Performance profiles for 59 Product `Bundling` problems.*

4.2. Nonconvex objective function with linear constraints.

4.2.1. `MIPLIB`. We chose MILP instances from `MIPLIB` 2003 and modified the objective function to a bilinear function. Thus, the feasible region for these instances is a polyhedron and all nonconvexities appear in the objective. For general MILPs, only those eight instances with less than 1000 integer and 1000 continuous variables were selected and the objective was

$$\max \sum_{i=1}^{n} y_i(x_i + x_{i+1} + x_{i+2}). \tag{4.3}$$

Here x_i and y_i are bounded continuous and integer variables, respectively, and the indexing of these variables is as determined by `CPLEX` after importing the `.mps` input file for the `MIPLIB` instance. The summation in (4.3) is taken only over those variables which were either originally bounded or their LP based bounds (maximizing and minimizing each variable over the LP relaxation of the feasible set) were finite[4].

[4]The indexing of integer and continuous variables is maintained separately and depends on the order in which they are read from the `.mps` file. We first drop the $\{0,1\}$ variables. Then we generate LP based bounds on the remaining variables and subsequently drop the unbounded variables. With this final ordering, there are n integer variables and m continuous variables. Each integer variable y_i is matched with three continuous variables x_i, x_{i+1}, x_{i+2}. If $n > m$, then we simply loop over the indexing of continuous variables.

Table 4.7
General MILP test instances from MIPLIB

Instance	Couenne		M-BLP		B-BLP + Cuts				B-BLP	
	Nds	T	Nds	T	Nds	T	Cuts	Rgp-cl	Nds	T
arki001	1497	*	47635	*	50233	*	131	7%	20457	*
noswot	98018	*	213037	*	4398	5	0	0	4398	5
gesa2	46	124	0	1	0	1	0	0	0	1
gesa2-o	261	*	54322	*	69	3	782	99%	36644	*
rout	87613	*	44	1	50	1	5	0	31	1
timtab1	37294	*	291471	*	311058	*	63	7%	339376	*
timtab2	48624	*	136749	*	133526	*	136	6%	138606	*
roll3000	3	*	42147	*	28678	461	28	1%	27649	496

Table 4.8
% Optimality gaps for test instances from MIPLIB. gesa2 is excluded from this table since it is solved by all four methods.

Instance	Couenne		M-BLP		B-BLP + Cuts		B-BLP	
	$\mu_{\mathcal{I}}$	$\omega_{\mathcal{I}}$	$\mu_{\mathcal{I}}$	$\omega_{\mathcal{I}}$	$\mu_{\mathcal{I}}$	$\omega_{\mathcal{I}}$	$\mu_{\mathcal{I}}$	$\omega_{\mathcal{I}}$
arki001	19%	–	21%	8%	4%	0.3%	7%	0
noswot	6%	20%	46%	27%	0	0	0	0
gesa2-o	6%	–	1%	0	0	0	0	0
rout	0	5%	0	0	0	0	0	0
timtab1	38%	10%	15%	7%	9%	0	10%	0.2%
timtab2	39%	–	31%	11%	28%	0	29%	0.1%
rol3000	65%	–	92%	63%	0	0	0	0

Only three out of the total eight instances remained unsolved for B-BLP + Cuts, least amongst all four methods. For `arki001`, `timtab1`, and `timtab2`, our cuts seemed helpful in closing some gap at the root node. For `gesa2-o`, our cuts helped solve the problem very quickly. Observe that for `arki001`, `gesa2-o`, `timtab2`, and `roll3000`, `Couenne` was unable to find a integer feasible solution within the time limit and in fact, could process only three nodes for `roll3000`, likely because of the large number of general integers in this instance. On these same four instances, our cuts were either able to solve the binary reformulation or could reduce the optimality gap.

4.2.2. `BoxQP`. Here we consider box constrained nonconvex quadratic problems

$$\begin{aligned} \min \quad & \tfrac{1}{2}\hat{x}^T Q \hat{x} + f_0^T \hat{x} \\ \text{s.t.} \quad & \hat{x} \in [0, \hat{a}] \cap \mathbb{Z}_+^n. \end{aligned} \tag{Integer BoxQP}$$

Introducing a new continuous variable $y = Qx$, we can rewrite the above problem with a bilinear objective and linear constraints as

$$\begin{aligned} \min \quad & \tfrac{1}{2}\hat{x}^T \hat{y} + f_0^T \hat{x} \\ \text{s.t.} \quad & \hat{y} = Q\hat{x} \\ & \hat{y}_i^L \le \hat{y}_i \le \hat{y}_i^U, \quad i = 1, \dots, n \\ & \hat{x} \in [0, \hat{a}] \cap \mathbb{Z}_+^n. \end{aligned} \tag{Bilinear Integer BoxQP}$$

where $\hat{y}_i^L := \sum_{j:\, q_{ij}<0} q_{ij}\hat{a}_j$ and $\hat{y}_i^U := \sum_{j:\, q_{ij}>0} q_{ij}\hat{a}_j$, for $i = 1, \dots, n$. In this transformed problem, every bilinear term $\hat{w}_i = \hat{x}_i\hat{y}_i, i = 1, \dots, n$, is a product between a bounded integer variable and a bounded continuous variable and hence conforms to the assumptions of this paper.

The test instances for our computational experiments were obtained from the 54 random instances of [35], where the authors studied $[0, 1]$ constrained nonconvex QPs. The value of n, i.e. the number of variables in (Integer BoxQP), lies in $\{20, 30, 40, 50, 60\}$ for these instances. For every instance of $[0, 1]$ box QP, we generated values of integral upper bounds $\hat{a}_i$ uniformly at random between 10 and 100, for all i. Then after a suitable scaling of the coefficient matrix and cost vector with these upper bounds, we obtain an instance for (Integer BoxQP).

The results of our experiment are summarized in Table 4.9. The second column in Table 4.9 corresponds to the solution of the the reformulated bilinear problem (Bilinear Integer BoxQP) with `Couenne`. Of the unsolved instances, the average values of $\omega_{\mathcal{I}}$ are lowest for the two binary formulations, indicating that good quality solutions are obtained by solving the MILP formulation. Our cuts close around 41% of the root gap, which translates into lower termination gap $\mu_{\mathcal{I}}$ (43% < 66%) and helps `CPLEX` spend more time in obtaining good feasible solutions for the most number of unsolved instances (41 out of 52).

Table 4.9
54 instances of (Integer BoxQP) *where Couenne is solved as* (Bilinear Integer BoxQP).

	Couenne	M-BLP	B-BLP + Cuts	B-BLP
Average Nds	850670	222470	433045	1056528
Average T (sec.)	3501	3600	3491	3588
Average Cuts	–	–	113	–
Average % Rgp-cl	–	–	41%	–
# solved	2	0	2	1
# fastest optimal	2	0	0	0
# best feasible	7	1	41	34
Average $\mu_{\mathcal{I}}$	31%	68%	43%	66%
Average $\omega_{\mathcal{I}}$	1.6%	9%	0.23%	0.18%

Only 2 out of 54 instances were solved by any of the four methods. The performance profile of feasible solutions for the 52 unsolved instances is plotted in Figure 4.2. As discussed before, user cuts obtain best quality feasible solution on 80% of the instances. The solutions obtained from the binary formulation (B-BLP) (with and without cuts) are superior to both `Couenne` and (M-BLP). The addition of our cuts significantly reduces the root gap and the optimality gap $\mu_{\mathcal{I}}$ (cf. Table 4.9) but does not seem to vastly improve the quality of solutions obtained upon termination. This is perhaps to be expected since user cuts typically do not improve the performance of primal heuristics in `Cplex`. Thus, our cuts seem to be doing their primary job of obtaining better relaxation bounds.

4.2.3. `Disjoint` bilinear problems. 100 random instances of disjoint bilinear problems were created using the instance generator of [36]. These test instances have a bilinear objective function and the feasible region is defined by a cartesian product of two polyhedra, one in x-space and another in y-space. The y variables are restricted to be integer.

$$\begin{aligned} \min \quad & \hat{x}^T Q_0 \hat{y} + f_0^T \hat{x} + g_0^T \hat{y} \\ \text{s.t.} \quad & \hat{x} \in X := \{\hat{x} \in \Re^{2\kappa_2} : A\hat{x} \le h_a\} \\ & \hat{y} \in Y := \{\hat{y} \in \mathbb{Z}^{\kappa_1+\kappa_2} : B\hat{y} \le h_b\}. \end{aligned} \qquad \text{(Disjoint BLP)}$$

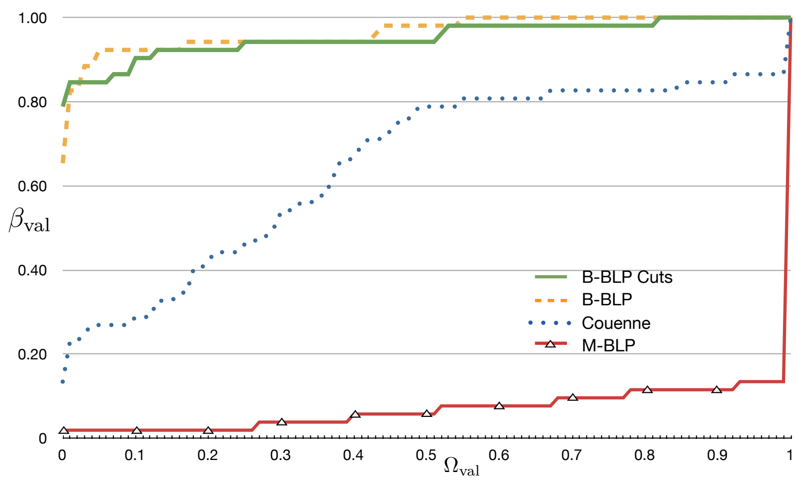


FIG. 4.2. *Performance profile of feasible solutions for 52 unsolved instances of* `BoxQP` *problems.*

The values $\delta = 2$ and $\rho = 0$ were used while generating components of matrices A and B and the final values of $Q_0, f_0, g_0, A, B, h_a, h_b$ were obtained using randomized Householder matrices, the seed for which was set equal to $instanceid \times \kappa_1 \times \kappa_2$. A more detailed description of the instance generator can be found in [36]. The parameters κ_1 and κ_2 control the size of the problem. The total number of variables and constraints in (Disjoint BLP) is equal to $\kappa_1 + 3\kappa_2$ and $2\kappa_1 + 4\kappa_2$, respectively. LP based bounds are generated for each variable and any unbounded variable is given an artificial upper (lower) bound of 100 (-100). The instances are divided into two subgroups: half of them were generated with $\kappa_1 = 2, \kappa_2 = 4$ and the other half for $\kappa_1 = 3, \kappa_2 = 5$.

TABLE 4.10
`Disjoint` *bilinear instances:* $\kappa_1 = 2, \kappa_2 = 4$. *Fifty random instances.*

	Couenne	M-BLP	B-BLP + Cuts	B-BLP
Average Nds	24289	78414	44322	44090
Average T (sec.)	45	3351	24	23
Average Cuts	–	–	96	–
Average % Rgp-cl	–	–	34%	–
# solved	50	6	50	50
# fastest optimal	12	0	30	16

In Table 4.10, all the methods, except (M-BLP), were able to solve all 50 instances to optimality. The binary formulation with user cuts was fastest on 60% (30 out of 50) of the instances. This was primarily because the minimal cover inequalities closed about 34% of the root gap. `Couenne` was fastest on only 24% (12 out of 50) of the instances. Note that there exist some instances on which more than one method solved in shortest time.

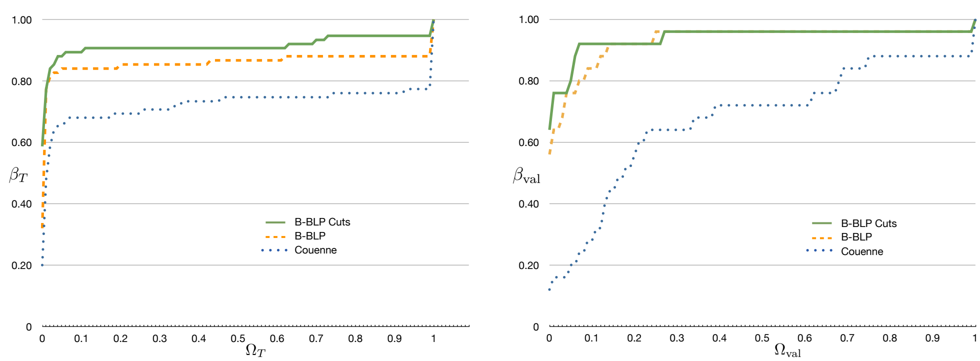
The instances in Table 4.11 are larger in size due to the higher values of κ_1 and κ_2. Once again, the binary formulation with user cuts solved the most number of instances to optimality (21 out of 50) and also in shortest time (14 out of 21). The root gap closed by our cuts was about 33%. The binary formulation (both with and

Table 4.11
`Disjoint` bilinear instances: $\kappa_1 = 3, \kappa_2 = 5$. Fifty random instances.

	Couenne	M-BLP	B-BLP + Cuts	B-BLP
Average Nds	1144272	51190	1686068	1855911
Average T (sec.)	3424	3605	2565	2773
Average Cuts	–	–	172	–
Average % Rgp-cl	–	–	33%	–
# solved	8	0	21	16
# fastest optimal	3	0	14	8
# best feasible	6	0	19	16
Average $\mu_{\mathcal{I}}$	2.11%	22%	2.21%	2.91%
Average $\omega_{\mathcal{I}}$	0.039%	0.261%	0.013%	0.016%

without cuts) produced the best feasible solution on most of the 25 instances that remained unsolved after 1 hour. Good quality feasible solutions were obtained after 1hr. by (B-BLP) with user cuts since the average $\omega_{\mathcal{I}} = 0.013\%$ was the least for this approach. User cuts reduced the average value of $\mu_{\mathcal{I}}$ to 2.21%, which was close to the average $\mu_{\mathcal{I}}$ of 2.11% obtained from `Couenne`.

The performance profiles are plotted in Figure 4.3. As seen in Tables 4.10 and 4.11, the performance of (M-BLP) is quite bad and is hence not plotted. From Figure 4.3(a) we see that our user cuts helped solved the largest fraction of instances (about 60%) in the shortest amount of time. (B-BLP) + user cuts obtains the best feasible solutions on about 65% of the unsolved instances in Figure 4.3(b). From both these plots we observe that our user cuts provide the most dominant performance profile.



(a) *CPU times for 75 solved instances.* (b) *Feasible solutions for 25 unsolved instances.*

Fig. 4.3. *Performance profiles for 100 `disjoint` bilinear problems. The metrics for* (M-BLP) *are not plotted since they were poor in comparison to the other three methods.*

5. Conclusion. In this study, we presented a MILP reformulation (B-BLP) for the mixed integer bilinear problem (BLP). The idea behind constructing the reformulation was to use binary expansion of general integer variables. We investigated this reformulation by conducting a polyhedral study in the extended space. The set of interest turned out to be a special case of the sequential knapsack polytope. A polynomial size description was provided for the convex hull of this set using a previous result on minimal covers of superincreasing knapsacks. We implemented our cuts on five sets of instances and compared their performance against (i) a MINLP solver for problem (BLP), and (ii) a branching scheme within a MILP solver for relaxation (M-BLP).

Our experiments suggest that the cuts were more effective for test instances with a bilinear objective function and linear constraints. Even if our cuts were not always successful in closing a significant amount of the root gap on general bilinear problems, they often helped branch-and-cut search deeper down the tree. The results lend credence to our primary motivation for this study: that on certain class of problems, adopting a MILP solution procedure for solving mixed integer bilinear problems can be beneficial. Finally, we emphasize that the cuts derived in this paper are by no means exhaustive and one may seek to derive additional valid inequalities by exploiting the structure of binary expansion within the constraints of a particular problem, thus potentially expanding the usefulness of this MILP approach to a wider class of problems.

Acknowledgements. The authors would like to thank the two anonymous referees for their helpful comments and suggestions. Particular thanks for encouraging us to experiment on more instances, helping us simplify the proof of Proposition 3.4, and for correcting an earlier version of Proposition 2.3. Thanks to Juan Pablo Vielma for providing the `watts` instances used in §4.1.2.

REFERENCES

[1] W.P. Adams and H.D. Sherali, *Mixed-integer bilinear programming problems*, Mathematical Programming, 59 (1993), pp. 279–305.

[2] N. Adhya, M. Tawarmalani, and N.V. Sahinidis, *A Lagrangian approach to the pooling problem*, Industrial and Engineering Chemistry Research, 38 (1999), pp. 1956–1972.

[3] F.A. Al-Khayyal and J.E. Falk, *Jointly constrained biconvex programming*, Mathematics of Operations Research, 8 (1983), pp. 273–286.

[4] K.M. Anstreicher, *On convex relaxations for quadratically constrained quadratic programming*, Mathematical Programming, 136 (2012), pp. 233–251.

[5] C. Audet, J. Brimberg, P. Hansen, S. Le Digabel, and N. Mladenović, *Pooling problem: Alternate formulations and solution methods*, Management Science, 50 (2004), pp. 761–776.

[6] E. Balas, *Disjunctive programming: Properties of the convex hull of feasible points*, Discrete Applied Mathematics, 89 (1998), pp. 3–44.

[7] P. Belotti, *Couenne: a user's manual*, June 2012. Available online at COIN-OR: https://projects.coin-or.org/Couenne/browser/trunk/Couenne/doc/.

[8] P. Belotti, J. Lee, L. Liberti, F. Margot, and A. Wächter, *Branching and bounds tightening techniques for non-convex MINLP*, Optimization Methods and Software, 24 (2009), pp. 597–634.

[9] A. Ben-Tal, G. Eiger, and V. Gershovitz, *Global minimization by reducing the duality gap*, Mathematical Programming, 63 (1994), pp. 193–212.

[10] D. Bienstock, *Histogram models for robust portfolio optimization*, Journal of Computational Finance, 11 (2007), pp. 1–64.

[11] M.R. Bussieck, A.S. Drud, and A. Meeraus, *MINLPLib - A collection of test models for mixed-integer nonlinear programming*, INFORMS Journal on Computing, 15 (2003), pp. 114–119.

[12] A. Caprara and M. Monaci, *Bidimensional packing by bilinear programming*, Mathematical Programming, 118 (2009), pp. 75–108.

[13] C. D'Ambrosio, A. Frangioni, L. Liberti, and A. Lodi, *Experiments with a feasibility pump approach for nonconvex MINLPs*, in Proceedings of the 9th Symposium on Experimental Algorithms (SEA 2010), P. Festa, ed., vol. 6049 of Lecture Notes in Computer Science, Springer, 2010, pp. 350–360.

[14] E. Dolan and J. Moré, *Benchmarking optimization software with performance profiles*, Mathematical Programming, 91 (2002), pp. 201–213.

[15] C.A. Floudas and V. Visweswaran, *A global optimization algorithm (GOP) for certain classes of nonconvex NLPs–I. Theory*, Computers and Chemical Engineering, 14 (1990), pp. 1397–1417.
[16] L.R. Foulds, D. Haugland, and K. Jörnsten, *A bilinear approach to the pooling problem*, Optimization, 24 (1992), pp. 165–180.
[17] A.S. Freire, E. Moreno, and J.P. Vielma, *An integer linear programming approach for bilinear integer programming*, Operations Research Letters, 40 (2012), pp. 74–77.
[18] F. Glover, *Improved linear integer programming formulations of nonlinear integer problems*, Management Science, 22 (1975), pp. 455–460.
[19] I. Harjunkoski, T. Westerlund, R. Pörn, and H. Skrifvars, *Different transformations for solving non-convex trim loss problems by MINLP*, European Journal of Operational Research, 105 (1998), pp. 594–603.
[20] M.M. Hasan and I.A. Karimi, *Piecewise linear relaxation of bilinear programs using bivariate partitioning*, AIChE Journal, 56 (2010), pp. 1880–1893.
[21] S. Henry, *Tight Polyhedral Representations of Discrete Sets using Projections, Simplices, and Base-2 Expansions*, PhD thesis, Clemson University, 2011.
[22] R. Karuppiah and I.E. Grossmann, *Global optimization for the synthesis of integrated water systems in chemical processes*, Computers and Chemical Engineering, 30 (2006), pp. 650–673.
[23] M. Laurent and A. Sassano, *A characterization of knapsacks with the max-flow-min-cut property*, Operations Research Letters, 11 (1992), pp. 105–110.
[24] G.P. McCormick, *Computability of global solutions to factorable nonconvex programs: Part I. convex underestimating problems*, Mathematical Programming, 10 (1976), pp. 147–175.
[25] C.A. Meyer and C.A. Floudas, *Global optimization of a combinatorially complex generalized pooling problem*, AIChE journal, 52 (2006), pp. 1027–1037.
[26] R. Misener and C.A. Floudas, *Advances for the pooling problem: Modeling, global optimization, and computational studies*, Appl. Comput. Math, 8 (2009), pp. 3–22.
[27] J.H. Owen and S. Mehrotra, *On the value of binary expansions for general mixed-integer linear programs*, Operations Research, 50 (2002), pp. 810–819.
[28] V. Pham, C. Laird, and M. El-Halwagi, *Convex hull discretization approach to the global optimization of pooling problems*, Industrial and Engineering Chemistry Research, 48 (2009), pp. 1973–1979.
[29] Y. Pochet and R. Weismantel, *The sequential knapsack polytope*, SIAM Journal on Optimization, 8 (1998), pp. 248–264.
[30] I. Quesada and I.E. Grossmann, *Global optimization of bilinear process networks with multicomponent flows*, Computers and Chemical Engineering, 19 (1995), pp. 1219–1242.
[31] G. Rinaldi, U. Voigt, and G.J. Woeginger, *The mathematics of playing golf, or: a new class of difficult non-linear mixed integer programs*, Mathematical Programming, 93 (2002), pp. 77–86.
[32] H.D. Sherali and A. Alameddine, *A new reformulation-linearization technique for bilinear programming problems*, Journal of Global Optimization, 2 (1992), pp. 379–410.
[33] M. Tawarmalani, J-P.P. Richard, and K. Chung, *Strong valid inequalities for orthogonal disjunctions and bilinear covering sets*, Mathematical Programming, 124 (2010), pp. 481–512.
[34] M. Tawarmalani and N.V. Sahinidis, *Convexification and global optimization in continuous and mixed-integer nonlinear programming: theory, algorithms, software, and applications*, Kluwer Academic Publishers, 2002.
[35] D. Vandenbussche and G.L. Nemhauser, *A branch-and-cut algorithm for nonconvex quadratic programs with box constraints*, Mathematical Programming, 102 (2005), pp. 559–575.
[36] L.N. Vicente, P.H. Calamai, and J.J. Júdice, *Generation of disjointly constrained bilinear programming test problems*, Computational Optimization and Applications, 1 (1992), pp. 299–306.
[37] J.M. Zamora and I.E. Grossmann, *A branch and contract algorithm for problems with concave univariate, bilinear and linear fractional terms*, Journal of Global Optimization, 14 (1999), pp. 217–249.