Automatic Reformulation of Bilinear MINLPs

LEO LIBERTI

DEI, Politecnico di Milano, P.zza L. da Vinci 32, 20133 Milano, Italy

(liberti@elet.polimi.it)

June 27, 2004

Abstract

Solving Mixed-Integer Nonlinear problems (MINLPs) involving bilinear terms is a very hard task, both in terms of complexity measures and practically. Such problems occur in a variety of interesting real-life applications; many graph-theoretical problem can also be formulated as continuous nonconvex models. Obtaining tight lower bounds to these problems is useful for two reasons: (a) optimal solution via Branch-and-Bound algorithms; (b) estimation of heuristic solution quality. We derive tight relaxations of MINLPs which involve bilinear terms and linear equality constraints by automatically reformulating the problems to an equivalent form with less bilinear terms.

1 Introduction

In this paper we show how to derive tight convex relaxations of MINLPs involving bilinear terms and linear equation constraints in the following formulation:

$$\left.\begin{array}{rrcl} \min\limits_{x\in\mathbb{R}^n} & x^\top Qx & + & a^\top x \\ & Ax & = & b \\ & A'x & \le & b' \\ & g(x) & \le & 0 \\ & x^L \le & x & \le x^U \\ \forall\, i \in \mathbb{I} & (x_i & \in & \mathbb{Z}), \end{array}\right\} \quad (1)$$

where Q is a symmetric matrix, $a, x^L, x^U \in \mathbb{R}^n$, A is an $m \times n$ matrix with full rank $m > 0$, A' is $m' \times n$, $b \in \mathbb{R}^m$, $b' \in \mathbb{R}^{m'}$, $g : \mathbb{R}^n \to \mathbb{R}^{m''}$ are nonlinear inequalities containing any number of bilinear terms in the x variables, and $\mathbb{I}$ is a subset of $\{1, \ldots, n\}$. Thus, problem (1) is the most general form of equality-constrained mixed-integer nonlinear problem (MINLP) involving bilinear terms.

Such problems occur frequently in many application fields, ranging from continuous or mixed-integer problems arising in engineering fields [ATS99, TS02b, ABH$^+$02, GK97] to graph theory [BBPP98, ABP01, Ans03]. In Section 7 we show the application of our techniques to various classes of problems, namely to:

- the Pooling and Blending problem (PBP);
- the Quadratic Knapsack problem (QKP);
- the Quadratic Assignment problem (QAP);
- the Multi-Processor Scheduling problem with Communication Delays (MSPCD).

The reformulation is achieved by replacing some of the bilinear terms with appropriately generated linear constraints. The following example shows the basic idea on which our reformulation is based.

1.1 Example

Consider the problem $\min\{xy \mid x = 1, 0 \le x \le 2, -2 \le y \le 2\}$, *which has a bilinear term* xy *in the objective function and a linear equality constraint* $x = 1$. *It is clear that the reformulation* $\min\{w \mid w = xy, x = 1, 0 \le x \le 2, -2 \le y \le 2, -4 \le w \le 4\}$, *obtained by introducing a variable* w *and the constraint* $w = xy$, *is equivalent to the original problem. The feasible region is* $F \cap \{(x, y, w) \mid 0 \le x \le 2, -2 \le x \le 2, -4 \le w \le 4\}$, *where* $F = \{(x, y, w) \mid w = xy, x = 1\}$. *It is easy to see from Fig. 1 that the feasible region, although expressed in terms of a bilinear term and a linear equality constraint, is in fact linear. This fact can be made explicit in the definition of the set* F *as follows: multiply the linear constraint*

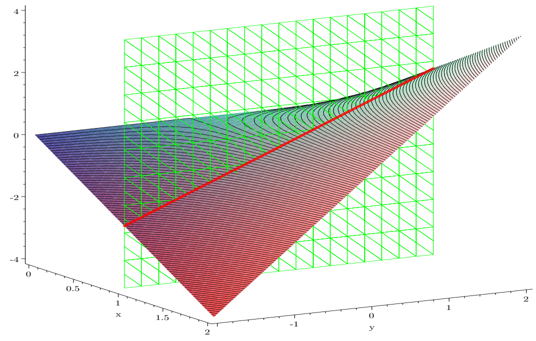

Figure 1: The feasible region of Example 1.1 defined by $w = xy, x = 1$.

$x = 1$ *by the variable* y, *to obtain the new constraint* $xy = y$; *since* $xy = w$, *this becomes a linear equation constraint* $w - y = 0$. *The corresponding plane can be used to replace the constraint* $w = xy$ *in the definition of* F, *so that* $F = \{(x, y, w) \mid w - y = 0, x = 1\}$. *This can be verified by inspection of Fig. 2.*

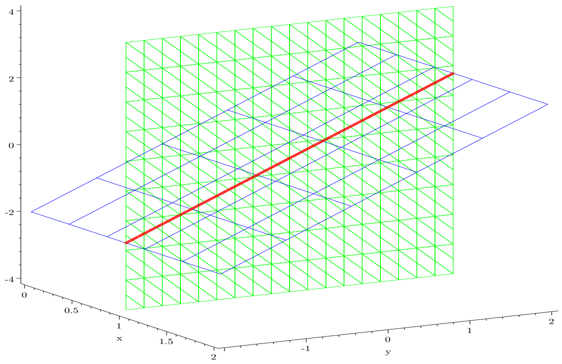

Figure 2: The feasible region of Example 1.1 can be expressed linearly by $w - y = 0, x = 1$.

Although the example above is too trivial to motivate the ideas in this paper, it nonetheless gives an insight into what we are trying to achieve. We shall see in what follows how the geometrical property which allows the replacement of bilinear terms with linear equality constraints can be applied in full generality.

The method we propose relies on a procedure which is similar to the Reformulation-Linearization Technique (RLT) [She02] applied to continuous bilinear problems [SA92]. In fact, the linear constraints that replace the bilinear terms are RLT-type constraints. The RLT linearizes the bilinear terms in the problem and generates a number of linear equations and inequalities by considering multiplications of bound factors and constraint factors. In a sequence of papers published from the 1980s onwards (see for example [SA86, SA92, ST97, SA99, She98, SW01]), it was shown that a full application of the RLT generates the convex envelope of any nonconvex problem (be it continuous or integer). This result finds its practical limitation in the extremely large number of added constraints necessary to achieve not only the convex envelope, but even a reasonably tight convex relaxation. A number of heuristic techniques was proposed in the papers cited above to help filter out RLT constraints which are not valid. We address this drawback by generating only those linear constraints which are guaranteed to replace a bilinear term in the problem. As it turns out, the set of replaced bilinear terms is usually not unique. We also explain in Section 6 how to choose the most convenient such set in terms of the volume gap between the original problem and its convex relaxation.

2 Linearizing the bilinear terms

As in Example 1.1, the first step is to reformulate the original problem (1) so that each bilinear terms $x_i x_j$ is substituted by a new variable w_i^j and a corresponding *defining constraint* $w_i^j = x_i x_j$ is added to the problem formulation. By commutativity, we shall consider w_i^j and w_j^i the same added variable, and write w_i^j with $i \le j$. Consider the set β of index pairs $\{i,j\}$ such that the bilinear term $x_i x_j$ appears in the original problem (1). This substitution process results in the following reformulation:

$$\left.\begin{array}{rrcl}
\min\limits_{x,w} & p^\top w & + & a^\top x \\
 & Ax & = & b \\
 & A'x & \le & b' \\
 & g'(x,w) & \le & 0 \\
\forall\, \{i,j\} \in \beta & w_i^j & = & x_i x_j \\
 & x^L \le & x & \le x^U \\
 & w^L \le & w & \le w^U \\
\forall\, i \in \mathbb{I} & (x_i & \in & \mathbb{Z}),
\end{array}\right\} \qquad (2)$$

where $w \in \mathbb{R}^{|\beta|}$ is the vector of added variables w_i^j, $p^\top w$ is the linear form obtained by the bilinear form $x^\top Q x$ where each bilinear term $x_i x_j$ has been substituted by the corresponding variable w_i^j, g' has been obtained from g via the same substitution, and $w^L, w^U \in \mathbb{R}^{|\beta|}$ are the variable bounds on w, obtained by simple interval arithmetic from the bounds on the x variables. Clearly, the projection of reformulation (2) to the original problem space results in the original problem itself (this reformulation is therefore *exact*).

3 Reduction constraints

In this section we shall give a theoretical foundation to the concept of replacing a bilinear term with a linear constraint called a reduction constraint. Reduction constraints have been discussed in [Lib04a, LP04, Lib04b, Lib03]; in particular, it was shown in [Lib03] that reduction constraints are a subset of RLT constraints. Theorem 3.1 shows how some of the defining constraints $w_i^j = x_i x_j$ can be implied by a set of linear equations (reduction constraints) relating the w and the x variables. Thus, they can be replaced by the reduction constraints without changing the feasible region.

Let β' be the set of *all possible* index pairs $\{i,j\}$ leading to distinct bilinear terms $x_i x_j$. Consider the following augmented reformulation:

$$\left.\begin{array}{rrcl}
\min\limits_{x,w} & p^{\mathsf{T}}w & + & a^{\mathsf{T}}x \\
 & Ax & = & b \\
 & A'x & \le & b' \\
 & g'(x,w) & \le & 0 \\
\forall\,\{i,j\}\in\beta' & w_i^j & = & x_ix_j \\
 & x^L \le & x & \le x^U \\
 & w^L \le & w & \le w^U \\
\forall\, i\in\mathbb{I} & (x_i & \in & \mathbb{Z}),
\end{array}\right\} \qquad (3)$$

obtained from reformulation (2) by replacing β with β'. This reformulation, although it might have many more bilinear terms than the original problem, is clearly still exact.

We now multiply the linear equation constraints $Ax = b$ by each of the variables $x_1, \ldots, x_n$, to obtain the system of equations $\forall i \le n\ ((Ax)x_i = bx_i)$. If we define $w_i = (w_i^1, \ldots, w_i^n)^{\mathsf{T}}$, we can substitute each x_ix_j by the corresponding w_i^j to obtain the linear system of equations

$$\forall\, i \le n\ (Aw_i - bx_i = 0). \qquad (4)$$

We call system (4) above a *reduction constraint system* (RCS) and each equation in the system a *reduction constraint* (RC). Note that RCSs have mn equations and $\frac{1}{2}n(n+1)$ unknowns. By construction, an RCS is redundant with respect to the feasible region of problem (3). In other words, adding an RCS to the feasible region does not change it. It does, however, make some of the w defining constraints redundant as well, thus allowing their deletion.

Observe that $b = Ax$, so the RCS (4) is equivalent to $\forall i \le n\ (Aw_i - (Ax)x_i) = 0$, i.e. to $\forall i \le n\ (A(w - x_ix) = 0)$. For each $\{i,j\} \in \beta'$, let $z_i^j = w_i^j - x_ix_j$, and let $z_i = (z_i^1, \ldots, z_i^n)^{\mathsf{T}}$. System (4) is then equivalent to

$$\forall\, i \le n\ (Az_i = 0). \qquad (5)$$

We call system (5) the *companion system* for the RCS (4). Each RCS is equivalent to the corresponding companion system. Let B, N be sets of index pairs $\{i,j\}$ such that $\{z_i^j \mid \{i,j\} \in B\}$ is a set of basic variables of the companion system and $\{z_i^j \mid \{i,j\} \in N\}$ is the corresponding set of nonbasic variables. Notice B, N form a partition of β'.

We can express the feasible region of problem (3) as $F \cap R$, where F is given by the linear equation constraints $Ax = b$ and the w defining constraints $w_i^j = x_ix_j$ for all index pairs $\{i,j\}$ in β', and R is given by the other constraints.

3.1 Theorem ([Lib03])
Let

$$\begin{array}{rcl}
F & = & \{(x,w) \mid Ax = b,\ \forall\{i,j\} \in \beta'\ (w_i^j = x_ix_j)\}, \\
F' & = & \{(x,w) \mid Ax = b,\ \forall i \le n\ (Aw_i - bx_i = 0),\ \forall\{i,j\} \in N\ (w_i^j = x_ix_j)\}.
\end{array}$$

Then $F = F'$.

Theorem 3.1 shows that it suffices to impose the w defining constraints $w_i^j = x_ix_j$ for a set of pairs $\{i,j\}$ corresponding to nonbasic variables z_i^j of the companion system. The other defining constraints are implied by the RCS. In other words, the RCS replaces the bilinear terms x_ix_j corresponding to basic variables. The following reformulation, called the *reduction constraint reformulation*, is therefore exact:

$$\left.\begin{array}{rrcl}
\min\limits_{x,w} & p^\mathsf{T} w & + & a^\mathsf{T} x \\
 & Ax & = & b \\
\forall i \le n & Aw_i - bx_i & = & 0 \\
 & A'x & \le & b' \\
 & g'(x,w) & \le & 0 \\
\forall \{i,j\} \in N & w_i^j & = & x_i x_j \\
 & x^L \le & x & \le x^U \\
 & w^L \le & w & \le w^U \\
\forall\, i \in \mathbb{I} & (x_i & \in & \mathbb{Z}).
\end{array}\right\} \qquad (6)$$

Notice that for most linear systems of equations, the partition of the variables in basic and nonbasic is not unique. Thus, there are many possible exact reformulations (6): one for each set N of nonbasic variables of the companion system. We shall explain in Section 6 how to choose the nonbasic set so that the convexity volume gap is minimized.

4 Reduction constraints in sparse bilinear problems

Recall that β is the set of bilinear terms in the original problem expressed as in (2). Generation of RCs is certainly useful if it reduces the number of bilinear terms in the problem, i.e. if the following requirement

$$|N| < |\beta| \qquad (7)$$

holds: any convex relaxation of the problem in form (6) is bound to be tighter than a convex relaxation of the problem in the original form (1), since it relaxes fewer nonconvex terms.

By Theorem 3.1, the number of bilinear terms that can be replaced by RCs is equal to the rank ρ of the RCS. Computational experiments seem to indicate that ρ is always strictly less than the number of variables $\frac{1}{2}n(n+1)$ in the RCS. This would imply that N is never the empty set. In particular, if the original problem has few bilinear terms, requirement (7) might not be satisfied. In practice, this occurrence is not rare. In the remainder of this section, we show some methods to address this problem.

Let J be a set of indices such that for all $j \in J$ the variable x_j appears with nonzero coefficient in at least one equation of $Ax = b$. Let I be a set of indices such that if $i \in I$, then for all $j \in J$ we have $\{i,j\} \in \beta$. We can construct a *restricted reduction constraint system* (RRCS) with respect to I, J by multiplying $Ax = b$ by each variable x_i with $i \in I$:

$$\forall i \in I\ (A\bar{w}_i - bx_i = 0),$$

where $\bar{w}_i$ is the vector with components w_i^j for each $j \in J$. Theorem 3.1 can easily be adapted to this new setting by replacing β' with β; in particular, this yields $|N| < |\beta|$. Thus, if a RRCS can be generated for a given problem, it is guaranteed to reduce the number of bilinear terms in the original problem, since requirement (7) is automatically satisfied. Notice that the index sets I, J have been chosen so that, when multiplying system $Ax = b$ by problem variables x_i with $i \in I$, the bilinear terms involved in this operation are exactly those corresponding to index pairs $(i,j) \in \beta$.

4.1 Example (Example 1.1 revisited)
In Example 1.1, the system $Ax = b$ only consists of the constraint $x = 1$. Setting $J = \{1\}$ and $I = \{2\}$ implies multiplying $x = 1$ by y, which yields the RRCS $w - y = 0$. This linear constraint replaces the bilinear defining constraint $w = xy$.

The following example shows a case where it is impossible to generate a RRCS as defined above. We tackle this case by restricting the set of linear equation constraints considered for generation of the RRCS.

In other words, reduction constraints systems can be restricted in the subset of multiplier variables as well as in the subset of linear constraints considered for RC generation.

4.2 Example
Consider the following problem.

$$\left.\begin{array}{llcl} \min\limits_{x_1,\ldots,x_4\geq 0} & x_1^2 + x_1x_2 - x_3x_4 & & \\ & x_1+x_2+x_3+x_4 & = & 1 \\ & x_1+2x_2 & = & 1 \end{array}\right\}$$

The companion system for the RCS of this problem, in row echelon form, is

$$\begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & -1 & 0 & -1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & -1 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} z = 0,$$

where $z = (z_1^1, z_1^2, z_1^3, z_1^4, z_2^2, z_2^3, z_2^4, z_3^3, z_3^4, z_4^4)^\mathsf{T}$. *By inspection,* $|N| = |\{z_3^3, z_3^4, z_4^4\}| = 3$; *since* $|\beta| = |\{\{1,1\},\{1,2\},\{3,4\}\}| = 3$, *requirement (7) does not hold. If we define* $J = \{1,2,3,4\}$ *(indices of variables appearing with nonzero coefficients in the linear constraints), it is clear than any nontrivial choice of* I *will produce a RRCS w.r.t.* I, J *involving at least a bilinear term* x_ix_j *with* $\{i,j\} \notin \beta$ *(in other words, multiplying the linear equations by any non-empty subset of problem variables will generate new bilinear terms which are not in the original problem formulation); so RRCSs, as defined above, cannot be applied.*

Notice, however, that if we only consider equation $x_1 + 2x_2 = 1$ *and we multiply it by* x_1 *and* x_2, *we obtain the following RRCS:*

$$\begin{array}{rcl} w_1^1 + 2w_1^2 - x_1 & = & 0 \\ w_1^2 + 2w_2^2 - x_2 & = & 0, \end{array}$$

with corresponding companion system

$$\begin{pmatrix} 1 & 2 & 0 \\ 0 & 1 & 2 \end{pmatrix} \bar{z} = 0,$$

where $\bar{z} = (z_1^1, z_1^2, z_2^2)^\mathsf{T}$. *If we now add the defining constraint* $w_2^2 = x_2^2$ *to the problem formulation, we can use the above RRCS w.r.t.* $I = \{1,2\}$, $J = \{1,2\}$ *to replace both bilinear terms* x_1^2, x_1x_2 *in the objective function. The following reformulation is therefore exact:*

$$\left.\begin{array}{llcl} \min\limits_{x,w\geq 0} & w_1^1 + w_1^2 - w_3^4 & & \\ & x_1+x_2+x_3+x_4 & = & 1 \\ & x_1+2x_2 & = & 1 \\ & w_1^1 + 2w_1^2 - x_1 & = & 0 \\ & w_1^2 + 2w_2^2 - x_2 & = & 0 \\ & w_3^4 & = & x_3x_4 \\ & w_2^2 & = & x_2^2. \end{array}\right\}$$

This reformulation contains only two bilinear terms, whereas the original problem contained three bilinear terms. Observe that in order to arrive at this reduction, we first increased the number of bilinear terms by introducing the term x_2^2, *then we looked for a subset of the linear constraints (corresponding to the set* J*) and a subset of the variables (corresponding to the set* I*) such that the resulting RRCS could be used to replace more bilinear terms than we needed to add.*

The example above shows that by carefully adjusting I, J and the subset of linear equality constraints considered for RC generation, it is possible to obtain RRCSs that can reduce the number of bilinear terms in the problem even when ordinary RCSs would not work. We therefore generalize the concept of RRCS to depend on I, J and a subset K of $\{1,\ldots,m\}$ of linear equality constraint indices. J is the subset of variable indices occurring with nonzero coefficients in the constraints indexed by K, and I is the maximal subset of variable indices such that the cardinality of $\gamma = \{\{i,j\} \mid i \in I, j \in J\}$ is strictly less than $|\beta|$. Here γ corresponds to the set of bilinear terms required to define the RRCS. Let $A_K x = b_K$ be the set of linear equality constraints indexed by K. A RRCS w.r.t. I, J, K is a reduction constraint system obtained by multiplying the subsystem $A_K x = b_K$ by all problem variables in x_i with $i \in I$. Since the number of bilinear terms reduced depends on the rank of the companion system of the RRCS, we aim to maximize $|K|$ subject to I (and hence γ) being non-empty.

Algorithmically, requirement (7) is difficult to verify, since it involves calculating a set of basic variables for each candidate RRCS. For this reason, we consider a relaxation of (7) that requires that a candidate RRCS should have more constraints than the number of new bilinear terms needed to define it. This relaxation rests on the experimental observation that on average, each new RC can be used to replace a bilinear term; so far, the results obtained by following this observation have been very good. Whilst requirement (7) is exact, however, we are not claiming that its relaxation is. Two fundamentally different algorithms for identifying RRCSs w.r.t. "adjusted" I, J, K are presented in [Lib03, LP04]. Both of them allow an increase in the number of bilinear terms if the subsequent RRCS contains strictly more reduction constraints than the number of new bilinear terms introduced in the problem.

5 Convex relaxation

Replacement of bilinear terms by RCs, as described in Sections 3 and 4, reformulates the original problem (1) to:

$$\left.\begin{array}{rrcl}
\min\limits_{x,w} & p^\top w & + & a^\top x \\
 & Ax & = & b \\
 & A'x & \leq & b' \\
 & g'(x,w) & \leq & 0 \\
 & (x,w) & \in & S \\
 & x^L \leq & x & \leq x^U \\
 & w^L \leq & w & \leq w^U \\
\forall\, i \in \mathbb{I} & (x_i & \in & \mathbb{Z}),
\end{array}\right\} \qquad (8)$$

where S is the set given by:

- an appropriate RRCS
- the defining constraints $w_i^j = x_i x_j$ where $\{z_i^j\}$ is a set of nonbasic variables of the corresponding restricted companion system.

We obtain a convex relaxation for problem (8) by applying the convexification methods described in [Smi96, SP99]. This first involves linearizing each nonconvex term by replacing it with an added variable w_k, and adding the corresponding defining constraint "w_k = nonconvex term" to the formulation (much in the same way as we did for bilinear terms in earlier sections). This process results in an exact reformulation of the original problem, which consists of a linear objective function, a set of linear constraints, and a set of simple defining constraints for each linearized variable w_k. At this stage the NLP is in *standard form*. An NLP in standard form is much more amenable to algorithmic symbolic manipulation than the original problem, because the data structures required to store a list of defining constraints are simpler than those required to store a general nonlinear equation. A convex relaxation of an NLP in standard form can be obtained by replacing each nonconvex defining constraint by a convex underestimating inequality and a concave overestimating inequality.

Tight convex underestimators and concave overestimators for many simple nonconvex terms exist in the literature [SP99, ADFN98, TS02c, Flo00]. In particular, extensive work has been carried out on bilinear terms [McC76, AKF83, SA92], linear fractional terms [ZG99, TS01, TS02a], piecewise convex and concave terms [LP03], polynomials [MF95, RS03, She98, GJS02], signomials [BLW03], and general nonconvex terms [AF96, Adj98].

In particular, we are concerned with the convex relaxation of defining constraints for bilinear terms, thus we deal with the cases $w_i^j = x_i x_j$, $w_i^i = x_i^2$ for suitable $i, j \leq n$. A linear relaxation is available for the former defining constraint, due to McCormick [McC76] and proven to be the envelope by Al-Khayyal and Falk [AKF83]:

$$\left.\begin{array}{rcl} w_i^j & \geq & x_i^L x_j + x_j^L x_i - x_i^L x_j^L \\ w_i^j & \geq & x_i^U x_j + x_j^U x_i - x_i^U x_j^U \\ w_i^j & \leq & x_i^L x_j + x_j^U x_i - x_i^L x_j^U \\ w_i^j & \leq & x_i^U x_j + x_j^L x_i - x_i^U x_j^L, \end{array}\right\} \tag{9}$$

where x_i^L, x_i^U are the bounds on the variable x_i. As for quadratic defining constraints of the form $w_i^i = x_i^2$, the convex/concave envelope is clearly provided by the function x_i^2 itself as the underestimator, and a chord as the overestimator:

$$\left.\begin{array}{rcl} w_i^i & \geq & x_i^2 \\ w_i^i & \leq & (x_i^L)^2 + \frac{(x_i^U)^2 - (x_i^L)^2}{x_i^U - x_i^L}(x_i - x_i^L). \end{array}\right\} \tag{10}$$

Notice that reformulation (8) is exact as long as the defining constraints $w_i^j = x_i x_j$ corresponding to the nonbasics N of the companion system are present in the formulation. When we replace these constraints by their convex relaxations, the reduction constraints fail to algebraically imply the bilinear terms corresponding to the basic variables B in the companion system. Therefore, it is possible that the convex relaxation of one of the bilinear terms B might be a valid cut to the convex relaxation of the problem. This shortcoming can be dispensed with by also including the convex relaxations of the bilinear terms B. In our computational experience, however, we have found no instance where this was necessary. Moreover, we were unable to construct a simple example showing this occurrence.

6 Reducing the convexity volume

Recall that in Section 4 we introduced new bilinear terms in the problem in order to be able to replace more bilinear terms with RCs than we introduced. This does in fact achieve the goal of reducing the overall number of bilinear terms in the problem. The ultimate goal of our reformulation methods, however, is to generate a tight convex relaxation of the problem. In this respect, the introduction of new bilinear terms may lead to trouble. Given a nonconvex term and its convex relaxation, the convexity gap between the nonconvex term and the relaxation is a measure of how tight the relaxation is. Suppose, as in Example 4.2, that we are introducing the term x_2^2 and replacing terms x_1^2, x_1x_2 with suitable RCs. It might happen that the convexity gap relative to x_2^2 is "larger" than the combined convexity gaps deriving from x_1^2 and x_1x_2. Thus, even though we have in fact reduced the number of bilinear terms in the problem, overall we made things worse by generating a looser convex relaxation than the one we started with. To avoid this situation, we must be careful in the choice of the new bilinear terms introduced in the problem.

One further motivation for an in-depth analysis of the convexity gap between a nonconvex term and its relaxation is that, as has been observed at the end of Section 3, the set N of nonbasic variables of a linear equation system is usually not unique. Since by Theorem 3.1 N determines the bilinear terms that should remain in the problem after the reformulation, it makes sense to choose N so that the total convexity gap is minimized. This further tightens the convex relaxation.

We start by introducing a more precise definition of the convexity gap. Consider a function $f : X \subseteq \mathbb{R}^n \to \mathbb{R}$. Let $\underline{f}(x)$ be a convex lower bounding function for f and $\bar{f}(x)$ be a concave upper bounding function for f. Then the set $\bar{S} = \{(x,w) \mid \mathit{underline} f(x) \le w \le \bar{f}(x)\}$ is a convex relaxation of the set $S = \{(x,w) \mid w = f(x)\}$. We define the *convexity gap* $V(S)$ between S and $\bar{S}$ to be the volume of the set $\bar{S}$; namely,

$$V(S) = \int_{x \in X} (\bar{f}(x) - \underline{f}(x))dx.$$

We denote the convexity volume for a bilinear term $x_i x_j$ with V_i^j.

6.1 Convexity volume for a quadratic term x_i^2

As has been mentioned in Section 5, the convex relaxation of the set $\xi = \{(x_i, w_i^i) \mid w_i^i = x_i^2, x_i^L \le x_i \le x_i^U\}$ (where $i \le n$) consists of the area between the function x_i^2 and the chord, as shown in Fig. 3. The

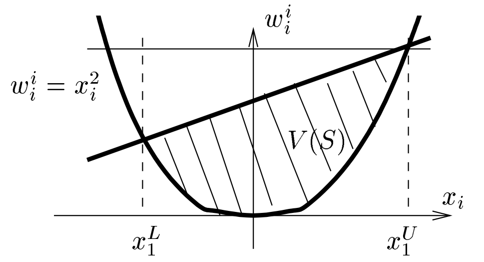

Figure 3: Convex relaxation of the defining constraint $w_i^i = x_i^2$.

convexity volume of ξ is given by:

$$V_i^i = \int_{x_i^L}^{x_i^U} \left((x_i^L)^2 + \frac{(x_i^U)^2 - (x_i^L)^2}{x_i^U - x_i^L}(x_i - x_i^L) - x_i^2 \right) dx = \frac{1}{6}(x_i^U - x_i^L)^3.$$

6.2 Convexity gap for a bilinear term $x_i x_j$

It was shown in [AKF83] that the set defined by the linear inequalities in system (9) is the convex envelope of the set $\zeta = \{(x_i, x_j, w_i^j) \mid w_i^j = x_i x_j, x_i^L \le x_i \le x_i^U, x_j^L \le x_j \le x_j^U\}$ (where $i \le j \le n$). System (9) defines a tetrahedron in $\mathbb{R}^3$ whose vertices are (x_i^U, x_j^U), (x_i^U, x_j^L), (x_i^L, x_j^U), (x_i^L, x_j^L) (see Fig. 4). The volume $\tau(a,b,c,d)$ of a tetrahedron with vertices (a,c),(a,d),(b,c),(b,d) can be calculated

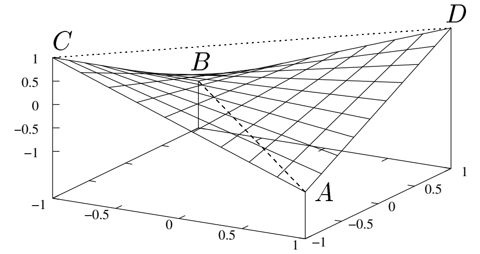

Figure 4: Tetrahedron $ABCD$: convex (linear) envelope of the defining constraint $w_i^j = x_i x_j$.

through elementary geometry, and it turns out to be

$$\frac{(b-a)(d-c)\sqrt{(b^2+a^4-(c(b-a))^2a^2-2bc^2a+a^2c^2b^2-2a^3c^2b-2ba+b^2c^2+a^2c^2+a^2b^2-2a^3b+a^4c^2+a^2)(d-c)^2}}{6\sqrt{a^2+c^2+1}}.$$

In order to obtain this formula, we computed the area $h = \frac{1}{2}\sqrt{||CD||^2||BD||^2 - (CD \cdot BD)^2}$ of the triangle BCD and the distance $l = (ad + bc - ac - bd)/\sqrt{b^2 + d^2 + 1}$ of A from the plane containing BCD. The formula for h was found by choosing BD as the base of the triangle BCD and computing the length of the segment from C perpendicular to BD. The formula for l was derived from the usual formula of the distance from a point $x^* = (x_1^*, x_2^*, x_3^*)$ to a plane $\alpha_1 x_1 + \alpha_2 x_2 + \alpha_3 x_3 + \alpha_4$, given by $|\alpha_1 x_1^* + \alpha_2 x_2^* + \alpha_3 x_3^* + \alpha_4|/\sqrt{\alpha_1^2 + \alpha_2^2 + \alpha_3^2}$. The volume of the tetrahedron is then given by $\frac{1}{3}hl$. The formula for $\tau(a, b, c, d)$ was obtained by running the MAPLE V commands below. Tested with the calculation of the volume of the tetrahedron defined by $a = -1, b = 1, c = -1, d = 1$ (MAPLE command: "`evalf(vol(-1,1,-1,1));`"), the script returned 2.666, which is (correctly) a third of the volume of the cube with side length 2.

```
f   := (p,q) -> (p[1]-q[1])^2 + (p[2]-q[2])^2 + (p[3] - q[3])^2; // distance
vtx := (x,y) -> vector([x,y,x*y]);             // vector on bilinear surface
A   := vtx(xL,yL);                                                // vertices
B   := vtx(xL,yU);
C   := vtx(xU,yL);
DD  := vtx(xU,yU);                        // 'D' is already a protected name
with(linalg):
h   := factor(sqrt(f(DD,C)*f(B,DD)-(dotprod(DD-C,B-DD))^2));       // 2*area
l   := (yU*xL+xU*yL-xL*yL-xU*yU)/(sqrt(yU^2+xU^2+1));              // height
vol := unapply(abs(factor(l*h)/6),xL,xU,yL,yU);                    // volume
```

The convexity volume of ζ is thus given by $V_i^j = \tau(x_i^U, x_i^L, x_j^U, x_j^L)$.

6.3 Tightness test for the reduction constraint reformulation

Equipped with closed-form expressions for the convexity volume, we can now perform a test to verify that the application of reduction constraints to a particular problem has indeed been successful. Let N be the set of index pairs $\{i, j\}$ present in the reduction constraint reformulation (8). If

$$V_N = \sum_{\{i,k\} \in N} V_i^k < \sum_{\{i,k\} \in \beta} V_i^k, \tag{11}$$

then the total convexity volume of the reformulation is smaller than that of the original problem. In Eq. (11), β is the set of index pairs (i, k) corresponding to bilinear terms $x_i x_k$ present in the original problem; N is the set indices (i, k) of nonbasic variables z_i^k of the restricted companion system. The set γ of bilinear terms necessary to define the RRCS (i.e. all (i, j) such that w_i^j appears in the RRCS) might be larger than β, as described in Sect. 4. Thus N is not necessarily a subset of β.

6.4 Choosing a convenient basis for the companion system

There is often more than one way to choose a set of nonbasic variables of a linear system of equations. Thus, when choosing the set of nonbasic variables N of the companion system of an RCS, we can choose it so that the total convexity volume V_N is minimized. In practice, it is easier to find a set B of basic variables of the companion system such that $V_B = \sum_{\{i,j\} \in B} V_i^j$ is maximized. This problem has a matroidal structure, since in fact it reduces to finding sets of linearly independent weighted columns of B. Thus, its solution is achieved by the greedy algorithm in Fig. 5, which can also be applied to RRCSs.

Notation: Let γ be the set of index pairs $\{i,j\}$ such that w_i^j is a variable in the RRCS. For all $\{i,j\} \in \gamma$ let $c(i,j)$ be the column corresponding to variable z_i^j of the companion system, and let ρ be its rank.

1. Order γ so that $\{i,j\} < \{k,l\} \Leftrightarrow V_i^j < V_k^l$ and initialize $B = 0$.
2. Let $\{i,j\} = \max(\gamma, <)$ (this picks the max. convexity volume gap pair).
3. If $c(i,j)$ is linearly independent from the vectors in $\{c(k,l) \mid \{k,l\} \in B\}$ then update $B \leftarrow B \cup \{i,j\}$.
4. Update $\gamma \leftarrow \gamma \backslash \{i,j\}$.
5. If $|B| = \rho$, terminate.
6. Go back to step 2.

Figure 5: Greedy algorithm to construct a basis B of the companion system which maximizes the convexity volume.

7 Applications of reduction constraints

In this section we show four different applications of the reduction constraints methods presented in this paper. The classes of problems we worked on range from continuous nonconvex to combinatorial optimization.

7.1 Pooling and Blending problems

Pooling and Blending problems (PBPs) involve the determination of optimal amounts of different raw materials that need to be mixed to produce required amounts of end-products with desired properties. Such problems occur frequently in the petrochemical industry and are well known to exhibit multiple local minima. There is a vast literature on these problems [FHJ92, VF93, BTEG94, VF96, ATS99, TS99, ABH+02, TS02b]. Full computational results relating to the application of RCs to this problem are reported in [LP04]. The formulation we employed in the tests is that of [ATS99] (also known as the p-formulation [TS02b]):

$$\min_{f,q,x} \sum_{j=1}^{p} \sum_{i=1}^{n_j} c_{ij} f_{ij} - \sum_{k=1}^{r} d_k \sum_{j=1}^{p} x_{jk} \tag{12}$$

$$\sum_{i=1}^{n_j} f_{ij} - \sum_{k=1}^{r} x_{jk} = 0, \qquad \forall j \le p \tag{13}$$

$$q_{jw} \sum_{k=1}^{r} x_{jk} - \sum_{i=1}^{n_j} \lambda_{ijw} f_{ij} = 0, \qquad \forall j \le p \; \forall w \le l \tag{14}$$

$$\sum_{j=1}^{p} x_{jk} \le S_k, \qquad \forall k \le r \tag{15}$$

$$\sum_{j=1}^{p} q_{jw} x_{jk} - Z_{kw} \sum_{j=1}^{p} x_{jk} \le 0, \qquad \forall k \le r \; \forall w \le l \tag{16}$$

$$f^L \le f \le f^U, q^L \le q \le q^U, x^L \le x \le x^U, \tag{17}$$

where f_{ij} is the flow of input stream i into pool j, x_{jk} is the total flow from pool j to product k and q_{jw} is the w-th quality of pool j; p is the number of pools, r the number of products, l the number of qualities, n_j the number of streams; $c_{ij}, d_k, S_k, Z_{kw}, \lambda_{ijw}$ are given parameters.

Solving several PBP instances to optimality with a spatial Branch-and-Bound (sBB) algorithm showed an improvement of the reduction constraint reformulation over the original formulation ranging up to five orders of magnitude. Various instances were solved at the first sBB iteration, meaning that the objective function value of the convex relaxation was the same as that of the original problem. Such gains were simply due to the reformulation, as no other parameter was changed in the sBB run. This exceptional performance was explained by noting that the constraint sets (14) and (16) (the bilinear constraints in the general blending problem formulation above) define more bilinear products than is really necessary. If we were to re-write the first term on the left hand side of (14) as $\sum_{k=1}^{r} q_{jw}x_{jk}$, we would not need to create all the variables t_{jw}. In other words, reduction constraints automatically "recognized" that distributing products over sums is advantageous. It is important to note that this is typically an easy "cognitive task" for a human mind, but very hard to formalize for a computer.

7.2 The Quadratic Knapsack problem

This problem occurs frequently in resource allocation and multicommodity network flow problems [FV95]. Its formulation is as follows:

$$\left.\begin{array}{rcl} \min\limits_{x} & c^{\mathsf{T}}x + x^{\mathsf{T}}Qx & \\ & a^{\mathsf{T}}x = & \gamma \\ x^L \leq & x & \leq x^U, \end{array}\right\} \tag{18}$$

where $x, x^L, x^U, a \in \mathbb{R}^n$, Q is an $n \times n$ symmetric matrix, and $\gamma \in \mathbb{R}$. Reduction constraints can be generated by multiplying the knapsack constraint $a^{\mathsf{T}}x = \gamma$ by all problem variables to obtain

$$\forall k \leq n \quad aw_i^k = \gamma x_k. \tag{19}$$

The companion system

$$\begin{pmatrix} a_1 & a_2 & a_3 & \dots & a_n & 0 & & & & \dots & & & & & & & \\ & a_1 & 0 & \dots & 0 & a_2 & a_3 & \dots & a_n & & 0 & \dots & & & & & \\ & & a_1 & 0 & \dots & 0 & a_2 & 0 & \dots & 0 & a_3 & \dots & a_n & 0 & \dots & & \\ & & & \ddots & & & & \ddots & & & & \ddots & & \ddots & & & \\ & & & & a_1 & & & & a_2 & & & & a_3 & & a_n & & \end{pmatrix} z = 0$$

is already in echelon form. Supposing $a_i \neq 0$ for all $i \leq n$, the set $z_1^1, z_1^2, \ldots, z_1^n$ is a set of basic variables of the companion system. In fact, by applying any permutation to the original variable vector x it is easy to see that each set $Z_i = \{z_i^k \mid k \leq n\}$ is a basic set for each $i \leq n$. By applying the methods in Section 6 we can therefore choose the most convenient Z_i in terms of convexity volume gap. Since the bilinear terms $x_i x_k$ corresponding to the basic variables $z_i^k \in Z_i$ get replaced by the reduction constraints, we select Z_i such that the total convexity volume $V_{Z_i} = \sum_k V_i^k$ is maximum. V_{Z_i} turns out to be a function of x_i^L, x_i^U, $i = 1, \ldots, n$ as explained in Sections 6.1, 6.2, and can be readily computed. Computational work on this class of problems is still ongoing.

7.3 The Quadratic Assignment problem

The Quadratic Assignment Problem (QAP) can be formulated as the following integer bilinear problem:

$$\left.\begin{array}{rrcl} \min\limits_x & \sum\limits_{i,j,k,l}^n a_{ij}b_{kl}x_{ik}x_{jl} & + & \sum\limits_{i,j}^n c_{ij}x_{ij} \\ \forall i \le n & \sum\limits_{j=1}^n x_{ij} & = & 1 \\ \forall j \le n & \sum\limits_{i=1}^n x_{ij} & = & 1 \\ \forall i,j \le n & x_{ij} & \in & \{0,1\}, \end{array}\right\} \tag{20}$$

where x_{ij} is an $n \times n$ array of binary variables and $A = (a_{ij}), B = (b_{ij}), C = (c_{ij})$ are given $n \times n$ matrices. This is known as the Koopmans-Beckmann formulation [KB57]. Since the problem has bilinear terms and linear equality constraints, it is amenable to the application of reduction constraints.

We now generate reduction constraints for (20) by multiplying the assignment constraints by each problem variable x_{kl}, to obtain the following RCS:

$$\begin{array}{rrcl} \forall i,k,l \le n & \sum\limits_{j=1}^n w_{ijkl} & = & x_{kl} \\ \forall j,k,l \le n & \sum\limits_{i=1}^n w_{ijkl} & = & x_{kl}, \end{array}$$

where w_{ijkl} is the linearizing variable that replaces the bilinear term $x_{ij}x_{kl}$. We show next that these reduction constraints imply $w_{ijkl} = x_{ij}x_{kl}$.

7.1 Lemma
Let $x_{ij} \in \{0,1\}$ for all $i,j \le n$ and $w_{ijkl} \in [0,1]$. If the assignment constraints $\forall i \le n \sum_{j=1}^n x_{ij} = 1$ (AC) and the derived reduction constraints $\forall i,k,l \le n \sum_{j=1}^n w_{ijkl} = x_{kl}$ (RC) hold, provided $w_{ijkl} = w_{klij}$ for all $i,j,k,l \le n$, we have $w_{ijkl} = x_{ij}x_{kl}$, and in particular, $w_{ijkl} \in \{0,1\}$.

Proof. Constraints (RC) and the fact that $w_{ijkl} = w_{klij}$ imply that $\forall i,j,k,l \le n (w_{ijkl} \le x_{kl} \wedge w_{ijkl} \le x_{ij})$ (LT), whence for any subset J of $\{1,\ldots,n\}$ we have $\sum_{f\in J} w_{ifkl} \le \sum_{f \in J} x_{if}$. Pick $j \le n$. By considering $J = \{1,\ldots,j-1,j+1,\ldots n\}$, for each $k,l \le n$ we obtain:

$$\begin{array}{rcll} \sum\limits_{f\neq j} x_{if} & \ge & \sum\limits_{f \neq j} w_{ifkl} & \Rightarrow \text{(add and subtract } w_{ijkl}) \\ \sum\limits_{f\neq j} x_{if} & \ge & \sum\limits_{f \le n} w_{ifkl} - w_{ijkl} & \Rightarrow \text{(substitute } x_{kl} \text{ by (RC))} \\ w_{ijkl} & \ge & x_{kl} - \sum\limits_{f\neq j} x_{if} & \Rightarrow \text{(add and subtract } x_{ij}) \\ w_{ijkl} & \ge & x_{ij} + x_{kl} - \sum\limits_{f \le n} x_{if} & \Rightarrow \text{(substitute 1 by (AC))} \\ w_{ijkl} & \ge & x_{ij} + x_{kl} - 1 & \text{(GT).} \end{array}$$

Thus, if $x_{ij} = 1$ and $x_{kl} = 1$, inequality (GT) implies $w_{ijkl} \ge 1$, i.e. $w_{ijkl} = 1$. If either x_{ij} or x_{kl} is zero, inequalities (LT) imply $w_{ijkl} \le 0$, i.e. $w_{ijkl} = 0$, as claimed. □

The result above was cited in [SB94] as appearing originally in [SL96], but the proof given therein is different.

Thus, reduction constraints provide an exact MILP reformulation of problem (20). Since the converse does not hold, reduction constraints actually provide a reformulation whose associated continuous relaxation is tighter than that derived from the "usual" linearization constraints (LT) and (GT). Notice also that enforcing the symmetry constraints $w_{ijkl} = w_{klij}$ is equivalent to eliminate some of the w variables, so the number of linearization variables is $\frac{1}{2}n^2(n^2+1)$ rather than n^4.

We can further tighten the formulation by noting that multiplying the equation $x_{ij} = 1$ by the problem variable x_{ij} generates the reduction constraints $w_{ijij} = x_{ij}$ for each $i, j \le n$. Although fixing all $x_{ij} = 1$ is not a valid problem constraint, the derived reduction constraints hold even when $x_{ij} = 0$, so we can include them in the formulation. What these reduction constraints express is that since x_{ij} is a binary 0-1 variable, $x_{ij}^2 = x_{ij}$.

As was the case for the PBP, this reformulation has already been proposed and studied in [FY83], and employed extensively in many other works about the QAP. Again, reduction constraints automatically generated a result which was already known, but which cost people many hours of toil to think up, verify and publish.

7.4 Multi-processor Scheduling with Communication Delays

The Multi-processor Scheduling problem with Communication Delays (MSPCD) arises in parallel computing. It consists of scheduling dependent tasks with communication delays (due to data transfer) onto homogeneous, arbitrary connected multiprocessor architecture such that the total completion time is minimum. The model [DMM03] is further complicated by the fact that communication delays between tasks also depend on what processors the tasks are being executed on; namely, we assume that the connections among the processors do not form a complete graph, so transferring data from a processor to an adjacent processor requires less time than between non-adjacent processors.

The MSPCD then can be formulated as follows:

$$\min_{y,t} \max_{j \le n} \{t_j + L_j\} \tag{21}$$

subject to:

$$\sum_{k=1}^{p} \sum_{s=1}^{n} y_{jk}^s = 1 \quad \forall j \le n \tag{22}$$

$$\sum_{j=1}^{n} y_{jk}^1 \le 1 \quad \forall k \le p \tag{23}$$

$$\sum_{j=1}^{n} y_{jk}^s \le \sum_{j=1}^{n} y_{jk}^{s-1} \quad \forall k \le p \quad \forall s \in \{2, \ldots, n\} \tag{24}$$

$$t_j \quad \ge \quad t_i + L_i + \sum_{k=1}^{p} \sum_{s=1}^{n} \sum_{l=1}^{p} \sum_{r=1}^{n} \gamma_{ij}^{kl} y_{ik}^s y_{jl}^r \quad \forall i \in \text{Pred}(j), \forall j \le n \tag{25}$$

$$t_j \quad \ge \quad t_i + L_i - \alpha \left[2 - \left(y_{ik}^s + \sum_{r=s+1}^{n} y_{jk}^r \right) \right] \quad \forall k \le p, \forall s \le n-1, \forall i, j \le n \tag{26}$$

$$y_{jk}^s \in \{0, 1\}, \quad \forall j, s \le n \quad \forall k \le p \tag{27}$$

$$t_j \ge T^L - L_j \quad \forall j \le n, \tag{28}$$

where p is the number of processors, n is the number of tasks, $\text{Pred}(j)$ is the set of tasks that have to be executed before task j, L_j is the length of task j, γ_{ij}^{kl} is the communication delay between tasks i

and j when they are executed on processors k and l, α is a sufficiently large penalty coefficient, y^s_{jk} if task j is the s-th process to be executed on processor k and t_j is the starting time of process j. T^L is a lower bound on the total completion time given by either load balancing considerations or a CPM method applied to the task precedence graph where the arcs are weighted by the running times L_j.

This MINLP formulation has $O(n^2p)$ binary variables and $O(n^4p^2)$ bilinear terms in constraints (25). A straightforward exact linearization of the bilinear terms, obtained by replacing each bilinear term $y^s_{ik}y^r_{jl}$ with a continuous variable $w^{sr}_{ijkl} \in [0,1]$ and adding the following constraints to the formulation (for all $i,j,s,r \leq n, k,l \leq p$):

$$w^{sr}_{ijkl} \leq y^s_{ik} \quad \wedge \quad w^{sr}_{ijkl} \leq y^r_{jl} \quad \wedge \quad w^{sr}_{ijkl} \geq y^s_{ik} + y^r_{jl} - 1, \tag{29}$$

yields an exact MILP reformulation with $O(n^2p)$ binary variables and $O(n^4p^2)$ continuous variables. Reduction constraints can be generated by multiplying the assignment constraints (22) by the y variables, obtaining

$$\sum_{k=1}^{p}\sum_{s=1}^{n} w^{sr}_{jikl} = y^r_{il} \quad \forall i,j,r \leq n, l \leq p.$$

As in the case of the QAP (Section 7.3), the linearizing constraints (29) can be inferred from assignment constraints (22) and reduction constraints (7.4) in a similar manner as Lemma 7.1, and can therefore be removed from the formulation, provided we add the symmetry constraints $w^{sr}_{jikl} = w^{rs}_{ijlk}$ (or we use the symmetry equations to eliminate some of the variables). Preliminary computational results using CPLEX 8.1 are very promising (up to 92% CPU time reduction when using the reduction constraints reformulation rather than the usual linearization constraints (29)). Note that in this case we are not finding improvements in the objective function value, since the objective function is "artificially" bounded below by T^L, which is in fact a very tight bound (in some instances it is a precise bound) for the optimal value. On the other hand, the set of reduction constraints derived by the assignment constraints is much smaller than the usual set of linearization constraints (29), thus improving the CPU time efficiency. Furthermore, the formulation is tighter in the sense that even in the LP relaxation many (in some cases, all) variables turn out to have integer values.

8 Conclusion

This paper introduces the concept of reduction constraints, which are a subset of RLT constraints derived by multiplying linear equality constraints by problem variables. Reduction constraints are used to generate (in a wholly automatic way) exact reformulations for bilinear problems involving a smaller number of bilinear terms; in practice, some of the bilinear terms are replaced by the (linear) reduction constraints. These reformulations are generally much tighter than the ones derived from the original problem formulation. We show that there is usually a choice of the set of bilinear terms being replaced. In order to further tighten the formulation, we propose replacing those bilinear terms causing the highest convexity volume gap between the original problem and the convex relaxation. Finally, we analyse reduction constraints reformulations of four of well-known classes of problems in the literature. In some cases, the reformulation generated by reduction constraints had already been discovered by other means, albeit not in an automatic way. This fact, aside from being a validation of this method, emphasizes its importance as an efficient automatic reformulation technique.

References

[ABH+02] C. Audet, J. Brimberg, P. Hansen, S. Le Digabel, and N. Mladenović. Pooling problem: Alternate formulations and solution methods. *Le Cahiers du GERAD*, G-2000-23, May 2000 (revised 2002).

[ABP01] J. Abello, S. Butenko, and P.M. Pardalos. Finding independent sets in a graph using continuous multivariable polynomial formulations. *Journal of Global Optimization*, 21:111–137, 2001.

[ADFN98] C.S. Adjiman, S. Dallwig, C.A. Floudas, and A. Neumaier. A global optimization method, αbb, for general twice-differentiable constrained nlps: I. theoretical advances. *Computers & Chemical Engineering*, 22(9):1137–1158, 1998.

[Adj98] C.S. Adjiman. *Global Optimization Techniques for Process Systems Engineering.* PhD thesis, Princeton University, June 1998.

[AF96] C.S. Adjiman and C.A. Floudas. Rigorous convex underestimators for general twice-differentiable problems. *Journal of Global Optimization*, 9(1):23–40, July 1996.

[AKF83] F.A. Al-Khayyal and J.E. Falk. Jointly constrained biconvex programming. *Mathematics of Operations Research*, 8(2):273–286, 1983.

[Ans03] K.M. Anstreicher. Recent advances in the solution of quadratic assignment problems. *Mathematical Programming B*, 97:27–42, 2003.

[ATS99] N. Adhya, M. Tawarmalani, and N.V. Sahinidis. A lagrangian approach to the pooling problem. *Industrial and Engineering Chemistry Research*, 38:1956–1972, 1999.

[BBPP98] I. Bomze, M. Budinich, P.M. Pardalos, and M. Pelillo. The maximum clique problem. In D.-Z. Du and P.M. Pardalos, editors, *Handbook of Combinatorial Optimization, Supp. A*, volume supp. A, pages 1–74, Dordrecht, 1998. Kluwer Academic Publishers.

[BLW03] K.-M. Björk, P.O. Lindberg, and T. Westerlund. Some convexifications in global optimization of problems containing signomial terms. *Computers & Chemical Engineering*, 27:669–679, 2003.

[BTEG94] A. Ben-Tal, G. Eiger, and V. Gershovitz. Global minimization by reducing the duality gap. *Mathematical Programming*, 63:193–212, 1994.

[DMM03] T. Davidovič, N. Maculan, and N. Mladenovič. A mathematical programming formulation for the multiprocessor scheduling problem with communication delays. In T. Davidovič and N. Mladenovič, editors, *SYM-OP-IS Conference Proceedings, Herceg-Novi*, pages 331–334, Beograd, September 2003. Mathematical Institute, Academy of Sciences.

[FHJ92] L.R. Foulds, D. Haughland, and K. Jornsten. A bilinear approach to the pooling problem. *Optimization*, 24:165–180, 1992.

[Flo00] C.A. Floudas. *Deterministic Global Optimization.* Kluwer Academic Publishers, Dordrecht, 2000.

[FV95] C. Floudas and V. Visweswaran. Quadratic optimization. In R. Horst and P.M. Pardalos, editors, *Handbook of Global Optimization*, volume 1, Dordrecht, 1995. Kluwer Academic Publishers.

[FY83] A.M. Frieze and J. Yadegar. On the quadratic assignment problem. *Discrete Applied Mathematics*, 5:89–98, 1983.

[GJS02] J. Garloff, C. Jansson, and A.P. Smith. Construction of lower bound functions for polynomials. *Technical Report n. 0201, Fachhochschule Konstanz*, 2002.

[GK97] I. Grossmann and Z. Kravanja. Mixed-integer nonlinear programming: a survey of algorithms and applications. In L.T. Biegler, T.F. Coleman, A.R. Conn, and F.N. Santosa, editors, *Large-scale Optimization with Applications*, pages 73–100. Springer-Verlag, 1997.

[KB57] T.C. Koopmans and M.J. Beckmann. Assignment problems and the location of economic activities. *Econometrica*, 25:53–76, 1957.

[Lib03] L. Liberti. Linearity embedded in nonconvex programs. *Journal of Global Optimization (accepted for publication)*, 2003.

[Lib04a] L. Liberti. Reduction constraints for the global optimization of nlps. *International Transactions in Operations Research*, 11(1):34–41, 2004.

[Lib04b] L. Liberti. *Reformulation and Convex Relaxation Techniques for Global Optimization*. PhD thesis, Imperial College London, UK, March 2004.

[LP03] L. Liberti and C.C. Pantelides. Convex envelopes of monomials of odd degree. *Journal of Global Optimization*, 25:157–168, 2003.

[LP04] L. Liberti and C.C. Pantelides. An exact reformulation algorithm for large nonconvex nlps involving bilinear terms. *Journal of Global Optimization (submitted)*, 2004.

[McC76] G.P. McCormick. Computability of global solutions to factorable nonconvex programs: Part i — convex underestimating problems. *Mathematical Programming*, 10:146–175, 1976.

[MF95] C. D. Maranas and C. A. Floudas. Finding all solutions to nonlinearly constrained systems of equations. *Journal of Global Optimization*, 7(2):143–182, 1995.

[PR02] P.M. Pardalos and H. Edwin Romeijn, editors. *Handbook of Global Optimization*, volume 2. Kluwer Academic Publishers, Dordrecht, 2002.

[RS03] H. Ryoo and N. Sahinidis. Global optimization of multiplicative programs. *Journal of Global Optimization*, 26(4):387–418, 2003.

[SA86] H.D. Sherali and W.P. Adams. A tight linearization and an algorithm for 0-1 quadratic programming problems. *Management Science*, 32(10):1274–1290, 1986.

[SA92] H.D. Sherali and A. Alameddine. A new reformulation-linearization technique for bilinear programming problems. *Journal of Global Optimization*, 2:379–410, 1992.

[SA99] H.D. Sherali and W.P. Adams. *A Reformulation-Linearization Technique for Solving Discrete and Continuous Nonconvex Problems*. Kluwer Academic Publishers, Dodrecht, 1999.

[SB94] H. Sherali and E. Brown. A quadratic partial assignment and packing model and algorithm for the airline gate assignment problem. In P. Pardalos and H. Wolkowicz, editors, *Quadratic Assignment and related problems*, pages 343–364. American Mathematical Society, 1994.

[She98] H.D. Sherali. Global optimization of nonconvex polynomial programming problems having rational exponents. *Journal of Global Optimization*, 12:267–283, 1998.

[She02] H.D. Sherali. Tight relaxations for nonconvex optimization problems using the reformulation-linearization/convexification technique (rlt). In Pardalos and Romeijn [PR02], pages 1–63.

[SL96] H.D. Sherali and Y. Lee. Tighter representations for set partitioning problems. *Discrete Applied Mathematics*, 68:153–167, 1996.

[Smi96] E.M.B. Smith. *On the Optimal Design of Continuous Processes*. PhD thesis, Imperial College of Science, Technology and Medicine, University of London, October 1996.

[SP99] E.M.B. Smith and C.C. Pantelides. A symbolic reformulation/spatial branch-and-bound algorithm for the global optimisation of nonconvex minlps. *Computers & Chemical Engineering*, 23:457–478, 1999.

[ST97] H. Sherali and C.H. Tuncbilek. New reformulation linearization/convexification relaxations for univariate and multivariate polynomial programming problems. *Operations Research Letters*, 21:1–9, 1997.

[SW01] H.D. Sherali and H. Wang. Global optimization of nonconvex factorable programming problems. *Mathematical Programming*, A89:459–478, 2001.

[TS99] M. Tawarmalani and N.V. Sahinidis. Global optimization of mixed integer nonlinear programs: A theoretical and computational study. *Mathematical Programming (to appear)*, `http://archimedes.scs.uiuc.edu/papers/comp.pdf`, 1999.

[TS01] M. Tawarmalani and N.V. Sahinidis. Semidefinite relaxations of fractional programming via novel techniques for constructing convex envelopes of nonlinear functions. *Journal of Global Optimization*, 20(2):137–158, 2001.

[TS02a] M. Tawarmalani and N. Sahinidis. Convex extensions and envelopes of semi-continuous functions. *Mathematical Programming*, 93(2):247–263, 2002.

[TS02b] M. Tawarmalani and N.V. Sahinidis. Convexification and global optimization of the pooling problem. *Mathematical Programming (submitted)*, 2002.

[TS02c] M. Tawarmalani and N.V. Sahinidis. Exact algorithms for global optimization of mixed-integer nonlinear programs. In Pardalos and Romeijn [PR02], pages 1–63.

[VF93] V. Visweswaran and C. A. Floudas. New properties and computational improvement of the gop algorithm for problems with quadratic objective functions and constraints. *Journal of Global Optimization*, 3:439–462, 1993.

[VF96] V. Visweswaran and C. A. Floudas. New formulations and branching strategies for the gop algorithm. In I.E. Grossmann, editor, *Global Optimization in Engineering Design*, Dordrecht, 1996. Kluwer Academic Publishers.

[ZG99] 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*, 14:217:249, 1999.