Global Optimization of Mixed-Integer Nonlinear Problems

C. S. Adjiman[1], I. P. Androulakis[2] and C. A. Floudas[3]

Department of Chemical Engineering,

Princeton University,

Princeton, NJ 08544

Abstract

Two novel deterministic global optimization algorithms for nonconvex mixed-integer problems (MINLPs) are proposed, using the advances of the αBB algorithm for nonconvex NLPs Adjiman *et al.* (1998a). The Special Structure Mixed-Integer αBB algorithm (SMIN-αBB addresses problems with nonconvexities in the continuous variables and linear and mixed-bilinear participation of the binary variables. The General Structure Mixed-Integer αBB algorithm (GMIN-αBB), is applicable to a very general class of problems for which the continuous relaxation is twice continuously differentiable. Both algorithms are developed using the concepts of branch-and-bound, but they differ in their approach to each of the required steps. The SMIN-αBB algorithm is based on the convex underestimation of the continuous functions while the GMIN-αBB algorithm is centered around the convex relaxation of the entire problem. Both algorithms rely on optimization or interval based variable bound updates to enhance efficiency. A series of medium-size engineering applications demonstrates the performance of the algorithms. Finally, a comparison of the two algorithms on the same problems highlights the value of algorithms which can handle binary or integer variables without reformulation.

1 Introduction

The decision-making processes that take place during the design of new products or chemical plants can be made more rational and efficient thanks to the use of mathematical models within a global optimization framework. For instance, approaches based on nonlinear programming (NLP), such as those described in Horst and Tuy (1996), have been used to determine optimum equipment sizes and operating conditions for a given process (Floudas, 1995; Grossmann, 1996). The economic benefits to be derived from identifying the *global* solution

[1]Current address: Centre for Process Systems Engineering, Imperial College of Science, Technology and Medicine, Prince Consort Road, London SW7 2BY, United Kingdom

[2]Current address: Corporate Strategic Research, ExxonMobil Research and Engineering Co., Annandale, NJ 08801-0998

[3]Author to whom all correspondence should be addressed

of the many nonconvex problems that arise in chemical engineering has been amply illustrated (Grossmann, 1996). The most significant contribution of mathematical approaches, however, comes from their ability to incorporate many alternative structures within a single problem. This is achieved through the introduction of integer variables which leads to the formulation of a mixed-integer nonlinear problem (MINLP) (Floudas, 1995; Floudas and Grossmann, 1995; Grossmann, 1996). Such an approach has already been used for a wide array of applications including process synthesis.

The solution of many MINLPs relevant to chemical engineering is made challenging not only by the presence of integer variables but also by the nonconvexities in the models. As a result, the potential contributions of mixed-integer nonlinear optimization to general design problems have not yet been fully realized. The deterministic approaches proposed to date for the solution of such problems can be expressed within the branch-and-bound framework, a general approach which has had a significant impact on the chemical engineering community since its introduction in the literature by Lee *et al.* (1970) and Westerberg and Stephanopoulos (1975). While such techniques have long been used to solve convex mixed-integer problems (see, for instance, Beale, 1977; Ostrovsky *et al.*, 1990), and certain types of nonconvex NLPs (Falk and Soland, 1969; McCormick, 1976), they have only recently started to be applied to the global optimization of nonconvex MINLPs. This introduction provides an overview of work in this specific area, and the reader is referred to Horst and Tuy (1996) for a general description of global optimization techniques.

The branch-and-reduce algorithm of Ryoo and Sahinidis (1995), relies on existing underestimation techniques such as those proposed by McCormick (1976) and focuses on the reduction of the size of the solution domain through the addition of feasibility and optimality tests. The interval analysis algorithm of Vaidyanathan and El-Halwagi (1996) can be presented as a branch-and-bound approach in which the function values are bounded using interval arithmetic, the domain size is reduced through partitioning, and fathoming is performed by applying upper bound, infeasibility, monotonicity, nonconvexity and lower bound tests, as well as the distrust-region method. The reformulation/spatial branch-and-bound algorithm of Smith and Pantelides (1997, 1999) is designed to address functions that involve binary arithmetic operators and concave or convex operators such as logarithms and exponentials. Finally, the Extended Cutting Plane algorithm of Westerlund *et al.* (1998), an alternative to branch-and-bound techniques, tackles problems involving pseudo-convex functions and is an extension of the ECP algorithm for convex MINLPs (Westerlund and Pettersson, 1995). A review of these algorithms is presented in Adjiman *et al.* (1999). More specialized algorithms have also been proposed for certain classes of applications, as in the work of Zamora and Grossmann (1998a) on heat exchanger networks.

Two new broadly applicable global optimization approaches are presented in this paper. They were briefly introduced in Adjiman *et al.* (1997), and this paper provides a complete description of the theoretical basis of the algorithms, and computational experiments which enable the determination of the most adequate implementation decisions. First, the Special Structure Mixed-Integer αBB algorithm (SMIN-αBB), designed for problems with general nonconvexities in the continuous variables and restricted participation of the binary variables, is discussed. It is based on the αBB global optimization algorithm for twice continuously differentiable NLPs (Androulakis *et al.*, 1995; Adjiman and Floudas, 1996; Adjiman *et al.*, 1998a). The performance of the algorithm is studied on a number of small literature problems and some medium-size heat exchanger network problems. Particular attention is paid to the selection of options for variable branching and bound updates that allow the fast convergence of the algorithm. In order to address the broader class of problems whose continuous relaxation is twice continuously differentiable, the General Structure Mixed-Integer αBB algorithm (GMIN-αBB) is then introduced. While based on a classical branch-and-bound approach for mixed-integer problems (Gupta and Ravindran, 1985), it circumvents the problems such algorithms encounter when dealing with nonconvex problems through efficient and rigorous strategies for bounding, branching and variable bound tightening. The algorithm is tested on the series of small literature problems, a pump network synthesis problem and four trim loss minimization problems. Finally, the applicability of the two algorithms is discussed in light of their differences.

2 Foundations of the SMIN-αBB algorithm

The SMIN-αBB algorithm is guaranteed to converge in a finite number of iterations to the global solution of MINLPs involving functions that can be separated into a twice continuously differentiable part, a mixed bilinear part and linear binary part. Thus, the assumption of convexity required for commonly used MINLP algorithms such as the OA/ER (Duran and Grossmann, 1986; Kocis and Grossmann, 1987) and the GBD (Geoffrion, 1972) is lifted.

Mathematically, the class of MINLPs for which the SMIN-αBB algorithm is designed is given by:

$$
\begin{array}{lc}
\min\limits_{\boldsymbol{x},\boldsymbol{y}} & f(\boldsymbol{x}) + \boldsymbol{x}^T A_f \boldsymbol{y} + c_f^T \boldsymbol{y} \\
s.t. & g_i(\boldsymbol{x}) + \boldsymbol{x}^T A_{g,i} \boldsymbol{y} + c_{g,i}^T \boldsymbol{y} \leq \ 0, \ i = 1, \ldots, m \\
& h_i(\boldsymbol{x}) + \boldsymbol{x}^T A_{h,i} \boldsymbol{y} + c_{h,i}^T \boldsymbol{y} = \ 0, \ i = 1, \ldots, p \\
& \boldsymbol{x} \in [\boldsymbol{x}^L, \boldsymbol{x}^U], \quad \boldsymbol{y} \in \{0,1\}^q
\end{array}
\tag{1}
$$

where $f(\boldsymbol{x})$, $g_i(\boldsymbol{x})$, $i = 1, \ldots, m$, and $h_i(\boldsymbol{x}), i = 1, \ldots, p$ are twice continuously differentiable functions, m is the number of inequality constraints, p is the number of equality constraints, q is the dimension of the binary variable vector, A_f, $A_{g,i}$ and $A_{h,i}$ are matrices of size $n \times q$, and c_f, $c_{g,i}$ and $c_{h,i}$ are vectors of size q.

This form is highly versatile. In process synthesis, for instance, the binary variables represent the existence of certain units and participate linearly in "big-M" constraints, or in mixed-bilinear terms. If the model used involves different regimes which are described through discontinuous functions, the discontinuities can be removed by introducing binary variables which activate the appropriate model equations depending on the region of space being explored (Floudas, 1995).

2.1 Generation of valid upper and lower bounds

The global optimality of the solution found through a branch-and-bound algorithm can be guaranteed only if the bounding step generates valid upper and lower bounds on the mixed-integer nonconvex problem.

2.1.1 Upper bound

A rigorous upper bound is obtained by solving the nonconvex MINLP (1) locally. The generalized Benders decomposition (GBD) (Benders, 1962; Geoffrion, 1972; Floudas *et al.*, 1989) may be used to obtain such a solution. When there are no mixed-bilinear terms, the outer-approximation with equality relaxation (OA/ER) (Duran and Grossmann, 1986; Kocis and Grossmann, 1987) or a local MINLP branch-and-bound algorithm (B&B) (Beale, 1977; Gupta and Ravindran, 1985; Ostrovsky *et al.*, 1990; Borchers and Mitchell, 1991; Quesada and Grossmann, 1992) may also be used.

2.1.2 Lower bound

In order to obtain a valid lower bound, a relaxed problem which can be solved to global optimality must be constructed from problem (1). The subclass of (1) in which the continuous functions $f(\boldsymbol{x})$, $g_i(\boldsymbol{x})$ and $h_i(\boldsymbol{x})$ are convex can be solved to global optimality using the GBD algorithm, and, when there are no mixed-bilinear terms, the OA/ER or B&B algorithms. A problem that meets these conditions and whose solution is a lower bound on problem (1) is derived by constructing convex underestimators for the nonconvex functions $f(\boldsymbol{x})$, $g_i(\boldsymbol{x})$ and $h_i(\boldsymbol{x})$. The rigorous convexification/relaxation strategy used in the αBB algorithm for nonconvex continuous problems (Adjiman and Floudas, 1996; Adjiman *et al.*, 1998a) allows the construction of the desired lower bounding MINLP. This scheme is based on a decomposition of the functions into a sum of terms with special mathematical structure, such as linear, convex, bilinear, trilinear, fractional, fractional trilinear, univariate concave and general nonconvex terms. A different convex relaxation technique is then applied for each class of term. The linear and convex terms are unchanged. The bilinear, trilinear, fractional, fractional trilinear are replaced by a new variable on which a set of convex constraints is imposed. These and any other terms that require the introduction of a new variable in the convex lower bounding problem are referred to as *substituted terms*. The univariate concave terms are linearized. Finally, a convex underestimator for a nonconvex term $F_{NC}(\boldsymbol{x})$ is given by

$$F_{NC}(\boldsymbol{x}) - \sum_{i=1}^{n} \alpha_i (x_i - x_i^L)(x_i^U - x_i), \tag{2}$$

where the α_i's are positive scalars such that $H_{F_{NC}}(\boldsymbol{x}) + 2\mathrm{diag}(\alpha_i)$ is positive semi-definite for all $\boldsymbol{x} \in [\boldsymbol{x}^L, \boldsymbol{x}^U]$, where $H_{F_{NC}}(\boldsymbol{x})$ is the Hessian matrix of the general nonconvex term. Appropriate α_i's can then be derived using one of the techniques presented in Adjiman and Floudas (1996) and Adjiman *et al.* (1998a). The reader is also referred to these publications for a description of the treatment of equality constraints. Thus, for MINLPs of type (1), the nonconvexities arising from the continuous terms are handled in the same manner as in the αBB algorithm, but the generation of a valid lower bound requires the solution of a convex MINLP rather than a convex NLP. In addition, the decomposition approach of the GBD is needed to tackle nonconvexities arising from mixed bilinear terms.

A general formulation for the lower bounding problem is given by

$$\begin{aligned}
\min_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{w}} \quad & \breve{f}(\boldsymbol{x},\boldsymbol{w}) + \boldsymbol{x}^T A_f \boldsymbol{y} + c_f^T \boldsymbol{y} \\
& \breve{g}_i(\boldsymbol{x},\boldsymbol{w}) + \boldsymbol{x}^T A_{g,i} \boldsymbol{y} + c_{g,i}^T \boldsymbol{y} \leq \ 0, \ i = 1, \dots, m \\
& \breve{h}_i(\boldsymbol{x},\boldsymbol{w}) + \boldsymbol{x}^T A_{h,i} \boldsymbol{y} + c_{h,i}^T \boldsymbol{y} = \ 0, \ i \in P_L \\
& \breve{h}_i(\boldsymbol{x},\boldsymbol{w}) + \boldsymbol{x}^T A_{h,i} \boldsymbol{y} + c_{h,i}^T \boldsymbol{y} = \ 0, \ i \in P_N \\
& \underset{\smile}{h}_i^-(\boldsymbol{x},\boldsymbol{w}) - \boldsymbol{x}^T A_{h,i} \boldsymbol{y} - c_{h,i}^T \boldsymbol{y} = \ 0, \ i \in P_N
\end{aligned} \tag{3}$$

$$\mathcal{C}(\boldsymbol{x}, \boldsymbol{w})$$
$$\boldsymbol{x} \in [\boldsymbol{x}^L, \boldsymbol{x}^U], \quad \boldsymbol{y} \in \{\boldsymbol{y}^L, \boldsymbol{y}^U\}, \quad \boldsymbol{w} \in [\boldsymbol{w}^L, \boldsymbol{w}^U]$$

where the $\breve{}$ superscript denotes the convex underestimator of the specified function valid for the current domain $[\boldsymbol{x}^L, \boldsymbol{x}^U]$; $\boldsymbol{w}$ denotes the set of variables that replace the substituted terms; and $\mathcal{C}(\boldsymbol{x}, \boldsymbol{w})$ denotes the set of additional constraints that arise from the underestimation of these terms. Finally P_L denotes the set of equality constraints which involve only linear and substituted terms, and P_N denotes all the other equalities.

An important feature of the lower bounding scheme is that the quality of the lower bounds improves with increasing variable ranges. This ensures ϵ-convergence of the algorithm and can be used to improve the convergence rate as discussed in subsequent sections.

2.2 Selection of branching variable

A list of lower bounds for the regions that have not yet been partitioned is maintained. The region with the lowest lower bound is selected for further exploration at each iteration.

The choice continuous or binary variable to be branched on at the chosen node can affect the performance of the algorithm significantly. If a continuous variable is judiciously chosen, the partition results in an improvement of the lower bound on the problem through a tightening of the convex underestimators for the nonconvex continuous functions. Binary variables also have an indirect effect on the quality of the convex underestimators as they influence the range of values that the continuous variables can take on. In addition, fixing a binary variable through branching reduces the complexity of the convex MINLP that must be solved to generate a lower bound. In this section, a new result is presented for continuous variable branching, and alternative strategies are introduced to exploit the structure of the problem.

2.2.1 Continuous variable branching

This first branching variable selection scheme exploits the direct relationship between the range of the continuous variables and the quality of the lower bounds by partitioning *only on these variables*. One of the rules available for the αBB algorithm can be used for the selection, such as the least-reduced axis, the term measure or the variable measure. The details of these strategies can be found in Adjiman *et al.* (1998b), where computational experience showed the term and variable measures to result in improved performance for different types of problems.

Given the proven effectiveness of the term measure as a branching criterion, we present a new expression to rigorously compute the maximum separation distance between a fractional term x_1/x_2 and its convex envelope. This distance depends on the signs of the variable bounds.

Property 2.1 *For $x_2^L > 0$, , the maximum separation distance is given by*

$$d_{max} = \begin{cases} \frac{x_1^L + x_1^U}{4}\left(\frac{1}{x_2^U} + \frac{1}{x_2^L}\right) & \textit{for } x_1^L \geq 0, \\ \frac{x_1^L - x_1^U}{x_2^L - x_2^U} + \frac{x_1^L\, {x_2^U}^2 - x_1^U\, {x_2^L}^2}{x_2^L\, x_2^U\left(x_2^L - x_2^U\right)} + \frac{2\left(x_1^L - 2\,x_1^U\right)\sqrt{x_2^L}}{\left(x_2^L - x_2^U\right)\sqrt{x_2^U}} + \frac{2\,x_1^L}{\sqrt{x_2^L\, x_2^U}} & \textit{for } x_1^U < 0, \\ \max\{d_1, d_2\} & \textit{for } x_1^L < 0,\ x_1^U \geq 0. \end{cases} \tag{4}$$

where

$$d_1 = \begin{cases} -\frac{x_1^L}{x_2^L} & \textit{if } x_1^L \leq \frac{x_1^L x_2^L}{x_2^L - x_2^U}, \\ 0 & \textit{otherwise.} \end{cases}$$

and, given $A = 54 x_1^L x_1^U x_2^L x_2^U \left({x_1^L}^2 + \sqrt{{x_1^L}^4 + x_1^L x_1^U x_2^L x_2^U}\right)$,

$$d_2 = \begin{cases} \frac{54^{\frac{2}{3}} {x_1^L}^2 x_1^U x_2^L x_2^U A^{\frac{2}{3}}}{\left(x_2^U - x_2^L\right)\left(A^{\frac{2}{3}} - 54^{\frac{2}{3}} x_1^L x_1^U x_2^L x_2^U\right)^2} - \frac{1458^{\frac{1}{3}} x_1^L x_1^U x_2^U}{\left(x_2^U - x_2^L\right)\left(A^{\frac{2}{3}} - 54^{\frac{2}{3}} x_1^L x_1^U x_2^L x_2^U\right)} \\ \quad + \frac{A^{\frac{2}{3}} - 54^{\frac{2}{3}} x_1^L x_1^U x_2^L x_2^U}{54^{\frac{1}{3}}\left(x_2^U - x_2^L\right) x_2^L A^{\frac{1}{3}}} + \frac{2 x_1^U x_2^U - x_1^U x_2^L + x_1^L x_2^U - x_1^L x_2^L}{\left(x_2^U - x_2^L\right) x_2^L}, \\ \quad \textit{if } {x_1^L}^4 + x_1^L x_1^U x_2^L x_2^U \geq 0,\ \textit{and } \frac{A^{\frac{2}{3}} - 54^{\frac{2}{3}} x_1^L x_1^U x_2^L x_2^U}{54^{\frac{1}{3}} x_1^L A^{\frac{1}{3}}} \in [x_2^L, x_2^U], \\ 0,\ \textit{otherwise} \end{cases}$$

Proof The proof of this property is presented in Appendix A.

2.2.2 Sequential branching: Binary-continuous

This second approach aims to tackle the combinatorial aspects of the problem and its nonconvexity sequentially. In the first levels of the branch-and-bound tree, branching takes place on the binary variables only. This policy is followed until all binary variables have been fixed. Thus, binary variable branching can go on for at most q levels of the branch-and-bound tree, where q is the number of binary variables. In instances where variable bound updates allow to discard some of the values of the binary variables, fewer levels are needed to fix all binary variables. The nonconvexities are dealt with on subsequent levels of the tree, by branching on the continuous variables, following the branching rules described in Section 2.2.1.

The choice of a specific binary variable for branching is based on random selection or/and on a priority ranking of the variables. The highest ranking binary variables are those that influence the bounds on the greatest number of variables. In general, variables that play a similar role in determining the structure of the problem are assigned the same priority and the branching variable is chosen randomly from the highest ranking group.

An important aspect of sequential branching from the computational standpoint is that, once all the binary variables have been fixed, the problems that must be solved to obtain upper and lower bounds on the solution are *continuous* nonconvex and convex problems. The bounding of the nodes during the continuous variable branching phase is therefore less computationally intensive than during the binary branching phase.

2.2.3 Hybrid branching

This third approach also involves branching on the continuous and binary variables although the type of variable selected is no longer determined from the level in the tree. To increase the impact of binary variable branching on the quality of the lower bound, such a variable is selected when a continuous relaxation of the problem indicates that the two newly created children nodes may have significantly different lower bounds. Thus, if one of the binary variables is close to 0 or 1 at a local solution of the continuous relaxation, it is branched on. The degree of closeness is an arbitrary parameter which is typically set to 0.1 or 0.2. If no "almost-integer" binary variable is found, a continuous variable is selected for branching using one of the rules described in Section 2.2.1.

2.3 Variable bound updates

The tightening of variable bounds can have a significant impact on the quality of the underestimators and lead to a reduction in iteration number. Hamed and McCormick (1993) highlight the reliance of many global optimization algorithms on variable bounds and propose an approach to determine suitable bounds from constraints. The positive effects of reductions in the range of continuous variables has been amply demonstrated in the literature on the global optimization of NLPs. Hansen *et al.* (1991) combine lower bounding functions to recursively reduce variable ranges. Visweswaran and Floudas (1993) exploit the structure of the problem to update variable bounds and use an optimization approach at each iteration to further improve the performance of the GOP algorithm (Visweswaran and Floudas, 1996a,b). Ryoo and Sahinidis (1995, 1996) and Shectman and Sahinidis (1998) use linear parametric analysis, the Lagrange multipliers of active inequality and bound constraints, and the Lagrange multipliers obtained after the solution of modified convex relaxations in the case of inactive constraints. Zamora and Grossmann (1998b) have also proposed "contraction steps" based on the Lagrange multiplier, together with a technique to select variables for bound tightening. Smith and Pantelides (1996) resort to a multiple-pass tightening procedure based on their reformulation strategy and feasibility tests, and also resort to an optimization-based strategy (Smith and Pantelides, 1999). Maranas and Floudas (1997), Zamora and Grossmann (1998a) and Adjiman *et al.* (1998b) rely on the solution of convex relaxations of the original problem with a modified objective function.

The very effective optimization-based variable bound updates come at a computational cost, so that they can sometimes result in a decrease in the number of iterations to convergence but an increase in CPU time (Adjiman *et al.*, 1998b). Since this cost is expected to rise in the context of nonconvex MINLPs, the determination of the optimal strategy becomes more critical: should they be performed at every iteration or with a lesser frequency? Should all variables be treated? Although the best approach is likely to be problem dependent, only a subset of the variable bounds should in general be tightened at any given node. For the continuous variables, this subset consists of a few variables with a high variable measure as defined in Section 2.2.1. In addition, we advocate bound updates for the binary variables because they are beneficial in two ways: they indirectly lead to the construction of tighter underestimators and they allow a binary variable to be fixed and therefore decrease the number of combinations that potentially need to be explored. The branching priorities defined in Section 2.2.2 may also be used as bound update priorities.

Two specific methods to tighten the variable bounds have been implemented in the SMIN-αBB algorithm for the continuous and binary variables. The first is an optimization-based approach in which a convex MINLP is formulated and solved for each bound to be updated. The second is an iterative approach based on interval arithmetic. This latter method can be modified to enable the tightening of binary variable bounds.

2.3.1 Optimization-based approach

In the SMIN-αBB algorithm, a new lower (upper) bound on variable x_i can be obtained by solving a convex MINLP where the objective is to minimize x_i $(-x_i)$ subject to the convexified constraints from the original problem. In addition, a so-called "objective function cut" can be added to the problem (Zamora and Grossmann, 1998a) as a constraint:

$$\breve{f}(\boldsymbol{x}, \boldsymbol{w}) + \boldsymbol{x}^T A_f \boldsymbol{y} + c_f^T \boldsymbol{y} \leq \overline{f}^*, \tag{5}$$

where the $\breve{}$ superscript denotes the convex underestimator of the specified function valid for the current domain $[\boldsymbol{x}^L, \boldsymbol{x}^U]$, $\overline{f}^*$ denotes the best upper on the global optimum and $\boldsymbol{w}$ denotes the variables that replace the substituted terms. Although this objective function cut is optional, it may help further reduce the variable ranges.

In order to get even tighter bounds, the latest improved bound replaces the old bound in the bound constraint $\boldsymbol{x} \in [\boldsymbol{x}^L, \boldsymbol{x}^U]$ before the next bound update is performed. To fully reflect the reduction in range, the underestimators which depend on the improved bound should also be updated. However, this last step can sometimes double the computational cost of each variable bound update (Adjiman *et al.*, 1998b) and should be applied with caution.

The SMIN-αBB algorithm is also set-up to carry out optimization-based bound updates on the binary variables by minimizing y_i or $-y_i$. The effect of such a procedure on algorithmic performance has not been previously studied and will therefore be examined in the computational section.

2.3.2 Interval-based approach

Interval arithmetic (Moore, 1979; Neumaier, 1990) can be used to update the binary and continuous variable bounds. This approach is based on the interval evaluation of the constraints in the nonconvex MINLP as well as Eq. (5), the objective function cut. The binary variables that have not yet been fixed are treated as continuous variables. The interval evaluation gives enclosures for the values of all constraints in the relaxed space $(\boldsymbol{x}, \boldsymbol{y}) \in [\boldsymbol{x}^L, \boldsymbol{x}^U] \times [\boldsymbol{y}^L, \boldsymbol{y}^U]$. The range $[F^L, F^U]$ is an enclosure for a function $F(\boldsymbol{x}, \boldsymbol{y})$ if $F(\boldsymbol{x}, \boldsymbol{y}) \in [F^L, F^U]$ for all $(\boldsymbol{x}, \boldsymbol{y}) \in [\boldsymbol{x}^L, \boldsymbol{x}^U] \times [\boldsymbol{y}^L, \boldsymbol{y}^U]$. Given $[F^L, F^U]$, the feasibility of the constraint can be assessed. A necessary feasibility condition for the inequality constraint $F(\boldsymbol{x}, \boldsymbol{y}) \leq 0$ is that $F^L \leq 0$. Similarly, a necessary feasibility condition for the equality constraint $F(\boldsymbol{x}, \boldsymbol{y}) = 0$ is $0 \in [F^L, F^U]$. If the interval evaluation of a single constraint does not satisfy the appropriate necessary condition, the current region is infeasible. A bisection procedure is used to improve the variable bounds: for a lower (upper) bound, the aim is to identify an infeasible region on the lower (upper) end of the variable range. Initially, half the variable range is considered. The interval feasibility of the left (right) half is tested. If this region is infeasible, it is discarded and the right (left) half is then tested. Otherwise, the region is further split in two and the leftmost (rightmost) range is once again tested. This procedure is repeated until the original range has been determined to be entirely infeasible or until the range being tested is smaller than a user-specified tolerance. This approach is illustrated in Figure 1 for the lower bound of a variable originally in the range $[0, 4]$, with a tolerance of 1. A general description of the iterative approach used to update the lower (upper) bound on a continuous variable x_i is given in Table 1 in pseudo-code format.

For a binary variable, the interval-based procedure for continuous variables can be easily modified to account for the nature of the variable. In this case, no iterations are needed: to update both bounds on binary variable y_i, the interval feasibility of $y_i = 0$ and $y_i = 1$ is tested. If both tests reveal infeasibility, the entire region is discarded; if one test only is feasible, y_i is fixed to the bound corresponding to the feasible test; and if both tests are feasible, the bounds on y_i remain unchanged.

The interval-based bound updates are less computationally intensive than the optimization-based strategy, but use generally looser relaxations and are therefore less effective at reducing variable ranges. Thus, at early levels of the branch-and-bound tree, when variable bound updates are most critical, the overestimation of the interval computations is most significant. As a result, the optimization-based approach leads to tighter bounds.

2.4 Algorithmic procedure

2.4.1 Flowchart

In the SMIN-αBB algorithm, the upper and lower bounding procedures, and the branching and variable bound update strategies are embedded within a branch-and-bound algorithm. The flowchart shown in Figure 2 highlights the main features of the algorithm.

2.4.2 Illustrative example

In order to illustrate the algorithmic procedure, a small example proposed by Kocis and Grossmann (1989) is used. It is a simple design problem where one of two reactors must be chosen to produce a given product at the lowest possible cost. It involves two binary variables, one for each reactor, and seven continuous variables. In the following formulation, the constraint $z_1 + z_2 = 10$ has been used to tighten the bounds on z_1 and z_2:

$$
\begin{aligned}
\min \quad & 7.5y_1 + 5.5y_2 + 7v_1 + 6v_2 + 5x \\
s.t. \quad & z_1 - 0.9\left(1 - e^{-0.5v_1}\right)x_1 = 0 \\
& z_2 - 0.8\left(1 - e^{-0.5v_2}\right)x_2 = 0 \\
& x_1 + x_2 - x = 0 \\
& z_1 + z_2 = 10 \\
& v_1 - 10y_1 \leq 0 \\
& v_2 - 10y_2 \leq 0 \\
& x_1 - 12y_1 \leq 0 \\
& x_2 - 12y_2 \leq 0 \\
& y_1 + y_2 = 1
\end{aligned}
$$

$$0 \leq x_1, x_2 \leq 20; 0 \leq z_1, z_2 \leq 10$$
$$0 \leq v_1, v_2 \leq 10; 0 \leq x \leq 20$$
$$(y_1, y_2) \in \{0, 1\}^2$$

Because of the linear participation of the binary variables, the SMIN-αBB algorithm is well-suited to solve this nonconvex MINLP. It identifies the global solution of 99.2 within a relative tolerance of 10^{-3} after eight iterations and under 2 CPU seconds on an HP-C160, when bound updates are performed at every iteration and branching takes place on the binary variables first. The selection of a branching variable selection is random for the binary variables and based on the term measures for the continuous variables. At the global solution, the binary variable values are $y_1 = 1$ and $y_2 = 0$. The steps of the algorithm are shown in the branch-and-bound tree of Figure 3. Smith and Pantelides (1999) solve this problem to a relative tolerance of 10^{-2} in 19 nodes and under 2 CPU seconds on a SUN SPARCstation 5.

At the first node, the initial lower bound is 11.4 and an upper bound of 99.2 is found. The binary variable y_1 is selected as a branching variable. The region $y_1 = 0$ is infeasible and can therefore be fathomed (black node), while an improved lower bound is found for $y_1 = 1$. This latter region is therefore chosen for exploration at the second iteration. Variable bound updates reveal that $y_2 = 1$ is infeasible so that y_2 can be fixed to zero. Branching on the continuous variables may now begin. The first selected variable is x_1 and regions $0 \leq x_1 \leq 10$ and $10 \leq x_1 \leq 20$ are created. Since the left region has the lowest lower bound (36.3), it is examined at iteration 3 by branching on x_1. Both children nodes are found to be infeasible. The algorithm proceeds to node 4 for which v_1 is selected as a branching variable. The right region, $5.6 \leq v_1 \leq 10$, is fathomed since it has a lower bound greater than 99.2. The algorithm progresses along the branch-and-bound tree until, at iteration 8, only one node is left open with a lower bound of 99.2. This is within the specified relative tolerance, so the procedure is terminated.

3 Computational studies for the SMIN-αBB algorithm

A complete implementation of the algorithm has been developed, with links to the Outer Approximation and the Generalized Benders Decomposition algorithms that are provided as part of the MINOPT package (Schweiger *et al.*, 1997). This section is dedicated to the investigation of the performance of the proposed SMIN-αBB algorithm on a series of small test examples from the literature and three heat exchanger network synthesis problems. These latter examples are used to study the effect of branching and bound update strategies on the SMIN-αBB algorithm. All runs are performed on an HP-C160 with a relative tolerance of 10^{-3}.

3.1 Small examples

The examples used here have all appeared in the literature as test examples (Kocis and Grossmann, 1988; Floudas *et al.*, 1989; Ryoo and Sahinidis, 1995; Cardoso *et al.*, 1997). The results are summarized in Table 2.

Example 3.1.1 This example, which was first proposed by Kocis and Grossmann (1988), involves three binary variables and two continuous variables. The formulation is given by

$$
\begin{aligned}
\min_{\boldsymbol{x},\boldsymbol{y}} \quad & 2x_1 + 3x_2 + 1.5y_1 + 2y_2 - 0.5y_3 \\
s.t. \quad & x_1^2 + y_1 = 1.25 \\
& x_2^{1.5} + 1.5y_2 = 3 \\
& x_1 + y_1 \leq 1.6 \\
& 1.333x_2 + y_2 \leq 3 \\
& -y_1 - y_2 + y_3 \leq 0 \\
& x_1, x_2, x_3 \geq 0 \\
& (y_1, y_2, y_3) \in \{0,1\}^3.
\end{aligned}
$$

The value of the objective function at the global solution is 7.67 and the optimum solution vectors are $\boldsymbol{x}^* = (1.12, 1.31)^T$ and $\boldsymbol{y}^* = (0,1,1)^T$. The results shown in Table 2 are achieved using sequential branching, optimization-based bound updates for the continuous variables and interval-based bound updates for the binary variables. The branch-and-reduce algorithm of Ryoo and Sahinidis (1995) identifies the solution at the root node.

Example 3.1.2 This example is example 6.6.5 from Floudas (1995). The formulation involves two continuous variables and only one binary variable.

$$
\begin{aligned}
\min_{\boldsymbol{x},\boldsymbol{y}} \quad & -0.7y + 5(x_1 - 0.5)^2 + 0.8 \\
s.t. \quad & -e^{(x_1 - 0.2)} - x_2 \leq 0 \\
& x_2 + 1.1y \leq 1 \\
& x_1 - 1.2y \leq 0.2 \\
& 0.2 \leq x_1 \leq 1 \\
& -2.22554 \leq x_2 \leq -1 \\
& y \in \{0,1\}
\end{aligned}
$$

The global solution has an objective value of 1.07654 and the corresponding solution vectors are $\boldsymbol{x}^* = (0.94194, -2.1)^T$ and $y^* = 1$. The same strategy as in Example 3.1.1 is used.

Example 3.1.3 This example was proposed by Yuan *et al.* (1988). It involves three continuous variables and four binary variables. The formulation is

$$
\begin{aligned}
\min_{\boldsymbol{x},\boldsymbol{y}} \quad & (y_1-1)^2+(y_2-2)^2+(y_3-1)^2-\ln(y_4+1) \\
& \qquad + (x_1-1)^2+(x_2-2)^2+(x_3-3)^2 \\
s.t. \quad & y_1+y_2+y_3+x_1+x_2+x_3 \le 5 \\
& y_3^2+x_1^2+x_2^2+x_3^2 \le 5.5 \\
& y_1+x_1 \le 1.2 \\
& y_2+x_2 \le 1.8 \\
& y_3+x_3 \le 2.5 \\
& y_4+x_1 \le 1.2 \\
& y_2^2+x_2^2 \le 1.64 \\
& y_3^2+x_3^2 \le 4.25 \\
& y_2^2+x_3^2 \le 4.64 \\
& x_1, x_2, x_3 \ge 0 \\
& \boldsymbol{y} \in \{0,1\}^4.
\end{aligned}
$$

Since the binary variables participate nonlinearly in this problem, it must be reformulated to be solved by the SMIN-αBB algorithm. The new formulation involves seven continuous variables and four binary variables as well as four new linear equality constraints. This larger problem is a *convex* MINLP which can be solved using the OA/ER. The global solution has an objective value of 4.5796 and solution vectors $\boldsymbol{x}^* = (0.2, 0.8, 1.908)^T$ and $\boldsymbol{y}^* = (1,1,0,1)^T$. As expected, the SMIN-$\alpha$BB algorithm converges in only one iteration. The branch-and-reduce algorithm of Ryoo and Sahinidis (1995) explores between three and seven nodes.

Example 3.1.4 This example was first presented by Berman and Ashrafi (1993). It involves three continuous variables and eight binary variables that participate in a highly nonlinear manner.

$$
\begin{aligned}
& \min_{\boldsymbol{x},\boldsymbol{y}} \quad -x_1x_2x_3 \\
s.t. \quad & x_1 + 0.1^{y_1}0.2^{y_2}0.15^{y_3} = 1 \\
& x_2 + 0.05^{y_4}0.2^{y_5}0.15^{y_6} = 1 \\
& x_3 + 0.02^{y_7}0.06^{y_8} = 1 \\
& -y_1 - y_2 - y_3 \le -1 \\
& -y_4 - y_5 - y_6 \le -1 \\
& -y_7 - y_8 \le -1 \\
& 3y_1 + y_2 + 2y_3 + 3y_4 + 2y_5 + y_6 + 3y_7 + 2y_8 \le 10 \\
& 0 \le x_1, x_2, x_3 \le 1 \\
& \boldsymbol{y} \in \{0,1\}^8.
\end{aligned}
$$

The global optimum solution has an objective function value of -0.94347 and solution vectors $\boldsymbol{x}^* = (0.97, 0.9925, 0.98)^T$ and $\boldsymbol{y}^* = (0,1,1,1,0,1,1,0)^T$. For the SMIN-$\alpha$BB algorithm to solve this problem, it must be reformulated through a logarithmic transformation. For instance, the first constraint is equivalent to the linear constraint

$$\ln(1-x_1) - y_1 \ \ln(0.1) - y_2 \ \ln(0.2) - y_3 \ \ln(0.15) = 0.$$

Example 3.1.5 This example is a purely integer nonconvex problem presented in Pörn *et al.* (1997). It involves two variables.

$$
\begin{aligned}
& \min_{\boldsymbol{y}} \quad 7y_1 + 10y_2 \\
s.t. \quad & y_1^{1.2}y_2^{1.7} - 7y_1 - 9y_2 \le 24 \\
& -y_1 - 2y_2 \le 5 \\
& -3y_1 + y_2 \le 1 \\
& 4y_1 - 3y_2 \le 11 \\
& y_1, y_2 \in [1,5] \cap \mathbb{N}
\end{aligned}
$$

The global optimum solution is 31, with $(y_1, y_2) = (3,1)^T$. To be solved with the SMIN-αBB algorithm, this problem must be reformulated by expressing each integer variable y_i in terms of binary variables and new constraints:

$$y_i = 1 + z_{i1} + 2z_{i2} + 4z_{i3}; \quad z_{i1} + z_{i3} \le 1; \quad z_{i1} + z_{i2} \le 1; \quad z_{i1}, z_{i2}, z_{i3} \in \{0,1\}.$$

The y_i variables can then be treated as continuous variables and the nonconvex inequality can be underestimated using the α parameters. Pörn *et al.* (1997) solved this problem after re-formulating it as a convex MINLP.

Example 3.1.6 This example taken from Pörn *et al.* (1997) involves one continuous and one integer variable.

$$
\begin{aligned}
&\min_{x,y} \quad 3y - 5x \\
s.t. \quad & 2y^2 - 2y^{0.5} - 2x^{0.5}y^2 + 11y + 8x \leq 39 \\
& -y + x \leq 3 \\
& 2y + 3x \leq 24 \\
& 1 \leq x \leq 10 \\
& y \in [1, 6] \cap \mathbb{N}
\end{aligned}
$$

The optimal solution is -17, with $x^* = 4$ and $y^* = 1$.

3.2 Heat exchanger network synthesis

The performance of the SMIN-αBB algorithm is studied on three heat exchanger network synthesis problems, using the Chen approximation of the logarithmic mean temperature difference or the arithmetic mean temperature difference in order to compute the area. These larger problems are used to explore the effects of the options previously discussed.

3.2.1 Chen approximation

The design of a heat exchanger network involving two hot streams, two cold streams, one hot and one cold utility is first chosen. The formulation of Yee and Grossmann (1991) is used. The superstructure for this problem is shown in Figure 4. There are 12 possible matches and therefore 12 binary variables. The annualized cost of the network is expressed as the summation of the utility costs, the fixed charges for the required heat-exchangers and an area-based cost for each heat-exchanger. Since the Chen approximation for the logarithmic mean of the temperature difference is used (Chen, 1987), the area is a highly nonlinear function of the heat duty and the temperature differences at both ends of the heat exchanger. The binary variables, which represent the existence of a given heat-exchanger, participate linearly in the problem. All the constraints are linear.

Given the set ST of temperature locations, the number of stages NS, the set HP of hot process streams and the set CP of cold process streams, the general problem formulation is as follows.

$$
\begin{aligned}
\min \sum_{i\in HP} C_{CU} Q_{CU,i} &+ \sum_{j\in CP} C_{HU} Q_{HU,j} \\
&+ \sum_{i\in HP}\sum_{j\in CP}\sum_{k\in ST} CF_{ij} z_{ijk} + \sum_{i\in HP} CF_{i,CU} z_{CU,i} + \sum_{j\in CP} CF_{j,HU} z_{HU,j} \\
&+ \sum_{i\in HP}\sum_{j\in CP}\sum_{k\in ST} \frac{C_{ij} Q_{ijk}}{U_{ij}\left[\Delta T_{ijk}\Delta T_{ijk+1}\left(\Delta T_{ijk}+\Delta T_{ijk+1}\right)/2\right]^{\frac{1}{3}}} \\
&+ \sum_{i\in HP} \frac{CA,i,CU Q_{CU,i}}{U_{CU,i}\left[\Delta T_{CU,i}\left(T_{out,i}-T_{in,CU}\right)\left(\Delta T_{CU,i}+T_{out,i}-T_{in,CU}\right)/2\right]^{\frac{1}{3}}} \\
&+ \sum_{j\in CP} \frac{CA,j,HU Q_{HU,j}}{U_{HU,j}\left[\Delta T_{HU,j}\left(T_{in,HU}-T_{out,j}\right)\left(\Delta T_{HU,j}+T_{in,HU}-T_{out,j}\right)/2\right]^{\frac{1}{3}}}
\end{aligned}
$$

$$
\begin{aligned}
s.t. \quad \left(T_{in,i}-T_{out,i}\right) Fcp_i &= \sum_{k\in ST}\sum_{j\in CP} Q_{ijk} + Q_{CU,i}, && \forall\, i\in HP \\
\left(T_{out,j}-T_{in,j}\right) Fcp_j &= \sum_{k\in ST}\sum_{i\in HP} Q_{ijk} + Q_{HU,j}, && \forall\, j\in CP \\
\left(T_{i,k}-T_{i,k+1}\right) Fcp_i &= \sum_{j\in CP} Q_{ijk}, && \forall\, k\in ST,\ \forall\, i\in HP \\
\left(T_{j,k}-T_{j,k+1}\right) Fcp_j &= \sum_{i\in HP} Q_{ijk}, && \forall\, k\in ST,\ \forall\, j\in CP \\
T_{in,i} &= T_{i,1}, && \forall\, i\in HP \\
T_{in,j} &= T_{j,NS+1}, && \forall\, j\in CP \\
T_{i,k} &\geq T_{i,k+1}, && \forall\, k\in ST,\ \forall\, i\in HP \\
T_{j,k} &\geq T_{j,k+1}, && \forall\, k\in ST,\ \forall\, j\in CP \\
T_{out,i} &\leq T_{i,NS+1}, && \forall\, i\in HP \\
T_{out,j} &\geq T_{j,1}, && \forall\, j\in CP \\
\left(T_{i,NS+1}-T_{out,i}\right) Fcp_i &= Q_{CU,i}, && \forall\, i\in HP \\
\left(T_{out,j}-T_{j,1}\right) Fcp_j &= Q_{HU,j}, && \forall\, j\in CP \\
Q_{ijk} - \Omega z_{ijk} &\leq 0, && \forall\, k\in ST,\ \forall\, i\in HP,\ \forall\, j\in CP \\
Q_{CU,i} - \Omega z_{CU,i} &\leq 0, && \forall\, i\in HP \\
Q_{HU,j} - \Omega z_{HU,j} &\leq 0, && \forall\, j\in CP \\
T_{i,k} - T_{j,k} + \Gamma(1-z_{ijk}) &\geq \Delta T_{ijk}, && \forall\, k\in ST,\ \forall\, i\in HP,\ \forall\, j\in CP \\
T_{i,k+1} - T_{j,k+1} + \Gamma(1-z_{ijk}) &\geq \Delta T_{ijk+1}, && \forall\, k\in ST,\ \forall\, i\in HP,\ \forall\, j\in CP \\
T_{i,NS+1} - T_{out,CU} + \Gamma(1-z_{CU,i}) &\geq \Delta T_{CU,i}, && \forall\, i\in HP \\
T_{out,HU} - T_{j,1} + \Gamma(1-z_{HU,j}) &\geq \Delta T_{HU,j}, && \forall\, j\in CP \\
\Delta T_{ijk} &\geq 10, && \forall\, k\in ST,\ \forall\, i\in HP,\ \forall\, j\in CP \\
z_{ijk}, z_{CU,i}, z_{HU,j} &\in \{0,1\}, && \forall\, k\in ST,\ \forall\, i\in HP,\ \forall\, j\in CP
\end{aligned}
$$

The continuous variables are T_{ik}, the temperature of hot stream i at the hot end of stage k; T_{jk}, the temperature of cold stream j at the cold end of stage k, Q_{ijk}, the heat exchanged between hot stream i and cold stream j at temperature location k; $Q_{CU,i}$, the heat exchanged between hot stream i and the cold utility at temperature location k; $Q_{HU,j}$, the heat exchanged between cold stream j and the hot utility at temperature location k; ΔT_{ijk}, the temperature approach for the match of hot stream i and cold stream j at temperature location k; $\Delta T_{CU,i}$, the temperature approach for the match of hot stream i and the cold utility at temperature location k; $\Delta T_{HU,j}$, the temperature approach for the match of cold stream j and the hot utility at temperature location k. The binary variables are z_{ijk}, for the existence of a match between hot stream i and cold stream j at temperature location k; $z_{CU,i}$, for the existence of a match between hot stream i and the cold utility at temperature location k; $z_{HU,j}$, for the existence of a match between cold stream j and the hot utility at temperature location k.

The parameters are T_{in}, the inlet temperature of a stream; T_{out}, the outlet temperature; Fcp, the heat capacity flowrate of a stream; Ω, the upper bound on heat exchange; Γ, the upper bound on the temperature difference. The stream data for the problem are summarized in Table 3. There are two temperature intervals. The steam utility cost is C_{HU} = \$80/kW-yr and the cooling water cost is C_{CU} = \$15/kW-yr. The fixed charges CF for the heat exchangers amount to \$5500/yr, regardless of the type of heat exchanger (hot process stream/cold utility, cold process stream/hot utility or hot process stream/cold process stream). The cost coefficient for the area-dependent part of the heat exchanger costs is C = \$300/yr. The overall heat transfer coefficients are $U_{ij} = 0.5$ kW/m^{2}K for the hot stream-cold stream units, $U_{CU,i} = 0.83333$ kW/m^{2}K for the cold stream-hot utility units and $U_{HU,j} = 0.5$ kW/m^{2}K for the hot stream-cold utility units. The global optimum configuration involves six heat exchangers and is shown in Figure 5. It corresponds to an annualized network cost of \$154,997.

Due to the linear participation of the binary variables, the problem can be solved to global optimality using the SMIN-αBB algorithm. The area-dependent cost of the heat exchangers must be underestimated using the general convex lower bounding functions discussed in Section 2.1. The Outer Approximation algorithm is used to solve the underestimating convex MINLP at each node of the tree. When this MINLP is feasible, an upper bound on the objective function is obtained by solving the nonconvex MINLP locally in the same region. The effects of the branching and variable bound update strategies on convergence characteristics are studied for this example.

Among the continuous variables, the temperature variables, T_{ik} and T_{jk}, play a particular role. Decreasing the range of one of these variables results in a decrease of the range of some of the heat loads and stream temperature differences. To reflect this interdependence, the branching variable is selected from the set of stream temperature variables exclusively, even though they do not participate explicitly in the nonconvex terms. The z_{ijk} binary variables representing the existence of hot stream/cold stream heat exchangers affect a larger number of variables than the $z_{CU,i}$ and $z_{HU,j}$ variables representing utility heat exchangers. They are therefore assigned a higher branching priority. Finally, to maximize the effect of branching, the bounds on the heat loads are expressed as a function of the bounds on the stream temperatures and the binary variables:

$$\begin{aligned}
&0 \leq Q_{ijk} \leq z_{ijk}^U \ \min\left\{Fcp_i\left(T_{i,k}^U - T_{i,k+1}^L\right), Fcp_j\left(T_{j,k}^U - T_{j,k+1}^L\right)\right\}, \\
&\qquad\qquad\qquad\qquad\qquad\qquad \forall\, k \in ST,\ \forall\, i \in HP,\ \forall\, j \in CP, \\
&0 \leq Q_{CU,i} \leq z_{CU,i}^U\ Fcp_i\left(T_{i,NS+1}^U - T_{out,i}\right),\ \forall\, i \in HP, \\
&0 \leq Q_{HU,j} \leq z_{HU,i}^U\ Fcp_j\left(T_{out,j} - T_{j,1}^L\right), \forall\, j \in CP.
\end{aligned} \tag{6}$$

Similarly, the bounds on the temperature differences can be expressed in terms of the bounds on the branching variables.

$$10 \leq \Delta T_{ij1} \leq 10\ \left(1 - z_{ij1}^U\right) + z_{ij1}^U\ \max\left\{T_{i,1}^U - T_{j,1}^L, 10\right\}, \forall\, i \in HP,\ \forall\, j \in CP,$$

$$\begin{aligned}
&10 \leq \Delta T_{ijk} \leq 10\ \left(1 - z_{ijk-1}^U\right)\left(1 - z_{ijk}^U\right) \\
&\qquad + \left(1 - \left(1 - z_{ijk-1}^U\right)\left(1 - z_{ijk}^U\right)\right)\ \max\left\{T_{i,k}^U - T_{j,k}^L, 10\right\}, \\
&\qquad\qquad\qquad\qquad\qquad \forall\, i \in HP,\ \forall\, j \in CP, \forall 1 < k < NS, \\
&10 \leq \Delta T_{ijNS+1} \leq 10\ \left(1 - z_{ijNS}^U\right) + z_{ijNS}^U\ \max\left\{T_{i,NS+1}^U - T_{j,NS+1}^L, 10\right\}, \\
&\qquad\qquad\qquad\qquad\qquad\qquad \forall\, i \in HP,\ \forall\, j \in CP, \\
&10 \leq \Delta T_{CU,i} \leq 10\ \left(1 - z_{CU,i}^U\right) + z_{CU,i}^U\ \max\left\{T_{i,NS+1}^U - T_{out,CU}, 10\right\},\ \forall\, i \in HP, \\
&10 \leq \Delta T_{HU,j} \leq 10\ \left(1 - z_{HU,j}^U\right) + z_{HU,j}^U\ \max\left\{T_{out,HU} - T_{j,1}^L, 10\right\},\ \forall\, j \in CP.
\end{aligned} \tag{7}$$

Several combinations of the branching and variable bound update strategies are tested. Each run is labeled with a number and a letter. The number refers to the variable bound update strategy, as shown in Table 4. In all instances, the only continuous variables selected for bound updates are the temperatures. All other continuous variables are automatically updated through Eqs.(6) and (7). The letter relates to the branching strategy as shown in Table 5. In the case of hybrid branching, a binary variable z is defined as close to its bounds if its value z^* at the solution of the relaxed MINLP is such that $\min\{z^*, 1 - z^*\} \leq zdist$. The results are presented in Table 6.

Branching strategies – Run series 1 Although the nonconvexities arise only from the continuous variables, the SMIN-αBB algorithm does not converge after 1000 iterations when the continuous variable branching strategy is used (Run 1a). At that point, the relative difference between the upper and lower bound on the global solution is 2.5 % and the lower bound increases at the slow rate of $2\ 10^{-6}\%$ per iteration. Given this poor performance, this first branching strategy is dropped from consideration. Branching on the binary variables yields much better results. A total of 509 convex MINLPs and 798 convex NLPs are solved when using sequential branching (Run 1b). As a result, the average time per iteration is cut in half. The rate of lower bound improvement is also much faster and convergence is achieved in only 655 iterations. Higher improvement rates are obtained when using the hybrid strategy with $zdist = 0.1$ and $zdist = 0.2$. The deviation of the binary variables from integrality at the solution of a continuous relaxation is therefore a good measure of their effect on the quality of the lower bounding problem. However, in both hybrid branching runs, the majority of lower bounding problems are convex MINLPs. In addition, the identification of a good branching variable requires the local solution of a nonconvex NLP. Both these factors combine to increase the CPU requirement per iteration to twice that of Run 1b. Consequently, it appears that sequential branching is a better overall strategy.

Bound updates on the binary variables – Run series 2 & 3 In Run 2b, interval-based bound tightening on binary variables results in all binary variables being fixed at some of the nodes above level 12. Only 271 out of the 601 nodes explored in this initial part of the tree are MINLPs. A total of 830 NLPs are solved. Not only does the number of iterations decrease from Run 1b to Run 2b, but the CPU time per iteration is also cut by 50%, in spite of the added processing required by interval bound updates. Similar trends are obtained for the hybrid branching strategy of Runs 2c and 2d.

In all cases for run series 3, the use of the optimization-based approach for *all* bound updates leads to a decrease in the number of iterations but a slight increase in the CPU requirements. The combination of optimization-based bound updates for the continuous variables and interval-based bound updates for the binary variables therefore seems to provide sufficient tightening and it is adopted for the remaining runs.

The objective function cut – Run series 4 Due to the linearity of the constraints in the original formulation, the bound update problem without the objective function cut is an MILP which can be solved at relatively low computational expense. The nonlinear underestimator for the objective function transforms this problem into a more challenging convex MINLP. The inclusion of the new constraint leads to significant decreases in the number of iterations for convergence in Runs 4b, 4c and 4d. A reduction by almost tenfold is observed for Run 4b. The total CPU time, however, increases by a factor of two to three in all cases, reflecting the surge in computational requirements for each node.

Bound update frequency – Run series 5 Since the optimization-based procedure results in much tighter bounds when the bound improvement constraint is included, the value of this strategy is further investigated by varying the frequency of bound updates. As can be expected, the average time per iteration decreases with the frequency of bound updates. Overall, this compromise in the quality of the convex underestimators has a positive effect on the performance of the algorithm. For Runs 5b and 5c, the optimum frequency in terms of CPU time is every three iterations, while for Run 5d, it is every four iterations.

Number of bound update variables – Run series 6 Varying the number of variables for which bound updates are performed also decreases the average iteration costs and increases the total number of iterations. The best results are obtained by updating four temperature variables.

Summary of results In the best overall run (Run 6b), the heat exchanger network synthesis problem is solved in 180 iterations and 315 CPU seconds. The results presented highlight the trade-offs that must be made between the quality of the lower bounding problem and the time expended to generate it. They also demonstrate that the presence of binary variables can be exploited successfully to improve the performance of the SMIN-αBB algorithm both through branching and bound updates.

3.2.2 Arithmetic mean temperature difference

Two examples taken from Zamora and Grossmann (1998a) are presented. The general formulation for these examples is similar to that of the previous example. One significant difference lies in the expression used for the area of the heat exchangers, which is based on the arithmetic mean temperature difference rather than the logarithmic mean or the Chen approximation. In the present case, the area of a heater for cold stream j is therefore given by

$$\frac{C_{j,HU} Q_{HU,j}}{\frac{U_{HU,j}}{2}\left(\Delta T_{HU,j} + T_{in,HU} - T_{out,j}\right)}. \tag{8}$$

For a cooler for hot stream i, the area is approximated by

$$\frac{C_{i,CU} Q_{CU,i}}{\frac{U_{CU,i}}{2}\left(\Delta T_{CU,i} + T_{out,i} - T_{in,CU}\right)}. \tag{9}$$

Finally, for a heat exchanger between two streams i and j in interval k, it is given by

$$\frac{C_{ij} Q_{ijk}}{\frac{U_{ij}}{2}\left(\Delta T_{ijk} + \Delta T_{ijk+1}\right)}. \tag{10}$$

In addition, the assumption of no stream splitting is imposed by adding two new sets of linear constraints, namely

$$\sum_{j \in CP} z(i,j,k) = 1, \qquad \forall\, i \in HP,\ \forall k \in ST, \tag{11}$$

$$\sum_{i \in HP} z(i,j,k) = 1, \qquad \forall\, j \in CP,\ \forall k \in ST. \tag{12}$$

In the following examples, a new variable ΔT^{+}_{ijk} is introduced to replace $\Delta T_{ijk} + \Delta T_{ijk+1}$ in Eq.(10). Thus, the area is expressed as

$$\frac{C_{ij} Q_{ijk}}{\frac{U_{ij}}{2}\Delta T^{+}_{ijk}}, \quad \forall\, i \in HP,\ \forall\, j \in CP,\ \forall k \in ST, \tag{13}$$

and the new linear constraint

$$\Delta T^{+}_{ijk} = \Delta T_{ijk} + \Delta T_{ijk+1}, \quad \forall\, i \in HP,\ \forall\, j \in CP,\ \forall k \in ST, \tag{14}$$

is incorporated in the problem. Thanks to this transformation, all nonconvex terms in the problem are fractional, of the form x_1/x_2, and can therefore be underestimated using the convex envelope developed by Maranas and Floudas (1995). Thus, the area term for a cooler, Eq. (9), is replaced by a new variable $w_{CU,i}$ such that

$$\begin{aligned} w_{CU,i} \geq\ & \frac{C_{i,CU} Q^{L}_{CU,i}}{\frac{U_{CU,i}}{2}\left(\Delta T_{CU,i} + T_{out,i} - T_{in,CU}\right)} + \frac{C_{i,CU} Q_{CU,i}}{\frac{U_{CU,i}}{2}\left(\Delta T^{U}_{CU,i} + T_{out,i} - T_{in,CU}\right)} \\ & - \frac{C_{i,CU} Q^{L}_{CU,i}}{\frac{U_{CU,i}}{2}\left(\Delta T^{U}_{CU,i} + T_{out,i} - T_{in,CU}\right)} \end{aligned} \tag{15}$$

$$\begin{aligned} w_{CU,i} \geq\ & \frac{C_{i,CU} Q^{U}_{CU,i}}{\frac{U_{CU,i}}{2}\left(\Delta T_{CU,i} + T_{out,i} - T_{in,CU}\right)} + \frac{C_{i,CU} Q_{CU,i}}{\frac{U_{CU,i}}{2}\left(\Delta T^{L}_{CU,i} + T_{out,i} - T_{in,CU}\right)} \\ & - \frac{C_{i,CU} Q^{U}_{CU,i}}{\frac{U_{CU,i}}{2}\left(\Delta T^{L}_{CU,i} + T_{out,i} - T_{in,CU}\right)} \end{aligned} \tag{16}$$

and similar equations are used for a heater. The area cost for a hot stream-cold stream heat exchanger, as shown in Eq. (13) is replaced by the new variable w_{ijk} such that

$$w_{ijk} \geq \frac{C_{ij} Q^{L}_{ijk}}{\frac{U_{ij}}{2}\Delta T^{+}_{ijk}} + \frac{C_{ij} Q_{ijk}}{\frac{U_{ij}}{2}\Delta T^{+,U}_{ijk}} - \frac{C_{ij} Q^{L}_{ijk}}{\frac{U_{ij}}{2}\Delta T^{+,U}_{ijk}} \tag{17}$$

$$w_{ijk} \geq \frac{C_{ij} Q^{U}_{ijk}}{\frac{U_{ij}}{2}\Delta T^{+}_{ijk}} + \frac{C_{ij} Q_{ijk}}{\frac{U_{ij}}{2}\Delta T^{+,L}_{ijk}} - \frac{C_{ij} Q^{U}_{ijk}}{\frac{U_{ij}}{2}\Delta T^{+,L}_{ijk}} \tag{18}$$

Finally, in the examples studied here, branching is performed on the variables that appear in the fractional terms, therefore Eqs. (6-7) are not used to update the bounds on these variables.

Example 1 This corresponds to example 4 of Zamora and Grossmann (1998a). Two hot streams and two cold streams are considered in a network with three temperature intervals. There are 16 binary variables, 64 continuous variables and 108 linear constraints. There are 16 fractional terms to be underestimated, one for each heat exchanger. A minimum approach temperature of 1 K is used for this problem, providing a lower bound on the ΔT and ΔT^{+} variables. The required stream data are listed in Table 7. The steam cost is C_{HU} = \$80/kW-yr, the cooling water cost is C_{CU} = \$20/kW-yr. The overall heat transfer coefficient for the process stream heat exchangers and the coolers is 0.8 kW/m^2-K, and for the heaters, it is 1.2 kW/m^2-K. The cost coefficients for heat exchangers and coolers are CF = \$6250/yr and C = \$83.26/yr; for the heaters, they are CF = \$6250/yr and C = \$99.91/yr. The global solution of the problem has a network cost of \$74,711/yr and involves heat exchangers between hot stream H2 and the cooling water, streams H1 and C1 in temperature intervals 1 and 3, streams H1 and C2 in temperature interval 2, and streams H2 and C1 in temperature interval 2.

The number of iterations and the CPU time are reported for different branching and bound update strategies in Table 8. For this formulation, it is best not to branch on the binary variables at all. When the continuous branching variable selection is based on the term measure, the heat duty and temperature difference variables are branched on equally. A similar trend is observed when the maximum separation distance for fractional term is used. When the variable measure is used instead, only the heat duties are selected and better performance is obtained. This problem benefits from a tight initial lower bound of \$50,741/yr. Thus, while tightening the variable bounds enables the generation of a slightly higher initial lower bound (\$54,597/yr), the computational expense of this operation is not justified by the decrease in number of iterations. The best performance is obtained without any variable bound updates, when the algorithm converges in 69 iterations and 82.8 CPU seconds. Zamora and Grossmann (1998a) solved this problem with tailored underestimators after exploring 3 branch-and-bound nodes (156 CPU seconds on an IBM RISC/6000 workstation), for an absolute tolerance of 1.

Example 2 This problem is example 5 of Zamora and Grossmann (1998a). This time, three hot streams and two cold streams are considered over three temperature intervals.

There are therefore 23 binary variables, 90 continuous variables and 147 linear constraints. The stream data for this problem are listed in Table 9. The overall heat transfer coefficients are computed based on the film heat transfer coefficients listed in the table. The steam cost is C_{HU} = \$110/kW-yr and the cooling water cost is C_{CU} = \$10/kW-yr. The area cost of all heat exchangers is based on coefficients CF = \$7400/yr and C = \$80/yr. A minimum approach temperature of 1^oC is imposed. The global optimum solution has an annualized network cost of \$82,043/yr. It involves heat exchangers between hot streams H1 and H2 and the cooling water, cold stream C2 and the steam utility, streams H3 and C2 in temperature interval 1, and streams H3 and C1 in temperature interval 3.

The results of several runs are shown in Table 10. The same trends are observed as for the previous example. When branching is performed on the continuous variables only, using the variable measure, and no bound updates are involved, the SMIN-αBB algorithm identifies the global optimum solution in 65 iterations and only 577.6 CPU seconds. Zamora and Grossmann (1998a) solved this problem in 17 branch-and-bound nodes and 6 CPU hours on an IBM RISC/6000 workstation. They also found that local optimization systematically leads to suboptimal configurations, highlighting the need for a global optimization approach.

3.2.3 Summary of results

The SMIN-αBB algorithm performed very well on two heat exchanger network synthesis problems in which the arithmetic mean temperature difference served to approximate the area. In the first instance, a relative convergence of 10^{-3} for a problem involving two hot streams, two cold streams and three temperature intervals was achieved in 67 iterations and 83 CPU seconds on an HP-C160. For the second problem, involving three hot streams, two cold streams and three temperature intervals, convergence was reached in 65 iterations and 578 CPU seconds. The most successful strategy was based on using the convex envelope for fractional terms, the variable measure to select a branching variable among the heat duties and no variable bound tightening. In marked difference to the heat exchanger network synthesis problem with the Chen approximation, branching on binary variables does not prove a worthwhile strategy.

4 The GMIN-αBB algorithm

Although the SMIN-αBB algorithm presented in Sections 2 and 3 is applicable to a large class of problems, the participation of the binary variables is restricted to linear and mixed-bilinear terms. While many problems can be expressed in this form, equivalent but smaller problems can often be formulated by allowing greater flexibility in the functionality of the integer variables. The GMIN-αBB algorithm is therefore introduced to tackle the broad class of problems represented by

$$\begin{array}{lll} \min\limits_{\boldsymbol{x},\boldsymbol{y}} & f(\boldsymbol{x},\boldsymbol{y}) & \\ s.t. & \boldsymbol{g}(\boldsymbol{x},\boldsymbol{y}) & \leq 0 \\ & \boldsymbol{h}(\boldsymbol{x},\boldsymbol{y}) & = 0 \\ & \boldsymbol{x} & \in [\boldsymbol{x}^L, \boldsymbol{x}^U] \\ & \boldsymbol{y} & \in [\boldsymbol{y}^L, \boldsymbol{y}^U] \cap \mathcal{N}^q \end{array} \tag{19}$$

where $f(\boldsymbol{x},\boldsymbol{y})$, $\boldsymbol{g}(\boldsymbol{x},\boldsymbol{y})$ and $\boldsymbol{h}(\boldsymbol{x},\boldsymbol{y})$ are functions whose continuous relaxation is twice continuously differentiable, q is the number of integer variables and $\mathcal{N}$ is the set of integers.

The GMIN-αBB algorithm overcomes the limitations of the standard branch-and-bound approaches for MINLPs (Beale, 1977; Gupta and Ravindran, 1985; Ostrovsky *et al.*, 1990; Borchers and Mitchell, 1991; Quesada and Grossmann, 1992) by making use of the αBB algorithm. The solution space is explored by branching on at least one integer variable at each node. A lower bound on the solution of the problem within a given region is obtained by solving a continuous relaxation of the mixed-integer problem. The validity of the lower bound and hence the optimality properties of the final solution depend on the ability to identify the global optimal solution of the relaxation. For the general problems represented by (19), the efficient generation of a valid lower bound is a challenging issue and is discussed in the next section.

4.1 Generation of a valid lower bound

The nonconvex MINLP at the current node of the branch-and-bound tree is relaxed to a nonconvex NLP. The αBB global optimization algorithm for twice continuously differentiable NLPs is then used to obtain a valid lower bound at that node. At every iteration of the αBB algorithm, a lower bounding problem for the nonconvex NLP is formulated. It is derived in same way as problem (3), the lower bounding problem for the SMIN-αBB algorithm, in the case where there are no binary variables. One of two approaches can be used when running the αBB algorithm to get a lower bound on the continuous relaxation.

Strategy 1 Run the αBB algorithm to completion to obtain the global solution of the continuous relaxation,

Strategy 2 Run the αBB algorithm for a few iterations to obtain a valid lower bound on the global solution of the continuous relaxation.

In all branch-and-bound algorithms proposed to date, the lower bounding problem is solved to completion, following Strategy 1. Strategy 2 recognizes the fact that a lower bound on the solution of the relaxed MINLP is also a lower bound on the solution of the nonconvex MINLP. In order to ensure that the GMIN-αBB algorithm converges in a finite number of iterations, the validity of the lower bounds is not sufficient (Horst and Tuy, 1996) and two additional conditions must be met:

1. The global solution *must* be identified for nodes on the terminal level of the B&B tree, when all integer variables are fixed and the upper and lower bounding problems are identical.

2. A nondecreasing sequence of lower bounds must be generated for the GMIN-αBB algorithm (consistent bounding procedure),

If the continuous relaxation is solved to global optimality at every node (Strategy 1), these two conditions are always satisfied. The only way to meet the requirements of Condition 1 is to apply Strategy 1 at all terminal nodes of the branch-and-bound tree. In a consistent bounding procedure, the lower bound for any given node is equal to or greater than the lower bound generated for the parent node. This additional criterion must be tested whenever Strategy 2 is used and the αBB algorithm is stopped early at a non-terminal node.

4.1.1 Stopping criteria

Strategy 2 makes use of the fact that the rate of improvement of the lower bound on the global solution of a nonconvex NLP is usually high at early iterations of the αBB algorithm and then gradually tapers off. The number of iterations required to reach the slow convergence phase depends on the characteristics of the NLP. To illustrate this point, we define the relative quality, ρ, of the lower bound on the objective function at iteration i, $\underline{f}_i^{NLP}$, as $\rho = (\underline{f}_i^{NLP} - \underline{f}_0^{NLP})/(f^{NLP,*} - \underline{f}_0^{NLP})$, where $f^{NLP,*}$ is the global solution. The relative quality ρ is plotted in Figure 6 for five nonconvex continuous relaxations of the pump network synthesis problem to be presented in Section 5.2. In each case, some of the integer variables are fixed to the values indicated on the figure and the αBB is run for forty iterations. For $z = (0, 0, 1)$, the lower bound improves significantly over the first 30 iterations, while for $z = (1, 1, 0)$, rapid progress is seen for only 15 iterations.

To maximize the lower bound improvement rate for the GMIN-αBB algorithm, the αBB algorithm should not be run in the regions of slow convergence. An adaptive rule that monitors the progress of the αBB run is proposed in order to ensure that the algorithm is stopped when the convergence rate tapers off. It checks whether over the last m iterations, where m is a user-specified quantity, the convergence of the algorithm has at least improved by some fraction r. Let $\overline{f}_i^{NLP}$ be the best known upper bound on the solution of the NLP at iteration i and let $\underline{f}_i^{NLP}$ be the lower bound at iteration i. Then, the αBB run is allowed to proceed to iteration $i+1$, $i \geq m$, as long as

$$\frac{\left(\overline{f}_i^{NLP} - \underline{f}_i^{NLP}\right) - \left(\overline{f}_{i-m}^{NLP} - \underline{f}_{i-m}^{NLP}\right)}{\overline{f}_{i-m}^{NLP} - \underline{f}_{i-m}^{NLP}} \geq r. \tag{20}$$

Typically, m can be set to 3 iterations and r to 30%. To preserve the consistency of the bounding procedure, this stopping criterion must only be imposed on the current node *after* its lower bound $\underline{f}_i^{NLP}$ greater than or equal to $\underline{f}^*$, the lowest lower bound on the global solution of the original nonconvex MINLP.

An effective way to further reduce the computational expense of αBB runs is to make use of the best known upper bound on the global optimum solution of the nonconvex MINLP, $\overline{f}^*$. This can either be incorporated in the NLP as a constraint that specifies an upper bound on the objective function, or it can be added as a stopping criterion for the NLP algorithm. This latter option was chosen in the implementation.

The procedure used to generate a lower bound on a nonconvex MINLP for a non-terminal node is summarized in Figure 7.

4.2 Generation of a valid upper bound

An upper bound on the solution of a given node can be obtained in several ways. If the solution of the continuous relaxation is integer-feasible, that is, all the relaxed integer variables have integer values at the solution, this solution is both a lower and an upper bound on the current node. If the αBB algorithm was run for only a few iterations and the relaxed integer variables are integer at the lower bound, they can be fixed to these integer values and the resulting nonconvex NLP can be solved locally to yield an upper bound on the solution of the node. Finally, a set of integer values satisfying the integer constraints can be used to construct a nonconvex NLP whose local solutions are upper bounds on the solution of the current node.

4.3 Branching variable selection

A number of selection rules have been discussed in the literature on branch-and-bound algorithms for convex MINLPs: most fractional variable (Gupta and Ravindran, 1985; Ostrovsky *et al.*, 1990), branching priorities (Gupta and Ravindran, 1985) and pseudo-costs

(Benichou *et al.*, 1971; Gupta and Ravindran, 1985). However, rules that systematically lead to the optimal performance of the algorithm remain elusive. Several approaches have been implemented in the GMIN-αBB algorithm. As in the SMIN-αBB algorithm, an integer variable can be chosen randomly or according to branching priorities. An additional rule consists of selecting the most or least fractional variable at the solution of a continuous relaxation of the problem. This comes at no additional cost since a continuous relaxation is solved at each node. Whether the most or least fractional variable is used, the expected outcome is that the lower bounds on the children node should be dramatically different from that of the parent node. The bisection of the range of the selected variable takes place at the midpoint or at the solution of the lower bounding problem. It is also possible to branch on more than one variable at a given node, or to perform k-section on one of the variables.

4.4 Variable bound updates

The same general strategies are used for variable bound tightening as in the SMIN-αBB algorithm, namely an optimization-based or an interval-based approach. By taking advantage of the integrality of the $\boldsymbol{y}$ variables, these two techniques can be extended to update bounds on integer variables.

4.4.1 Optimization-based approach

In the optimization approach, the lower or upper bound on variable y_i is improved by first relaxing the integer variables, and generating a convex lower bounding NLP based on the constraints in the original problem and, optionally, the objective function cut. The minimization or maximization of y_i is taken as the objective. The final updated lower or upper bound on the integer variable is obtained by setting $y_i^L = \lceil y^* \rceil$ or $y_i^U = \lfloor y^* \rfloor$.

4.4.2 Interval-based approach

In the interval-based approach, an iterative procedure is followed based on an interval test which provides necessary conditions for the feasibility of the original constraints and the bound improvement constraint $f(\boldsymbol{x}, \boldsymbol{y}) \leq \overline{f}^*$, given the relaxed region $(\boldsymbol{x}, \boldsymbol{y}) \in [\boldsymbol{x}^L, \boldsymbol{x}^U] \times [\boldsymbol{y}^L, \boldsymbol{y}^U]$. The approach is similar to that followed for continuous variables in Section 2.3.2. The modified procedure to improve the lower (upper) bound on variable y_i ensures that the integrality of the variable bounds is maintained at every iteration, as shown in Table 11.

The procedure that was developed for the SMIN-αBB algorithm in order to tighten binary variable bounds is a special case of the general approach presented here.

4.5 Reducing the integrality gap

The use of continuous relaxation to solve a mixed-integer problem may result in a large integrality gap, especially at early nodes in the branch-and-bound tree. In order to narrow this gap, it is important to preserve the discrete nature of the problem as much as possible. We propose several measures which help achieve this goal.

4.5.1 Partial relaxation

To obtain a tighter lower bound, we can identify all *binary* variables in the problem which participate only in linear or mixed-bilinear terms. If such variables exist, they do not need to be relaxed to formulate the lower bounding problem. The partial continuous relaxation that results from this approach is a nonconvex MINLP of type (1) which can be solved to global optimality using the SMIN-αBB algorithm. The overall lower bounding strategy remains the same, but the αBB algorithm is replaced by the SMIN-αBB algorithm.

4.5.2 Modified branching and bound updates

For its success, the GMIN-αBB algorithm relies heavily on the ability of the αBB algorithm to solve the subproblems generated by the main driver. The use of specialized algorithms to solve subproblems is a common feature of branch-and-bound and decomposition algorithms, as exemplified by the OA/ER (Duran and Grossmann, 1986; Kocis and Grossmann, 1989) or the GBD (Geoffrion, 1972). In general, the interaction between the higher level driver and the algorithms it uses is limited to the exchange of subproblem formulation and solution. This level of communication has already been increased in order to improve the performance of the GMIN-αBB algorithm: the higher level driver imposes a new, tailored termination criterion on the αBB algorithm. By passing more information down to the αBB algorithm, further improvements can be obtained. Thus, although the integer variables are relaxed during any αBB run, their underlying integrality should be exploited in order to increase the quality of the lower bounds generated. The branching and variable bound update strategies of the αBB algorithm can be modified as follows.

Branching Whenever a relaxed integer variable is branched on within the αBB algorithm, the bounds on that variable should be rounded to the nearest integer. Thus, if branching takes place through a bisection at the midpoint for variable $y \in [y^L, y^U]$, the two new regions should correspond to $y \in [y^L, \lfloor y^M \rfloor]$ and $y \in [\lceil y^M \rceil, y^U]$, with $y^M = \frac{y^U + y^L}{2}$, as already happens in the higher level driver. This strategy in turn affects the theoretical basis of the overall algorithm: by communicating the integrality constraints to the lower level algorithm, we remove the clear-cut distinction between the mixed-integer problem and its relaxation. As a result, the lower bound generated by the αBB algorithm is not necessarily a valid lower bound on the continuous relaxation. However, it is *always* a lower bound on the global solution of the nonconvex MINLP at the current node. The theoretical limitation of the integrality gap is thus removed. The minimum achievable distance between the global solution of the nonconvex MINLP at the current node and the generated lower bound cannot be predicted, but it is smaller than or equal to the integrality gap.

Variable bound updates When variable bound updates are also used to improve the quality of the lower bounding problem during an αBB run, a larger reduction in the solution space can be achieved by applying one of the *integer* bound update strategies described in Section 4.4 to the relaxed $\boldsymbol{y}$ variables.

4.6 Algorithmic procedure

4.6.1 Flowchart

The upper and lower bound generation schemes, the branching strategies for integer variables and the bound update techniques are combined to produce the GMIN-αBB algorithm. The main steps of the algorithm are shown in Figure 8.

4.6.2 Illustrative example

The algorithmic procedure for the GMIN-αBB algorithm is illustrated using the same example as for the SMIN-αBB algorithm. The branch-and-bound tree is shown in Figure 9 using the same notation as previously. At the first node, the continuous relaxation of the nonconvex MINLP is solved for ten αBB iterations to yield a lower bound of 48.6. No upper bound is found. Next, the binary variable y_2 is chosen for branching and the continuous relaxation of the problem with $y_2 = 0$ is solved. A lower bound of 92.2 is found as the global solution to this nonconvex NLP. The variable y_1 is equal to 1 at this solution which therefore provides an upper bound on the global optimum solution of the nonconvex MINLP. The region $y_2 = 1$ is then examined and the lower bound on the the NLP is found to be 97.0 at the sixth αBB iteration. Since this is greater than the best upper on the global solution of the MINLP, the αBB run is terminated and this node can be fathomed. The bounds on the solution are within the pre-specified relative tolerance of 10^{-3} at the left child of node 1 and convergence has therefore been achieved. The run is terminated in 2 CPU seconds on an HP-C160.

5 Computational studies for the GMIN-αBB algorithm

The performance of the GMIN-αBB algorithm is tested on a series of literature problems and on MINLPs of a very general form, representing a pump network synthesis problem and problems from the paper cutting industry. All runs are carried out on an HP-C160, with a relative tolerance of 10^{-3}.

5.1 Literature examples

The same literature problems as for the SMIN-αBB algorithm are solved with the GMIN-αBB algorithm. Although no reformulation is needed in this case, both reformulated problems and original formulations are solved. The results are reported in Table 2.

Both the SMIN-αBB and GMIN-αBB algorithms performed well on this suite of small test problems: they were able to identify the global optimum solution with little computational effort. For larger problems, the SMIN-αBB algorithm yielded slightly better results than the GMIN-αBB algorithm, possibly because it does not require the relaxation of the integrality conditions on the integer variables. However, the generality of the GMIN-αBB algorithm means that no reformulation is necessary to solve highly nonconvex problems in the integer variables.

5.2 Pump network synthesis

In this example, taken from Westerlund *et al.* (1994), the aim is to identify the least costly configuration of centrifugal pumps that achieves a pre-specified pressure rise based on a given total flowrate. The problem belongs to the class of nonconvex MINLPs that can be addressed by the GMIN-αBB algorithm: it involves integer variables that participate in highly nonconvex terms. As an illustration, Figure 10 shows a schematic representation of a three level pump configuration, where each level corresponds to a different pump type. Figure 11 shows the global optimal configuration, corresponding to an annualized cost of 128,894 FIM[4].

The structural decisions for an L level superstructure are represented by a number of discrete variables. The binary variables z_i, $i = 1, \ldots, L$, denote the existence of level i. The

[4]FIM stands for Finmark

integer variables Np_i, $i = 1, \ldots, L$, denote the number of parallel lines at level i ($Np_i \in \{0, \ldots, NP\}$, where NP is the maximum number of parallel lines). The integer variables Ns_i, $i = 1, \ldots, L$, denote the number of pumps in series at level i ($Ns_i \in \{0, \ldots, NS\}$, where NS is the maximum number of pumps in series). The relevant continuous variables are the fraction of total flow going to level i, x_i, the flowrate on each line at level i, $\dot{v}_i$, the rotation speed of all pumps on level i, ω_i, the power requirements at level i, P_i and the pressure rise at level i, Δp_i. For a three-level superstructure, the minimization of the annualized cost is given by

$$
\begin{array}{rl}
\min & \sum_{i=1}^{3} (C_i + C_i' P_i) Np_i Ns_i z_i \\
s.t. & \sum_i x_i = 1 \\
& \left.\begin{array}{l}
P_i - \alpha_i (\frac{\omega_i}{\omega_{max}})^3 - \beta_i (\frac{\omega_i}{\omega_{max}})^2 \dot{v}_i - \gamma_i (\frac{\omega_i}{\omega_{max}}) \dot{v}_i^2 = 0 \\
\Delta p_i - a_i (\frac{\omega_i}{\omega_{max}})^2 - b_i (\frac{\omega_i}{\omega_{max}}) \dot{v}_i - c_i \dot{v}_i^2 = 0 \\
\dot{v}_i Np_i - x_i V_{tot} = 0 \\
\Delta P_{tot} z_i - \Delta p_i Ns_i = 0 \\
0 \leq x_i \leq 1, \quad 0 \leq \dot{v}_i \leq V_{tot} \\
0 \leq \omega_i \leq \omega_{max}, \quad 0 \leq P_i \leq P_i^{max} \\
0 \leq \Delta p_i \leq \Delta P_{tot} \\
Np_i \in \{0, 1, 2, 3\}, \quad Ns_i \in \{0, 1, 2, 3\} \\
z_i \in \{0, 1\}
\end{array}\right\} i = 1, 2, 3
\end{array}
\qquad (21)
$$

The parameter values are given in Table 12. The capital costs are annualized by multiplying the fixed pump costs by a factor of 0.1627 which corresponds to a 10% interest rate and a 10 year life. The operating costs are based on 6000 hours per year and an electricity cost of 0.3 FIM/kWh. An explicit formulation of this problem is given in Floudas *et al.* (1999).

Formulation (21) is nonconvex in the continuous and integer variables. Thirty seven local minima have previously been reported for this problem by Westerlund *et al.* (1994) who solved it locally only. Considering the range of values of the integer variables, over 32,000 combinations could be obtained. The reformulation/branch-and-bound global optimization algorithm of Smith and Pantelides (1997, 1999) required the exploration of 1238 nodes and over twelve hours of CPU time on a Sun SPARCstation 10/51. The GMIN-αBB algorithm is used here to tackle the problem. Different strategies are tested for lower bound generation, branching variable selection and variable bound tightening.

Some of the integer variables contribute more than others to the complexity of the problem. Specifically, the binary variables, z_i, associated with the existence or non-existence of a level, are important because they help reduce the size of the NLP problem by setting the variables associated with a particular level to zero when that level does not exist. The following additional linear constraints can reflect this fact.

$$\begin{array}{llll} P_i \leq z_i\ P_i^{max}; & \Delta p_i \leq z_i\ \Delta P_{tot}; & \dot{v}_i \leq z_i\ V_{tot}; & x_i \leq z_i; \\ \omega_i \leq z_i\ \omega_{max}; & Np_i \leq z_i\ NP; & Ns_i \leq z_i\ NS. & \end{array}$$

5.2.1 Lower bound generation

Because the relaxed NLP formulation is highly nonlinear, the αBB algorithm is not run to completion unless all integer variables are fixed. As discussed in Section 4.1, a balance must be found between the computational cost of generating a lower bound and its quality, that is, its proximity to the global solution of the relaxation. The problem is first solved by imposing different limits on the maximum number of αBB iterations, $iter_{max}$. Along the GMIN-αBB tree, the most fractional variable at the solution of the lower bounding problem is selected for branching and no bound updates are performed. When the αBB algorithm is entered, the bounds on all variables are updated once and the third branching strategy is used (see Section 2.2.1) to select a continuous or relaxed integer variable. The stopping criterion used for the αBB algorithm includes the maximum number of iterations, $iter_{max}$, the upper bound on the global optimum solution of the nonconvex MINLP and the lower bound on the relaxation of the parent node (see Figure 7). The results are shown in Runs 1 to 4 of Table 13. The average time requirements per node when $iter_{max}$ is two iterations only (Run 1) are twelve times smaller than when it is one hundred (Run 4). In Run 1, the αBB run is stopped upon reaching $iter_{max}$ for 53% of the nodes in the GMIN-αBB tree. Only one αBB iteration is performed at another 39% of the nodes. This means that, at the first iteration, the underestimating problem is found to be infeasible or its solution is greater than the upper bound on the global optimum solution of the MINLP. For the remaining 8% of the nodes, all integer variables are fixed and $iter_{max}$ can no longer be used as a stopping criterion. These nodes require between 2 and 135 iterations. In Run 3, the maximum number of αBB iterations is used as a stopping criterion for 52% of the nodes. For 35% of the GMIN-αBB nodes, only one αBB iteration is needed. The remaining 56 nodes require a wide array of run lengths, with 50 nodes involving less than twenty αBB iterations.

5.2.2 Branching strategy

The importance of the z_i variables suggests a branching scheme in which high priorities are assigned to these three binary variables. After the third level of the GMIN-αBB tree, the remaining integer variables are branched on based on the most fractional variable rule. The results for this strategy using different values of $iter_{max}$ are shown in Runs 5 to 8 of Table 13.

This strategy has a beneficial effect on all runs. The number of nodes generally decreases and the CPU requirements are reduced in all cases, and by up to 21%. The generation of tighter convex relaxations that results from fixing important variables first permits the fast elimination of an increased number of nodes.

5.2.3 Variable bound updates

The bounds on the integer variables can be updated at every iteration of the GMIN-αBB algorithm using the optimization and interval based approaches described in Section 4.4. The results using the optimization-based procedure are shown in Runs 9 to 12 of Table 13. The number of nodes decreases by at least 69% and the reduction in CPU time ranges from 63 to 79%, showing the value of bound updates for the integer variables. When the problem is solved with interval-based bound updates, this strategy does not result in a significant reduction of the variable ranges.

5.2.4 Automatic αBB run termination

The runs completed so far have demonstrated the importance of the maximum number of αBB iterations, the prioritization of the branching variables and the optimization-based bound updates for the success of a GMIN-αBB run. As discussed in Section 4.1, the ideal value of $iter_{max}$, the maximum number of αBB iterations, depends on the structure of the problem. The results shown in Table 14 are obtained by imposing the adaptive test of Eq. (20), with different values of m and r. The first conclusion that may be drawn from this table is that the automatic stopping criterion leads to runs of the same quality as a user-specified maximum number of iterations. Overall, the behavior of the algorithm seems quite robust to changes in the values of m and r. Figures 12 and 13 show the number of αBB iterations for each GMIN-αBB node as the search progresses for $r = 0.1$ and $m = 1$ or 15. The same general trends are observed in both cases. Initially, most αBB runs carry on for the minimum number of iterations as the continuous relaxation has a large feasible region and the progress of the lower bound is slow. Then, as more integer variables are fixed, the αBB algorithm is run for longer periods. This generates tight upper and lower bounds on the solution. In the latter stage of the GMIN-αBB run, nodes can be fathomed after only a few αBB iterations by using the best know upper bound. Imposing a minimum of fifteen αBB iterations before performing the first test of the rate of lower bound improvement clearly requires more resources than testing at every iteration.

5.2.5 Summary of results

The GMIN-αBB algorithm successfully solves a highly nonlinear mixed integer problem to global optimality. The different schemes described in Section 4 have a significant effect on the performance of the algorithm. In particular, a limited number of αBB iterations, the appropriate selection of a branching variable and tight variable bounds all contribute to increasing its efficiency. With this strategy, a relative convergence of 10^{-3} is achieved after 147 GMIN-αBB nodes and 927 CPU seconds on an HP-C160.

5.3 Trim loss minimization problems

Trim loss minimization problems arise in the paper cutting industry and have been studied in the context of integer nonconvex optimization by Harjunkoski *et al.* (1996) and Harjunkoski (1997), who have studied the use of reformulation to solve the problem. The main task is to cut out some paper products of different sizes from a large raw paper roll, in order to meet a customer's order. Each product paper roll is characterized by its width b. All product rolls are assumed to be of equal length. The raw paper roll has a width B_{max}. In general, it is not possible to cut out an entire order without throwing away some of the raw paper. The optimum cutting scheme minimizes the waste paper or *trim loss*. In order to identify the best overall scheme, a maximum number of different cutting patterns P is postulated, where a pattern is defined by the position of the knives. Each cutting pattern may have to be repeated several times in the overall scheme to meet the demand. There are N different product sizes and n_i rolls of size b_i must be cut. The existence of each pattern is denoted by a binary variable z_j, $j = 1, \dots, P$. The number of repeats of pattern j is denoted by the integer variable m_j. The number of products of size i in pattern j is given by the integer variable r_{ij}. A sample cut is shown in Figure 14 for

$$N = 3,\ n = \begin{pmatrix} 4 \\ 5 \\ 5 \end{pmatrix},\ m = \begin{pmatrix} 2 \\ 1 \\ 1 \end{pmatrix} \text{ and } r = \begin{pmatrix} 0 & 2 & 3 \\ 4 & 1 & 0 \\ 0 & 0 & 2 \end{pmatrix}.$$

Some additional constraints are imposed on the cutting patterns. For instance, each pattern must have a minimum total width of $B_{max} - \Delta$. The number of knives is limited to Nk_{max} so there can be no more than Nk_{max} products in a pattern. The general problem formulation is mostly linear except for the product-order constraints which are bilinear in the m_j and r_{ij} variables.

$$
\begin{aligned}
& \min_{m_j, y_j, r_{ij}} \sum_{j=1}^{P} c_j m_j + C_j y_j \\
s.t. \quad & \sum_j m_j \; r_{ij} \geq n_i, \; i = 1, \ldots, N \\
& (B_{max} - \Delta)\, y_j \leq \sum_i b_i \; r_{ij} \leq B_{max} \; y_j, \; j = 1, \ldots, P \\
& y_j \leq \sum_i r_{ij} \leq Nk_{max} \; y_j, \; j = 1, \ldots, P \\
& y_j \leq m_j \leq M_j y_j, \; j = 1, \ldots, P \\
& \sum_j m_j \geq \max \left\{ \left\lceil \frac{\sum_i n_i}{Nk_{max}} \right\rceil, \left\lceil \frac{\sum_i b_i n_i}{B_{max}} \right\rceil \right\} \\
& y_{k+1} \leq y_k, \; k = 1, \ldots, P-1 \\
& m_{k+1} \leq m_k, \; k = 1, \ldots, P-1 \\
& y_j \in \{0, 1\}, \; j = 1, \ldots, P \\
& m_j \in [0, M_j] \cap \mathbb{N}, \; j = 1, \ldots, P \\
& r_{ij} \in [0, Nk_{max}] \cap \mathbb{N}, \; i = 1, \ldots, N, \; j = 1, \ldots, P
\end{aligned}
$$

where $c_j = 1, \; j = 1, \ldots, P$ and $C_j = 0.1 \; j, j = 1, \ldots, P$.

This model is used for the four problems listed in Appendix B and discussed in (Floudas *et al.*, 1999). The number of combinations of the integer variables are respectively 10^{16}, 10^{16}, 10^{23} and 10^{31}. The set of *feasible* combinations is smaller, but cannot be determined *a priori*. The value of the objective function at the global optimum solution is shown in Table 15 for each problem, together with values of the variables. There are usually several globally optimum sets of patterns and only one is listed here. An interesting feature of these problems is the small size of the integrality gap (Table 15). It indicates that the generation of tight lower bounds on the nonconvex MINLPs should not be a bottleneck.

This set of trim loss minimization problems is solved using the GMIN-αBB algorithm with different options. The nature of the problem is such that the lower bound often reaches the value of the global optimum solution even though not all variables are integer. Since convergence can only be declared when an integer solution is found, the performance statistics can vary dramatically from run to run. Each problem is therefore solved one hundred times and the mean and standard deviation of the number of iterations and CPU time are reported here.

5.3.1 Branching variable selection

When the branching variable is selected based on the most fractional variable at the solution of the lower bounding problem, all runs take over one hour to converge. Since the non-existence of a pattern i ($y_i = 0$) eliminates the corresponding m_i and r_{ij} variables from the problem, the y variables are assigned the highest priority. The m variables are branched on next, and the r variables follow. The results using this branching strategy, a maximum of one hundred αBB iterations and no bound updates are shown in Table 16 under strategy (a). Although the average CPU time per run is greatly improved, the standard deviation on every run is large.

5.3.2 Variable bound updates

The bounds on all integer variables are now tightened through the optimization-based procedure. The results, presented in Table 16 under strategy (b), show that the mean number of nodes decreases by 53 to 93% and the mean CPU time decreases by 74 to 94%.

5.3.3 Automatic αBB run termination

The test of lower bound improvement rate is imposed on the αBB runs using values of m and r that were successful for the pump network synthesis example, namely $m = 3$ iterations and $r = 30\%$. The results, shown in Table 16 under strategy (c), indicate that the criterion used is too stringent for this problem as a few more nodes must be explored on average before the global optimum solution is found than when one hundred αBB iterations are allowed. However, the computational requirements are sufficiently close to the best values obtained so far to warrant adopting the automatic stopping criterion with $m = 3$ iterations and $r = 0.3$.

5.3.4 Summary of results

The GMIN-αBB algorithm succesfully identifies the global optimum solution of a series of nonconvex integer programs. In spite of the large number of possible combinations of the integer variables, all problems are solved to global optimality after exploring less than 100 nodes and in under one minute of CPU time on an HP-C160, for a relative tolerance of 10^{-3}. The trim loss minimization and pump network synthesis case studies highlight the benefits of new stopping criteria developed for the solution of the continuous relaxation. The selection of appropriate branching variables also plays a key role in ensuring high performance of the algorithm.

6 Comparison of the SMIN-αBB and GMIN-αBB algorithms

Any problem of the general form (19) can be transformed to an equivalent problem of the more restrictive form (1). The resulting problem can then be solved with the SMIN-αBB and the GMIN-αBB algorithm. Given this flexibility, the issue of algorithm choice must explored to determine whether any benefits can be derived by implementing the two algorithms, and what criteria should be applied to decide which algorithm to use.

In previous sections, the physical characteristics of the medium-size problems considered dictated the mathematical formulation used through the choice of integer or binary variables and the types of nonconvexities involved. When a problem was found to fall into the form shown in (1), it was solved with the SMIN-αBB algorithm. In all other cases, it was solved with the GMIN-αBB algorithm. The issue of algorithm selection is investigated in this section by comparing the performance of the SMIN-αBB and GMIN-αBB algorithms on the same problems. The heat exchanger problems are thus solved using the GMIN-αBB algorithm. All other problems are reformulated into form (1) and solved with the SMIN-αBB algorithm. The reformulation is also solved with the GMIN-αBB algorithm. Since the number of iterations or nodes have very different meanings for the two algorithms, they cannot be compared usefully. The most interesting result is therefore the CPU time, which is reported in Table 17 for the best set of options for each choice of algorithm and formulation. In the case where two different formulations are solved with the GMIN-αBB algorithm, the number of nodes is also noted.

For problems of form (1), the SMIN-αBB algorithm outperforms the GMIN-αBB algorithm in most cases. This can be attributed to the fact that the binary variables are not relaxed in the SMIN-αBB algorithm, leading to tighter lower bounds and improved convergence characteristics. However, a different conclusion is reached when comparing problems with unrestricted participation of the integer variables — solved by the GMIN-αBB algorithm — and their reformulated equivalent — solved by the SMIN-αBB algorithm. In every instance, better performance is found with the GMIN-αBB algorithm despite the relaxation of the integer variables. While no single factor explaining this behavior can be found, problem size may play a role as the number of variable increases through reformulation to form (1). The transformation of a single integer variable to a set of binary variables also leads to some disaggregation whereby a physically meaningful entity is considered as several independent mathematical objects during the solution procedure. Thus branching on an integer z at its midpoint creates two subregions which represent small values of z and large values of z

respectively. On the other hand, if z is expressed as a sum of n binaries $z = \sum_{i=0}^{n-1} 2^i y_i$, branching on a variable y_i, $0 < i < n-1$, leads to two regions void of physical meaning. As a result, the branching variable selection tends to be less effective for the reformulation.

Based on this series of comparative runs, it is therefore recommended to formulate the optimization problem using integer variables that closely reproduce the physical structure of the problem, and to avoid relaxation of the binary variables.

7 Conclusions

Two global optimization techniques for nonconvex MINLPs that capitalize on the advances of the αBB algorithm have been proposed. The first approach, the SMIN-αBB algorithm, identifies the global optimum solution of problems in which binary variables participate in linear or mixed-bilinear terms and continuous variables appear in continuous twice-differentiable functions. The partitioning of the solution space takes place in both the continuous and binary domains. The GMIN-αBB algorithm is designed to locate the global optimum solution of problems involving integer and continuous variables in functions whose continuous relaxation is continuous and twice-differentiable. In this algorithm, branching occurs on the integer variables only and a continuous relaxation of the problem is constructed during the bounding step. It uses the αBB algorithm for the efficient and rigorous generation of lower bounds. Both algorithms are widely applicable and have been successfully tested on a variety of medium-size nonconvex MINLPs. A comparative study has highlighted the importance of physically meaningful integer/binary variables in the formulation of the problem. This, in turn, motivates the need for algorithms which can handle binary or integer variables without reformulation.

Acknowledgments

The authors gratefully acknowledge financial support from the National Science Foundation, the Air Force Office of Scientific Research, the National Institutes of Health, Exxon Foundation and Mobil Technology Company.

References

Adjiman, C. S., I. P. Androulakis, and C. A. Floudas, "Global Optimization of MINLP Problems in Process Synthesis and Design," *Computers chem. Engng.*, **21**, S445 (1997).

Adjiman, C. S., I. P. Androulakis, and C. A. Floudas, "A Global Optimization Method, αBB, for General Twice-Differentiable Constrained NLPs – II. Implementation and Computational Results," *Computers and Chemical Engineering*, **22**, 1159 (1998b).

Adjiman, C. S., S. Dallwig, C. A. Floudas, and A. Neumaier, "A Global Optimization Method, αBB, for General Twice-Differentiable NLPs – I. Theoretical Advances," *Computers and Chemical Engineering*, **22**, 1137 (1998a).

Adjiman, C. S. and C. A. Floudas, "Rigorous Convex Underestimators for General Twice-Differentiable Problems," *J. of Glob. Opt.*, **9**, 23 (1996).

Adjiman, C. S., C. A. Schweiger, and C. A. Floudas, "Mixed-Integer Nonlinear Optimization in Process Synthesis," *Handbook of Combinatorial Optimization*, Du, D.-Z. and P. Pardalos, eds., Kluwer Academic Publishers (1999).

Androulakis, I. P., C. D. Maranas, and C. A. Floudas, "αBB : A Global Optimization Method for General Constrained Nonconvex Problems," *J. of Glob. Opt.*, **7**, 337 (1995).

Beale, E. M. L., "Integer Programming," *The State of the Art in Numerical Analysis*, 409–448, Academic Press (1977).

Benders, J. F., "Partitioning Procedures for Solving Mixed-Variables Programming Problems," *Numer. Math.*, **4**, 238 (1962).

Benichou, M., J. M. Gauthier, P. Girodet, G. Hentges, G. Ribiere, and O. Vincent, "Experiments in Mixed-Integer Linear Programming," *Math. Programming*, **1**(1), 76 (1971).

Berman, O. and N. Ashrafi, "Optimization Models for Reliability of Modular Software Systems," *IEEE Transactions on Software Engineering*, **19**(11), 1119 (1993).

Borchers, B. and J. E. Mitchell, "An Improved Branch and Bound Algorithm for Mixed Integer Nonlinear Programs," Tech. Rep. 200, Renssellaer Polytechnic Institute (1991).

Cardoso, M. F., R. L. Salcedo, S. Feyo de Azevedo, and D. Barbosa, "A Simulated Annealing Approach to the Solution of MINLP Problems," *Computers and Chemical Engineering*, **21**(12), 1349 (1997).

Chen, J. J. J., "Comments on Improvements on a Replacement for the Logarithmic Mean," *Chemical Engineering Science*, **42**(10), 2488 (1987).

Duran, D. A. and I. E. Grossmann, "An Outer-Approximation Algorithm for a Class of Mixed-Integer Nonlinear Programs," *Math. Prog.*, **36**, 307 (1986).

Falk, J. E. and R. M. Soland, "An Algorithm for Separable Nonconvex Programming Problems," *Management Science*, **15**(9), 550 (1969).

Floudas, C. A., *Nonlinear and Mixed-Integer Optimization. Fundamentals and Applications*, Oxford University Press, New York (1995).

Floudas, C. A., A. Aggarwal, and A. R. Ciric, "Global Optimum Search for Nonconvex NLP and MINLP Problems," *Computers and Chemical Engineering*, **13**(10), 1117 (1989).

Floudas, C. A. and I. E. Grossmann, "Algorithmic Approaches to Process Synthesis: Logic and Global Optimization," *FOCAPD'94*, vol. 91 of *AICHE Symposium Series*, 198–221 (1995).

Floudas, C. A., P. M. Pardalos, C. S. Adjiman, W. R. Esposito, Z. H. Gümüs, S. T. Harding, J. L. Klepeis, C. A. Meyer, and C. A. Schweiger, *Handbook of Test Problems in Local and Global Optimization*, Kluwer Academic Press (1999).

Geoffrion, A. M., "Generalized Benders Decomposition," *J. of Optimization Theory and Applications*, **10**, 237 (1972).

Grossmann, I. E., ed., *Global Optimization in Engineering Design*, Kluwer Academic Publishers, Dordrecht (1996).

Gupta, O. K. and R. Ravindran, "Branch and Bound Experiments in Convex Nonlinear Integer Programing," *Management Science*, **31**, 1533 (1985).

Hamed, A. and G. P. McCormick, "Calculation of Bounds on Variables Satisfying Nonlinear Equality Constraints," *J. Global Optimization*, **3**, 25 (1993).

Hansen, P., B. Jaumard, and S.-H. Lu, "An Analytical Approach to Global Optimization," *Mathematical Programming*, **52**, 227 (1991).

Harjunkoski, I., *Application of MINLP Methods on a Scheduling Problem in the Paper Converting Industry*, Ph.D. thesis, Åbo Akademi University, Åbo, Finland (1997).

Harjunkoski, I., T. Westerlund, J. Isaksson, and H. Skrifvars, "Different Formulations for Solving Trim Loss Problems in a Paper-Converting Mill with ILP," *Computers and Chemical Engineering*, **20**, S121 (1996).

Horst, R. and H. Tuy, *Global Optimization: Deterministic Approaches*, Springer-Verlag, Berlin (1996), 3rd edition.

Kocis, G. R. and I. E. Grossmann, "Relaxation Strategy for the Structural Optimization of Process Flow Sheets," *Industrial and Engineering Chemistry Research*, **26**, 1869 (1987).

Kocis, G. R. and I. E. Grossmann, "Global Optimization of Nonconvex Mixed-Integer Nonlinear Programming (MINLP) Problems in Process Synthesis," *I&ECR*, **27**, 1407 (1988).

Kocis, G. R. and I. E. Grossmann, "A Modeling and Decomposition Strategy for the MINLP Optimization of Process Flowsheets," *Computers and Chemical Engineering*, **13**(7), 797 (1989).

Lee, K. F., A. H. Masso, and D. F. Rudd, "Branch and Bound Synthesis of Integrated Process Designs," *Ind Eng Chem Fund*, **9**, 48 (1970).

Maranas, C. D. and C. Floudas, "Finding all solutions of nonlinearly constrained systems of equatio ns," *Journal of Global Optimization*, **7**(2), 143 (1995).

Maranas, C. D. and C. Floudas, "Global optimization in generalized geometric programming," *Computers chem. Engng.*, **21**(4), 351 (1997).

McCormick, G. P., "Computability of Global Solutions to Factorable Nonconvex Programs : Part I – Convex Underestimating Problems," *Math. Programming*, **10**, 147 (1976).

Moore, R. E., *Interval Analysis*, Prentice-Hall, Englewood Cliffs, NJ (1979).

Neumaier, A., *Interval Methods for Systems of Equations*, Encyclopedia of Mathematics and its Applications, Cambridge University Press (1990).

Ostrovsky, G. M., M. G. Ostrovsky, and G. W. Mikhailow, "Discrete Optimization of Chemical Processes," *Computers and Chemical Engineering*, **14**, 111 (1990).

Pörn, R., I. Harjunkoski, and T. Westerlund, "Convexification of Different Classes of Non-Convex MINLP Problems," Submitted to *Computers and Chemical Engineering.*

Quesada, I. and I. E. Grossmann, "An LP/NLP Based Branch and Bound Algorithm for Convex MINLP Optimization Problems," *Computers and Chemical Engineering*, **16**, 937 (1992).

Ryoo, H. S. and N. V. Sahinidis, "Global Optimization of Nonconvex NLPs and MINLPs with Applications in Process Design," *Computers chem. Engng*, **19**(5), 551 (1995).

Ryoo, H. S. and N. V. Sahinidis, "A Branch-and-Reduce Approach to Global Optimization," *J. Global Optimization*, **8**, 107 (1996).

Schweiger, C. A., A. Rojnuckarin, and C. A. Floudas, *MINOPT: a Software Package for Mixed-Integer Nonlinear Optimization, User's Guide*, Computer-Aided Systems Laboratory, Dept. of Chemical Engineering, Princeton University, NJ (1997), http://titan.princeton.edu/MINOPT/minopt.html.

Shectman, J. P. and N. V. Sahinidis, "A Finite Algorithm for Global Minimization of Separable Concave Programs," *J. Global Optimization*, **12**, 1 (1998).

Smith, E. M. B. and C. C. Pantelides, "Global Optimisation of General Process Models," *Global Optimization in Engineering Design*, Grossmann, I. E., ed., 355–386, Kluwer Academic Publishers (1996).

Smith, E. M. B. and C. C. Pantelides, "Global Optimisation of Nonconvex MINLPs," *Computers and Chemical Engineering*, **21**, S791 (1997).

Smith, E. M. B. and C. C. Pantelides, "A Symbolic Reformulation/Spatial Branch-and-Bound Algorithm for the Global Optimisation of Nonconvex MINLPs," *Computers and Chemical Engineering*, **23**, 457 (1999).

Vaidyanathan, R. and M. El-Halwagi, "Global Optimization of Nonconvex MINLP's by Interval Analysis," *Global Optimization in Engineering Design*, Grossmann, I. E., ed., 175–193, Kluwer Academic Publishers (1996).

Visweswaran, V. and C. A. Floudas, "New Properties and Computational Improvement of the GOP Algorithm for Problems with Quadratic Objective Function and Constraints," *Journal of Global Optimization*, **3**(3), 439 (1993).

Visweswaran, V. and C. A. Floudas, "New Formulations and Branching Strategies for the GOP Algorithm," *Global Optimization in Engineering Design*, Grossmann, I. E., ed., 75–100, Kluwer Academic Publishers (1996a).

Visweswaran, V. and C. A. Floudas, "Computational Results for an Efficient Implementation of the GOP Algorithm and its Variants," *Global Optimization in Engineering Design*, Grossmann, I. E., ed., 101–153, Kluwer Academic Publishers (1996b).

Westerberg, A. W. and G. Stephanopoulos, "Studies in Process Synthesis - I. Branch and Bound Strategy with List Techniques for the Synthesis of Separation Schemes," *Chem Eng Sci*, **30**, 963 (1975).

Westerlund, T. and F. Pettersson, "An Extended Cutting Plane Method for Solving Convex MINLP Problems," *Computers chem. Engng, Suppl.*, **19**, 131 (1995).

Westerlund, T., F. Pettersson, and I. E. Grossmann, “Optimization of Pump Configuration Problems as a MINLP Probem,” *Computers and Chemical Engineering*, **18**(9), 845 (1994).

Westerlund, T., H. Skrifvars, I. Harjunkoski, and R. Pörn, “An Extended Cutting Plane Method for a Class of Non-Convex MINLP Problems,” *Computers and Chemical Engineering*, **22**(3), 357 (1998).

Yee, T. F. and I. E. Grossmann, “Simultaneous Optimization Model for Heat Exchanger Network Synthesis,” *Chemical Engineering Optimization Models with GAMS*, Grossmann, I. E., ed., vol. 6 of *CACHE Design Case Studies Series* (1991).

Yuan, X., S. Zhang, L. Pibouleau, and D. S., “A Mixed-Integer Nonlinear-Programming Method for Process Design,” *RAIRO - Recherche Opérationnelle-Operations Research*, **22**(4), 331 (1988).

Zamora, J. M. and I. E. Grossmann, “A Global MINLP Optimization Algorithm for the Synthesis of Heat Exchanger Networks with no Stream Splits,” *Computers and Chemical Engineering*, **22**(3), 367 (1998a).

Zamora, J. M. and I. E. Grossmann, “Continuous Global Optimization of Structured Process Systems Models,” *Computers and Chemical Engineering*, **22**(12), 1749 (1998b).

A Maximum separation distance for fractional terms

The underestimation of a fractional term x_1/x_2 over the domain $[x_1^L, x_1^U] \times [x_2^L, x_2^U]$ with $x_2^L > 0$ involves the introduction of a new variable w_F and the linear constraints (Maranas and Floudas, 1995)

$$\begin{array}{lll} w_F & \geq & \begin{cases} \frac{x_1^L}{x_2} + \frac{x_1}{x_2^U} - \frac{x_1^L}{x_2^U} & \text{if}\, x_1^L \geq 0 \\ \frac{x_1}{x_2^U} - \frac{x_1^L x_2}{x_2^L x_2^U} + \frac{x_1^L}{x_2^L} & \text{if}\, x_1^L < 0 \end{cases} \\ w_F & \geq & \begin{cases} \frac{x_1^U}{x_2} + \frac{x_1}{x_2^L} - \frac{x_1^U}{x_2^L} & \text{if}\, x_1^U \geq 0 \\ \frac{x_1}{x_2^L} - \frac{x_1^U x_2}{x_2^L x_2^U} + \frac{x_1^U}{x_2^U} & \text{if}\, x_1^U < 0 \end{cases} \end{array} \tag{22}$$

In this appendix, the maximum separation distance d_{max} between x_1/x_2 and w_F is derived for all cases in the underestimating scheme.

A.1 Case 1: $x_1^L \geq 0$

In this case, $d_{max} = \max\limits_{x_1,x_2} \left[\frac{x_1}{x_2} - \max\left\{ \frac{x_1^L}{x_2} + \frac{x_1}{x_2^U} - \frac{x_1^L}{x_2^U}, \frac{x_1^U}{x_2} + \frac{x_1}{x_2^L} - \frac{x_1^U}{x_2^L} \right\} \right]$. First, we determine whether the solution to this problem lies on an edge of the rectangle $[x_1^L, x_1^U] \times [x_2^L, x_2^U]$. Let $x_1 = x_1^L$. Then,

$$d_{max} = \max_{x_2} \left[\frac{x_1^L}{x_2} - \max\left\{ \frac{x_1^L}{x_2}, \frac{x_1^U}{x_2} - \frac{x_1^U - x_1^L}{x_2^L} \right\} \right].$$

But,

$$\frac{x_1^L}{x_2} \geq \frac{x_1^U}{x_2} - \frac{x_1^U - x_1^L}{x_2^L} \Leftrightarrow \frac{x_1^U - x_1^L}{x_2^L} \geq \frac{x_1^U - x_1^L}{x_2}.$$

This is always true and so $d_{max} = 0$ for $x_1 = x_1^L$. It can also be shown that $d_{max} = 0$ for $x_1 = x_1^U$ or $x_2 = x_2^L$ or $x_2 = x_2^U$. Hence, the maximum separation distance does not occur on one of the edges of the box $[x_1^L, x_1^U] \times [x_2^L, x_2^U]$.

The problem is now reformulated as a constrained NLP:

$$d_{max} = \begin{cases} -\min\limits_{x_1,x_2,w_F} & -\frac{x_1}{x_2} + w_F \\ \quad s.t. & \frac{x_1^L}{x_2} + \frac{x_1}{x_2^U} - \frac{x_1^L}{x_2^U} - w_F \leq 0 \\ & \frac{x_1^U}{x_2} + \frac{x_1}{x_2^L} - \frac{x_1^U}{x_2^L} - w_F \leq 0 \\ & x_1^L \leq x_1 \leq x_1^U \\ & x_2^L \leq x_2 \leq x_2^U \end{cases} \tag{23}$$

The Lagrange multipliers for the bound constraints are equal to zero because the solution does not lie on an edge. The Lagrangian for problem (23) is given by

$$-\frac{x_1}{x_2} + w_F + \mu_1 \left(\frac{x_1^L}{x_2} + \frac{x_1}{x_2^U} - \frac{x_1^L}{x_2^U} - w_F \right) + \mu_2 \left(\frac{x_1^U}{x_2} + \frac{x_1}{x_2^L} - \frac{x_1^U}{x_2^L} - w_F \right).$$

Given that $x_2 > 0$, the KKT conditions are

$$\begin{aligned}
-\frac{1}{x_2} + \frac{\mu_1}{x_2^U} + \frac{\mu_2}{x_2^L} &= 0 && (24)\\
x_1 - \mu_1 x_1^L - \mu_2 x_1^U &= 0 && (25)\\
1 - \mu_1 - \mu_2 &= 0 && (26)\\
\mu_1\left(\frac{x_1^L}{x_2} + \frac{x_1}{x_2^U} - \frac{x_1^L}{x_2^U} - w_F\right) &= 0 && (27)\\
\mu_2\left(\frac{x_1^U}{x_2} + \frac{x_1}{x_2^L} - \frac{x_1^U}{x_2^L} - w_F\right) &= 0 && (28)\\
\mu_1, \mu_2 &\geq 0. && (29)
\end{aligned}$$

Eq. (26) is satisfied in one of the following three cases:

(i) $\mu_1 = 1$ and $\mu_2 = 0$,

(ii) $\mu_1 = 0$ and $\mu_2 = 1$,

(iii) $0 < \mu_1 < 1$ and $\mu_2 = 1 - \mu_1$.

If $\mu_1 = 1$ and $\mu_2 = 0$, Eq. (25) yields $x_1 = x_1^L$ and the solution is on an edge. Similarly, if $\mu_1 = 0$ and $\mu_2 = 1$, Eq. (25) yields $x_1 = x_1^U$ and the solution is on an edge. Thus, we must have $0 < \mu_1 < 1$ and $\mu_2 = 1 - \mu_1$. Substituting for μ_2 in Eq. (25), we find $\mu_1 = \frac{x_1^U - x_1}{x_1^U - x_1^L}$ and therefore, $\mu_2 = \frac{x_1 - x_1^L}{x_1^U - x_1^L}$. Using the expressions for the multipliers to solve the system of equations, the values of x_1 and x_2 at the optimum solution are

$$x_1^* = \frac{x_1^L + x_1^U}{2}, \quad x_2^* = 2\left(\frac{1}{x_2^U} + \frac{1}{x_2^L}\right)^{-1}.$$

Finally, the maximum separation distance for $x_1^L \geq 0$ is given by

$$d_{max} = \frac{x_1^L + x_1^U}{4}\left(\frac{1}{x_2^U} + \frac{1}{x_2^L}\right).$$

A.2 Case 2: $x_1^U < 0$

In this case, $d_{max} = \left[\max\limits_{x_1,x_2} \frac{x_1}{x_2} - \max\left\{\frac{x_1}{x_2^U} - \frac{x_1^L x_2}{x_2^L x_2^U} + \frac{x_1^L}{x_2^L}, \frac{x_1}{x_2^L} - \frac{x_1^U x_2}{x_2^L x_2^U} + \frac{x_1^U}{x_2^U}\right\}\right]$. First, we identify the region of validity of each underestimator:

$$\frac{x_1}{x_2^U} - \frac{x_1^L x_2}{x_2^L x_2^U} + \frac{x_1^L}{x_2^L} \geq \frac{x_1}{x_2^L} - \frac{x_1^U x_2}{x_2^L x_2^U} + \frac{x_1^U}{x_2^U} \Leftrightarrow x_2 \geq \frac{x_2^U - x_2^L}{x_1^U - x_1^L} x_1 + \frac{x_1^U x_2^L - x_1^L x_2^U}{x_1^U - x_1^L}.$$

The limit between the two regions is the diagonal line joining the lower left corner and the upper right corner of the box $[x_1^L, x_1^U] \times [x_2^L, x_2^U]$. Let Region 1 be the upper left of the rectangle and Region 2 be the lower right, as shown in Figure 15. The maximum separation distance for each of these two regions can now be determined.

A.2.1 Region 1

In this region, the maximum separation distance is given by

$$d_{max} = \begin{cases} -\min\limits_{x_1, x_2} & -\frac{x_1}{x_2} + \frac{x_1}{x_2^U} - \frac{x_1^L x_2}{x_2^L x_2^U} + \frac{x_1^L}{x_2^L} \\ s.t. & \frac{x_2^U - x_2^L}{x_1^U - x_1^L}(x_1 - x_1^L) + x_2^L - x_2 \leq 0 \\ & x_1^L \leq x_1 \leq x_1^U \\ & x_2^L \leq x_2 \leq x_2^U \end{cases} \tag{30}$$

Hypothesis 1: First, we postulate that one of the bound constraints is active. For $x_1 = x_1^U$, or $x_2 = x_2^L$, or $x_2 = x_2^U$, the solution to problem (30) reduces to $d_{max} = 0$ which is clearly not the optimum solution. For $x_1 = x_1^L$, we have

$$d_{max} = \begin{cases} -\min\limits_{x_2} & -\frac{x_1^L}{x_2} + \frac{x_1^L}{x_2^U} - \frac{x_1^L x_2}{x_2^L x_2^U} + \frac{x_1^L}{x_2^L} \\ s.t. & x_2^L - x_2 \leq 0 \\ & x_2 \leq x_2^U \end{cases} \tag{31}$$

The multipliers for the bound constraints in (31) must be zero, and the KKT conditions yield $(x_1^*, x_2^*) = (x_1^L, \sqrt{x_2^L x_2^U})$ and

$$d_{max} = \frac{-x_1^L}{x_2^L x_2^U}\left(\sqrt{x_2^L} - \sqrt{x_2^U}\right)^2. \tag{32}$$

Hypothesis 2: We now assume that the bound constraints are inactive. The Lagrangian is then

$$L(x_1, x_2, \mu) = -\frac{x_1}{x_2} + \frac{x_1}{x_2^U} - \frac{x_1^L x_2}{x_2^L x_2^U} + \frac{x_1^L}{x_2^L} + \mu\left(\frac{x_2^U - x_2^L}{x_1^U - x_1^L}(x_1 - x_1^L) + x_2^L - x_2\right).$$

The KKT conditions are

$$-\frac{1}{x_2} + \frac{1}{x_2^U} + \mu\frac{x_2^U - x_2^L}{x_1^U - x_1^L} = 0 \tag{33}$$

$$\frac{x_1}{x_2^2} - \frac{x_1^L}{x_2^L x_2^U} + \mu = 0 \tag{34}$$

$$\mu\left(\frac{x_2^U - x_2^L}{x_1^U - x_1^L}(x_1 - x_1^L) + x_2^L - x_2\right) = 0 \tag{35}$$

$$\mu \geq 0 \tag{36}$$

If $\mu = 0$, Eq. (33) gives $x_2 = x_2^U$, which contradicts the hypothesis. Hence $\mu > 0$. Solving Eqs. (33) and (35) simultaneously, two solutions are found for x_2, $x_2^* = \pm\sqrt{x_2^L x_2^U}$. However, only $x_2^* = \sqrt{x_2^L x_2^U}$ satisfies $x_2^L \leq x_2^* \leq x_2^U$. The following solution is then found for x_1

$$x_1^* = \frac{x_1^L x_2^U - x_1^U x_2^L + \left(x_1^U - x_1^L\right)\sqrt{x_2^L x_2^U}}{x_2^U - x_2^L}. \tag{37}$$

Substituting x_1^* and x_2^* into the definition of d_{max}, we find

$$d_{max} = \frac{x_1^L - x_1^U}{x_2^L - x_2^U} + \frac{x_1^L\, {x_2^U}^2 - x_1^U\, {x_2^L}^2}{x_2^L\, x_2^U\, (x_2^L - x_2^U)} + \frac{2\left(x_1^L - 2\, x_1^U\right)\sqrt{x_2^L}}{\left(x_2^L - x_2^U\right)\sqrt{x_2^U}} + \frac{2\, x_1^L}{\sqrt{x_2^L\, x_2^U}}. \tag{38}$$

The two hypotheses give different expressions for the maximum separation distance. In order to compare them, let x_1^* be defined by Eq. (37). Then,

$$d_{max}\left(x_1^L, \sqrt{x_2^L\, x_2^U}\right) \geq d_{max}\left(x_1^*, \sqrt{x_2^L\, x_2^U}\right) \tag{39}$$

$$\Leftrightarrow \frac{2x_1^L}{\sqrt{x_2^L x_2^U}} - \frac{x_1^L}{x_2^U} - \frac{x_1^L}{x_2^L} \geq \frac{x_1^* + x_1^L}{\sqrt{x_2^L x_2^U}} - \frac{x_1^*}{x_2^U} - \frac{x_1^L}{x_2^L} \tag{40}$$

$$\Leftrightarrow \left(x_1^L - x_1^*\right)\left(\frac{1}{\sqrt{x_2^L\, x_2^U}} - \frac{1}{x_2^U}\right) \geq 0 \tag{41}$$

$$\Leftrightarrow x_1^L - x_1^* \geq 0. \tag{42}$$

Since by definition $x_1^* > x_1^L$, the above condition is never met and d_{max} for Region 1 is always greatest at $\left(x_1^*, \sqrt{x_2^L\, x_2^U}\right)$.

A.2.2 Region 2

In the domain of validity of the second underestimator, the maximum separation distance is given by

$$d_{max} = \begin{cases} -\min\limits_{x_1, x_2} & -\frac{x_1}{x_2} + \frac{x_1}{x_2^L} - \frac{x_1^U x_2}{x_2^L x_2^U} + \frac{x_1^U}{x_2^U} \\ s.t. & \frac{x_2^U - x_2^L}{x_1^U - x_1^L}(x_1^L - x_1) + x_2 - x_2^L \leq 0 \\ & x_1^L \leq x_1 \leq x_1^U \\ & x_2^L \leq x_2 \leq x_2^U \end{cases} \tag{43}$$

This problem can be solved following the same approach as for Region 1. When the bound constraints are active, there exists a solution at $(x_1^U, \sqrt{x_2^L\, x_2^U})$ and the maximum separation distance at this point is

$$d_{max} = \frac{-x_1^U}{x_2^L x_2^U}\left(\sqrt{x_2^L} - \sqrt{x_2^U}\right)^2. \tag{44}$$

A comparison of this equation with Eq. (32) reveals that d_{max} at $(x_1^U, \sqrt{x_2^L\, x_2^U})$ is less than d_{max} at $(x_1^L, \sqrt{x_2^L\, x_2^U})$ and is, *a fortiori*, less than the value of d_{max} given by Eq. (38). When the bound constraints are inactive, the KKT conditions for problem (43) require the solution to lie on the line separating Regions 1 and 2. The solution of problem (43) is therefore the same as the solution of problem (30).

Thus, for $x_1^U < 0$, d_{max} is given by Eq. (38).

A.3 Case 3: $x_1^L < 0$ and $x_1^U \geq 0$

In this case, $d_{max} = \left[\max\limits_{x_1,x_2} \frac{x_1}{x_2} - \max\left\{\frac{x_1}{x_2^U} - \frac{x_1^L x_2}{x_2^L x_2^U} + \frac{x_1^L}{x_2^L}, \frac{x_1^U}{x_2} + \frac{x_1}{x_2^L} - \frac{x_1^U}{x_2^L}\right\}\right]$. The first underestimator is valid if and only if

$$\frac{x_1}{x_2^U} - \frac{x_1^L x_2}{x_2^L x_2^U} + \frac{x_1^L}{x_2^L} \geq \frac{x_1^U}{x_2} + \frac{x_1}{x_2^L} - \frac{x_1^U}{x_2^L} \tag{45}$$

$$\Leftrightarrow \left(\frac{1}{x_2^L} - \frac{1}{x_2^U}\right) x_1 + \frac{x_1^L}{x_2^L\, x_2^U} x_2 - \frac{x_1^U}{x_2} + \frac{x_1^L + x_1^U}{x_2^L} \leq 0. \tag{46}$$

Region 1 is defined as the region for which Eq. (46) holds.

A.3.1 Region 1

In this region, the maximum separation distance is given by

$$d_{max} = \begin{cases} -\min\limits_{x_1,x_2} & -\frac{x_1}{x_2} + \frac{x_1}{x_2^U} - \frac{x_1^L x_2}{x_2^L x_2^U} + \frac{x_1^L}{x_2^L} \\ s.t. & \left(\frac{1}{x_2^L} - \frac{1}{x_2^U}\right) x_1 + \frac{x_1^L}{x_2^L\, x_2^U} x_2 - \frac{x_1^U}{x_2} + \frac{x_1^L + x_1^U}{x_2^L} \leq 0 \\ & x_1^L \leq x_1 \leq x_1^U \\ & x_2^L \leq x_2 \leq x_2^U \end{cases} \tag{47}$$

Hypothesis 1: It is assumed that the solution lies on an edge of the box $[x_1^L, x_1^U] \times [x_2^L, x_2^U]$.

- For $x_1 = x_1^L$, problem (47) becomes

$$d_{max} = \begin{cases} -\min\limits_{x_2} & -\frac{x_1^L}{x_2} + \frac{x_1^L}{x_2^U} - \frac{x_1^L x_2}{x_2^L x_2^U} + \frac{x_1^L}{x_2^L} \\ s.t. & -\frac{x_1^L}{x_2^U} - \frac{x_1^U}{x_2^L} + \frac{x_1^L x_2}{x_2^L x_2^U} + \frac{x_1^U}{x_2} \leq 0 \\ & x_2^L \leq x_2 \leq x_2^U \end{cases} \tag{48}$$

The bound constraints on x_2 are inactive since $d_{max} = 0$ at the corner points. If the multiplier for the inequality constraint is zero, we must have $x_2 = \sqrt{x_2^L\, x_2^U}$ and

$$d_{max} = -\frac{x_1^L}{x_2^L\, x_2^U}\left(\sqrt{x_2^U} - \sqrt{x_2^L}\right)^2. \tag{49}$$

If it is nonzero, the KKT conditions reduce to a quadratic polynomial in x_2 whose solutions are x_2^L (i.e., $d_{max} = 0$) and $x_1^U x_2^U / x_1^L$ which is negative and therefore outside of the feasible region.

- For $x_1 = x_1^U$, problem (47) becomes

$$d_{max} = \begin{cases} -\min\limits_{x_2} & -\frac{x_1^U}{x_2} + \frac{x_1^U}{x_2^U} - \frac{x_1^L x_2}{x_2^L x_2^U} + \frac{x_1^L}{x_2^L} \\ s.t. & -\frac{x_1^U}{x_2^U} - \frac{x_1^L}{x_2^L} + \frac{x_1^L x_2}{x_2^L x_2^U} + \frac{x_1^U}{x_2} \leq 0 \\ & x_2^L \leq x_2 \leq x_2^U \end{cases} \tag{50}$$

If the multiplier for the inequality constraint is zero, the KKT conditions reduce to $x_2^2 = x_2^L x_2^U x_1^U / x_1^L$. But the right hand side is negative and there is therefore no solution to this problem. Similarly, if the multiplier is nonzero, the only feasible solution is at $x_2 = x_2^U$ and therefore $d_{max} = 0$.

• For $x_2 = x_2^L$, problem (47) becomes

$$d_{max} = \begin{cases} -\min\limits_{x_1} & \left(x_1 - x_1^L\right)\left(\frac{1}{x_2^U} - \frac{1}{x_2^L}\right) \\ s.t. & x_1 - \frac{x_1^L x_2^L}{x_2^L - x_2^U} \leq 0 \\ & x_1^L \leq x_1 \leq x_1^U \end{cases} \tag{51}$$

The KKT conditions require the multiplier for the inequality constraint to be non-zero. The solution to the problem lies at $x_1^* = x_1^L x_2^L / (x_2^L - x_2^U)$. Provided $x_1^L \leq x_1^* \leq x_1^U$, the maximum separation distance is then

$$d_{max} = -\frac{x_1^L}{x_2^L} \tag{52}$$

• For $x_2 = x_2^U$, problem (47) yields $d_{max} = 0$.

Hypothesis 2: It is now assumed that the solution does not lie on one of the edges of the box. The KKT conditions for the problem are then

$$-\frac{1}{x_2} + \frac{1}{x_2^U} + \mu\left(\frac{1}{x_2^L} - \frac{1}{x_2^U}\right) = 0 \tag{53}$$

$$\frac{x_1}{x_2^2} - \frac{x_1^L}{x_2^L x_2^U} + \mu\left(\frac{x_1^L}{x_2^L x_2^U} - \frac{x_1^U}{x_2^2}\right) = 0 \tag{54}$$

$$\mu\left(\left(\frac{1}{x_2^L} - \frac{1}{x_2^U}\right)x_1 + \frac{x_1^L}{x_2^L x_2^U}x_2 - \frac{x_1^U}{x_2} - \frac{x_1^L + x_1^U}{x_2^L}\right) = 0 \tag{55}$$

$$\mu \geq 0 \tag{56}$$

There is no solution to this system of equations for $\mu = 0$. The solution must therefore lie on the limit between Regions 1 and 2. Finding the optimum solution involves solving a cubic equation and this step is left until KKT conditions for Region 2 are obtained.

A.3.2 Region 2

In this region, the maximum separation distance is given by

$$d_{max} = \begin{cases} -\min\limits_{x_1, x_2} & \left(x_1^U - x_1\right)\left(\frac{1}{x_2} - \frac{1}{x_2^L}\right) \\ s.t. & \left(\frac{1}{x_2^U} - \frac{1}{x_2^L}\right)x_1 - \frac{x_1^L}{x_2^L x_2^U}x_2 + \frac{x_1^U}{x_2} - \frac{x_1^L + x_1^U}{x_2^L} \leq 0 \\ & x_1^L \leq x_1 \leq x_1^U \\ & x_2^L \leq x_2 \leq x_2^U \end{cases} \tag{57}$$

HYPOTHESIS 1: By forcing one of the bound constraints to be active, it is found that, for $x_1 = x_1^L$, there is no solution and for $x_1 = x_1^U$ or $x_2 = x_2^L$ or $x_2 = x_2^U$, $d_{max} = 0$.

HYPOTHESIS 2: If none of the bound constraints are active, the KKT conditions for the problem are

$$
\begin{aligned}
-\frac{1}{x_2} + \frac{1}{x_2^L} + \mu\left(\frac{1}{x_2^U} - \frac{1}{x_2^L}\right) &= 0 && (58)\\
\frac{x_1}{x_2^2} - \frac{x_1^U}{x_2^2} + \mu\left(\frac{x_1^U}{x_2^2} - \frac{x_1^L}{x_2^L x_2^U}\right) &= 0 && (59)\\
\mu\left(\left(\frac{1}{x_2^U} - \frac{1}{x_2^L}\right)x_1 - \frac{x_1^L}{x_2^L x_2^U}x_2 + \frac{x_1^U}{x_2} + \frac{x_1^L + x_1^U}{x_2^L}\right) &= 0 && (60)\\
\mu &\geq 0 && (61)
\end{aligned}
$$

Once again, there is no solution to this problem for $\mu = 0$ and the solution must lie on the border between the two regions. By definition, the two underestimators match at this border and therefore

$$
x_1 = \frac{x_1^U x_2^L x_2^U}{x_2^U - x_2^L}\frac{1}{x_2} - \frac{x_1^L}{x_2^U - x_2^L}x_2 - \frac{(x_1^U + x_1^L)x_2^U}{x_2^U - x_2^L} \qquad (62)
$$

The maximum separation distance d_{max} is then expressed as

$$
\max_{x_2^L \leq x_2 \leq x_2^U}\left(\frac{x_1^U x_2^L x_2^U}{(x_2^U - x_2^L)x_2^2} - \frac{3x_1^U x_2^U}{(x_2^U - x_2^L)x_2} + \frac{x_1^L x_2}{x_2^U - x_2^L} + \frac{2x_1^U x_2^U - x_1^U x_2^L + x_1^L x_2^U - x_1^L x_2^L}{(x_2^U - x_2^L)x_2^L}\right). \qquad (63)
$$

The maximum occurs at

$$
x_2 = \frac{A^{\frac{2}{3}} - 54^{\frac{2}{3}} x_1^L x_1^U x_2^L x_2^U}{54^{\frac{1}{3}} x_1^L A^{\frac{1}{3}}} \qquad (64)
$$

with $A = 54 x_1^L x_1^U x_2^L x_2^U \left(x_1^{L^2} + \sqrt{x_1^{L^4} + x_1^L x_1^U x_2^L x_2^U}\right)$. For certain values of the variable bounds the argument of the square root may be negative. HYPOTHESIS 2 can only hold if the value given by Eq. (64) is real and in $[x_2^L, x_2^U]$. If this is true, then the maximum separation distance under HYPOTHESIS 2 is given by

$$
\begin{aligned}
d_{max} = \; & \frac{54^{\frac{2}{3}} x_1^{L^2} x_1^U x_2^L x_2^U A^{\frac{2}{3}}}{\left(x_2^U - x_2^L\right)\left(A^{\frac{2}{3}} - 54^{\frac{2}{3}} x_1^L x_1^U x_2^L x_2^U\right)^2} - \frac{1458^{\frac{1}{3}} x_1^L x_1^U x_2^U}{\left(x_2^U - x_2^L\right)\left(A^{\frac{2}{3}} - 54^{\frac{2}{3}} x_1^L x_1^U x_2^L x_2^U\right)} \\
& \frac{A^{\frac{2}{3}} - 54^{\frac{2}{3}} x_1^L x_1^U x_2^L x_2^U}{54^{\frac{1}{3}}\left(x_2^U - x_2^L\right)x_2^L A^{\frac{1}{3}}} + \frac{2x_1^U x_2^U - x_1^U x_2^L + x_1^L x_2^U - x_1^L x_2^L}{\left(x_2^U - x_2^L\right)x_2^L}
\end{aligned} \qquad (65)
$$

Summarizing the results for Case 3, the maximum separation distance is given by the maximum of Eqs. (52) and (65).

B Formulations for the trim loss minimization problems

The complete formulation for every instance of the trim loss minimization example is given in this appendix. Problems 1 and 2 have four products, Problem 3 has five and Problem 4 has six.

B.1 Problem 1

$$
\begin{aligned}
& \min_{m_j, y_j, r_{ij}} m_1 + m_2 + m_3 + m_4 + 0.1y_1 + 0.2y_2 + 0.3y_3 + 0.4y_4 \\
s.t. \quad & m_1 r_{11} + m_2 r_{12} + m_3 r_{13} + m_4 r_{14} \geq 15 \\
& m_2 r_{21} + m_2 r_{22} + m_3 r_{23} + m_4 r_{24} \geq 28 \\
& m_3 r_{31} + m_2 r_{32} + m_3 r_{33} + m_4 r_{34} \geq 21 \\
& m_4 r_{41} + m_2 r_{42} + m_3 r_{43} + m_4 r_{44} \geq 30 \\
& 1750 y_1 \leq 290 r_{11} + 315 r_{21} + 350 r_{31} + 455 r_{41} \leq 1850 y_1 \\
& 1750 y_2 \leq 290 r_{12} + 315 r_{22} + 350 r_{32} + 455 r_{42} \leq 1850 y_2 \\
& 1750 y_3 \leq 290 r_{13} + 315 r_{23} + 350 r_{33} + 455 r_{43} \leq 1850 y_3 \\
& 1750 y_4 \leq 290 r_{14} + 315 r_{24} + 350 r_{34} + 455 r_{44} \leq 1850 y_4 \\
& y_1 \leq r_{11} + r_{21} + r_{31} + r_{41} \leq 5\ y_1 \\
& y_2 \leq r_{12} + r_{22} + r_{32} + r_{42} \leq 5\ y_2 \\
& y_3 \leq r_{13} + r_{23} + r_{33} + r_{43} \leq 5\ y_3 \\
& y_4 \leq r_{14} + r_{24} + r_{34} + r_{44} \leq 5\ y_4 \\
& y_1 \leq m_1 \leq 30 y_1 \\
& y_2 \leq m_2 \leq 30 y_2 \\
& y_3 \leq m_3 \leq 30 y_3
\end{aligned}
$$

$$
\begin{aligned}
& y_4 \leq m_4 \leq 30 y_4 \\
& m_1 + m_2 + m_3 + m_4 \geq 19 \\
& y_1 \geq y_2 \geq y_3 \geq y_4 \\
& m_1 \geq m_2 \geq m_3 \geq m_4 \\
& (y_1, y_2, y_3, y_4) \in \{0, 1\}^4 \\
& (m_1, m_2, m_3, m_4) \in [0, 30]^4 \cap \mathbb{N}^4 \\
& r_{ij} \in [0, 5] \cap \mathbb{N}, \ \ i = 1, \dots, 4, j = 1, \dots, 4.
\end{aligned}
$$

B.2 Problem 2

$$
\begin{aligned}
&\min_{m_j, y_j, r_{ij}} \; m_1 + m_2 + m_3 + m_4 + 0.1y_1 + 0.2y_2 + 0.3y_3 + 0.4y_4 \\
s.t.\quad & m_1 r_{11} + m_2 r_{12} + m_3 r_{13} + m_4 r_{14} \geq 9 \\
& m_2 r_{21} + m_2 r_{22} + m_3 r_{23} + m_4 r_{24} \geq 7 \\
& m_3 r_{31} + m_2 r_{32} + m_3 r_{33} + m_4 r_{34} \geq 12 \\
& m_4 r_{41} + m_2 r_{42} + m_3 r_{43} + m_4 r_{44} \geq 11 \\
& 1700y_1 \leq 330r_{11} + 360r_{21} + 385r_{31} + 415r_{41} \leq 1900y_1 \\
& 1700y_2 \leq 330r_{12} + 360r_{22} + 385r_{32} + 415r_{42} \leq 1900y_2 \\
& 1700y_3 \leq 330r_{13} + 360r_{23} + 385r_{33} + 415r_{43} \leq 1900y_3 \\
& 1700y_4 \leq 330r_{14} + 360r_{24} + 385r_{34} + 415r_{44} \leq 1900y_4 \\
& y_1 \leq r_{11} + r_{21} + r_{31} + r_{41} \leq 5\ y_1 \\
& y_2 \leq r_{12} + r_{22} + r_{32} + r_{42} \leq 5\ y_2 \\
& y_3 \leq r_{13} + r_{23} + r_{33} + r_{43} \leq 5\ y_3 \\
& y_4 \leq r_{14} + r_{24} + r_{34} + r_{44} \leq 5\ y_4 \\
& y_1 \leq m_1 \leq 15y_1 \\
& y_2 \leq m_2 \leq 12y_2 \\
& y_3 \leq m_3 \leq 9y_3 \\
& y_4 \leq m_4 \leq 6y_4 \\
& m_1 + m_2 + m_3 + m_4 \geq 8 \\
& y_1 \geq y_2 \geq y_3 \geq y_4 \\
& m_1 \geq m_2 \geq m_3 \geq m_4 \\
& (y_1, y_2, y_3, y_4) \in \{0, 1\}^4 \\
& (m_1, m_2, m_3, m_4) \in [0, 15] \times [0, 12] \times [0, 9] \times [0, 6] \cap \mathbb{N}^4 \\
& r_{ij} \in [0, 5] \cap \mathbb{N},\; i = 1, \ldots, 4, j = 1, \ldots, 4.
\end{aligned}
$$

B.3 Problem 3

$$
\begin{aligned}
&\min_{m_j,y_j,r_{ij}} m_1 + m_2 + m_3 + m_4 + m_5 + 0.1y_1 + 0.2y_2 + 0.3y_3 + 0.4y_4 + 0.5y_5 \\
s.t.\quad & m_1r_{11} + m_2r_{12} + m_3r_{13} + m_4r_{14} + m_5r_{15} \geq 12 \\
& m_1r_{21} + m_2r_{22} + m_3r_{23} + m_4r_{24} + m_5r_{25} \geq 6 \\
& m_1r_{31} + m_2r_{32} + m_3r_{33} + m_4r_{34} + m_5r_{35} \geq 15 \\
& m_1r_{41} + m_2r_{42} + m_3r_{43} + m_4r_{44} + m_5r_{45} \geq 6 \\
& m_1r_{51} + m_2r_{52} + m_3r_{53} + m_4r_{54} + m_5r_{55} \geq 9 \\
& 1800y_1 \leq 330r_{11} + 360r_{21} + 370r_{31} + 415r_{41} + 435r_{51} \leq 2000y_1 \\
& 1800y_2 \leq 330r_{12} + 360r_{22} + 370r_{32} + 415r_{42} + 435r_{52} \leq 2000y_2 \\
& 1800y_3 \leq 330r_{13} + 360r_{23} + 370r_{33} + 415r_{43} + 435r_{53} \leq 2000y_3 \\
& 1800y_4 \leq 330r_{14} + 360r_{24} + 370r_{34} + 415r_{44} + 435r_{54} \leq 2000y_4 \\
& 1800y_5 \leq 330r_{15} + 360r_{25} + 370r_{35} + 415r_{45} + 435r_{55} \leq 2000y_5 \\
& y_1 \leq r_{11} + r_{21} + r_{31} + r_{41} + r_{51} \leq 5\ y_1 \\
& y_2 \leq r_{12} + r_{22} + r_{32} + r_{42} + r_{52} \leq 5\ y_2 \\
& y_3 \leq r_{13} + r_{23} + r_{33} + r_{43} + r_{53} \leq 5\ y_3 \\
& y_4 \leq r_{14} + r_{24} + r_{34} + r_{44} + r_{54} \leq 5\ y_4 \\
& y_5 \leq r_{15} + r_{25} + r_{35} + r_{45} + r_{55} \leq 5\ y_5 \\
& y_1 \leq m_1 \leq 15y_1 \\
& y_2 \leq m_2 \leq 12y_2 \\
& y_3 \leq m_3 \leq 9y_3 \\
& y_4 \leq m_4 \leq 6y_4 \\
& y_5 \leq m_5 \leq 6y_5 \\
& m_1 + m_2 + m_3 + m_4 + m_5 \geq 10 \\
& y_1 \geq y_2 \geq y_3 \geq y_4 \geq y_5 \\
& m_1 \geq m_2 \geq m_3 \geq m_4 \geq m_5 \\
& (y_1, y_2, y_3, y_4, y_5) \in \{0, 1\}^5 \\
& (m_1, m_2, m_3, m_4, m_5) \in [0, 15] \times [0, 12] \times [0, 9] \times [0, 6] \times [0, 6] \cap \mathbb{N}^5 \\
& r_{ij} \in [0, 5] \cap \mathbb{N},\ \ i = 1, \ldots, 5, j = 1, \ldots, 5.
\end{aligned}
$$

B.4 Problem 4

$$\min_{m_j, y_j, r_{ij}} m_1 + m_2 + m_3 + m_4 + m_5 + m_6 + 0.1y_1 + 0.2y_2 + 0.3y_3 + 0.4y_4 + 0.5y_5 + 0.6y_6$$

$$\begin{aligned}
s.t.\quad & m_1 r_{11} + m_2 r_{12} + m_3 r_{13} + m_4 r_{14} + m_5 r_{15} + m_6 r_{16} \geq 8 \\
& m_1 r_{21} + m_2 r_{22} + m_3 r_{23} + m_4 r_{24} + m_5 r_{25} + m_6 r_{26} \geq 16 \\
& m_1 r_{31} + m_2 r_{32} + m_3 r_{33} + m_4 r_{34} + m_5 r_{35} + m_6 r_{36} \geq 12 \\
& m_1 r_{41} + m_2 r_{42} + m_3 r_{43} + m_4 r_{44} + m_5 r_{45} + m_6 r_{46} \geq 7 \\
& m_1 r_{51} + m_2 r_{52} + m_3 r_{53} + m_4 r_{54} + m_5 r_{55} + m_6 r_{56} \geq 14 \\
& m_1 r_{61} + m_2 r_{62} + m_3 r_{63} + m_4 r_{64} + m_5 r_{65} + m_6 r_{66} \geq 16 \\
& 2100y_1 \leq 330r_{11} + 360r_{21} + 380r_{31} + 430r_{41} + 490r_{51} + 530r_{61} \leq 2200y_1 \\
& 2100y_2 \leq 330r_{12} + 360r_{22} + 380r_{32} + 430r_{42} + 490r_{52} + 530r_{62} \leq 2200y_2 \\
& 2100y_3 \leq 330r_{13} + 360r_{23} + 380r_{33} + 430r_{43} + 490r_{53} + 530r_{63} \leq 2200y_3 \\
& 2100y_4 \leq 330r_{14} + 360r_{24} + 380r_{34} + 430r_{44} + 490r_{54} + 530r_{64} \leq 2200y_4 \\
& 2100y_5 \leq 330r_{15} + 360r_{25} + 380r_{35} + 430r_{45} + 490r_{55} + 530r_{65} \leq 2200y_5 \\
& 2100y_6 \leq 330r_{16} + 360r_{26} + 380r_{36} + 430r_{46} + 490r_{56} + 530r_{66} \leq 2200y_6 \\
& y_1 \leq r_{11} + r_{21} + r_{31} + r_{41} + r_{51} + r_{61} \leq 5\ y_1 \\
& y_2 \leq r_{12} + r_{22} + r_{32} + r_{42} + r_{52} + r_{62} \leq 5\ y_2 \\
& y_3 \leq r_{13} + r_{23} + r_{33} + r_{43} + r_{53} + r_{63} \leq 5\ y_3 \\
& y_4 \leq r_{14} + r_{24} + r_{34} + r_{44} + r_{54} + r_{64} \leq 5\ y_4 \\
& y_5 \leq r_{15} + r_{25} + r_{35} + r_{45} + r_{55} + r_{65} \leq 5\ y_5 \\
& y_6 \leq r_{16} + r_{26} + r_{36} + r_{46} + r_{56} + r_{66} \leq 5\ y_6 \\
& y_1 \leq m_1 \leq 15y_1 \\
& y_2 \leq m_2 \leq 12y_2 \\
& y_3 \leq m_3 \leq 8y_3 \\
& y_4 \leq m_4 \leq 7y_4 \\
& y_5 \leq m_5 \leq 4y_5 \\
& y_6 \leq m_6 \leq 2y_6 \\
& m_1 + m_2 + m_3 + m_4 + m_5 + m_6 \geq 16 \\
& y_1 \geq y_2 \geq y_3 \geq y_4 \geq y_5 \geq y_6 \\
& m_1 \geq m_2 \geq m_3 \geq m_4 \geq m_5 \geq m_6 \\
& (y_1, y_2, y_3, y_4, y_5, y_6) \in \{0,1\}^6 \\
& (m_1, m_2, m_3, m_4, m_5, m_6) \in [0,15] \times [0,12] \times [0,8] \times [0,7] \times [0,4] \times [0,2] \cap \mathbb{N}^6 \\
& r_{ij} \in [0,5] \cap \mathbb{N},\ i = 1, \ldots, 6, j = 1, \ldots, 6.
\end{aligned}$$

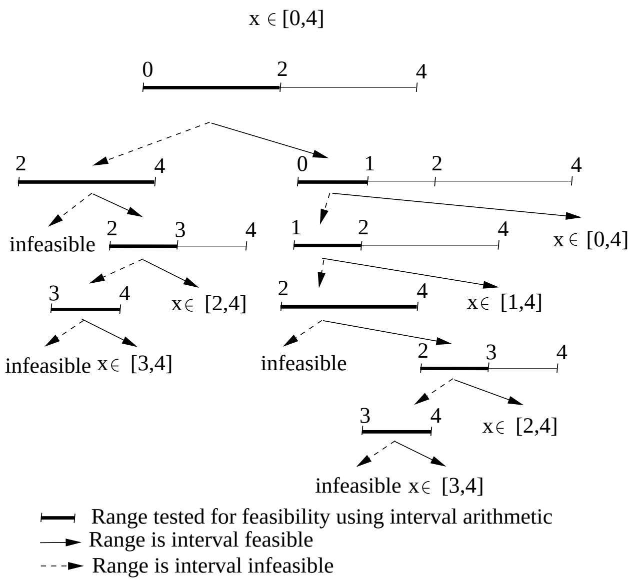

Figure 1: Interval bound update procedure for the lower bound for $0 \leq x \leq 4$. The minimum range width for testing is 1.

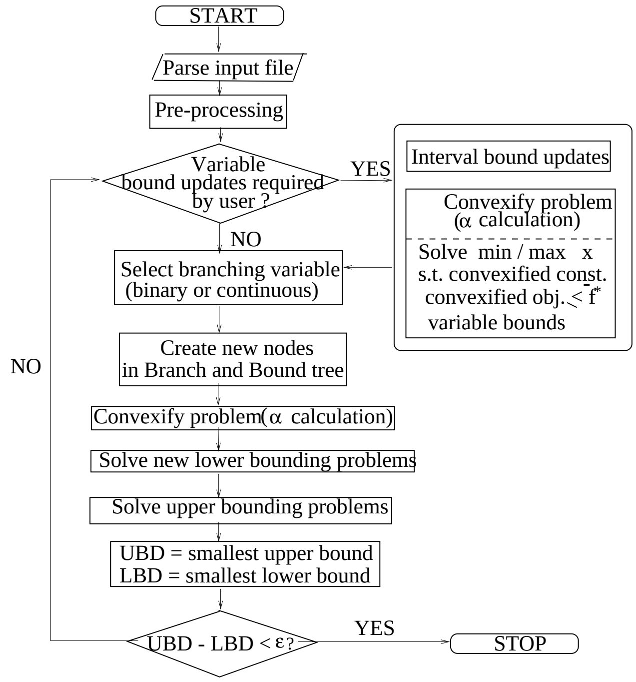

Figure 2: Flowchart for the SMIN-αBB algorithm

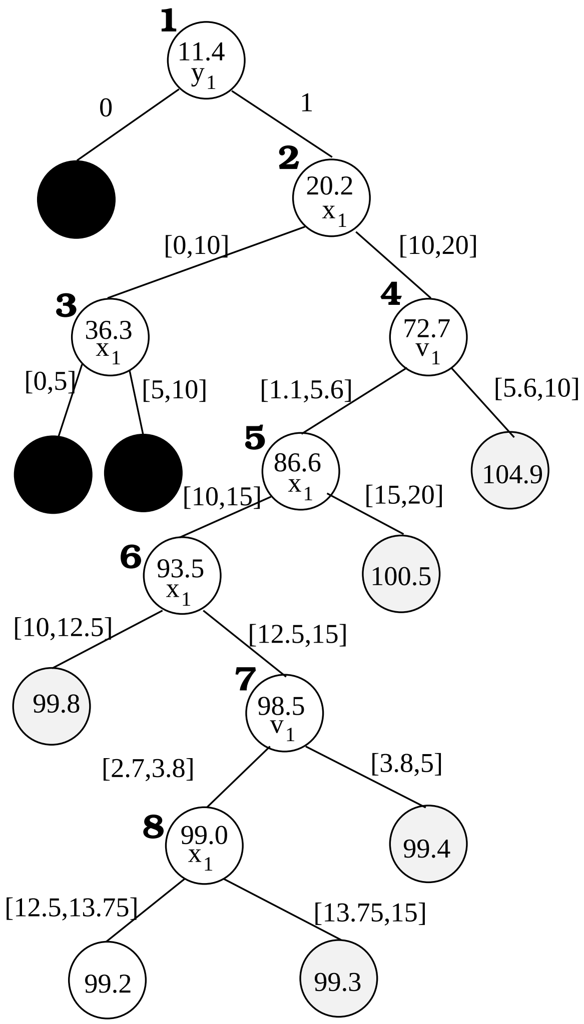

Figure 3: SMIN-αBB branch-and-bound tree. Boldface numbers indicate the order in which the nodes were explored. The lower bound and branching variable are indicated inside each node. The range to which this branching variable is restricted is displayed along each branch. Note that the bounds shown are based on the results of the bound tightening procedure. A black node indicates the lower bounding problem was found infeasible and a shaded node is fathomed because its lower bound is greater than the current upper bound on the solution.

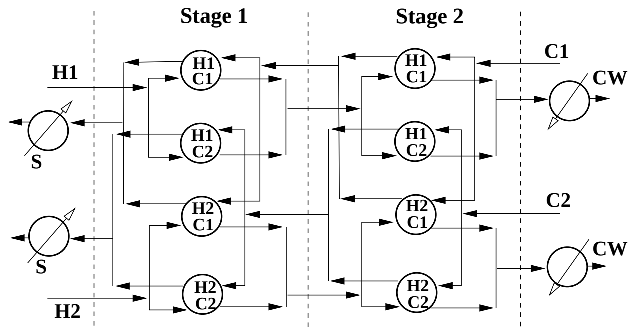

Figure 4: Superstructure for the heat exchanger network problem.

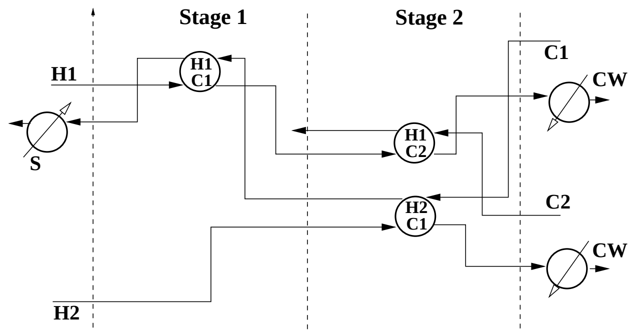

Figure 5: Optimum configuration for the heat exchanger network problem.

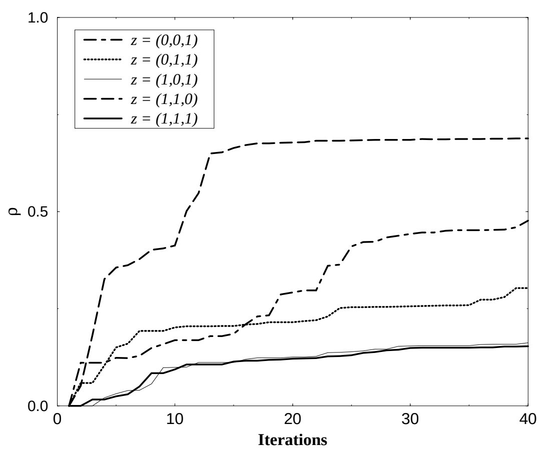

Figure 6: Progress of ρ, the relative quality of the lower bound, for five continuous relaxations.

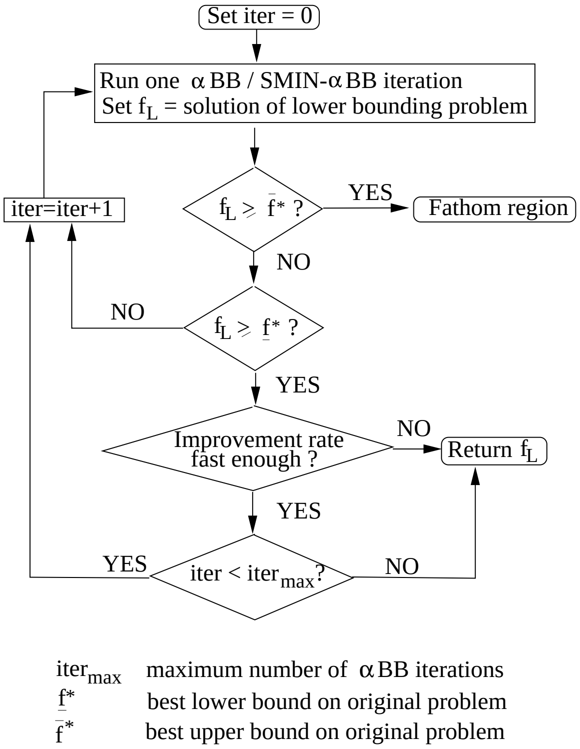

Figure 7: Generation of a lower bound on a nonconvex MINLP at a non-terminal node, using Strategy 2.

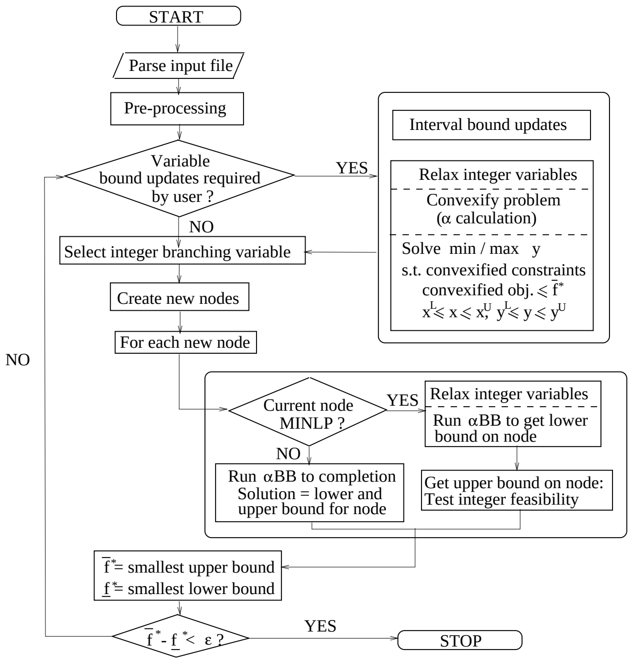

Figure 8: Flowchart for the GMIN-αBB algorithm

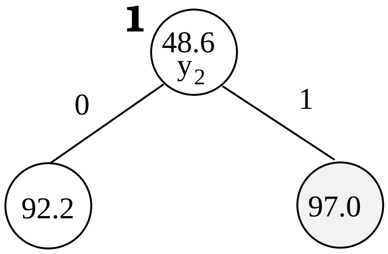

Figure 9: GMIN-αBB branch-and-bound tree

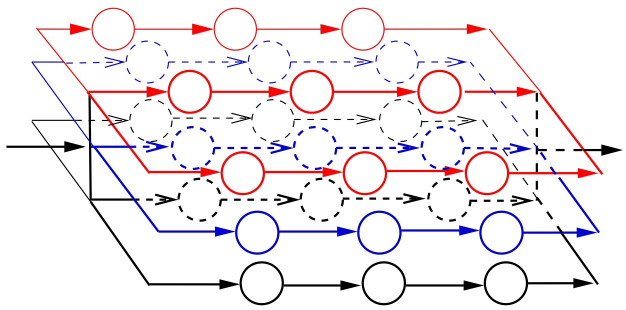

Figure 10: Superstructure for the pump network synthesis problem

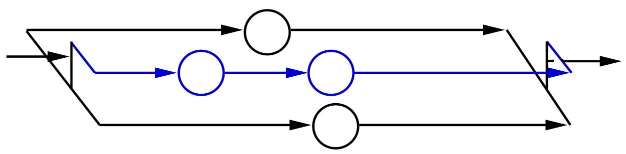

Figure 11: Global optimal configuration for pump network synthesis problem

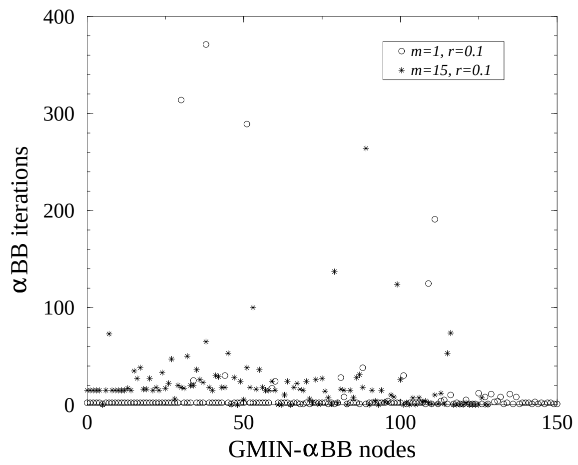

Figure 12: Number of αBB iterations per GMIN-αBB node for two values of m.

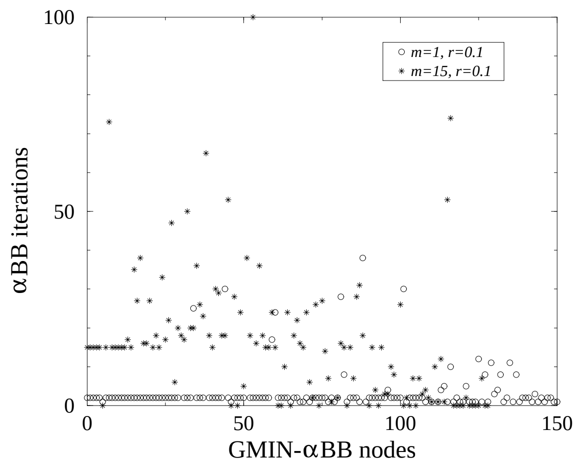

Figure 13: αBB runs with less than one hundred iterations for two values of m (close-up of Figure 12).

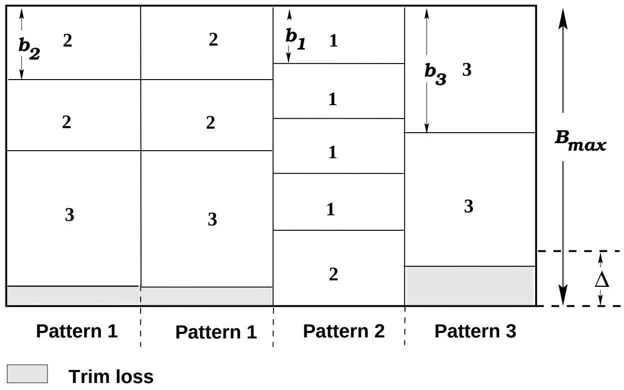

Figure 14: Trim loss minimization problem.

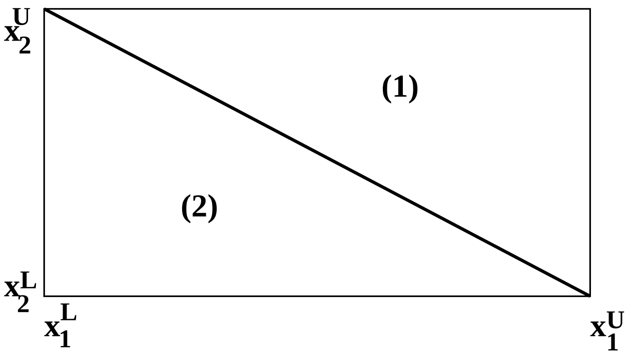

Figure 15: Two regions for Case 2, $x_1^U < 0$.

```
PROCEDURE interval-based continuous variable bound update()
  Set initial bounds L = x_i^L and U = x_i^U;
  Set iteration counter k = 0, maximum iteration number K;
  DO k < K
    Compute midpoint M = (U + L)/2;
    Set left region {(x, y) : (x, y) ∈ F, x_i ∈ [L, M]};
    Set right region {(x, y) : (x, y) ∈ F, x_i ∈ [M, U]};
    Test interval feasibility of left (right) region;
    IF feasible, set U = M  (L = M);
    ELSE, test interval feasibility of right (left) region;
      IF feasible, set L = M  (U = M);
      ELSE, set L = U  (U = L) and U = x_i^U  (L = x_i^L);
    IF k = 0 and L = x_i^U  (U = x_i^L), RETURN(infeasible node);
    Set k = k + 1;
  OD;
  RETURN(x_i^L = L  (x_i^U = U));
END interval-based continuous variable bound update;
```

Table 1: Interval-based bound update procedure for continuous variables. $\mathcal{F}$, the feasible region for the bound update problem at the current node, and is defined as $(\boldsymbol{x}, \boldsymbol{y}) : f(\boldsymbol{x}) + \boldsymbol{x}^T A_f \boldsymbol{y} + c_f^T \boldsymbol{y} \leq \overline{f}^*; g_i(\boldsymbol{x}) + \boldsymbol{x}^T A_{g,i} \boldsymbol{y} + c_{g,i}^T \boldsymbol{y} \leq \; 0, \; i = 1, \ldots, m; h_i(\boldsymbol{x}) + \boldsymbol{x}^T A_{h,i} \boldsymbol{y} + c_{h,i}^T \boldsymbol{y} = 0, \; i = 1, \ldots, p; (bx, \boldsymbol{y}) \in [\boldsymbol{x}^L, \boldsymbol{x}^U] \times [\boldsymbol{y}^L, \boldsymbol{y}^U]\}$.

Run	Number of updated Ts	Update Frequency	Iterations	CPU sec	Deepest level	Binary branches
1a*	8	1	1000	2091	60	—
1b	8	1	655	644	38	317
1c	8	1	447	844	23	135
1d	8	1	445	812	28	158
2b	8	1	550	491	34	198
2c	8	1	302	538	19	103
2d	8	1	436	596	23	133
3b	8	1	441	521	19	72
3c	8	1	287	644	28	93

3d	8	1	336	598	23	118
4b	8	1	62	1370	10	55
4c	8	1	81	1428	15	53
4d	8	1	81	1847	11	55
5b	8	2	83	868	13	66
5b	8	3	100	545	14	70
5b	8	4	113	663	16	90
5c	8	2	105	1245	16	55
5c	8	3	122	924	18	57
5c	8	4	225	1177	15	81
5d	8	2	65	623	12	42
5d	8	3	147	1044	13	75
5d	8	4	162	965	18	92
6b	6	3	148	532	16	81
6b	**4**	**3**	**180**	**315**	**20**	**88**
6b	2	3	443	376	21	208
6c	6	3	212	848	19	89
6c	4	3	263	626	16	89
6c	2	3	436	766	22	108
6d	6	3	222	802	17	109
6d	4	3	332	806	22	119
6d	2	3	618	715	29	123

Table 6: Heat exchanger network synthesis results for different branching and variable bound update strategies.

* Run 1a has not converged after 1000 iterations.

	SMIN-αBB		GMIN-αBB	
Example	Iter.	CPU s.	Iter	CPU s.
1	4	0.4	root	
2	9	0.5	2	0.1
3	—	—		
3*	root	0.3	4	0.1
4	—	—	18	4.7
4*	8	1.3	10	3
5	—	—	4	0.9
5*	2	0.5	6	2.2
6	—	—	root	0.7
6*	4	0.5	root	0.7

Table 2: Comparative results for small literature problems. * denotes the reformulated problem.

Stream	T_{in} (K)	T_{out} (K)	Fcp (kW/K)
Hot 1	650	370	10.0
Hot 2	590	370	20.0
Cold 1	410	650	15.0
Cold 2	350	500	13.0
Steam	680	680	—
Water	300	320	—

Table 3: Stream data for heat exchanger network problem of Section 4.2.1.

Label	Strategy							
	Continuous variables				Binary variables			
	Number	Method	Obj. fun. cut	Freq.	Number	Method	Obj. fun. cut	Freq.
1	8	Opt	No	1	0	–	–	–
2	8	Opt	No	1	12	Int	No	1
3	8	Opt	No	1	12	Opt	No	1
4	8	Opt	Yes	1	12	Int	No	1
5	8	Opt	Yes	2/3/4	12	Int	No	1
6	2/4/6	Opt	Yes	3	12	Int	No	1

Table 4: Labeling for variable bound updates options for the first heat exchanger network synthesis problem

Label	Strategy
a	Continuous variables (temperatures) only
b	Sequential branching
c	Hybrid branching with $zdist = 0.1$
d	Hybrid branching with $zdist = 0.2$

Table 5: Labeling for branching strategies in the first heat exchanger network synthesis problem.

Stream	T_{in} (K)	T_{out} (K)	Fcp (kW/K)
H1	443	333	30
H2	423	303	15
C1	293	408	20
C2	353	413	40
Steam	450	450	—
Water	293	313	—

Table 7: Stream data for Example 1 of Section 4.2.2.

Branching strategy	Bound update strategy	Number of iterations	CPU time (seconds)
Sequential and variable measure	No bound updates	465	250.6
Continuous only with variable measure	**No bound updates**	**69**	**82.8**
Continuous only with term measure	No bound updates	77	98.2
Continuous only with variable measure	First iteration	35	109.3
Continuous only with variable measure	Every iteration	21	800.5

Table 8: Results for Example 1 of Section 4.2.2.

Stream	T_{in} (^oC)	T_{out} (^oC)	Fcp (kW/^oC)	h (kW/(m^2 ^oC))
H1	159	77	2.285	0.10
H2	267	80	0.204	0.04
H3	343	90	0.538	0.50
C1	26	127	0.933	0.01
C2	118	265	1.961	0.50
Steam	300	300	—	0.05
Water	20	60	—	0.20

Table 9: Stream data for Example 2 of Section 4.2.2.

Branching strategy	Bound update strategy	Number of iterations	CPU time (seconds)
Sequential and variable measure	No bound updates	1699	2191.2
Continuous only with variable measure	**No bound updates**	**65**	**577.6**
Continuous only with term measure	No bound updates	1034	14818.3
Continuous only with variable measure	First iteration	39	893.5
Continuous only with variable measure	Every iteration	31	1110.8

Table 10: Results for Example 2 of Section 4.2.2.

```
PROCEDURE interval-based integer variable bound update()
  Set initial bounds L = y_i^L and U = y_i^U;
  Set iteration counter k = 0 and maximum iteration number K;
  DO k < K and L ≠ U
    Compute integer 'midpoint' M = ⌊(U + L)/2⌋;
    Set left region {(x, y) : (x, y) ∈ F, y_i ∈ [L, M]};
    Set right region {(x, y) : (x, y) ∈ F, y_i ∈ [M + 1, U]};
    Test interval feasibility of left (right) region;
    IF feasible, set U = M  (L = M);
    ELSE, test interval feasibility of right (left) region;
      IF feasible, set L = M  (U = M);
      ELSE,
        IF k = 0, RETURN(infeasible node);
        ELSE, set L = U  (U = L) and U = y_i^U  (L = y_i^L);
    Set k = k + 1;
  OD;
  RETURN(y_i^L = L  (y_i^U = U));
END interval-based bound update;
```

Table 11: Interval-based integer variable bound update procedure. $\mathcal{F}$ is the feasible region defined by $\{(\boldsymbol{x},\boldsymbol{y}) : f(\boldsymbol{x},\boldsymbol{y}) \leq \overline{f}^*; \boldsymbol{g}(\boldsymbol{x},\boldsymbol{y}) \leq 0; \boldsymbol{h}(\boldsymbol{x},\boldsymbol{y}) = 0; (\boldsymbol{x},\boldsymbol{y}) \in [\boldsymbol{x}^L, \boldsymbol{x}^U] \times [\boldsymbol{y}^L, \boldsymbol{y}^U]\}$.

	Pump 1	Pump 2	Pump 3
Fixed cost (FIM)	38,900	15,300	20,100
C_i (FIM)	6,329.03	2,489.31	3,270.27
$C_{i'}$ (FIM/kW)	1,800	1,800	1,800
α_i	19.9	1.21	6.52
β_i	0.161	0.0644	0.102
γ_i	-0.000561	-0.000564	-0.000232
a_i	629.0	215.0	361.0
b_i	0.696	2.95	0.530
c_i	-0.0116	-0.115	-0.00946
P_i^{max} (kW)	80	25	35
$V_{tot} = 350$ m^3/h	$\Delta P_{tot} = 400$ kPa	$\omega_{max} = 2950$ rpm	

Table 12: Data for the pump network synthesis problem.

Run number	Maximum αBB iterations	Branching	Bound upd.	GMIN-αBB nodes	CPU seconds
1	2	Most frac.	No	625	2153
2	10	Most frac.	No	503	3029
3	20	Most frac.	No	453	4650
4	100	Most frac.	No	275	11280
5	2	Priority	No	621	1835
6	10	Priority	No	455	2965
7	20	Priority	No	421	3702
8	100	Priority	No	342	9646
9	2	Priority	Yes	143	670
10	10	Priority	Yes	121	781
11	20	Priority	Yes	129	738
12	100	Priority	Yes	101	2537

Table 13: Results for the pump network synthesis problem varying the maximum number of αBB iterations, the branching and bound update strategies

		GMIN-αBB		αBB iterations		
m	r	nodes	CPU seconds	Average	Maximum	Total
1	0.05	135	354	2.6	14	356
1	0.1	135	351	4.3	14	584
1	0.3	135	400	8.6	395	1163
3	**0.1**	**115**	**324**	**3.8**	**12**	**435**
3	0.3	121	376	8.6	290	1037
7	0.1	127	428	11.9	343	1514
7	0.3	109	383	17.7	268	1929
15	0.1	121	417	22.8	63	2763
15	0.3	111	424	25.4	283	2821

Table 14: Results for the pump network synthesis problem using automatic stopping criteria for the αBB runs, prioritized variable branching and optimization-based bound updates.

	Objective function	y	m	r	Integrality gap
Problem 1	19.6	$\begin{pmatrix}1\\1\\1\\0\end{pmatrix}$	$\begin{pmatrix}1\\3\\2\\0\end{pmatrix}$	$\begin{pmatrix}1&0&1&0\\2&0&0&0\\0&5&3&0\\2&0&1&0\end{pmatrix}$	0.50
Problem 2	8.6	$\begin{pmatrix}1\\0\\0\\0\end{pmatrix}$	$\begin{pmatrix}11\\0\\0\\0\end{pmatrix}$	$\begin{pmatrix}1&0&0&0\\1&0&0&0\\2&0&0&0\\1&0&0&0\end{pmatrix}$	0.50
Problem 3	10.3	$\begin{pmatrix}1\\0\\0\\0\\0\end{pmatrix}$	$\begin{pmatrix}15\\0\\0\\0\\0\end{pmatrix}$	$\begin{pmatrix}1&0&0&0&0\\1&0&0&0&0\\1&0&0&0&0\\1&0&0&0&0\\1&0&0&0&0\end{pmatrix}$	0.20
Problem 4	15.3	$\begin{pmatrix}1\\1\\0\\0\\0\\0\end{pmatrix}$	$\begin{pmatrix}8\\7\\0\\0\\0\\0\end{pmatrix}$	$\begin{pmatrix}1&0&0&0&0&0\\2&0&0&0&0&0\\0&2&0&0&0&0\\0&1&0&0&0&0\\0&2&0&0&0&0\\1&0&0&0&0&0\end{pmatrix}$	0.08

Table 15: Optimal solutions for four instances of the trim loss problem.

	Strategy	GMIN-αBB nodes		CPU seconds	
		Mean	Std. dev.	Mean	Std. dev.
	a	545	289	239	81
Problem 1	b	89	22	62	14
	c	95	30	60	18
	a	949	207	314	94
Problem 2	b	85	38	48	16
	c	105	37	55	19
	a	171	79	226	35
Problem 3	b	27	6	25	4
	c	37	6	28	4
	a	343	165	570	143
Problem 4	b	22	8	33	8
	c	44	11	54	12

Table 16: Results for the trim loss minimization problems for three strategies: (a) prioritized branching, no bound updates, $iter_{max} = 100$ (b) prioritized branching, optimization-based bound updates, $iter_{max} = 100$ (c) prioritized branching, optimization-based bound updates and automatic αBB run termination.

	SMIN-αBB algorithm		GMIN-αBB algorithm			
Problem	Original	Reformulation	Original		Reformulation	
hline	CPU sec	CPU sec	Nodes	CPU sec	Nodes	CPU sec
HEN-Chen	315	—	—	617	—	—
HEN-AM1	83	—	—	1699	—	—
HEN-AM2	578	—	—	18120	—	—
Pump network	—	6304	147	927	147	1031
Trimloss 1	—	256	95	60	6307	6383
Trimloss 2	—	22840	85	48	217	302
Trimloss 3	—	1027	27	25	2929	10269
Trimloss 4	—	fail*	22	33	1275	427

Table 17: Comparative results for different formulations and algorithms. * The run did not terminate due to the failure of the MILP solver used for the lower bounding problem.