Available online at www.sciencedirect.com
SCIENCE DIRECT
Computers and Chemical Engineering 29 (2005) 1914–1933
Computers & Chemical Engineering
www.elsevier.com/locate/compchemeng

Logic-based outer approximation for globally optimal synthesis of process networks

Maria Lorena Bergamini [a], Pio Aguirre [a], Ignacio Grossmann [b,*]

[a] *INGAR, Instituto de Desarrollo y Diseño, Avellaneda 3657, S3002GJC Santa Fe, Argentina*
[b] *Department of Chemical Engineering, Carnegie Mellon University, Pittsburgh, PA 15213, USA*

Received 2 October 2004; received in revised form 23 March 2005; accepted 5 April 2005
Available online 1 June 2005

Abstract

Process network problems can be formulated as generalized disjunctive programs where a logic-based representation is used to deal with the discrete and continuous decisions. A new deterministic algorithm for the global optimization of process networks is presented in this work. The proposed algorithm, which does not rely on spatial branch-and-bound, is based on the logic-based outer approximation that exploits the special structure of flowsheet synthesis models. The method is capable of considering non-convexities, while guaranteeing globality in the solution of an optimal synthesis of process network problem. This is accomplished by solving iteratively reduced NLP subproblems to global optimality and MILP master problems, which are valid outer approximations of the original problem. Piecewise linear under and overestimators for bilinear and concave terms have been constructed with the property of having zero gap in a finite set of points. The global optimization of the reduced NLP may be performed either with a suitable global solver or using the inner optimization strategy that is proposed in this work. Theoretical properties are discussed as well as several alternatives for implementing the proposed algorithm. Several examples were successfully solved with this algorithm. Results show that only few iterations are required to solve them to global optimality.

PACS: 02.60.Pn

Keywords: Global optimization; Disjunctive programming; Process synthesis

1. Introduction

The synthesis of process networks can be formulated as generalized disjunctive programming (GDP) problems. GDP is an alternative to mixed integer non-linear programming (MINLP) for modeling problems where both continuous and discrete decisions are involved. GDP allows the combination of algebraic and logic equations to represent a synthesis problem in a more natural way.

GDP problems can be solved as MINLP problems by replacing each disjunction with its big-M or its convex hull reformulation (Lee & Grossmann, 2003). Major methods for MINLP problems include branch-and-cut, which is a generalization of the linear case (Stubbs & Mehrotra, 1999) generalized benders decomposition (GBD) (Geoffrion, 1972), outer approximation (OA) (Duran & Grossmann, 1986; Fletcher & Leyffer, 1994) and extended cutting plane (ECP) method (Westerlund & Petterson, 1995). GBD and OA are iterative methods that solve a sequence of alternate NLP subproblems with all the discrete variables fixed, and MILP master problems that perform the optimization in the discrete space. The ECP method relies on successive linearizations to build MILP approximation problems.

There are also specific algorithms that exploit the disjunctive structure of the model. In the solution method by Hooker and Osorio (1999) for linear problems, a search tree is created by branching on the logic expressions. A continuous relaxation of the problem is solved at each node of the tree.

* Corresponding author. Tel.: +1 41 22 68 3642; fax: +1 41 22 68 7139.
E-mail address: grossmann@cmu.edu (I. Grossmann).

0098-1354/$ – see front matter © 2005 Elsevier Ltd. All rights reserved.
doi:10.1016/j.compchemeng.2005.04.003

Lee and Grossmann (2003) presented an optimization algorithm for solving general non-linear GDP problems. This algorithm consists of a branch-and-bound search that branches on terms of the disjunctions and considers the convex hull relaxation of the remaining disjunctions. Türkay and Grossmann (1996) proposed a logic-based outer approximation algorithm that solves non-linear GDP problems for process networks involving two terms in the disjunction. Since the NLP subproblem only involves the active terms of the disjunctions, this algorithm overcomes difficulties that arise in the synthesis of process network problems, such as singularities that are due to zero flows. This algorithm has been implemented in LOGMIP, a computer code developed by Vecchietti and Grossmann (1999).

All the methods mentioned above assume convexity to guarantee convergence to a global solution. Therefore, when applied to non-convex problems, these algorithms may cut off the global optimum.

Viswanathan and Grossmann (1990) proposed a heuristic modification to the OA algorithm for MINLP in order to reduce the likelihood of cutting-off part of the feasible region. They introduced slacks in the linearization of non-convex constraints, and included them in an augmented penalty function. The search is stopped when there is no improvement in the NLP subproblems.

Rigorous global optimization methods for addressing non-convexities in NLP problems have been developed when special structures are assumed in the continuous terms (Horst & Tuy, 1996; Floudas, 2000; Quesada & Grossmann, 1995; Ryoo & Sahinidis, 1995; Visweswaran & Floudas, 1996; Zamora & Grossmann, 1999). Tawarmalani and Sahinidis (2002) have developed the Branch-And-Reduce-Optimization-Navigator (BARON), a software for general purpose global optimization that implements a spatial branch-and-bound method combined with reduction techniques for the variables bounds. For non-convex MINLP problems Adjman, Androulakis, and Floudas (2000), Kesavan and Barton (2000), Smith and Pantelides (1999), and Tawarmalani and Sahinidis (2004) have proposed global optimization algorithms based on spatial branch-and-bound search. Lee and Grossmann (2001) proposed a two-level branching scheme for solving non-convex GDP problems to global optimality and specialized the algorithm to GDP problem with bilinear equality constraints (2002).

Spatial branch-and-bound methods can be computationally expensive, since the tree may not be finite (except for ε-convergence). For the case of process networks, there is the added complication that the NLP subproblems are usually difficult and expensive to solve. Thus, there is a strong motivation for developing a decomposition algorithm for this class of problems that does not rely on spatial branch-and-bound.

An outer-approximation strategy for addressing the global optimization of non-convex MINLP problems was recently proposed by Kesavan, Allgor, Gatzke, and Barton (2004). The algorithm solves alternatively relaxed master MILP problems and primal and primal bounding NLP problems. The bounding problems are constructed replacing the non-convex function by known underestimating functions. Solution of primal problems involves the application of NLP global optimization algorithm.

In this work, we propose a new algorithm for solving non-convex GDP problems that arise in process synthesis. It exploits the particular structure of this kind of model, as in the case of the Logic-Based OA algorithm by Türkay and Grossmann (1996). The proposed modifications make the algorithm capable of handling non-convexities, while guaranteeing the global optimum of the synthesis of process networks. This is accomplished by constructing a master problem that is based on valid piecewise bounding representations of the original problem and by solving the NLP subproblems to global optimality. An NLP global optimization strategy is also proposed in this work.

Theoretical properties are discussed as well as several alternatives for implementing the proposed algorithm. Several numerical examples are presented to illustrate the performance of this method.

2. Background

The GDP model for synthesis of process networks is given as follows:

$$
\begin{aligned}
&\min\ Z = \sum_j c_j + f(x)\\
&\text{s.t.}\quad g(x) \le 0\\
&\begin{bmatrix} Y_j \\ h_j(x) \le 0 \\ c_j = \gamma_j \end{bmatrix} \vee \begin{bmatrix} \neg Y_j \\ B^j x = 0 \\ c_j = 0 \end{bmatrix} \quad j \in D \qquad \text{(O-GDP)}\\
&\Omega(Y) = \text{true}\\
&x \ge 0,\quad c \ge 0,\quad Y_j \in \{\text{true, false}\}
\end{aligned}
$$

The non-linear GDP model (O-GDP) contains continuous variables x and c, and Boolean variables Y. The disjunctions D apply for the processing units. If a process unit exists (Y_j = true), the constraints h_j describing that unit are enforced, and a fixed charge γ_j is applied. Otherwise (Y_j = false), a subset of continuous variables and the fixed charges are set to zero. The matrix B^j is such that the ith row is the unit vector, $b_i^j = e_i$, the ith variables must be set to zero for Y_j = false and zero row for variables that must not be set to zero for Y_j = false. For convenience in the presentation, we consider that the units are modeled with inequalities. This is not a severe restriction, since it is always possible to relax an equality constraint into two inequality constraints. Alternatively, they may be relaxed as inequalities if prior analysis is performed to determine the sign of its Lagrange multipliers (e.g. see Bazaraa, Sherali, & Shetty, 1993).

The OA algorithm requires the solution of NLP subproblems, which are obtained by fixing the Boolean variables and

MILP master problems. The master problem is formulated by using hyperplanes that replace the non-linear functions. If the original problem is convex, these hyperplanes underestimate the objective function and overestimate the original feasible region, and therefore, the master problem provides a lower bound of the optimal solution of (O-GDP) (e.g. see Duran & Grossmann, 1986).

The NLP subproblem for fixed values $\{Y_j^k\}_{j \in D}$ that satisfy $\Omega(Y^k) = \text{true}$, is as follows:

$$
\begin{aligned}
&\min\ Z = \sum_j c_j + f(x)\\
&\text{s.t.}\quad g(x) \le 0\\
&\left.\begin{aligned} h_j(x) &\le 0\\ c_j &= \gamma_j \end{aligned}\right\} \text{for } Y_j^k = \text{true}\\
&\left.\begin{aligned} B^j x &= 0\\ c_j &= 0 \end{aligned}\right\} \text{for } Y_j^k = \text{false}\\
&x \ge 0,\quad c \ge 0
\end{aligned}
\qquad \text{(R-NLP)}
$$

This NLP may be non-convex, and therefore, it may not have a unique local optimum.

As it was mentioned before, the master MILP problem in the Logic-Based OA by Türkay and Grossmann (1996) is obtained by linearizing the non-linear terms, and applying the convex hull of the disjunctions. However, if the NLP is non-convex, this process does not provide a valid bounding relaxation of the original problem, and therefore, the OA algorithm can be trapped in a suboptimal solution. This is illustrated in the next section.

3. Motivating example

Let us consider the following simple GDP problem, to illustrate how the Logic-Based OA algorithm can fail to find the global solution.

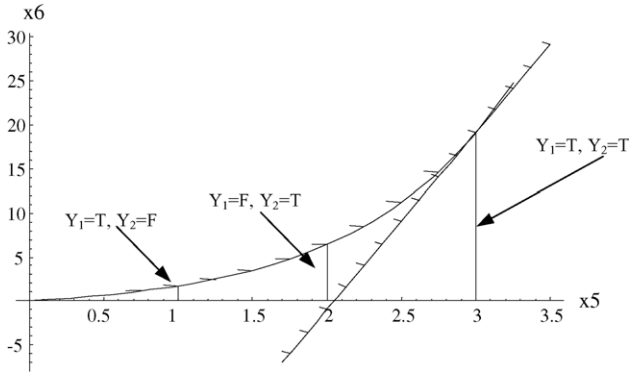

Fig. 1. Feasible region for disjunction 3 at first master.

$$
\begin{aligned}
&\min\ Z = -1.8x_6 + c_1 + c_2 + c_3\\
&\text{s.t.}\quad x_5 = x_3 + x_4\\
&\begin{bmatrix} Y_1\\ x_3 = 5x_1 - 9\\ x_1 = 2\\ c_1 = 30 \end{bmatrix} \vee \begin{bmatrix} \neg Y_1\\ x_1 = x_3 = 0\\ c_1 = 0 \end{bmatrix}\\
&\begin{bmatrix} Y_2\\ x_4 = 3x_2 - 1\\ x_2 = 1\\ c_2 = 55 \end{bmatrix} \vee \begin{bmatrix} \neg Y_2\\ x_2 = x_4 = 0\\ c_2 = 0 \end{bmatrix}\\
&\begin{bmatrix} Y_3\\ x_6 + 1 - \exp(x_5) \le 0\\ c_3 = 9 \end{bmatrix} \vee \begin{bmatrix} \neg Y_3\\ x_6 = x_5 = 0\\ c_3 = 0 \end{bmatrix}\\
&Y_1 \Rightarrow Y_3 \quad Y_2 \Rightarrow Y_3\\
&Y_1 \vee Y_2\\
&Y_j \in \{\text{true, false}\}\\
&x_6 \le 25\\
&c_j, x_i \ge 0, i = 1, \ldots, 6, j = 1, 2, 3
\end{aligned}
$$

If one were to solve this problem with the Logic-Based OA, one NLP subproblem has to be solved in order to obtain a feasible point for the linearization of the constraint in the third disjunction. Let us consider the first NLP corresponding to $Y = \{\text{true, true, true}\}$. The optimal solution of this first subproblem is $x_3 = 1, x_4 = 2, x_5 = 3, x_6 = 19.09, Z = 59.65$. The linear constraint that replaces the non-linear inequality in the third disjunction is,

$$x_6 + 41.17 - 20.08x_5 \le 0$$

With this inequality, the master problem is now infeasible, since the discrete decisions that could be taken ($Y = \{\text{true, false, true}\}$ and $Y = \{\text{false, true, true}\}$) are both infeasible in the x-space (Fig. 1) and the algorithm stops. However, the global optimum occurs when units 1 and 3 are selected, with $x_5 = 1$, $x_6 = 1.72$ and $Z = 35.91$.

4. Lower bounding master problem

The proposed algorithm iterates between the subproblems (R-NLP) where all the boolean variables of the GDP are fixed, and master problem (MILP-1) that predicts new values for the boolean variables. The key point of the algorithm is the construction of master problem (MILP-1) that rigorously overestimates the original feasible region. To accomplish this a convex GDP is derived, replacing the non-convex terms in the functions g, f and h by valid convex underestimators. The underestimators are constructed over a partition of the original domain. This convex GDP is then linearized and converted into an MILP problem by formulating the convex hull of the disjunctions. In order to improve the outer approximation, the partition is refined and

supporting hyperplanes are added to the master problem. The estimation over a partition of the entire domain will require additional continuous and discrete variables.

Problems (R-NLP) must be solved to global optimality. A local lower bounding problem (MILP-2) is constructed to find rigorous lower bound to the global optimum of problem (R-NLP).

4.1. Transformation strategies

It will be assumed that the non-convex terms are univariate concave and bilinear functions. This is not a very restrictive assumption since Smith and Pantelides (1999) have shown that a suitable reformulation in terms of convex, univariate concave, bilinear and linear fractional functions can be applied to any model of process synthesis that involves algebraic functions. The convex envelopes of these types of non-convex functions are widely known (McCormick, 1976; Tawarmalani & Sahinidis, 2002; Zamora & Grossmann, 1999) and they provide the tightest relaxation for the corresponding function. Moreover, every problem with concave univariate, bilinear and linear fractional functions can be reformulated so that it involves only concave and bilinear functions. This just requires the introduction of a new variable $z_{ij} = \frac{x_i}{x_j}$. The new variable z_{ij} replaces every occurrence of the fractional term and the bilinear constraint $x_j z_{ij} = x_i$ is added to the model.

However, another alternative for certain terms that do not belong to the classes listed before is a variable transformation strategy. The idea in variable transformation is to express the constraints in a different space, such that they become convex. An example are exponential transformations applied to Geometric Programs to convexify these problems. For Generalized Geometric Programs, Pörn, Björn and Westerlund (2005) propose a single variable transformation and approximation of the inverse transformation function by piecewise linear function. Different transformation functions have been proposed by these authors for signomial functions (Björn, Lindberg, & Westerlund, 2003). These transformations will not be explored in this paper.

In the next subsection, special piecewise estimators are derived for concave univariate and bilinear functions.

4.2. Under and overestimators for non-convex terms constructed on partitions of the original domain

Approximation of non-linear separable functions by piecewise-linear estimators has been addressed for linearizing a non-linear problem (Dantzig, 1963; Nemhauser & Wosley, 1999). Piecewise linear estimators are valid underestimators for concave terms and valid overestimators for convex terms, but they lack bounding properties for non-concave and non-convex terms. In this section, valid piecewise underestimators are formulated in disjunctive form.

Let f: $R^m \rightarrow R$ be a non-convex function and let D be the domain of interest. Let $f_D^{\mathrm{u}} : R^m \rightarrow R$ be an underestimator of f with the following property:

$$\sup\{f(x) - f_D^{\mathrm{u}}(x), x \in D\} \leq C\,\delta(D)$$

where C is a non-negative constant, independent of D and δ is a measure of sets in R^m. Note that the convex envelope of bilinear and concave univariate terms exhibits this property (Floudas, 2000). This is the underlying fact that supports convergence of spatial branch-and-bound algorithms.

Consider a partition $\{D_k\}_{k\in I}$ of D ($D_k \cap D_{k'} = \emptyset$ for $k \neq k'$ and $\cup_{k\in I} D_k = D$) and let $f_{D_k}^{\mathrm{u}}$ be the underestimator of f constructed over D_k. Define the piecewise underestimator $f^{\mathrm{u}}(x) = \sum_{k\in I} f_{D_k}^{\mathrm{u}}(x)\chi_k(x)$, where χ_k denotes the characteristic function of D_k in D: $\chi_k(x) = 1$ if $x \in D_k$, 0 otherwise. Then,

$$\sup\{f(x) - f^{\mathrm{u}}(x), x \in D\} = \max_{k\in I}\{\sup\{f(x) - f_{D_k}^{\mathrm{u}}(x), x \in D_k\}\} \leq C \max_{k\in I}\{\delta(D_k)\}$$

Thus, it is possible to tighten this underestimator as much as it may be required by considering an appropriate partition.

Given a partition $\{D_k\}_{k\in I}$ of D, the estimator f^{u} is mathematically formulated through the following disjunction:

$$\bigvee_{k\in I}\begin{bmatrix} w_k \\ f^{\mathrm{u}}(x) = f_{D_k}^{\mathrm{u}}(x) \\ x \in D_k \end{bmatrix}$$

where w_k is a Boolean variable for activating/deactivating the kth term of the disjunction.

It is interesting to note that when f is bilinear or concave univariate and the underestimator $f_{D_k}^{\mathrm{u}}$ is a convex envelope on D_k, the projection of the convex hull formulation of this disjunction onto the (x, f)-space (let us denote it by $P_{x,f}$) recovers the convex envelope f^{ce} in D. To show this, let us note first that $P_{x,f}$ is a convex set and belongs to the hypograph of f, and therefore, $f_P(x) = \min\{y{:}(x,y)\in P_{x,f}\}$ is a convex function satisfying $f_P(x) \leq f(x)$. Then, $f_P(x) \leq f^{\mathrm{ce}}(x)$.

Conversely, $P_{x,f}$ contains the sets $\phi_k = \{(x, f_{D_k}^{\mathrm{u}}(x)), x \in D_k\}$, for all $k \in I$ (ϕ_k is the projection of the facet defined by $w_k = 1$). Actually, $P_{x,f}$ is the convex hull of the union $\cup_{k\in I}\phi_k$. Then, since $f_{D_k}^{\mathrm{u}}$ is the convex envelope of f on D_k, ϕ_k is contained in the epigraph of f^{ce}, and also $P_{x,f}$ is in it. Then, $f_P(x) \geq f^{\mathrm{ce}}(x)$.

In the remaining part of this section, the specific piecewise underestimators are obtained.

4.2.1. Univariate concave terms

The convex envelope of a univariate concave function over an interval $I = [x^{\mathrm{lo}}, x^{\mathrm{up}}]$ is the linear function matching the original one at the extreme points of the interval. The underestimator constructed on a partition $\{I_k\}_{k=1,\ldots,K}$ of I ($I_k = [x^k, x^{k+1}]$) is piecewise linear and matches the function in $K+1$ points $\{x^k\}_{k=1,\ldots,K+1}$. The mathematical formulation in terms of mixed-integer linear constraints is (see Appendix

A for derivation):

$$x = \sum_{k=1}^{K} \lambda_k x^k + (w_k - \lambda_k) x^{k+1}$$
$$f^{\mathrm{u}} = \sum_{k=1}^{K} \lambda_k f(x^k) + (w_k - \lambda_k) f(x^{k+1})$$
$$0 \le \lambda_k \le w_k$$
$$\sum_{k=1}^{K} w_k = 1$$
$$w_k \in \{0, 1\}$$

4.2.2. Bilinear terms

The convex envelope of bilinear terms on a rectangular domain D is given in McCormick (1976). It estimates a bilinear function with zero gap in the boundary of D and the maximum approximation gap depends linearly on the area of D.

Let us consider the bilinear term $f(x,y) = xy$, defined in the domain $D = [x^{\mathrm{lo}}, x^{\mathrm{up}}] \times [y^{\mathrm{lo}}, y^{\mathrm{up}}]$, and consider the $K+1$ points $x^{\mathrm{lo}} = x^1, x^2, \ldots, x^{K+1} = x^{\mathrm{up}}$. In Appendix B, the derivation of the piecewise convex underestimator of f over the partition $\{D_k\}_{k=1,\ldots,K}$, $D_k = [x^k, x^{k+1}] \times [y^{\mathrm{lo}}, y^{\mathrm{up}}]$ is presented. The following formulation is obtained,

$$x = \upsilon^1 + \upsilon^2 + \cdots + \upsilon^K$$
$$y = \gamma^1 + \gamma^2 + \cdots + \gamma^K$$
$$f^{\mathrm{u}} = \sum_{k=1}^{K} \max\{\upsilon^k y^{\mathrm{lo}} + x^k \gamma^k - x^k y^{\mathrm{lo}} w^k, \upsilon^k y^{\mathrm{up}} + x^{k+1} \gamma^k - x^{k+1} y^{\mathrm{up}} w^k\}$$
$$x^k w^k \le \upsilon^k \le x^{k+1} w^k$$
$$y^{\mathrm{lo}} w^k \le \gamma^k \le y^{\mathrm{up}} w^k \quad k = 1, \ldots, K$$
$$\sum_{k=1}^{K} w^k = 1$$
$$w^k \in \{0, 1\}$$

Note that $f^{\mathrm{u}} = xy$ when $x = x^k$ for some $k = 1, \ldots, K+1$ or when $y = y^{\mathrm{lo}}$ or $y = y^{\mathrm{up}}$.

This formulation provides an underestimation for the bilinear term xy. Overestimation is required for bilinear terms appearing with negative coefficient, that is, $-xy$. In such a case, the previous formulation is applied to the bilinear term zy, where $z = -x$.

Also note that the partition is performed in one unique dimension. Partition in both variables is possible, but the formulation requires many more binary and continuous variables.

4.3. Bounding problem

Assume that the functions f, g and h in (O-GDP), after a possible variable transformation, are expressed as follows:

$$f(x) = f^{\mathrm{o}}(x) + \sum_{i \in F} f_i^{\mathrm{nc}}(x)$$
$$g(x) = g^{\mathrm{o}}(x) + \sum_{i \in G} g_i^{\mathrm{nc}}(x)$$
$$h_j(x) = h_j^{\mathrm{o}}(x) + \sum_{i \in H_j} h_{ji}^{\mathrm{nc}}(x)$$

where $f^{\mathrm{o}}, h^{\mathrm{o}}, g^{\mathrm{o}}$ are convex terms and $f_i^{\mathrm{nc}}, h_i^{\mathrm{nc}}, g_{ji}^{\mathrm{nc}}$ are the non-convex terms (concave univariate or bilinear terms) of the corresponding function. Given a gridpoint set K, the hybrid convex bounding GDP problem is as follows:

$$\min Z^L = \sum_j c_j + \alpha$$
$$\text{s.t.} \quad \alpha \ge f^{\mathrm{o}}(x) + \sum_{i \in F} z_i^f$$
$$g^{\mathrm{o}}(x) + \sum_{i \in G} z_i^g \le 0$$
$$f_{i,K}^{\mathrm{u}}(x, w, t) \le z_i^f \quad i \in F$$
$$g_{i,K}^{\mathrm{u}}(x, w, t) \le z_i^g \quad i \in G$$
$$\begin{bmatrix} Y_j \\ h_j^{\mathrm{o}}(x) + \sum_{i \in H} z_{ji}^h \le 0 \\ h_{ji,K}^{\mathrm{u}}(x, w, t) \le z_{ji}^h \quad i \in H_j \\ c_j = \gamma_j \end{bmatrix} \vee \begin{bmatrix} \neg Y_j \\ B^j \begin{pmatrix} x \\ w \\ t \end{pmatrix} = 0 \\ c_j = 0 \end{bmatrix} \quad j \in D$$
$$\Omega(Y) = \text{true}$$
$$\alpha \in R,\ x \ge 0,\ c \ge 0,\ Y \in \{\text{true, false}\}^m$$
$$z_i^f, z_i^g, z_{ji}^h \in R,\ w \in \{0, 1\}^{k \times s},\ t \in R^{p \times q} \qquad \text{(C-GDP)}$$

New variables z_i^f, z_i^g and z_{ij}^h are added, representing the non-convex terms in f, g and h_j, respectively. $f_{i,K}^{\mathrm{u}}$, $g_{i,K}^{\mathrm{u}}$ and $h_{ji,K}^{\mathrm{u}}$ are piecewise underestimators of the non-convex terms. They are expressed in terms of the original variables x, the new 0–1 variables w and the continuous variables t that are needed for defining the approximation in the grid. The subindex K means that these estimators are constructed using the gridpoint set K. The problem (C-GDP) is a relaxation of (O-GDP), and therefore, the optimal solution of (C-GDP) is a lower bound to the solution of (O-GDP).

The following theorem is important to validate the algorithm:

Theorem. *If the optimal solution of (C-GDP) belongs to the set of grid points, this corresponds to the global solution of (O-GDP).*

Proof. Let us denote (x^*, w^*, t^*, Y^*) the optimal point in (C-GDP) and assume x^* is a grid point. Thus, the piecewise underestimators have zero gap in x^*, that is, $f_{i,K}^{\mathrm{u}}(x^*, w^*, t^*) = f_i(x^*)$ for $i \in F$, $g_{i,K}^{\mathrm{u}}(x^*, w^*, t^*) = g_i(x^*)$ $i \in G$ and $h_{ji,K}^{\mathrm{u}}(x^*, w^*, t^*) = h_{ji}(x^*)$ for $i \in H_j$ and $Y_j^* = \text{true}$.

Moreover, $B^j(x^*, w^*, t^*)^T = 0$ for $Y_j^* =$ false. Therefore, (x^*, Y^*) is feasible in (O-GDP). Since x^* is an optimal point, the first and third global constraints in (C-GDP) are active, and $\alpha = f^o(x^*) + \sum_{i \in F} f_{i,K}^u(x^*, w^*, t^*)$. Thus,

$$Z^{L^*} = \sum_j c_j + \alpha = \sum_j c_j + f^o(x^*) + \sum_{i \in F} f_{i,K}^u(x^*, w^*, t^*)$$
$$= \sum_j c_j + f^o(x^*) + \sum_{i \in F} f_i(x^*) = \sum_j c_j + f(x^*) = Z^*$$

This proves that the optimal objective value of (C-GDP) is equal to the objective value in a feasible point in (O-GDP). Since the (C-GDP) problem is a relaxation of the (O-GDP), Z^* is the best value for the objective in (O-GDP).

It should be noted, however, that if the global optimum of (O-GDP) is a grid point of (C-GDP), this point might not be the optimum of (C-GDP), due to the underestimation gap.

The disjunctive problem (C-GDP) is then linearized using supporting hyperplanes derived at solution points, similarly as in the OA algorithm, and converted into an MILP problem, by formulating the convex hull representation of the disjunctions and replacing the boolean variables with binary variables y. The resulting MILP has binary variables of two different types: the variables w, introduced in the piecewise underestimators and the variables y denoting the existence of units. Let us denote this problem (MILP-1).

Assume that L subproblems (R-NLP) have been solved, with solution points $\{x^l, l = 1, \ldots, L\}$. The convex part of the objective function and the global constraints are linearized in such L points. The convex part of the constraints in disjunction j is linearized in the subset of points $\{x^l, l \in L^j\}$, where L^j is the set of iterations with $Y_j =$ true. Specifically, the problem (MILP-1) is constructed as follows:

$$\min Z^L = \sum_j c_j + \alpha$$
$$\text{s.t.} \quad \alpha \geq f^o(x^l) + \nabla f^o(x^l)(x - x^l) + \sum_{i \in F} z_i^f$$
$$g^o(x^l) + \nabla g^o(x^l)(x - x^l) + \sum_{i \in G} z_i^g \leq 0, \quad l = 1, \ldots, L$$
$$f_{i,K}^u(x, w, t) \leq z_i^f, \quad i \in F$$
$$g_{i,K}^u(x, w, t) \leq z_i^g, \quad i \in G$$
$$\begin{bmatrix} Y_j \\ h_j^o(x^l) + \nabla h_j^o(x^l)(x - x^l) \sum_{i \in H} z_{ji}^h \leq 0 \quad l \in L^j \\ h_{ji,K}^u(x, w, t) \leq z_{ji}^h \quad i \in H_j \\ c_j = \gamma_j \end{bmatrix} \vee \begin{bmatrix} \neg Y_j \\ B^j \begin{pmatrix} x \\ w \\ t \end{pmatrix} = 0 \\ c_j = 0 \end{bmatrix} \quad j \in D \qquad \text{(MILP-1)}$$
$$\Omega(Y) = \text{true}$$
$$\alpha \in R, x \geq 0, c \geq 0, Y \in \{\text{true, false}\}^m$$
$$z_i^f, z_i^g, z_{ji}^h \in R, w \in \{0, 1\}^{k \times s}, t \in R^{p \times q}$$

5. Reduced NLP

Reduced NLP (R-NLP) problems are solved iteratively with the master problem. Similarly to the Logic-Based OA, these NLPs are reduced, in the sense that fixing the Boolean variables means that a set of continuous variables (those related to non-existent units) is set to zero and removed from the NLP, as well as the constraints modeling those units. The NLPs have to be solved to global optimality. Having fixed unit configurations in the network allows us to contract the bounds, and therefore, reduce the search region.

In order to solve (R-NLP) to global optimality, the algorithm relies on the local lower bounding problem (C-MINLP). This problem is obtained from (C-GDP) by fixing the boolean variables Y_j or, in other way, by introducing the piecewise underestimators in (R-NLP). The local bounding problem is as follows:

$$\min Z = \sum_j c_j + \alpha$$
$$\text{s.t.} \quad \alpha \geq f^o(x) + \sum_{i \in F} f_{i,K}^u(x, w, t)$$
$$g^o(x) + \sum_{i \in G} g_{i,K}^u(x, w, t) \leq 0$$
$$\left.\begin{array}{l} h_j{}^o(x) + \sum_{i \in H_j} h_{ji,K}^u(x, w, t) \leq 0 \\ c_j = \gamma_j \end{array}\right\} Y_j = \text{true} \qquad \text{(C-MINLP)}$$
$$\left.\begin{array}{l} B^j \begin{pmatrix} x \\ w \\ t \end{pmatrix} = 0 \\ c_j = 0 \end{array}\right\} Y_j = \text{false}$$
$$\alpha \in R, x \geq 0, c \geq 0,$$
$$w \in \{0, 1\}^{k \times s}, t \in R^{p \times q}$$

Let us denote by (MILP-2) the MILP problem that is the linearization of the problem (C-MINLP). Note that (MILP-2) is also obtained by fixing the binary variables y in (MILP-1).

In Fig. 2, the relation between the different previously defined problems is shown. Upper bounding problems are obtained by moving to the right in the figure. Lower bounding problems appear by moving down.

Note that in some cases some simplifications are possible. For example, in bilinear programs, (C-GDP) is the same as

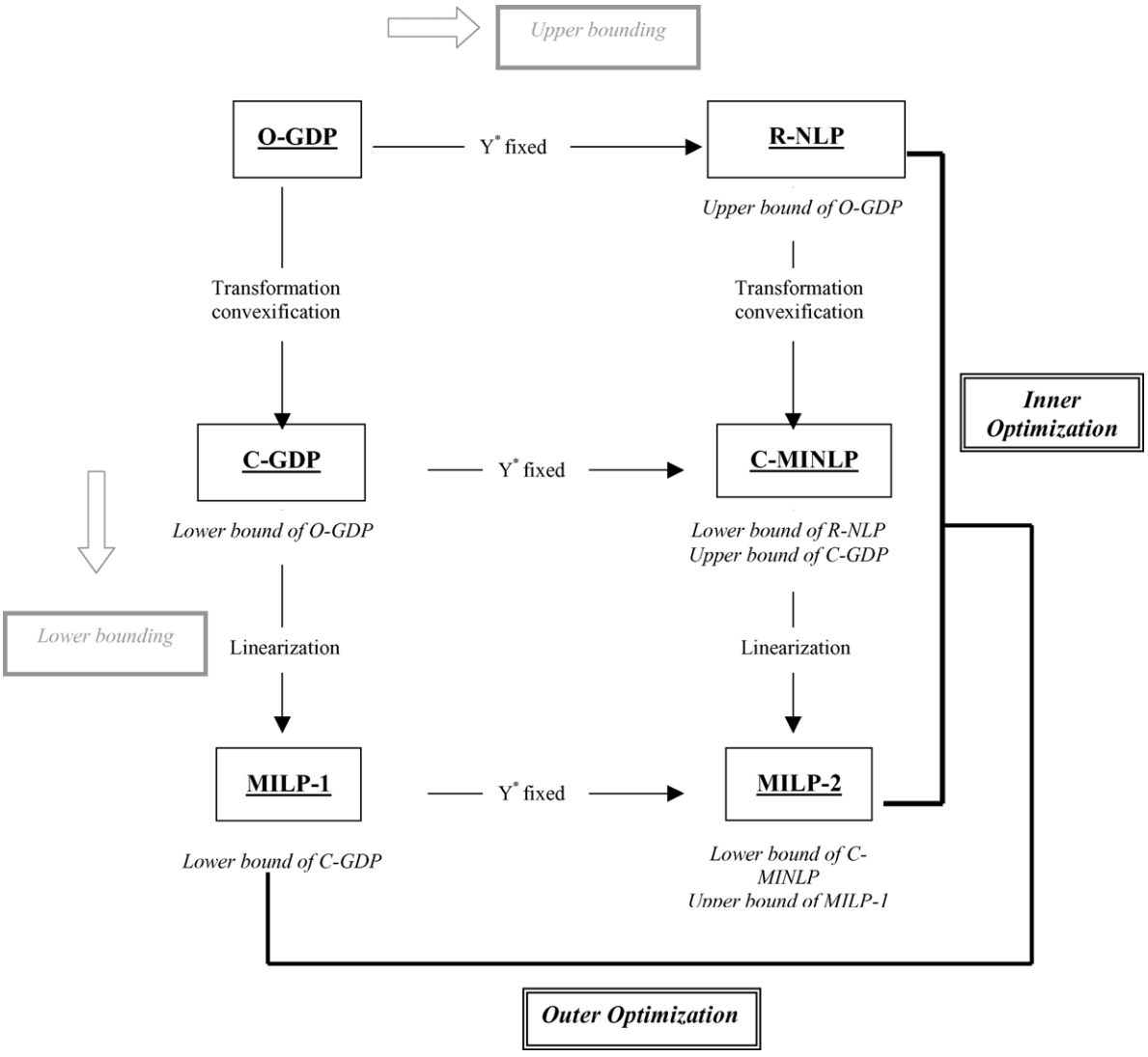

Fig. 2. Relations between the original and bounding problems.

(MILP-1) and (C-MINLP) is the same as (MILP-2), since there are no non-linear convex terms in the original problem or any possible variable transformation. Certainly, if the original problem is convex, problems (O-GDP) and (C-GDP), and problems (R-NLP) and (C-MINLP) are identical. It may also be the case that although the original problem is non-convex, a convex NLP arises by fixing the boolean variables. In such a case, (R-NLP) and (C-MINLP) are the same problem, perhaps in different variable spaces (e.g. geometric programs).

6. Algorithm

The algorithm has two main phases as can be seen in Fig. 3.

6.1. Outer optimization

This phase calculates a global lower bound (GLB) of the optimum of problem (O-GDP). The problem (MILP-1) is solved using an initial grid and initial linearization points, to predict a new structure in the network and a new global lower bound. An increasing sequence of global lower bounds is obtained in the successive iterations of this phase. This is true because (MILP-1) is modified by adding integer cuts in Y_j that avoid repeating structures and supporting hyperplanes of the convex functions.

The initial grid can be redefined when solving (MILP-1) or it can accumulate the grid points generated during the inner optimization. The cumulative option has the disadvantage of exponentially increasing the size of the model (MILP-1), making it very difficult to solve. Both alternatives are implemented in the numerical examples.

6.2. Inner optimization

A fixed structure is globally optimized. This is performed by iteratively solving the problems (R-NLP) and (MILP-2) that bound the global solution of the reduced NLP.

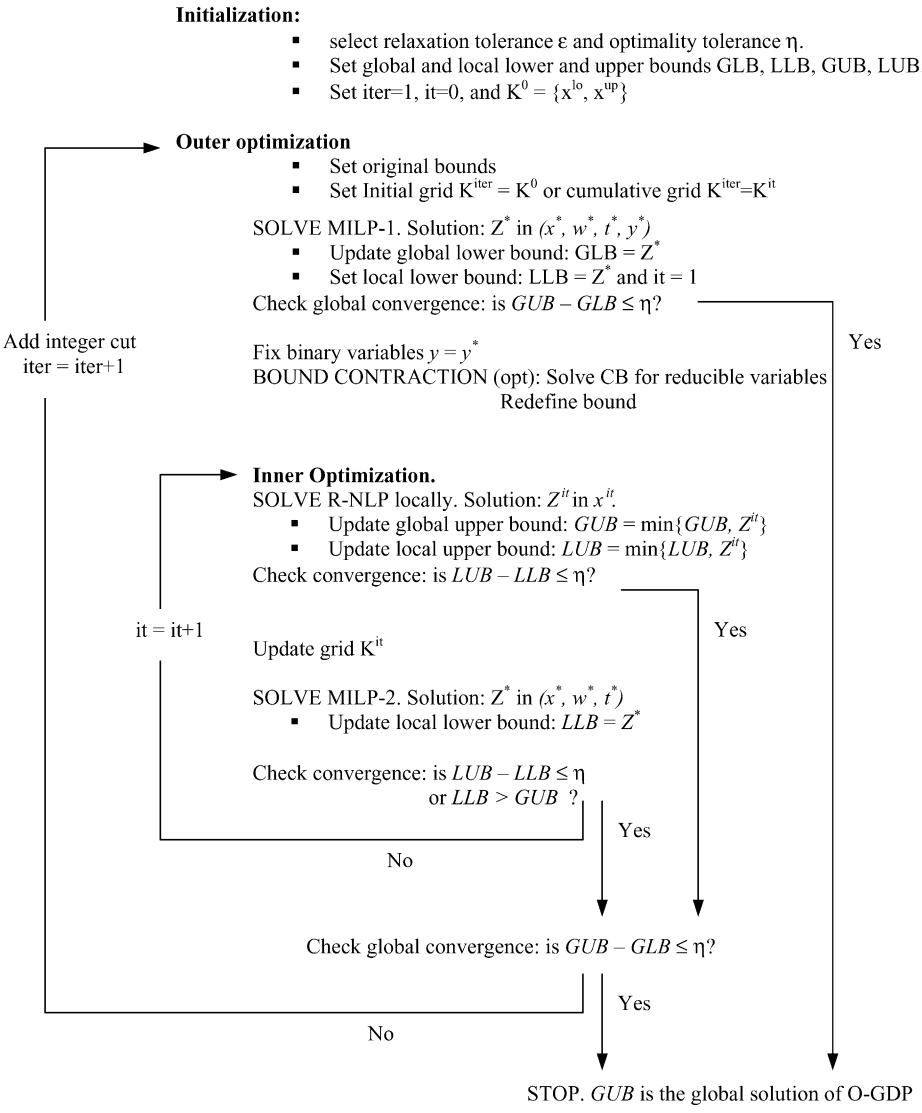

Fig. 3. Scheme of the algorithm.

Solutions of (R-NLP) provide feasible solutions of (O-GDP), and allow to update the local and global upper bound (LUB and GUB, respectively). Tighter local lower bounds (LLB) arise refining the grid and solving the local bounding problem (MILP-2), which is actually a relaxation of (R-NLP).

There may be cases where fixing the boolean variables Y, the resulting NLP problem is convex, or it is known that it has a unique optimal solution. An example of this kind of problem is the GDP model for the synthesis and design of a batch plant formulated by Lee and Grossmann (2001). In such cases, the inner optimization can be accomplished by simply solving the problem (R-NLP) with a local solver.

Alternatively, one might resort to a global NLP optimizer (e.g. BARON, Sahinidis, 1996) that will take advantage of the tighter variable bounds that arise in a fixed configuration.

6.3. Bound contraction

Since the elimination of non-optimal subregions is crucial in accelerating the search, an optional bound contraction procedure is considered in order to reduce the search space in the global optimization of the NLP subproblems. This contraction is performed before the algorithm enters in the inner optimization phase. The scheme for contraction adopted in this work is the same as the one proposed by Zamora and Grossmann (1999). Basically, the problem solved at each contraction step is the following:

$$\begin{aligned} &\min / \max\ x_i \\ &\text{s.t.}\quad Z \le \text{GUB} \\ &\text{constraints in C-MINLP} \end{aligned} \qquad \text{(CB)}$$

This problem is a convex problem whose feasible region overestimates the subregion of (R-NLP) where the objective

function can be improved. The aim of this problem is to eliminate part of the original feasible region where the global optimum does not exist.

Note that in general, (CB) is a MINLP problem, since binary variables w related to the initial grid are involved. However, if the initial grid consists of only variable bounds, and therefore, the original domain is not really subdivided, (CB) can be solved as an NLP.

The bound contraction is performed on those variables that are involved in the relaxation so that the underestimators can be tightened.

6.4. Grid update

The grid is updated for each non-convex term. The idea in refining the grid is to include in it those points obtained as optimal points in the relaxed problem.

The decision of adding a new point to the grid is based on the error between the non-convex term ζ_i^{nc} and the substituting variable z_i^{ζ} in the solution $(x^*, z_i^{\zeta *})$ of (MILP-1) or (MILP-2) where $\zeta = f$, g or h. The following criterion is adopted: If $|z_i^{\zeta *} - \zeta_i^{\mathrm{nc}}(x^*)| > \varepsilon|\zeta_i^{\mathrm{nc}}|$, then add x^* to the grid corresponding to ζ_i^{nc}, where ε is a specified tolerance.

An alternative strategy for updating the grid is to include in it the middle point of the active subinterval in the solution of the master problem. If the solution $(x^*, z_i^{\zeta *})$ of the master problem is such that $x^k \leq x^* \leq x^{k+1}$ (interval k is active) then, the grid corresponding to ζ_i^{nc} is modified by adding the point $\frac{x^k + x^{k+1}}{2}$.

6.5. Convergence

The proposed underestimators are constructed over a partition of the domain, and they involve an approximation error that depends on the size of each subdomain. Then, as the dimension of the subdomains is reduced by further partitions, the gap of approximation is also reduced.

7. Illustrative example

Let us consider again the illustrative example discussed in Section 2.

The proposed algorithm starts solving the MILP obtained by replacing the concave constraint in the third disjunction with the piecewise linear relaxation constructed over the interval defined by the bounds of x_5 and replacing the disjunctions with their convex hull reformulation. This first master problem (MILP-1) predicts the lower bound GLB = 25.19, with $Y = \{\text{true, false, true}\}$ with $x_5^* = 1$, $x_6^* = 7.67$ (see Fig. 4). The NLP subproblem corresponding to these boolean values predicts an upper bound GUB = 35.91. Since there is a gap between the lower and upper bounds, the problem (MILP-2) is solved, including x_5^* in the grid. This problem

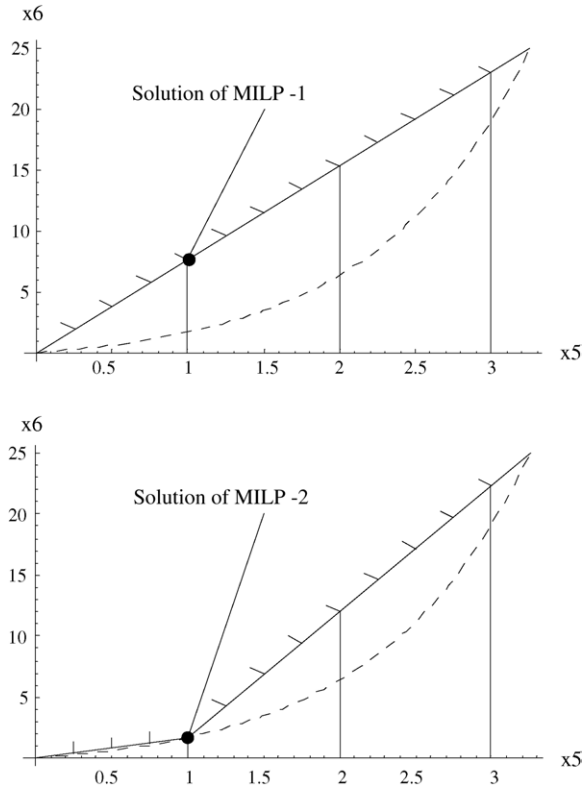

Fig. 4. Feasible region and solution for MILP-1 and MILP-2 in the first iteration in the illustrative example.

has an optimal solution $Z = 35.91$ with $x_5^* = 1$ and $x_6^* = 1.72$, which in fact is the global optimum of this configuration.

In the second outer iteration, the new global lower bound obtained is GLB = 36.38, with $Y = \{\text{false, true, true}\}$. This bound is greater than the best-known solution; therefore, the algorithm stops with the global solution $Z = 35.91$.

8. Numerical examples

The proposed algorithm was implemented in GAMS (Brooke, Kendrick, Meeraus, & Raman, 1997) and five examples were solved on a 1.8 GHz Pentium 4 PC with 256 Mb memory. GAMS/CONOPT2 and GAMS/BARON 5.0 (Sahinidis, 1996) were used with their default options to solve the reduced NLP problems and GAMS/CPLEX 8.1 for the MILP problems.

Example 1. A process network problem, which is a variation of the problem in Duran and Grossmann (1986) was solved using the proposed algorithm. The problem involves eight processes, with 25 flow streams (Fig. 5). The objective function to be minimized considers fixed costs c_j for selected units and operating costs for stream x_i, with coefficients p_i, which are shown in Fig. 5. The GDP formulation of the model

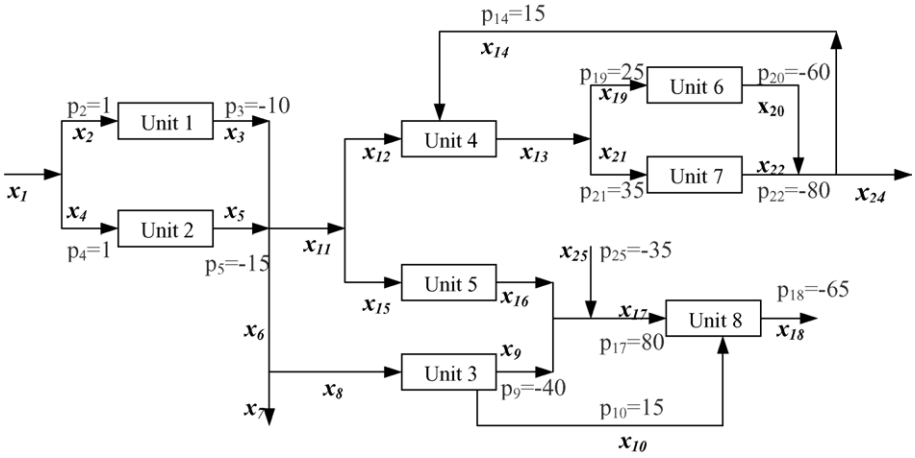

Fig. 5. Superstructure for Example 1.

is as follows:

$$\min \sum_{j=1}^{8} c_j + 122 + \sum_{i \in L} p_i x_i - p_2(x_2 - 3)^2 + p_3(x_3 - 0.7)^2 + p_{10}\sqrt{15 - 0.3(x_{10} - 4)^2} - p_{19}(x_{19} - 3)^2 + p_{21}\sqrt{10 - 0.5(x_{21} - 1.2)^2}$$

$$\text{s.t.} \quad x_1 - x_2 - x_4 = 0$$
$$x_3 - x_5 - x_6 - x_{11} = 0$$
$$x_{13} - x_{19} - x_{21} = 0 \qquad x_{10} - 0.8x_{17} \le 0$$
$$x_{17} - x_9 - x_{16} - x_{25} = 0 \qquad x_{10} - 0.4x_{17} \ge 0$$
$$x_{11} - x_{12} - x_{15} = 0 \qquad x_{12} - 5x_{14} \le 0$$
$$x_6 - x_7 - x_8 = 0 \qquad x_{12} - 2x_{14} \ge 0$$
$$x_{23} - x_{20} - x_{22} = 0$$
$$x_{23} - x_{14} - x_{24} = 0$$

$$\begin{bmatrix} Y_1 \\ e^{x_3} - 1 - x_2 \le 0 \\ c_1 = 25 \end{bmatrix} \vee \begin{bmatrix} \neg Y_1 \\ x_3 = x_2 = 0 \\ c_1 = 0 \end{bmatrix}$$

$$\begin{bmatrix} Y_2 \\ e^{x_5/1.2} - 1 - x_4 \le 0 \\ c_2 = 40 \end{bmatrix} \vee \begin{bmatrix} \neg Y_2 \\ x_5 = x_4 = 0 \\ c_2 = 0 \end{bmatrix}$$

$$\begin{bmatrix} Y_3 \\ 1.5x_9 - x_8 + x_{10} = 0 \\ c_3 = 30 \end{bmatrix} \vee \begin{bmatrix} \neg Y_3 \\ x_9 = x_8 = x_{10} = 0 \\ c_3 = 0 \end{bmatrix}$$

$$\begin{bmatrix} Y_4 \\ 1.25x_{12}x_{14} - x_{13} = 0 \\ c_4 = 50 \end{bmatrix} \vee \begin{bmatrix} \neg Y_4 \\ x_{13} = x_{12} = x_{14} = 0 \\ c_4 = 0 \end{bmatrix}$$

$$\begin{bmatrix} Y_5 \\ x_{15} - 2x_{16} \le 0 \\ c_5 = 30 \end{bmatrix} \vee \begin{bmatrix} \neg Y_5 \\ x_{15} = x_{16} = 0 \\ c_5 = 0 \end{bmatrix}$$

$$\begin{bmatrix} Y_6 \\ e^{x_{20}/1.5} - 1 - x_{19} \le 0 \\ c_6 = 35 \end{bmatrix} \vee \begin{bmatrix} \neg Y_6 \\ x_{20} = x_{19} = 0 \\ c_6 = 0 \end{bmatrix}$$

$$\begin{bmatrix} Y_7 \\ e^{x_{22}} - 1 - x_{21} \le 0 \\ c_7 = 20 \end{bmatrix} \vee \begin{bmatrix} \neg Y_7 \\ x_{22} = x_{21} = 0 \\ c_7 = 0 \end{bmatrix}$$

$$\begin{bmatrix} Y_8 \\ e^{x_{18}} - 1 - x_{10} - x_{17} \le 0 \\ c_8 = 25 \end{bmatrix} \vee \begin{bmatrix} \neg Y_8 \\ x_{18} = x_{10} = x_{17} = 0 \\ c_8 = 0 \end{bmatrix}$$

$$\neg Y_6 \vee \neg Y_7$$
$$\neg Y_4 \vee \neg Y_5$$

$$x_i,\ c_j \ge 0,\ Y_j \in \{\text{true, false}\},\ i=1, 2, \ldots, 25,\ j=1, 2, \ldots, 8$$

When the Logic-Based OA algorithm by Türkay and Grossmann (1996) is applied in this problem, using the termination criterion of no improvement in the objective of the NLP solutions, it stops in the third major iteration with a suboptimal solution $Z = 10.627$. Also, none of the master solutions is lower than the global optimum. If the termination criterion is not applied and we let the algorithm continue iterating, the global solution is found in major iteration 18. However, there is no guarantee of globality.

The problem was also formulated as an MINLP using the Big-M formulation of the disjunctions (with $M = 100$) and solved using the GAMS/DICOPT solver, which implements the AP/OA/ER algorithm (Viswanathan & Grossmann,

Table 1
Results using GAMS/DICOPT with different initial points in Example 1

Initialization for variables x	Optimal solution	Stopping criterion	CPU time (s)		Major iterations	Units
			MIP	NLP		
$x = x^{up}$	82.627	3	0.19	0.30	3	7 and 8
$x = x^{lo}$	117.627	3	0.21	0.20	3	1, 3 and 8
$x = x^{opt}$	55.627	3	0.24	0.17	3	1, 7 and 8
$x = x^{opt}$	10.627	0	0.90	0.71	10	1

1990). The solution depends strongly on the initial point. Several initial points were used, but none of the runs finds the global solution. Some results are shown in Table 1. Using the stopping criterion 3, DICOPT stops when the solutions of the NLP subproblems have no improvement, and the stopping criterion 0 forces DICOPT to continue performing a specified number of iterations (10 iteration in the results of Table 1).

The algorithm proposed in this work obtains the optimal structure (units 1, 4 and 7) in two outer iterations. The configuration obtained in the first master (MILP-1) consists of units 1, 3, 4, 7 and 8, and the lower bound is GLB = −93.53. This structure is optimized in four inner iterations. The corresponding (MILP-2) subproblems are set up adding in the grid the variable values obtained in the optimal solution of the master problem, and adding the linearizations of the convex term in the solution of the NLP subproblem. An integer cut is added in order to make this configuration infeasible in subsequent master problems. The gridpoint set is updated by simply adding the new point to the grid of the previous iteration.

The optimal structure with objective $f = 7.011$ and involving units 1, 4 and 7 is selected in the next outer iteration, and it requires one inner iteration to prove globality in the solution of the subproblem. One additional outer iteration is required to check convergence to the global optimum.

The algorithm requires less than 1 CPU second in solving the MILP subproblem and 0.5 CPU second in solving the NLP subproblems. Details of the solution steps and problem sizes can be seen in Table 2. The problem was also solved with BARON (Sahinidis, 1996), which required 0.3 CPU-second and 25 nodes in the branch-and-bound tree, yielding the same solution of $f = 7.011$.

Example 2. The next example was taken from Kocis and Grossmann (1987). It involves the selection of the optimal separation scheme to be used to separate a multicomponent process stream into a set of product streams with given purity specifications. The superstructure consists of feed and product mixers, two possible separation units and a splitter that splits the feed into streams towards the separators or towards the final mixers (Fig. 6). The alternative schemes include the use of flash separation, distillation, or the elimination of the complete separation process if it is proven to be unprofitable. The nonconvex (bilinear) GDP model for this problem is as follows:

$$\min z = -35p_a^1 - 30p_b^2 + 10f^1 + 8f^2 + f_a^4 + f_b^4 + 4f_a^5 + 4f_b^5 + cf + cd$$

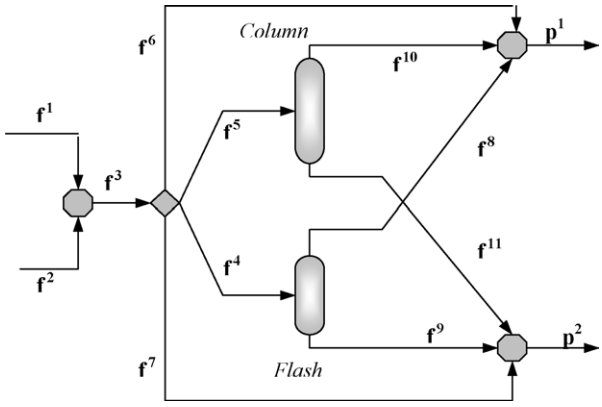

Fig. 6. Superstructure in the Example 2.

$$f_a^3 = 0.55f^1 + 0.50f^2 \qquad f_b^3 = 0.45f^1 + 0.50f^2$$
$$p_a^1 = f_a^8 + f_a^{10} + f_a^6 \qquad p_b^1 = f_b^8 + f_b^{10} + f_b^6$$
$$p_a^2 = f_a^9 + f_a^{11} + f_a^7 \qquad p_b^2 = f_b^9 + f_b^{11} + f_b^7$$
$$f_a^6 = \xi^6 f_a^3 \qquad f_b^6 = \xi^6 f_b^3$$
$$f_a^7 = \xi^7 f_a^3 \qquad f_b^7 = \xi^7 f_b^3$$
$$\xi^4 + \xi^5 + \xi^6 + \xi^7 = 1$$
$$p_a^1 \geq 4.0p_b^1 \quad p_b^2 \geq 3.0p_a^2$$
$$p_a^1 + p_b^1 \leq 15 \quad p_a^2 + p_b^2 \leq 18$$

$$\begin{bmatrix} Y_f \\ f_a^4 = \xi^4 f_a^3, \quad f_b^4 = \xi^4 f_b^3 \\ 2.5 \leq f_a^4 + f_b^4 \leq 25 \\ f_a^8 = 0.85f_a^4, \quad f_b^8 = 0.20f_b^4 \\ f_a^9 = 0.15f_a^4, \quad f_b^9 = 0.80f_b^4 \\ cf = 2 \end{bmatrix} \vee \begin{bmatrix} \neg Y_f \\ f_a^4 = f_b^4 = 0 \\ f_a^8 = f_b^8 = 0 \\ f_a^9 = f_b^9 = 0 \\ \xi^4 = 0 \\ cf = 0 \end{bmatrix}$$

$$\begin{bmatrix} Y_d \\ f_a^5 = \xi^5 f_a^3, \quad f_b^5 = \xi^5 f_b^3 \\ 2.5 \leq f_a^5 + f_b^5 \leq 25 \\ f_a^{10} = 0.975f_a^5, \quad f_b^{10} = 0.050f_b^5 \\ f_a^{11} = 0.025f_a^5, \quad f_b^{11} = 0.950f_b^5 \\ cd = 50 \end{bmatrix} \vee \begin{bmatrix} \neg Y_d \\ f_a^5 = f_b^5 = 0 \\ f_a^{10} = f_b^{10} = 0 \\ f_a^{11} = f_b^{11} = 0 \\ \xi^5 = 0 \\ cd = 0 \end{bmatrix}$$

Table 2
Solution steps and problem sizes for Example 1

Outer iteration	Inner iteration	Solution MILP-1	Solution NLP	Solution MILP-2	Binary variables	Cont vars	
						NLP	MILP
1	1	−93.530	–	–	14	21	68
			83.317	71.910	8		74
	2		83.317	79.636	10		80
	3		83.317	81.647	12		86
	4		83.317	82.937	14		92
2	1	6.261	–	–	25	14	100
			7.011	7.011	18		104
3		10.627	–	–	26	–	104

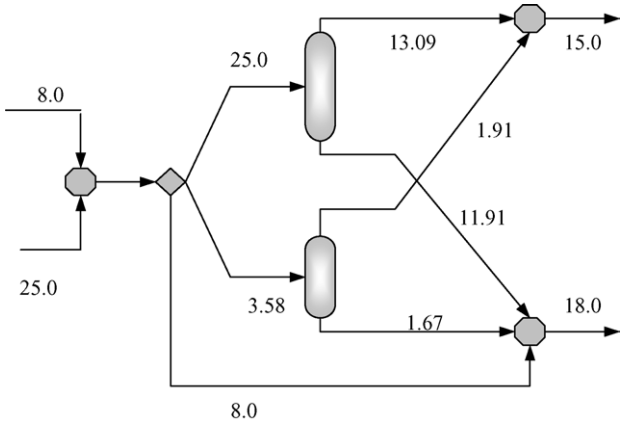

Fig. 7. Optimal solution of Example 2.

$0 \le cf, cd; \quad f^1, f^2 \le 25; \quad 0 \le \xi^4, \xi^5, \xi^6, \xi^7 \le 1;$

$Y_f, Y_d \in \{\text{true, false}\}$

The eight bilinear terms are replaced by the proposed piecewise underestimators, partitioning the domain through the split fractions ζ.

The first master problem, using the bound of ζ as initial gridpoints, predicts a lower bound GLB = −539.66, with Y_f = true, Y_d = false. No bound contraction is performed. The corresponding NLP has a solution Z = −470.13. Since there is a gap between the lower and upper bounds, the MILP-2 is solved, including the solution of the previous master in the gridpoint set. It takes four inner iterations to converge the local lower and upper bounds.

The second outer iteration solves the master with the piecewise underestimator constructed on the accumulated gridpoints. It provides a new global lower bound of GLB = −510.39 with Y_f = true, Y_d = true. The corresponding NLP subproblem has a solution of Z = −510.08. The global lower and upper bounds are within 0.5% tolerance and no inner iterations are required. The algorithm stops with the global optimal Z = −510.08, involving both column and flash separator (see Fig. 7). The total time is less than 1.5 CPU second. Table 3 shows the progress of the algorithm through the outer and inner iterations, as well as the model sizes for this example.

The solution of the first master problem provides a very weak lower bound for the correspondent NLP solution. It was noted that in the solution of that MILP problem, the streams involved in the initial splitter do not maintain the relative order of component flowrates. Kocis and Grossmann (1987) propose valid relaxations of the bilinear mass balances in the multistream splitter that overcomes this weakness.

When these relaxations are added to the master problems in the algorithm, the optimal configuration is obtained in the first master problem, providing a lower bound GLB = −515.55, with Y_f = true, Y_d = true. The global optimization of the NLP subproblem (within 0.5% tolerance) takes one inner iteration if bound contraction is performed in the variables involved in bilinear terms (ζ^4, ζ^5, ζ^6, ζ^7, f_a^3, f_b^3). Bound contraction requires solving 12 LP problems (problem (CB) is linear because the partition consists of a unique subinterval). Without bound contraction, the inner optimization takes three iterations.

The second outer iteration solves the master problem using the accumulated grid points. It provides a new global

Table 3
Solution steps and problem sizes for Example 2

Outer iteration	Inner iteration	Solution MILP-1	Solution NLP	Solution MILP-2	Binary variables	Cont vars	
						NLP	MILP
1		−539.66			6	25	71
	1		−470.13	−481.88	8		89
	2		−470.13	−476.91	12		107
	3		−470.13	−473.75	16		125
	4		−470.13	−471.44	20		167
2		−510.39		–	22		167
	1		−510.08		–	31	

lower bound GLB = −487.512, which is greater than the best feasible solution found. Then, the global optimum is the solution obtained in the first outer iteration, with objective $Z = -510.08$.

Example 3. The following GDP problem was formulated by Lee, Colberg, Siirola, & Grossmann (2002) to model a X-monomer process. The objective of the model is to find the best reaction path from the given raw materials to the final product, which minimizes the total annual cost. The superstructure proposed by the authors of the mentioned work involves a number of interconnected reaction units whose selection is modeled with disjunctions. Due to confidentiality reasons, we cannot disclose the details of this model.

The superstructure consists of two raw materials, eight intermediate chemicals, one product and two by-products. There are 14 reaction units and 3 separation units. Linear mass balances define the input and output streams in each unit. The objective function takes into account the annualized cost of raw material, utility, waste treatment, packaging (with cost coefficient RM, UT, WT and PK, respectively) labor and capital. The model is as follows:

$$\min\ Z = \sum_i \{(\mathrm{RM}_i + \mathrm{UT}_i + \mathrm{WT}_i + \mathrm{PK}_i)p_i + \mathrm{LC}_i + \Phi_i(p_i)\}$$

$$\text{s.t.} \quad Ax = b$$

$$\begin{bmatrix} Y_i \\ \mathrm{Yield}_i \times x_i^{\mathrm{IN}} = x_i^{\mathrm{OUT}} \\ p_i = x_i^{\mathrm{OUT}} \\ LC_i = \alpha_i \\ 0 \le x_i^{\mathrm{IN}}, x_i^{\mathrm{OUT}}, p_i \le \mathrm{XUB} \end{bmatrix} \vee \begin{bmatrix} \neg Y_i \\ x_i^{\mathrm{IN}} = 0 \\ x_i^{\mathrm{OUT}} = 0 \\ p_i = 0 \\ LC_i = 0 \end{bmatrix}, \forall i \in I$$

$$\Omega(Y) = \mathrm{True}$$

$$0 \le x_n, \quad x_i^{\mathrm{IN}}, \quad x_i^{\mathrm{OUT}}, \quad p_i \le \mathrm{XUB}, \quad \forall i \in I, \quad \forall n \in N$$

$$0 \le \mathrm{LC}_i, \quad \forall i \in I \qquad Y_i \in \{\mathrm{true}, \mathrm{false}\}, \quad \forall i \in I$$

I denotes the set of units and N the set of chemicals. The variables x_n represents the molar flowrate of component n, and x_i^{IN} and x_i^{OUT} are the inlet and oulet flowrates in unit i. The production of each unit is represented with p_i. It is assumed that the conversion of unit i, Yield_i, is given.

The capital cost $\Phi_i(p_i)$ is a concave function of the production rate. The master problems are set up replacing each of these terms with a variable bounded by the piecewise linear underestimator.

GAMS/DICOPT solves the Big-M reformulation of this problem providing a local solution Z = USM$ 246.342 per year, for a production of 450 Mlb per year of X-monomer and no by-product production. This solution involves seven reaction units and two separation units. DICOPT stops with worsening of the NLP solutions at the second major iteration. If we allow the solver to go on the search until a maximum of 20 major iteration, the best-found solution is Z = USM$242.760 and none of the master objective is below this value.

The global optimal reaction path involves five reaction units and two separation units (see Fig. 8). The production of X-monomer is 450 Mlb per year with a by-product production of 26.1 Mlb per year. The total annual cost is USM$ 214.711 per year.

The sequence of steps for obtaining the global solution using the proposed algorithm is shown in Table 4, as well as the progress of the lower and upper bounds. No bound contraction was performed. Four outer iterations were required to obtain a global lower bound greater than the best feasible solution. Each NLP subproblem was solved to global optimality in one inner iteration and the gridpoint sets were updated with the solution of the MILP problems. The grid was not reset in the outer iterations but it accumulated all the added points. Table 5 shows the CPU time required in each step and the size of each solved subproblem. Note that the total CPU time used is 5.441 s.

This example was also solved using GAMS/BARON in two ways. In the first one, BARON was used to solve the NLP subproblems to global optimality instead of performing the inner loop in Fig. 3. The optimal objective values obtained with this alternative are the same as shown in Table 4 for the problems (MILP-1) and NLP subproblems and the CPU time are shown in Table 6. As can be seen the CPU-time is slightly lower (5.212 s versus 5.441 s). The second way that BARON was used was to directly solve the full problem O-GBD (its Big-M reformulation). In this case BARON could not solve the problem O-GDP in less than 960 s. At that point, the search was interrupted, and the lower bound that BARON provided (109.018) was about 50% below the global optimal solution (214.711).

Example 4. This example corresponds to a synthesis problem of a distributed wastewater multicomponent network, which is taken from Example 10 of Galan and Grossmann (1998). Given a set of process liquid streams with known composition, a set of technologies for the removal of pollutants, and a set of mixers and splitters, the objective is to find the interconnections of the technologies and their flowrates to meet the specified discharge composition of pollutant at minimum total cost. Discrete choices involve deciding what equipment to use for each treatment unit. Lee and Grossmann (2001) formulated the problem as a GDP model.

The superstructure is shown in Fig. 9, involving three inlet streams, which are split into streams going into the treatment units. There are three different equipment available for removal of each of the pollutants. Each equipment has different removal ratio of the pollutants and cost function. The outlet stream of each treatment unit is again split and then a fraction of the stream is recycled, while the rest of the stream is sent to the final mixer for discharge. The data for this example is given in Lee and Grossmann (2001).

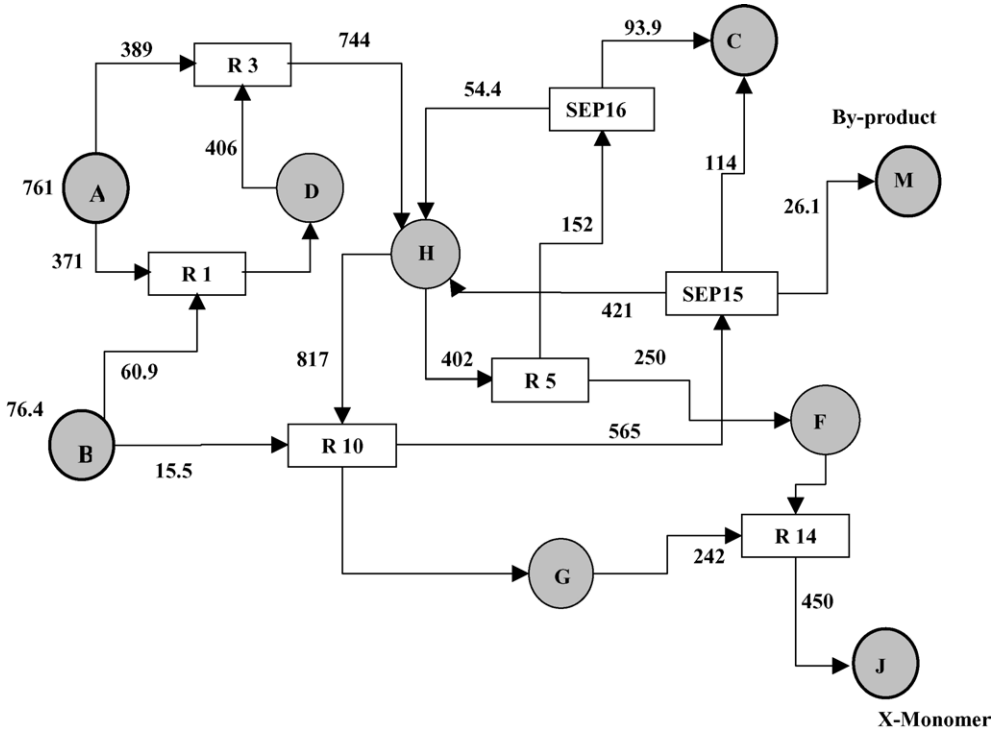

Fig. 8. Optimal solution of Example 3.

Table 4
Solution steps for Example 3

Outer iteration	Solution MILP-1	Solution NLP	Solution MILP-2	GUB	LUB	GLB	LLB
1	186.276	216.920		216.920	216.920	186.276	186.276
			216.918		216.920		216.918
2	199.702	214.711		214.711	214.711	199.702	199.702
			214.710		214.711		214.710
3	210.602	236.567		214.711	236.567	210.602	210.602
			236.567		236.567		236.567
4	215.619			214.711		215.619	

Table 5
CPU time and model sizes in the solution of Example 3

Outer iteration	Solution MILP-1		Solution NLP		Solution MILP-2	
	CPU time (s)	Disc vars/Cont vars	CPU time (s)	Cont vars	CPU time (s)	Disc vars/Cont vars
1	0.203	249/34	0.039	67	0.109	283/34
2	0.625	283/51	0.066	59	0.140	317/51
3	1.921	317/68	0.027	71	0.140	341/68
4	2.171	351/85				
Total time	4.920		0.132		0.389	

Table 6
CPU time using BARON for solving the NLP subproblems in Example 3

Outer iteration	MILP-1	NLP
1	0.203	0.060
2	0.640	0.110
3	1.968	0.060
4	2.171	
Total	4.982	0.230

The non-linearities in this model are due to the bilinearities that arise in the component mass balances in the final splitters and the concave cost functions.

This problem was solved to global optimality with our algorithm in just under 2 min. Bound reduction was performed in the complicating variables representing the total flows in the treatment units. These variables are involved in the bilinear mass balances in the final splitters. The initial grid for the outer iterations was set up with three points: the lower and upper bounds and the middle point. Within each

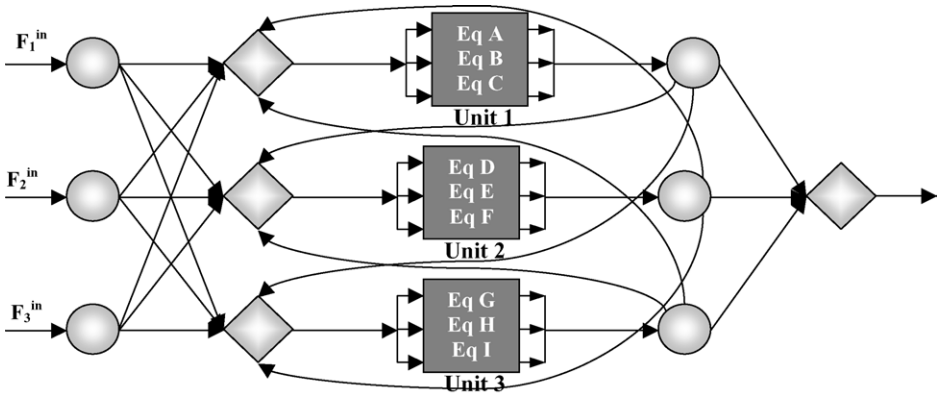

Fig. 9. Superstructure for Example 4.

inner iteration, the gridpoint sets were updated using the middle point of the active subinterval. Adding the master solution point to the grid causes slower convergence to the global solution of the reduced NLP.

The global optimum solution is shown in Fig. 10. Six outer iterations were necessary to prove globality of the solution as seen in Table 7. In the third outer iteration (MILP-1), selected the optimal equipment and obtained a lower bound within a tolerance of 0.5% requiring five iterations in the inner optimization. Table 8 shows the computing times and the problem sizes. The total time required by the algorithm was 11.31 s for solving the (MILP-1) problems, 0.54 s for solving (R-NLP) subproblems, 8 s for reducing bounds in total flows and 117.56 s in solving the (MILP-2) subproblems.

The most time consuming step in this example is the inner optimization of the optimal structure. Due to the bound contraction procedure, the reduced NLP could be solved to global optimality with the solver BARON 6.0. It rapidly detected the infeasibility of the first two NLP subproblems. In the third equipment selection, BARON found the global optimum of the NLP in 20 CPU seconds. The (MILP-1) problems in the following outer iterations detected infeasible structures. The total time required with this implementation of the method was approximately 38 CPU seconds, which is considerably lower than the 118 s with the algorithm of Fig. 3.

Example 5. The next example is a wastewater treatment network problem where the separation is performed using nondispersive solvent extraction (NDSX) (see Galan and Grossmann, 1998). For NDSX technologies, the outlet concentration depends on the inlet concentration of the pollutant and on the flowrate. However, the flowrate of the inlet stream is assumed not to change during the treatment, since the concentration of the pollutants is low. A short-cut model of the NDSX is used. The equation for the NDSX treatment is as follows:

$$\mathrm{He}^j \mathrm{cs}^j - \mathrm{Co}^j = \exp\left(-\frac{a_\mathrm{t} K_\mathrm{m} \mathrm{HeNM}}{\mathrm{FLOWT}}\right)(\mathrm{He}^j \mathrm{ce}^j - \mathrm{Co}^j)$$

where cs^j is the outlet concentration of pollutant j, ce^j the inlet concentration of j, a_t the surface area of the hollow fiber module (135 m^2), NM the number of modules, K_m the membrane transport coefficient (a value of 2.2×10^{-8} m/s was used), He^j the distribution constant of the pollutant between the organic phase and the aqueous phase, and Co^j is the concentration of the contaminant in the organic phase. In the simplified case, where extraction and back-extraction are

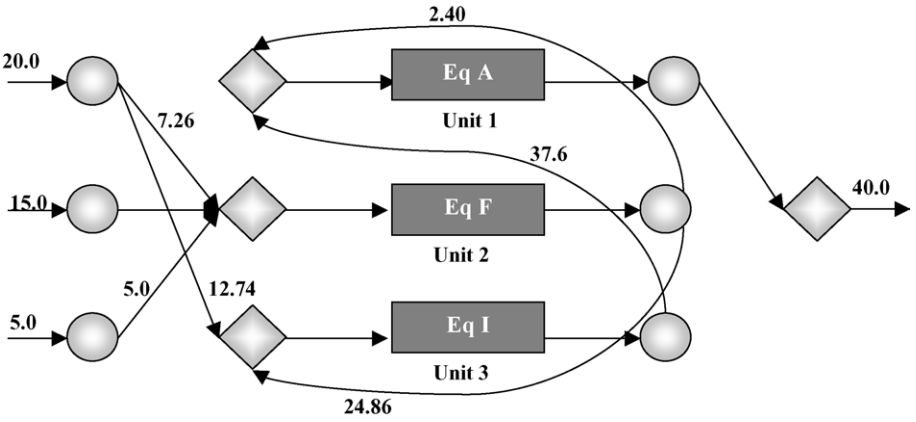

Fig. 10. Optimal solution for Example 4.

Table 7
Solution steps for Example 4

Outer iteration	Inner iteration	Solution MILP-1	Solution NLP	Solution MILP-2	GUB	LUB	GLB	LLB
1	1	1080714.18	Infeasible	Infeasible	–	–	1080714.18	1080714.18
2	1	1082892.69	Infeasible	Infeasible	–	–	1082892.69	1082892.69
3		1235559.63					1235559.63	1235559.63
	1		1992836.21	1449071.22	1992836.21	1992836.21		1449071.22
	2		1692583.88	1482263.35	1692583.88	1692583.88		1482263.35
	3		1992836.21	1508500.95				1508500.95
	4		1692583.88	1635451.81				1635451.81
	5		1697253.17	1683607.48				1683607.48
4		1235559.63*					1235559.63	
5		1235559.63*					1235559.63	
6		Infeasible						

* The selection of the equipment from the MILP-1 is proven to be worse than the best solution in the reduction steps.

Table 8
Model sizes and solution time for Example 4

Outer iteration	Inner iteration	Solution MILP-1		Solution NLP		Solution MILP-2	
		CPU time (s)	Disc vars/Cont vars	CPU time (s)	Cont vars	CPU time (s)	Disc vars/Cont vars
1	1	2.062	33/544	0.039	180	1.265	30/616
2	1	1.953	33/544	0.098	180	1.453	30/616
3		3.359	33/544				
	1			0.059	180	1.593	31/624
	2			0.121	180	11.921	41/740
	3			0.090	180	15.468	48/820
	4			0.095	180	30.670	51/856
	5			0.041	180	55.187	57/916
4		1.437	33/544				
5		1.218	33/544				
6		1.281	33/544				
Total time		11.310		0.543		117.557	

carried out at the same rate, we can assume that Co^j remains constant.

The superstructure for this problem is identical to Example 4. The data for the equipment, inlet streams and costs are shown in Tables 9–11.

The global optimum (US$ 30,481.13) was found in the first outer iteration, but the convergence within 1% tolerance of the global optimum was obtained in 10 outer iterations. The first selected structure required four inner iterations each to check globality. The gridpoint sets were updated in each inner optimization using the middle point of the active subinterval. Details of the solution in each iteration can be seen in Table 12, as well as the global and local lower and upper bounds. Fig. 11 shows the progress of the bounds. Note that the global lower bound defines a piecewise increasing path, and the global upper bounds describes a piecewise decreasing path, always above the global lower bound line. This does not occur with the local bounds. Local bounds involve discontinuities when the inner loop finishes and outer iteration changes. Also note that inner loop stops if the local lower bound reaches the global local bound.

(MILP-1) problems have 51 binary and 790 continuous variables, whilst the (MILP-2) problems have on average 60 binary variables and 973 continuous variables in the first in-

Table 9
Distribution of the pollutant He^j and concentration of pollutant in organic phase Co^j

Unit		A	B	C
Treatment X	He^j	1900	1700	0
	Co^j	200	200	0
Treatment XX	He^j	0	1700	1900
	Co^j	0	200	200
Treatment XXX	He^j	1700	0	1500
	Co^j	200	0	200

Table 10
Inlet streams data for Example 5

Inlet Stream	Flowrate (tonnes/h)	Pollutant	ppm
1	13.1	A	390
		B	100
		C	250
2	32.7	A	168
		B	110
		C	400
3	56.5	A	250
		B	100
		C	350

Table 11
Cost and removal ratio data for the equipments in Example 5

Treatment unit k	Equipment H	NM	Cost function* ($\alpha F^{0.6} + \gamma F$)	
			Investment α	Operating γ
1	EA	15	250.00	0.0180
	EB	20	301.40	0.0247
	EC	25	348.45	0.0316
2	ED	15	250.00	0.0180
	EE	20	301.40	0.0247
	EF	25	348.45	0.0316
3	EG	15	250.00	0.0180
	EH	20	301.40	0.0247
	EI	25	348.45	0.0316

* F is the treated flowrate, given in ton/h.

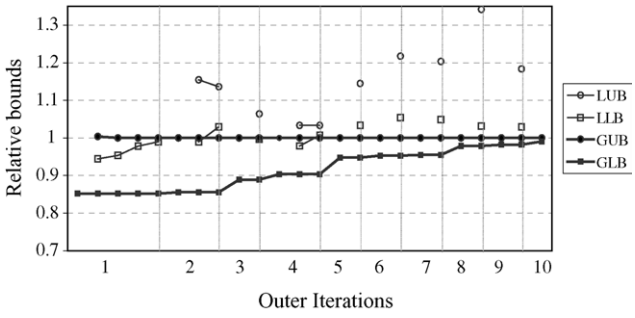

Fig. 11. Bound progress in Example 4.

ner iteration, and their size grow as the inner iterations proceed. The fourth (MILP-2) in outer iteration 1 has 114 binary variables and 1522 continuous variables. The time required to solve the 10 outer master problems is 0.33 min aproximately; the bound reduction steps take a total of 0.83 min. The algorithm spends 2.5 s in solving the NLPs problems and 18 min in solving the bounding problems (MILP-1). The optimal values for the flows are shown in Fig. 12 (flow values are given in tonnes/h).

Numerical difficulties were experienced with BARON, which prevented convergence to feasible solutions; and hence, a comparison of computational times was not possible for this problem.

Table 12
Solution steps for Example 5

Outer iteration	Inner iteration	Solution MILP-1	Solution NLP	Solution MILP-2	GUB	LUB	GLB	LLB
1		25963.96			–	–	25963.96	25963.96
	1		30598.67	28773.15	30598.67	30598.67		28773.15
	2		30481.13	29051.94	30481.13	30481.13		29051.94
	3		30481.13	29809.68		30481.13		29809.68
	4		30481.13	30170.21		30481.13		30170.21
2		26070.73			30481.13		26070.73	26070.73
	1		35182.82	30167.93		35182.82		30167.93
	2		31972.22	31373.77		31972.22		31373.77
3	1	27100.94			30481.13		27100.94	27100.94
			35531.09	30351.09		35531.09		30351.09
4		27533.17			30481.13		27533.17	27533.17
	1		31488.26	29830.80		31488.26		29830.80
	2		31796.13	30700.72		31488.26		30700.72
5	1	28876.28			30481.13		28876.28	28876.28
	1		34882.90	31494.12		34882.90		31494.12
6		29038.51			30481.13		29038.51	29038.51
	1		37100.28	32135.22		37100.28		32135.22
7		29098.96			30481.13			29098.96
	1		36675.09	31969.28		36675.09		31969.28
8		29832.45			30481.13			29832.45
	1		40905.71	31442.17		40905.71		31442.17
9		29924.85			30481.13		29924.85	29924.85
			36071.18	31369.43		36071.18		31369.43
10		30191.21			30481.13		30191.21	

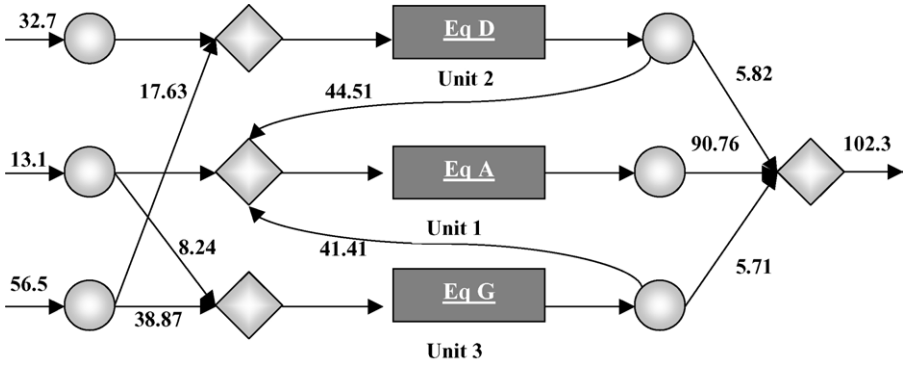

Fig. 12. Global optimal solution for Example 5.

9. Conclusions and future work

A new deterministic algorithm for the global optimization of synthesis of processes network problems has been presented. It is based on a new methodology for constructing underestimators of nonconvex functions based on partitions of the entire domain. In this work, the derivation of this class of estimators for univariate concave terms and bilinear terms has been developed.

The proposed algorithm relies on an outer approximation methodology. The global solution of the problem is achieved by solving problems that are relaxations of the original one. As iterations proceed, the bounding problem approximates the original problem with more accuracy.

The effectiveness of the proposed algorithm has been illustrated in several examples as well as comparisons with other existent algorithm to solve this class of problems. The computational experience, although still limited, suggests that this algorithm has several advantages with respect to spatial branch-and-bound algorithms, particularly in regard to ease of implementation and potential strengthening of lower bounds.

For larger problems, however, the relaxed MILP problems predict bounds with significant gap and convergence is achieved at high computational cost. A modification of the algorithm is being studied, involving the solution of the convexified C-MINLP problem. Also, most of the computing time is spent in the inner optimization. This is due to the iterative procedure and the increasing size of the MILP-2 problems. An alternative methodology for obtaining the global solution of the reduced NLPs is also being investigated. It involves the simultaneous grid update and solution of the local bounding problem.

Acknowledgments

The authors would like to acknowledge financial support from CONICET, ANCyT and UNL from Argentina, the Center for Advanced Process Decision-making at Carnegie Mellon, and the National Science Foundation under grant INT-0104315.

Appendix A. Derivation of piecewise linear underestimators of concave univariate functions

The convex envelope of a concave function on an interval $I = [x^{lo}, x^{up}]$ is

$$f^{u}(x) = \lambda f(x^{lo}) + (1-\lambda) f(x^{up})$$

where λ is such that $x = \lambda x^{lo} + (1-\lambda)x^{up}$.

Given the partition $\{I_k\}_{k=1}^{K}$, with $I_k=[x^k, x^{k+1}]$, $k=1,\ldots,K$, $x^1 = x^{lo}$, $x^{K+1} = x^{up}$, the piecewise underestimator can be formulated as a disjunction with k terms:

$$\bigvee_{k=1,\ldots,K} \begin{bmatrix} W_k \\ x = \lambda x^k + (1-\lambda)x^{k+1} \\ f^{u} = \lambda f(x^k) + (1-\lambda) f(x^{k+1}) \\ 0 \le \lambda \le 1 \end{bmatrix}$$

The mixed-integer formulation based on the convex hull relaxation (Raman and Grossmann, 1994) is as follows:

$$\begin{aligned} x &= \sum_{k=1}^{K} \lambda_k x^k + (w_k - \lambda_k) x^{k+1} \\ &= \lambda_1 x^1 + (w_1 - \lambda_1 + \lambda_2) x^2 + \cdots + (w_K - \lambda_K) x^{K+1} \\ f^{u} &= \sum_{k=1}^{K} \lambda_k f(x^k) + (w_k - \lambda_k) f(x^{k+1}) \\ &= \lambda_1 f(x^1) + (w_1 - \lambda_1 + \lambda_2) f(x^2) \\ &\quad + \cdots + (w_K - \lambda_K) f(x^{K+1}) \\ 0 &\le \lambda_k \le w_k \\ \sum_{k=1}^{K} w_k &= 1 \\ w_k &\in \{0, 1\} \end{aligned}$$

Let us define $\gamma_k = w_{k-1} - \lambda_{k-1} + \lambda_k$, $k = 2, \ldots, K$, $\gamma_1 = \lambda_1$ and $\gamma_{K+1} = w_K - \lambda_K$. With these weights, the convex combination can be expressed as the equivalent formulation:

$$x = \sum_{k=1}^{K+1} \gamma_k x^k$$

$$f = \sum_{k=1}^{k+1} \gamma_k f(x^k)$$

$$0 \le \gamma_1 \le w_1$$

$$0 \le \gamma_k \le w_k + w_{k-1}, \quad k = 2, \ldots, K$$

$$0 \le \gamma_{K+1} \le w_K$$

$$\sum_{k=1}^{K} w_k = 1$$

This second formulation is the same as the formulation given in Nemhauser and Wosley (1999).

An interesting discussion about the quality of two formulations of piecewise-linear estimators can be found in Padberg (2000).

Appendix B. Piecewise underestimators for bilinear terms

Consider the bilinear term $f(x,y) = xy$, defined in the domain $D = [x^{\mathrm{lo}}, x^{\mathrm{up}}] \times [y^{\mathrm{lo}}, y^{\mathrm{up}}]$, and consider the partition $\{D_k\}_{k=1}^{K}$, with $D_k = [x^k, x^{k+1}] \times [y^{\mathrm{lo}}, y^{\mathrm{up}}]$, $k = 1, \ldots, K$, $x^1 = x^{\mathrm{lo}}$, $x^{k+1} = x^{\mathrm{up}}$. A piecewise linear underestimator f^{u} will be derived, such that $f^{\mathrm{u}}(x^k, y) = f(x^k, y)$.

$$\bigvee_{k=1,\ldots,K} \begin{bmatrix} W_k \\ a^k = xy^{\mathrm{lo}} + x^k y - x^k y^{\mathrm{lo}} \\ b^k = xy^{\mathrm{up}} + x^{k+1} y - x^{k+1} y^{\mathrm{up}} \\ f^{\mathrm{u}} = \max\{a^k, b^k\} \\ x^k \le x \le x^{k+1} \end{bmatrix}$$

The mixed-integer formulation based on the convex hull relaxation is as follows:

$$x = \upsilon^1 + \upsilon^2 + \cdots + \upsilon^K$$

$$y = \gamma^1 + \gamma^2 + \cdots + \gamma^K$$

$$f^{\mathrm{u}} = f^{\mathrm{u}^1} + f^{\mathrm{u}^2} + \cdots + f^{\mathrm{u}^K}$$

$$a^k = \upsilon^k y^{\mathrm{lo}} + x^k \gamma^k - x^k y^{\mathrm{lo}} w^k$$

$$b^k = \upsilon^k y^{\mathrm{up}} + x^{k+1} \gamma^k - x^{k+1} y^{\mathrm{up}} w^k, \quad k = 1, \ldots, K$$

$$f^{\mathrm{u}^k} = \max\{a^k, b^k\}$$

$$x^k w^k \le \upsilon^k \le x^{k+1} w^k$$

$$y^{\mathrm{lo}} w^k \le \gamma^k \le y^{\mathrm{up}} w^k$$

$$\sum_{k=1}^{K} w^k = 1$$

$$w^k \in \{0, 1\}$$

References

Adjman, C. S., Androulakis, I. P., & Floudas, C. A. (2000). Global optimization of mixed-integer nonlinear problems. *AIChE Journal*, *46*(9), 1769–1797.

Bazaraa, M. S., Sherali, H. D., & Shetty, C. M. (1993). *Nonlinear programming, theory and algorithms* (2nd ed.). New York: Wiley.

Björn, K. M., Lindberg, P. O., & Westerlund, T. (2003). Some convexifications in global optimization of problems containing signomial terms. *Computer and Chemical Engineering*, *27*, 669.

Brooke A., D. Kendrick, A. Meeraus, & R. Raman (1997). *GAMS Language Guide*, Release 2.25, Version 92. GAMS Development Corporation.

Dantzig, G. B. (1963). *Linear programming and extensions*. Princeton: Princeton University Press.

Duran, M. A., & Grossmann, I. E. (1986). An outer-approximation algorithm for a class of mixed-integer nonlinear programms. *Mathematical Programming*, *36*, 307.

Fletcher, R., & Leyffer, S. (1994). Solving mixed Integer nonlinear programs by outer approximation. *Mathematical Programming*, *66*, 550.

Floudas, C. A. (2000). *Deterministic global optimization: Theory, methods and applications*. Dordrecht, The Netherlands: Kluwer Academic Publishers.

Galan, B., & Grossmann, I. E. (1998). Optimal design of distributed wastewater treatment networks. *Industrial and Engineering Chemistry Research*, *37*(10), 4036.

Geoffrion, A. M. (1972). Generalized benders decomposition. *Journal of Optimization Theory and Applications*, *10*(4), 237.

Hooker, N., & Osorio, M. A. (1999). Mixed logical/linear programming. *Discrete Applied Mathematics*, *96–97*, 395.

Horst, R., & Tuy, P. M. (1996). *Global optimization: Deterministic approaches* (3rd ed.). Berlin: Springer-Verlag.

Kesavan, P., Allgor, R. J., Gatzke, E. P., & Barton, P. I. (2004). Outer approximation algorithms for seperable nonconvex mixed-integer nonlinear programs. *Mathematical Programming*, *100*, 517.

Kesavan, P., & Barton, P. I. (2000). Generalized Branch and Cut framework for mixed-integer Nonlinear Optimization Programs. *Computer and Chemical Engineering*, *24*, 1361.

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

Lee, S., Colberg, R. D., Siirola, J. J., & Grossmann, I. E. (2002). Global optimization of disjunctive program for the superstructure of an industrial monomer process, *Annual Meeting AIChE*, Indianapolis.

Lee, S., & Grossmann, I. E. (2001). A global optimization algorithm for nonconvex Generalized Disjuntive Programming and Applications to Process Systems. *Computer and Chemical Engineering*, *25*, 1675.

Lee, S., & Grossmann, I. E. (2003). Global optimization for nonlinear generalized disjunctive programming with bilinear equality constraints: Applications to process networks. *Computer and Chemical Engineering*, *27*(11), 1557.

McCormick, G. P. (1976). Computability of global solutions to factorable nonconvex programs. Part I. Convex underestimating problems. *Mathematical Programming*, *10*, 146.

Nemhauser, & Wosley. (1999). *Integer and combinatorial optimization*. New York: John Wiley and Sons.

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

Pörn, P., Björn, K., Westerlund, T. (2005). Global solution of optimization problems with signomial parts. *Discrete Optimization*.

Quesada, I., & Grossmann, I. E. (1995). A global optimization algorithm for linear fractional and bilinear programs. *Journal of Global Optimization*, *6*, 39.

Raman, R., & Grossmann, I. E. (1994). Modeling and computational techniques for logic based integer programming. *Computers and Chemical Engineering*, *18*, 563.

Ryoo, H. S., & Sahinidis, N. (1995). Global optimization of nonconvex NLPs and MINLPs with applications in process design. *Computer and Chemical Engineering*, *19*(5), 551.

Sahinidis, N. (1996). BARON: A general purpose global optimization software package. *Journal of Global Optimization*, *8*(2), 201.

Smith, E. M. B., & Pantelides, C. C. (1999). A symbolic reformulation/spatial branch and bound algorithm for the global optimization of nonconvex MINLPs. *Computer and Chemical Engineering*, *23*, 457.

Stubbs, R. A., & Mehrotra, S. (1999). A branch and cut method for 0–1 mixed convex programming. *Mathematical Programming*, *86*(3), 515.

Tawarmalani, M., & Sahinidis, N. V. (2004). Global optimization of mixed-integer nonlinear programs: A theoretical and computational study. *Mathematical Programming*, *99*(3), 563.

Tawarmalani, M., & Sahinidis, N. (2002). *Convexification and global optimization in continuous and mixed-integer nonlinear programming*. Kluwer.

Türkay, M., & Grossmann, I. E. (1996). Logic-based MINLP algorithm for the optimal synthesis of process networks. *Computer and Chemical Engineering*, *20*, 959.

Vecchietti, A., & Grossmann, I. E. (1999). LOGMIP: A disjunctive 0-1 nonlinear Optimizer for process system models. *Computer and Chemical Engineering*, *23*, 555.

Viswanathan, J., & Grossmann, I. E. (1990). A combined penalty function and outer approximation method for MINLP optimization. *Computer and Chemical Engineering*, *14*, 769.

Visweswaran, V., & Floudas, C. A. (1996). New reformulations and branching strategies for the GOP algorithm. In I. E. Grossmann (Ed.), *Global Optimization in Engineering Design*. Kluwer.

Westerlund, T., & Petterson, F. (1995). A cutting plane method for solving convex MINLP problems. *Computer and Chemical Engineering*, *19*, S131–S136.

Zamora, J. M., & Grossmann, I. E. (1999). A branch and contract algorithm for problems with concave univariate, bilinear and linear fractional terms. *Journal of Global Optimization*, *14*, 217.