DETERMINISTIC GLOBAL OPTIMIZATION IN DESIGN, CONTROL, AND COMPUTATIONAL CHEMISTRY

CHRISTODOULOS A. FLOUDAS*

Abstract. This paper presents an overview of the deterministic global optimization approaches and their applications in the areas of Process Design, Control, and Computational Chemistry. The focus is on (i) decomposition-based primal dual methods, (ii) methods for generalized geometric programming problems, and (iii) global optimization methods for general nonlinear programming problems. The classes of mathematical problems that are addressed range from indefinite quadratic programming to concave programs, to quadratically constrained problems, to polynomials, to general twice continuously differentiable nonlinear optimization problems. For the majority of the presented methods nondistributed global optimization approaches are discussed with the exception of decomposition-based methods where a distributed global optimization approach is presented.

1. Background. A significant effort has been expended in the last five decades toward theoretical and algorithmic studies of applications that arise in Process Synthesis, Design, Control, and Computational Chemistry. In the last decade the area of global optimization has attracted a lot of interest from the Operations Research and Applied Mathematics community, while in the last five years we have experienced a resurgence of interest in Chemical Engineering for new methods of global optimization as well as the application of available global optimization algorithms to important engineering problems. This recent surge of interest is attributed to three main reasons. First, a large number of process synthesis, design, control, and computational chemistry problems are indeed global optimization problems. More specifically, in the area of Process Synthesis and Design, global optimization problems arise in phase equilibrium, nonideal separations, energy optimization, reactor-based systems, parameter estimation, data reconciliation, and metabolic reaction pathways. In the area of Process Control, global optimization issues are in the robust control analysis of systems with real parametric uncertainty, stability analysis of polytopes of matrices, optimal control of complex reaction mechamisms, and nonlinear model predictive control. In the area of Process Operations, the design of systems under uncertainty, and the planning and scheduling of batch, semi-continuous, and continuous processes result in global optimization problems. In the area of Computational Chemistry, global optimization problems arise in the clusters of atoms and molecules, the design of small organic molecules, the three-dimensional structure prediction of oligopeptides, and polypeptides, the prediction of protein structure, the interaction of proteins, the refinement of X-ray and NMR data, and the design of constrained peptides. Second, the existing local nonlinear

* Department of Chemical Engineering, Princeton University, Princeton, N.J. 08544-5263.

optimization approaches (e.g., generalized reduced gradient and successive quadratic programming methods) may either fail to obtain even a feasible solution or are trapped to a local optimum solution which may differ in value significantly form the global solution. Third, the global optimum solution may have a very different physical interpretation when it is compared to local solutions (e.g., in phase equilibrium a local solution may provide incorrect prediction of types of phases at equilibrium, as well as the components' composition in each phase).

The existing approaches for global optimization are classified as deterministic or probabilistic. The deterministic approaches include: (a) Lipschitzian methods (e.g. Hansen et al. 1992 a, b), (b) Branch and Bound methods (e.g. Al-Khayyal and Falk 1983; Horst and Tuy, 1987; Al-Khayyal 1990), (c) Cutting Plane methods (e.g. Tuy et al. 1985), (d) Difference of Convex (D.C.) and Reverse Convex methods (e.g.. Tuy 1987 a,b), (e) Outer Approximation methods (e.g. Horst et al. 1992), (f) Primal-Dual methods (e.g. Shor 1990; Floudas and Visweswaran 1990, 1993; Ben-Tal et al. 1994), (g) Reformulation-Linearization methods (e.g. Sherali and Alameddine, 1992; Sherali and Tuncbilek 1992), and (h) Interval methods (e.g. Hansen 1979). The probabilistic methods include (i) random search approaches (e.g. Kirkpatrick et al. 1983), and (ii) clustering methods (e.g. Rinnoy Kan and Timmer 1987). Recent books that discuss the above classes are available by Pardalos and Rosen (1987), Torn and Zilinskas (1989), Ratschek and Rokne (1988), Horst and Tuy (1990), Floudas and Pardalos (1992), Horst and Pardalos (1995), Horst et al. (1995), Pinter (1996), Grossmann (1996) and Floudas and Pardalos (1996).

Contributions from the chemical engineering community to the area of global optimization can be traced to the early work of Stephanopoulos and Westerberg (1975), Westerberg and Shah (1978), and Wang and Luus (1978). Renewed interest in seeking global solution was motivated form the work of Floudas et al (1989). The first exact primal-dual global optimization approach was proposed by Floudas and Visweswaran (1990), (1993) and its features were explored for quadratically constrained and polynomial problems in the work of Visweswaran and Floudas (1992), (1993). Swaney (1990) proposed a branch and bound global optimization approach and more recently Quesada and Grossmann (1993, 1995) combined convex underestimators in a branch and bound framework for linear fractional and bilinear programs. Manousiouthakis and Sourlas (1992) proposed a reformulation to a series of reverse convex problems, and Tsirukis and Reklaitis (1993 a,b) proposed a feature extraction algorithm for constrained global optimization. Maranas and Floudas (1992, 1993, 1994a,b) proposed a novel branch and bound method combined with a difference of convex functions transformation for the global optimization of molecular conformation problems that arise in computational chemistry. Vaidyanathan and El-Halwagi (1994) proposed an interval analysis based global optimization method and Ryoo and Sahinidis (1995) suggested the application of reduction tests within the framework of branch and bound methods. Androulakis et al. (1995) proposed the global optimization method $\alpha\mathbf{BB}$ which addresses general continuous optimization problems with nonconvexities in the objective function and/or constraints. This approach classifies the nonconvexities as special structure (e.g., bilinear, signomial, univariate) or generic structure and is based on convex relaxations and a branch and bound framework. Maranas and Floudas (1995) proposed a new approach for enclosing all ϵ-feasible solutions of nonlinearly constrained systems of equations. This approach transforms the problem into a min-max form and corresponds to enclosing all multiple global optima via the $\alpha\mathbf{BB}$ global optimization approach. A variety of convex underestimators for trilinear, fractional, generalized polynomial, and products of univariate functions were proposed. Maranas and Floudas (1996) proposed a global optimization approach for generalized geometric programming problems that have many applications in robust control and engineering design problems. In a series of papers McDonald and Floudas (1994;1995a,b,c) addressed the fundamental problems of (i)minimization of the Gibbs free energy and (ii) the tangent plane stability criterion that arise in phase and chemical reaction equilibrium as global optimization problems for the first time. They proposed decomposition based approaches for biconvex problems that result from the use of the NRTL equation, and branch and bound approaches for the UNIQUAC, UNIFAC, ASOG, and TK-Wilson activity coefficient models. McDonald and Floudas (1996) proposed the combination of the two aforementioned classes of problems, developed a special purpose program GLOPEQ, and performed an extensive computational study on difficult phase equilibrium problems.

The books of Floudas and Pardalos (1996) and Grossmann (1996) contain a number of recent chemical engineering contributions which are briefly discussed in the following. Staus et al. (1996) formulated the combined adaptive controller design and estimation problem as a nonconvex problem with convex objective and bilinear constraints, and proposed a branch and bound global optimization method which is based on the McCormick underestimators. Visweswaran et al. (1996) addressed bilevel linear and quadratic programming problems to global optimality by employing the basic principles of the GOP and developing additional theoretical properties that exploit the underlying mathematical structure of such problems. Androulakis et al. (1996) developed a distributed version of the GOP, and discussed the key theoretical and implementation issues along with extensive computational results on large scale indefinite quadratic and pooling/blending problems. Shectman and Sahinidis (1996) proposed a finite algorithm for separable concave programs, discussed the design of such branch and bound approaches, and presented computational results employing domain reduction tests. McKinnon et al. (1996) addressed the global optimization in phase and chemical reaction equilibrium using interval analysis coupled with the tangent plane stability criterion of

Gibbs. Mockus and Reklaitis (1996) proposed a continuous formulation for the short term batch scheduling propblem, and developed a global optimization approach based on the Bayesian heuristic. Lucia and Xu (1996) studied nonconvexity issues in sparse successive quadratic programming, and Banga and Seider (1996) introduced a stochastic global optimization approach which they applied to the optimal design of a fermentation process, phase equilibrium problems and optimal control problems. Epperly and Swaney (1996a,b) proposed a branch and bound method with a new linear programming underestimating problem, and provided extensive computational studies. This approach is applicable to NLPs in factorable form which includes quadratic objective and constraints, and twice differentiable transcendental functions. Visweswaran and Floudas (1996a) proposed new formulations for the GOP algorithm which are based on a branch and bound framework, allow the implicit solution of the relaxed dual problems which are formulated in a single MILP model, and feature a linear branching scheme. Visweswaran and Floudas (1996b) discussed the implementation issues of the GOP and provide extensive computational experience on a variety of chemical engineering probelms. Byrne and Bogle (1996) and Vaidyanathan and El-Halwagi (1996) proposed global optimization methods that are based on interval analysis for constrained NLPs and MINLPs respectively. Liu et al. (1996) proposed a new approach for planning of chemical process networks which is based on global concave minimization. The approach is based on their earlier work on finite global optimization approaches and their computational studies revealed the efficiency of the proposed approach. Ierapetritou and Pistikopoulos (1996) studied the global optimization of stochastic planning, scheduling and design problems, applied the decomposition-based approach, GOP, and demonstrated that significant reductions in the number of relaxed dual problems can be achieved by exploiting the mathematical structure further. Iyer and Grossmann (1996) extended the global optimization approach of Quesada and Grossmann (1993) to the multiperiod heat exchanger networks that feature fixed configuration, linear cost functions, arithmetic mean driving forces, and isothermal mixing. Quesada and Grossmann (1996) studied further the use of alternative bounding approximations and applied them to a variety of engineering design problems that include structural design, batch processes, layout design, and portfolio problems. Smith and Pantelides (1996) proposed a symbolic manipulation algorithm for the automatic reformulation of algebraic constraints and introduced a spatial type branch and bound approach within the gPROMS framework.

Recently, Adjiman et al. (1996) and Adjiman and Floudas (1996) proposed novel approaches for the rigorous determination of the α parameters that are employed in the αBB global optimization approach. These methods are based on interval anlysis of the hessian matrices and calculate rigorous bounds on the minimum eigenvalue for general twice differentiable problems.

In this paper, we will focus on deterministic global optimization methods that have been developed in the Computer-Aided Systems Laboratory, CASL, of the Department of Chemical Engineering of Princeton University. These will be classified as (i) decomposition-based primal-relaxed dual methods, (ii) methods for generalized geometric programming models, and (iii) methods for general nonlinear programming problems.

2. Decomposition methods.

2.1. The primal-relaxed dual approach, GOP.

Floudas and Visweswaran (1990,1993) proposed a deterministic primal-relaxed dual global optimization approach, **GOP**, for solving several classes of nonconvex optimization models for their global solutions.

2.1.1. Formulation.

The general form of the optimization problem addressed by the **GOP** approach is:

$$\min_{x,y} \; f(x,y)$$

$$\begin{aligned} s.t. \quad g(x,y) &\le 0 \\ h(x,y) &= 0 \\ x &\in X \\ y &\in Y \end{aligned} \tag{2.1}$$

where X and Y are non-empty, compact, convex sets, $f(x,y)$ is the objective function to be minimized, $g(x,y)$ is a vector of inequality constraints and $h(x,y)$ is a vector of equality constraints. It is assumed that these functions are continuously differentiable over $X \times Y$. For the sake of convenience, it will be assumed that the set X is incorporated into the first two sets of constraints. In addition, the problem is also assumed to satisfy the following conditions:

Conditions (A):

(a) $f(x,y)$ and $g(x,y)$ are convex in x for every fixed y, and convex in y for every fixed x,

(b) $h(x,y)$ is affine in x for every fixed y, and affine in y for every fixed x,

(c) $Y \subseteq V$, where $V \equiv \{y : g(x,y) \le 0, \; h(x,y) = 0, \text{ for some } x \in X\}$, and

(d) An appropriate constraint qualification (e.g., Slater's, linear independence qualification) is satisfied for fixed y.

It has been shown Floudas and Visweswaran (1990, 1993) that the class of problems that satisfies these conditions includes, but is not restricted to, bilinear problems, quadratic problems with quadratic constraints and polynomial and rational polynomial problems. Recently, it has also been shown (Liu and Floudas 1993, 1995) that a very large class of smooth optimization problems can be converted to a form where they satisfy *Conditions (A)*, and hence are solvable by the GOP algorithm. Liu and Floudas (1996) proposed a generalized primal-relaxed dual approach which contains the GOP as a special case, extended the GOP to certain classes of nonsmooth problems, and suggested a penalty type implementation for improving the computational efficiency.

2.1.2. Concepts and properties of the GOP approach. The GOP algorithm utilizes primal and relaxed dual subproblems to obtain upper and lower bounds on the global solution. The primal problem results from fixing the y variables to some value, say y^k, and is defined as follows:

$$\min_x \quad f(x, y^k),$$

$$\text{(2.2)} \qquad \begin{aligned} s.t. \quad g(x, y^k) &\leq 0 \\ h(x, y^k) &= 0 \end{aligned}$$

where $y^k \in Y$. It has been assumed here that any bounds on the x variables are incorporated into the first set of constraints. Notice that because of the introduction of additional constraints by fixing the y variables, this problem provides an upper bound on the global optimum of (2.1). Moreover, $P^k(y^k)$, the solution of this problem yields a solution x^k for the x variables and Lagrange multipliers λ^k and μ^k for the equality and inequality constraints respectively[1].

The Lagrange function constructed from the primal problem is given as:

$$\text{(2.3)} \qquad L^k(x, y, \lambda^k, \mu^k) = f(x, y) + \lambda^k h(x, y) + \mu^k g(x, y).$$

The x variables that are present in the linearization of the Lagrange function around x^k, and for which the gradients of the Lagrange functions with respect to x at x^k are functions of the y-variables, are called the *connected* variables. It can easily be shown that the linearization of the Lagrange function around x^k can also be written in the form:

$$\text{(2.4)} \qquad L^k(x, y, \lambda^k, \mu^k)\big|_{x^k}^{lin} = L_0^k(y, \lambda^k, \mu^k) + \sum_{i=1}^{NI_C^k} x_i g_i^k(y)$$

where NI_C^k is the number of *connected* variables at the k^{th} iteration, and $L_0^k(y, \lambda^k, \mu^k)$ represents all the terms in the linearized Lagrange function

[1] It is assumed here that the primal problem is feasible for $y = y^k$. See Floudas and Visweswaran (1990, 1993) for the treatment of the cases when the primal problem is infeasible for a given value of y.

that depend only on y. The positivity and negativity of the functions $g_i^k(y)$, which represent the gradients of the Lagrange function with respect to the variables x_i at iteration k, define a set of equations that are called the *qualifying* constraints of the Lagrange function at the k^{th} iteration, and which partition the y variable space into $2^{NI_c^k}$ subregions. In each of these subregions, a Lagrange function can be constructed (using the bounds for the x variables) that underestimates the global solution in the subregion, and can therefore be minimized to provide a lower bound for the global solution in that region.

Consider the first iteration of the GOP algorithm. The initial *parent* region is the entire space $y \in Y$ from the original problem. This region is subdivided into $2^{NI_c^1}$ subregions, and in each of these subregions, a subproblem of the following form is solved:

$$\min_{y \in Y, \mu_B} \quad \mu_B$$

$$\begin{array}{lrcl} s.t. & \mu_B & \geq & L^1(x^{B_l}, y, \lambda^1, \mu^1)\big|_{x^1}^{lin}, \\ & & & \left.\begin{array}{rcl} g_i^1(y) & \geq 0 & \text{if} \quad x_i^{B_l} = x_i^L \\ g_i^1(y) & \leq 0 & \text{if} \quad x_i^{B_l} = x_i^U \end{array}\right\} \ \forall\, i \in I_c^1, \end{array}$$

where I_c^1 is the set of *connected* variables at the first iteration, NI_C^k is the number of *connected* variables, and x_i^L and x_i^U are the lower and upper bounds on the i^{th} *connected* variable respectively. This subproblem corresponds to the minimization of the Lagrange function, with the *connected* variables replaced by a combination of their lower and upper bounds. Note the presence of the *qualifying* constraints in the problem. These constraints ensure that the minimization is carried out in a subregion of the *parent* node. If this problem has a value of μ_B that is lower than the current best upper bound obtained from the primal problem, then it is added to the set of candidate lower bounds; otherwise, the solution is fathomed, that is, removed from consideration for further refinement.

Consider a problem with two x and two y variables. In the first iteration, assuming that both x_1 and x_2 are in the set of *connected* variables for the first iteration, there are four relaxed dual subproblems solved. These problems are shown in Figure 2.1(a). It can be seen that the *qualifying* constraints partition the y-space into the four regions. Each of the relaxed dual subproblems solved provides a valid underestimator for the corresponding region as well as a solution point (denoted in the figure by y^A, y^B, y^C and y^D) in the region. Figure 2.1(b) shows the corresponding branch-and-bound tree created by the solution of these four problems. The starting point y^1 is the root node, and it spawns four leaf nodes. The infimum of the four nodes provides the point for the next iteration, in this case, say y^A.

In the second iteration, the relaxed dual problem is equivalent to further partitioning the subregion that was selected for refinement. In each of these partitions, a *relaxed dual* subproblem is solved. Figure 2.2(a) shows the subregions created in the example, assuming that there was only one *connected* variable in this iteration. The two relaxed dual subproblems solved in this iteration give new solutions y^E and y^F and are possible candidates for entering at future iterations. Figure 2.2(b) shows the corresponding nodes in the branch-and-bound tree created by this iteration.

The preceding discussion illustrates the key features of a branch and bound framework for the algorithm. The framework is based upon the successive refinement of regions by partitioning on the basis of the *qualifying* constraints.

Visweswaran and Floudas (1990, 1992) demonstrated that the GOP can address several classes of problems that include: (i) Bilinear, negative definite and indefinite quadratic programming problems; (ii) Quadratic programming problems with quadratic constraints; and (iii) Unconstrained and constrained optimization of polynomial functions. For the case of polynomial functions in one variable Visweswaran and Floudas (1992) showed that the primal is a single function evaluation while the relaxed dual becomes a system of two linear equations. Visweswaran and Floudas (1993) proposed new theoretical properties that enhance significantly the computational performance of the GOP algorithm. The effect of the new properties is illustrated through application of the GOP algorithm to a difficult inde finite quadratic problem, a multiperiod tankage quality problem that occurs frequently in the modeling of refinery processes, and a set of pooling/blending problems f rom the literature. In addition, extensive computational experience is reported for randomly generated concave and indefinite quadratic programming problems of different sizes. The results show that the properties help to make the algorithm computational ly efficient for fairly large problems.

2.1.3. Branch and bound framework for the GOP. The branch and bound framework for the GOP is based on the following definitions. For a node j in the branch and bound tree, P_j is its parent node, and I_j is the iteration at which node j is created. R_j is the set of constraints defining the region corresponding to node j. At any point, N_j denotes the total number of nodes in the tree, and C denotes the current node.

Root node and starting region : At the beginning of the algorithm, there are no subdivisions in the y-space. Therefore, the root node in the branch and bound tree is simply the starting point for the algorithm, y^1. The region of application for this node (i.e., the *current* region) is the entire y-space.

Reduction tests at each node : At each node, the current region of application is divided into several subregions using the *qualifying* constraints of the current Lagrange function. It is possible to conduct simple

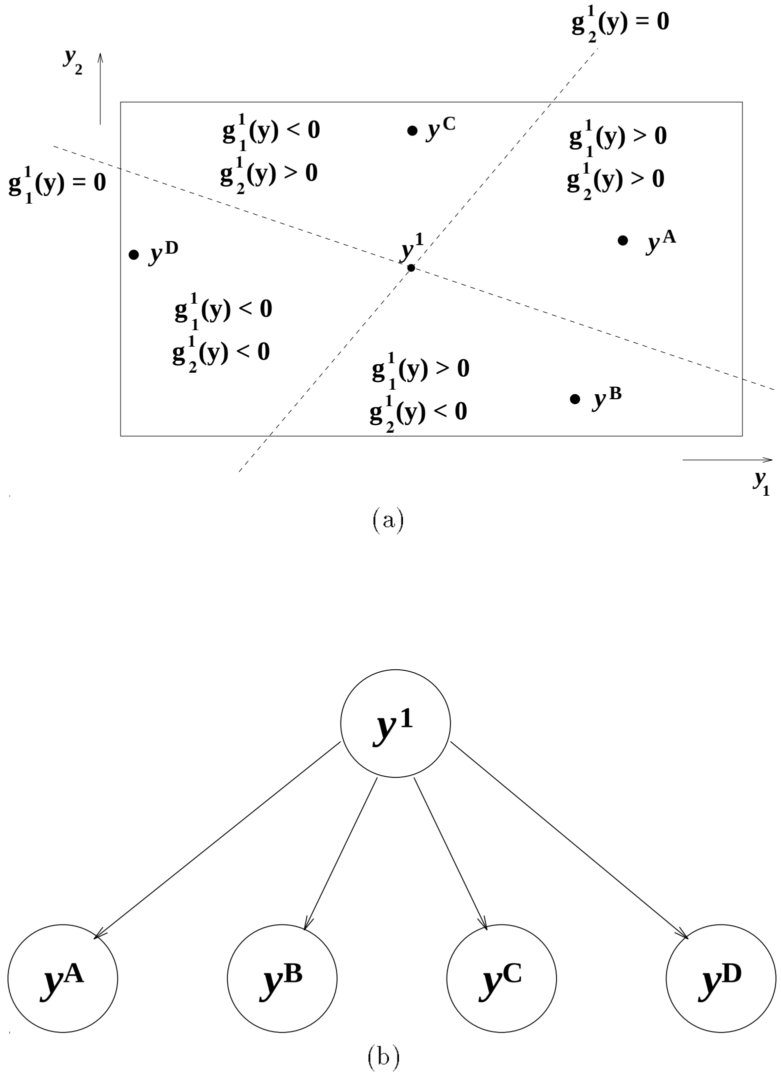
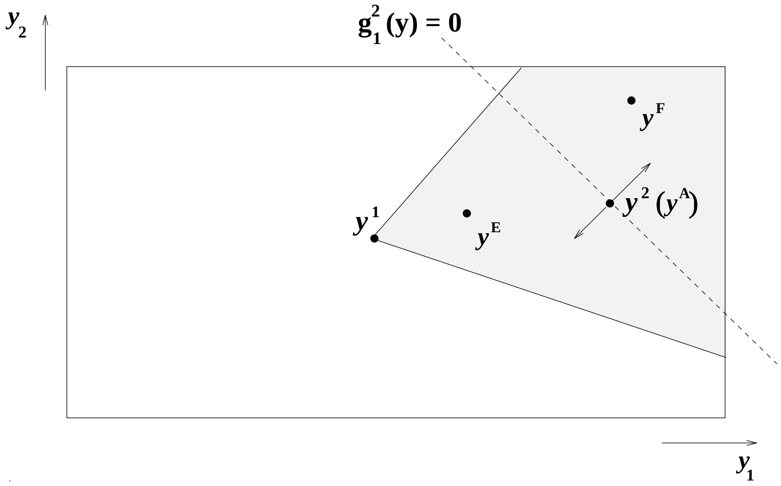
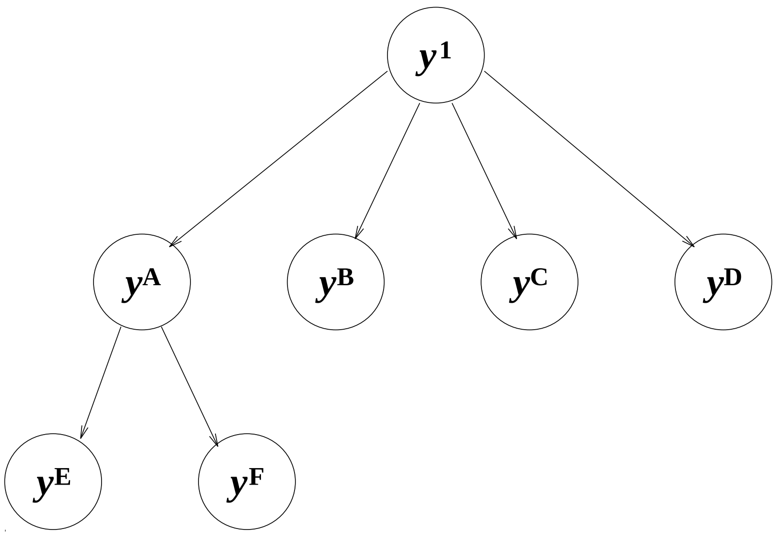


Fig. 2.1. *Partition in the y-space for the first iteration with two* connected *variables*



(a)



(b)

FIG. 2.2. *Partition in the y-space for the second iteration with one* connected *variable*

tests on the basis of the signs of the *qualifying* constraints that can be used to reduce the number of *connected* variables. One such test, based upon the properties first presented in Visweswaran and Floudas (1993), is presented below:

Suppose a node j is to be partitioned in the k^{th} iteration (i.e., $I_j = k$). Then,

(i) If $g_i^k(y) \geq 0 \quad \forall y \in R_j$, set $x_i = x_i^L$ in $L^k(x, y, \lambda^k, \mu^k)$ and remove i from the set of *connected* variables.

(ii) If $g_i^k(y) \leq 0 \quad \forall y \in R_j$, set $x_i = x_i^U$ in $L^k(x, y, \lambda^k, \mu^k)$ and remove i from the set of *connected* variables.

The proofs of the validity of these reductions can be easily obtained by considering that the term $x_i g_i^k(y)$ can be underestimated by $x_i^L g_i^k(y)$ for all positive $g_i^k(y)$ and $x_i^U g_i^k(y)$ for all negative $g_i^k(y)$.

Evaluation of bounds for the x variable : Tighter bounds on the x variables can be obtained by considering the current domains in the y variable space, the existing linear and convex constaints, and convex relaxations of the nonconvex constraints. One way of obtaining such bounds is by minimizing and maximizing each variable such to the aforementioned set of constraints.

The major steps of the branch-and-bound version of the GOP algorithm are described in Visweswaran and Floudas (1996a,b) and are also shown in Figure 3. Floudas and Visweswaran (1990, 1993) showed that the GOP algorithm attains finite convergence to an ϵ-global minimum solution.

2.2. The distributed computing GOP approach. Androulakis et al. (1996) studied the distributed computing issues for the GOP and demonstrated that large-scale quadratic programming and large scale pooling problems can be addressed. They identified the following three major sources of computational challenges : (i) the update of the bounds on the connected x variables, (ii) the solution of the relaxed dual problems, and (iii) the routing of the appropriate data.

The computational difficulty of the GOP algorithm manifests itself in the solution of 2^{NI_c} problems, where NI_c is the number of connected variables. The connected x variables form a sub–set of the original x–type variables. It was shown theoretically, Visweswaran and Floudas (1993), and observed computationally that obtaining tight bounds on the optimization variables, for both the x–type as well as the y–type, is very helpful in the convergence rate of the algorithm. In order to calculate tight variable bounds one has to solve $2(N_x + N_y)$ convex NLP's, where N_x and N_y if the total number of x–type and y–type variables respectively. Therefore, the search for tighter variable bounds problems can be computationally improved if these problems are solved in parallel.

The major computational bottleneck of the method is the solution of a potentially very large number of relaxed dual problems at a given iteration, 2^{NI_c}. Therefore, major emphasis has to be placed on the most efficient

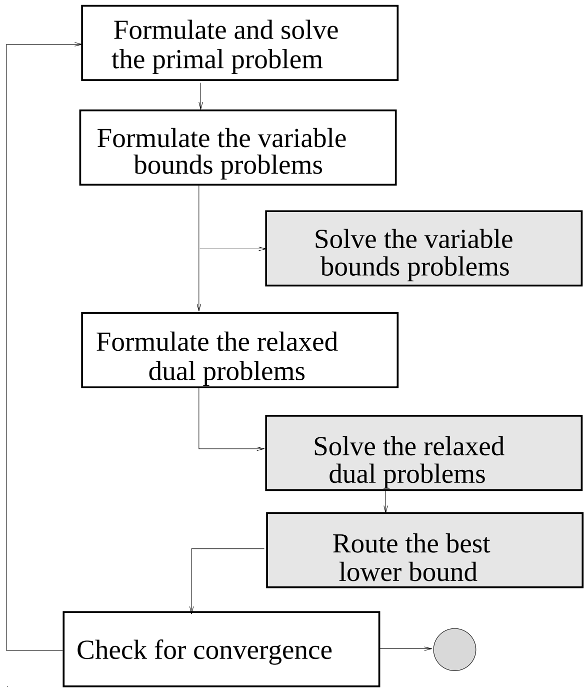


FIG. 2.3. *Flow Diagram of distributed GOP*

solution, from the computational point of view, of this large number of convex or linear optimization problems.

Finally, issues related to the routing of the appropriate data, once a lower bound has been identified, will also be addressed. Such issues require the implementation of a parallel *routing/sorting* algorithm. Figure 2.3 depicts the basic steps of the distributed implementation of the GOP algorithm and highlights the parallelized steps.

2.2.1. Updating the variable bounds. In order to identify the tightest possible variable bounds, we have to calculate the maximum and minimum possible values of all the variables within the current domain of interest. Based on the partitioning induced by the GOP algorithm, the domain of interest for the solution of the relaxed dual problem, is defined by three set of constraints : (a) original convex constraints, (b) Original convexified constraints, and (c) previous qualifying constraint. Sets (a) and (b), define implicitly the range of variables with respect to the original problem. Obviously, any convex constraint, that is convex inequality and/or affine equality, will not alter the convexity characteristics of the problem and thus can be used. Any convexification of the original non–convex constraints will be an overestimation of the feasible region, and it would restrict the domain for the purpose of identifying tighter variable bounds. In addition, the current domain of interest, over which the new lower bound will be sought, is implicitly defined by the set of the previous qualifying constraints.

It was also observed computationally that the frequency at which these problems are solved can be treated as a decision variable. For certain classes of problems, (e.g., indefinite quadratic), computing tight bounds once at the very beginning was adequate, whereas for other classes of problems, (e.g., pooling and blending), the variable bounds had to be updated at each iteration. It is clear that the total number of variable bounds problems that have to be solved are $2(N_x+N_y)$, implying that for large scale optimization problems the framework of distributed computing is needed. With respect to the implementation, it is first identified whether it is worth solving the bounds problems in parallel. Then, the vector of variables is divided into smaller groups and these groups are assigned to nodes who are responsible for solving the variable bounds problems associated with variables. The master node is then collecting the partial vector. The collection process has an unavoidable sequential character but the gains from solving the variable bounds in parallel outperform any potential performance degradation.

2.2.2. Solving the relaxed dual problems. The parallel solution of the relaxed dual problems aims at addressing the need to reduce the computational burden associated with the solution of 2^{NIc} problems at each iteration.

Based on the theoretical analysis of the method, it is clear that all the relaxed dual problems that have to be solved, have the same functional form, and only the bound combinations of the x–type variables will be different. Therefore, what distinguishes one relaxed dual problem problem from the others is the bound combination at which the linearization will be computed, as well as the qualifying constraints that have to be included. As can be seen in Figure 2.4, the y–domain is partitioned based on the signs of the qualifying constraints. In this simple illustration we assume that there exist 2 connected variables that give rise to four bound combinations, that is four possible sign combinations of the qualifying constraints. A particular node in the parallel architecture is responsible for solving the primal problem and preparing all the necessary data for the formulation of the relaxed dual problems. Subsequently, each node, based on the number of connected variables that have been identified, determines whether it is responsible for solving any relaxed dual problems. The next step is, for every node, to proceed on the solution of the relaxed dual problems corresponding to the bound combinations that have been assigned to it. Once the assigned problems have been solved, all the feasible solutions are stored in the local CPU's and only the best lower bound generated at each processing element is being propagated to the "master" node. This issue brings us naturally to the third implementational issue associated with the distributed implementation of the GOP algorithm, that is the routing of the best lower bound.

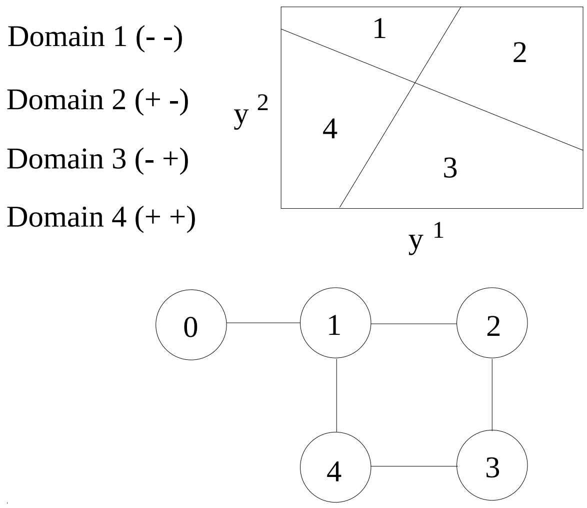


FIG. 2.4. *Parallel Solution of the Relaxed Dual Problems.*

2.2.3. routing of the best lower bound. Poor data communication in parallel architectures can create substantial bottlenecks, thus degrading the overall performance of the algorithm. Based on the previous section it is clear that for the "master" node to proceed with the solution of the next primal problem only information related to the best lower bound is needed. Furthermore, it is rare to envision a situation in which hundreds of processing elements attempt to, almost simultaneously, access a particular node in order to provide certain data. The queuing problems that would arise will be very significant. Therefore, a routing algorithm was implemented which would, in $\lfloor log(P)+1 \rfloor$ steps, where P is the number of nodes, transmit to node 0 the best lower bound. This is described in detail in Androulakis et al. (1996).

2.3. The GOP in bilevel linear and quadratic programming. Visweswaran et al. (1996) addressed bilevel linear and quadratic programming problems using as a basis the key concepts of the GOP and studying further the mathematical structure of such models.

2.3.1. Formulation. Bilevel programming refers to optimization problems in which the constraint region is implicitly determined by another optimization problem, as follows:

$$
\begin{array}{l}
\min\limits_{x} \quad F(x,y) \\
s.t. \\
G(x,y) \leq 0 \\
y \in \left\{ \begin{array}{lll} \min\limits_{y} f(x,y) & & \\ s.t. \quad g(x,y) & \leq & 0 \\ \quad x \in X, \quad y & \in & Y \end{array} \right\}
\end{array}
\tag{P}
$$

where $G(x,y)$ is the vector valued function $X \times Y \rightarrow R^p$, $g(x,y)$ is the vector valued function $X \times Y \rightarrow R^m$, and X and Y are compact convex sets.

Problem (P) can be interpreted in the following way. At the higher level the decision maker (leader) has to choose first a vector $x \in X$ to minimize his objective function F; then in light of this decision the lower level decision maker (follower) has to select the decision vector $y \in Y$ that minimizes his own objective f.

Applications of bilevel programming are diverse, including (i) design optimization problems of chemical plants where regions of different models should be examined (as for example in equilibrium calculations where the different regions correspond to different number and type of phases), (ii) long-range planning problems followed by short-term scheduling in chemical and other industries, (iii) hierarchical decision making policy problems in mixed economies, where policy makers at the top level influence the decisions of private individuals and companies, and (iv) energy consumption of private companies, which is affected by imported resources controlled by government policy.

If all functions are linear, problem (P) gives rise to the following bilevel linear programming formulation:

$$
\begin{aligned}
&\min_{x} \quad F(x,y) = c_1^T x + d_1^T y \\
&s.t. \\
&G(x,y) \le 0 \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{(P2)}\\
&y \in \left\{ \begin{array}{ll} \min_{y} & f(x,y) = c_2^T x + d_2^T y \\ s.t. & g(x,y) = Ax + By - b \;\; \le \;\; 0 \\ & \qquad x \;\ge\; 0 \end{array} \right\}
\end{aligned}
$$

For the sake of simplicity, the constraints $G(x,y)$ will be ignored in the sequel. However, it is easy to show that the results obtained below hold in the presence of general convex constraints at the outer level. It should also be noted that any bounds on y are assumed to be incorporated into the inner level inequality constraints.

Rather than working with problem (P2) in its hierarchical form the analysis begins by converting it into a single mathematical program. This can be achieved by replacing the follower's optimization problem with the necessary and sufficient KKT optimality conditions. This results in the following problem:

$$
\left.
\begin{array}{ll}
 & \min_{x,y,u} \quad c_1^T x + d_1^T y \\
\text{s.t.} & d_2 + u^\top B = 0 \\
 & u_i(Ax + By - b)_i = 0, \;\; i = 1,..,m \\
 & Ax + By \le b \\
 & x \ge 0, \; y \ge 0, \; u_i \ge 0, \; i = 1,..,m
\end{array}
\right\} \quad \text{(P2S)}
$$

where u_i is the Lagrange multiplier of the i^{th} follower's constraint $(Ax + By - b)_i, \;\; i = 1,..,m$. Note that the optimality conditions assume the existence of a stable point for the inner optimization problem, and therefore assume the satisfaction of an appropriate constraint qualification.

Problem (P2S) is a single nonlinear optimization problem, albeit nonconvex due to the presence of bilinear terms in the complementarity conditions. Floudas and Visweswaran (1990, 1993) demonstrated that this class of problems can be solved to global optimality through their primal-dual decomposition algorithm (GOP). Here, by exploiting the special problem structure and introducing extra 0-1 variables to express the tightness of the follower's constraints a modified and more efficient algorithm is developed.

2.3.2. Mathematical properties. Consider the following partition of the variables $Y = u, X = (x,y)$ which satisfies Conditions (A) of the GOP algorithm (Floudas and Visweswaran, 1990, 1993). For fixed $Y = Y^k$, the primal problem can be written as:

$$\left.\begin{array}{ll} & \min\limits_{x,y} \quad c_1^T x + d_1^T y \\ \text{s.t.} & Y_i^k (Ax + By - b)_i = 0, \quad i = 1,..,m \\ & Ax + By \leq b \\ & x \geq 0 \end{array}\right\} \quad (\text{P2S}')$$

Note that the KKT gradient conditions in problem (P2S), which are in the variables u, can be used directly in the dual problem. The solution to this primal problem, if feasible, yields the multipliers λ^k and μ^k for the equality and inequality constraints in (P2S$'$). Note that when $u_i^k = 0$, the corresponding constraint drops out from the set of equality constraints, and there will be no multiplier for that constraint, implying that $\lambda_i^k = 0$ for this case. Conversely, when $u_i^k > 0$, the corresponding constraint is active, and therefore the value of μ_i^k is zero.

Visweswaran et al. (1996) proved the following property :

PROPERTY 2.1. *Suppose that the minimum value of the Lagrange function*
L^*, $L^*(\bar{x}, \bar{y}, u, \mu^k, \lambda^k) = \min\limits_{x,y} \quad L(x, y, u, \mu^k, \lambda^k)$
occurs at $(\bar{x}, \bar{y})$; *then,*

$$L^*(\bar{x}, \bar{y}, u, \mu^k, \lambda^k) \geq \min_{B_j \in CB} \left\{ \begin{array}{l} \sum\limits_{i=1}^{m} [\lambda_i^k (u_i - u_i^k) S_i^{B_j} - (\mu_i^k + \lambda_i^k u_i^k) b_i] \\ \lambda_i^k (u_i - u_i^k) \leq 0, \quad \forall S_i^{B_j} = S_i^L \\ \lambda_i^k (u_i - u_i^k) \geq 0, \quad \forall S_i^{B_j} = S_i^U \end{array} \right\}$$

where $S = (Ax + By - b)$ *are slacks* ($S \geq 0$) *introduced for ease in the presentation;* S_i^L, S_i^U *are the lower and upper bounds on the constraints* $(Ax{+}By{-}b)_i$*, respectively;* B_j *corresponds to a combination of lower/upper bounds of constraints;* S^{B_j} *is the vector of lower/upper bounds of the constraints corresponding to the bound combination* B_j*; and* CB *is the set of all bound combinations.*

The above property preserves the important feature of the GOP algorithm that the solution of problem (RD) can be equivalently substituted by a series of optimization subproblems corresponding to different partitions of the Y-space.

It can be seen that the Lagrange function is essentially expressed in terms of the follower's constraints. This implies that from a computational point of view, the complexity of the relaxed dual problem is determined by the number of active inner problem constraints (i.e., those constraints for which $\lambda_i^k \neq 0$). This can be of great significance in problems with large number of variables but few constraints. For instance, for the case of two x and two y variables with two constraints, the number of subproblems that would be needed is reduced from 2^4 to only 3 (since the combination of the zero upper bounds for all the constraints results in redundant RD subproblem).

2.3.3. Introduction of 0-1 variables. It is clear that each combination of the u variables corresponds to a vertex of the followers feasible region. However, different combinations with the same set of nonzero u_i correspond to the same vertex. It is desirable to avoid such nonzero combinations from being generated more than once. This can be ensured by the introduction of binary variables, as shown below.

Consider the set of binary variables a_i, $i = 1, \ldots, m$, associated with each one of the follower's constraints as follows:

$$a_i = \begin{cases} 1, \text{if constraint } (Ax + By - b)_i \text{ is active} \\ 0, \text{otherwise} \end{cases}$$

The following set of *big-M* constraints are also introduced to establish one-to-one correspondence between the multiplier u_i of constraint i and the corresponding 0-1 variable a_i:

$$(1/M)a_i \leq u_i \leq Ma_i \tag{2.5}$$

Constraint (6) implies that if $a_i = 0 \Rightarrow 0 \leq u_i \leq 0 \Rightarrow u_i = 0$, i.e. the multiplier is also zero, forcing the corresponding constraint to be tight, whereas if $a_i = 1 \Rightarrow (1/M) \leq u_i \leq M$, the associated multiplier has nonzero value implying an inactive constraint.

The incorporation of constraints (6) along with the 0-1 variables a_i into (P2S) results in:

$$\left.\begin{array}{ll} & \min\limits_{x,y,u} \quad c_1^T x + d_1^T y \\ \text{s.t.} & d_2 + u^\top B = 0 \\ & a_i(Ax + By - b)_i = 0, \;\; i = 1, .., m \\ & u_i \leq Ma_i, \;\; i = 1, .., m \\ & a_i \leq Mu_i, \;\; i = 1, .., m \\ & Ax + By \leq b \\ & x \geq 0, \; y \geq 0, \; u \geq 0, \; a_i = \{0-1\} \end{array}\right\} \text{(P3S)}$$

By augmenting the Y-vector to include the 0-1 variables, the following primal problem can be derived for $Y = Y^k = (u^k, a^k)$:

$$\left.\begin{array}{ll} & \min\limits_{x,y,u} \quad c_1^T x + d_1^T y \\ \text{s.t.} & a_i^k(Ax + By - b)_i = 0, \;\; i = 1, .., m \\ & Ax + By \leq b \\ & x \geq 0, \; y \geq 0 \end{array}\right\} \text{(P3S}'\text{)}$$

Property 2.1 can then be recast as follows:

$$L^*(\bar{x},\bar{y},u,\mu^k,\lambda^k) \geq \min_{B_j \in CB} \left\{ \begin{array}{l} \sum_{i=1}^{m}[\lambda_i^k(a_i - a_i^k)S_i^{B_j} - (\mu_i^k + \lambda_i^k a_i^k)b_i] \\ \lambda_i^k(a_i - a_i^k) \leq 0, \quad \forall S_i^{B_j} = S_i^L \\ \lambda_i^k(a_i - a_i^k) \geq 0, \quad \forall S_i^{B_j} = S_i^U \end{array} \right\}$$

Consider the i^{th} term. It is clear that if $a_i^k = 0$, the corresponding constraint would have been absent from the primal problem (P3S$'$), leading to $\lambda_i^k = 0$, so that this term would be absent from the summation. Therefore, only the case of $a_i^k = 1$ is important. Then, since a_i is always less than or equal to a_i^k, the minimum of $L(x,y,a,\mu^k,\lambda^k)$ occurs at the lower (upper) bound of $(Ax + By - b)_i$ if $\lambda_i^k \leq 0 (\lambda_i^k \geq 0)$. Therefore, it is sufficient to set each active constraint in the summation to the appropriate bound, and the following result is always true:

Only one relaxed dual problem is solved at every iteration regardless of the size of the problem.

Another advantage of (PS3) problem formulation is that additional constraints (integer cuts) in the 0-1 variables, a_i, can be used together with the Lagrangian cut to improve the solution efficiency of the resulting MILP relaxed dual problem. In particular, as has been showed by Hansen et al. (1990), in any optimal solution of bilevel programming problem (PS1) the active constraints of the follower's problem satisfy the following conditions:

$$\sum_{i}^{I_p(i)} a_i \geq 1, \text{ if } d_i > 0, \; i = 1, \ldots m$$
$$\sum_{i}^{I_n(i)} a_i \geq 1, \text{ if } d_i < 0, \; i = 1, \ldots m$$

where $I_p(i), I_n(i)$ are the sets of constraints in which variable y_i appears with positive and negative sign, respectively. Also, an active set strategy suggests that:

$$\sum_{i=1}^{m} a_i \leq |y|$$

where $|y|$ is the cardinality of the follower's decision vector y. It can be seen that these and other preprocessing steps can be done on the binary variables to eliminate certain combinations.

Based on the above analysis, a modified algorithm for global optimization of bilevel linear programming programs is outlined in Visweswaran et al. (1996).

2.3.4. Linear-quadratic and quadratic-quadratic bilevel problems. In this section the solution approach is extended to consider the linear/quadratic as well as the quadratic/quadratic bilevel programming problems of the following general form:

$$\begin{array}{l} \min\limits_{x} \quad F(x,y) \\ s.t. \\ y \in \left\{ \begin{array}{ccc} & \min\limits_{y} f(x,y) & \\ s.t. & Ax+By \quad \leq & b \\ & x \geq 0 & \end{array} \right\} \end{array} \qquad (\mathrm{P}')$$

where $F(x,y)$ is a convex function of x and y, and $f(x,y) = d_2 y + x^\top Q_1^2 y + y^\top Q_2^2 y$. For sake of simplicity, it is assumed that $F(x,y) = c_1^T x + d_1^T y$. It can easily be shown, however, that the following analysis is valid for any convex form of $F(x,y)$. It is also assumed that $f(x,y)$ is a convex quadratic function. Then, the KKT conditions for the inner problem are both necessary and sufficient for inner optimality which preserves the equivalence of problems (P′) and (PS′) below:

$$\left. \begin{array}{l} \min\limits_{x,y,u} \quad F(x,y) = c_1^T x + d_1^T y \\ \text{s.t.} \;\; Ax + By \leq b \\ \quad 2y^\top Q_2^2 + x^\top Q_1^2 + u^\top B_2 + d_2 = 0 \\ \quad u_i(Ax+By-b)_i = 0, \;\; i = 1,..,m \\ \quad x \in X, \;\; y \in Y, \;\; u \geq 0 \end{array} \right\} \; (\mathrm{PS}')$$

Introducing the set of 0-1 variables a_i results in the following equivalent formulation:

$$\left. \begin{array}{l} \min\limits_{x,y,a,u} \quad F(x,y) = c_1^T x + d_1^T y \\ \text{s.t.} \;\; Ax + By \leq b \\ \quad 2y^\top Q_2^2 + x^\top Q_1^2 + u^\top B_2 + d_2 = 0 \\ \quad a_i(Ax+By-b)_i = 0, \;\; i = 1,..,m \\ \quad u_i \leq M a_i, \;\; i = 1,..,m \\ \quad a_i \leq M u_i, \;\; i = 1,..,m \\ \quad x \in X, \;\; y \in Y, \;\; a_i = \{0-1\}, \;\; u_i \geq 0, \;\; i = 1,..,m \end{array} \right\} \; (\mathrm{P3S}')$$

As in the linear case, the variables can be partitioned into $Y = (a,u)$, and $X = (x,y)$. Then, for fixed $Y = Y^k$ the primal problem becomes

$$\left. \begin{array}{l} \min\limits_{x,y} \quad F(x,y) = c_1^T x + d_1^T y \\ \text{s.t.} \;\; Ax + By \leq b \\ \quad 2y^\top Q_2^2 + x^\top Q_1^2 + u^{k\top} B_2 + d_2 = 0 \\ \quad a_i^k(Ax+By-b)_i = 0, \;\; i = 1,..,m \\ \quad x \in X, \;\; y \in Y \end{array} \right\} \; (\mathrm{P4S}')$$

Visweswaran et al. (1996) showed that the Lagrange function can be reduced to

$$L(x, y, a, u, \mu^k, \lambda^k, \nu^k) = \sum_{i=1}^{m} \lambda_i^k (a_i - a_i^k)(Ax + By - b)_i + \sum_{i=1}^{n_y} \nu_i^k (u_i - u_i^k) B_i + (c_1^T x + d_1^T y)^k$$

Then, it is obvious that Property 2.1 holds:

$$\min_{x,y} \; L(x, y, a, u, \mu^k, \lambda^k, \nu^k) \geq \sum_{i=1}^{m} \lambda_i^k (a_i - 1)(Ax + By - b)_i^L + \sum_{i=1}^{n_y} \nu_i^k (u_i - u_i^k) B_i + (c_1^T x + d_1^T y)^k$$

and consequently only one relaxed dual subproblem has to be solved per iteration.

Since the stationary conditions for are functions of X and Y variables, they appear to both primal and relaxed-dual subproblems. Moreover, for the case of quadratic outer objective $F(x, y)$ the primal problem corresponds to a nonlinear programming problem. However, under the convexity assumptions it can be solved using a conventional NLP solver.

2.4. New formulations for the GOP approach. Visweswaran and Floudas (1996a,b) introduced new formulations for the GOP which consist of an MILP reformulation of the relaxed dual problems and a linear partitioning of the domain. These developments are discussed in the sections 2.4.1 and 2.4.2.

2.4.1. MILP reformulation of the relaxed duals. The solution of the relaxed dual subproblems at each node is the most time consuming step in the algorithm. The reduction test mentioned can help to prune the branch-and-bound tree at each node; however, it is still necessary to solve a large number of subproblems at each iteration. It is very likely that the solution of most of these subproblems are useless as far as the succeeding iterations are concerned, i.e., most of the nodes will be fathomed as soon as they are spawned. Naturally, this raises the question whether these subproblems can be solved implicitly. This section presents one possible approach for reformulation of the relaxed dual problem at each iteration so that the implicit enumeration of all the solutions can be achieved by solution of an MILP problem.

At the K^{th} iteration, the Lagrange function has the form given by (2.4). Consider the i^{th} term in the summation. In each of the $2^{NI_C^k}$ relaxed dual subproblems, this term takes on either of two values:

$$x_i g_i^K(y) \;=\; \begin{cases} x_i^L g_i^K(y) & \text{if} \quad g_i^K(y) \geq 0 \\ x_i^U g_i^K(y) & \text{if} \quad g_i^K(y) \leq 0 \end{cases}$$

Now, x_i can be implicitly expressed as a combination of its lower and upper bounds:

$$x_i = (1 - \alpha_i^K) x_i^L + \alpha_i^K x_i^U \tag{2.6}$$

where $\alpha_i^K \in \{0, 1\}$.

This leads to the following formulation for the i^{th} term in (2.4):

$$x_i g_i^K(y) \;=\; t_i + x_i^L g_i^K(y)$$

where

$$\begin{aligned} t_i &\geq \alpha_i^K (x_i^U - x_i^L) \underline{g_i^K} \\ t_i &\geq (x_i^U - x_i^L)(g_i^K(y) - (1 - \alpha_i^K)\overline{g_i^K}) \\ \alpha_i^K \underline{g_i^K} &\leq g_i^K(y) \leq (1 - \alpha_i^K)\overline{g_i^K} \end{aligned}$$

where $\underline{g_i^K}$ and $\overline{g_i^K}$ are respectively the lower and upper bounds on the *qualifying* constraints. As the following property shows, this can be used to reformulate the relaxed dual problem as a mixed integer linear program (MILP):

PROPERTY 2.2. *Suppose that, at the K^{th} iteration, C denotes the current node to be partitioned, and R_C denotes the set of constraints defining the region associated with C. Then, the best solution from all the relaxed dual subproblems at this iteration can be obtained as the optimal solution of the following mixed-integer linear program.*

$$\begin{aligned} & \min_{\substack{y \in Y, \mu_B \\ t, \alpha}} \; \mu_B \\ \text{s.t.} \quad \mu_B &\geq \sum_{i=1}^{NI_c^K} t_i^K + \sum_{i=1}^{NI_c^K} x_i^L g_i^K(y) + L_0^K(y, \lambda^K, \mu^K) \\ t_i^K &\geq \alpha_i^K (x_i^U - x_i^L) \underline{g_i^K} \\ t_i^K &\geq (x_i^U - x_i^L)(g_i^K(y) - (1 - \alpha_i^K)\overline{g_i^K}) \\ \alpha_i^K \underline{g_i^K} &\leq g_i^K(y) \leq (1 - \alpha_i^K)\overline{g_i^K} \\ (y, \mu_B) &\in R_C \end{aligned} \tag{2.7}$$

where $\underline{g_i^K}$ and $\overline{g_i^K}$ are the lower and upper bounds on $g_i^K(y)$ over Y.

If $\overline{L_0^K}(y, \lambda^K, \mu^K)$ are convex functions in y, then (2.7) is a convex MINLP, and can be solved with the Generalized Benders Decomposition (Geoffrion (1972), Floudas et al. (1989)) or the Outer Approximation algorithm Duran and Grossmann (1986). The recent book of Floudas (1995) presents a theoretical, algorithmic, and applications oriented exposure of approaches for MINLP problems.

It should be noted that the reduction tests of Section 2 can also be applied to the MILP formulation, as shown by the following property.

PROPERTY 2.3. *At the K^{th} iteration,*

(i) *If $g_i^K(y) \geq 0$ for all y (respectively $g_i^K(y) \leq 0$ for all y) then variable α_i^K can be fixed to 0 (respectively 1.)*

(ii) *If $g_i^K(y) = 0$ for all y then variable α_i^K vanishes from formulation (2.7).*

Backtracking : With the MILP reformulation, it is possible to solve the relaxed dual subproblems implicitly for the best solution at each iteration. However, it is not sufficient to find the best solution; it must also be determined whether any of the other partitions can provide a useful solution for further refinement.

Consider the relaxed dual subproblems solved when node j is being partitioned. Suppose that this node was partitioned during iteration K. Then, there are NI_c^K binary variables, and $2^{NI_c^K}$ partitions to consider. Solving the problem (2.7) gives the best solution among these partitions. Suppose that this solution corresponds to the combination α^C. Suppose also that J_C is the set of binary variables that are equal to 1 in this combination, and that there are NJ_C of them. Consider now the following cut

$$\sum_{i \in J_C} \alpha_i - \sum_{i \notin J_C} \alpha_i \leq NJ_C - 1$$

If problem (2.7) is resolved with the above cut added to the problem, then the solution will have a value for α different from α^C, and will therefore correspond to a different subregion of the current problem. Note that the objective value of this problem represents the "second" best possible solution. The best solution, of course, is the one corresponding to the solution of the first MILP problem, with $\alpha = \alpha^C$. Therefore, this methodology is sufficient to go back to a partitioned node at any point.

Note that although the size of the MILP problems increases slightly at each iteration due to the accumulation of constraints from previous iterations, the number of binary variables present in these problems is equal to the number of *connected* variables for each iteration. In other words, the number of binary variables in the MILP problems is bounded by the number of x variables in the original problem. The detailed GOP/MILP Algorithm description is presented in Visweswaran and Floudas (1996a,b).

REMARK 2.1. After the MILP problem has been solved, an integer cut is added to the corresponding formulation which ensures that that solution cannot be repeated. This implies that the same MILP formulation might be solved several times over the course of the iterations with small differences arising from the additional integer cuts. Subsequently, there is considerable potential for storing the tree information from these problems for use in future iterations.

REMARK 2.2. At each iteration of the algorithm, there is a *single* MILP problem solved as compared to the original algorithm, which needs to solve $2^{NI_c^K}$ subproblems at the K^{th} iteration. The number of binary variables present in any MILP formulation during all the iterations is bounded by the maximum number of x variables. However, it is usually the case that the number of *connected* variables is a fraction of the total number of x variables, implying that the MILP problems are likely to have few binary variables.

REMARK 2.3. The major advantage of the MILP problem appears when there are more than about 15 *connected* variables at any iteration. In such cases, the original algorithm would need to solve over 2 million problems at that iteration, the vast majority of which would never be considered as candidate solutions for further branching. In the case of the MILP algorithm, the implicit enumeration allows for far fewer problems to be solved. The maximum number of MILP problems solved is twice the number of iterations of the algorithm.

2.4.2. A linear branching scheme for the GOP algorithm. In both the GOP and GOP/MILP algorithms, the *qualifying* constraints (i.e., the gradients of the Lagrange function) are used to partition the y-space. The reduction properties presented in Section 2.1.3 can provide a significant reduction in the number of *connected* variables and subsequently the number of partitions. However, in the worst case, the number of subproblems solved still increases exponentially with the number of *connected* variables. It is then natural to ask the following question: *Is it possible to develop a valid lower bound at each iteration using only a linearly increasing number of* relaxed dual *subproblems?* In this section, we present one branching scheme that achieves this goal. This scheme originates from the study of Barmish *et al.* (1995) on the stability of polytopes of matrices of robust control systems.

Reformulation of qualifying constraints

Consider the relaxed dual problem at the k^{th} iteration. This problem has the constraint

$$\mu_B \geq L(x^k, y, \lambda^k, \mu^k) + \sum_{i=1}^{NI_C^k} g_i^k(y) \cdot (x_i - x_i^k).$$

where NI_C^k is the number of *connected* variables at the k^{th} iteration. Rearranging the terms leads to

$$\mu_B \geq L_{new}^k(y) + \sum_{i=1}^{NI_C^k} g_i^k(y) \cdot x_i,$$

where

$$L_{new}^k(y) = L(x^k, y, \lambda^k, \mu^k) + \sum_{i=1}^{NI_C^k} g_i^k(y) \cdot x_i^k$$

is a convex function in y.

Suppose that all the x variables are bounded between -1 and 1. If this is not the case, it can be achieved by use of the following linear transformation. Suppose that $x^L \leq x \leq x^U$. Then, define x' such that $-1 \leq x' \leq 1$, and

$$x = a \cdot x' + b$$

The substitution of the lower and upper bounds gives

$$x^L = a \cdot (-1) + b \ , \quad \text{and} \quad x^U = a \cdot (1) + b$$

leading to

$$a = \frac{x^U - x^L}{2} \ , \quad \text{and} \quad b = \frac{x^U + x^L}{2}$$

The variables x' can then be substituted for x using the above transformation, leading to a Lagrange function in y and x'. We will continue the presentation in this section by considering the case $-1 \leq x \leq 1$.

The following observation is now made:

(a) If $g_i^k(y) \geq 0$,

$$x_i g_i^k(y) \geq x_i^L g_i^k(y) \quad \Rightarrow \quad x_i g_i^k(y) \geq -g_i^k(y)$$

(b) If $g_i^k(y) \leq 0$,

$$x_i g_i^k(y) \geq x_i^U g_i^k(y) \quad \Rightarrow \quad x_i g_i^k(y) \geq g_i^k(y)$$

Combining these two cases leads to the inequality

$$x_i g_i^k(y) \geq -|g_i^k(y)|$$

and

$$\mu_B \geq L_{new}^k(y) - \sum_{i=1}^{NI_C^k} |g_i^k(y)| \tag{2.8}$$

The first term on the right hand side is convex, and can remain unaltered. Consider now the summation term. Using the concept of the infinity norm, (2.8) can be written as

$$\mu_B \geq L_{new}^k(y) - NI_C^k \cdot (\max_i |g_i^k(y)|) \tag{2.9}$$

For any value of y, there is some $j \in 1, \ldots, NI_C^k$ such that

$$|g_j^k(y)| = \max_{i=1,\ldots,NI_C^k} |g_i^k(y)|$$

implying that

$$|g_j^k(y)| \geq |g_i^k(y)|, \quad i = 1, \ldots, NI_C^k \tag{2.10}$$

Consider the following two possibilities:

(a) If $g_j^k(y) \geq 0$, then $|g_j^k(y)| = g_j^k(y)$, and (2.10) reduces to the two inequalities

$$\left.\begin{array}{rcl} g_j^k(y) & \geq & g_i^k(y) \\ g_j^k(y) & \geq & -g_i^k(y) \end{array}\right\} \quad i = 1, \ldots, NI_C^k,\ i \neq j \tag{2.11}$$

and (2.9) becomes

$$\mu_B \geq L_{new}^k(y) - NI_C^k \cdot g_i^k(y)$$

(b) If $g_j^k(y) \leq 0$, then $|g_j^k(y)| = -g_j^k(y)$, and (2.10) reduces to the two inequalities

$$\left.\begin{array}{rcl} g_j^k(y) & \leq & g_i^k(y) \\ g_j^k(y) & \leq & -g_i^k(y) \end{array}\right\} \quad i = 1, \ldots, NI_C^k,\ i \neq j \tag{2.12}$$

and (2.9) becomes

$$\mu_B \geq L_{new}^k(y) + NI_C^k \cdot g_i^k(y)$$

The two cases presented above indicate how the summation in (2.8) can be replaced by a linear term when $g_j^k(y)$ represents the maximum of all the *qualifying* constraints at a given value of y. This concept can then be extended to cover the entire region for y. To do this, the above procedure needs to be repeated for all values of j, resulting in $2 \times NI_C^k$ subproblems that need to be solved in order to properly underestimate the Lagrange function at all values of y.

3. The generalized geometric programming approach, GGP. Maranas and Floudas (1996) introduced a global optimization approach for generalized geometric programming models that have a variety of applications in engineering design and robust control.

Generalized geometric or signomial programming **(GGP)** is the class of optimization problems where the objective function and constraints are the difference of two *posynomials*. A posynomial $G(\mathbf{x})$ is simply the sum of a number of *posynomial terms* or *monomials* $g_k(\mathbf{x})$, $k = 1, \ldots, K$ multiplied by some positive real constants c_k, $k = 1, \ldots, K$.

$$G(\mathbf{x}) \ = \ c_1 g_1(\mathbf{x}) + c_2 g_2(\mathbf{x}) + \cdots + c_K g_K(\mathbf{x})$$

Note that $c_k \in \Re^+$, $k = 1, \ldots, K$. Each monomial $g(\mathbf{x})$ is in turn the product of a number of positive variables each of them raised to sum real power,

$$g(\mathbf{x}) = x_1^{d_1} x_2^{d_2} \cdots x_n^{d^N}$$

where $d_1, d_2, \ldots, d_N \in \Re$. The term geometric programming was adopted because of the key role that the well known arithmetic–geometric inequality played in the initial developments.

By grouping together monomials with identical sign, the generalized geometric **(GGP)** problem can be formulated as the following nonlinear optimization problem:

$$\begin{array}{llrcl}
 & \min\limits_{\mathbf{t}} & G_0(\mathbf{t}) & = & G_0^+(\mathbf{t}) - G_0^-(\mathbf{t}) \\
(\mathbf{GGP}) & \text{s.t.} & G_j(\mathbf{t}) & = & G_j^+(\mathbf{t}) - G_j^-(\mathbf{t}) \le 0, \quad j = 1, \ldots, M \\
 & & t_i & \ge & 0, \quad i = 1, \ldots, N \\
 & \text{where} & G_j^+(\mathbf{t}) & = & \sum\limits_{k \in K_j^+} c_{jk} \prod\limits_{i=1}^{N} t_i^{\alpha_{ijk}}, \quad j = 0, \ldots, M \\
 & & G_j^-(\mathbf{t}) & = & \sum\limits_{k \in K_j^-} c_{jk} \prod\limits_{i=1}^{N} t_i^{\alpha_{ijk}}, \quad j = 0, \ldots, M
\end{array}$$

where $\mathbf{t} = (t_1, \ldots, t_N)$ is the positive variable vector; G_j^+, G_j^-, $j = 0, \ldots, M$ are positive posynomial functions in $\mathbf{t}$; α_{ijk} are arbitrary real constant exponents; whereas c_{jk} are given *positive* coefficients. Finally, sets K_j^+, K_j^- count how many positively/negatively signed monomials form posynomials G_j^+, G_j^- respectively.

3.1. DC transformation. The objective function as well as the constraints in the original formulation **GGP** are in general nonconvex functions. Based on an eigenvalue analysis, it is quite straightforward to show that the Hessian matrices of these nonlinear functions involve eigenvalues of nonconstant sign implying that they are neither convex nor concave. However, by applying the transformation,

$$t_i = \exp z_i, \quad i = 1, \ldots, N$$

on the original formulation **(GGP)** we obtained the following programming problem **(DC)**.

$$\min_{\mathbf{z}} \quad G_0(\mathbf{z}) = G_0^+(\mathbf{z}) - G_0^-(\mathbf{z})$$

$$
\begin{aligned}
\textbf{(DC)} \quad \text{s.t.} \quad & G_0(\mathbf{z}) \;=\; G_j^{+}(\mathbf{z}) \;-\; G_j^{-}(\mathbf{z}) \;\leq\; 0, \quad j = 1, \ldots, M \\
& z_i^L \;\leq\; z_i \;\leq\; z_i^U, \quad i = 1, \ldots, N \\
\text{where} \quad & G_j^{+}(\mathbf{z}) \;=\; \sum_{k \in K_j^{+}} c_{jk} \exp\left\{ \sum_{i=1}^{N} \alpha_{ijk}\, z_i \right\}, \quad j = 0, \ldots, M \\
& G_j^{-}(\mathbf{z}) \;=\; \sum_{k \in K_j^{-}} c_{jk} \exp\left\{ \sum_{i=1}^{N} \alpha_{ijk}\, z_i \right\}, \quad j = 0, \ldots, M
\end{aligned}
$$

3.2. Lower bounding. A lower bound on the solution of problem **(DC)** can be obtained by solving a convex relaxation of the original problem **(DC)**. Such a convex relaxation can be realized by underestimating every concave function, $-G_j^{-}(\mathbf{z})$ with a linear function $-L_j^{-}(\mathbf{z})$ for every $j = 0, \ldots, M$. This linear function is constructed by underestimating every implicitly separable term $-\exp\left\{\sum_{i=1}^{N} \alpha_{ijk}\, z_i\right\}$ with a linear function. This defines the following relaxed convex programming problem **(R)** whose solution provides a lower bound on the solution of **(DC)**.

$$
\begin{aligned}
\min_{\mathbf{z}} \quad & G_0^{conv}(\mathbf{z}) \;=\; G_0^{+}(\mathbf{z}) \;-\; L_0^{-}(\mathbf{z}) \\
\text{s.t.} \quad & G_j^{conv}(\mathbf{z}) \;=\; G_j^{+}(\mathbf{z}) \;-\; L_j^{-}(\mathbf{z}) \;\leq\; 0, \quad j = 1, \ldots, M \\
& z_i^L \;\leq\; z_i \;\leq\; z_i^U, \quad i = 1, \ldots, N \\
\text{where} \quad & G_j^{+}(\mathbf{z}) \;=\; \sum_{k \in K_j^{+}} c_{jk} \exp\left\{ \sum_{i=1}^{N} \alpha_{ijk}\, z_i \right\}, \quad j = 0, \ldots, M \\
& L_j^{-}(\mathbf{z}) = \sum_{k \in K_j^{-}} c_{jk} \left\{ A_{jk} + B_{jk} \left(\sum_{i=1}^{N} \alpha_{ijk} z_i \right) \right\}, \; j = 0, ., M \\
\text{and} \quad & A_{jk} \;=\; \frac{Y_{jk}^U \exp\left(Y_{jk}^L\right) - Y_{jk}^L \exp\left(Y_{jk}^U\right)}{Y_{jk}^U - Y_{jk}^L}, \\
& B_{jk} \;=\; \frac{\exp\left(Y_{jk}^U\right) - \exp\left(Y_{jk}^L\right)}{Y_{jk}^U - Y_{jk}^L}, \\
& Y_{jk}^L \;=\; \sum_{i=1}^{N} \min\left(\alpha_{ijk} z_i^L, \alpha_{ijk} z_i^U\right),
\end{aligned}
$$

$$Y_{jk}^U = \sum_{i=1}^N \max\left(\alpha_{ijk} z_i^L, \alpha_{ijk} z_i^U\right)$$

Note that the linear underestimator $-L_j^-(\mathbf{z})$ of $-G_j^-(\mathbf{z})$ is composed by the sum of a number of linear functions each one of which is lower bounding an implicitly univariate function of the form $-\exp(Y)$. Clearly, the smaller the difference between the original functions $G_j^-(\mathbf{z})$ and the linearizations $L_j^-(\mathbf{z})$ the closer the solution of **(R)** will be to the solution of **(DC)**. The quality of this lower bounding can be analyzed by examining the tightness of underestimation of every concave term of the form $-\exp(Y)$ with a linear function inside some interval $[Y^L, Y^U]$. Let $\Delta(Y)$ be the separation between the concave function $-\exp(Y)$ and the linear underestimator inside the interval $[Y^L, Y^U]$

$$\Delta(Y) = -\exp(Y) - \left(\frac{Y^U \exp(Y^L) - Y^L \exp(Y^U)}{Y^U - Y^L} + \frac{\exp(Y^U) - \exp(Y^L)}{Y^U - Y^L} Y\right)$$

This separation function $\Delta(Y)$ is concave in Y and it reaches its single maximum at

$$Y^* = \log\left(\frac{\exp(Y_{jk}^U) - \exp(Y_{jk}^L)}{Y_{jk}^U - Y_{jk}^L}\right)$$

with a value

$$\Delta_{max} = \exp(Y^L)\left(1 - Z + Z\log(Z)\right)$$

$$\text{where} \quad Z = \frac{\exp(\delta) - 1}{\delta}, \quad \delta = Y^U - Y^L$$

Note that as the interval width $\delta = Y^U - Y^L$ goes to zero, Z approaches one and therefore the maximum separation goes to zero.

$$\delta \longrightarrow 0, \;\; Z \longrightarrow 1, \text{ and } \Delta_{max} \longrightarrow 0.$$

The rate at which this maximum separation goes to zero can be determined by Taylor expanding $\Delta_{max}(\delta)$ at $\delta = 0$.

$$\frac{\Delta_{max}}{\exp(Y^L)} = \frac{\delta^2}{8} + \frac{\delta^3}{16} + \frac{11\delta^4}{576} + \frac{5\delta^5}{1152} + \frac{41\delta^6}{51840} + \frac{5\delta^7}{41472} + \mathcal{O}(\delta^8)$$

By considering only the first leading term of the positive termed series expansion we deduce that the rate at which Δ_{max} approaches zero as δ goes to zero is

$$\Delta_{max} \approx \mathcal{O}(\delta^2), \;\; \text{as } \delta \longrightarrow 0.$$

On the other hand, as δ goes to infinity Δ_{max} goes to infinity as well. By considering only the leading term in the expression for $\Delta_{max}(\delta)$ we conclude that Δ_{max} goes to infinity as

$$\Delta_{max} \approx \mathcal{O}(\exp(\delta)), \;\; \text{as } \delta \longrightarrow +\infty.$$

3.3. Scaling of variables. This objective can be accomplished by first scaling all variables t_i in the original formulation **(GGP)** and then employing the exponential transformation. Such a scaling is the following

$$t_i \longleftarrow t_i^L + \frac{t_i^U - t_i^L}{t_i^{U,new} - t_i^{L,new}} \left(t_i^{new} - t_i^{L,new} \right), \quad i = 1, \ldots, N$$

where $t_i^{L,new}, t_i^{U,new}$ are selected so as $\log(t_i^{U,new}) - \log(t_i^{L,new})$ is small. Maranas and Floudas (1996) present (i) different ways of transforming the inequalities, (ii) reduction approaches of the partitioned domains, (iii) the use of monotonicity analysis for variable elimination, (iv) a complete description of the algorithmic steps, and (v) a proof of convergence to an ϵ-global solution.

4.. Global optimization for general NLPs. In this section, we will discuss global optimization methods for general twice differentiable nonlinear programming problems.

4.1. Existence theorem. A very important theoretical advance has been made by Liu and Floudas (1993) who showed that the GOP can be applied to very general classes of NLPs defined as:

$$\min_{x \in X} \; F(x)$$

$$subject\ to \quad G(x) \;\leq\; 0$$

where X is a non empty, compact, convex set in R^n , and the functions $F(x)$, $G(x)$ are C^2 continuous on X. This result is very significant because it extends the classes of mathematical problems that the GOP can be applied from polynomials or rational polynomials to arbitrary nonlinear objective function and constraints that may include exponential terms and trigonometric terms with the only requirement that these functions have continuous first and second order derivatives.

4.2. The αBB approach for general NLPs. A novel branch and bound global optimization approach which combines a special type of difference of convex functions' transformation with lower bounding understimating functions was proposed by Maranas and Floudas (1994 a,b). Androulakis et al. (1995) extended this approach to address the general class of nonlinear constrained optimization problems noted in section 4.1.

4.2.1. Formulation. The formulation of such general nonlinear optimization problems is :

$$\textbf{(P0)} \qquad \min_{\mathbf{x}} \quad f(\mathbf{x})$$

$$
\begin{aligned}
\text{s.t.} \quad h_j(\mathbf{x}) &= 0, \;\; j = 1, \ldots, M \\
g_k(\mathbf{x}) &\leq 0, \;\; k = 1, \ldots, K \\
A\mathbf{x} &\leq \mathbf{c} \\
\mathbf{x}^L &\leq \mathbf{x} \leq \mathbf{x}^U
\end{aligned}
$$

Here $\mathbf{x}$ denotes the vector of variables, $f(\mathbf{x})$ is the nonlinear objective function, $h_j(\mathbf{x})$ is the set of nonlinear equality constraints and $g_k(\mathbf{x})$, $k = 1, \ldots, K$ is the set of nonlinear inequality constraints. Formulation **(P0)** in general corresponds to a nonconvex optimization problem possibly involving multiple local and disconnected feasible regions. The αBB approach is based on the convex relaxation of the original nonconvex formulation **(P0)**. This requires the convex lower bounding of all nonconvex expressions appearing in **(P0)**. These terms can be partitioned into three classes:

(i) convex,
(ii) nonconvex of special structure,
(iii) nonconvex of generic structure.

Clearly, no convex lower bounding action is required for convex functions. For nonconvex terms of special structure (e.g. bilinear, univariate concave functions), tight specialized convex lower bounding schemes already exist and therefore can be utilized. Based on this partitioning of different terms appearing in the objective function and constraints, formulation **(P0)** is rewritten equivalently as follows:

$$
\begin{aligned}
(\mathbf{P}) \;\; \min_{\mathbf{x}} \quad & C^0(\mathbf{x}) + \sum_{k \in \mathcal{K}^0} NC_k^0(\mathbf{x}) + \sum_{i=1}^{N-1} \sum_{i'=i+1}^{N} b_{i,i'}^0 x_i x_{i'} \\
\text{s.t.} \quad & C^j(\mathbf{x}) + \sum_{k \in \mathcal{K}^j} NC_k^j(\mathbf{x}) + \sum_{i=1}^{N-1} \sum_{i'=i+1}^{N} b_{i,i'}^j x_i x_{i'}, \quad \leq 0 \\
& j = 1, \ldots, (2M + K) \\
& A\mathbf{x} = \mathbf{c}, \quad \mathbf{x}^L \leq \mathbf{x} \leq \mathbf{x}^U \\
\text{where} \quad & NC_k^j(\mathbf{x}) \text{ with } \mathbf{x} \in \left\{ x_i \;:\; i \in \mathcal{N}_k^j \right\}, \;\; j = 0, \ldots, (2M + K)
\end{aligned}
$$

Note that all nonlinear equality constraints $h_j(\mathbf{x}) = 0$ appearing in **(P0)** have been replaced by two inequalities in **(P)**. $C^0(\mathbf{x})$ is the convex part of the objective function; $NC_k^0(\mathbf{x})$ is the set of $\mathcal{K}^0$ generic nonconvex terms appearing in the objective function; $\mathcal{N}_k^0$ is the subset of variable $\mathbf{x}$ participating in each generic nonconvex term k in the objective; and $b_{i,i'}^0 x_i x_{i'}$

the bilinear terms. Similarly, for each constraint j, there exists a convex part $C^j(\mathbf{x})$, $\mathcal{K}^j$ generic nonconvex terms $NC_k^j(\mathbf{x})$, with $\mathcal{N}_k^j$ variables $\mathbf{x}$ per term, and the bilinear terms $b_{i,i'}^j x_i x_{i'}$. Additionally, linear equality constraints and variable bounding constraints appear explicitly in the model **(P)**. Clearly, for each optimization problem that falls within formulation **(P0)** there exist several ways of reformulating it into **(P)**. In the current implementation of αBB the only nonconvex terms recognized as having special structure are the bilinear terms. Work is currently under way to include in the set of nonconvex terms of special structure additional nonconvex functions such as univariate concave, signomial functions, and products of univariate functions Maranas and Floudas (1995). In the next section the derivation of a convex relaxation **(R)** of **(P)** is discussed.

4.2.2. Convex relaxation. A convex relaxation of **(P)** can be constructed by replacing each generic nonconvex term, $NC_k^j(\mathbf{x})$, and each bilinear term, $b_{i,i'}^j x_i x_{i'}$, $j = 0, \ldots, (2M + K)$, with one or more convex lower bounding functions.

Nonconvex terms of special structure :

As it is shown in Al-Khayyal and Falk (1983), the tightest possible convex lower bounding of a bilinear term $b_{i,i'} x_i x_{i'}$ inside some rectangular domain $\left[x_i^L, x_i^U\right] \times \left[x_{i'}^L, x_{i'}^U\right]$ (convex envelope) corresponds to the maximum of the following two linear cuts.

$$\begin{aligned} b_{i,i'} x_i x_{i'} \;\geq\; s_{i,i'}(x_i, x_{i'}) &= \max\left(Y_i^L x_{i'} + Y_{i'}^L x_i - Y_i^L Y_{i'}^L,\right. \\ &\qquad \left. Y_i^U x_{i'} + Y_{i'}^U x_i - Y_i^U Y_{i'}^U\right) \end{aligned}$$

$$\text{where} \qquad \begin{aligned} Y_i^L &= \min\left(b_{i,i'} x_i^L, b_{i,i'} x_i^U\right), \\ Y_{i'}^L &= \min\left(b_{i,i'} x_{i'}^L, b_{i,i'} x_{i'}^U\right), \\ Y_i^U &= \max\left(b_{i,i'} x_i^L, b_{i,i'} x_i^U\right), \\ Y_{i'}^U &= \max\left(b_{i,i'} x_{i'}^L, b_{i,i'} x_{i'}^U\right) \end{aligned}$$

$s_{i,i'}(x_i, x_{i'})$ is the convex envelope of $b_{i,i'} x_i x_{i'}$ inside the rectangular domain $\left[x_i^L, x_i^U\right] \times \left[x_{i'}^L, x_{i'}^U\right]$ and therefore, it can become arbitrarily close to $b_{i,i'} x_i x_{i'}$ for a small enough rectangular domain.

It can be shown that the maximum separation between $b_{i,i'} x_i x_{i'}$ and $s_{i,i'}$ inside the domain $\left[x_i^L, x_i^U\right] \times \left[x_{i'}^L, x_{i'}^U\right]$ can be at most one fourth of the area of the rectangular domain multiplied by the absolute value of $b_{i,i'}$:

$$|b_{i,i'}| \, \frac{\left(x_i^U - x_i^L\right)\left(x_{i'}^U - x_{i'}^L\right)}{4}.$$

Lemma 1 : The maximum separation of the bilinear term $x\, y$ from its convex envelope,

$\max[(x^L y + x y^L - x^L y^L),\ (x^U y + x y^U - x^U y^U)]$, inside the rectangle $\left[x^L, x^U\right] \times \left[y^L, y^U\right]$ occurs at the middle point :

$$x^m = \frac{x^L + x^U}{2}, \quad y^m = \frac{y^L + y^U}{2}$$

and is equal to one fourth of the area of the rectangular domain.

Nonconvex terms of generic structure :

The convex lower bounding of the generic nonconvex terms NC_k^j is motivated by the approach introduced in Maranas and Floudas (1994a). For each one of the generic nonconvex functions,

$$\begin{aligned} & NC_k^j(\mathbf{x}), \;\; j = 0, \ldots, (2M+K), \;\; k \in \mathcal{K}^j \\ \text{where} \quad & NC_k^j(\mathbf{x}) \text{ with } \mathbf{x} \in \left\{x_i \; : \; i \in \mathcal{N}_k^j\right\}, \;\; j = 0, \ldots, (2M+K) \end{aligned}$$

a convex lower bounding function $NC_k^{j,conv}$ can be defined by augmenting the original nonconvex expression with the addition of a separable convex quadratic function of $(x_i, \; i \in \mathcal{N}_k^j)$.

$$\begin{aligned} & NC_k^{j,conv}(\mathbf{x}) \;=\; NC_k^j(\mathbf{x}) \\ & + \sum_{i \in \mathcal{N}_k^j} \alpha_{i,k}^j(\mathbf{x}^L, \mathbf{x}^U)\left(x_i^L - x_i\right)\left(x_i^U - x_i\right), \;\; j = 0, \ldots, (2M+K), \;\; k \in \mathcal{K}^j \end{aligned}$$

$$\text{where} \quad \alpha_{i,k}^j(\mathbf{x}^L, \mathbf{x}^U) \;\geq\; \max\left\{0, -\frac{1}{2} \min_{\mathbf{x}^L \leq \mathbf{x} \leq \mathbf{x}^U} \lambda(\mathbf{x})\right\}$$

Note that $\alpha_{i,k}^j$ are nonnegative parameters which must be greater or equal to the negative one half of the minimum eigenvalue of the Hessian matrix of $NC_k^{j,conv}$ over $x_i^L \leq x_i \leq x_i^U, \; i \in \mathcal{N}_k^j$. These parameters $\alpha_{i,k}^j$ can be estimated either through the solution of an optimization problem or by using the concept of the measure of a matrix. The effect of adding the extra separable quadratic term on the generic nonconvex terms is to construct new convex functions by overpowering the nonconvexity characteristics of the original nonconvex terms with the addition of the terms $2\alpha_{i,k}^j$ to all of their eigenvalues. These new functions $NC_k^{j,conv}$ defined over the rectangular domains $x_i^L \leq x_i \leq x_i^U, \; i \in \mathcal{N}_k^j$ involve a number of important properties. These properties are as follows:

Property 1: $NC_k^{j,conv}$ is a valid *underestimator* of NC_k^j.

$$\forall \; x_i \in \left[x_i^L, x_i^U\right], \; i \in \mathcal{N}_k^j \text{ we have } NC_k^{j,conv}(\mathbf{x}) \;\leq\; NC_k^j(\mathbf{x}).$$

Property 2: $NC_k^{j,conv}(\mathbf{x})$ matches NC_k^j at all corner points.

Property 3: $NC_k^{j,conv}(\mathbf{x})$ is *convex* in $x_i \in \left[x_i^L, x_i^U\right], \; i \in \mathcal{N}_k^j$.

Property 4: The maximum separation between the nonconvex term of generic structure $NC_k^{j,conv}$ and its convex relaxation NC_k^j is *bounded* and proportional to the positive parameters $\alpha^j_{i,k}$ and to the square of the diagonal of the current box constraints.

$$\begin{aligned} \max_{\mathbf{x}^L \leq \mathbf{x} \leq \mathbf{x}^U} \quad & \left(NC^j_k(\mathbf{x}) - NC^{j,conv}_k(\mathbf{x}) \right) \\ = \quad & \frac{1}{4} \sum_{i \in \mathcal{N}^j_k} \alpha^j_{i,k}(\mathbf{x}^L, \mathbf{x}^U) \left(x^U_i - x^L_i \right)^2 \end{aligned}$$

Property 5: The underestimators constructed over supersets of the current set are always *less tight* than the underestimator constructed over the current box constraints for every point within the current box constraints. Clearly, the smaller the values of the positive parameters $\alpha^j_{i,k}$, the narrower the separation between the original nonconvex terms and their respective convex relaxations will be. Therefore fewer iterations will also be required for convergence. To this end, customized α parameters are defined for each variable, term and constraint. Furthermore, an updating procedure for the α's as the size of the partition elements decreases is currently under investigation.

This type of convex lower bounding is utilized for nonconvex functions which lack any specific structure that might enable the construction of customized convex lower bounding functions. Clearly, the α–based convex lower bounding can be applied to bilinear terms as well without having to introduce additional variable and constraints. However, in this case the maximum separation will be larger than the one based on the linear cuts. More specifically, the maximum separation for the α convex lower bounding scheme is,

$$\frac{\left(x^U - x^L \right)^2 + \left(y^U - y^L \right)^2}{8}.$$

This is always greater than

$$\frac{\left(x^U - x^L \right) \left(y^U - y^L \right)}{4}$$

unless $x^U - x^L = y^U - y^L$. Based on the aforementioned convex lower bounding procedures for bilinear terms and generic nonconvex terms, a convex relaxation **(R)** of **(P)** is proposed.

$$\begin{aligned} (\mathbf{R}) \min_{\mathbf{x}} \quad & C^0(\mathbf{x}) + \sum_{k \in \mathcal{K}^0} NC^0_k(\mathbf{x}) \\ & + \sum_{i \in \mathcal{N}^0_k} \alpha^0_{i,k}(\mathbf{x}^L, \mathbf{x}^U) \left(x^L_i - x_i \right) \left(x^U_i - x_i \right) + s^0_{i,i'} \\ \text{s.t.} \quad & C^j(\mathbf{x}) + \sum_{k \in \mathcal{K}^j} NC^j_k(\mathbf{x}) \end{aligned}$$

$$+\alpha^j_{i,k}(\mathbf{x}^L, \mathbf{x}^U) \sum_{i \in \mathcal{N}^j_k} \left(x^L_i - x_i\right)\left(x^U_i - x_i\right) + s^j_{i,i'} \ \le \ 0,$$
$$j = 1, \ldots, (2M + K)$$

$$\begin{aligned} s^j_{i,i'} \ \ge \ & \max\Big(Y^{j,L}_i x_{i'} + Y^{j,L}_{i'} x_i - Y^{j,L}_i Y^{j,L}_{i'}, \\ & Y^{j,U}_i x_{i'} + Y^{j,U}_{i'} x_i - Y^{j,U}_i Y^{j,U}_{i'}\Big), \quad j = 0, \ldots, (2M+K) \end{aligned}$$

where
$$\begin{aligned} Y^{j,L}_i &= \min\left(b^j_{i,i'} x^L_i, b^j_{i,i'} x^U_i\right), \\ Y^{j,L}_{i'} &= \min\left(b^j_{i,i'} x^L_{i'}, b^j_{i,i'} x^U_{i'}\right), \\ Y^{j,U}_i &= \max\left(b^j_{i,i'} x^L_i, b^j_{i,i'} x^U_i\right), \\ Y^{j,U}_{i'} &= \max\left(b^j_{i,i'} x^L_{i'}, b^j_{i,i'} x^U_{i'}\right) \end{aligned}$$

$$A\mathbf{x} \ = \ \mathbf{c}, \quad \mathbf{x}^L \ \le \ \mathbf{x} \ \le \ \mathbf{x}^U$$

and
$$NC^j_k(\mathbf{x}) \text{ with } \mathbf{x} \ \in \ \left\{x_i \ : \ i \in \mathcal{N}^j_k\right\}, \ \ j = 0, \ldots, (2M+K)$$

Formulation **(R)** is a convex programming problem whose global minimum solution can be routinely found with existing local optimization solvers such as MINOS5.4. Formulation **(R)** is a relaxation of **(P)** and therefore its solution is a valid lower bound on the global minimum solution of **(P)**.

In the next section, we will see how this convex lower bounding formulation **(R)** can be utilized in a branch and bound framework for locating the global minimum solution of **(P)**.

4.2.3. Global optimization algorithm, αBB. A global optimization procedure, αBB, is proposed for locating the global minimum solution of **(P)** based on the refinement of converging lower and upper bounds. Lower bounds are obtained through the solution of convex programming problems **(R)** and upper bounds based on the solution of **(P)** with local methods.

As it has been discussed in the previous subsection, the maximum separation between the generic and bilinear nonconvex terms and their respective convex lower bounding functions is bounded. For the generic nonconvex terms this maximum separation is proportional to the square of the diagonal of the rectangular partition element and for the bilinear terms proportional to the area of the rectangular domain. Furthermore, as the size of the rectangular domains approaches zero, these maximum separations go to zero as well. This implies that as the current box constraints $[\mathbf{x}^L, \mathbf{x}^U]$ collapse into a point; (i) the maximum separation between the original objective function of $(\mathbf{P})$ and its convex relaxation in $(\mathbf{R})$ becomes zero; and (ii) by the same argument, the maximum separation between the original constraint set in $(\mathbf{P})$ and the one in $(\mathbf{R})$ goes to zero as well. This implies that for every positive number ϵ_f and $\mathbf{x}$ there always exists a positive number δ such that by reducing the rectangular region $[\mathbf{x}^L, \mathbf{x}^U]$ around $\mathbf{x}$ so as $||\mathbf{x}^U - \mathbf{x}|| \leq \delta$ differences between the feasible region of the original problem $(\mathbf{P})$ and its convex relaxation $(\mathbf{R})$ become less than ϵ_f. Therefore, any feasible point $\mathbf{x}^c$ of problem $(\mathbf{R})$ (even the global minimum solution) becomes at least ϵ_f–feasible for problem $(\mathbf{P})$ by sufficiently tightening the bounds on $\mathbf{x}$ around this point.

The next step, after establishing an upper and a lower bound on the global minimum, is to refine them. This is accomplished by successively partitioning the initial rectangular region into smaller ones. The number of variables along which subdivision is required is equal to the number of variables $\mathbf{x}$ participating in at least one nonconvex term in formulation $(\mathbf{P})$. The partitioning strategy involves the successive subdivision of a rectangle into two subrectangles by halving on the middle point of the longest side of the initial rectangle (bisection). Therefore, at each iteration a lower bound of the objective function of $(\mathbf{P})$ is simply the minimum over all the minima of problem $(\mathbf{R})$ in every subrectangle composing the initial rectangle. Therefore, a straightforward (bound improving) way of tightening the lower bound is to halve at each iteration, only the subrectangle responsible for the infimum of the minima of $(\mathbf{R})$ over all subrectangles, according to the rules discussed earlier. This procedure generates a *nondecreasing* sequence for the lower bound. An *nonincreasing* sequence for the upper bound is derived by solving locally the nonconvex problem $(\mathbf{P})$ and selecting it to be the minimum over all the previously recorded upper bounds. Clearly, if the single minimum of $(\mathbf{R})$ in any subrectangle is greater than the current upper bound we can safely ignore this subrectangle because the global minimum of $(\mathbf{P})$ cannot be situated inside it (fathoming step).

Because the maximum separations between nonconvex terms and their respective convex lower bounding functions are bounded and continuous functions of the size of rectangular domain, arbitrarily small ϵ_f feasibility and ϵ_c convergence tolerances are reached for a finite size partition element.

The basic steps of the αBB global optimization algorithm are described in Androulakis et al. (1995). A mathematical proof that the αBB global optimization algorithm converges to the the global minimum is based on the analysis of standard deterministic global optimization algorithms as shown in Maranas and Floudas (1994a,b).

5. Computational studies and application areas.

5.1. The GOP and its variants. Visweswaran and Floudas (1996b) provided a complete implementation of the new versions of the GOP algorithm, including reduction tests and local enhancements at each node of the tree. Detailed computational results of applying the resulting implementation to various classes of nonconvex optimization problems, were also presented. In the following, we will present a summary of the computational studies performed employing the GOP and its variants in a number of application areas in process synthesis, and design, as well as in concave and indefinite quadratic programming problems.

5.1.1. Implementation of the GOP and its variants. The **cGOP** package is written entirely in the C programming language, and consists of approximately 8000 lines of source code, of which around 30% are comments. The algorithms can be called either in standalone mode or as subroutines from within another program. The primal and relaxed dual subproblems are solved either using CPLEX (for linear or mixed integer linear) problems or MINOS for nonlinear problems. Various options are available to change the routines that are used, such as obtaining tighter bounds on the x variables and $g_i^k(y)$ (the gradients of the Lagrange function), as well as solving the full problem as a local optimization problem at each node.

5.1.2. Data structures. Since the **cGOP** package is written in C, it is highly convenient to aggregate the data transfer from one routine to another using structures (equivalent to COMMON blocks in Fortran). The primary data structures used in the package describe the problem data, the solutions of the various primal problems, the data for the various Lagrange functions, and the solutions of the relaxed dual subproblems at each iteration.

The most important group of data is obviously the problem data itself. In order to facilitate easy and general use of this data, the implementation was written assuming that the following types of problems would be solved:

$$\begin{array}{lll} \min\limits_{x,y} & c_0^T x + d_0^T y + x^T Q_0 y + F_0(x) + G_0(y) & \\ s.t. & l_j \le c_j^T x + d_j^T y + x^T Q_j y \le u_j, & i = 1, \ldots, M_1 \\ & F_j(x) + G_j(y) \le u_j & i = M_1 + 1, \ldots, M_2 \\ & L \le \begin{pmatrix} x \\ y \end{pmatrix} \le U & \end{array}$$

where $j = 1, \ldots, M_1$ are the set of bilinear constraints, and $j = M_1 + 1, \ldots, M_2$ are the set of general nonlinear constraints. It is assumed that the functions $F_j(x)$ and $G_j(y)$ are convex in x and y respectively. Note also that while the bilinear constraints can be equalities or inequalities, the other nonlinear terms in the constraints are assumed to lie in convex inequalities.

Given the above formulation, the data for the problem can be separated into one part containing the linear and bilinear terms, and another part containing the nonlinear terms $F_i(x)$ and $G_i(y)$. The first part can be

TABLE 5.1
Heat Exchanger Network and Nonsharp Separation Design Problems

Problem Name	*Problem Size*		*GOP Algorithm*	
	Variables	Constraints	Iterations	CPU (sec)
HEN-1	12	13	4	0.09
HEN-2	12	13	3	0.06
HEN-3	11	9	3	0.10
HEN-4	11	9	8	0.20
HEN-5	26	30	4	0.11
HEN-6	17	13	11	0.54
HEN-7	27	19	39	54.62
SEP-1	38	32	17	3.84

specified through a data file or as arguments during the subroutine call that runs the algorithm. The nonlinear terms, which in general cannot be specified using data files, can be given through user defined subroutines that compute the contribution to the objective function and constraints from these terms, as well as their contribution to the Hessian of the objective function and the Jacobian of the constraints. The detailed structure of the cGOP is presented in Visweswaran and Floudas (1996b).

5.1.3. Computational results. In this section, we present the results of the application of the **cGOP** package to various problems in chemical engineering design and control and mathematical programming.

Table 5.1 presents the computational results for heat exchanger network problems with linear cost functionals, nonlinear costs, and nonsharp separation problems. The first five examples have linear cost functionals are taken from Quesada and Grossmann (1993), the sixth and seventh have nonlinear costs and bilinear constraints and are taken from Floudas and Ciric (1989), while the last one corresponds to nonsharp separation sequencing and is taken from Floudas and Aggarwal (1990).

Tables 5.2, 5.3, and 5.4 present the computational results on pooling problems. Table 5.2 addresses the Haverly pooling problems. Table 5.3 the pooling problems studied by Ben-Tal and Gershovitz (1994), while Table 5.4 addresses the multiperiod tankage design problem of Visweswaran and Floudas (1990). Three cases of the pooling problem have been solved using the GOP and GOP/MILP algorithms. The data for these three cases, as well as the average number of iterations required by the algorithms to converge, are given in Table 5.2. It can be seen that in all cases, the algorithms require less than 15 iterations to identify and converge to the global solution.

TABLE 5.2
Data and results for the Haverly Pooling Problem

Case	Bounds		Cost of	Optimal Solution		GOP Algorithm		GOP/MILP	
	x^U	y^U	B	$f^\star$	$p^\star$	Iter.	CPU	Iter.	CPU
I	100	200	\$16	-\$400	1.0	12	0.22	12	0.49
II	600	200	\$16	-\$600	3.0	12	0.21	12	0.45
III	100	200	\$13	-\$750	1.5	14	0.26	14	0.56

TABLE 5.3
Pooling Problems From Ben-Tal and Gershovitz (1994)

Problem No.	*Problem Size*				*GOP Algorithm*	
	I	J	K	L	Iterations	CPU (HP730)
1.	4	2	1	1	7	0.95
2.	5	5	2	1	41	5.80

TABLE 5.4
Multiperiod Tankage Quality Problem

Starting Point	*Original GOP*			*GOP/MILP*	
(y)	Iter.	Subproblems	CPU	Iter	CPU
Lower bound	8	18	3.66	7	14.7
Upper bound	9	19	3.68	9	13.1
$q_{t1} = 100, q_{t2} = 70$	11	18	3.95	13	22.4
$q_{t1} = 80, q_{t2} = 100$	9	19	3.23	13	16.5

Table 5.5 presents computational results for phase and chemical equilibrium problems which are of crucial importance in several process separation applications. For conditions of constant pressure and temperature, a global minimum of the Gibbs free energy function describes the equilibrium state. Moreover, the Gibbs tangent plane criterion can be used to test the intrinsic thermodynamic stability of solutions obtained via the minimization of the Gibbs free energy. Simply stated, this criterion seeks the minimum of the distance between the Gibbs free energy function at a given point and the tangent plane constructed from any other point in the mole fraction space. If the minimum is positive, then the equilibrium solution is stable.

The GOP algorithm was applied to solve several problems of the phase stability criterion and the results are shown in Table 5.5. In Table 5.5, a comparison with the results obtained by the specialized implementation GLOPEQ of McDonald and Floudas (1996) is also presented.

TABLE 5.5
Results for the Phase Stability Problem

Problem	*Problem Size*			*GOP*		*GLOPEQ**	
Name	N_X	N_Y	N_C	Iterations	CPU (sec)	Iterations	CPU (sec)
BAW2L	2	2	3	27	0.68	32	0.15
BAW2G	2	2	3	30	0.75	36	0.16
TWA3T	6	3	4	13	0.86	16	0.22
TWA3G	6	3	4	121	9.00	85	0.96
PBW3T1	6	3	4	82	6.33	53	0.63
PBW3G1	6	3	4	393	35.21	213	2.37
PBW3T6	6	3	4	1366	134.99	549	4.98
PBW3G6	6	3	4	1886	207.19	757	7.09

Tables 5.6 and 5.7 present the computational results for concave and indefinite quadratic programming problems that were constructed employing the approach of Phillips and Rosen (1988).

In all the cases, it can be seen that the algorithm generally requires very few iterations for the upper and lower bounds to be within 10% of the optimal solution; generally, the convergence to within 1% is achived in a few more iterations. Moreover, certain trends are noticeable in all cases. For example, as the number of constraints (m) grows, the problems generally become easier to solve. Conversely, as the size of the linear variables (k) increases, the algorithm requires more time for the solution of the dual problems, leading to larger overall CPU times.

5.2. The parallel GOP. Androulakis et al. (1996) proposed a distributed implementation of the GOP algorithm whose key issues are summarized in section 2.2. The parallel GOP was implemented on an Intel-Paragon machine. In this section, we will discuss the results of the parallel

Table 5.6
Concave Quadratic Problems from Phillips and Rosen (1988)

Run	Problem size			Iterations	CPU (sec)	
	m	n	k		GOP	GOP/MILP
CLR1	50	50	50	2.0	0.120	0.116
CLR2	50	50	100	2.0	0.145	0.141
CLR3	50	50	200	2.2	6.047	1.574
CLR4	50	50	500	3.0	—-	14.125
CLR5	50	100	100	2.0	0.217	1.373
CLR6	50	100	200	2.0	0.360	11.982
CLR7	100	100	100	2.0	0.305	0.306
CLR8	100	100	200	2.0	0.374	0.369
CLR9	100	100	200	2.0	0.374	0.369
CLR10	100	100	500	3.0	—-	80.028
CLR11	100	150	400	1.7	—-	182.208

Table 5.7
Indefinite Quadratic Problems from Phillips and Rosen (1988)

Run	Problem size			$\epsilon = 0.1$		$\epsilon = 0.01$	
	m	n	k	Iter	CPU	Iter	CPU
ILR1	25	25	25	2.0	0.232	2.200	0.312
ILR2	25	25	50	2.0	0.416	2.600	0.606
ILR3	25	25	100	2.2	1.522	3.000	2.030
ILR4	25	50	100	4.0	13.19	11.50	37.56
ILR5	50	50	50	2.0	0.864	2.400	1.504
ILR6	50	50	100	2.0	1.264	2.800	3.018
ILR7	25	75	100	3.0	68.86	30.00	294.3
ILR8	50	75	100	2.0	1.564	3.600	9.724
ILR9	75	75	100	2.0	2.120	2.800	6.304
ILR10	25	75	150	4.0	115.80	—-	—-
ILR11	50	75	150	2.2	9.5380	—-	—-
ILR12	75	75	150	2.0	2.9560	—-	—-
ILR13	25	100	50	3.6	23.21	23.50	118.6
ILR14	50	100	50	2.2	2.130	3.800	6.510
ILR15	75	100	50	2.2	3.544	2.800	5.244

GOP applied to large-scale indefinite quadratic programming problems and large-scale pooling problems.

5.2.1. Indefinite quadratic problems. The generic formulation of Phillps and Rosen (1988) is considered. By construction, we generate half of the eigenvalues, λ_i, positive and half negative. Reportedly, this is the most difficult problem to address since the solution, unlike strictly concave problems, may not lie on a vertex point. Several runs, for different problem sizes, were performed and the results are analyzed with respect to (i) the effect of the linear constraints, and linear variables; (ii) the effect of linear constraints; (iii) the effect of linear variables. Finally, some very computationally intensive tasks are discussed. In all runs we denote by k the number of linear variables, m the number of linear constraints, and n the number of quadratic variables.

The first set of computational results aims at demonstrating the effect on the performance of the GOP for different values of k and m. Typical results are tabulated in Table 5.8.

TABLE 5.8
Combined Effect of m and k. $N = m+n$, $M = m + 2(k+n)$.

k	m	n	N	M	CPU(s)
80	120	100	180	480	4.49
100	100	100	200	500	5.42
100	200	200	300	800	10.3
100	300	100	200	700	9.38
150	150	250	400	950	12.3
200	200	100	300	800	6.26
200	200	200	400	1000	7.86
200	200	220	420	1040	17.4
200	200	250	450	1100	4.50
200	300	100	300	900	5.08
300	300	100	400	1100	9.58
400	200	100	500	1200	13.2

Based on these results the following observation can be made : *as long as $\frac{k+m}{n} \geq 1$ and $\frac{m}{n} \geq 1$ the GOP algorithm identifies the global minimum with maximum efficiency.*The GOP algorithm takes two iterations and solves only two primal problems and two relaxed dual problems. Qualitatively, this implies that, for this particular structure of problems such a combination of the parameters forces the solution to lie close to a vertex point. The GOP algorithm, identifies that fact very efficiently and converges in the minimum number of iterations.

The second set of computational experiments deals with certain instances which are computationally very intensive in terms of the connected variables and number of iterations. The results have been summarized in Table 5.9. It is important to notice from Table 5.9 the fact that although

TABLE 5.9
Some computationally intensive tasks.

k	m	n	NI_c	Itn	PE	CPU(s)
30	20	100	9	54	32	100.
20	20	150	7	7	64	14.0
20	20	200	16	60	64	1847
300	100	200	8	3	32	54.0
50	50	220	10	3	64	31.6
50	50	250	14	16	64	796.
50	50	275	16	3	64	1800
50	50	300	17	3	64	3260
75	75	300	15	3	64	942.
75	75	350	11	3	64	209.

the absolute size of the problems might not be that large ($k = 30$, $m = 20$, and $n = 100$ for instance) the difficulty of the problem is noticeable both in terms of the number of iterations required as well as in terms of the number of connected variables. Note that for all the representative runs of Table 5.9 the relations between k, m, and n that define an "*easy*" problem for GOP are violated. By combining the theoretical advances of the GOP, along with the distributed implementation of the algorithm we were able to address problems of significant size.

As a last qualitative remark we will observe the computational requirements, in terms of the total CPU time, as a function of the the number of connected variables. As can be seen from Figure 5.1 there exists a linear relationship between the logarithm of the CPU time and the number of connected variables, NI_c as expected since the number of relaxed duals increases exponentially with NI_c.

Summarizing, the computational results for indefinite quadratic problems we can observe that :

- problems of 400 linear variables, 100 nonlinear variables, 200 linear constraints, and 500 bound constraints can be solved in 13.2 s. as shown in Table 5.8.
- increasing the size of the linear constraints makes the problem easier for the GOP (e.g., problems with 300 linear constraints, 200 linear variables, 100 nonlinear variables require 7.6 s.)
- increasing the number of linear variables to 400 while maintaining 100 nonlinear variables and 100 constraints increases the CPU to

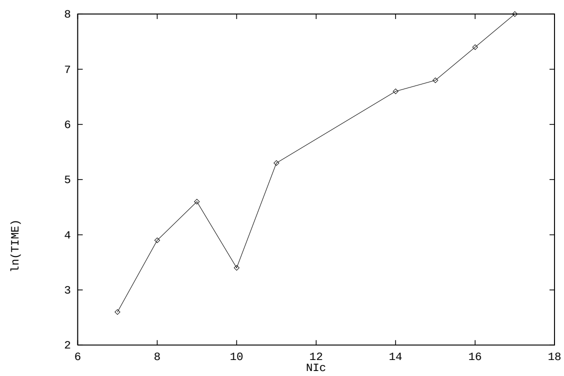


FIG. 5.1. *Time vs.* NI_s

198 s., and

- problems that correspond to increasing the number of nonlinear variables up to 350 can still be solved with reasonable computational effort, as it is shown in Table 5.9.

5.2.2. Large scale blending and pooling problems. In this section we will discuss the solution of a specific formulation of pooling/blending problems using the GOP. Such problems are very often encountered in various chemical processes. They corresponds to a quadratically constrained problem with a quadratic objective function. Different instances of randomly generated Blending and Pooling Problems based on the above reformulation were generated and solved. Typical results are shown in Table 5.10.

The distributed implementation of the GOP allowed us to address problems with 20 connected variables, which require the solution of 1,048,576 relaxed dual problems, in very realistic computational times. For instance, as can be seen from Table 5.10 pooling problems of 128 variables and 330 constraints can be solved within 840–870 s., even though they have 20 connected variables.

5.3. The GOP for bilevel linear problems. Visweswaran et al. (1996) studied the GOP approach applied to bilevel linear and quadratic optimization problems. The key ideas of this approach are presented in section 2.3. Their proposed modified GOP algorithm has been coded in C language and tested for a series of small example problems appeared in the literature. The results are summarized in Table 5.11.

A number of randomly generated problems with the same characteristics as in previous similar studies with 40% and 33% density, have also been considered. The results are summarized in Table 5.12. All compu-

Table 5.10
Blending and Pooling Problems

ncomp	nprod	npool	nqual	nvar	ncon	NI_c	PE	CPU(s)
5	5	3	5	55	186	15	64	15.3
10	4	4	9	96	300	15	64	36.3
10	4	4	16	96	356	15	64	42.0
10	4	4	18	96	372	16	64	39.0
10	4	4	25	96	428	16	64	44.9
10	4	5	30	110	468	20	64	843.
12	4	4	9	112	336	16	64	37.3
12	4	4	25	112	464	16	64	50.1
12	4	4	30	112	504	16	64	44.9
12	4	5	4	128	330	20	64	869.

Table 5.11
Results for small linear examples

EXAMPLES	n_x	n_y	Outer Constraints	Inner Constraints	Iterations	CPU time (s)
EX1	2	3	2	6	5	0.59
EX2	1	1	1	7	2	0.11
EX3	1	1	1	6	3	0.29
EX4	6	3	6	10	3	0.75
EX5	1	1	1	5	3	0.29
EX6	1	2	1	4	2	0.16
EX7	1	1	1	4	3	0.23
EX8	1	1	1	4	3	0.22
EX9	1	1	1	5	3	0.29
EX10	1	2	2	4	2	0.16
EX11	2	3	3	6	5	0.82

tations are performed using CPLEX for the solution of linear problems using a HP-730. Performance measures include CPU time and the number of iterations required to obtain the global solution. The main parameters considered are the number of the follower's constraints and the numbers n_x, n_y of the leader's and the follower's variables. Problems involving 12–17 constraints and 40–50 variables were solved.

As expected the CPU time and the number of iterations increases with the size of the problem. Notice however, that the number of required iterations remains relatively low. Also, as seen in Table 5.12, large differences in computation effort are observed between problems of the same size (for example, for the case of 26 outer and 14 inner variables, computation times of 867 and 81 sec have been reported for two different examples).

TABLE 5.12
Computational Results for randomly generated linear problems

DENSITY	n_x	n_y	Inner Constraints	Iterations	CPU time (s)
40%	28	12	12	7	8.4
40%	28	12	12	57	121.0
40%	28	12	12	9	8.0
40%	28	12	12	14	6.8
40%	26	14	14	26	81.1
40%	26	14	14	88	867.6
40%	25	15	15	12	42.8
40%	25	15	15	20	52.1
40%	24	17	17	5	34.0
40%	30	15	15	36	129.8
40%	35	15	15	26	282.3
40%	35	15	15	85	455.3
33%	27	13	13	54	249.6
33%	27	13	13	17	52.6
33%	27	13	13	21	98.8

5.4. The GGP approach. The proposed deterministic global optimization algorithm for generalized geometric programming problems by Maranas and Floudas (1996) has been implemented in GAMS and computational times reported on a HP–730 workstation are shown in Table 5.13.

The first four problems in Table 5.13 correspond to process design applications for the alkylation process, heat exchanger design, and reactor

TABLE 5.13

Computational Results the Generalized Geometric Programming Problems

Example #	N_{var}	N_{con}	N_{iter}	CPU
1	7	14	200	30
2	6	5	178	6
3	8	6	1600	100
4	8	4	71	6.8
5	5	6	30	2
6	3	1	50	17
7	3	2	7	0.4
8	8	4	82	12
9	10	6	290	22
10	11	9	2950	427
11	4	4	15	0.5
12	3	4	26	1.4
13	4	5	51	5
14	5	10	3076	485.75
15	2	4	109	22
16	8	7	4896	10,000

designs. The following six problems are small problems taken from the GGP early papers. The last six examples correspond to robust control problems with real parametric uncertainty and are presented in increasing difficulty. In particular, the last problem corresponds to a real world application and it is the most difficult problem available in the literature. The detailed mathematical formulations are presented in Maranas and Floudas (1996).

As can be seen from Table 5.13, the proposed global optimization approach can address very difficult GGP problems to global optimality within reasonable computational effort. An efficient C implementation of the algorithmic procedure is currently under way, and is expected to significantly reduce the CPU requirements.

5.5. The αBB approach. In section 4, we discussed the aBB global optimization approach for general nonlinear optimization problems. In this section, we will discuss some key aspects of its implementation and present computational results for a variety of design, control, and computational chemistry problems.

5.5.1. Implementation of αBB. One of the key characteristics of the αBB method is that it is a generic global optimization method for constrained optimization problems involving only continuous variables. The algorithm is implemented in C and at this point the user has the capability of selecting from four different types of functional forms to define the optimization model. These forms include (i) linear, (ii) convex, (iii) bilinear, and (iv) nonconvex terms. The original data are pre–processed so that any linear part in the model, (i.e. linear constraints and linear cuts), are identified at the very beginning thus reducing the amount of time that is needed to set up the problem in subsequent stages of the algorithm. The user has the capability to supply the values for the parameters α which are defined for each variable $i = 1, \ldots, N$ participating in term $k \in \mathcal{K}^j$ and constraint (or objective function) $j = 0, \ldots, M$. In principle, tailoring the α parameters for each variable, term and constraint generates tighter convex underestimators than by simply defining a single generic α for all the variables and nonconvex terms. Furthermore, the user also decides along which variables branching will be performed. These variables are typically the ones that appear in at least one nonconvex term.

The information required by the user, in the current implementation, consists of an input file and a set of user specified functions.

- *Input File* : This file provides, in a user–friendly format, information such as (i) the number of variables and constraints; (ii) the number of different functional forms (i.e. linear, convex, bilinear, and nonconvex) appearing in the model; (iii) the actual linear and bilinear entries; (iv) values for the parameter $\alpha^j_{i,k}$ for each variable, term, and constraint or objective function; and finally (v) the variables along which branching will be performed.
- *User Specified Functions* : The nonlinear, (i.e. convex and nonconvex), terms of the formulation have to be explicitly provided by the user in a form of a C or F77 subroutine. Here the user specifies, for each function (as defined in the input file), the convex and nonconvex terms.

An efficient parsing phase which would significantly simplify the problem input and declaration is currently incorporated in the version of αBB. Further work is in progress towards the evaluation of customized parameters α for different partition elements, as well as the incorporation of rigorous calculations of the parameters α for general twice-differentiable problems employing the recent results of Adjiman and Floudas (1996).

5.5.2. Computational results. The first set of results is concerned with the performance of αBB on non–convex optimization problems with simple bound constraints. A very challenging class of problems is being selected for this task, namely the minimization of the total potential energy of oligopeptides which is at the core of one of the most important problems biochemistry, namely protein folding. Table 14 summarizes computational results for all 20 naturally occurring amino acids, depicting the total number of variables, the globally minimum potential energy, the number of iterations required, and finally the CPU requirements (on an HP-730

TABLE 5.14
Computational Results for all Naturally occurring amino acids

Amino acid	N_{var}	Energy	N_{iter}	CPU	$< CPU >$
Pro	5	-19.81	28	6	**6**
Gly	6	-6.33	67	14	**14**
Ala	7	-5.18	141	55	
Cys	7	-5.84	142	45	**50**
His	8	-8.92	298	173	
Phe	8	-8.43	298	169	
Ser	8	-7.86	184	102	
Trp	8	-9.56	306	227	**167**
Asn	9	-22.95	345	220	
Asp	9	-20.05	452	239	
Thr	9	-9.59	285	208	
Tyr	9	-8.48	753	506	
Val	9	-4.19	644	387	**312**
Gln	10	-18.99	601	460	
Glu	10	-15.87	640	386	
Ile	10	-2.54	388	352	
Leu	10	-5.72	1123	613	
Met	10	-6.91	1284	641	**480**
Lys	11	-7.98	922	1070	**1070**
Arg	13	-31.84	1000	1660	**1660**

machine) per amino acid as well as the average CPU time of all amino acids with the same number of variables. In this work the empirical model ECEPP3 is used to model the energetic interactions. Such models are characterized by complex interactions resulting in severe non–linearities giving rise to an exponentially increasing number of local minima. Identifying the global minimum among them, is a major computational challenge. One of the most important conclusions to be drawn is the fact that the computational requirements of αBB compare favorably with the reported results employing simulated annealing.

The αBB has been tested on a large number of test problems from the literature and representative results will now be presented. The first 2 examples, as presented in Table 5.15 describe a very challenging, for local solvers, robust control synthesis problem. It is characterized by strong nonlinearities in the objective function and simple bound constraints. Two cases are provided. Interestingly enough the success of a local solver to identify the global minimum for example 2 is 1%. The next 3 examples, describe the phase equilibrium of two ternary and a binary mixtures. These types of physical computation is both very important, from an engineering point of view, as well as very challenging from the global optimization point of view. Example 4 exhibits a local solution extremely close, in objective function values, to the global optimum but with major differences in terms of the physical characteristics of the solution. Example 6,7, and 8 are three well known formulation describing the so-called blending and pooling problems. Three cases are defined in the literature and have been extensively used for testing global optimization algorithms and all of them very successful addressed. Example 9, is another test problem taken form the literature featuring non–convextities in the form of bilinearities in the constraint set. Example 10 is a very challenging problem describing the separation of a three–component feed mixture. The goal is to achieve the desired separation at the minimum possible capital cost. The next example, discusses a reactor network design problem and it is known to have caused problems for local solvers due to the existence of a local solution very close to the global one. The computational results as presented here represent a substantial improvement when compared to the original reference. Example 12 describes a large scale chemical reactor network design problem. The goal being the identification of the sizes of the required reactors as well as the configuration of the reactor network. This problem is characterized by non–convexities in terms of bilinear terms as well as general non–convex terms describing the complex reactions kinetics. The last two problems describe general non–convex optimization formulations. Example 13 is a test problem having 6 local solutions. The last is a very challenging and important one with major applications. It addresses the optimal blank nesting problem. the objective is to minimize the amount of scrap metal. It was identified that this problem poses major difficulties to a local solver not only because of the existence of a plethora of local solutions, but also because of a fairly high failure rate of local solver due to the presence of severe non–convexities.

6. Discussion. It should be pointed out that the three approaches presented in this paper address different classes of nonconvex optimization problems. The GOP approach is best suited for quadratic, bilinear, and biconvex problems. The GGP approach is specific to the signomials. The αBB approach was developed for general twice-differentiable optimization problems that exhibit highly nonlinear terms. Even though the αBB approach can be applied to quadratic problems and generalized geometric programming problems, its performance should not be expected to outperform, in general, the specialized GOP and GGP global optimization approaches that exploit further the mathematical structure of the problem. In the case of quadratic functions, for instance, the αBB provides convex underestimators and the parameters α are constant and hence cannot be updated at each iteration. Furthermore, in the case of concave

Table 5.15
αBB Example Description

Example #	Description
1	Example 6.3(a), Androulakis*et al*, 1995
2	Example 6.3(b), Androulakis*et al*, 1995
3	Example 6.3(I), Androulakis*et al*, 1995
4	Example 6.3(II), Androulakis*et al*, 1995
5	Example 6.3(II), Androulakis*et al*, 1995
6	Example 6.1(I), Androulakis*et al*, 1995
7	Example 6.1(II), Androulakis*et al*, 1995
8	Example 6.1(III), Androulakis*et al*, 1995
9	Example 3.1.1, Floudas and Pardalos, 1990
10	Example 5.4, Floudas and Pardalos, 1990
11	Example 20, Ryoo and Sahinidis, 1995
12	Example 9.2, Floudas and Pardalos, 1990
13	Example 6.5(I), Androulakis*et al*, 1995
14	Example 6.5(II), Androulakis*et al*, 1995

Table 5.16
Computational Results for General Nonconvex Optimization Problems using αBB

Example #	N_{var}	N_{con}	N_{iter}	CPU
1	4	0	17	0.64
2	4	0	400	11.6
3	4	2	57	1.3
4	6	3	433	13.6
5	4	2	56	2.1
6	9	6	14	1.48
7	9	6	17	1.78
8	9	6	14	0.90
9	8	6	202	40.51
10	38	32	123	901
11	6	5	56	16
12	29	41	293	4179
13	5	3	406	168
14	8	118	250	3,153

quadratic problems or the case of concave programming, the αBB underestimators being convex are weaker than the known linear underestimators. One of the primary advantages of the αBB approach is that the underestimators do not require the introduction of new variables and constraints but they simply correspond to reformulation of the objective function and constraints. Hence, the size of the underestimating problems remains always the same as the original one. This can be a significant advantage for problems that have many bilinear terms in the objective function and constraints. In such a class of problems, the alternative underestimation schemes (e.g., McCormick (1976), Al-Khayyal and Falk (1983), Sherali and Alameddine (1992)) will require the addition of many new variables and constraints. There is of course the trade-off between the size of the underestimating problems and the quality of bounds that they provide and can be specific to the class of problems addressed. Preliminary results in small and medium size quadratic and bilinear literature problems such as pooling problems and linearly constrained concave quadratic programming problems indicate that the GOP performs slightly better than the αBB. Comparing the αBB with the GOP approach for problems with many bilinear terms it is expected that the GOP/MILP variant will be competitive for up to twenty connected variables. Preliminary results on certain generalized geometric programming problems indicate that the GGP and the αBB behave comparably.

Concluding remarks. This paper has presented an overview of the advances in the area of deterministic global optimization. This overview started with a general classification of the different approaches and continued with a primary focus on our recent contributions on (i) decomposition methods, (ii) generalized geometric programming problems, and (iii) methods for general nonlinear optimization problems. The theoretical advances have been supported by ample computational work on challenging literature problems and randomly generated problems, as well as important applications in Process Synthesis, Design, Robust Control, and Computational Chemistry and Biology. The reported theoretical, and computational results, along with forthcoming availability of global optimization tools strongly suggest the potential of the area of deterministic global optimization toward addressing challenging and very important engineering and applied science problems.

Ackowledgements. Financial support from the National Science Foundation, the Air Force Office of Scientific Research, the National Institute of Health, and the Binational Science Foundation is gratefully acknowledged.

REFERENCES

[1] Adjiman, C.S., I.P. Androulakis, C.D. Maranas, and C.A. Floudas, A Global Optimization Method, α**BB**, for Process Design, *Computers and Chemical En-*

gineering, accepted for publication, (1996).

[2] Adjiman, C.S. and C.A. Floudas, Rigorous Convex Underestimators for General Twice Differentiable Problems, *J. Global Optim.*, accepted for publication, (1996).

[3] Al-Khayyal, F.A., and Falk, J.E., Jointly Constrained Biconvex Programming, *Mathematics of Operations Research*, **8**, (1983).

[4] Al-Khayyal, F.A., Jointly Constrained Bilinear Programs and Related Problems : An Overview , *Computers in Mathematical Applications*, **19**, 53 (1990).

[5] Androulakis, I.P., C.D. Maranas, and C.A. Floudas, α**BB**: A Global Optimization Method for General Constrained Nonconvex Problems, *Journal of Global Optimization*, 7, 337-363, (1995).

[6] Androulakis, I.P., Visweswaran, V. and C.A. Floudas, Distributed Decomposition-based Approaches in Global Optimizattion, *State of the Art in Global Optimization : Computational Methods and Applications, (Eds. C.A. Floudas and P.M. Pardalos)*, Kluwer Academic Publishers, (1996).

[7] Bangia, J.R. and W.D. Seider, Global Optimization of Chemical Processes using Stochastic Algorithms, *State of the Art in Global Optimization : Computational Methods and Applications, (Eds. C.A. Floudas and P.M. Pardalos)*, Kluwer Academic Publishers, (1996).

[8] Barmish, B.R., C.A. Floudas, C.V. Hollot, and R. Tempo, A Global Linear Programming Solution to Some Open Robustness Problems Including Matrix Polytope Stability, *Proceedings of American Control Conference, ACC*, (1995).

[9] Ben-Tal, A., Eiger, G., and V. Gershovitz, Global Minimization by Reducing the Duality Gap, *Mathematical Programming*, 63, 193-212 (1994).

[10] Byrne, R.P., and I.D.L. Bogle, Solving Nonconvex Process Optimization Problems Using Interval Subdivision Algorithms, *Global Optimization in Engineering Design, (Ed. I.E. Grossmann)*, Kluwer Academic Publishers, pp. 155-174, (1996).

[11] Epperly, T.G.W., and R.E. Swaney, Branch and Bound for Global NLP : New Bounding LP, *Global Optimization in Engineering Design, (Ed. I.E. Grossmann)*, Kluwer Academic Publishers, pp. 1-36, (1996).

[12] Epperly, T.G.W., and R.E. Swaney, Branch and Bound for Global NLP : Iterative LP Algorithm and Results, *Global Optimization in Engineering Design, (Ed. I.E. Grossmann)*, Kluwer Academic Publishers, pp. 37-74, (1996).

[13] Floudas, C.A., *Nonlinear and Mixed-Integer Optimization : Fundamentals and Applications*, Oxford University Press, 1995.

[14] Floudas, C.A., Aggarwal, A., and Ciric, A.R., Global Optimum Search for Nonconvex NLP and MINLP problems, *Computers and Chemical Engineering*, **13**, 1117 (1989).

[15] Floudas, C.A., and Pardalos, P.M., A Collection of Test Problems for Constrained Global Optimization Algorithms, *Lecture Notes in Computer Science*, Eds. G. Goos and J. Hartmanis, Vol. 455, Springer-Verlag (1990).

[16] Floudas, C.A., and Visweswaran, V., A Global Optimization Algorithm (GOP) For Certain Classes of Nonconvex NLPs : I. Theory, *Computers and Chemical Engineering*, Vol. 14, No. 12, 1397-1417 (1990).

[17] Floudas, C.A. and I.E. Grossmann, Algorithmic Approaches to Process Synthesis : Logic and Global Optimization, *Proc. of FOCAPD'94*, (Eds.) L. Biegler and M. Doherty, p. 198 (1995).

[18] Floudas, C.A. and P.M. Pardalos, (Eds.), State of the Art in Global Optimization : Computational Methods and Applications, Proceedings of Conference, Kluwer Academic Publishers (1996).

[19] Grossmann, I.E., (Ed.), Global Optimization in Engineering Design, Kluwer Academic Publishers (1996).

[20] Geoffrion, A.M., Generalized Benders Decomposition, *Journal of Optimization Theory and Applications*, **10**, 237 (1972).

[21] Hansen, E.R., Global Optimization using Interval Analysis: The One-Dimensional Case, *Journal of Optimization Theory and Applications*, **29**, 331 (1979).
[22] Hansen, P., Jaumard, B., and G. Savard, New Branching Rules for Linear Bilevel Programming, *SIAM J. Sci. Stat. Comp.* (1990). Global Optimization of Univariate Lipschitz functions : I. Survey and Properties, *Mathematical Programming*, 55, 251-272 (1992a).
[23] Hansen, P., Jaumard, B., and Lu, S-H., Global Optimization of Univariate Lipschitz functions : I. Survey and Properties, *Mathematical Programming*, 55, 251-272 (1992a).
[24] Hansen, P., Jaumard, B., and Lu, S-H., Global Optimization of Univariate Lipschitz functions : II. New Algorithms and Computational Comparisons, *Mathemetical Programming*, 55, 273-292 (1992b).
[25] Hansen, P., Jaumard, B., and Lu, S-H., An Analytical Approach to Global Optimization, *To appear in Mathematical Programming* (1990).
[26] Horst, R., and Tuy, H., On the Convergence of Global Methods in Multiextremal Optimization, *Journal of Optimization Theory and Applications*, **54**, 253 (1987).
[27] Horst, R., and Tuy, H., *Global Optimization: Deterministic Approaches*, Springer-Verlag (1990).
[28] Horst, R. and P.M. Pardalos, Handbook of Global Optimization, Kluwer Academic Publishers, (1995).
[29] Horst, R., P.M. Pardalos, and N.V. Thoai, Introduction to Global Optimization, Kluwer Academic Publishers, (1995).
[30] Horst, R., N.V. Thoai, and H. Tuy, On an Outer Approximation Concept in Global Optimization, *Optimization*, 20, 255-264 (1989).
[31] Ierapetritou M.G. and E.N. Pistikopoulos, Global Optimization for Stochastic Planning, Scheduling and Design Problems, *Global Optimization in Engineering Design, (Ed. I.E. Grossmann)*, Kluwer Academic Publishers, pp. 231-288, (1996).
[32] Iyer, R.R., and I.E. Grossmann, Global Optimization of Heat Exchanger Networks with Fixed Configuration for Multiperiod Design, *Global Optimization in Engineering Design, (Ed. I.E. Grossmann)*, Kluwer Academic Publishers, pp. 289-308, (1996).
[33] Kirkpatrick, S., Gelatt, C.D, and Vecchi, M.P., Optimization by Simulated Annealing, *Science*, **220**, 671 (1983).
[34] Liu, M.L., N.V. Sahinidis, and J.P. Shectman, Planning of Chemical Process Networks via Global Concave Minimization, *Global Optimization in Engineering Design, (Ed. I.E. Grossmann)*, Kluwer Academic Publishers, pp. 195-230, (1996).
[35] Liu, W. and C.A. Floudas, A Remark on the GOP Algorithm for Global Optimization, *J. of Global Optimization*, 3, 4, 519-522 (1993).
[36] Liu, W. and C.A. Floudas, Convergence of the GOP Algorithm for a Large Class of Smooth Optimization Problems, *J. of Global Optimization*, 6, 207-211 (1995).
[37] Liu, W. and C.A. Floudas, Generalized Primal-Relaxed Dual Approach for Global Optimization, *J. Optim. Theory and its Applications*, in press, (1996).
[38] Lucia, A., and J. Xu, Nonconvexity and Descent in Nonlinear Programming, *State of the Art in Global Optimization : Computational Methods and Applications, (Eds. C.A. Floudas and P.M. Pardalos)*, Kluwer Academic Publishers, (1996).
[39] Manousiouthakis, V. and D. Sourlas, A Global Optimization Approach to Rationally Constrained Rational Programming, *Chem. Eng. Comm.*, 115, 127-147 (1992).
[40] Maranas, C.D. and C.A. Floudas, A Global Optimization Approach for Lennard-Jones Microclusters, *J. Chemical Physics*, 97, 10, 7667-7678, (1992).
[41] Maranas, C.D. and C.A. Floudas, Global Optimization for Molecular Conformation Problems, *Annals of Operations Research*, 42, 85-117, (1993).
[42] Maranas, C.D. and C.A. Floudas, Global Minimum Potential Energy Conforma-

tions of Small Molecules, *J. of Global Optimization*, 4, 2, 135-170 (1994a).
[43] Maranas C.D. and C.A. Floudas, A Deterministic Global Optimization Approach for Molecular Structure Determination, *J. of Chemical Physics*, 100, 2, 1247-1261 (1994b).
[44] Maranas C.D. and C.A. Floudas, Finding All Solutions of Nonlinearly Constrained Systems of Equations, *J. Global Optimization*, 7, 143-182, (1995).
[45] Maranas C.D. and C.A. Floudas, Global Optimization in Generalized Geometric Programming, *Computers & Chemical Engineering*, in press (1996).
[46] McCormick, G.P., Computability of Global Solutions to Factorable Nonconvex Programs : Part I - Convex Underestimating Problems, *Mathematical Programming*, 10, 147-175, (1976).
[47] McDonald, C.M. and C.A. Floudas, Decomposition Based and Branch and Bound Global Optimization Approaches for the Phase Equilibrium Problem, *J. Global Optim.*, 5,205-251, (1994).
[48] McDonald, C.M. and C.A. Floudas, Global Optimization for the Phase and Chemical Equilibrium Problem : Application to the NRTL Equation, *Computers & Chemical Engineering*, 19, 11, 1111-1139, (1995a).
[49] McDonald, C.M. and C.A. Floudas, Global Optimization and Analysis for the Gibbs Free Energy Function Using the UNIFAC, Wilson and ASOG Equations, *I&EC Research*, 34, 1674-1687, (1995b).
[50] McDonald, C.M. and C.A. Floudas, Global Optimization for the Phase Stability Problem, *AIChE J.*, 41, 7, 1798-1814, (1995c).
[51] McDonald, C.M. and C.A. Floudas, GLOPEQ : A New Computational Tool for the Phase and Chemical Equilibrium Problem, *Computers & Chemical Engineering*, in press (1996).
[52] McKinnon, K.I.M., Millar, C., and M. Mongeau, Global Optimization for the Chemical and Phase Equilibrium Problem using Interval Analysis, *State of the Art in Global Optimization : Computational Methods and Applications, (Eds. C.A. Floudas and P.M. Pardalos)*, Kluwer Academic Publishers, (1996).
[53] Mockus, L. and G.V. Reklaitis, A New Global Optimization Algorithm for Batch Process Scheduling, *State of the Art in Global Optimization : Computational Methods and Applications, (Eds. C.A. Floudas and P.M. Pardalos)*, Kluwer Academic Publishers, (1996).
[54] Pardalos, P.M., and Rosen, J.B., Constrained Global Optimization : Algorithms and Applications, *v. 268 of Lecture Notes in Computer Science*, Springer Verlag (1987).
[55] Pinter, J., Global Optimization in Action, Kluwer Academic Publishers, (1996).
[56] Quesada, I. and I.E. Grossmann, Global Optimization Algorithm in Heat Exchanger Networks, *Ind. Eng. Chem. Res.*, 32, 487-499 (1993).
[57] Quesada, I. and I.E. Grossmann, A Global Optimization Algorithm for Linear Fractional and Bilinear Programs, *J. Global Optim.*, 6, 39-76 (1995).
[58] Quesada, I. and I.E. Grossmann, Alternative Bounding Approximations for the Global Optimization of Various Engineering Design Problems, *Global Optimization in Engineering Design, (Ed. I.E. Grossmann)*, Kluwer Academic Publishers, pp. 309-332, (1996).
[59] Ratschek, H., and Rokne, J., *New Computer Methods for Global Optimization*, Halsted Press (1988).
[60] Rinnoy Kan, A.H.G., and Timmer, G.T., Stochastic Global Optimization Methods. Part I: Clustering Methods, *Mathematical Programming*, **39**, 27 (1987).
[61] Ryoo, H.S. and N.V. Sahinidis, Global Optimization of Nonconvex NLPs and MINLPs with Applications in Process Design, *Computers and Chemical Engineering*, 19, pp. 551-566 (1995).
[62] Sherali, H., and A. Alameddine, A New Reformulation-Linearization Technique for Bilinear Programming Problems, *Journal of Global Optimization*, 2, pp. 379-410 (1992).
[63] Shectman, J.P. and N.V. Sahinidis, A Finite Algorithm for Global Minimization of

Separable Concave Programs, *State of the Art in Global Optimization : Computational Methods and Applications, (Eds. C.A. Floudas and P.M. Pardalos)*, Kluwer Academic Publishers, (1996).

[64] Shor N.Z., Dual Quadratic Estimates In Polynomial and Boolean Programming, *Annals of Operations Research*, Special Volume On Computational Methods In Global Optimization, (Eds. P.M. Pardalos and J.B. Rosen), Vol.27 (1990).

[65] Staus, G.H., L.T. Biegler, and B.E. Ydstie, Adaptive Control via Non-Convex Optimization, *State of the Art in Global Optimization : Computational Methods and Applications, (Eds. C.A. Floudas and P.M. Pardalos)*, Kluwer Academic Publishers, (1996).

[66] Smith, E.M.B. and C.C. Pantelides, Global Optimization of General Process Models, *Global Optimization in Engineering Design, (Ed. I.E. Grossmann)*, Kluwer Academic Publishers, pp. 355-386, (1996).

[67] Stephanopoulos, G., and Westerberg, A.W., The Use of Hestenes' Method of Multipliers to Resolve Dual Gaps in Engineering System Optimization, *Journal of Optimization Theory and Applications*, **15**, 285 (1975).

[68] Swaney, R.E., Global Solution of Algebraic Nonlinear Programs, presented at Annual AIChE Meeting, Chicago, November (1990).

[69] Tsirukis, A.G. and G.V. Reklaitis, Feature Extraction Algorithms for Constrained Global Optimization, I and II, *Annals of Operations Research*, 42, 229-275 (1993).

[70] Törn, A., and Zilinskas, A., Global Optimization , *Lecture Notes in Computer Science, No. 350*, Springer-Verlag (1987).

[71] Tuy, H., Thieu, T.V., and Thai, N.Q., A Conical Algorithm for Globally Minimizing a Concave Function Over a Closed Convex Set, *Mathematics of Operations Research*, **10**, 498 (1985).

[72] Tuy, H., Global Minimization of a Difference of Two Convex Functions, *Mathematical Programming Study*, 30, 150-182 (1987a).

[73] Tuy, H., Convex Programs with an Additional Reverse Convex Constraint, *Journal of Optimization Theory and its Applications*, 52, 463-485 (1987b).

[74] Westerberg, A.W. and Shah, J.V., Assuring a Global Minimum by the Use of an Upper Bound on the Lower (Dual) Bound, *Computers and Chemical Engineering*, **2**, 83 (1978).

[75] Zilinskas, A., *Global Optimization – Axiomatics of Statistical Models, Algorithms and Their Applications*, Moklas, Vilnius (1986)

[76] Vaidyanathan, R, and M. El-Halwagi, Global Optimization of Nonconvex Nonlinear Programs via Interval Analysis, *Computers and Chemical Engineering*, 18, pp. 889-897 (1994).

[77] Vaidyanathan, R. and M. El-Halwagi, Global Optimization of Nonconvex MINLPs by Interval Analysis, *Global Optimization in Engineering Design, (Ed. I.E. Grossmann)*, Kluwer Academic Publishers, pp. 175-194, (1996).

[78] Visweswaran, V. and Floudas, C.A., A Global Optimization Algorithm (GOP) For Certain Classes of Nonconvex NLPs : II. Application of Theory and Test Problems, *Computers and Chemical Engineering*, Vol. 14, No. 12, 1419-1434(1990).

[79] Visweswaran V. and C.A. Floudas, Unconstrained and Constrained Global Optimization of Polynomial Functions in One Variable, *Journal of Global Optimization*, 2, 73-99 (1992).

[80] Visweswaran, V. and C.A. Floudas, New Properties and Computational Improvement of the GOP Algorithm for Problems with Quadratic Objective and Constraints, *J. of Global Optimization*, 3, 4, 439-462 (1993).

[81] Visweswaran, V. and C.A. Floudas, New Formulations and Branching Strategies for the GOP Algorithm, *Global Optimization in Chemical Engineering Design, (Ed. I.E. Grossmann)*, Kluwer Academic Publishers, pp. 75-110, (1996a).

[82] Visweswaran, V. and C.A. Floudas, Computational Results for an Efficient Implementation of the GOP Algorithm and its Variants, *Global Optimization in Chemical Engineering Design, (Ed. I.E. Grossmann)*, Kluwer Academic

Publishers, pp. 111-154, (1996b).

[83] Visweswaran, V., Floudas, C.A., Ierapatritou, M.G., and E.N. Pistikopoulos, A Decomposition-based Global Optimization Approach for Solving Bilevel Linear and Quadratic Programs, *State of the Art in Global Optimization, (Eds. C.A. Floudas and P.M. Pardalos)*, Kluwer Academic Publishers, (1996).