GLOBAL OPTIMIZATION

Scientific and Engineering Case Studies
Nonconvex Optimization and Its Applications

VOLUME 85

Managing Editor:

Panos Pardalos
University of Florida, U.S.A.

Advisory Board:

J. R. Birge
University of Chicago, U.S.A.

Ding-Zhu Du
University of Minnesota, U.S.A.

C. A. Floudas
Princeton University, U.S.A.

J. Mockus
Lithuanian Academy of Sciences, Lithuania

H. D. Sherali
Virginia Polytechnic Institute and State University, U.S.A.

G. Stavroulakis
Technical University Braunschweig, Germany

H. Tuy
National Centre for Natural Science and Technology, Vietnam
GLOBAL OPTIMIZATION

Scientific and Engineering Case Studies

Edited by

JÁNOS D. PINTÉR
Pintér Consulting Services Inc., Halifax, Nova Scotia, Canada
“Far and away the best prize that life offers is the chance to work hard at work worth doing.”

Theodore Roosevelt (1858–1919)

This volume is dedicated to people around the world who need our work – research, models, and techniques – to help them live, and to live better.
Contents

Preface and Acknowledgments
  J. D. Pintér

Chapter 1  Global Optimization in Solvent Design
  L. E. K. Achenie, G. M. Ostrovsky and M. Sinha

Chapter 2  Feeding Strategies for Maximising Gross Margin in Pig Production

Chapter 3  Optimized Design of Dynamic Networks with Heuristic Algorithms
  S. Allen, S. Hurley, V. Samko and R. Whitaker

Chapter 4  A New Smoothing-based Global Optimization Algorithm for Protein Conformation Problems
  A. M. Azmi, R. H. Byrd, E. Eskow and R. B. Schnabel

Chapter 5  Physical Perspectives on the Global Optimization of Atomic Clusters
  J. P. K. Doye

Chapter 6  Efficient Global Geometry Optimization of Atomic and Molecular Clusters
  B. Hartke

Chapter 7  Computational Analysis of Human DNA Sequences: An Application of Artificial Neural Networks
  A. G. Hatzigeorgiou and M. S. Megraw

Chapter 8  Determination of a Laser Cavity Field Solution Using Global Optimization
  G. Isenor, J. D. Pintér and M. Cada

Chapter 9  Computational Experience with the Molecular Distance Geometry Problem
  C. Lavor, L. Liberti and N. Macaulan

Chapter 10 Non-linear Optimization Models in Water Resource Systems
  S. Liberatore, G. M. Sechi and P. Zuddas
Chapter 11  Solving the Phase Unwrapping Problem by a Parameterized Network Optimization Approach  
P. Maponi and F. Zirilli  
243

Chapter 12  Evolutionary Algorithms for Global Optimization  
A. Osyczka and S. Krenich  
267

Chapter 13  Determining 3-D Structure of Spherical Viruses by Global Optimization  
O. Ozturk, P. C. Doerschuk and S. B. Gelfand  
301

Chapter 14  A Collaborative Solution Methodology for Inverse Position Problem  
C. S. Pedamallu and L. Özdamar  
331

Chapter 15  Improved Learning of Neural Nets through Global Search  
V. P. Plagianakos, G. D. Magoulos and M. N. Vrahatis  
361

Chapter 16  Evolutionary Approach to Design Assembly Lines  
B. Rekiek, P. De Lit and A. Delchambre  
389

Chapter 17  Agroecosystem Management  
R. Seppelt  
413

Chapter 18  Finding the Minimal Root of an Equation – Applications and Algorithms Based on Lipschitz Condition  
Ya. D. Sergeyev  
441

Chapter 19  Optimization of Radiation Therapy Dose Delivery with Multiple Static Collimation  
J. Tervo, P. Kolmonen, J. D. Pintér and T. Lyyra-Lahtinen  
461

Chapter 20  Parallel Triangulated Partitioning for Black Box Optimization  
Y. Wu, L. Özdamar and A. Kumar  
487

Chapter 21  A Case Study: Composite Structure Design Optimization  
Z. B. Zabinsky, M. E. Tuttle and C. Khompatraporn  
507

Chapter 22  Neural Network Enhanced Optimal Self-tuning Controller Design for Induction Motors  
Q. M. Zhu, L. Z. Guo and Z. Ma  
529
Preface

Man-made systems and controlled procedures can often be described, at least to a postulated “reasonable degree of accuracy”, by continuous linear functions. For prominent instances of such descriptions, one may think of production and distribution systems and their basic quantitative models known from the operations research, management science, and industrial engineering literature. For illustrative purposes, we refer to the 50th Anniversary Issue of Operations Research (2002), and to the many topical entries of Greenberg’s (2005) Mathematical Programming Glossary (on the web), and the Handbook of Applied Optimization edited by Pardalos and Resende (2002).


Prescriptive (control, management, optimization) models based on a nonlinear systems description often may – or provably do – possess multiple local optima. The objective of global optimization (GO) is to find the “best possible” solution of multiextremal problems. Formally, the prototype continuous global optimization problem (GOP) can be stated as

\[
\begin{align*}
\text{min} & \quad f(x) \\
\text{subject to} & \quad x \in D = \{l \leq x \leq u; \quad f_j(x) \leq 0 \quad j = 1, \ldots, J\} \subset \mathbb{R}^n.
\end{align*}
\]

The relations (1)-(2) describe a very general optimization model type defined by the following key ingredients:

- \(x\) real-valued \(n\)-vector that describes the decision alternatives
- \(f(x)\) continuous objective function; \(f_0(x) := f(x)\)
• $D$ non-empty set of feasible decisions
• $f_j(x)$ continuous constraint functions, for $j=1,...,J$
• $l, u$ explicit, finite (component-wise) bounds of $x$.

Applying these basic analytical assumptions, it is easy to verify (by the extreme value theorem of classical analysis) that the optimal solution set of the GOP is non-empty. We shall denote the set of globally optimal solutions by $X^*$. The solution of the GOP theoretically requires the determination of the set $X^*$, or at least an exact global solution $x^* \in X^*$ and the corresponding optimum value $f^* = f(x^*)$. In practice, this is often not possible (not only in the context of global optimization, but across the various classes of continuous optimization problems). Therefore our standard numerical objective is the sufficiently precise approximation of $X^*$ or an $x^*$, and of $f^*$, based on a finite number of algorithmically chosen search steps. This objective requires the generation of a sample point sequence $\{x_k\}$ and corresponding model function evaluations $\{f_j(x_k)\}$ for $j=0,1,...,J$. The function evaluations optionally may include also the calculation or estimation of higher-order or other – local or global – information.

To illustrate the potential difficulty of the general GOP, consider the following (merely two-dimensional, and only box-constrained) model instance. (For simplicity, the variables are denoted by $x$ and $y$, to avoid the use of indices.)

\[
\begin{align*}
\text{(3) } \min f(x,y) \quad f(x,y) := & (\sin(xy) + \sin(3y-5x) + \sin(x^2-4y) - 2)^2 \\
\text{subject to } & -3 \leq x \leq 3, -2 \leq y \leq 5.
\end{align*}
\]

The surface and contour plots of the function (3) in the region (4) are shown by Figures 1 and 2.
These figures illustrate two important facts.

1) Global optimization models can be difficult (perhaps immensely difficult), even in very low-dimensions.
2) The classical repertoire of (local) numerical optimization is not suitable on its own to handle this problem and similar multi-extremal models. Indeed, depending on the starting point of any given local optimization method, it can easily get “trapped” in one of the many valleys (regions of attraction) of the objective function surface.

As a side note for the interested reader, several numerical global solutions of the GOP (3)-(4) are listed below:

<table>
<thead>
<tr>
<th>(x^*)</th>
<th>(y^*)</th>
<th>(f(x^<em>, y^</em>))</th>
</tr>
</thead>
<tbody>
<tr>
<td>-0.4891396742</td>
<td>-0.2256392574</td>
<td>1.6537752573033925 \times 10^{-16}</td>
</tr>
<tr>
<td>-1.1840237053</td>
<td>2.9661319066</td>
<td>1.3696484154236246 \times 10^{-15}</td>
</tr>
<tr>
<td>-0.4129962319</td>
<td>-0.3441255592</td>
<td>4.3970820983759176 \times 10^{-17}</td>
</tr>
<tr>
<td>-2.8343606253</td>
<td>-0.0078477508</td>
<td>1.090931979062312 \times 10^{-15}</td>
</tr>
</tbody>
</table>

Figure 2. Contour plot of the objective function in the GOP (3)-(4).
In the case of multiple global optima, it may be of interest to select a specific solution such as the one that has a minimal Euclidean norm. The corresponding solution is

\[ x^* \approx -0.4148624957, \quad y^* \approx -0.254331095, \quad \| (x^*,y^*) \| = 0.4866160665476753. \]

Each of these approximate global solutions have been generated using the MathOptimizer Professional software with an external Lipschitz Global Optimizer (LGO) solver link (Pintér, 1996a; Pintér, 1996...2005; Pintér and Kampas, 2003). The solution times are less than 0.3 second in each case, on a desktop PC equipped with an Intel Pentium 4 1.6 GHz processor.

Proceeding further, let us remark next that even relatively “simple” specific instances of the GOP may have an exponentially increasing number of global and local solutions. One can imagine a 10- or 100-variable direct extension of the model (3)-(4) in which selected two-dimensional search subspaces would show similar complexity to Figures 1 and 2. Since there are no universally applicable analytical criteria to verify global optimality, the required algorithmic search effort could grow exponentially in terms of the model size, since the algorithm should have to “visit” around the entire search region in sufficient detail. The model size of the GOP is characterized here simply by \( n \) and \( m \), without further consideration given to the specific form of the model constraint functions. These functions could also be complicated: for a visual example, one can think of a collection of disjoint subsets (“islands of an archipelago”) that together form \( D \).

The general GO paradigm expressed by the model statement (1)-(2) is in contrast to traditional optimization methods. The latter – as a rule – will find only local optima of the GOP, based on a user-supplied initial (local) guess of the solution. Global optimization encompasses and extends local nonlinear optimization. This is certainly valid in a formal theoretical sense, but is pertinent also in numerical practice since GO strategies eventually need to have the convergence guarantee and precision of local search methodology, at least with respect to the global solution(s).

The field of global optimization has been gaining increasing attention in recent decades, and it has reached a certain level of maturity. The number of textbooks focused on GO is in the hundreds worldwide. The book series titled Nonconvex Optimization and Its Applications in itself includes nearly one hundred volumes, as of 2005. From this series, consult e.g. the introductory volume by Horst, Pardalos, and Thoai (1995), the two Handbooks edited by Horst and Pardalos (1995) and by Pardalos and Romeijn (2002), or the volumes by Kearfott (1996), Pintér (1996a), Tawarmalani and Sahinidis (2002), and Zabinsky (2003). The recent book chapter by Neumaier (2004) also provides a detailed overview of rigorous deterministic GO approaches.

Algorithmic advances – together with readily accessible and relatively inexpensive computational power – have led to a growing range of global optimization software implementations. This development has been greatly facilitated by significant progress in the areas of core professional (mainly C and
Fortran) compilers, spreadsheet-based modeling, algebraic modeling languages (with a focus on optimization), and integrated scientific-technical computing systems. Without going into details on any of these software systems, please consult e.g. the following references and the software products discussed therein.

- Algebraic modeling languages:
  - AIMMS (Paragon Decision Technology, 2005)
  - AMPL (Fourer, Gay, and Kernighan, 1993)
  - GAMS (Brooke, Kendrick, and Meeraus, 1988)
  - LINDO Solver Suite (Schrage, 2001; LINDO Systems, 2005)
  - LPL (Virtual Optima, 2005)
  - MPL (Maximal Software, 2005)
  - TOMLAB for MATLAB (TOMLAB Optimization, 2004)
- Integrated scientific-technical computing systems:
  - Maple (Lopez (2005), Maplesoft (2005), Parlar (2000), Wright (2002))

With respect to modeling environments and GO software implementations, see also the edited volumes (Kallrath, 2004), (Liberti and Maculan, 2005), and (Pinter, 2005). The websites maintained by Fourer (2005), Mittelmann and Spellucci (2005), Neumaier (2005), and by the Optimization Technology Center (2005) also offer valuable topical information.

While most GO software products a decade ago have been arguably more "academic" than "professional" (Pintér, 1996b), today a growing number of companies offer professionally developed and maintained GO software, often as a solver component or option of modeling language and systems. Global optimization is also becoming part of the "mainstream" operations research curriculum: for instance, the prominent textbook by Hillier and Lieberman (2005) now offers also GO demo software (the model size-limited MPL/LGO implementation) as part of its electronic supplement.

The present volume illustrates the applicability of global optimization strategies and software to a broadening range of practically important issues. The emphasis is on real-world applications – including also open problems – that apparently need genuine GO methodology. The contributed chapters cover applications from the following areas:

- agroecosystem management
- assembly line design
- bioinformatics
- biophysics
These applications can be broadly classified as belonging to the areas of natural sciences (agriculture, biology, computational chemistry, environment) and engineering (design and process optimization), while mathematical modeling, optimization and computer science are the unifying concepts. Let us remark here that in addition to engineering and scientific applications (represented by the studies of this volume) important areas of GO applications emerge also in econometrics and finance.

Due to the large variety of model types encompassed by the general GO paradigm (1)-(2), there is no “universally best” global optimization strategy or software that will handle all GO models with theoretical rigor and competitive efficiency, within the framework of a prefixed amount of resources (time or model function evaluations). This is true even if the size of models (variables, functions) is a priori limited: recall Figures 1 and 2, which illustrate the potential difficulty of merely two-dimensional, box-constrained models. Of course, this does not mean that models of practical relevance can not be tackled successfully; however, one may have to (in fact, typically should) rely on modeling insight, intelligent – and perhaps model-dependent – combinations and/or adaptations of GO approaches. As a general guideline, even a rudimentary global scope search can lead to better solutions than the most sophisticated local search method started from the “wrong valley”... (Recall again Figures 1 and 2.)

The solution strategies discussed in this volume illustrate the above points by encompassing a range of practically viable methods. The contributing authors have made an honest effort to illustrate not only the successes but also the difficulties and the current limitations of practical global optimization.

Specifically, the chapters discuss both rigorous (theoretically globally convergent) and heuristic GO approaches such as
• adaptive random search
• branch-and-bound strategies
• evolutionary search approaches
• flexible simplex search heuristics
• genetic algorithms
• hybrid (stochastic-deterministic) approaches
• mixed integer nonlinear programming methods
• neural networks
• simulated annealing
• stochastic simulation
• "traditional" local nonlinear optimization

The methods listed above can be broadly categorized as follows:

• theoretically rigorous with a deterministic guarantee of global convergence (branch-and-bound, "exact" mixed integer GO, and other approaches)
• theoretically rigorous with a probabilistic guarantee of global convergence (adaptive random search, properly designed combined stochastic-deterministic approaches, properly implemented simulated annealing, and others)
• "obvious" extensions of traditional local search methodology (such as a limited globalized search effort combined with local optimization)
• heuristic direct search methods (e.g., flexible simplex search)
• metaheuristics (evolutionary search, genetic algorithms, neural networks)

Although the last three GO approaches do not have generally valid (provable) theoretical convergence properties in the continuous GO context, such heuristic methods still can be very useful in practice. Note furthermore that these heuristic methods can be adapted to guarantee at least stochastic global convergence. In addition to the already mentioned references on rigorous deterministic or stochastic global optimization, a few useful references on heuristic methods are Glover and Laguna (1997), Goldberg (1989), Michalewicz (1996), Osman and Kelly (1996), Rothlauf (2002), Rudolph (1997), Voss, Martello, Osman, and Roucairol (1999). It is worthwhile pointing out that although so far most heuristic approaches have been designed to solve combinatorial (discrete) optimization models, such methods can be adapted to tackle also continuous models.

Let us also remark that the GO literature offers a growing number of sufficiently detailed comparative numerical studies which shed light on the applicability of the most prominent methods and software to models of realistic complexity, in addition to "standard" GO test models that have been used for a few decades. Consult e.g. the test model library compiled by Floudas, Pardalos, Adjiman, Esposito, Gumus, Harding, Klepeis, Meyer, and Schweiger (1999), as well as topical expositions by Pintér (2002, 2003), Ali, Khompatraporn, and Zabinsky (2005), Khompatraporn, Pintér, and Zabinsky (2005). The websites by Fourer (2005), Mittelmann and Spellucci (2005), Neumaier (2005), and the
Optimization Technology Center (2005) also offer useful information regarding this point.

The individual chapters of the present volume have been written with the objective of addressing both experts and non-experts in the specific application area discussed. Therefore the authors attempted to follow a "tutorial" style, providing sufficient background to the key issues, model formulation and solution approaches presented. Most chapters are strongly application-oriented, in accordance with the overall objectives of this book.

We trust that our work will be of interest to researchers and practitioners in academia, research and consulting organizations, and industry. The book presents GO challenges and real-world case studies in sufficient detail, to enable graduate level classroom discussions and independent studies. The book can also be used in the framework of a practically motivated seminar or lecture series on nonlinear modeling and optimization.

The contributing authors and myself welcome your comments and suggestions related to this volume. Thanks for your attention, and enjoy the book!

János D. Pintér
PCS Inc.
Halifax, NS, Canada
References to Preface

This illustrative list of references includes books, software systems, and websites that are (mostly or at least partially) related to the practice of nonlinear systems modeling and global optimization, including software and applications. Many of the websites listed below offer extensive further information of relevance.


Acknowledgments

First of all, I would like to thank all contributing authors for their work, patience, and support that made this edited book project possible.

I also thank all reviewers for providing constructive comments that helped to improve both contents and style of the contributions. The following colleagues served as reviewers of one or several chapters: Luke Achenie, Julio Banga, George Corliss, Jon Doye, Bernd Hartke, Glenn Isenor, Andrzej Osyczka, Brahim Rekiek, Graham Wood, Quan Zhu, and Francesco Zirilli, in addition to the editor.

Baker Kearfott and Arnold Neumaier served as reviewers for the entire edited volume: thanks for your constructive comments.

I wish to thank Springer, in particular John Martindale and Robert Saley, for supporting and managing this project.

Last but very far from least, I thank Susan for her love and quiet, but ever-present, support.

JDP
Abstract: Solvent design can be modeled as a mixed integer nonlinear programming problem (MINLP) in which discrete variables denote the presence or absence of molecular structural entities and to what extent they occur in the pure component compound or mixture. On the other hand, continuous variables denote process variables such as temperature and flow rates. In the MINLP model the number of discrete variables can range from several tens to several hundreds. Therefore the use of the standard branch-and-bound method for solving the problem can be computationally intensive since all the variables (discrete and or continuous) must be used as branching variables. To overcome this problem, we have proposed a new strategy in which branching is done using branching functions instead of all the search variables. This approach results in a decrease in the number of branching variables. During branch and bound, the bounding operation is performed in the search variables space, while the branching operation is performed in a reduced dimension space defined by the branching (or splitting) functions. The branching functions are determined from the special tree function representation of both the objective function and constraints. The suggested MINLP solution approach is demonstrated on a solvent design application.
1. INTRODUCTION

Chemical product design addresses the design of single component chemical compounds and/or mixtures (blends) of compounds with pre-specified thermo-physical properties. In recent years, the traditional wet chemistry based chemical product design is being supplemented with computer-aided approaches, namely computer-aided molecular design (CAMD). The CAMD problem can often be posed as a mathematical program in which a number of binary and continuous variables define the search space (Duvedi and Achenie, 1996; Churi and Achenie, 1996; Maranas, 1997; Odele and Machietto, 1993; Pistikopoulos and Stefanis, 1998). A binary variable is an integer variable that can have one of two possible values, for example 0 and 1. This chapter discusses a globally optimal branch and bound approach to solving the resulting mathematical program. The approach is more fully discussed in a similar chapter in Sinha et al., 2002.

2. PROBLEM DEFINITION

A typical molecular design problem may be modeled as a single objective minimization or maximization subject to structural and performance constraints. Thus a CAMD problem for single component molecular design in which thermo-physical property matching is sought may be modeled as

$$\min_{x,v,\theta} f(x, v, \theta)$$

$$\varphi_j (x, v, \theta) \leq 0, \quad j = 1, \ldots, m_1$$

$$h_i (x, v, \theta) = 0, \quad i = 1, \ldots, m_2$$

where \(v\) is a vector of binary variables that define the molecular structure, \(x\) is a vector of continuous variable such as process variables (pressure, temperature, etc.) and \(\theta\) is a vector of group contribution parameters. Note that additional binary variables may be included in \(v\) to indicate additional constraints on the kind of molecular structures that can be generated. \(f(x, v, \theta)\) is the performance objective function (for example some undesirable property such as a compound's ozone depletion potential). The
Global Optimization in Solvent Design

group contribution model is a structure-property correlation that has found wide use in the chemical process industry.

The constraints involve (a) structural feasibility, (b) physical property targets, and (c) process constraints. The constraints associated with structural feasibility are usually linear. Physical property targets often have the form \( p_k^L \leq p_k(x,v,\theta) \leq p_k^U \). If \( p_k(x,v,\theta) \) is modeled using group contribution, then it may have the form

\[
p_k = \sum_j n_j \theta^1_j \left/ \sum_j n_j \theta^2_j \right. .
\]

Here \( \theta^1_j \) and \( \theta^2_j \) are elements in \( \theta \) and \( n_j \) is the number of \( \theta^1_j \) or \( \theta^2_j \) present in the molecule. Transformation of such constraints into a linear form is straightforward. The function \( p_k(x,v,\theta) \) can also have the form

\[
p_k = f_{NL}^1 \left( \sum_j n_j \theta^a_j \right) \left/ f_{NL}^2 \left( \sum_j n_j \theta^b_j \right) \right.
\]

where \( f_{NL}^1 \) and \( f_{NL}^2 \) are nonlinear functions; in addition \( \theta^a_j \) and \( \theta^b_j \) are parameters. Property constraints, which employ the given form, include solubility parameter based models often used in solvent design. It is not always possible to reformulate these constraints into linear or convex forms.

The nonlinear mathematical programming model for the CAMD problem (PMD) has the following features: (a) it is a nonconvex mixed integer nonlinear problem (MINLP) problem involving a large number of binary variables, (b) the number of linear constrains is larger than the number of nonlinear constraints, and (c) most of the components of the design vector \( u \) participate in the nonlinear terms. Previous attempts using global optimization are either geared to small size problems or use soft computing approaches (such as simulated annealing and genetic algorithms). The approach discussed here is based on the branch and bound (BB) algorithm. The basic BB algorithm may encounter a large number of branching variables for product design problems. To address this, the branch-and-bound global optimization algorithm presented here exploits the problem structure and allows significant reduction in branching expressions. A discussion of the algorithm is based on the papers (Sinha, Achenie and Ostrovksy, 1999) and (Ostrovksy, Achenie and Sinha, 2000).

In group contribution based computer aided single component product design, solvents are formed from certain combinations of a set of structural groups. The pre-specified set of \( m \) structural groups is called the basis set. The size and composition of the basis set depends on the intended
application, the availability of accurate property prediction models and the computational resources available. First, we define a set of variables based on an initial set of structural groups as

\[
u_{ik} = \begin{cases} 
1 & \text{if the $i$-th group in the molecule is the $k$-th structural group in the basis-set} \\
0 & \text{otherwise}
\end{cases} \quad \text{Churi - Achenie model}
\]

The Churi-Achenie model uses variables defined as above to represent the presence or absence of a group in the molecule or in the basis-set. However, it does not consider the information that determines how the groups are connected to each other in the molecule.

Odele and Machietto (1993) proposed a formulation that ensured that the valence of each structural group was satisfied. This formulation only accounts for the presence and absence of structural groups in the molecule. However, it does not consider the information that determines how the groups are connected to each other. To overcome this limitation, Churi and Achenie (1996) proposed a model that gives complete information with regard to how the groups are connected to each other. Presently there is no known group contribution method that takes advantage of the connectivity information of the Churi-Achenie model. In the latter model, the following variables were introduced:

\[
u_i = \begin{cases} 
1 & \text{if the $i$-th structural group in the basis-set is in the molecule} \\
0 & \text{otherwise}
\end{cases} \quad \text{Odele - Machietto model}
\]

\[
z_{ijp} = \begin{cases} 
1 & \text{if the $i$-th group's $j$-th site is attached to the $p$-th group} \\
0 & \text{otherwise}
\end{cases}
\]

\[
w_i = \begin{cases} 
1 & \text{if the $i$-th group in a molecule does not have a group attached} \\
0 & \text{otherwise}
\end{cases}
\]
For single component solvents structural constraints are imposed for (a) limiting the number of structural groups in a molecule; (b) ensuring that the number of bonds attached to a group equals the valence of the group; and (c) ensuring that each group in a molecule is attached to at least one other group. The formulation is effective in specifying whether the molecule is acyclic or cyclic. Moreover the maximum number of cycles can also be controlled. This representation is also effective in distinguishing between isomers. If the chemical process is not accounted for, then the pure component molecular design problem involves only binary variables. The maximum number of groups in a molecule is \( n_{\text{max}} \); the number of groups in the basis set is \( m \) with the maximum valence of \( s_{\text{max}} \). In this case the search dimension is then given by \( n_{\text{max}} \times m + n_{\text{max}} \times s_{\text{max}} \). Here the number of binary variables is equal to the sum of the dimensions of \( u \), \( z \) and \( w \), respectively (assuming the Churi-Achenie model is used). The number of linear structural constraints employed are \( n_{\text{max}}^2 + n_{\text{max}} \times m + 3n_{\text{max}} + s_{\text{max}} + 1 \). For example, a CAMD problem with \( n_{\text{max}} = 5 \), \( m = 10 \), and \( s_{\text{max}} = 2 \) results in 93 linear constraints. The number of nonlinear constraints is generally small compared to the number of linear constraints. Let all the binary variables in the problem be assembled in the vector \( v \) (q-dimensional). If the Odele-Machietto model is employed then \( v = u \); on the other hand if the Churi-Achenie model is employed then \( v = [u, z, w] \). Then the solvent design problem (see Eq. (1), (2), (3)) can be expressed compactly as a mixed integer nonlinear program in the general form

\[
P: \quad f = \min_{x, v \in D} f(x, v) \tag{6}
\]

such that

\[
D = \{ x, v : c \leq x \leq d, \varphi_i(x, v) \leq 0, \quad i = 1, \ldots, m, \quad h(x, v) = 0, \quad x \in X \subseteq \mathbb{R}^n, \quad v \in \{0,1\}^q \}
\]
3. PROPOSED SOLUTION METHOD

3.1 Branch-and-Bound Preliminaries

The branch and bound (BB) method (Horst and Tuy, 1990) has been used for solving several problems in chemical engineering (Ostrovsky et al., 1990, Friedler et al., 1998, Quesada and Grossmann, 1995, Ryoo and Sahinidis, 1996, Maranas and Floudas 1997, Adjiman et al., 1998). The generic BB method looks for a minimum of the objective function $f(x,v)$ by partitioning the region $D$ into subregions $D_i$ with respect to the search variables. At each iteration, a subregion $D_i$ is further partitioned into $D_{ip}$ and $D_{iq} (D_i = D_{ip} \cup D_{iq})$. The generic BB method consists of the following:

(i) An algorithm for estimating a lower bound (LB) $\mu_i$ on the objective function $f(x,v)$ in any subregion $D_i \in D$ such that $\mu_i \leq f(x,v) \ \forall x, v \in D_i$

(ii) An algorithm for estimating an upper bound (UB) $\eta_j$ on $f(x,v)$ in any $D_j \in D$ such that $\eta_j \geq f(x,v) \ \forall x, v \in D_j$

(iii) An algorithm for partitioning $D_i$

Designate the set of subregions at the k-th iteration of the BB method as $L^{(k)} = \{ D_i, i = 1, ..., N_k \}$. Let $I^{(k)}$ be the index set of the subregions belonging to $L^{(k)}$. Then the algorithm for the BB method is as follows:

Step 1: Set $k=1$. Give an initial set $L^{(0)}$ of the subregions $D_i$ ($i=1,...,N_0$, usually $N_0=1$).

Step 2: Calculate an LB for each $D_i \in L^{(k)}$

Step 3: Determine the subregion with the least LB. Let it be the $l_{im}$-th region then

$$\mu_{l_{im}} = \min_{l \in I^{(k)}} \mu_l$$

(7)
Step 4: Split $D_{lm}$ into two subregions $D_p$ and $D_q$ ($D_{lm} = D_p \cup D_q$) such that

$$D_p = \{ x : x \in D_{lm}, x_s \leq c_s \}, \quad D_q = \{ x : x \in D_{lm}, x_s \geq c_s \}$$

The variable, $x_s$, is the branching variable and $c_s$ is the branching point.

Step 5: Determine LB and UB for subregions $p$ and $q$.

Step 6: Determine the least upper bound $\eta^{(k)}$ at the $k$-th iteration.

$$\eta^{(k)} = \min (\eta^{(k-1)}, \eta_p, \eta_q)$$

For the first iteration $\eta^{(0)} = \infty$

Step 7: If $\eta^{(k)} - \mu_{lm} \leq \varepsilon$ then STOP.

Step 8: If

$$\mu_j > \eta^k$$

is met for $j = p$ or $j = q$ then the corresponding subregion is eliminated from consideration.

Step 9: Form a new set $L^{(k)}$ of the remaining subregions as follows

$$L^{(k)} = (D_1, \ldots, D_{l_{n-1}}, D_{l_{n+1}}, \ldots, D_{N_{l-1}}, \overline{L})$$

where

$$\overline{L} = \begin{cases} D_p, D_q & \text{if } \mu_j \leq \eta^{(k)} \quad j = p, q \\ D_q, & \text{if } \mu_q < \eta^{(k)} < \mu_p \\ D_p, & \text{if } \mu_q < \eta^{(k)} < \mu_p \end{cases}$$
Step 10: Set \( k = k + 1 \), and go to Step 3

Each BB method needs to develop algorithms for partitioning and for estimating lower and upper bounds. Thus we describe algorithms we have developed for estimating lower and upper bounds for the mixed integer nonlinear program arising from our formulation of the computer aided molecular design problem. Let us consider the partitioning algorithm. At each iteration in a standard BB method, the "optimal" subregion \( D_i \) is partitioned into two subregions \( D_p \) and \( D_q \) using the constraints \( x_i \leq x_i^* \)
and \( x_i \geq x_i^* \) or \( v_i \leq v_i^* \) and \( v_i \geq v_i^* \) as follows

\[
D_p = \{ x : x \in D_{lm}, x_s \leq c_s \}, D_q = \{ x : x \in D_{lm}, x_s \geq c_s \}
\]

The variable, \( x_s \), is the branching variable and \( c_s \) is the branching point. Different BB methods have different ways of selecting these. Thus in this case \( n+q \) branching variables are used. In a realistic product design problem, the number of branching variables can be several hundred. It is known that the number of branching nodes grows exponentially. To alleviate this problem, we will use the following new partition algorithm. Instead of branching on the variables \((x,v)\), we will use appropriate functions \( \psi_j(x,v), j = 1, \ldots, p \) of the search variables for branching. Subsequently, \( D_i \) will be determined by the set of inequalities

\[
a_j^l \leq \psi_j(x,v) \leq b_j^u, j = 1, \ldots, p,
\]

where the lower and upper bounds \( a_j^l \) and \( b_j^u \) are the dimensions of the multidimensional box (subregion) \( D_i \) are determined by the branch-and-bound strategy. Thus \( D_i \) has the form

\[
D_i = \{ x, v : x, v \in D; a_j^l \leq \psi_j(x,v) \leq b_j^u, j = 1, \ldots, p, \}
\]

Problem \( \mathbf{P} \) for subregion \( D_i \) is written as

\[
\mathbf{P}_i^1: \quad f_i = \min_{x, v \in D_i} f(x,v)
\]

A direct solution of the above problem is very difficult. Instead, the approach to be described finds the solution indirectly by successively
estimating lower and upper bounds for the performance objective function $f_i$. In the limit, these bounds should collapse into one to give a solution to the above problem. Thus it is appropriate to discuss how these bounds are obtained.

### 3.2 Lower Bound Algorithm

A lower bound $f_i^L$ for $f_i$ on $D_i$ is obtained by solving the following problem

$$
\begin{align*}
\textbf{P}_i^L: \quad f_i^L &= \min_{x,v \in \tilde{D}_i} L[f(x,v); \tilde{D}_i] \\
\text{where} \\
\tilde{D}_i &= \{ x,v : L[\phi_k; D_i] \leq 0 \quad k = 1, \ldots, m; \\
L[\psi_j; D_i] &\leq b_j^i; \quad L[-\psi_j; D_i] \leq -a_j^i; \\
v \in \{0,1\}^q \}
\end{align*}
$$

and $L[g(x,v); D_i]$ is a convex underestimator for the generic function $g(x,v)$. Then it is easy to verify that $D_i \subseteq \tilde{D}_i$.

Some alternatives for estimating lower bounds are: (a) The use of linear or convex nonlinear underestimators; (b) Enforcing the integrality of all the binary variables $v$ at each iteration (Pantelides, 1996); (c) The variables $v$ are considered as continuous variables such that $0 \leq v \leq 1$. In the latter, the variables become binary only at termination of the algorithm. We will construct linear underestimators and we will enforce integrality of $v$ at each iteration as in (b). The resulting problem (PiL) is a mixed integer linear program (MILP).

### 3.3 Upper Bound Algorithm

The upper bound $f_i^U$ for $f_i$ on $D_i$ can be found by computing $f_i^U = f(\overline{x}, v)$, where $[\overline{x}, v]$ is a feasible point for problem (10). The latter can be obtained by solving