PROCEEDINGS OF THE GLOBAL OPTIMIZATION WORKSHOP 2012

Edited by

Daniel Aloise
Universidade Federal do Rio Grande do Norte, Natal, Brazil

Pierre Hansen
HEC Montreal and GERAD, Montreal, Canada

Caroline Rocha
Universidade Federal do Rio Grande do Norte, Natal, Brazil

Preface

The methodological advances in solving global optimization problems has allowed us to consider problems which a few years ago were considered intractable. Either through the use of deterministic or stochastic algorithms, the ability to solve multiextremal global optimization is of practical interest for a variety of applications in biology, engineering, finance, chemistry, etc. Consequently, coupled with increasing computing power, we have seen the creation of new global optimization algorithms and methods.

The Global Optimization Workshop is a forum for both academic and industrial communities to present and discuss the latest results and challenges in global optimization. Global Optimization Workshops are organized periodically by members of the Global Optimization scientific community. The preceding workshops have been periodically held as follows: Szeged (Hungary 1995), Florence (Italy, 1999), Hanmer Springs (New Zealand, 2001), Santorini (Greece, 2003), San José (Spain, 2005), Mykonos (Greece, 2007), Skukuza (South Africa, 2008), and Toulouse (France, 2010). The Global Optimization Workshop (GOW 2012) in Natal, Brazil, maintains this tradition. As is customary, a special issue of the Journal of Global Optimization will be dedicated to this workshop.

The organizing committee gratefully acknowledges financial support from the Brazilian Agencies Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) and Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES). In addition, we acknowledge support from the Groupe de d'Études et de Recherche en Analyse des Décisions (GERAD).

We would like to thank the invited speakers and the authors in advance for their participation and we are looking forward to a hopefully enjoyable and enriching workshop which will contribute to this interesting field and important community.

Daniel Aloise (Natal, Brazil)
Pierre Hansen (Montreal, Canada)
Caroline Rocha (Natal, Brazil)
GOW 2012 Co-chairs

Program Committee

Daniel Aloise	UFRN	Brazil
Pierre Hansen	HEC Montreal and GERAD	Canada

Scientific Committee

Charles Audet	Montreal, Canada
Pietro Belotti	Clemson, USA
Sonia Cafieri	Toulouse, France
Manoel Campelo	Fortaleza, Brazil
Emilio Carrizosa	Sevilla, Spain
Marcia Fampa	Rio de Janeiro, Brazil
Xavier Gandibleux	Nantes, France
Eligius Hendrix	Malaga, Spain
Marco Locatelli	Parma, Italy
Nelson Maculan	Rio de Janeiro, Brazil
Marcel Mongeau	Toulouse, France
Dominique Orban	Montreal, Canada
Sylvain Perron	Montreal, Canada
Maurício G. C. Resende	Florham Park, USA
Celso C. Ribeiro	Rio de Janeiro, Brazil
Caroline Rocha	Natal, Brazil
Sebastian Sager	Heidelberg, Germany
Fabio Tardella	Rome, Italy
Ider Tseveendorj	Versailles, France
Robert Weismantel	Zürich, Switzerland
Tapio Westerlund	Abo, Finland

Local Organizing Committee

Daniel Aloise	Natal, Brazil
Caroline Rocha	Natal, Brazil
Dario José Aloise	Mossoró, Brazil
Lucídio dos Anjos Formiga Cabral	João Pessoa, Brazil
Mariana Almeida	Natal, Brazil
Pedro Fernandes Ribeiro Neto	Mossoró, Brazil
Samuel Xavier de Souza	Natal, Brazil

Sponsors

Conselho Nacional de Desenvolvimento Científico e Tecnológico - CNPq	Brazil
Comissão de Aperfeiçoamento de Pessoal de Nível Superior - CAPES	Brazil
Groupe d'Études et de Recherche en Analyse des Décisions - GERAD	Canada

Contents

Plenary Talks

Extended Abstracts

PLENARY TALKS

Discrete approaches for distance geometry and applications on molecular structure calculations

C. Lavor,[1] A. Mucherino,[2] L. Liberti,[3] N. Maculan[4]

[1]*IMECC-UNICAMP, Campinas-SP, Brazil,* clavor@ime.unicamp.br

[2]*IRISA, University of Rennes 1, Rennes, France,* antonio.mucherino@irisa.fr

[3]*LIX, École Poleytechnique, Palaiseau, France,* liberti@lix.polytechnique.fr

[4]*COPPE, Federal University of Rio de Janeiro, Rio de Janeiro-RJ, Brazil,* maculan@cos.ufrj.br

Abstract Finding energetically stable conformations of proteins is one of the most interesting and difficult problems in biology. Given the chemical composition of a molecule, its three-dimensional conformation is of interest because it is directly related to the function it is able to perform. Since some years, we have been working on an easier problem, where the identification of protein conformations is aided by information obtained by experiments of Nuclear Magnetic Resonance (NMR). Under certain assumptions, we are able to discretize the problem and use an efficient Branch & Prune (BP) algorithm for its solution. In this work, we move towards the more difficult situation in which the information given by NMR is not available. The original BP does not work in this case, because there is no information for performing its pruning phase, that is the strong point of the algorithm. In this paper, we study and present a new energy-based pruning device to be added to the BP algorithm. First computational experiences on a set of small homopolymers show that this approach is promising for the identification of low-energy conformations of molecules.

Keywords: homopolymer conformation, energy minimization, discretization, branch-and-prune

1. Introduction

Proteins are composed by smaller molecules called amino acids. They are able to perform several important functions in bodies of living beings, and the functions they perform are strongly dependent on their three-dimensional conformations. While modern technologies are currently able to easily identify the sequence of amino acids for a given protein, the identification of the corresponding three-dimensional conformation is still a real challenge.

There are different approaches to this problem, which depend on the information that are actually available and that can be efficiently used for its solution. Experimental techniques, such as the Nuclear Magnetic Resonance (NMR), are able to provide a subset of lower and upper bounds on some inter-atomic distances, and this information can be exploited for identifying possible three-dimensional conformations for a given molecule. However, if the only knowledge about the molecule is given by its chemical composition, the only way to find an approximation of its conformation is by identifying the most stable conformation from an energetic point of view.

Since years, we are working on the problem of identifying the conformation of a molecule by exploiting information on the distances that NMR experiments are able to provide (see [2–4, 6, 7]). This problem is known in the scientific literature as the Molecular Distance Geometry Problem (MDGP) [1], and it is an NP-hard problem. In this context, our main contribution is given by a discretization process that we can apply for reducing the search space from a continuous domain to a discrete one. Even if this transformation does not decrease the complexity of the problem (which is still NP-hard), it allowed us to conceive an efficient Branch & Prune (BP) algorithm for the solution of MDGPs that can be discretized. We named this class of problems Discretizable MDGP (DMDGP). An instance of the MDGP belongs to the DMDGP class if and only if some assumptions are satisfied [2].

The discretization is performed by intersecting three spheres centered in three consecutive atoms and having three known distances as radii. This intersection, with probability 1, gives two points only, which correspond to the possible coordinates for the successive atom in the sequence [2]. This allows to generate a tree where each layer contains the potential coordinates for the same atom. As a consequence, a solution to the DMDGP can be searched by identifying a path from the root node (representing the first atom) to one of the feasible leaf nodes (representing the last atom) of the tree. Each branch included in the solution path corresponds to a certain torsion angles ω (defined by 4 consecutive atoms of the molecule). The discretization is possible when the reference distances (the radii of the spheres) are exact, as well as when one of the three reference distances is represented by an interval [4].

Even for small-sized molecules, the complete construction of the tree can be too computationally expensive. However, distances (that NMR can provide and that are not exploited in the construction of the tree) can be used in the BP algorithm for *pruning* a part of the branches where there are no feasible solutions. The pruning phase in BP allows therefore to focus the search on branches of the tree where there can be solutions. In protein instances of the DMDGP, distances used for pruning are available and they allow to prune large parts of the tree, so that the problem can be solved very efficiently by the BP algorithm [7].

In a recent work [4], we proposed a special ordering for the atoms forming protein backbones: all instances in which the atoms are sorted accordingly to this special ordering belong to the DMDGP class. Moreover, all the distances that are necessary for applying the discretization process do not have to be computed by NMR experiments, but they can rather be obtained by simple observations on the chemical composition of protein backbones. Many distances are related to bonded atoms and other ones are related to atoms bonded to a common atom.

It is important to remark that, if this special ordering is employed, the discretization process is completely independent on the DMDGP instance at hand. Actually, any instance related to any protein backbone of a given size has the same tree as a search domain. NMR data can be used for selecting the branches of the tree that are compatible with the distances.

In this paper, we consider for the first time the possibility of replacing distance-based pruning devices with energy-based ones. Since our search trees are independent on NMR, these experiments are not necessary for performing the discretization. We can suppose therefore that the only available information about the molecule concerns its chemical composition. At this point, the only possibility for discarding the infeasible branches of BP trees is through energy-based criteria.

This work opens the doors to more difficult problems in biology for which we could supply a suitable discretization. We will present in Section 2 a new pruning device based on energetic criteria, and we will show some preliminary computational experiments in Section 3.

2. New pruning devices for BP

In the BP algorithm, the idea is to generate the possible atomic coordinates for each atom of the molecule, and to verify their feasibility right away after their generation [6]. In the original BP, the pruning phase is performed by exploiting information on a subset of distances that are not considered in the discretization process (supposed to be obtained through NMR experiments). In this work, we suppose instead that all distances necessary for the discretization are obtained by observations on the chemical structure of the molecule at hand, and that there is no information on other additional distances. Therefore, we need to consider the internal energy of the molecule and the pruning phase needs to be based on this energy.

An accurate description of all interactions among the atoms in a molecule can be very complex. There are however approximations of this potential energy that take into consideration the most important interactions. We will consider the same expression used in [8], given by:

$$E_n = E_{bond} + E_{angle} + E_{torsion} + E_{LJ}, \tag{1}$$

where

$$E_{bond} = \frac{1}{2}\sum_i k_d(d_i - d_0)^2, \qquad E_{angle} = \frac{1}{2}\sum_i k_\theta(\theta_i - \theta_0)^2,$$
$$E_{torsion} = \frac{V_{\bar{n}}}{2}\sum_i \left[1 + \cos(\bar{n}\omega_i + \delta)\right], \qquad E_{LJ} = \sum_{i,j}\varepsilon_{i,j}\left[\left(\frac{\sigma_{i,j}}{r_{i,j}}\right)^{12} - 2H_{i,j}\left(\frac{\sigma_{i,j}}{r_{i,j}}\right)^{6}\right].$$

The term E_{bond} considers the energy given by the interaction between two bonded atoms. Depending on the kind of atoms, indeed, there is a typical inter-atomic distance for the two atoms. Any modification on this distance makes the energetic term positive. The term models therefore the repulsive force between the two atoms if they get too close to each other, as well as the attractive force between them in case they get too far. The parameter k_d is the "bond stretching" constant, and d_0 is the preferred value for the inter-atomic distance.

The energetic term E_{angle} is used similarly to model the local interactions among three atoms X, Y and Z such that atom X is bonded to Y, and Y is bonded to Z. Depending on the kind of atoms, there is a typical local conformation for the three atoms that correspond to a typical angle between the segment XY and the segment YZ. The term E_{angle} ensures that the value of this angle is close to the typical one by penalizing any modifications (in both directions). The parameter k_θ is the "angle bending" constant, and θ_0 is the preferred value for the bond angle.

The third term $E_{torsion}$ allows to define a certain subset of preferred torsion angles ω for the considered conformations. For example, if $\bar{n} = 3$ and $\delta = 0$, then the preferred torsion angles are $60°$, $180°$ and $300°$. The energetic term gives a penalty to all other torsion angle values, and the penalty is proportional to the difference between the selected torsion angle and the closest among the preferred ones. $V_{\bar{n}}$ is a "torsional" constant, which depends on the choice of $\bar{n}$ [8].

Finally, the last term is the Lennard Jones potential [5]. $\varepsilon_{i,j}$ and $\sigma_{i,j}$ are two parameters that can be defined by the relationships between the pairs of atoms (or agglomerate of atoms) which are interacting. The parameter $H_{i,j}$ is related to the hydrophobicity and hydrophilicity of the interacting atoms. We will suppose that $H_{i,j}$ is always equal to 1.

Pruning devices have the difficult role of identifying the atomic positions from which infeasible branches start. During the execution of BP, every time a leaf node is reached, the full set of coordinates for a conformation is available, and hence the energy E_n for this conformation can be computed. Let us suppose that $\hat{E}_n$ is the lowest energy found so far. Our idea is to verify in advance whether new branches of the tree can actually contain conformations with an energy that can be smaller than $\hat{E}_n$. This is done by computing a lower bound on the energy concerning all the conformations belonging to a common branch.

The terms E_{bond}, E_{angle} and $E_{torsion}$ are always positive, and hence the lower bound for their values can be 0. E_{LJ} can be negative, but, depending on the range in which the inter-atomic distances can vary, we can compute an accurate lower bound for the actual value. Let us suppose that we are executing BP and that the current layer is the k^{th}, where we have a partial energy value $E_{n(\leq k)}$ (computed by using the available coordinates) and a lower bound $L_{(>k)}$ on the energy $E_{n(>k)}$. If $E_{n(\leq k)} + L_{(>k)} > \hat{E}_n$, then there is no hope to identify a conformation with an energy smaller than $\hat{E}_n$ by exploring the current branch of the tree. This branch can therefore be pruned. The same strategy can be applied for electrostatic potentials

3. Preliminary computational experiments

We present in this section some preliminary experiments. All codes were written in C programming language and all the experiments were carried out on an Intel(R) Xeon(TM) CPU 3.40GHz with 4GB RAM, running Linux. The codes have been compiled by the GNU C compiler v.4.1.1.

We consider homopolymers, that consist of strings of bonded atoms having the same chemical properties. Bond lengths and bond angles are considered fixed, so that the first two terms of the potential energy (1) disappear. All bond lengths are fixed to the preferred value

n	4	5	6	7	8	9	10	11	12
energy (kcal)	-0.12	-0.26	-0.42	-0.64	-0.96	-1.45	-1.99	-2.51	-3.27
time (no devices)	0.0s	0.0s	0.0s	0.6s	19.5s	9min 32s	4h 35min	$>$ 12h	$>$ 12h
time (energy-based device)	0.0s	0.0s	0.0s	0.3s	5.6s	1min 2s	9min 1s	1h 4min	10h 11min

Table 1. Experiments with homopolymers having size n ranging from 4 to 12.

$d_0 = 1.526$Åand all bond angles are fixed to $\theta_0 = 109^{\circ}\!.47$. Moreover, in the term $E_{torsion}$, $\bar{n} = 3$, $\delta = 0$ and $V_{\bar{n}} = 1.3$. Finally, in the Lennard Jones term, $\varepsilon_{i,j} = 0.181$ and $\sigma_{i,j} = 3.3$.

Table 1 shows some experiments with homopolymers having size n ranging from 4 to 12 (in [8], they consider a little larger instances, but use a much more stronger computer system).

When BP is executed without pruning devices, the computational cost is naturally very expensive. When the energy-based pruning device is instead employed, the same solutions can be identified in a shorter amount of time. For example, the same conformation with $n = 10$ and energy $E_n = -1.99$ kcal can be identified in 9 minutes when our new pruning device is employed, and in about 4 hours and half otherwise. Therefore, the proposed pruning device is actually able to identify and prune the branches of the tree where there cannot be conformations with a lower energy. Future works will be devoted to the development of other pruning devices and ad-hoc strategies for speeding up the search.

Acknowledgments

This work is partially supported by the ANR project "Bip:Bip". The authors would also like to thank FAPESP, CNPq (Brazil), CNRS and École Polytechnique (France) for financial support.

References

[1] G. Crippen, T. Havel, *Distance Geometry and Molecular Conformation*, Wiley, New York, 1988.

[2] C. Lavor, L. Liberti, N. Maculan, A. Mucherino, *The Discretizable Molecular Distance Geometry Problem*, to appear in Computational Optimization and Applications, 2012.

[3] C. Lavor, L. Liberti, N. Maculan, A. Mucherino, *Recent Advances on the Discretizable Molecular Distance Geometry Problem*, European Journal of Operational Research **219**, 698–706, 2012.

[4] C. Lavor, L. Liberti, A. Mucherino, *The interval Branch-and-Prune Algorithm for the Discretizable Molecular Distance Geometry Problem with Inexact Distances*, to appear in Journal of Global Optimization, 2012.

[5] J.E. Lennard-Jones, *Cohesion*, Proceedings of the Physical Society **43**, 461–482, 1931.

[6] L. Liberti, C. Lavor, and N. Maculan, *A Branch-and-Prune Algorithm for the Molecular Distance Geometry Problem*, International Transactions in Operational Research **15**, 1–17, 2008.

[7] L. Liberti, B. Masson, C. Lavor, A. Mucherino, *Branch-and-Prune Trees with Bounded Width*, Proceedings of the 10^{th} Cologne-Twente Workshop on Graphs and Combinatorial Optimization (CTW11), 189–193, 2011.

[8] A.T. Phillips, J.B. Rosen, V.H. Walke, *Molecular Structure Determination by Convex Underestimation of Local Energy Minima*, In P.M. Pardalos, D. Shalloway, and G. Xue (Eds.), Global Minimization of Nonconvex Energy Functions: Molecular Conformation and Protein Folding, American Mathematical Society **23**, 181–198, 1996.

Several stories involving quadratic functions in mathematical optimization

Jon Lee

Industrial and Operations Engineering, University of Michigan, Michigan, USA jonxlee@umich.edu

Quadratic functions are in some sense the simplest and most vexing functions to consider for moving beyond the linear world. I will survey a variety of situations in mathematical optimization where I have focused on quadratic functions, sometimes with good results, but always with some frustration. Subjects include combinatorial optimization, nonlinear integer programming and continuous global optimization.

Keywords: quadratic, integer nonlinear programming, global optimization, combinatorial optimization

Symmetry in Mathematical Programming

Leo Liberti

LIX, École Polytechnique, Palaiseau, France liberti@lix.polytechnique.fr

We survey techniques for automatically detect symmetries in mathematical programming formulations of several types (MILP, NLP, MINLP), and present some approaches to exploit them in order to reduce the number of nodes in Branch-and-Bound type algorithms.

Keywords: group theory, reformulation, graph symmetry, branch-and-bound

Interval Branch and Bound Algorithms: from theory to applications

Frédéric Messine

University of Toulouse, ENSEEIHT-IRIT, 2 rue Camichel, 31000 Toulouse, France, Frederic.Messine@n7.fr

Interval Branch and Bound based methods are well known to be reliable (in the numerical sense) to determine a global minimum of a non-linear and non-convex problem [3, 4, 7]. Indeed, no numerical error due to floating point computations can yield wrong results. Moreover, the accuracy on the solution is fixed by the users and the global minima provided by such based algorithms are certified.

In a first part, the main principle of interval Branch and Bound methods, including propagation techniques, will be presented and the way to certify the solutions will be discussed.

Affine arithmetic was introduced by Comba et al. in [1], it was extended by considering new forms by Messine in [5]. The way to compute reliable bounds using affine relaxations associated with a rigorous resolution of linear programs due to Neumaier and Shcherbina [6] yields to improve the efficiency of such based interval algorithms.

A code named *IBBA* (for Interval Branch and Bound Algorithm) was developed and was used to solve MINLP (Mixed-Integer Non-Linear Problem) arising in electromagnetism for the design of electrical machines, [2].

Keywords: Interval Analysis, Affine Arithmetic, Affine Relaxation, Constraint Propagation, Reliable Global Optimization

References

[1] J. L. D. Comba and J. Stolfi, *Affine arithmetic and its applications to computer graphics*, in Proceedings of SIBGRAPI'93 - VI Simpósio Brasileiro de Computação Gráfica e Processamento de Imagens, pp. 9–18, 1993.

[2] E. Fitan, F. Messine, and B. Nogarede, *The Electromagnetic Actuator Design Problem: A General and Rational Approach*, IEEE Transactions on Magnetics, Vol. 40, N. 3, pp. 1579–1590, 2004.

[3] E.R. Hansen and W. Walster, *Global Optimization using Interval Analysis*. Marcel Dekker Incorporated, New York, 2003.

[4] R.B. Kearfott, *Rigorous Global Search: Continuous Problems*, Kluwer Academic Publishers, Drodrecht, Boston, London, 1996.

[5] F. Messine, *Extentions of affine arithmetic: Application to unconstrained global optimization*, Journal of Universal Computer Science, Vol. 8, pp. 992–1015, 2002.

[6] A. Neumaier and O. Shcherbina, *Safe bounds in linear and mixed-integer linear programming*, Mathematical Programming, Vol. 99, No. 2, pp. 283–296, 2004

[7] H. Ratschek, J. Rokne, *New Computer Methods for Global Optimization*, Ellis Horwood, Chichester, 1988.

Proceedings of GOW 2012, pp. 13 – 13.

Recent advances in Variable Neighborhood Search metaheuristic

Nenad Mladenović

School of Mathematics, Brunel University, West London, United Kingdom nenad.mladenovic@brunel.ac.uk

Variable neighborhood search is metaheuristic whose main idea is change of neighborhood structures in both, local search step for intensification and in shaking step for diversification of the search for better solution. In this talk I will present some new vatiants of this methodology for solving continuous and discrete global optimization problems.

Global optimality conditions in non-convex optimization

Panos M. Pardalos[1,2] and Pando G. Georgiev[2]

[1]*Center for Applied Optimization (CAO), University of Florida, Gainesville, USA*

[2]*Laboratory of Algorithms and Technologies for Networks Analysis (LATNA)*
National Research University, Higher School of Economics
20 Myasnitskaya Str., Moscow, 101000, Russia

pardalos@ufl.edu, pandogeorgiev@gmail.com

In this talk we are going to present recent results regarding global optimality conditions for general non-convex optimization problems. First we are going to discuss complexity issues regarding the existence of points satisfying optimality conditions and the connection to complementarity problems. In addition, we are going to discuss surprising connections between optimality conditions and continuous formulations of discrete optimization problems.

In the second part of the talk we are going to discuss our recent result regarding optimality conditions of locally Lipschitz functions. Namely, we show how the necessary conditions for a local minimum can be used to obtain a sufficient optimality condition of first order for a global minimum of a locally Lipschitz function on a closed convex set in a Banach space. Using a theorem of F. Clarke, we obtain a short proof and an extension to Banach spaces of a result of J.-B. Hiriart-Urruty and J.S. Ledyaev. This result generalizes previous work of A. Strekalovsky and M. Dür, R. Horst, and M. Locatelli.

References

[1] A. Chinchuluun, A. Migdalas, P.M. Pardalos, and L. Pitsoulis (Eds.) Pareto Optimality, Game Theory and Equilibria, Springer, (2008).

[2] R. Horst, P.M. Pardalos and N.V. Thoai, Introduction to Global Optimization, Second Edition, Nonconvex Optimization and its Applications Vol. 48, Kluwer Academic Publishers, (2000).

[3] P. Georgiev, A. Chinchuluun and P. Pardalos, Optimality conditions of first order for global minima of locally Lipschitz function, Optimization, Volume 60, Issue 1 & 2 (January 2011), pp. 277 - 282.

[4] L. Gibbons, D. Hearn, P. Pardalos and M. Ramana, A continuous characterization of the maximum clique problem, Mathematics of Operations Research Vol. 22, No. 3 (1997), pp. 754- 768.

[5] A. Chinchuluun, P.M. Pardalos, and E. Rentsen, *A Numerical Method for Concave Programming Problems*, In: Jeyakumar, V. and Rubinov, A., editors, Continuous Optimization: Current Trends and Modern Applications, Springer, (2005), pp. 251–272.

[6] F. H. Clarke, *Optimization and nonsmooth analysis*, (Classics in Applied Mathematics), SIAM, 1990.

[7] M. Dür, R. Horst, and M. Locatelli, *Necessary and Sufficient Global Optimality Conditions for Convex Maximization Revisited*, Journal of Mathematical Analysis and Applications, 217 (1998), pp. 637–49

[8] I. Ekeland, *On the Variational Principle*, J. Math. Anal. Appl., 47(1974), pp. 324–353

[9] P. G. Georgiev, P. M. Pardalos, and A. Chinchuluun, *Localization of minimax points*, J. Global Optimization, 40 (2008), pp. 489-Ű494.

[10] J.-B. Hiriart-Urruty and J.S. Ledyaev, *A note on the characterization of the global maxima of a (tangentially) convex function over a convex set*, J. of Convex Analysis, 3 (1996), 55Ű61.

[11] A.S. Strekalovsky, *On the global extremum problem*, *Soviet Dokl.*, 292 (1987), pp. 1062–1066 [in Russian]

[12] A.S. Strekalovsky, *Global Optimality Conditions for Nonconvex Optimization*, Journal of Global Optimization, 12 (1998), pp. 415–434.

EXTENDED ABSTRACTS

A column generation algorithm for semi-supervised minimum sum-of-squares clustering

Daniel Aloise[1], Pierre Hansen[2], and Caroline Rocha[1]

[1]*Universidade Federal do Rio Grande do Norte, Campus Universitário s/n, 59072-970, Natal-RN Brazil*
aloise@dca.ufrn.br, caroline.rocha@ect.ufrn.br

[2]*GERAD and HEC Montréal, 3000, Chemin de la Côte-Sainte-Catherine, Montréal, Québec, H3T 2A7, Canada*
pierre.hansen@gerad.ca

Abstract Clustering is a powerful tool for automated analysis of data. It addresses the following problem: given a set of entities find subsets, called clusters, which are homogeneous and/or well separated. In addition to the entities themselves, in many applications, information is also available regarding their relations in the space. Once a priori knowledge is incorporated into the clustering learning process, we have a **semi-supervised** classification task. This work addresses a basic problem in semi-supervised clustering. It presents a column generation algorithm for minimum sum-of-squares clustering in the presence of must-link and cannot-link pairwise constraints. Computational experiments, including a comparison with Xia's algorithm [10], are reported.

Keywords: semi-supervised clustering, column generation, sum-of-squares

1. Introduction

Classification tasks can be categorized in two broad classes. In *supervised classification* there exists a data training set, for which classes are known, and the objective is to infer a discriminant function that correctly classifies it. Posteriorly, the same funcion is used to classify new data, *hopefully* with high accuracy. When no background knowledge is available, hidden structures are sought from data by *unsupervised classification* procedures. Clustering is a major representative of unsupervised learning approaches. It deals with the problem of given a set of entities find subsets, called *clusters*, which are homogeneous and/or well separated. Homogeneity means that entities in the same cluster are similar and separation that entities in different clusters must differ one from another. Many different criteria are used in the literature to express homogeneity and/or separation of the clusters to be found. Among them, a frequently used one is the minimum sum of squared Euclidean distances from each entity to the centroid of the cluster to which it belongs.

One of the most frequently used types of clustering is *partitioning*, where given a set $V = \{v_1, v_2, \ldots, v_n\}$ of n entities we look for a partition $P_k = \{C_1, C_2, \ldots, C_k\}$ of V into k clusters such that: (i) $C_j \neq \emptyset \quad j = 1, 2, \ldots, k$; (ii) $C_i \cap C_j = \emptyset \quad i, j = 1, 2, \ldots, k$ and $i \neq j$; and (iii) $\bigcup_{j=1}^{k} C_j = V$. Partitioning n entities into k clusters with this criterion is known as minimum sum-of-squares clustering (MSSC).

Although clustering is often seen as an unsupervised classification task, data analysts have very often some knowledge about the data itselt which could be used to yield a clustering more coupled with expert's data representation. *Semi-supervised clustering* incorporates this domain knowlege, provided by experts, into the learning process.

Instance-level constraints are a typical way to express a priori knowledge about which data entities should or should not be clustered together [9]. A *must-link* constraint obliges two entities to be clustered together while a *cannot-link* constraint assures that two entities are placed in different clusters. In this work, we will present a column generation algorithm for semi-supervised MSSC in the presence of instance-level constraints, denoted here semi-supervised MSSC. Section 2 presents the different mathematical formulations used to approach the problem as well as the fundamentals of our column generation algorithm. Section 3 presents preliminary computational results. Finally, conclusions are given in Section 4.

2. Methodology

A mathematical programming formulation of the semi-supervised MSSC is as follows:

$$\begin{aligned}
\min_{x,y} \quad & \sum_{i=1}^{n}\sum_{j=1}^{k} x_{ij}\|p_i - y_j\|^2 && && (1)\\
s.t. \quad & \sum_{j=1}^{k} x_{ij} = 1, && \forall i = 1,\ldots,n && (2)\\
& x_{i_1 j} = x_{i_2 j} && \forall (i_1,i_2) \in \mathcal{M}, \forall j = 1,\ldots,k && (3)\\
& x_{i_1 j} + x_{i_2 j} \le 1 && \forall (i_1,i_2) \in \mathcal{C}, \forall j = 1,\ldots,k && (4)\\
& x_{ij} \in \{0,1\} && \forall i = 1,\ldots,n; \forall j = 1,\ldots,k && (5)\\
& y_j \in \mathbb{R}^s && \forall j = 1,\ldots,k. && (6)
\end{aligned}$$

The n entities $V = \{v_1, v_2, \ldots, v_n\}$ to be clustered are at given points $p_i = (p_i^r, r = 1,\ldots,s)$ of $\mathbb{R}^s$ for $i = 1,\ldots,n$; k cluster centers must be located at unknown points $y_j \in \mathbb{R}^s$ for $j = 1,\ldots,k$ (the number of entities n is assumed to be greater than k, otherwise the problem is trivially solved by locating one cluster center at the position of each entity); the norm $\|\cdot\|$ denotes the Euclidean distance between the two points in its argument in the s-dimensional space under consideration. The decision variables x_{ij} express the assignment of the entity v_i to the cluster j. Thus, constraints (2) assure that each entity is assigned to exactly one cluster. Sets $\mathcal{M}$ and $\mathcal{C}$ contain pairs of entities involved in must-link and cannot-link constraints, represented by (3) and (4), respectively.

From Huygens' theorem, which states that the sum of squared distances from all entities of a given cluster to its centroid is equal to the sum of squared distances between pairs of entities of this cluster divided by its cardinality, the objetive function (1) can be expressed by

$$\min_x \sum_{j=1}^{k} \frac{\sum\limits_{i_1=1}^{n-1}\sum\limits_{i_2=i_1+1}^{n} \|p_{i_1} - p_{i_2}\|^2 x_{i_1 j} x_{i_2 j}}{\sum\limits_{i=1}^{n} x_{ij}} \qquad (7)$$

As proposed in [7], a must-link constraint with $(i_1, i_2) \in \mathcal{M}$ can be removed from the model if we replace entities v_{i_1} and v_{i_2} by a single *superentity* $v_{i_1 i_2}$. Whenever this is done, we reduce by one the number of variables in the model and coefficients are accordingly updated. In the case of a cannot-link constraint with $(i_1, i_2) \in \mathcal{C}$, it suffices to set coefficient $\|p_{i_1} - p_{i_2}\|^2$ to an arbitrary large value M.

Partitioning problems in cluster analysis can also be mathematically formulated by considering all possible clusters (i.e. those satisfying must-link and cannot-link constraints). Let us consider any cluster C_t for which $a_{it} = 1$ if entity p_i belongs to cluster C_t, and 0 otherwise, and let us denote by y_t the centroid of points p_i such that $a_{it} = 1$. Thus, the cost c_t of cluster C_t can be written as $c_t = \sum_{i=1}^{n} \|p_i - y_t\|^2 a_{it}$. An alternative formulation for semi-supervised MSSC is then given by

$$\begin{aligned}
& \min_z \sum_{t\in T} c_t z_t \\
& \text{subject to} \\
& \sum_{t\in T} a_{it} z_t = 1, \quad \forall i = 1,\ldots,n \\
& \sum_{t\in T} z_t = k \\
& z_t \in \{0,1\} \quad \forall t \in T,
\end{aligned} \qquad (8)$$

where $T = \{1, \ldots, 2^n - 1\}$. The z_t variables are equal to 1 if cluster C_t is in the optimal partition and to 0 otherwise. The first set of constraints state that each entity belongs to one cluster, and the following constraint expresses that the optimal partition contains exactly k clusters.

This is a extremely large set partitioning problem with a side constraint, for which the number of variables is exponential in the number of entities. Therefore, it cannot be explicitly written and solved in a straightforward way unless n is small. The column generation method proposed in [1] for MSSC works with a reasonably small subset $T' \subseteq T$ of the columns in (6), i.e., with a *restricted master problem*. The method is combined with branch-and-cut in order to solve exactly (6) for instances with about 2000 entities, with n/k roughly equal to 10. For the Euclidean plane, the method is able to solve instances with n up to 2392 entities and $k \geq 2$.

Problem (6) is solved iteratively, augmenting the number of columns in the restricted master problem until optimality is proved with the columns at hand. Entering columns are found by solving an auxiliary problem, i.e., finding the list of entities of a cluster whose associated variable in (6) has negative reduced cost. The auxiliary problem is expressed by:

$$\sigma + \min_{v \in \{0,1\}^n} \frac{\sum_{i=1}^{n-1} \sum_{j=i+1}^{n} (\|p_i - p_j\|^2 - \lambda_i - \lambda_j) v_i v_j - \sum_{i=1}^{n} \lambda_i v_i}{\sum_{i=1}^{n} v_i}, \tag{9}$$

where λ_i, for $i = 1, \ldots, n$, and σ are the dual variables associated to constraints of (6). Problem (9) is a hyperbolic (or fractional) program in 0-1 variables with quadratic numerator and linear denominator. This problem is solved by an adaptation to binary variables of Dinkelbach's algorithm [4]. This algorithm begins with a tentative value for (7) then reduces the problem to unconstrained quadratic 0-1 optimization by multiplying both sizes by the denominator and regrouping terms. If a positive value is obtained for the optimal solution of this last problem its corresponding value in (7) is computed and the procedure iterated. Its most expensive step is the resolution of a sequence of unconstrained quadratic 0-1 programs, which are solved by a VNS heuristic as long as an improving column can be found. Then, optimality must be checked by a branch-and-bound algorithm.

3. Preliminary computational experiments

To the best of our knowledge, the first and up to now the unique global optimization algorithm for semi-supervised MSSC is due to Xia [10]. It consists of an adaptation of Tuy's [8] cutting plane method to solve the problem. Approximate results are reported for a version where this algorithm is halted before global convergence. From our experience with the unsupervised MSSC, this kind of adaptation of Tuy's method proposed by the author leads to exact solution of small instances with about only 25 entities. Moreover, the available code at http://www.uoguelph.ca/~yxia (accessed in April/2012) is limited in CPU memory to approximately 3000 concavity cuts, so that we cannot state the efficiency of the algorithm while searching for the global optimum.

We tested our algorithm in the same data provided by Xia with the available code. It consists of the Wine data set [2] with $n = 178$, $s = 13$, 6 cannot-link constraints and 13 must-link constraints. The tests were performed in a AMD Phenom II with a 800 Mhz clock and 8 Gb of RAM memory. The restricted master problem was iteratively solved by CPLEX 12 and unconstrained 0-1 quadratic programs were solved by a specialized branch-and-bound algorithm proposed in [6]. Remark that the problem of knowing if there exists a k-partition of n entities given a general set of must-link and cannot-link constraints is NP-hard using a reduction from the k-coloring problem [3]. The tested instances were all feasible. Table 1 presents the computational results obtained by Xia's algorithm (used in a heuristic way: the best of 5 runs is reported), denoted $concave$, and by our column generation algorithm, denoted cg, stabilized [3] with the solution obtained by $concave$. All instances were solved at the root node, without branching. We notice from the table that cg improves its perfomance as the number of clusters increases, making the number of entities per cluster small. In other direction, the per-

Table 1. Solution values and computing times obtained by algorithm *concave* and *cg* for instances of the Wine data with 6 cannot-link constraints and 13 must-link constraints

# clusters	*concave*		*cg*	
	cost	CPU time(s)	cost	CPU time(s)
2	4948494.5	0.03	4948494.5	295460.66
5	1380458.7	0.13	1380458.7	8080.99
10	688553.4	0.27	675021.9	1614.83
20	555786.0	0.50	517466.1	932.78

formance of algorithm *concave* deteriorates as the number of cluster increases both in terms of solution quality and computing times.

Acknowledgments

This work is supported by CNPq (Brazil) grant 305070/2011-8 and by HEC Montreal Data Mining Chair (Canada).

References

[1] Aloise, D., Hansen, P. and Liberti, L. (2012). "An improved column generation algorithm for minimum sum-of-squares clustering", Mathematical Programming, v. 131, 195–220.

[2] Asuncion, A and Newman, D.J. (2007), "UCI Machine Learning Repository", http://www.ics.uci.edu/∼mlearn/MLRepository.html

[3] Davidson, I. and Ravi, S. S. (2005). "Clustering with constraints: Feasibility issues and the k-means algorithm". Proceedings of 6th SIAM Intl. Conf. on Data Mining.

[4] Dinkelbach, W. (1967). "On nonlinear fractional programming". Management Science, v. 13, 492–498.

[5] du Merle O., Villeneuve D., Desrosiers J., Hansen P. (1999). "Stabilized column generation". Discrete Mathematics, v. 194, 229–237.

[6] Hansen, P., Jaumard, B. and Meyer, C. (2000). "A simple enumerative algorithm for unconstrained $0-1$ quadratic programming", Les Cahiers du GERAD, G-2000-59.

[7] du Merle, O., Hansen, P., Jaumard, B. and Mladenović, N. (2000). "An interior point algorithm for minimum sum-of-squares clustering". SIAM Journal Scientific Computing, v. 21, 1485–1505.

[8] Tuy, H. (1964). "Concave programming under linear constraints". Soviet Math, v. 5, 1437–1440

[9] Wagstaff, K., Cardie C., Rogers, S., and Schroedl, S. (2001). "Constrained k-means clustering with background knowledge". In Proc. of the 18th International Conference on Machine Learning, 577–584.

[10] Xia, Y. (2009). "A global optimization method for semi-supervised clustering". Data mining and Knowledge Discovery, v. 18, 214–256.

Proceedings of GOW 2012, pp. 23 – 24.

Preprocessing of Unconstrained Nonlinear Optimization Problems by Symbolic Computation Techniques

Elvira Antal,[1] Tibor Csendes[2]
University of Szeged, Institute of Informatics, H-6720 Szeged, Árpád tér 2, Hungary

[1] antale@inf.u-szeged.hu

[2] csendes@inf.u-szeged.hu

Formalization decisions in mathematical programming could significantly influence the complexity of the problem, and so the efficiency of the applied solver methods. This widely accepted statement induced investigations for the reformulation of optimization problems in the hope of getting easier to solve problem forms, e.g. in integer programming. These transformations usually go hand in hand with relaxation of some constraints and with the increase of the number of the variables. However, the quick evolution and the widespread use of computer algebra systems in the last few years motivated us to use symbolic computation techniques also in the field of global optimization.

We are interested in potential simplifications generated by symbolic transformations in global optimization, and especially in automatic mechanisms producing equivalent expressions while possibly decrease the dimension of the problem. As it was pointed out by Csendes and Rapcsák [1, 6], it is possible in some cases to simplify the unconstrained nonlinear objective function by nonlinear coordinate transformations. That means mostly symbolic replacement of redundant subexpressions hopefully resulting in less computation effort of the solver, while the simplified task remains equivalent to the original in the sense that a conversion between the solutions of the two forms is possible.

Consider the unconstrained nonlinear optimization problem

$$\min_{x \in \mathbb{R}^n} f(x), \tag{1}$$

where $f(x) : \mathbb{R}^n \to \mathbb{R}$ is a nonlinear, twice continuously differentiable function, given by symbolic expression, a formula. Our aim is to produce an equivalent problem form:

$$\min_{y \in \mathbb{R}^m} g(y), \tag{2}$$

where $g(y) : \mathbb{R}^m \to \mathbb{R}$ is simpler than $f(x)$, and a direct transformation between the optimal solutions y^* and x^* is possible. We mean a problem is simpler that an other one, if the earlier is easier to solve by optimization methods.

This kind of reformulation is usually done by hand in the stage of stating the problem, but that is not the only possibility. There are several techniques, that do automatic manipulations on optimization problems in order to increase the efficiency of the solver. However, such a usage of symbolic computation is applied better just for linear and integer programming. One can mention examples as the "presolving" mechanism of the AMPL processor [2], LP preprocessing[5], reformulation with relaxation for IP/MINLP solving [4], and also unusual problem solving approaches that use algebraic techniques as quantifier elimination (QE), Gröbner bases, etc. for symbolic optimization [3].

Our approach is different, because it is not a relaxation, and we do not want to solve the complete problem with algebraic techniques (since it can be quite slow and may be not even successful), just to make our problem easier for the solver. We are interested in mechanisms producing equivalent transformations and such that possibly decrease the dimension of the problem.

Our reformulation method for unconstrained nonlinear optimization problems based on nonlinear coordinate transformations described by Csendes and Rapcsák [1, 6]. The possible aims of the automatic simplifier method for unconstrained nonlinear optimization problems are as follows:

substitute some subexpressions in the objective function in order to

- eliminate parts of the computation tree,
- recognize unimodality (generalized unimodal property of n-dimensional functions),
- get an equivalent simpler form of the problem requiring less computation, and
- reduce (or at least not extend) the dimension of the problem.

We present a proper implementation of the referred theoretical algorithm in a popular symbolic programming environment, and testing on some examples both from the original publications and from the set of standard global optimization test problems to illustrate the capabilities of the method. Our results show, that this kind of preprocessing of an unconstrained optimization problem could be done usually relatively quickly, and the result is favorable in many cases.

Keywords: unconstrained nonlinear optimization, symbolic computation, reformulation

References

[1] Csendes, T. and T. Rapcsák. (1993). “Nonlinear Coordinate Transformations for Unconstrained Optimization. I. Basic Transformations,” J. of Global Optimization 3(2):213–221

[2] Gay, D.M. (2001). “Symbolic-Algebraic Computations in a Modeling Language for Mathematical Programming,” In Symbolic Algebraic Methods and Verification Methods, G. Alefeld, J. Rohn, and T. Yamamoto, eds, Springer-Verlag, pp. 99–106

[3] Kanno, M., K. Yokoyama, H. Anai, and S. Hara. (2008). “Symbolic Optimization of Algebraic Fuctions,” ISSAC’08, Hagenberg, Austria

[4] Liberti, L., S. Cafieri, and D. Savourey. (2010). “The Reformulation-Optimization Software Engine,” Mathematical Software - ICMS 2010, LNCS, 6327:303–314

[5] Mészáros, Cs. and U. H. Suhl. (2003). “Advanced preprocessing techniques for linear and quadratic programming,” OR Spectrum, 25(4):575–595

[6] Rapcsák, T. and T. Csendes. (1993). “Nonlinear Coordinate Transformations for Unconstrained Optimization. II. Theoretical Background,” J. of Global Optimization 3(3):359–375

Upper Bounding in Inner Regions for Global Optimization under Inequality Constraints

Ignacio Araya,[1] Gilles Trombettoni,[2] Bertrand Neveu,[3] and Gilles Chabert[4]

[1] *UTFSM, Universitad Federico Santa Maria, Valparaiso, Chile* iaraya@inf.utfsm.cl

[2] *IRIT, I3S, INRIA, Université Nice–Sophia, France* Gilles.Trombettoni@inria.fr

[3] *Imagine LIGM Université Paris–Est, France* Bertrand.Neveu@enpc.fr

[4] *LINA, Ecole de Mines de Nantes, France* Gilles.Chabert@emn.fr

Abstract

In deterministic constrained global optimization, upper bounding the objective function generally resorts to local minimization at the nodes of the branch and bound. The local minimization process is sometimes costly when constraints must be respected.

We propose in this paper an alternative approach when the constraints are inequalities or relaxed equalities so that the feasible space has a non-null volume. First, we extract an *inner region*, i.e., an (entirely feasible) convex polyhedron or box in which all points satisfy the constraints. Second, we select a point inside the extracted inner region and update the upper bound with its cost.

We use two inner region extraction algorithms implemented in our interval B&B called `IbexOpt` [7]. This upper bounding shows good performance in medium-sized systems proposed in the COCONUT suite.

Keywords: global optimization, upper bounding, intervals, branch and bound, inner regions

1. Upper bounding in inner regions

In deterministic constrained global optimization, upper bounding the objective function consists in finding a feasible point that improves the best cost already found in the branch and bound. Most global optimizers resort to local minimization[1] using a Lagrangian relaxation. The considered function is sometimes big, which may render the local minimization slow.

This paper describes an alternative approach for global optimization under *inequality* constraints defined by: $\min_{x\in[x]} f(x)$ *subject to* $g(x) \leq 0$, where $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is the real-valued objective function and $g : \mathbb{R}^n \rightarrow \mathbb{R}^m$ is a vector-valued function. $x = \{x_1, ..., x_i, ...x_n\}$ is a vector of variables varying in a box $[x]$.[2] x is said to be **feasible** if it satisfies the constraints.

The main idea is to exploit so called *inner regions*, i.e., subsets of the search space in which all points are feasible.

Definition 1. *Consider a system* $(f, g, x, [x])$ *comprising only inequality constraints. An* **inner region** r^{in} *is a feasible subset of* $[x]$*, i.e.,* $r^{in} \subset [x]$ *and all points* $x \in r^{in}$ *satisfy* $g(x) \leq 0$*.*

At every node (iteration) of our interval B&B named `IbexOpt` [7], the cost is bounded above by using two inner region extraction algorithms, called `InHC4` and `InnerPolytope`. `InHC4` is described in Section 2. It tries to extract an inner box from the current outer box. If it fails, one simply picks a point randomly inside the outer box and checks its feasibility. If it succeeds, a simple monotonicity analysis of f replaces the intervals of the monotonic variables by the

[1] We consider minimization in this paper without loss of generality.

[2] An interval $[x_i] = [\underline{x_i}, \overline{x_i}]$ defines the set of reals x_i s.t. $\underline{x_i} \leq x_i \leq \overline{x_i}$. A *box* $[x]$ is the Cartesian product of intervals $[x_1] \times ... \times [x_i] \times ... \times [x_n]$.

adequate bounds in the found inner box and the other values are randomly chosen. Then, InnerPolytope [7] builds a hyperplane for every inequality constraint. The hyperplane is produced by a special convex form of interval Taylor where the expansion point is chosen at a corner of the studied outer box. If it succeeds in building an inner polytope, the point minimizing the linearized form of the objective function is used to update the upper bound.

Contribution and limits

Contrarily to existing approaches, the proposed inner region extraction algorithms separate the feasibility part (handled first, by inner region extraction) and the computation of the cost (handled next, inside the found inner region).

It is important to highlight that, like the other inner region extraction algorithms, ours can fail in finding an inner region even if one such region exists. However, they are rather inexpensive. In particular, InHC4 is faster when it fails in finding an inner region for a given constraint because the loop on all the constraints can be prematurely stopped (see below).

This upper bounding based on inner region extraction could also apply to *"thick"* and relaxed equations that define a feasible space with a non-null volume. A thick equation has at least one coefficient that can be modeled by an interval constant. This parameter corresponds to a bounded uncertainty, e.g., an imprecision on a measurement, or an irrational constant, like π. A pure equality $f_k(x) = 0$ can also be handled with a relaxation as a thick equation $f_k(x) \in [-\epsilon_{eq}, +\epsilon_{eq}]$, i.e, two inequalities $-\epsilon_{eq} \leq f_k(x) \leq \epsilon_{eq}$. In this case of course, we can only guarantee the global optimum of the relaxed system, but ϵ_{eq} can often be chosen almost arbitrarily small.

2. The InHC4 inner box extraction algorithm

InHC4 follows the simple and general scheme proposed in [1, 4]. A main loop handles every constraint once in sequence and intersects incrementally the different boxes built.[3]

- The handling of the j^{th} constraint uses as input the inner box returned by the handling of constraint $g_{j-1}(x) \leq 0$. This box is inner w.r.t. the first $j-1$ contraints. Handling the first constraint $g_1(x) \leq 0$ is achieved with the outer box.
- Handling the constraint $g_j(x) \leq 0$ consists in finding, inside the input box under construction, a box which is inner w.r.t. this single constraint.

Thus, if a box is returned by the handling of the last constraint, this box is inner w.r.t. all the constraints.

The handling of an individual constraint in InHC4 is radically different from [4, 1]. Contrarily to their refutation process, our InHC4-Revise procedure tries to extract an inner region at each operator of the constraint. Like the main procedure of the state-of-the-art constraint propagation algorithm HC4 [2, 5], our InHC4-Revise procedure (InHC4R) works with a tree representation of the constraint, as illustrated in Fig. 1.

Let us denote by $[x]$ the input box and $g_j(x) \leq 0$ the constraint. Each node of the tree is associated to an interval, the intervals related to the leaves are initialized with the corresponding values in $[x]$. Then, the following two phases are performed:

- Bottom-up evaluation (see Fig. 1–left): The tree is traversed from the leaves to the root and intervals associated to an operator are computed with interval arithmetics. For example, the node pointed by the arrow is initialized with the interval $[0, 10] - [0, 15] = [-15, 10]$. Thus, every node contains an interval corresponding to the *natural* interval evaluation of the subexpression.
- Top-down inner projection (see Fig. 1–right): In each node related to a binary operator *op* and to an interval $[z]$, the 2-dimensional box corresponding to its children x_1 and x_2 is reduced to an inner box $[x_1]^{in} \times [x_2]^{in}$ such that:

[3] This scheme radically differs from constraint propagation achieved by HC4 that can handle a constraint several times.

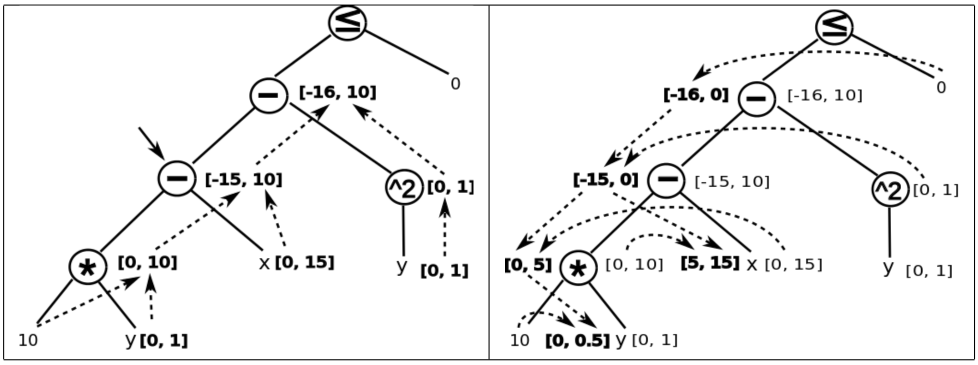


Figure 1. Binary tree representation of the constraint $10y - x - y^2 \leq 0$. **Left:** First bottom-up evaluation phase. **Right:** top-down inner projection phase.

$$\forall (x_1, x_2) \in [x_1]^{in} \times [x_2]^{in} : \ \ x_1 \; op \; x_2 \in [z] \tag{1}$$

If op corresponds to a unary operator, its unique child is reduced to $[x]^{in}$, such that:

$$\forall x \in [x]^{in} : \ \ op(x) \in [z] \tag{2}$$

If the inner projection returns an empty box (i.e., no box satisfies relation (1) or (2)), then the top-down process is interrupted. It means that InHC4R has failed in finding a box that is inner w.r.t. $g_j(x) \leq 0$. Since the approach is not complete because not all the feasible space is extracted during the top-down traversal, an inner box could be indeed missed by the process.

Consider the product operator of Fig. 1–right and its two children. The reduced intervals appear in bold in the left side of each node. After the reduction of the product operator, its interval becomes $[0, 5]$. Before reduction, its children are associated to the intervals $[10, 10]$ and $[0, 1]$. They are then reduced to $[10, 10]$ and $[0, 0.5]$ respectively. The reduction agrees with relation (1), i.e, $\forall y \in [0, 0.5] : \ 10 * y \in [0, 5]$.

2.1 Inner projection for unary and binary basic operators

For unary, monotonic and continous operators, like log and exp, the inner projection is trivial and computes the (maximum) inner interval (i.e, no feasible point is lost, modulo roundoffs). It is very close to a standard projection in HC4R. However, for managing floating-point roundoff errors, the outward rounding of HC4R is replaced by inward rounding.

For non monotonic unary operator like x^2 or $sinus$, a union of intervals is computed by HC4R, before returning the hull of these intervals. For an inner projection in InHC4R instead, only a single interval is kept since holes between these intervals contain inconsistent points.

For binary operators, Chabert and Beldiceanu in [3] proposed inner projections, but with a case-by-case approach. We have extended their approach and built a more generic projection based on monotonicity properties. There usually exists an infinite number of maximal boxes (as depicted in Fig. 2), and we have succeeded in designing inner projection operators that select randomly one *maximal* inner box. Note that these inner projections lead to heuristic choices since a single box cannot include the whole inner/feasible space. Also note that the two inequalities $\underline{z} \leq (x_1 \; op \; x_2) \leq \overline{z}$ are handled in sequence, the inner box computed for one inequality being used as input of the second one.

For binary (or n-ary) operators that are monotonic w.r.t. each of their variables, a generic procedure, called MonoMaxInnerBox, can compute randomly one maximal inner box, if one such box exists, as shown in Fig. 2. This procedure is of course used for the addition and subtraction operators. It is also used for handling several (monotonic) subcases of the non monotonic binary operators: the multiplication and the division. Fig. 3 illustrates the two main cases for the multiplication $x_1 * x_2 \in [z]$, depending whether 0 belongs or not to $[z]$.

Handling the division operator amounts in rewriting $x_1/x_2 = x_1 * \frac{1}{x_2} \in [z]$, although a direct implementation would also be possible.

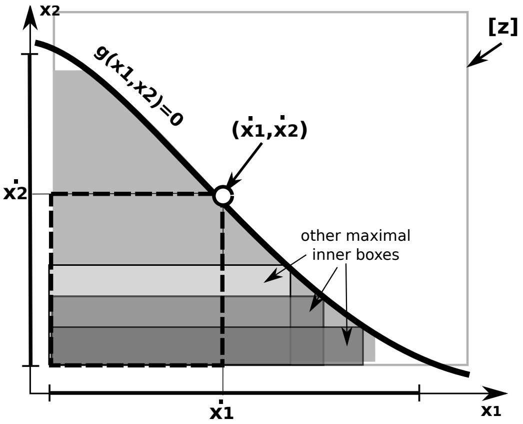


Figure 2. The dotted box corresponds to a maximal inner box of $[z]$ w.r.t. the constraint $g(x_1, x_2) \leq 0$. A point $\dot{x_1}$ is randomly picked inside the range of allowed values illustrated by the horizontal segment. Only one remaining value $\dot{x_2}$ can then make the computed inner box maximal.

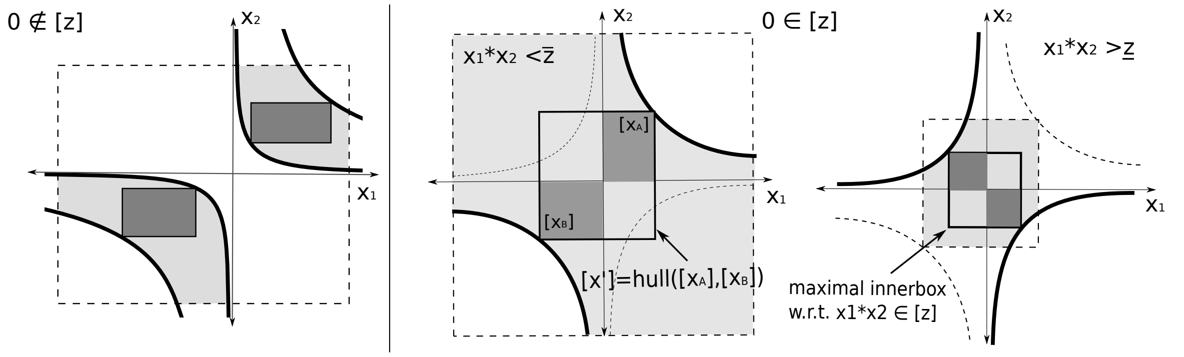


Figure 3. Inner projection for the binary multiplication. **Left:** Two maximal boxes that can indifferently be computed by MonoMaxInnerBox in the two disjoint inner regions (quadrants) defined by the operator $x_1 * x_2 \in [z] \geq 0$. **Middle and right:** Maximal box computed for $x_1 * x_2 \in [z] \ni 0$ ($\overline{z} \geq -\underline{z}$) with four calls to MonoMaxInnerBox (boxes in grey).

2.2 Properties

We have proven that every implemented unary and binary operator computes a maximal inner box, modulo the loss involved by inward roundoffs. In case a constraint contains only a single occurrence of each variable, InHC4R thus computes a maximal inner box, when one such box is found. The result finally holds for a system of inequality constraints handled by InHC4.

3. Experiments

We have tested our original upper bounding procedure on a sample of about thirty constrained global optimization found in the COCONUT benchmark suite. 24 of them correspond to the most difficult systems selected by Ninin et al. [6]. Equations $f_k(x) = 0$ are relaxed by inequalities $-\epsilon_{eq} \leq f_k(x) \leq \epsilon_{eq}$, with $\epsilon_{eq} =$ 1.e-8. The main results are the following. A first experiment highlights the benefits of this upper bounding, compared to a simple probing in every explored outer box. A second experiment underlines that, in a large majority of the tested systems, the upper bounding is satisfactory since the upper bound converges faster than the lower bound towards the final value. Third, a qualitative study determines which of the two inner region extraction heuristics is the most useful in every system. A last study analyzes the size of the outer boxes in which the algorithms succeed in extracting an inner region.

References

[1] F. Benhamou and F. Goualard. Universally Quantified Interval Constraints. In *Proc. CP, Constraint Programming, LNCS 1894*, pages 67–82, 2004.

[2] F. Benhamou, F. Goualard, L. Granvilliers, and J.-F. Puget. Revising Hull and Box Consistency. In *Proc. ICLP*, pages 230–244, 1999.

[3] G. Chabert and N. Beldiceanu. Sweeping with Continuous Domains. In *Proc. CP, LNCS 6308*, pages 137–151, 2010.

[4] H. Collavizza, F. Delobel, and M. Rueher. Comparing Partial Consistencies. *Reliable Comp.*, 5(3):213–228, 1999.

[5] F. Messine. *Méthodes d'Optimisation Globale basées sur l'Analyse d'Intervalle pour la Résolution des Problèmes avec Contraintes*. PhD thesis, LIMA-IRIT-ENSEEIHT-INPT, Toulouse, 1997.

[6] J. Ninin, F. Messine, and P. Hansen. A Reliable Affine Relaxation Method for Global Optimization. Technical Report RT-APO-10-05, IRIT, 2010.

[7] G. Trombettoni, I. Araya, B. Neveu, and G. Chabert. Inner Regions and Interval Linearizations for Global Optimization. In *AAAI*, pages 99–104, 2011.

Solving the planar leader-follower problem with variable demand via hybrid parallel algorithms*

Aránzazu G. Arrondo[1], Juana L. Redondo[2], José Fernández[1], and Pilar M. Ortigosa[3]

[1] *Dpt. of Statistics and Operations Research, University of Murcia, Murcia, Spain*
agarrondo@um.es, josefdez@um.es

[2] *Dpt. of Computer Architecture and Technology, University of Granada, Granada, Spain*
jlredondo@atc.ugr.es

[3] *Dpt. of Computer Architecture and Electronics, University of Almería, ceiA3, Almería, Spain*
ortigosa@ual.es

Abstract

A parallel strategy for solving a centroid problem with variable demand is proposed. In the competitive location problem considered in this paper, the aim is to maximize the profit obtained by a chain (the leader) knowing that a competitor (the follower) will react by locating another single facility after the leader locates its own facility. The demand is supposed to be concentrated at n demand points, which split their buying power among the facilities proportionally to the attraction they feel for them. The attraction for a facility depends on both the location and the quality of the facility. In most competitive location literature it is assumed that the demand is deterministic, i.e., fixed regardless the conditions of the market. However, demand can vary depending on prices, distances to the facilities, etc. Taking variable demand into consideration, as we do here, increases the complexity of the problem and, therefore, the computational effort needed to solve it, but it makes some models more realistic. Several heuristic methods were proposed to cope with this hard-to-solve global optimization problem. Through a comprehensive computational study, it was shown that the evolutionary algorithm TLUEGO was the heuristic which provides the best solutions. Nevertheless, TLUEGO requires high computational effort, even to manage problems with small sizes. This is mainly due to the high cost at evaluating the leader's objective function, which requires the resolution of another hard-to-solve optimization problem, namely, the follower's problem. In this work, we propose the development of a hybrid parallel strategy, where two programming paradigms are implemented simultaneously, i.e. both shared memory and distributed programming are considered at different levels of the algorithm.

Keywords: Nonlinear bi-level programming problem, centroid (or Stackelberg) problem, evolutionary algorithm, high performance computing approaches, parallelism, distributed memory, shared memory

1. Introduction

Competitive location deals with the problem of locating facilities to provide a service (or goods) to the customers (or consumers) of a given geographical area where other competing facilities offering the same service are already present (or will enter the market in the near future).

The scenario considered in this paper is that of a *duopoly*. A chain, the *leader*, wants to locate a new single facility in a given area of the plane, where there already exist m facilities offering

*This work has been funded by grants from the Spanish Ministry of Science and Innovation (TIN2008-01117, ECO2011-24927), Junta de Andalucía (P08-TIC-3518 and P10-TIC-6002) and Fundación Séneca (The Agency of Science and Technology of the Region of Murcia, 00003/CS/10 and 15254/PI/10), in part financed by the European Regional Development Fund (ERDF). Juana López Redondo is a fellow of the Spanish 'Juan de la Cierva' contract program.
Corresponding author: P.M. Ortigosa.

the same goods or product. The first k of those m facilities belong to the chain ($0 \leq k < m$) and the other $m - k$ to a competitor chain, the *follower*. The leader knows that the follower, as a reaction, will subsequently position a new facility too. The demand is elastic (it varies depending on the location of the facilities) and is supposed to be concentrated at n demand points, whose locations p_i are known. The location f_j and quality of the existing facilities is also known. The aim is to maximize the profit obtained by the leader following the follower's entry. These types of bilevel programming problems were introduced by Hakimi, who introduced the terms *medianoid* for the follower problem, and *centroid* for the leader problem [2].

For handling the centroid problem, several heuristics were proposed in [5]. The computational studies showed that the evolutionary algorithm TLUEGO (*Two-Level Universal Evolutionary Global Optimizer*) was more robust than the other strategies. However, the computational time employed by TLUEGO for solving small size problems was very high. This clearly suggests that a parallelization of the algorithm is needed, especially if real problems, with more demand points, are to be solved.

It is important to mention that to solve a single centroid problem, many medianoid problems have to be solved, since the evaluation of the leader's objective function at a given point requires the resolution of the corresponding medianoid problem. Recently, in [4], the medianoid problem considered in this work, has been studied and solved using the evolutionary algorithm UEGO (*Universal Evolutionary Global Optimizer*), initially described in [3]. The computational studies showed that the heuristic algorithm UEGO was a good alternative to deal with the medianoid problem, and hence it will be considered in this work. Nevertheless, solving a medianoid problem is not a negligible task; on the contrary, the medianoid problem is a hard-to-solve global optimization problem (as most competitive location problems are).

In this work, we propose a parallel algorithm which, simultaneously, make use of two parallel programming paradigms. The aim is to increase the efficiency and scalability of the parallel version. On the one hand, we propose a shared memory programming model to achieve a substantial reduction in the computing effort of the sequential optimization algorithm UEGO, which is used to solve the *medianoid* problems. This means, that the evaluation of the the leader's objective function will be carried out concurrently. On the other hand, we propose a parallel model based on message-passing protocols, whose objective is to reduce the computational time of TLUEGO when coping with the centroid problem.

Currently, a comprehensive computational study is being carried out in the supercomputer Ben Arabi of the Supercomputing Center of Murcia, Spain. In particular, the executions are running in Arabi, which is a Blade Cluster with 816 cores, organized in 32 nodes with 16GB of memory each, and 70 nodes with 8GB each (102 nodes altogether). Each node has 8 cores, divided into 2 Intel Xeon Quad Core (E5450) to 3.0 GHz. At this moment, only a small portion of the studies has been executed, which is not enough to be able to infer conclusions. In future versions of this work, sound computational results will be shown. In the following, for the sake of completeness, some details about the parallel algorithm are shown.

2. The shared memory programming approach to solve the follower problem

The medianoid (follower) problem with variable demand (see [4] for an analytical description of the model) has been solved in [4] using the evolutionary algorithm UEGO. In this work, such an algorithm has been paralellized through a multithreaded approach which takes advantage of the computer power in multicore architectures.

In modern multicore systems, all the processing units have direct access to the whole memory. Processing units are connected to some interconnection network, through which they can access the common memory banks. There exist several ways to deal with parallelism in a shared memory model. For the problem at hand, OpenMP (Open Multi-Processing) (`www.openmp.org`) is recommended, since it is a portable and scalable model, and gives programmers a simple and flexible interface for developing parallel applications. Programmers use the OpenMP directives to tell the compiler which parts of the program must be executed concurrently and to specify synchronization points [1].

In SharedMemUEGO, as we will call the parallel implementation of UEGO, the parallelism comes from the concurrent execution of the *creation* and *optimization* procedures. Those procedures are in charge of generating a new offspring using recombination operators, and of improving the current population through local optimization mechanisms, respectively. Furthermore, it is worth to mention that there exist a "synchronization point" imposed by a *selection* procedure, since to be able to decide which portion of the population is going to continue as the next generation of the algorithm, it is necessary to have the whole population. Partial selections are also carried out concurrently, although finally a global one is required for a correct performance of the algorithm in terms of quality in the solutions.

In the parallel model, $MaxTh$ threads are created. This value refers to the maximum number of available process units to solve the problem. Basically, the algorithm distributes the individuals in the population among the $MaxTh$ threads. They apply either the *creation* or *optimization* procedures, and write back the evaluation results. Threads only receive the address memory of the corresponding individuals and they are in charge of either reading or updating through this value. Notice that the distribution is carried out so that a single individual is assigned to each thread each time. When a particular thread finishes its task, another single individual will be picked up for working on it.

In our implementation, this process will be executed at each of the nodes of Arabi.

3. The message-passing parallel algorithm to solve the leader problem

The centroid (leader) problem with variable demand was solved by means of TLUEGO algorithm in [5]. The interested reader is referred to that paper for an in-detailed description of both the model and the algorithm.

TLUEGO is parallelized through a master-slave technique. Master-slave is a communication model where one processing element (the master) has unidirectional control over one or more processing elements (the slaves). This technique is called "global parallel model" too, since the management of the population, is global (i.e. all the individuals in the population are considered when the selection procedure is carried out). For the problem at hand, the master takes charge of performing such a selection procedure. In our model, the parallelism comes from the evaluation of the individuals in the population. This is because the fitness of an individual is independent from the rest of the population, and there is no need to communicate during this phase. The evaluation of individuals is parallelized by assigning a fraction of the population to each 'node' of Arabi. Notice, that a single evaluation involve the resolution of a medianoid problem, which will be executed concurrently, as mentioned in Section 2.

Communications occur only as each node receives its subset of individuals for working on it and when the nodes return the result values. Two versions of this parallel algorithm will be studied. In the first one, the algorithm stops and waits to receive the fitness values for all the population before proceeding to the next generation; in this case, the global parallel algorithm is considered *synchronous* and it has exactly the same properties as the sequential one. For the second case, the algorithm does not stop to wait for all processing elements, it does not work exactly like the sequential algorithm, i.e. it is *asynchronous*, although it may increase the efficiency of the parallel algorithm.

MPI has been used to implement the message-passing procedures. It is a language independent communications protocol used to program parallel computers [6]. Processes are written in a sequential language (C,C++, FORTRAN), and communications and synchronizations are made by calling functions from the MPI library.

References

[1] B. Chapman, G. Jost and R. van der Pas. Using OpenMP. Portable shared memory parallel programming. The MIT Press, Cambridge, Massachusetts, 2008.

[2] S.L. Hakimi. On locating new facilities in a competitive environment. *European Journal of Operational Research*, 12(1):29–35, 1983.

[3] M. Jelásity, P.M. Ortigosa, and I. García. UEGO, an abstract clustering technique for multimodal global optimization. *Journal of Heuristics*, 7(3),215–233, 2001.

[4] J.L. Redondo, J. Fernández, A.G. Arrondo, I. García, and P.M. Ortigosa. Deterministic or variable demand? Does it matter when locating a facility? Omega, 40(1): 9–20, 2012.

[5] J.L. Redondo, A.G. Arrondo, J. Fernández, I. García, and P.M. Ortigosa. A Two-Level Evolutionary Algorithm for Solving the Facility Location and Design $(\mathbf{1}|\mathbf{1})$-centroid problem on the Plane with Variable Demand. Submitted

[6] M. Snir, S. Otto, S. Huss-Lederman, D. Walker and J. Dongarra. MPI: The Complete Reference. The MIT Press, Cambridge, Massachusetts. London, England, 1996.

Convex underestimation of edge-concave functions by a simultaneous convexification with multilinear monomials

Martin Ballerstein, and Dennis Michaels

Eidgenössische Technische Hochschule Zürich, Institut für Operations Research, Rämistrasse 101, 8092 Zürich, Switzerland,
{martin.ballerstein, dennis.michaels}@ifor.math.ethz.ch

Abstract We address the construction of tight convex underestimators for an edge-concave function over a box. In contrast to common relaxation methods we derive the underestimators by a simultaneous convexification of one edge-concave function together with the set of all multilinear monomials. We show that the tightest underestimators are polyhedral and give a complete facet-description. A computational case study demonstrates the usefulness of the proposed relaxation technique in global optimization.

Keywords: global optimization, convex relaxations, edge-concave functions, simultaneous convexification

1. Introduction

The efficiency of global optimization solvers heavily depends on the ability of constructing strong convex relaxations. Common methods typically derive convex relaxations for a general mixed-integer nonlinear optimization problem by replacing every nonlinear function $f : [l, u] \subseteq \mathbb{R}^n \to \mathbb{R}$ by convex under- and concave overestimating functions (e.g., see [13, 20, 4]). The underlying mathematical object is the convex set $\operatorname{conv}(\{(x, f(x)) \in \mathbb{R}^{n+1} \mid x \in [l, u]\})$. However, such a separate convexification strategy ignores the interactions between different nonlinearities and leaves potential to improve the quality of the resulting relaxations by a simultaneous consideration of several nonlinearities.

One way to take the interactions between the nonlinearities into account is to study the *simultaneous convex hull* $\operatorname{conv}(\{(x, f_1(x), \ldots, f_m(x)) \in \mathbb{R}^{n+m} \mid x \in [l, u]\})$ for a finite set of functions $f_i : [l, u] \subseteq \mathbb{R}^n \to \mathbb{R}$, $i = 1, \ldots, m$. A famous example, that has been intensively studied in the literature, is given when the set of functions consists of all quadratic monomials (e.g., see [15, 6, 3]). In this case, a tight relaxation is obtained by the Boolean quadric polytope and semidefinite constraints. The strength of this relaxation was shown both theoretically and computationally (cf. [3]).

Recently, Tawarmalani [19] has analyzed the simultaneous convex hull of finitely many functions more generally. He has derived necessary conditions on the extreme points of such sets. For a set of submodular functions, it has been, in particular, shown that parts of the simultaneous convex hull can be already described by all convex hulls of the corresponding single functions.

In this work, we focus on edge-concave functions. A function $f : \mathbb{R}^n \to \mathbb{R}$ is called *edge-concave* on a box $[l, u]$ if it is componentwise concave (cf. [17]). Thus, multilinear functions also belong to the class of edge-concave functions. It was shown in [17] that the tightest convex underestimator of a single edge-concave function is polyhedral. Explicit formulas are known up to dimension three [13, 14]. Formulas are not known for higher dimensions as this would require a complete enumeration of all possible (non-isomorphic) triangulations of the underlying box $[l, u]$. Such an approach is impracticable as already in dimension four, a full-dimensional box $[l, u]$ has 87,959,448 regular triangulations which can be partitioned into 235,277 symmetry classes (cf. [8, Thm. 2]).

In the following, we will derive an explicit formula for a convex underestimator of an edge-concave function on a box by a simultaneous convexification with all multilinear monomials. Our point of departure is the polyhedral description of the following set

$$\mathcal{Q}^{(n)}_{[l,u]} \quad := \quad \operatorname{conv}\left(\{F^{\,(n)}(x) \in \mathbb{R}^{2^n-1} \mid x \in [l,u]\}\right), \tag{1}$$

where $F^{\,(n)} := (x_1, \ldots, x_n, x_1x_2, \ldots, x_{n-1}x_n, x_1x_2x_3, \ldots, \prod_{i=1}^n x_i)$ denotes the vector consisting of all multilinear monomials in n variables. It has been shown in [2, 5] that $\mathcal{Q}^{(n)}_{[l,u]}$ is a simplex.

We extend the investigations by adding a further continuous function $f : [l,u] \subseteq \mathbb{R}^n \to \mathbb{R}$ to the set of all multilinear monomials. The corresponding simultaneous convex hull is then not necessarily polyhedral. In Section 2 we will present that the following part of the simultaneous convex hull is polyhedral

$$\mathcal{U}^{(n)}_{[l,u]}\,[f] \quad := \quad \operatorname{conv}(\{(F^{\,(n)}(x), \mu) \in \mathbb{R}^{2^n-1} \times \mathbb{R} \mid \mu \geq f(x),\ x \in [l,u]\}),$$

if f is edge-concave on $[l,u]$. We also discuss sufficient and necessary conditions for f such that $\mathcal{U}^{(n)}_{[l,u]}\,[f]$ is polyhedral and provide a complete facet-description. Finally, we give a computational case study in Section 3, where we apply the proposed polyhedral relaxation to test instances of the Molecular Distance Geometry Problem [12]. We show the practical efficiency of our approach in comparison to standard relaxation methods.

2. A tight polyhedral description underestimating edge-concave functions over the set $\mathcal{Q}^{(n)}_{[l,u]}$

Let $f : [l,u] \to \mathbb{R}$ be a real-valued, continuous function on $[l,u] \subseteq \mathbb{R}^n$. We first observe that the description of $\mathcal{Q}^{(n)}_{[l,u]}$ is essential to describe $\mathcal{U}^{(n)}_{[l,u]}\,[f]$. It is straightforward to check

Lemma 1. *Each facet-defining inequality of $\mathcal{Q}^{(n)}_{[l,u]}$ also induces a facet for $\mathcal{U}^{(n)}_{[l,u]}\,[f]$.*

Next, we consider *edge-concave* functions f. The following Theorem shows that a complete description for $\mathcal{U}^{(n)}_{[l,u]}\,[f]$ requires only one further inequality, in addition to $\mathcal{Q}^{(n)}_{[l,u]}$.

Theorem 2. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a continuous function that is edge-concave on $[l,u]$. Then,*

$$\mathcal{U}^{(n)}_{[l,u]}\,[f] \quad = \quad \{(z,\mu) \in \mathbb{R}^{2^n-1} \times \mathbb{R} \mid z \in \mathcal{Q}^{(n)}_{[l,u]},\ \mu \geq a_{\emptyset} + \sum_{\emptyset \neq J \subseteq I} a_J z_J\},$$

where $I := \{1, \ldots, n\}$ and for all $J \subseteq I$

$$a_J = \Big(\sum_{v \in \operatorname{vert}([l,u])} (-1)^{\alpha(v)+|J|}\ f(\bar{v}) \prod_{i \in I \setminus J} v_i \quad \Big) \quad / \quad \prod_{i \in I} (u_i - l_i) \tag{2}$$

where $\alpha(v)$ denotes the number of components of v which are at their lower bounds and $\bar{v}$ is the opposite vertex of v, i.e. defined by $\bar{v}_j = l_j$, if $v_j = u_j$ and $\bar{v}_j = u_j$, if $v_j = l_j$.

A sufficient and necessary condition for Theorem 2 on f is that f is underestimated over the box $[l,u]$ by the following multilinear function $m_f(x) := \sum_{J \subseteq I} a_J \prod_{j \in J} x_j$, where the coefficients are given in Equation (2). In fact the values of the multilinear function m_f and f coincide at the vertices of $[l,u]$. Therefore, the additional cut is induced by the linearized version of the function m_f. Thereby the description can also be deduced from the RLT-approach introduced by Sherali and Adams (cf. [16, 2, 10] and references therein). This is also remarked in [19].

Note that a necessary condition on functions f for Theorem 2 is that f must be a function whose best possible convex underestimator is *vertex-polyhedral* as defined by Tardella in [18]. Edge-concave functions also belong to this class of functions. However, the property of vertex-polyhedrality is not a sufficient condition for Theorem 2.

3. Computations

In this section we present a computational case study which compares the relaxation technique based on sets $\mathcal{U}^{(n)}_{[l,u]}[f]$ with standard relaxation methods. We consider a series of instances from the Molecular Distance Geometry Problem (MDGP) (e.g., see [12]). The MDGP is to determine the three-dimensional structure of a molecule consisting of a finite set $A = \{1, \ldots, s\}$ of atoms and given distances $d_{i,j} \geq 0$ between every two atoms $\{i, j\} \in E \subseteq A \times A$. The MDGP is solved if the optimal objective function value of the following unconstrained nonconvex optimization problem equals zero:

$$\min \sum_{\{i,j\}\in E} \left(||\xi^i - \xi^j||^2 - d_{ij}^2\right)^2 \quad \text{s. t.} \quad \xi := (\xi^1, \ldots, \xi^s) \in \mathbb{R}^{3s}, \tag{3}$$

where $\xi^i := (\xi^i_1, \xi^i_2, \xi^i_3) \in \mathbb{R}^3$ represents the position of atom i in the three-dimensional space. In order to illustrate the impact of the proposed relaxation method we follow the work [7] and expand each summand in Equation (3). The expanded model formulation involves the following types of expressions: $x_1x_2,\ x_1x_2x_3,\ x_1x_2x_3x_4,\ x_1^2,\ x_1^4,\ -x_1^2x_2x_3,\ -x_1^3x_2$.

We implemented three different relaxation strategies. For **StandRelax**, each term is reformulated in terms of products of univariate or bilinear/trilinear terms and the formulas for their envelopes are applied. This approach follows the relaxation strategies given by Cafieri et al. [7]. For **Q4Relax**, we relax each quadrilinear term $x_1x_2x_3x_4$ and all involved bi- and trilinear monomials simultaneously by the facet-description of $\mathcal{Q}^{(4)}_{[l,u]}$. The third strategy **Q4Relax+** applies the relaxations $\mathcal{U}^{(2)}_{[l,u]}[-x_1^3x_2]$ and $\mathcal{U}^{(3)}_{[l,u]}[-x_1^2x_2x_3]$, for each term of the form $-x_1^3x_2$ or $-x_1^2x_2x_3$, respectively. For this, we add the corresponding inequalities $\mu \geq a_\emptyset + \sum_{\emptyset\neq J\subseteq I} a_J z_J$ of Theorem 2 to Q4Relax.

All instances were randomly generated as described in [11] and were given to us by Jon Lee. We implemented a branch-and-bound framework in the programming language C and solved all linear subproblems with SCIP 2.1.1 [1] together with CPLEX 12.3 [9]. The computations were accomplished on a 2.67 GHz INTEL X5650. The time limit for all computations was one hour. Table 1 summarizes our computational results. It is evident that the computations benefit from the derived simultaneous relaxations.

	Bound at root node			Bound after one hour		
	StandRelax	Q4Relax	Q4Relax+	StandRelax	Q4Relax	Q4Relax+
lav6	-36,871.1	-14,770.3	-14,770.3	-15,554.0	-7,212.0	-6,333.0
lav7	-141,278.7	-56,698.8	-56,698.8	-69,754.6	-32,564.3	-30,002.2
lav8	-176,869.1	-70,946.4	-70,946.4	-100,891.1	-46,212.7	-43,218.0
lav10	-602,754.7	-241,694.4	-241,694.4	-423,748.9	-184,078.4	-176,735.0
lav20	-15,840,033.3	-6,367,589.5	-6,367,589.5	-13,291,564.3	-5,618,446.6	-5,529,058.1

Table 1. The table reports on the lower bounds obtained at the root node and after one hour of computation time.

We also present a comparison of our relaxation techniques with state-of-the-art optimization software. We remark that the software package BARON [20] can solve the MDGP efficiently in the factorized model formulation of Equation (3), but it is outperformed by our ad-hoc implementation when the expanded model formulation is used.

Acknowledgments

This work is part of the Collaborative Research Centre "Integrated Chemical Processes in Liquid Multiphase Systems" (CRC/Transregio 63 "InPROMPT") funded by the German Research Foundation (DFG). Especially, the second author thanks the DFG for its financial support.

We would like to thank Jon Lee for providing the Lavor test instances used in this paper.

References

[1] Tobias Achterberg, *SCIP: Solving Constraint Integer Programs*, Mathematical Programming Computation **1** (2009), no. 1, 1–41.

[2] Warren P. Adams and Hanif D. Sherali, *A hierarchy of relaxations leading to the convex hull representation for general discrete optimization problems*, Annals of Operations Reserach **140** (2005), 21–47.

[3] Kurt M. Anstreicher and Samuel Burer, *Computable representations for convex hulls of low-dimensional quadratic forms*, Mathematical Programming **Ser. B (124)** (2010), 33–43.

[4] P. Belotti, *COUENNE: a user's manual*, Tech. report, Lehigh University, 2009.

[5] Christoph Buchheim and Giovanni Rinaldi, *Efficient reduction of polynomial zero-one optimization to the quadratic case*, SIAM Journal on Optimization **18** (2007), no. 4, 1398–1413.

[6] Samuel Burer and Adam Letchford, *On non-convex quadratic programming with box constraints*, SIAM Journal on Optimization **20** (2009), 1073–1089.

[7] Sonia Cafieri, Jon Lee, and Leo Liberti, *On convex relaxations of quadrilinear terms*, Journal on Global Optimization **47** (2010), no. 4, 661–685.

[8] Peter Huggins, Bernd Sturmfels, Josephine Yu, and Debbie Yuster, *The hyperdeterminant and triangulations of the 4-cube*, Mathematics of Computation **77** (2008), 1653–1679.

[9] IBM, *CPLEX*, `http://www.ibm.com/software/integration/optimization/cplex`.

[10] Monique Laurent, *A comparision of the Sherali-Adams, Lovász-Schrijver and Lassere relaxations for 0-1 programming*, Mathematics of Operations Reserach **28** (2003), no. 3, 470–496.

[11] Carlile Lavor, *On generating instances for the molecular distance geometry problem*, Global Optimization. From Theory to Implementation (Leo Liberti and Nelson Maculan, eds.), Springer, 2006, pp. 405–414.

[12] Carlile Lavor, Leo Liberti, and Nelson Maculan, *Molecular distance geometry problem*, Encyclopedia of Optimization (Christodoulos A. Floudas and Panos M. Pardalos, eds.), Springer, 2nd ed., 2009, pp. 2304–2311.

[13] Garth P. McCormick, *Computability of global solutions to factorable nonconvex programs. I: Convex underestimating problems*, Mathematical Programing **10** (1976), 147–175.

[14] Clifford A. Meyer and Christodoulos A. Floudas, *Convex envelopes for edge-concave functions*, Mathematical Programming **Ser. B, 103** (2005), 207–224.

[15] Manfred Padberg, *The boolean quadric polytope: Some characteristics, facets and relatives*, Mathematical Programming **45** (1989), 139–172.

[16] Hanif D. Sherali, *Convex envelopes of multilinear functions over a unit hypercube and over special sets*, Acta Mathematica Vietnamica **22** (1997), 245–270.

[17] Fabio Tardella, *On the existence of polyedral convex envelopes*, Frontiers in global optimization, Kluwer Academic Publisher, 2003, pp. 563–573.

[18] ———, *Existence and sum decomposition of vertex polyhedral convex envelopes*, Optimization Letters (2008), 2:363–375.

[19] Mohit Tawarmalani, *Inclusion certificates and simultaneous convexification of functions*, Optimization Online (2010), available at `http://www.optimization-online.org/DB_FILE/2010/09/2722.pdf`.

[20] Mohit Tawarmalani and Nikolaos V. Sahinidis, *A polyhedral branch-and-cut approach to global optimization*, Mathematical Programming **103** (2004), no. 2, Ser. B, 225–249.

On lower bounds using separable terms in interval B&B for one-dimensional problems *

José L. Berenguel,[1] L.G. Casado, [2] I. García, [3] E.M.T. Hendrix, [3] and F. Messine[1,3]

[1] *TIC 146: Supercomputing-Algorithms Research group,*
University of Almería, Agrifood Campus of International Excellence (ceiA3), 04120, Spain. jlberenguel@gmail.com

[2] *Department of Computer Architecture and Electronics,*
University of Almería, Agrifood Campus of International Excellence (ceiA3), 04120, Spain. leo@ual.es

[3] *Department of Computer Architecture,*
University of Málaga, Campus de Teatinos, 29017, Spain. igarcia@ual.es, Eligius@uma.es

[4] *University of Toulouse,*
ENSEEIHT-IRIT UMR-CNRS-5505, 2 rue Camichel, 31000 Toulouse, France. Frederic.Messine@n7.fr

Abstract Interval Branch-and-Bound (B&B) algorithms are powerful methods which aim for guaranteed solutions of Global Optimization problems. Lower bounds for a function in a given interval can be obtained directly with Interval Arithmetic. The use of lower bounds based on Taylor forms show a faster convergence to the minimum with decreasing size of the search interval. Our research focuses on one dimensional functions that can be decomposed into several terms (sub-functions). The question is whether using this characteristic leads to sharper bounds when based on bounds of the sub-functions. This paper deals with separable functions in two sub-functions.

The use of the separability is investigated for the so-called Baumann form and Lower Bound Value Form (LBVF). It is proven that using the additively separability in the LBVF form may lead to a combination of linear minorants that are sharper than the original one. Numerical experiments confirm this improving behaviour and also show that not all separable methods do always provide sharper additively lower bounds. Additional research is needed to obtain better lower bounds for multiplicatively separable functions and to address higher dimensional problems.

Keywords: separable functions, Interval Arithmetic, Taylor forms, branch-and-bound lower bound

1. Introduction

Interval Branch-and-Bound methods aim for guaranteed solutions of Global Optimization problems. Consider the one dimensional generic interval constrained global optimization problem, which is to find

$$f^* = \min_{x \in S} f(x) \tag{1}$$

where $S \in \mathbb{I}$ is the search region and $\mathbb{I}$ stands for the set of all one-dimensional closed real intervals.

Definition 1. *Function $f : S \subset \mathbb{R} \to \mathbb{R}$ is additively separable, if it can be written as*

$$f(x) = \sum_{j=1}^{p} f_j(x), \; x \in S. \tag{2}$$

*This work has been funded by grants from the Spanish Ministry of Science and Innovation (TIN2008-01117), and Junta de Andalucía (P11-TIC-7176), in part financed by the European Regional Development Fund (ERDF). Eligius M.T. Hendrix is a fellow of the Spanish "Ramón y Cajal" contract program, co-financed by the European Social Fund.

We have

$$\min_{x\in S} f(x) \geq \sum_{j=1}^{p} \min_{S} f_j(x). \tag{3}$$

Let $\underline{F}_j$ be a lower bound of f_j over S. Then we have

$$\min_{x\in S} f(x) \geq \sum_{j=1}^{p} \underline{F}_j. \tag{4}$$

To create a lower bound $\underline{F}$ of f over interval X in an interval B&B framework, can be done in several ways. Sharper bounds are better, i.e. higher values of $\underline{F}$ lead to more efficient performance of the B&B algorithm. Considering functions that have an additively separable structure (2), our research question is: for which cases

$$\underline{F} \leq \sum_{j=1}^{p} \underline{F}_j? \tag{5}$$

Alternatively, the question is to find ways to combine minorants on the separable terms, such that we get sharper bounds.

2. Taylor forms

Besides the standard IA bounding, called "natural interval extension" $F = [\underline{F}, \overline{F}]$ of f [4, 5], one can obtain an inclusion function of f using the inclusion function F' of f'. Consider the first order Taylor expression

$$T(c, X) := f(c) + (X - c)F'(X), \tag{6}$$

where $c \in X$. Notice that this expression is mainly of interest if the function is not monotonous on X, so at least $0 \in F'(X)$. By taking for c the middle $m = \frac{\underline{X}+\overline{X}}{2}$ of the interval, we have what is called a center form of the inclusion

In [1], Baumann proves that taking $c = b^-$ in the Taylor expression, leads to the best lower bound, where:

$$b^- = \begin{cases} \dfrac{\underline{X}_i\overline{F'}(X) - \overline{X}\underline{F'}(X)}{\overline{F'}(X) - \underline{F'}(X)} & , 0 \in F'(X) \\ \overline{X} & , \overline{F'}(X) \leq 0 \\ \underline{X} & , \underline{F'}(X) \geq 0 \end{cases}$$

So,

$$f(X) \geq \underline{T}(b^-, X). \tag{7}$$

An additively separable Baumann form bound $ASB(X)$ can be constructed in a straightforward way evaluating the Taylor expression (6) for the two sub-functions in their Baumann point and adding the resulting lower bounds,

$$f(X) \geq ASB(X) = \underline{T}_1(b_1^-, X) + \underline{T}_2(b_2^-, X). \tag{8}$$

Example 2. *Consider function $f(x) = f_1(x)+f_2(x) = (x+1)^2+(x-1)^2$ on the interval $X = [-2, 2]$. The minima of the sub-functions is 0, whereas the minimum of f itself is $f(0) = 2$. Figure 1 illustrates this idea and also draws lower bounds of all functions based on Baumann point. $\underline{T}(b^-, X) = -14$ and $\underline{T}_i(b_i^-, X) = -6$ such that $\underline{T}_1(b_1^-, X) + \underline{T}_2(b_2^-, X) = -12$, illustrating question (5).*

3. Lower Bound Value Form

Another way to compose derivative based linear minorants is the so-called Lower Boundary Value Form (LBVF), ([6] p. 60 and [2, 3]) that uses the evaluation of the end-points of the interval. Consider the most left point of X. Function

$$\varphi l(x) = \underline{F}(\underline{X}) + \underline{F'}(X)(x - \underline{X}), \tag{9}$$

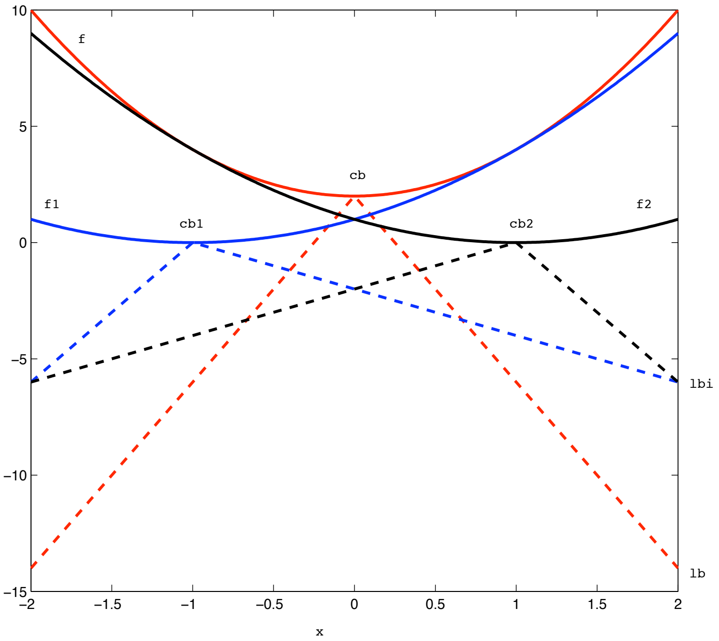


Figure 1. Quadratic illustration of (3) and (5)

provides an affine minorant. Similarly, the right most point of X provides

$$\varphi r(x) = \underline{F}(\overline{X}) - \overline{F'}(X)(\overline{X} - x) = \underline{F}(\overline{X}) + \overline{F'}(X)(x - \overline{X}). \tag{10}$$

The values $\underline{\varphi l}(\overline{X})$ and $\underline{\varphi r}(\underline{X})$ are lower bounds of $f(X)$ over X. A sharper lower bound can be obtained when $0 \in \overline{F'}(X)$ by combining (9) and (10) in lower bounding function

$$\varphi m(x) = \max\{\varphi l(x), \varphi r(x)\}. \tag{11}$$

The Lower Boundary Value Form $\underline{\varphi m}(X)$ follows from finding y for which (9) and (10) are equal

$$\underline{\varphi m}(X) = \varphi m(y) = \frac{\underline{F}(\underline{X})\overline{F'}(X) - \underline{F}(\overline{X})\underline{F'}(X)}{w(F'(X))} + \frac{w(X)\overline{F'}(X)\underline{F'}(X)}{w(F'(X))}. \tag{12}$$

So,

$$f(X) \geq \underline{\varphi m}(X),\ 0 \in F'(X). \tag{13}$$

An Additively Separable Lower Bound Value form can be constructed in the following way:

$$f(X) \geq ASLBV(X) = \underline{\varphi m}_1(X) + \underline{\varphi m}_2(X),\ 0 \in F_1'(X),\ 0 \in F_2'(X). \tag{14}$$

We focus further on the LBVF minorants of both sub-functions in order to obtain a sharper lower bound than $\underline{\varphi m}(X)$ without worrying about the monotonicity of the sub-functions for a given interval. Notice that only the case where the composite function f is not monotonous, $0 \in F'(X)$ is interesting. Consider the addition of the separate minorant terms

$$\varphi(x) = \varphi m_1(x) + \varphi m_2(x),$$

where φm_i is defined by (11). First of all, notice that φ is a piecewise linear minorant function and the maximum of four different affine terms:

$$\varphi(x) = \max\left\{\begin{array}{l} \varphi l(x) := \varphi l_1(x) + \varphi l_2(x) \\ \varphi a(x) := \varphi l_1(x) + \varphi r_2(x) \\ \varphi b(x) := \varphi r_1(x) + \varphi l_2(x) \\ \varphi r(x) := \varphi r_1(x) + \varphi r_2(x) \end{array}\right\}. \tag{15}$$

Then, one can see that $\varphi(x)$ is a sharper minorant than $\varphi m(x)$.

Theorem 3. *Let* $\forall x \in X, f(x) = f_1(x) + f_2(x)$ *and* φl, φr *and* φm *be defined by (9), (10) and (11).* $\forall x \in X, \varphi m_1(x) + \varphi m_2(x) \geq \varphi m(x)$.

Proof. Given equivalence (15), we have that

$$\varphi m(x) = \max\{\varphi l(x), \varphi r(x)\} \leq \max\{\varphi l(x), \varphi a(x), \varphi b(x), \varphi r(x)\} = \varphi(x).$$

□

Theorem 3 provides us with a new Additively Separable Lower Bound $ASLB\varphi$ defined by

$$f(X) \geq ASLB\varphi(X) = \underline{\varphi}(X) \tag{16}$$

4. Summary

For $ASLBV\varphi$, it is proven that the corresponding minorant is sharper than the standard one for LBVF. How to evaluate $ASLB\varphi(X)$ and numerical experiments will be shown in GOW 2012. Numerical results confirm this improving behaviour, although monotonicity of the subfunction and the composite function over an interval reduces this effect. Numerical results also show that separable variant for the Baumann lower bound is usually worse than the original one.

Future investigation could focus on the question how to extend the $ASLBV\varphi$ lower bound for n-dimensional functions. Another question is the derivation of specific interval based bounds for multiplicative terms.

References

[1] E. Baumann. Optimal centered forms. *BIT*, 28(1):80–87, 1988.

[2] L.G. Casado, I. García, J.A. Martínez, and Ya. D. Sergeyev. New interval analysis support functions using gradient information in a global minimization algorithm. *Journal of Global Optimization*, 25(4):345–362, 2003. DOI:10.1023/A:1022512411995.

[3] Pierre Hansen, Jean-Louis Lagouanelle, and Frédéric Messine. Comparison between Baumann and Admissible Simplex Forms in Interval Analysis. *Journal of Global Optimization*, 37:215–228, 2007. DOI:10.1007/s10898-006-9045-9.

[4] R.E. Moore. *Interval analysis.* Prentice-Hall, New Jersey, (USA), 1966.

[5] R.E. Moore, R.B. Kearfott, and M.J. Cloud. *Introduction to Interval analysis.* SIAM, Philadelphia, (USA), 2009.

[6] A. Neumaier. *Interval Methods for Systems of Equations.* Cambridge University Press, Cambridge, 1990.

Aircraft conflict avoidance: a mixed-integer nonlinear optimization approach

Sonia Cafieri

Laboratoire MAIAA, École Nationale de l'Aviation Civile, 7 Ave. E. Belin 31055 Toulouse, France sonia.cafieri@enac.fr

Abstract Detecting and solving aircraft conflicts, which occur when aircraft sharing the same airspace are too close to each other according to their predicted trajectories, is a crucial problem in Air Traffic Management. We focus on mixed-integer optimization models based on speed regulation. We first solve the problem to global optimality by means of an exact solver. The problem being very difficult to solve, we also propose a heuristic procedure where the problem is decomposed and it is locally solved by an exact solver. Computational results show that the proposed approach provides satisfactory results in reasonable time.

Keywords: air traffic management, conflict avoidance, MINLP, modeling, global exact solution, heuristic

1. Introduction

The air traffic level currently attained in Europe is around tens of thousands of flights per day and it is expected to be multiplied by a factor of two during next 20 years. Air traffic is therefore at the core of the social and economic dynamism of our society. The European project SESAR gives the guidelines to go towards an Air Traffic Management (ATM) characterized by more efficiency and more safety, which should essentially result from a higher level of automation of ATM. The need for automatical tools to integrate human work specially arises in the context of aircraft conflicts detection and resolution.

Aircraft potential conflicts can be solved in different ways. The most commonly exploited is based on the idea of achieving separation changing the trajectory (heading angle) or the flight level of the aircraft involved in the conflict. Another way is based on the idea of separating aircraft by slightly changing their speeds but keeping the predicted trajectories. A speed regulation which occurs in a reasonable small range allows a *subliminal control* as suggested by the European ERASMUS project [3]. This project showed the advantage of such a control, which is not even perceived by air traffic controllers. Conflict avoidance is expected to be performed while deviating as little as possible from the original aircraft flight plan, minimizing the impact of the separation maneuvers. Various solution strategies have been proposed. A review is provided in [6]. Solution algorithms are currently mainly based on evolutionary computation [4]. These methods are computationally efficient, but the global optimal solution and even a feasible solution (with no conflicts) is not guaranteed to be achieved in a given time. Recent advances in mixed-integer linear and nonlinear programming open new perspective for modeling and efficiently solving the addressed problem. The first attempt to use mixed-integer optimization is by Pallottino et al. in [8], where, though under very stringent hypothesis, a geometrical construction leads to a mixed-integer linear programming model. More recently, mixed-integer programming has been proposed again for aircraft conflict resolution (see e.g.[7]).

In this paper, the speed regulation strategy is modeled by mixed-integer nonlinear programming, building on [3]. A deterministic global solution is first proposed, using a general-purpose solver for MINLP. Then, to deal with the difficulty of the problem, another strategy is also proposed, where the optimality guarantee is forsaken in exchange for the computational efficiency. This solution strategy is based on hybridizing mathematical programming and a heuristic tailored on the problem.

The paper is organized as follows. In Sect. 2 we review mixed-integer nonlinear modeling of the aircraft conflict avoidance problem. In Sect. 3 we present a global exact solution of a few randomly generated instances and we propose a heuristic tailored on the problem to gain computational efficiency. Sect. 4 concludes the paper.

2. Modeling the aircraft conflict avoidance problem

Aircraft are said to be potentially *in conflict* when their horizontal or altitude distances are less than given standard separation distances (5NM and 1000 ft[1]). So, assuming the aircraft flying on a horizontal plane, the separation between aircraft i and j at the instant time t is expressed by the following condition:

$$||\mathbf{x}^r_{ij}(t)|| \geq d, \tag{1}$$

where d is the minimum required separation distance and $\mathbf{x}^r_{ij}(t)$ is the vector of relative distance between i and j. We assume, as usually done, that speed changes occur instantaneously. We can therefore consider uniform motion laws. Hence, for each t, we have:

$$\mathbf{x}^r_{ij}(t) = \mathbf{x}^{rd}_{ij} + \mathbf{v}^r_{ij}t, \tag{2}$$

where $\mathbf{x}^{rd}_{ij}$ is the relative initial position of aircraft i and j and $\mathbf{v}^r_{ij}$ is their relative speed. Observing that the minimum t is given by $t_m = -\mathbf{v}^r_{ij}\mathbf{x}^{rd}_{ij}/(v^r_{ij})^2$ the separation condition can be rewritten as follows:

$$(x^{rd}_{ij})^2 - \frac{(\mathbf{v}^r_{ij}\mathbf{x}^{rd}_{ij})^2}{(v^r_{ij})^2} - d^2 \geq 0. \tag{3}$$

Note that condition (3) has to be checked only when the inner product $\mathbf{v}^r_{ij}\mathbf{x}^{rd}_{ij}$ is negative. In this case, indeed, aircraft are converging.

In our model, conflict avoidance is achieved by performing a speed change maneuvre. Aircraft which are in conflict accelerate or decelerate in order to cross their conflict zone at different instant times, solving the conflict. Let A be the set of n aircraft. Decision variables are $q_k,\ k \in A$, expressing the percentage of speed change of each aircraft with respect to its original speed. As prescribed by the ERASMUS project [2], we impose bounds on these variables in order to have speed changes ranging between -6% and $+3\%$ of the original speeds. In this way, the so-called *subliminal control* is achieved. We minimize the speed change for each aircraft together with time intervals during which it flies with a modified speed, in order to deviate as less as possible from the original flight plan:

$$\min \sum_{k \in A} q_k^2 (t_{2k} - t_{1k})^2, \tag{4}$$

where t_{1k} and t_{2k} are decision variables representing starting and ending instant times for aircraft k changing its speed. The order of instant times when aircraft change speed being unknown, 6 possible time configurations have to be considered for each pair of aircraft and, for each of them, 5 time intervals $[t_s, t'_s]$. The constraints of the problem impose aircraft separation (3) for each time configuration and time interval. This needs the introduction of new (integer) variables and constraints. See [3] for details. The described model can be relaxed, for example imposing that aircraft speed changes occur at the instant time $t = 0$ and that the new speeds are kept during the trajectories. We consider in the following this relaxed model. First, we have to express aircraft speeds in (3) in terms of their original speed v and speed modification q. We also have to check if t_m is greater than 0. Equation for t_m gives rise to a constraint, for each pair of aircraft, defining the minimum instant time. To check if $t_m \geq 0$,

[1] 1 NM (nautical mile)= 1852 m, 1 ft (feet) = 0.3048 m

a binary variable y_{ij} for each $(i{,}j)$ is introduced ($y_{ij} = 1$ if $t_m \geq 0,$ and 0 otherwise), and constraints are adjoined accordingly. The separation condition is then imposed, for each pair of aircraft, only when $t_m \geq 0$:

$$\forall i,j \in A, i \neq j, \quad y_{ij}\left(\mathbf{x}^{rd}_{ij}(\mathbf{v}^{r}_{ij})^2 - (\mathbf{v}^{r}_{ij}\mathbf{x}^{rd}_{ij})^2 - (d^2(\mathbf{v}^{r}_{ij})^2)\right) \geq 0. \tag{5}$$

The obtained mathematical programming model has as many (nonlinear) separation constraints as pairs of aircraft.

3. Solving the conflict avoidance problem

3.1 Global exact solution

We use as a testbed n aircraft in 2-dimensional space, placed on a circle of a given radius r, with speed v and a heading angle such that their trajectory is toward the center of the circle (or slightly deviated with respect to such direction). The zone of conflict is around the center of the circle where aircraft are placed, and each aircraft is in conflict with each other. It is easy to see that the number of conflicts is $n(n-1)/2$, so a large number of conflicts is generated in the same conflict zone. We solve the problem to global optimality using COUENNE [1], which implements a spatial Branch-and-Bound based on convex relaxations. Results are reported in Table 1a (v=400 NM/h). They show that we are able to obtain global exact solutions up to $n = 6$ (i.e. 15 conflicts). However, an exact solution turns to be high memory and time demanding, even for a small number of aircraft, due to the high number of conflicts and the number of variables and constraints largely increasing with n. Hence, we are not able to solve the problem for $n > 6$ even with the relaxed modeling. Objective function values show that aircraft separation is always achieved with very slight speed changes.

3.2 A heuristic based on local exact solutions

We then propose a heuristic procedure where we solve at global optimality subproblems involving up to 4 aircraft at a time, based on the observation that a solution can be efficiently computed for problems involving a small number of aircraft.

Let a *cluster* be the transitive closing on conflicting pairs of aircraft (see, e.g., [5]). The heuristic is based on the idea of decomposing the problem in subproblems (clusters) and solve the conflict avoidance problem on clusters. Let ncl be the number of clusters. At each step, ncl problems are sequentially solved by using an exact solver (COUENNE). Combining together all the results, in general the conflicts are not all solved because aircraft inside clusters are typically in conflict with aircraft inside other clusters too. After the resolution step on subproblems, the number of remaining conflicts is computed. If it is greater than 0, a new step is performed. To do so, the initial speed (which together with the initial position represents the data of the problem) of aircraft that are still in conflict is re-initialized taking into account the solution obtained at the last step. That is, if the (optimal) solution obtained for cluster i is such that an aircraft in this cluster has been accelerated with respect to its original speed, then its speed is modified by a random slight further increase. If it has been decelerated, then its speed is modified by a random slight further decrease. In this way, the information obtained at the previous step is preserved and the chances to keep aircraft separated inside clusters increase. To update the speeds, a local search is performed testing a number of candidates and choosing the one that minimizes the sum, over all conflicting aircraft, of the maximum violation of the separation constraints for each considered aircraft, divided by the number of remaning conflicts. When only one conflict is to be solved, this search is intensified to increase the chances to solve the problem. Aircraft speeds have to be bounded in the small range $[-6\%v, +3\%v]$, so when speeds are modified these bounds have to be checked and speeds adjusted to fulfill this requirement. This may eventually lead to change the speed scenario provided by local solutions.

Results are reported in Table 1b. Values are averaged over 10 runs. For all test problems, all conflicts are solved. Comparing with global exact solutions (Table 1a), it appears that decomposing the problem does not significantly affect the quality of the result. In general, increasing n, faster solutions are obtained using a higher number of smaller subproblems. Solutions are obtained in reasonable time on problems involving up to 10 aircraft.

Table 1a. Results obtained with COUENNE

ID	n	r	*obj*	CPU time (sec.)
pb_n2	2	1×10^2	0.002531	0.15
pb_n3	3	2×10^2	0.001667	1.45
pb_n4	4	2×10^2	0.004009	12.87
pb_n5	5	3×10^2	0.003033	841.33
pb_n6	6	3×10^2	0.006033	51863.37

n= number of aircraft
r= radius of the circle (NM)
obj= objective function value
ncl= number of aircraft clusters

Table 1b. Results obtained with the proposed heuristic

ID	n	r	*ncl*	*obj*	CPU time (sec.)
pb_n4	4	2×10^2	2	0.005151	26.97
pb_n5	5	3×10^2	2	0.004729	17.98
pb_n6	6	3×10^2	2	0.006402	17.33
pb_n6	6	3×10^2	3	0.007438	341.12
pb_n7	7	3×10^2	2	0.009215	131.34
pb_n7	7	3×10^2	3	0.008144	22.99
pb_n8	8	4×10^2	2	0.008220	759.40
pb_n8	8	4×10^2	3	0.007551	39.66
pb_n8	8	4×10^2	4	0.012034	48.99
pb_n9	9	4×10^2	3	0.009238	97.41
pb_n10	10	4×10^2	3	0.014047	484.49

4. Conclusions

We presented an approach based on mixed-integer nonlinear optimization for the aircraft conflict avoidance problem. We are able to obtain global exact solutions for problems with up to 6 aircraft, while a new heuristic tailored on the problem and based on local exact solutions allow us to obtain good quality results even on problems involving many conflicts at a time.

References

[1] P. Belotti, J. Lee, L. Liberti, F. Margot, and A. Wächter. Branching and bounds tightening techniques for non-convex MINLP. *Optimization Methods and Software*, 24(4):597–634, 2009.

[2] D. Bonini, C. Dupré, and G. Granger. How erasmus can support an increase in capacity in 2020. In *Proceedings of the 7th Int. Conf. on Computing, Communications and Control Technologies, Orlando*, 2009.

[3] S. Cafieri and N. Durand. A mixed-integer optimization model for air traffic deconfliction. In *Proceedings of the Toulouse Global Optimization workshop*, 2010.

[4] N. Durand, J.M. Alliot, and J. Noailles. Automatic aircraft conflict resolution using genetic algorithms. In *Proceedings of the Symposium on Applied Computing, Philadelphia*. ACM, 1996.

[5] G. Granger and N. Durand. A traffic complexity approach through cluster analysis. In *Proceedings of the 5th ATM R&D Seminar, Budapest, Hungary*, 2003.

[6] J. Kuchar and L. Yang. A review of conflict detection and resolution modeling methods. *IEEE Trans. on Intelligent Transportation Systems*, 1(4):179–189, 2000.

[7] F.J. Martin-Campo. *The collision avoidance problem: methods and algorithms*. PhD thesis,Rey Juan Carlos University, 2010.

[8] L. Pallottino, E. Feron, and A. Bicchi. Conflict resolution problems for air traffic management systems solved with mixed integer programming. *IEEE Transactions on Intelligent Transportation Systems*, 3(1):3–11, 2002.

Proceedings of GOW 2012, pp. 47 – 51.

Branch and Bound Optimization with Heuristics, Sequencing and Ending Subset Optimization applied to Clusterwise Regression

Réal A. Carbonneau, Gilles Caporossi, and Pierre Hansen

GERAD and HEC Montréal, Montréal (Québec) Canada, H3T 2A7
real.a.carbonneau@gmail.com, gilles.caporossi@gerad.ca, pierre.hansen@gerad.ca

Abstract Optimizing the clusterwise regression problem using a branch and bound search enhanced by heuristics, observation sequencing and ending subset optimization is proposed and tested. This optimization strategy is an extension and a generalization of Brusco's repetitive branch and bound algorithm (RBBA). Heuristics can improve the upper bound, observation sequencing can improve the search path and can increase fathoming, while the ending subsets can recursively strengthen the lower bounds of the search. Additionally, symmetry breaking and incremental regression calculations are employed to further speed up the optimization. Experiments demonstrate that the proposed optimization strategy is significantly faster than CPLEX and that the combination of all the components is significantly faster than each one individually.

Keywords: Global optimization, Combinatorial optimization, Clusterwise regression, Branch and bound, Sequencing, Heuristics

1. Introduction

The clustering technique called clusterwise regression fits multiple lines or hyperplanes to mutually exclusive subsets of a dataset such that the sum of squared errors (SSE) from each observation to its cluster's line is minimized [1–3]. It is a cubic optimization problem defined by: the number of clusters (K), the number of independent dimensions (D), and the number of observations (O). The iterators are for: a cluster ($k \in \{1,\ldots,K\}$), an independent dimension ($d \in \{1,\ldots,D\}$), and an observation ($o \in \{1,\ldots,O\}$). The model parameters are: the independent variable for an observation and dimension (x_{od}) and the dependent variable for an observation (y_o). The model variables are: the cluster assignment of an observation to a cluster (z_{ok}), the regression coefficient (aka β) for a dimension of a cluster (b_{dk}) and the error for an observation of a cluster (e_{ok}). The cubic model is as follows:

$$\text{SSE } = \min \sum_{k=1}^{K}\sum_{o=1}^{O}\left(z_{ok}e_{ok}^2\right) \tag{1}$$

$$\text{s.t. } \sum_{d=1}^{D}(b_{dk}x_{od}) + e_{ok} = y_o \qquad o = 1,\ldots,O, \quad k = 1,\ldots,K \tag{2}$$

$$\sum_{k=1}^{K} z_{ok} = 1 \qquad o = 1,\ldots,O \tag{3}$$

$$z_{ok} \in \{0,1\} \qquad o = 1,\ldots,O, \quad k = 1,\ldots,K \tag{4}$$

The objective (1) is the minimization of the sum over all clusters of the sum of squared errors (SSE) for their observations relative to their regression line. The constraint (2) fits the regression lines to the data by adjusting the coefficient and error terms. An observation can only be assigned to one cluster at a time (3) and the cluster assignment is binary (4).

2. Optimization Methods

The cubic clusterwise regression mathematical programming formulation can be re-formulated into a quadratic problem. However the big-M mixed integer quadratic programming (MIQP) formulation cannot guarantee globally optimal results [4]. This occurs because the slope of a line can be arbitrarily steep, and thus the error, can be arbitrarily large. Consequently, the big-M constant cannot be guaranteed to be large enough for any possible optimal solution nor for any possible intermediate solution of the MIQP branch and bound procedure. In an approach called mixed logical-linear programming (MLLP), or in this case, mixed logical-quadratic programming (MLQP), the logical propositions remain in their natural formulation while at the same time taking advantage of the strength of both logic processing and linear or quadratic programming [5–8].

A branch and bound algorithm can also be used for solving the clusterwise regression problem optimally. Although this is a difficult task, symmetry breaking, identifying stronger bounds and controlling the path through the search space can reduce the actual size of the search and incremental regression calculations will reduce the number of operations for each evaluation. The upper bound can be strengthened by heuristic optimization. The lower bound can be strengthened by exact global optimization of ending subsets as in RBBA [9]. An appropriate observation sequence can strengthen the bounds and enhance fathoming.

Symmetry breaking is performed in the branch and bound algorithm by only using an additional cluster (up to K) when needed. Incremental regression calculations [10, 11] are performed at every node simply by adding the current observation to the target cluster using the calculation cache retained for the parent node. For the heuristics, since there is no known way to determine a priori how much optimization effort should be put into heuristic optimization before starting the exhaustive branch and bound search, an iterative heuristic optimization is used. In the current experiments, approximately 10% of processing time is spent on heuristic optimization in time slices of 1 second.

Next, as detailed in the repetitive branch and bound algorithm (RBBA) [9], stronger lower bounds can be calculated using the optimal solutions of ending subsets. Additionally, when using incrementally larger ending subsets and performing the branch and bound search in the forward direction, the search can benefit recursively from the lower bounds. Both the sequence and direction of optimizing incrementally larger ending subsets are critical to achieve the most benefits. The branch and bound algorithm for an individual ending subset must process the observations in the forward direction, which permits the use of the lower bounds identified by previous smaller ending subset searches. However, the incrementally larger subsets always end at the last observation, thus the subsets are increased in size in the reverse direction, meaning the starting observation backs up, but the ending observation is always the last of the complete set. Additionally, only a limited number of ending subsets are optimization. In this experiment, steps of 10 observations are used and planed such that last 10 largest ending subsets are skipped since they are often the longest to optimized other than the complete set.

Finally, sequencing can greatly affect the effort required to solve an optimization problem. Although an algorithm may be deterministic and even exact, thus always finding the global optimum, the path taken through the search space may vary greatly simply as a result of the sequence in which the data points are processed. Extreme variations in processing times can occur in branch and bound algorithms (Figure 2). From these and other results, it seems that the processing time follows a log-normal distribution.

The proposed sequencing strategy is to sequence the observations in descending order of error from their cluster identified by heuristic optimization, while forcing alternating clusters. Thus, the first observation in the sequence will be the one with the largest error of those assigned to cluster 1, the next from cluster 2, etc, up to cluster K. Subsequently, the observations with the next largest error will follow. When the branch and bound algorithm processes this sequence of observations, there will rapidly be a representative sample in each cluster and these observations will increase the error faster than an average random sequence. In addition, incremental ending subset optimization will also have a representative sample of

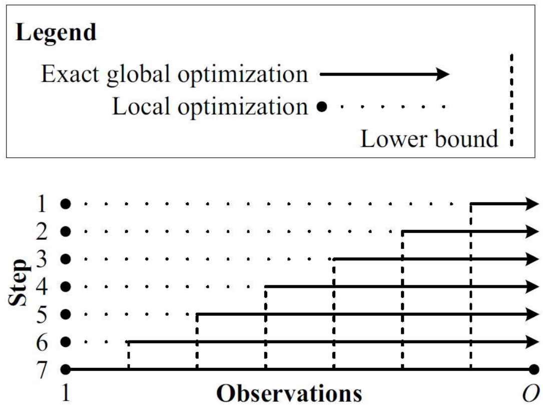


Figure 1. Overview of incremental ending subset optimization.

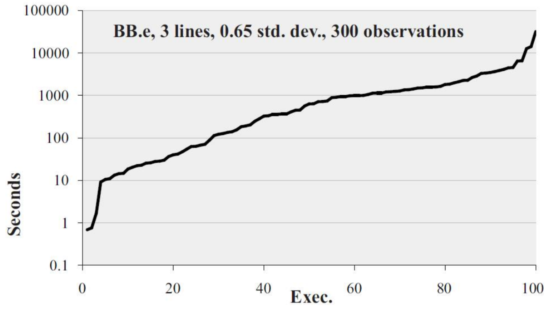


Figure 2. Cumul. distr. of proc. times for BB optim. with ending subsets, 3 lines, 0.65 std. dev., 300 obs.

observations, with lower error, thus their processing times will be a fraction of that of the complete set.

Most of the main components of the presented strategy are general to any clustering problem for which an incremental calculation can be defined. Although there may exist specialized efficient heuristics for specific problems, there also exists heuristics, such as tabu search (TS), genetic search (GS), variable neighborhood search (VNS), simulated annealing (SA), that can minimize any function, thus is it assumed that the heuristic parts of the presented algorithm can also be generalized to most clustering problems. On the other hand, the incremental and error calculations are specific to each clustering problem. These may also require problem specific optimizations since these calculations are the innermost loop of the complete algorithm, and thus can have a large impact on the algorithm's performance. For applications to other branch and bound problems, the change to the objective function would be considered instead of the problem specific term error used in this article.

In this paper, the mixed logical-quadratic programming model is optimized using the IBM ILOG OPL-CPLEX (OPL 6.3 and CPLEX 12.1.0) environment, thus the model is implemented using the OPL programming language [12–14].

Table 1. Algorithm codes and descriptions.

Code	Description
CPLEX	Mixed logical-quadratic formulation optimized by CPLEX
BB...	Branch and bound optimization
...h...	Iterative heuristic optimization
...s1...	Sequencing by descending error in cluster with forced alternating of clusters
...e	Incremental ending subsets optimization with a step size of 10

3. Results and discussions

Optimization of the clusterwise regression problem is performed on a synthetic series of datasets with two and three lines (Figure 3) and increasing perturbations.

The results presented in Table 2 indicate that both CPLEX and the simpler branch and bound algorithms (BB and BB.h) were relatively inefficient at solving the problem compared individual parts of the proposed algorithm (BB.e, BB.h.s) and the complete combination (BB.h.s.e). Combining heuristic optimization with sequencing by descending within cluster error and ending subset searches (BB.h.s.e) provided the fastest optimization times.

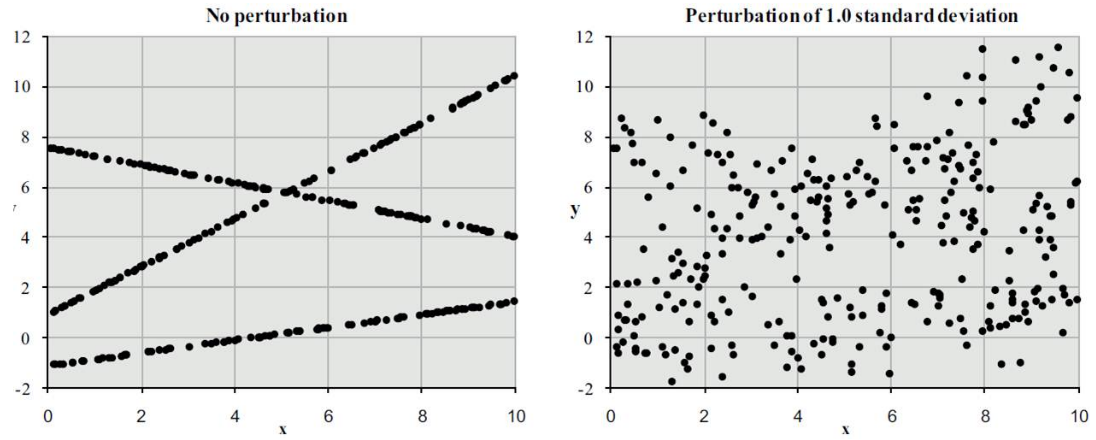


Figure 3. Plots of the three lines dataset without perturbations and of one standard deviation.

References

[1] Charles, C. (1977). Régression typologique et reconnaissance des formes. PhD Thesis, Université de Paris IX.

[2] Diday, E. (1979). Optimisation en Classification Automatique. Le Chesnay: INRIA.

[3] Späth, H. (1979). Algorithm 39 Clusterwise linear regression. Computing, 22(4):367–373.

[4] Carbonneau, R.A., Caporossi, G. and Hansen, P. (2011). Globally optimal clusterwise regression by mixed logical-quadratic programming. European Journal of Operational Research, 212(1):213–222.

[5] Hooker, J.N. and Osorio, M.A. (1999). Mixed logical-linear programming. Discrete Applied Mathematics, 96:395–442.

Table 2. Statistics for 2 lines dataset, 300 observations, varying normal distribution perturbation, average seconds for 100 randomized sequences.

Std.		**BB.h.s.e**	**BB.e**	**BB.h.s**	**BB.h**	**BB**	**CPLEX**	
Dev.	*K*	**Avg.**	**Avg.**	**Avg.**	**Avg.**	**Avg.**	**Avg.**	**Optimal SSE**
0.00	2	0.1	0.0	0.1	0.1	> 1h	146.8	0.0025009204738
0.10	2	0.1	0.0	0.1	2.7		2147.6	3.2665102368
0.20	2	0.1	0.0	0.1	> 1h		> 1h	12.335947506
0.30	2	0.1	0.3	0.2				27.455510479
0.40	2	0.1	1.2	4.8				48.926938261
0.50	2	0.2	115.7	155.9				76.470119913
0.60	2	0.6	> 1h	> 1h				109.22501160
0.70	2	2.2						147.70999194
0.80	2	4.7						190.08004822
0.90	2	20.4						237.10335939
1.00	2	1056.3						288.47061601
StDv	n/a	2.4%	735.1%	2.8%	109.2%	n/a	392.1%	n/a
0.00	3	0.1	0.0	0.1	0.1	> 1h	> 1h	0.0026426623973
0.10	3	0.1	0.0	0.1	0.2			3.0860096710
0.20	3	0.1	0.0	0.1	39.6			11.549167409
0.30	3	0.1	0.1	0.2	> 1h			25.829580541
0.40	3	0.1	0.4	2.7				46.237962733
0.50	3	0.2	18.9	181.0				72.557256342
0.60	3	1.3	1088.8	> 1h				103.91878126
0.70	3	7.2	> 1h					140.01306220
0.80	3	237.1						179.53596560
StDv	n/a	7.0%	529.8%	1.9%	116.7%	n/a	n/a	n/a

[6] Hooker, J.N., Ottosson, G., Thorsteinsson, E.S. and Kim, H.J. (2000). A scheme for unifying optimization and constraint satisfaction methods. The Knowledge Engineering Review, 15(1):11–30.

[7] Hooker, J.N. (2007). Integrated Methods for Optimization. New York: Springer.

[8] Hooker, J.N. (2002). Logic, optimization, and constraint programming. INFORMS Journal on Computing, 14(4):295–321.

[9] Brusco, M.J. (2006). A repetitive branch-and-bound procedure for minimum within-cluster sums of squares partitioning. Psychometrika, 71(2):347–363.

[10] Gentleman, W.M. (1974). Algorithm AS 75: Basic procedures for large, sparse or weighted linear least problems. Applied Statistics, 23(3):448–454.

[11] Gentleman, W.M. (1973). Least squares computations by given transformations without square roots. IMA Journal of Applied Mathematics, 12(3):329–336.

[12] IBM. (2009). IBM ILOG CPLEX 12.1.0. Armonk, NY: IBM.

[13] IBM. (2009). IBM ILOG OPL 6.3. Armonk, NY: IBM.

[14] Van Hentenryck, P., Lustig, I., Michel, L. and Puget, J.F. (1999). The OPL Optimization Programming Language. Cambridge: MIT Press.

A biased random-key genetic algorithm for the robust shortest path problem

Amadeu A. Coco,[1] Thiago F. Noronha,[1] and Andréa C. Santos[2]

[1] *Universidade Federal de Minas Gerais, DCC,*
Avenida Antônio Carlos 6627, Belo Horizonte, Brazil,
{amadeuac,tfn}@dcc.ufmg.br

[2] *ICD-LOSI, Université de Technologie de Troyes*
12, rue Marie Curie, BP 2060, 10010, Troyes CEDEX, France.
andrea.duhamel@utt.fr

Abstract A Biased Random-Key Genetic Algorithm (BRKGA) is proposed in this paper for the NP-hard Robust Shortest Path problem (RSP). To the best of our knowledge, this is the first metaheuristic for the RSP with interval data and with min-max regret optimization criteria. Computational experiments show that the BRKGA found optimal solutions for instances with up to 1000 vertices.

Keywords: Robust shortest paths, Robust optimization, Genetic algorithms, Interval data

1. Introduction

Given a connected digraph $G = (V, A)$ with a set V of vertices and a set A of arcs. Let $c_{ij} \in \mathbb{R}$ be the cost associated to each arc $(i, j) \in A$. The well-known Shortest Path problem (SP) consists in finding the shortest path from a source $s \in V$ to a destination $t \in V$ such that the total cost is minimized. A solution exists if there is no negative-weight cycle reachable from s to t . Several polynomial-time algorithms are available to solve SP. The Robust Shortest Path problem (RSP) extends SP. The cost associated to each arc is defined as an interval $[l_{ij}, u_{ij}]$, where $u_{ij}, l_{ij} \in \mathbb{R}$ and $u_{ij} \geq l_{ij}$ [7], or else several discrete values are associated to each arc [11].

Let a scenario r be an assignment of arc costs $c^r_{ij} \in [l_{ij}, u_{ij}]$, $\forall (i, j) \in A$. RSP aims at selecting a path from a source node $s \in V$ to a destination node $t \in V$. However, since the scenario that will occur is not known, the path computation has to be done under uncertainty. This problem can be solved by using robust optimization methods [3]. A solution is robust whenever it is the most adapted over all the scenarios considered. The optimization criteria strongly depends on the application needs. For instance, it is important to avoid the worst scenario when the decisions involve the monitoring of industrial sites and competition [1]. For such applications, the min-max criterion can be applied and, one seeks a solution whose worst case scenario has the best total cost.

In this paper, we tackle a more specialized optimization criterion called *min-max regret*. Let r be a scenario, that is a realization of arc costs $c^r_{ij} \in [l_{ij}, u_{ij}]$, $\forall (i, j) \in A$, and $\mathcal{P} \subseteq A$ be a path from s to t in G. The *robust deviation* of $\mathcal{P}$ in the scenario r is defined as the difference between the cost of $\mathcal{P}$ in r and the cost of the shortest path S^r from s to t in r. In other words, the *robust deviation* of $\mathcal{P}$ in r is the regret of using $\mathcal{P}$ instead of S^r in case scenario r occurs. The min-max regret RSP consists in finding a path $\mathcal{P}$ from s to t which minimizes its worst deviation over all possible scenarios. This problem is shown to be NP-hard in [11].

A pseudo-polynomial algorithm has been introduced in [11]. Preprocessing techniques for RSP with interval data are proposed in [4, 7]. They remove dominated arcs on special graphs such as acyclic and planar graphs. Instances with up to 400 nodes and arc density 0.3 were solved to optimality. A branch-and-bound algorithm is proposed in [8]. This algorithm solves instances with up to 500 nodes and arc density 0.01 and two real instances. A heuristic strategy is introduced in [5] to manage the set of critical tasks in a project under uncertainty. A RSP survey is found in [1].

To the best of our knowledge, this is the first work to propose a metaheuristic for the min-max regret RSP with interval data. A Mixed Integer Linear Programming formulation (MILP) is presented in Section 2. A biased-random key genetic algorithm is proposed in Section 3. Computational experiments and concluding remarks are drawn in Section 4.

2. MILP formulation

The MILP formulation (1)-(6) for the min-max regret RSP has been proposed in [7]. Let y_{ij} be the decision variables on the choice of arc (i,j) in the path. If arc (i,j) belongs to the solution $y_{ij} = 1$, otherwise $y_{ij} = 0$. Moreover, variables $x_i \geq 0, \forall i \in V$ define the cost of the shortest path from the source node s to node i in the worst case scenario for the path defined by variables y.

$$\min \quad z = \sum_{(i,j)\in A} u_{ij} \cdot y_{ij} - x_t \quad \text{subject to:} \tag{1}$$

$$\sum_{(j,k)\in A} y_{jk} - \sum_{(i,j)\in A} y_{ij} = \begin{cases} 1, & \text{if j = s} \\ -1, & \text{if j = t} \quad \forall j \in V \\ 0, & \textit{otherwise} \end{cases} \tag{2}$$

$$x_j \leq x_i + l_{ij} + (u_{ij} - l_{ij})y_{ij} \quad \forall (i,j) \in A \tag{3}$$

$$x_s = 0 \tag{4}$$

$$y_{ij} \in \{0,1\} \quad \forall (i,j) \in A \tag{5}$$

$$x_i \geq 0 \quad \forall i \in V \tag{6}$$

The objective function (1) minimizes the regret at the destination node t. Constraints (2) are the classic flow conservation constraints and ensure the path connectivity from nodes s to t. Inequalities (3) link variables y and x and determine a regret upper bound at node j. These constraints together with the objective function ensure the maximum regret to be minimized. Equation (4) sets x_s to zero. The variables are defined in (5) and (6).

3. Biased random-key genetic algorithm

The Biased Random-Key Genetic Algorithm (BRKGA) [6] is based on Bean [2]. It has been successful applied for a number of network applications which motivates this work. The chromosomes in BRKGA are real-valued vectors in the range $[0, 1]$. Each element of the vector is called a *key* and its value is randomly generated in the initial population. In our BRKGA-RSP heuristic, each chromosome represents a possible scenario for the cost of the arcs in A. In other words, a chromosome q has a key $k^q_{ij} \in [0, 1]$ for each arc $(i,j) \in A$, and the cost of this arc in the scenario r^q is defined as $c^q_{ij} = l_{ij} + (u_{ij} - l_{ij}) \cdot k^q_{ij}$. This scenario induces a non dominated path $\mathcal{P}^q$, which is the shortest path from s to t in that scenario. The robust cost of $\mathcal{P}^q$ is used as the fitness of the chromosome. The asymptotic worst case complexity to evaluate the fitness of a chromosome is $O(|V| \cdot \log |V|)$.

We use the *parameterized uniform crossover* scheme proposed in [9] to combine two parent solutions and produce an offspring solution. In this scheme, each key of the offspring comes from the best parent with probability 0.7 and from the worst parent with probability 0.3.

The proposed genetic algorithm does not use the standard mutation operator, where parts of the chromosomes are changed with small probability. Instead, the concept of *mutants* is used. In each generation, a fixed number of mutant solutions are introduced in the population. They are randomly generated the same way as the initial population. This operation plays the role of helping the procedure to escape from local optima.

At each new generation, the population is partitioned into two sets: TOP and $REST$. Consequently, the size of the population is $|TOP| + |REST|$. The best solutions are kept in TOP while the others are placed in $REST$. The chromosomes in TOP are selected, without any change, to be in the population of the next generation. The new mutants are placed in set BOT. The remaining elements of the new population are obtained by crossover with one parent randomly chosen from TOP and the other from $REST$. This distinguishes a biased random-key genetic algorithm from the random-key genetic algorithm of Bean [2]. In the latter both parents are randomly selected from the entire population. Since a parent solution can be chosen for crossover more than once in a given generation, elite solutions have a higher probability of transmitting their keys to the next generation. This way, $|REST| - |BOT|$ offspring solutions are created. The sizes of sets TOP, $REST$, and BOT are parameters to be tuned.

4. Experiments and concluding remarks

The heuristic `BRKGA-RSP` has been implemented in C++ and compiled with the *GNU GCC* compiler version 4.4.3. We use the BRKGA framework provided by [10]. In order to evaluate the performance of the proposed heuristic, optimal solutions were computed from the MILP formulation on CPLEX version 10.2 under default parameters, and the running time has been limited to two hours. The experiments were performed on an Intel Core 2 quad CPU with 2.5 GHz of clock and 8 Gb of RAM memory. The size of sets TOP, $REST$ and BOT was respectively set to 10, 90, and 20.

Several test sets of instances have been generated and experiments are reported for a subset of three test sets. The two first sets of instances are similar to those used in [5]. They mainly differ on the way the graph connectivity is ensured. The vertices are randomly generated in a square of sides 100 using a uniform low and only considering integer coordinates. The edges are randomly generated with probabilities 0.6 and 0.8. In STAR instances, the graph connectivity is ensured by linking one node (randomly chosen) with every other in the graph. In CYCLE instances, the graph connectivity is ensured by linking the set of vertices by an arbitrary Hamiltonian cycle. For STAR and CYCLE benchmarks, the lower bound l_{ij} for the cost of each arc $(i, j) \in A$ is randomly generated on the interval $]0, 200]$, and the respective upper bound u_{ij} is set to $l_{ij}(1+\alpha)$, where α is the *degree of uncertainty* parameter set to 0.25. The third test set contains GRID graphs. The interval costs associated to each arc are as follows: lower bounds l_{ij} are randomly selected in the interval $]0, 100]$, and upper bounds u_{ij} are set to $(l_{ij} + 10^5)$. For GRID instances, the origin and the destination nodes are respectively located at the lower left corner and at the upper right corner of the GRID.

Numerical results are reported in Table 1. Each line corresponds to an instance. Column 1 displays the instance name and "c", "s", "g" stand respectively for CYCLE, STAR, and GRID instances. Columns 2 to 4 give, respectively, the lower bound (lb), the upper bound (ub), and the CPU time (t(s)) in seconds spent by the branch-and-cut algorithm to solve the instances. The last three columns display respectively the average robust cost (avg.) over 10 runs, the average standard deviation (std. dev.), and the average CPU time (t(s)) in seconds spent by `BRKGA-RSP`. Results indicate the proposed BRKGA-RSP is able to find optimal solutions for all instances in these sets. GRID instances are more difficult to be solved. CPLEX solver is not able to find optimal solutions for the grids of size 9 (g_9x90) and 10 (g_10x100) in two hours. The explanation is probably that the diameter is longer in GRID graphs. Even for the largest instances, BRKGA-RSP is able to find good solution in a small running time.

These results motivate further studies on heuristics and metaheuristics for RSP. Future works will focus on providing a local search for RSP and on integrating them into genetic algorithms, and other metaheuristics. We also intend to identify harder instances and to study the impact of the cost interval for RSP.

Table 1. Performance evaluation of BRKGA-RSP.

	CPLEX			BRKGA-RSP		
Name	lb	ub	t(s)	avg.	std. dev.	t(s)
c_v750_d0.6	3.00	3.00	23.81	3.00	0.00	5.55
c_v750_d0.8	3.00	3.00	113.17	3.00	0.00	7.16
c_v1000_d0.6	0.00	0.00	3.79	0.00	0.00	6.75
c_v1000_d0.8	2.00	2.00	50.84	2.00	0.00	11.22
s_v750_d0.6	1.00	1.00	11.06	1.00	0.00	5.12
s_v750_d0.8	2.00	2.00	17.32	2.00	0.00	6.07
s_v1000_d0.6	3.00	3.00	122.93	3.00	0.00	9.05
s_v1000_d0.8	2.00	2.00	46.92	2.00	0.00	9.48
g_6x60	639,748.00	639,748.00	767.54	639,847.40	99.96	79.65
g_7x70	749,676.00	749,676.00	1465.89	749,874.10	196.15	132.02
g_8x80	859,803.00	859,803.00	6352.67	860,157.90	320.85	162.25
g_9x90	925,988.56	970,261.00	7200.00	970,566.60	196.92	159.29
g_10x100	1,027,501.40	1,080,339.00	7200.00	1,084,779.00	8439.80	193.00

References

[1] H. Aissi, C., and D. Vanderpooten. Min-max and min-max regret versions of combinatorial optimization problems: A survey. *European Journal of Operational Research*, 197:427–438, 2009.

[2] J.C. Bean. Genetic algorithms and random keys for sequencing and optimization. *ORSA Journal On Computing*, 2:154–160, 1994.

[3] A. Ben-Tal and A. Nemirovski. Robust optimization – methodology and applications. *Mathematical Programming*, 92:453–480, 2002.

[4] D. Catanzaro, M. Labbé, and M. Salazar-Neumann. Reduction approaches for robust shortest path problems. *Computers & Operations Research*, 38:1610–1619, 2011.

[5] E. Conde. A minmax regret approach to the critical path method with task interval times. *European Journal of Operational Research*, 197:235–242, 2009.

[6] J. F. GonÃğalves and M. G. C. Resende. Biased random-key genetic algorithms for combinatorial optimization. *Journal of Heuristics*, 17:487–525, 2011.

[7] O. E. Karasan, H. Yaman, and M. Ç. Pinar. The robust shortest path problem with interval data. Technical report, Bilkent University, Department of Industrial Engineering, 2001.

[8] R. Montemanni and L. M. Gambardella. A branch and bound algorithm for the robust spanning tree problem with interval data. *European Journal of Operational Research*, 161:771–779, 2005.

[9] W. Spears and K. DeJong. On the virtues of parameterized uniform crossover. In R. Belew and L. Booker, editors, *Proceedings of the Fourth International Conference on Genetic Algorithms*, pages 230–236, San Mateo, 1991.

[10] R. F. Toso and M. G. C. Rezende. A C++ application programmming interface for biased random-key algorithms, August 2011. http://www2.research.att.com/ mgcr/papers.html[Online; Accessed in 13/01/2012].

[11] G. Yu and J. Yang. On the robust shortest path problem. *Computers & Operations Research*, 25:457–468, 1997.

A MINLP Model for Crude Oil Selection and Refining

Ormeu Coelho,[1,2] Laura Bahiense,[2] and Virgílio J. M. Ferreira Filho[2,3]

[1] *CEFET-RJ / Celso Suckow da Fonseca, Department of Production Engineering, R. General Canabarro, 229, CEP 20271-110, Rio de Janeiro-RJ, Brazil,* ormeucoelho@gmail.com

[2] *COPPE-UFRJ, Production Engineering Program, Rio de Janeiro-RJ, Brazil,* laura.bahiense@gmail.com

[3] *UFRJ, Polithenic School, Department of Production Engineering, Rio de Janeiro-RJ, Brazil,* virgilio@ufrj.br

Abstract In this work a real-based refining planning problem is solved by MINLP techniques. The developed model aims to maximize the refiner profity by integrating decisions on operation of processing units and crude oil selection. An illustrative instance was solved by the global optimizer BARON 7.5.3 and by the (heuristic) AOA, both avaible in AIMMS 3.11.

Keywords: Oil Refining, Production Planning, Nonconvex MINLP

1. Introduction

The model presented in this work is an extension of the multiperiod formulation proposed by [1]. In each period of the planning horizon there are deterministic demands for oil derivatives (expressed as minimum and maximal volumes to be sold), and available volumes of crude oils and other inputs. For each crude oil two quantities may be available. A quantity that refinery must receive from supply source but not necessarily consume. And additional quantities that could be ordered if necessary, but within certain limits. Stocks of all streams (crude oils/inputs, intermediate and final products) could be carried from one period to the following.

The objective function aims to maximize the profit defined by revenues obtained by derivative sales minus costs incurred with raw material consumption and storage in tanks. For doing so, it requires the computation of flows among units and stock levels in each tank, its physicochemical properties, and operational variables on conversion units. Some decisions about flows have a discrete behaviour since it is usually required that they be bigger than a predefined minimum values. In this sense, in each planning period, one should determine which campaigns will be allocated to processing units and what kinds of additional crude oils (inputs) should be ordered. Preliminare results with this formulation were published in [2].

Given the large number equations defined by sums of bilinear and tri-linear terms, the resulting non-convex MINLP is highly nonlinear. Even having lot less integer variables than continuous ones, the generated programs are hard to solve to optimality.

2. Refining Planning and Input Selection Problem

In the following the Refining Planning and Input Selection Problem (RPISP), a non-convex MINLP, is mathetically defined.

- Sets:

U - set of units.

C_u - set of campaigns of unit u.

$S_{uc}^{I(O)}$ - set of inlet(outlet) streams at unit u during campaign c .

F - set of flows (u, c, s, u', c') considered in the planning scenario.

T - set of periods of the planning horizon.
$P^{V(M)I}_{uc}$ - set of volumetric (mass) properties of the load of unit u during campaign c.
$P^{V(M)O}_{ucs}$ - set of volum. (mass) properties of stream s that leaves unit u during campaign c.

- Parameters:
$K^{C-MIN(MAX)}_{uct}$ - min. (max.) load in processing unit u during campaign c in period t.
$K^{U-MIN(MAX)}_{ut}$ - min. (max.) load in processing unit u in period t.
Q^{F}_{ust} - flow rate of crude oil s (to be processed) available in source unit u in period t.
$Q^{A-MIN(MAX)}_{ust}$ - min. (max.) flow rate of additional crude oil (input) s available in source unit u in campaign (market) c in period t.
$H^{MIN(MAX)}_{ust}$ - min. (max) stock of stream s in storage unit (tank) u in period t.
$V^{MIN(MAX)}_{ucvt}$ - min. (max.) value of operational condition v of conversion unit u during campaign c in period t.

- Decision variables:
$q_{ucsu'c't}$ = flow rate of stream s from (u, c) to (u', c') in period t.
h_{uct} = stock level of stream s in storage unit (tank) u during campaign c in period t.
vo_{ucvt} = value of operational var. v in conversion unit u during campaign c in period t.
pi_{ucpt} = value of property p of the load that fed unit u during campaign c in period t.
po_{ucspt} = value of property p of stream s that leaves unit u during campaign c in period t.
y_{uct} =1, if campaign c is set for processing unit u in period t, 0, otherwise.
z_{ucst} =1, if additional crude oil s is bought at campaign c of unit u in period t, 0, otherwise.

- Objective function:

$$\max z = \sum_{ucst:u\in U^D} SP_{ucst} \sum_{u'c'} q_{u'c'suct} - \sum_{ucst:u\in U^H} C^H_{ucst} h_{ucst} \\ - \sum_{ucst:\in U^S} \left[C^{FI}_{ust} Q^F_{ucst} - C^A_{ust} \left(\sum_{u'c'} q_{ucsu'c't} - Q^F_{ucst} \right) \right]. \tag{1}$$

- Separation processes constraints:

$$\sum_{u'c'} q_{ucsu'c't} = f\left(q_{u'c's'uct}\right), \quad \forall u \in U^{PS}, c \in C_u, s \in S^O_{uc}, t \in T \tag{2}$$

$$po_{ucspt} = f\left(q_{ucsu'c't}, q_{u"c"s"uct}, po_{ucs'pt}\right), \quad \forall u \in U^{PS}, c \in C_u, s \in S^O_{uc}, p \in P^{VO}_{ucs} \bigcup P^{MO}_{ucs}, t \in T \tag{3}$$

- Conversion processes constraints:

$$\sum_{u'c'} q_{ucsu'c't} = f\left(\sum_{u'c's'} q_{u'c's'uct}, pi_{ucpt}, vo_{u,c,t} \right), \quad \forall u \in U^{PC}, c \in C_u, s \in S^O_{uc}, t \in T \tag{4}$$

$$po_{ucspt} = f\left(pi_{ucpt}, vo_{uct}\right), \quad \forall u \in U^{PC}, c \in C_u, s \in S^O_{uc}, p \in P^{VO}_{ucs} \bigcup P^{MO}_{ucs}, t \in T \tag{5}$$

- Mixture and storage processes contraints:

$$\sum_{u'c's} q_{u'c'suct} = \sum_{u'c's} q_{ucsu'c't}, \quad \forall u \in \bigcup_{i=M,H,PC} U^i, c \in C_u, t \in T \tag{6}$$

$$pi_{ucpt} \sum_{u'c's} q_{u'c's'uct} = \sum_{u'c's} po_{u'c'spt} q_{u'c'suct}, \quad \forall u \in \bigcup_{i=M,H,PC} U^i, c \in C_u, p \in P^{VI}_{uct}, t \in T \tag{7}$$

$$pi_{ucpt} pi_{ucp't} \sum_{u'c's'} q_{u'c's'uct} = \sum_{u'c's} po_{u'c'spt} po_{u'c'sp't} q_{u'c'su'c't}, \\ \forall u \in \bigcup_{i=M,H,PC} U^i, c \in C_u, p \in P^{MI}_{u,c,t}, t \in T \tag{8}$$

$$h_{uct} = h_{uct-1} + \sum_{u'c's} q_{u'c'suct} - \sum_{u'c's} q_{ucsu'c't}, \quad \forall u \in U^H, c \in C_u, t \in T \tag{9}$$

$$po_{ucspt} \left(\sum_{u'c's'} q_{u'c's'uct} + h_{uct-1}. \right) = po_{ucspt-1} h_{uct-1} + pi_{ucpt} \sum_{u'c's'} q_{u'c's'uct},$$
$$\forall u \in U^H, c \in C_u, s \in S^O_{uc}, p \in P^{VI}_{ucpt} \bigcap P^{VO}_{ucspt}, t \in T \tag{10}$$

$$po_{ucspt} po_{ucsp't} \left(\sum_{u'c's'} q_{u'c's'uct} + h_{uct-1}. \right) = po_{ucspt-1} po_{ucsp't-1} h_{u,c,t-1} +$$
$$pi_{ucpt} pi_{ucp't} \sum_{u'c's'} q_{u'c's'uct}, \quad \forall u \in U^H, c \in C_u, s \in S^O_{uc}, p \in P^{MI}_{ucpt} \bigcap P^{MO}_{ucspt}, t \in T \tag{11}$$

- Final products specificafion constraints:

$$P^{MIN}_{ucspt} \leq po_{ucspt} \leq P^{MAX}_{ucspt}, \quad \forall u \in U^H, c \in C_u, s \in S^O_{uc}, p \in P^{VO}_{ucspt} \bigcup P^{MO}_{ucspt}, t \in T \tag{12}$$

- Campaign allocation constraints:

$$Q^{C-MIN}_{uct} y_{uct} \leq \sum_{u'c's} q_{ucsu'c't} \leq Q^{C-MIN}_{u,c,t} y_{uct} \quad \forall u \in U^{PS} \bigcup U^{PC}, c \in C_u, t \in T \tag{13}$$

- Crude oil selection constraints:

$$Q^F_{ucst} + Q^{A-MIN}_{ucst} z_{ucst} \leq \sum_{u'c's} q_{ucsu'c't} \leq Q^F_{u,c,s,t} + Q^{A-MAX}_{ucst} z_{ucst}$$
$$\forall u \in U^S, c \in C_u, s \in S^O_{uc}, t \in T \tag{14}$$

The family of functions used for modeling the transformation processes (2)-(5) are defined by sums of linear and bilinear (or trilinear) terms.

3. A Small Example

Consider the hypothetical refinery shown in Figure 1. Few streams are produced in each processing unit, unlike the real cases in which lots of streams are produced and then mixed in intermediate tanks. Two crude oils are available and at least $300m^3$ of crude oil 2 might be processed by $400\$/m^3$. The refinery can still purchase up to $1500m^3$ of crude oil 1 by $375\$/m^3$, and $800m^3$ of crude oil 2 by $450\$/m^3$, but the minimum volumes are $300m^3$ and $200m^3$, respectively. There are no storage costs neither storage limits, but a restriction on the maximum volume of oil the refinery is able to receive in a certain period. Table 1 complements the data.

Table 1. Demands, prices, and specifications for final products.

Product	*Price ($)*	*Demand (t=1)*	*Demand (t=2)*	Specifications
LPG	312	-/250	-/200	Density: [0,0.57]
Gasoline	559	50/300	50/300	Sulfur: [0,0.143] Octane: [75,100]
Diesel	487	50/300	50/300	Sulfur: [0,0.57] Cetane Number [35,100]
Fuel oil	296	-/350	-/350	Sulfur: [01]

Formulation was coded in AIMMS 3.11 and solved by BARON 7.5.3 on a server powered by an Intel Core 2 Duo 2.26GHz, 8BG of RAM, under MS Windows Server 2003 64-bit. The generated mathematical program have 281 variables(12 binary) and 307 constraints. First solution was found in just $33.9s$ within an duality gap of 34.7%, and the best incumbent one in $6min$ approximately. But, after $2h$ there was a duality gap of almost 17.5%. This small case illustrates how hard is to solve RPISP to optmality. Even being about 10 times smaller than the

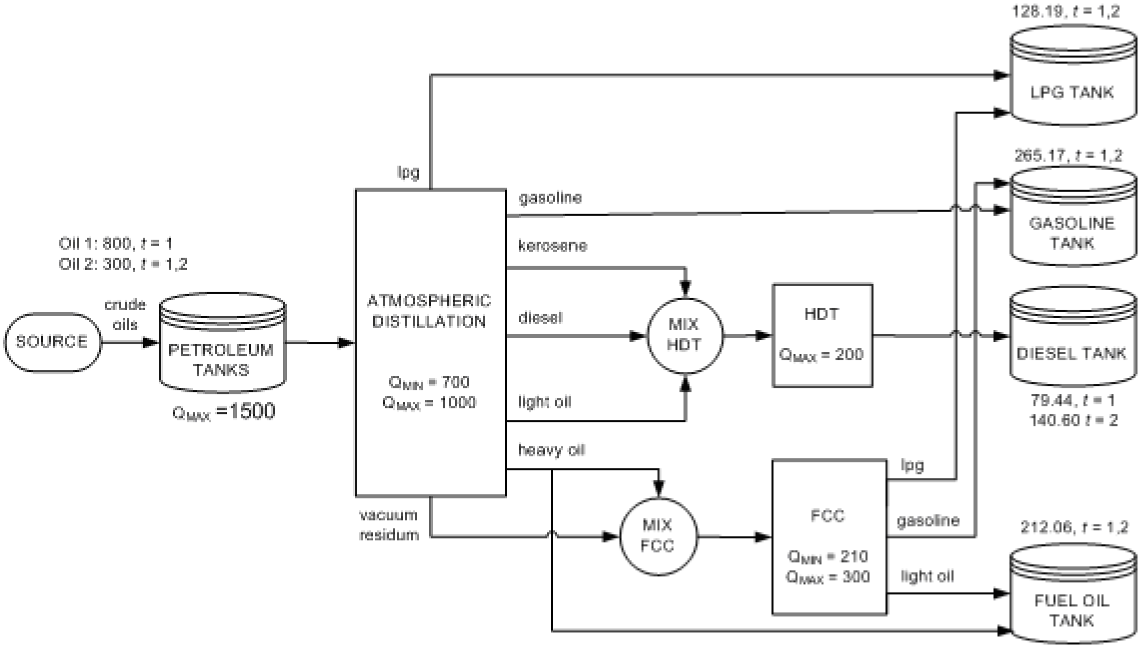


Figure 1. A hypothetical refinery specialized in the production of fuels.

real instances and having lot less nonlinear constraints (since there are few units, specially intermediate tanks), it requires a prohibitive computation time.

In both periods the best solution found suggests the allocation of campaigns 1 and 2 to UDA, and allocation of campaign 1 to FCC. Moreover, it suggests using extra volumes of crude oil 1 in period 1. Since the minimum demands are very low, this refining system is almost free for choosing the most profitable production plan.The solutions is also is indicated in Figure 1.

4. Summary

Find the global optimum of RPISP it is a very hard task. Altough BARON had been able to found sub-optimal solutions quickly for the hypothetical refinery, the same was not true in real cases that usually have dozens of binary variables and thousands of continous variables and constraints. BARON and AOA were also applied to a small set of real instances (not reported here), but failed in found a feasible solution in most cases. These instances have around 60 binary variables and 2000 continuous ones. As an alternative we have been working in matheuristics and bounding algorithms.

Acknowledgments

The authors wish to thank CNPq that supported this research under the grant 140174/2007-9.

References

[1] Neiro, S.M.S., Pinto, J.M. Multiperiod optimization for production planning of petroleum refineries, *Chemical Engineering Communications*, v. 192, p. 62-88, 2005.

[2] Coelho, O., Bahiense, L., Ferreira Filho, V.J.M. A mixed-integer nonlinear programming model for prodution planning in oil refineries. In *Rio Oil & Gas Proc.*, pages 1-10, Rio de Janeiro, 2010.

Metaheuristic Iterated Local Search and Variable Neighborhood Search for the Dial-a-Ride Problem *

Daniel L. V. Costa, Lucídio A. F. Cabral, and Clauirton A. Siebra

PPGI - Programa de Pós-Graduação em Informática, Universidade Federal da Paraíba, João Pessoa/PB, Brasil
dlvcpe@gmail.com, lucidio@di.ufpb.br, clauirton@di.ufpb.br

Abstract The Dial-a-Ride Problem have the objective attend all the requisitions (pickup and delivery locations) of a passengers group served by a fleet of vehicles. DARP aim to minimizes the operational costs and garantee the Quality of Service, given by constraints imposed by users and vehicles. In this work we presents a Iterated Local Search with Variable Neighbourhood Search metaheuristic to solve the DARP, a set of Nighborhood Structures and the Perturbation methods. We reported new best results for 9 out of 13 instances utilized as benchmark.

Keywords: Stochastic Methods, Metaheuristic, Dial-a-Ride Problem

1. Introduction

Dial-a-Ride Problem (DARP) is a particular instance of the Vehicle Routing Problem, whose main difference to other approaches is its aim in reaching an appropriate balance between the reduction of the passages dissatisfaction and operational costs [4].

DARP can be represented in different configurations. For example, it can consider dynamic or static requests, one or multiple vehicles, homogeneous or heterogeneous set of vehicles, and unique or diverse garages. Several works have explored such different configurations.

Jaw et al. [5] was one of the first works to propose the use of heuristics to the static modelling of DARP to multiple vehicles. In this approach, the authors used time windows for boarding and leaving, and a maximum time for trips was represented via a linear function of the trip time, which is imposed to each user. A heuristic selects the clients in order of the most viable boarding and gradually inserts such clients into the vehicles routes.

Toth and Vigo [10] also enables the specification of boarding and leaving requests with time windows. However, they use a limit related to the trip time, which is proportional to the distance of the trip. This work used a local search method, derived from the Tabu search, and heuristics for parallel intra-routes and inter-routes insertions.

Znamensky and Cunha [11] adapted the Parallel Insertion Heuristic, proposed by Madsern et al. [8] and applied improvements to the routes methods, similarly to Toth and Vigo [10]. This work was applied to the Transport Service for Deficient People of Sao Paulo city, via the ATENDE system. In order, this system uses vans to attend elderly and deficient people. Instances correspond to a day of operation, which has a total of 349 requests of clients using 84 vehicles distributed along 47 garages in several cities.

Cordeau and Laport [2] developed a multi-vehicle algorithm, to static DARP, applying three different Tabu search heuristics. The first heuristic aims to only reduce the violations in the time windows. The second heuristic uses this same idea and also tries to reduce the routes duration time. The last heuristic uses the ideas of the first and second approaches and also the time that users stay inside the vehicles.

*Acknowledge CAPES/REUNI the financial support given to the development of this work.

Jϕrgensen et al. [6] proposed the use of Genetic Algorithm (GA) to DARP. This model is a static version with multiple vehicles, heterogeneous set of vehicles and multiple garages. The approach to resolve the problem was to "first group" and "after routing". The GA arranges the clients to vehicles and determines which clients will be attended for which vehicles. While the routing stage determines the sequence that clients will be attended; the programming stage independently determines the times for each vehicle via a specific heuristic.

Courdeau [1] used a Branch-and-Cut algorithm to DARP. In this work, the problem is modelled as a static version, multiple vehicles, homogeneous set of vehicles and unique garage. The problem resolution is given by an exact approach. The used instances were randomly generated, with a maximum of 32 requests.

Mauri and Lorena [7] used the Simulated Anneling to DARP and proposed a multi-goal version, which aims to reduce both the operational costs and client inconveniences. In this approach, some constraint can be relaxed, so that they can compose the objective function.

Parragh et al. [9] proposed a VNS that aims to reduce the total routing cost, respecting the maximum limits of the routes duration, time windows and time limits related to rides.

The paper is organized as follows. Section 2 presents the Proposed Method for this work, Section 3 presents the Computational Results, and the Conclusion is shown in Section 4.

2. Proposed Method

This work uses the Iterated Local Search metaheuristic, together to the Variable Neighbourhood Search. As other works, the aim is to reduce the total routing cost, maintaining the client satisfaction. For that end, we have used the proposal of Mauri and Lorena [7], which considers the time of service starting, leaving and arriving times in the nodes, delay time of service, vehicle loading, trip time, routing time and time window in each node of the graph. This formulation enables that constraints can be relaxed For example, the maximum duration time of routes, maximum time of waiting in each local route, vehicles capacity and the total time that surpasses time windows.

The developed algorithm has a pre-processing to cut the graph, according to the method proposed by Cordeau [1]. After that, we use a heuristic to create an initial solution, which considers four insertion modes, as presented by Parragh [9]. The neighbourhood structures, applied in this work, were: RouteReordering[7], PointReallocation[7], PointsChange[7], Reverse, RouteShift(1), RouteShift(2), RouteShift(3), Swap(1,1), Swap(2,1), Shift(1,0) and Shift(2,0).

The RouteReordering[7] consists in a randomly choice of a route that is part of the current solution and the randomly choice of a new position to insert it, considering the precedence restriction (a leaving cannot be executed before its boarding).

The method PointReallocation[7] consists in a randomly selection of two routes and a request regarding the first route. This request is removed and randomly inserted into the second route.

The PointsChange[7] consists in a randomly choice of a pair of route and one requisition in each one, then swap the requisitions.

The Reverse neighbourhood structure enables the inversion of the route direction, so that the boarding and leaving points are changed.

The movements RouteShift(1)/RouteShift(2)/RouteShift(3) consist in selecting one of the routes and shifting its first element in one/two/three positions. If this process results in some problem regarding the precedence restriction, then we remove the problematic pair of the request and insert it in the best position according to the objective function.

Swap(1,1)/Swap(1,2) enables a randomly choice of two routes related to the current solution and permutes one/two requests from the first route for one from the second route. The insertion of the elements are carried out in accordance with the critic node, as proposed by Cordeau and Laporte [2].

Shift(1,0)/Shift(2,0) selects two any routes, remove one/two requests from the first route and insert them in the second route. Again, the insertion is carried out in accordance with the critical node.

The perturbation method for the ILS-VNS, we utilizes "Reallocate Blocks for Departure", Swap(m,n) and "Several Neighbourhoods" methods. The "Reallocate Blocks for Departure" selects a route, inverting the order of a continuous loading sequence. Swap(m,n) consists in selecting two routes, choosing m requests of the first route, n requests of the second route and permuting such choices. The insertion of the requests in the routes is carried out in accordance with the critical node. The "Several Neighbourhoods" method chooses a sequence of neighbourhood structures and sequentially applies such structures.

3. Computational Results

The ILS-VNS was implemented in C++ and all experiments were executed in a Core2Duo 2.26GHz processor with memory of 2GB. The instances used in our experiments were proposed by Cordeau and Laporte [3] and are available in `http://www.hec.ca/chairedistributique/data/darp/`.

As in other works [2],[6],[7],[9], we have sequentially executed 5 times each instance of the problem, and computed the best results for each instance. Next table(1) shows the results of our experiments, together with results of other approaches from the literature. The bold values represent the best values for each problem. We got nine new best results for the instances utilized as benchmark.

Table 1. Traveled Distance result

Instance	Tabu Search	Simulated Anneling	Genetic Algorithm	VNS	ILS-VNS
R1a	190, 02	252, 79	309	190, 02	**160.286**
R2a	302, 08	437, 45	539	301, 04	**281.045**
R3a	532, 08	831, 74	1047	532, 00	**490.491**
R5a	636, 97	1085, 45	1350	628, 11	**599.677**
R9a	672, 44	1064, 23	1343	658, 31	**654.001**
R10a	878, 76	1392, 09	1811	**857, 11**	865.417
R1b	164, 46	251, 85	284	164, 46	**155.357**
R2b	296, 06	436, 69	561	295, 66	**289.818**
R5b	589, 74	1010, 09	1344	**578, 61**	618.299
R6b	743, 60	1289, 31	1799	**740, 35**	832.962
R7b	248, 21	375, 67	478	248, 21	**239.158**
R9b	601, 96	1041, 09	1372	597, 75	**595.554**
R10b	798, 63	1414, 65	1740	**795, 16**	876.658

4. Conclusion

This paper discusses a new algorithm that uses two metaheuristics, Iterated Local Search and Variable Neighbourhood Search, to solve DARP with multiple vehicles and static requests. The experiments show a good performance when we compare this algorithm with other approaches from the literature. For future works, we intend to use exact local search methods to a better exploration of the neighbourhoods.

References

[1] Jean-François Cordeau. A Branch-and-Cut Algorithm for the Dial-a-Ride Problem. *Operations Research*, 54(3):573–586, May 2006.

[2] Jean-François Cordeau and Gilbert Laporte. A tabu search heuristic for the static multi-vehicle dial-a-ride problem. *Transportation Research Part B: Methodological*, 37(6):579–594, July 2003.

[3] Jean-François Cordeau and Gilbert Laporte. The Dial-a-Ride Problem (DARP): Variants, modeling issues and algorithms. *Quarterly Journal of the Belgian, French and Italian Operations Research Societies*, 1(2):89–101, June 2003.

[4] Jean-François Cordeau, Gilbert Laporte, and Stefan Ropke. Recent Models and Algorithms for One-to-One Pickup and Delivery Problems. *Distribution*, 2007.

[5] J.J. Jaw, A.R. Odoni, H.N. Psaraftis, and N.H.M. Wilson. A heuristic algorithm for the multi-vehicle advance request dial-a-ride problem with time windows. *Transportation Research Part B: Methodological*, 20(3):243–257, 1986.

[6] R M Jorgensen, K B Bergvinsdottir, and J Larsen. Solving the Dial-a-Ride problem using genetic algorithms. *Journal of the Operational Research Society*, 58(10):1321–1331, September 2006.

[7] Geraldo Regis Mauri and Luiz Antonio Nogueira Lorena. Uma nova abordagem para o problema dial-a-ride. *Produção*, 19(1):041–054, April 2009.

[8] Hans F. Ravn Oli B. G. Madsen and Jens Moberg Rygaard. A heuristic algorithm for a dial-a-ride problem with time windows, multiple capacities, and multiple objectives. *Annals of Operations Research*, 60(1):193 – 208, 1995.

[9] Sophie N. Parragh, Karl F. Doerner, and Richard F. Hartl. Variable neighborhood search for the dial-a-ride problem. *Computers & Operations Research*, 37(6):1129–1138, June 2009.

[10] Paolo Toth and Daniele Vigo. *Meta-Heuristics: Theory and Applications*, chapter 41, page 677. Kluwer Academic Publishers, Norwell, MA, USA, 1996.

[11] Andrei Znamensky and CB Cunha. Um modelo para o problema de roteirização e programação do transporte de deficientes. In *Congresso de Pesquisa em Transportes*, volume 13, pages 59–62, 1999.

A global derivative-free optimization method for expensive functions with bound constraints

Eugenio Echagüe[1], Frédéric Delbos[2] and Laurent Dumas [3]

[1]*IFPEN, 1 et 4 Avenue du Bois Préau, 92500 Rueil-Malmaison, France,* eugenio.echague@ifpen.fr

[2]*IFPEN, 1 et 4 Avenue du Bois Préau, 92500 Rueil-Malmaison, France,* frederic.delbos@ifpen.fr

[3]*Université de Versailles S.Q. , 45 Avenue des Etats-Unis, 78035 Versailles, France,* laurent.dumas@uvsq.fr

Abstract In this article we propose a new method in order to solve a general black-box global optimization problem where function evaluations are expensive. Our work was motivated by many problems in the oil industry, coming from several domains like reservoir engineering, molecular modeling, engine calibration and inverse problems in geosciences. Even if evolutionary algorithms are often a good tool to solve these problems, they sometimes need too many function evaluations, especially in high-dimension cases. To overcome this difficulty, we propose here a new approach, called GOSGrid, using as surrogate model the Sparse Grid interpolation method with a refinement process .

Keywords: Global optimization, Expensive functions, Surrogate models, Sparse Grid interpolation.

1. Introduction

In the context of oil industry, many problems consist in a global minimization of a computationally expensive function with bound constraints ([1]):

$$\begin{gathered}\text{Find } x^* = \text{argmin} f(x) \\ x_l \leq x \leq x_u \\ x \in \mathbb{R}^n\end{gathered}$$

where $f : \mathbb{R}^d \to \mathbb{R}$ is the computationally expensive function and x_l, $x_u \in \mathbb{R}^n$.

The values of f are in general the output of a complex simulator for which we don't have an explicit expression. The absence of any information on the function gradient narrows the resolution field to algorithms using no first or second order derivatives. There exists many different approaches in derivative free optimization, among which the most popular are direct search methods like Nelder Mead or MADS ([2]) and evolutionary algorithms like genetic algorithms ([3]), evolution strategies or particle swarm optimization (see [4] for a review of DFO methods). Unfortunately, all these approaches may exhibit a slow convergence behavior and thus be very expensive.

The use of a surrogate model is well suited for the type of optimization considered here. A surrogate model is a framework used to minimize a function by sequentially building and minimizing a simpler model (surrogate) of the original function. A widely used form of surrogate models consists of linear combinations of basis functions, for instance Radial Basis Functions ([5]) or Kriging. In general, the more points used when creating an interpolation model, the more accurate is the approximation.

In this work, we construct a new surrogate model by using the Sparse Grid interpolation method. Basically, the Sparse Grid approach is a hierarchical Lagrange approximation method which neglects the basis functions with the smallest supports. This approach was introduced in 1963 by Smolyak ([6]) in order to approximate integrals in high dimensions. It was applied for PDE approximations and more recently for sensitivity analysis ([7]) and optimization ([8]).

Compared to the approach in [8], a local refinement is constructed here in order to explore the more promising regions. The Sparse Grid interpolation method is recalled in section 2 whereas the new global optimization method is presented and applied for analytical test functions in section 3.

2. The Sparse Grid interpolation method

The Sparse Grid interpolation method uses Lagrange polynomials on the Chebyshev points as basis functions in dimension one. The extension to dimension d is done by simply tensoring the formulas obtained in dimension one. The hierarchical approach and the sparsity principle are respectively presented in the first two subsections. The refinement process is then described in subsection 2.3.

2.1 The 1-D case

For $i \in \mathbb{N}$, we call X^i the set of the Chebyshev points in the interval $[0, 1]$ of level i. These sets have the property that $X^i \subset X^{i+1}$. If we denote by a^i_j the Lagrange polynomial associated to each $x^i_j \in X^i$, the interpolation model of level i of f, called $m_i(f)$, is equal to

$$m_i(f) = \sum_{x^i_j \in X^i} f(x^i_j) a^i_j. \tag{1}$$

Define Δ^k as the difference between two consecutive models, then:

$$\Delta^k = m_k(f) - m_{k-1}(f) = \textstyle\sum_{x^k_j \in X^k} (f(x^k_j) - m_{k-1}(f)(x^k_j)) \,.\, a^k_j.$$

If we set $X^k_\Delta = X^k \backslash X^{k-1}$, as $X^{k-1} \subset X^k$ we get

$$\Delta^k = \sum_{x^k_j \in X^k_\Delta} \underbrace{(f(x^k_j) - m_{k-1}(f)(x^k_j))}_{w^k_j} \,.\, a^k_j.$$

It means that for computing Δ^k we only evaluate the function on the points that don't belong to the previous level sets. The telescopic sum principle and $m_0(f) = 0$ give us:

$$m_i(f) = \sum_{k=1}^{i} \Delta^k.$$

Thus, in order to get the approximation of level $i + 1$ we only need to compute the function values at the new interpolation points X^{i+1}_Δ.

2.2 The general case

For $k = 1, \dots, d$ let X^{i_k} be a set of Chebyshev points of some level i_k. By simply tensoring (1) we get the Lagrange interpolation formula on the set $\prod_{k=1}^d X^{i_k}$ as:

$$m_{(i_1,i_2,\dots i_d)}(f) = \sum_{x^{i_1}_{j_1} \in X^{i_1}} \cdots \sum_{x^{i_d}_{j_d} \in X^{i_d}} f(x^{i_1}_{j_1}, \dots, x^{i_d}_{j_d})(a^{i_1}_{j_1} \otimes \dots \otimes a^{i_d}_{j_d}).$$

With the same hierarchical approach done in dimension 1, we get

$$m_{(i_1,i_2,\dots i_d)}(f) = \sum_{k_1=1}^{i_1} \cdots \sum_{k_d=1}^{i_d} (\Delta^{k_1} \otimes \dots \otimes \Delta^{k_d}).$$

If we only apply the sum on the indexes $k = (k_1, \dots, k_n)$ such that $|k|_1 \leq d + N - 1$ we get the *Sparse Grid interpolation formula* of level N, which neglects the smallest support basis functions:

$$SG_N(f) = \sum_{|k|_1 \leq d+N} (\Delta^{k_1} \otimes \dots \otimes \Delta^{k_d}).$$

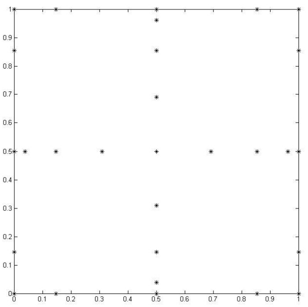


Figure 1. Sparse grid in 2-D for N=4. Number of points: 29

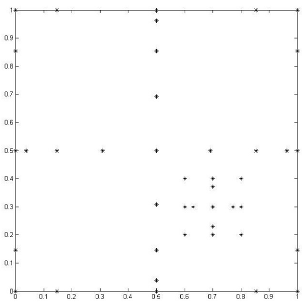


Figure 2. Sparse grid in 2-D for N=4 and a refinement process of order 3. Number of points: 42

The interpolation points of a Sparse Grid of level $N = 4$ in dimension 2 is depicted on Figure 1. Compared to a full grid of the same level which would contain 81 points, it is only made of 29 points. More generally, it can be proven that for a sufficently smooth function f, the approximation of f by a Sparse Grid model is of order $O(N^{-2}(\log N)^{d-1})$ with only $O(N(\log N)^{d-1})$ points.

2.3 The refinement process

In a global optimization problem, we need to explore the whole domain but we also need sometimes to focus our attention to special promising zones (the exploitation phase). To do so, a refinement process of the Sparse Grid interpolation model is used to construct a new and more precise model around a promising point. A local Sparse Grid model interpolates the error function in this area (see Figure 2) and can thus locally improve the current global model.

3. Global Optimization with a Sparse Grid model (GOSGrid)

Given a maximal number of function evaluations and bound constraints, the new optimization method, called GOSGrid, sequentially minimizes Sparse Grid models of the objective function f. In particular, during the process, it refines some zones in order to get a better solution. The hierarchical principle allows us to improve the global model without throwing away the previous one whereas the sparsity greatly reduces the number of exact evaluations, especially in high dimensions.

Starting from a given hierarchical level, the algorithm constructs a global model in the area described by the bound constraints. As the evaluation of the model is computationally inexpensive, a local multistart algorithm is run in order to find a global minimizer of the model. We then compare the function value at this point with the lowest value of the function at the grid and we keep the best of them. Then, we iteratively refine the global model in a hypercube centered at this point, and minimize with the same multistart algorithm the improved model. The construction of the next Sparse Grid level is done when a criteria based on the number of evaluations needed to perform the refinement or a relative rate of decrease is fulfilled.

Figure 3 gives a comparison between the GOSGrid method (continuous line) and a evolutionary algorithm, namely an evolution strategy with a cumulative step length adaptation, for the Michalewicz function in dimension $d = 5$.

The refinement process corresponds to the small decreasing branches starting from the continuous line (here for the levels $N = 3$ to $N = 5$). For a given number of evaluations, the cost function value for the evolutionary algorithm is higher than the corresponding one for the GOSGrid approach. The figure shows the importance of the refinement step as the best points are found after the refinement process. Other tests in higher dimension, which couldn't be included in the text due to space limitation, lead to the same conclusion.

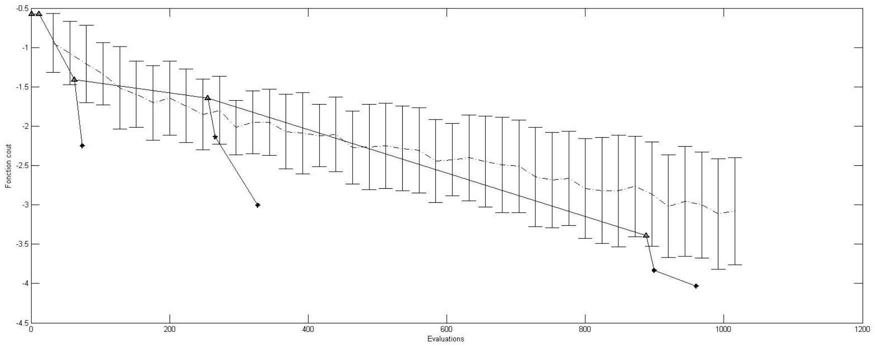


Figure 3. Comparison between GOSGrid (cont.) and an evolutionary algorithm (mean and standard deviation) for the Michalewicz function.

4. Conclusions and perspectives

We present here a new global optimization tool for expensive functions, called GOSGrid. It is based on the Sparse Grid interpolation method with a Sparse Grid refinement process. The hierarchical construction of the surrogate model and its sparsity allows to reach a faster decrease, in terms of cost function evaluations, compared to a classical evolutionary algorithm. The next step will consist to improve the refinement strategy by making it more adaptive and efficient (taking into account more promising points at the same time, for example) and to apply GOSGrid on real reservoir engineering cases.

References

[1] Oliver, Dean; Reynolds, Albert; Liu, Ning (2008). *Inverse theory for petroleum characterization and history matching* . Cambridge: Cambridge University Press.

[2] Audet, Charles; J.E. Dennis Jr (2006). *Mesh adaptive direct search algorithms for constraint optimization*. Philadelphia: SIAM J.Optim, Vol 17, pp. 188-217.

[3] Goldberg, Dave (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning*. Adison Wesley.

[4] Conn, Andrew; Scheinberg, Katya; Vicente, Luis (2009). *Introduction to derivative-free optimization*. Philadelphia: MPS-SIAM Series on Optimization.

[5] Dumas, Laurent; Herbert, Vincent; Muyl, Frédérique (2005). *Comparison of global optimization methods for drag reduction in the automotive industry*, Lecture Notes in Computer Science, Vol. 3483, pp. 948-957.

[6] Smolyak, S.A. (1963). *Quadrature and interpolation formulas for tensor products of certain classes of functions*, Soviet. Math. Dokl., Vol.4, pp. 240-243.

[7] Causse, Mathieu (2010). *Sparse Grid: de l'approximation des EDP au calcul de sensibilité*. Toulouse: Ph.D.

[8] Buzzard, G.; Donahue, M.; Rundell, A. (2009). *Parameter Identification with Adaptive Sparse Grid-based Optimization for Models of Cellular Processes*, Methods in Bioengineering: Systems Analysis of Biological Networks, A. Jayaraman and J. Hahn, Eds. Boston/London: Artech House, pp. 211-232.

A sequential linear approximation approach to the turning restriction design problem of traffic engineering

L. R. Foulds[1], D. C. S. Duarte[1], H. A. D. do Nascimento[1]*, H. Longo[1], B. R. Hall[2]

[1] *Instituto de Informática, UFG, Goiânia–GO, Brasil,* {lesfoulds,danielduarte,hadn,longo}@inf.ufg.br.

[2] *Instituto de Matemática e Estatística, UFG, Goiânia–GO, Brasil,* bryon@mat.ufg.br.

Abstract We present a sequential linear approximation method to heuristically solve the Turning Restriction Design Problem (TRDP). Experiments with a standard network example show favourable results compared to an existing nonlinear method.

Keywords: Urban traffic networks, Turning restriction design problem, Sequential linear approximation.

Recently, there has been significant interest in improving the performance of congested urban-traffic networks through global optimization. In general, a common objective of the *Network Design Problem* (*NDP*) [10] is to ease congestion and exhaust emissions by reducing the total travel cost of all users. The relatively inexpensive and easily implemented "non-intervention" family of improvement strategies for the NDP involves making non-physical adjustments to characteristics of the network such as flow direction, signal settings and tolls [2, 4–7].

Here we examine a particular low-cost, effective, non-intervention strategy, namely, that of deciding which turning restrictions at intersections should be imposed on the users of the network. That is, certain travel directions (left turn, right turn or drive straight ahead) may be prohibited at each particular intersection in order to prevent certain streams of users that arrive at the intersection from joining particular outgoing streams. The question is how to select which turns (if any) to restrict in order to enhance a given system performance measure. The advantages of introducing turning restrictions have been discussed in [3].

Long et al. [9] have introduced and defined the *Turning Restriction Design Problem* (*TRDP*) as that of determining the optimal set of turning restrictions to be imposed in order to minimize user equilibrium-based total cost, developing a bi-level model of the *TRDP*. A shortest path algorithm is used to establish user-equilibrium flows at the lower level and the resulting flows are used within a sensitivity algorithm to solve a relaxed version of the *TRDP* model. Finally, at the upper level, branch and bound strategies are used to solve the TRDP model to identify a promising set of turning restrictions at each iteration.

The method of Long et al. [9] begins with a network without any turning restrictions and progressively builds up a set of restrictions that are selected from a relatively limited subset of all possibilities, until some termination criterion is met. Only some so-called "crucial" intersections can be considered for restrictions and only left-turn restrictions are allowed. Also, the method is based on a highly nonlinear mixed integer programming model. For these reasons we have constructed a Sequential Linear Approximation (*SLA*) method that starts with a given set of turning restrictions, corresponding to the present situation in a given network, and aims to identify which additions or subtractions (allowing all practical possibilities) should be made in order to create the restriction regime that minimises user equilibrium-based total cost.

*Hugo do Nascimento is partially sponsored by a Scholarship of Research Productivity from CNPq (309463/2009-2).

The *TRDP* method we propose is based on a model to be solved using *SLA*. Let $G' = (N', A')$ be the underlying digraph that models a given traffic network with node set N' and arc set A'. Each node in N' represents either the beginning or the end of a street, that is, an entrance or departure point of an intersection, and each arc in A' represents either a street travel direction. Multiple arcs connecting the same pair of nodes are not permitted.

We model all turning possibilities at each intersection by replacing its representative node with a particular digraph due to Potts and Oliver [11]. This creates a modification of $G' = (N', A')$ that includes arcs representing all turning possibilities at the intersections. Arcs representing infeasible turns are permanently removed from A' and play no further part in the discussion. Next, we establish which further turning restrictions already exist in the given network, but that could possibly be rescinded, and denote the corresponding arc set by A_R. Finally, we define the set of remaining arcs that were created within the original nodes of G' as the class of turning restrictions that could possibly be introduced. We denote this arc set by A_I. Therefore, whenever flow is allowed in any arc $\beta \in A_R$, an existing turning restriction is rescinded. Whenever flow is prohibited in any arc $\beta \in A_I$, a new turning restriction is imposed. The original arcs A' representing the streets of the given network have remained untouched and none of them are subject to removal. We denote the final graph as $G = (N, A)$.

Let OD denote the set of origin-destination node pairs that make up the matrix $T = (t_{ij})$, where t_{ij} is the demand of private transportation from node i to node j. Let n_{ij} be the number of (i, j) routes and p_{ij}^k denote the k^{th} (i, j) route, for $k = 1, \dots, n_{ij}$. The route p_{ij}^k can be represented by a binary vector of elements $(p_{ij}^k)_\alpha$, corresponding to the arcs $\alpha \in A$, where $(p_{ij}^k)_\alpha$ is unity if α belongs to p_{ij}^k and is zero otherwise. Also, let x_{ij}^k be the number of users of p_{ij}^k and let the unit traversal cost of arc α, $\forall\, \alpha \in A$, be denoted by $t_\alpha(\cdot)$. The following popular arc cost function for any arc $\alpha \in A$ has been provided by the USA Bureau of Public Roads [1]:

$$t_\alpha(f_\alpha) = t_\alpha^F[1 + 0.15(f_\alpha/u_\alpha)^4], \tag{1}$$

where f_α is the flow, t_α^F is the congestion-free travel cost and u_α is the effective capacity. From (1) it can be seen that u_α has an inverse effect on t_α. This fact has lead us to adopt an arc capacity-based approach to the *TRDP*. Towards this end, for all arcs $\alpha \in A$ and $\beta \in A_R \cup A_I$, let s_α^β be a real number that denotes the change in u_α when the status of β is changed. That is, u_α becomes $u_\alpha + s_\alpha^\beta$ whenever either (i) $\beta \in A_R$ and flow is allowed in β (rescinding an existing turning restriction) or (ii) $\beta \in A_I$ and flow is prohibited in β (creating a new turning restriction). Note that s_α^β may be positive of negative. Furthermore, $\forall\, \alpha \in A_R \cup A_I$, let $E_\alpha^R = \{\beta \mid \beta \in A_R,\ s_\alpha^\beta \neq 0\}$, $E_\alpha^I = \{\beta \mid \beta \in A_I,\ s_\alpha^\beta \neq 0\}$, let c_α denote the cost of changing the status of arc α and let B denote the available budget for the total cost of all changes in arc status. Finally, the decision variables of the proposed model are: y_α, $\forall\, \alpha \in A_R \cup A_I$, where y_α is set to unity if $\alpha \in A_R \cup A_I$ and flow is allowed in α; otherwise y_α is set to zero. It is assumed that the flow pattern is in user equilibrium. That is, according to Wardrop's First Principle [12], all the routes actually used between any origin-destination pair of nodes should have close to equal travel costs and this cost must not exceed the cost of any unused route between this pair. To model this, let c_{ij}^k denote the unit cost of p_{ij}^k, $c_{ij}^* = \min\{c_{ij}^k \mid k = 1, \dots, n_{ij}\}$, $K_{ij} = \{k \mid k \in \{1, \dots, n_{ij}\}, c_{ij}^k = c_{ij}^*\}$ and $K'_{ij} = \{1, \dots, n_{ij}\} \backslash K_{ij}$.

We now introduce an *SLA* scheme where the arc flows f_α are held constant at each iteration in order to calculate the $t_\alpha(f_\alpha)$'s using (1). With these costs, we can identify least-cost routes for every $(i, j) \in OD$. These route costs are modified to provide the objective function coefficients of a mixed binary ILP that is used with the aim of identifying a promising regime of turning restrictions according to a user equilibrium flow assignment. Consider the following model, termed $(LIPSTUD)^r$, for the r^{th} *SLA* iteration that has as input the current flows f_α^r, $\forall\, \alpha \in A$, together with arc cost flows $t_\alpha(f_\alpha^r)$ found by substituting f_α^r in (1). This model is new and is one of the main contributions of the present paper.

$$\text{Minimise} \quad z = \sum\nolimits_{(i,j) \in OD} \sum\nolimits_{k=1}^{n_{ij}} (C_{ij}^k)^r . x_{ij}^k, \tag{2}$$

subject to

$$\sum\nolimits_{k=1}^{n_{ij}} x_{ij}^k = T_{ij}, \quad \forall\,(i,j) \in OD, \tag{3}$$

$$\sum\nolimits_{(i,j)\in OD} \sum\nolimits_{k=1}^{n_{ij}} (p_{ij}^k)_\alpha . x_{ij}^k \le u_\alpha + \sum\nolimits_{\beta\in E_\alpha^R} s_\alpha^\beta . y_\beta + \sum\nolimits_{\beta\in E_\alpha^I} s_\alpha^\beta .(1-y_\beta), \quad \forall\,\alpha \in A, \tag{4}$$

$$\sum\nolimits_{(i,j)\in OD} \sum\nolimits_{k=1}^{n_{ij}} (p_{ij}^k)_\alpha . x_{ij}^k \le M.y_\alpha, \quad \forall\,\alpha \in A_R \cup A_I, \tag{5}$$

$$\sum\nolimits_{\alpha\in A_R} c_\alpha . y_\alpha + \sum\nolimits_{\alpha\in A_I} c_\alpha .(1-y_\alpha) \le B, \tag{6}$$

$$x_{ij}^k \ge 0, \quad \forall\,(i,j) \in OD,\ k = 1,\ldots,n_{ij}, \tag{7}$$

$$y_\alpha \in \{0,1\}, \quad \forall\,\alpha \in A_R \cup A_I, \tag{8}$$

where $$(C_{ij}^k)^r = \sum\nolimits_{\alpha\in A} (p_{ij}^k)_\alpha . t_\alpha(f_\alpha^r), \quad \forall\, k \in (K_{ij})^r, \tag{9}$$

$$= M_1 . \sum\nolimits_{\alpha\in A} (p_{ij}^k)_\alpha . t_\alpha(f_\alpha^r), \quad \forall\, k \in (K_{ij}')^r, \tag{10}$$

and where $(K_{ij})^r$ and $(K_{ij}')^r$ are the versions of K_{ij} and K_{ij}' at the r^{th} iteration and M and M_1 are suitably chosen positive real numbers. The elements of the model are now explained. The function (2) is based on (9) and (10) and represents the objective of identifying a user equilibrium assignment. (3) is a conservation of flow constraint for all travel demand. (4) allows for arc capacity to be adjusted as a result of changes to the turning restriction regime. Next, (5) prevents travel in any arc that has a turning restriction. (6) introduces a budgetary constraint that enables control over the total cost of alterations that can be made to the original restriction regime. In the example discussed later the c_α's are set to unity and the budget B is set to various levels, controlling the total number of regime changes that can be made. B can be set to a relatively high number, allowing any possible combination of changes to be made. (8) and (9) are the usual non-negativity and binary conditions.

The *LIPSTUD* model is not designed to solve the nonlinear model of the *TRDP* obtained by substituting the nonlinear functions (1) into (9) and (10) and thus creating a nonlinear objective function in (2). Instead, *LIPSTUD* has been constructed to identify at each iteration a collection of O-D routes that can be substituted into (9) and (10) to compute a set of temporarily constant objective function coefficients. These coefficients are inserted into (2) to create a linear model. The optimal solution to this model is not necessarily optimal for the nonlinear model and thus *LIPSTUD*, like many *NDP* approaches, is an approximating, iterative, heuristic procedure. However, consider the special case where a user equilibrium assignment can be found that is a feasible solution to (3)–(8). Next, consider the problem (*LIPSTUD*)^r whose objective function coefficients have been calculated by substituting the arc flows of this assignment into (9) and (10). Then an optimal solution to (*LIPSTUD*)^r can be found that is a user equilibrium solution. In general, the procedure solves the linear problem (*LIPSTUD*)^r at the r^{th} iteration and then revises the arc flows f_α and the arc cost functions $t_\alpha(f_\alpha)$. These updates are used to revise the route costs to $(C_{ij}^k)^{r+1}$. From this, the next problem (*LIPSTUD*)$^{r+1}$ is formulated and solved. The process is repeated until some termination criterion is met.

We now discuss some preliminary computational experience with *LIPSTUD* on a standard network from the literature in comparison with the nonlinear method of Long et al. [9]. The numerical example is known as the Sioux Falls network and was first introduced in [8]. It has 24 nodes, 76 arcs, 528 O-D pairs and 178 possible turns at intersections (any of which can be restricted). The numerical results about to be reported are summarized in Table 1.

We first establish a user equilibrium assignment for the original network with no turning restrictions present, meaning that any user can make any turn. The total user cost of this assignment is z_o = 15,585,566.05 according to our calibrations and we use this cost as the basis for comparison. Long et al. [9] selected 22 "crucial" possible turnings restrictions about which to make decisions. Their method resulted in a turning regime with 15 of the 22 possible turns being restricted (T_1). For the same situation, *LIPSTUD* identified 12 restrictions, with an improved total user equilibrium cost (T_2). *LIPSTUD* was also run on the original network with a budget of $B = 22$ with each restriction introduction costing unity, that is, any combination of up to 22 restrictions can be imposed out of the possible 178. This resulted in a regime with the full limit of 22 restrictions imposed, with a slightly better total cost again (T_3). When B is increased to 178, allowing any combination of restrictions to be imposed, the best regime identified by *LIPSTUD* has 108 restrictions, leading to a significantly reduced total cost.

Table 1. The results of applying *LIPSTUD* to the Sioux Falls network.

Test	Method	Total UE Cost (z)	% Reduction in z_o	No. of Restrictions
T_1	Long et al.	15,087,557.74	3.20	15
T_2	*LIPSTUD*	14,794,305.21	5.08	12
T_3	*LIPSTUD*	14,679,976.38	5.81	22
T_4	*LIPSTUD*	14,338,358.57	8.00	108

We have presented a successive linear approximation method for identifying a heuristic solution to a nonlinear model of the *TRDP*. The method aims to adjust the current turning restriction regime in a given network in order to minimise the total cost when user route choice is driven by user equilibrium principles. The method has been compared with an existing nonlinear *TRDP* method using a standard network example from the literature. Preliminary computational experience with *LIPSTUD* compares favourably with that of the non-linear method. The authors are in the process of refining the *LIPSTUD* model, conducting further numerical experiments and investigating the convergence properties of the method.

References

[1] Bureau of Public Roads (BPR). Traffic assignment manual. Technical report, US Department of Commerce, Urban Planning Division, Washington DC, USA, 1964.

[2] G. E. Cantarella and A. Vitetta. The multi-criteria road network design problem in an urban area. *Transportation*, 33(6):567–588, 2006.

[3] K. Chen and Z. Z. Luo. Analysis of feature of left turn traffic at level crossing and countermeasures. *Technol. of Highway and Transport*, 22(2):114–118, 2006.

[4] E. Cipriani and G. Fusco. Combined signal setting design and traffic assignment problem. *Eur. J. Oper. Res.*, 155(3):569–583, 2004.

[5] L. R. Foulds. A multicommodity flow network design problem. *Transport. Res.*, 15B(4):273–284, 1981.

[6] M. Gallo, L. D'Acierno, and B. Montella. A meta-heuristic approach for solving the urban network design problem. *Eur. J. Oper. Res.*, 201(1):144–157, 2010.

[7] Z. Gao and Y. Song. A reserve capacity model of optimal signal control with user-equilibrium route choice. *Transport. Res.*, 36B(4):313–323, 2002.

[8] L. J. LeBlanc. An algorithm for the discrete network design problem. *Transport. Sci.*, 9(3):183–199, 1975.

[9] J. Long, Z. Gao, H. Zhang, and W. Y. Szeto. A turning restriction design problem in urban road networks. *Eur. J. Oper. Res.*, 206(3):569–578, 2010.

[10] T. L. Magnanti and R. T. Wong. Network design and transportation planning: models and algorithm. *Transport. Sci.*, 18(1):1–55, 1984.

[11] R. B. Potts and R. M. Oliver. *Flows in Transportation Networks.* Academic Press, New York, 1972.

[12] J. Wardrop. Some theoretical aspects of road traffic research. *Proceedings of the Institution of Civil Engineers, Part II*, 1(36):352–362, 1952.

Improved method to solve non convex quadratically constrained quadratic problems

Anthony Guillou[1], Pierre Hansen[2], and Sylvain Perron[3]

[1]*HEC Montréal,*
3000 Chemin de la Côte-Sainte-Catherine, Montréal (Québec) H3T 2A7, Canada anthony.guillou@gerad.ca

[2]*GERAD - HEC Montréal,*
3000 Chemin de la Côte-Sainte-Catherine, Montréal (Québec) H3T 2A7, Canada pierre.hansen@gerad.ca

[3]*GERAD - HEC Montréal,*
3000 Chemin de la Côte-Sainte-Catherine, Montréal (Québec) H3T 2A7, Canada sylvain.perron@gerad.ca

Abstract Non-convex quadratic programs with non-convex quadratic constraints can be solved by the exact method based on the branch and cut algorithm of Audet et al.[2]. The algorithm, however, may lead to solutions not necessarily strictly feasible. We present a new branching strategy improving the control on constraint feasibility.

Keywords: Global Optimization, Non convex quadratic programming, Branch and cut, Interval arithmetics

1. Introduction

We consider the general nonconvex quadratically constrained quadratic programming problem (QQP) that can be stated as follows:

$$\begin{aligned}
\min_{X\in\Omega} \quad & Q^0(X) = \sum_{(i,j)\in N^2} C^0_{ij}x_ix_j + \sum_{i\in N} c^0_i x_i^2 + \sum_{i\in N} d^0_i x_i \\
s.t. \quad & \\
& Q^p(X) = \sum_{(i,j)\in N^2} C^p_{ij}x_ix_j + \sum_{i\in N} c^p_i x_i^2 + \sum_{i\in N} d^p_i x_i = b^p \quad \forall p \in \{0\dots P\} \\
& Q^k(X) = \sum_{(i,j)\in N^2} C^k_{ij}x_ix_j + \sum_{i\in N} c^k_i x_i^2 + \sum_{i\in N} d^k_i x_i \le b^k \quad \forall k \in \{0\dots K\}
\end{aligned} \qquad (1)$$

Where $\Omega \subseteq \mathbb{R}^n$, X is a vector with coordinates x_i, and all the C_{ij}, c_i, d_i and b are real value parameters.

Such model encompasses many others problems including linear mixed $0-1$, fractional, bilinear, bilevel, generalized linear complementarity, and many more programming problems.

Different approaches have been developped to find the global optimum of such problems [1–3, 5–8, 11, 12]. However, Tuy and Hoai-Phuong[13] emphasize that numerical errors occur in such processes, which could lead to incorrect solutions.

Audet et al. [2] propose to solve the problem with an algorithm based on approximation of quadratic terms by means of Reformulation-Linearization Techniques (RLT). Our approach extends their method based on the use of a Branch & Cut enumeration tree.

2. New branching scheme

We propose a new hybrid method that combines Interval Branch & Bound Algorithm (IBBA) [9] and Reformulation and Linearization Techniques (RLT) [2]. The method also contains a new stopping condition for the RLT process that results in a better accuracy of the solution and avoids the unexpected behavior highlighted by Tuy and Hoai-Phuong[13].

2.1 Node preprocessing in the enumeration tree

Perron [10] proved that bounds tightening can lead to a significant improvement in terms of convergence for RLT. However, the preprocessing method described in [10] is time consuming and is not applied at every node of the enumeration tree.

We use the constraint propagation techniques based on interval arithmetics presented in [4, 9] to improve the bounds on the variables. It provides a very efficient way to compute good bounds in a reasonable amount of time. Moreover, it provides a lower bound on the optimal solution of problem (1) that can be used to prune the enumeration tree if a feasible solution has been encountered prior to the processing of the current node. The technique can also determine whether a node potentially contains a feasible solution for the initial problem. If it is not the case, no further processing is performed on the node.

The resolution of the linear relaxation also provides a lower bound to the solution of (1). Such value can be used to apply the constraint propagation to the objective function and improve the quality of the bound tightening.

2.2 Evaluation of the stopping condition

In the rest of the paper, we denote by V the vector of v_i and W the matrix of w_{ij}.

To solve the problem with the RLT method, the following problem, equivalent to (1), is used.

$$\min_{X,V,W} \quad [Q^0]_l(X,V,W) = \sum_{(i,j)\in N^2} C^0_{ij} w_{ij} + \sum_{i\in N} c^0_i v_i + \sum_{i\in N} d^0_i x_i$$

$$s.t.$$

$$[Q^p]_l(X,V,W) = \sum_{(i,j)\in N^2} C^p_{ij} w_{ij} + \sum_{i\in N} c^p_i v_i + \sum_{i\in N} d^p_i x_i = b^p \quad \forall p \in \{0 \dots P\} \tag{2}$$

$$[Q^k]_l(X,V,W) = \sum_{(i,j)\in N^2} C^k_{ij} w_{ij} + \sum_{i\in N} c^k_i v_i + \sum_{i\in N} d^k_i x_i \leq b^k \quad \forall k \in \{0 \dots K\} \tag{3}$$

$$v_i = x_i^2 \quad \forall i \in \{1 \dots n\} \tag{4}$$

$$w_{ij} = x_i x_j \quad \forall (i,j) \in \{1 \dots n\}^2 \tag{5}$$

$$v_i \geq 0, \tag{6}$$

A linear relaxation is obtained by relaxing the constraints (4) and (5).

Typically, branching methods stopping conditions are reached when the optimal solution $(\hat{X}, \hat{V}, \hat{W})$ of the linear relaxation does not violate the relaxed constraints (within a certain tolerance).

However, the relaxed problem described above, usually does not have a unique optimal solution and finding one of them that satisfies the constraints (4) and (5) can require the generation of many cuts and branches in the enumeration tree.

Our approach consists in inferring a point $(\dot{X}, \dot{V}, \dot{W})$ from $(\hat{X}, \hat{V}, \hat{W})$ that satisfies the relaxed constraints of the initial problem. The following rule is used to generate such a point:

$$\begin{aligned} \dot{x_i} &= \hat{x_i} && \forall i \in \{1 \dots n\} \\ \dot{v_i} &= \hat{x_i}^2 && \forall i \in \{1 \dots n\} \\ \dot{w_{ij}} &= \hat{x_i}\hat{x_j} && \forall (i,j) \in \{1 \dots n\}^2 \end{aligned}$$

A feasibility condition for the new point $(\dot{X}, \dot{V}, \dot{W})$ consists in verifying that constraints (2) and (3) are satisfied within a certain tolerance (to handle numerical precision errors). If such a condition is satisfied, then the evaluation of the objective function at the point $(\dot{X}, \dot{V}, \dot{W})$ gives an upper bound for the solution of (1).

Moreover, the point $(\dot{X}, \dot{V}, \dot{W})$ is said to verify the optimality condition, within a tolerance of ϵ_z, if it satisfies:

$$[Q^0]_l(\dot{X}, \dot{V}, \dot{W}) - [Q^0]_l(\hat{X}, \hat{V}, \hat{W}) \leq \epsilon_z \ . \tag{7}$$

As $[Q^0]_l(\hat{X}, \hat{V}, \hat{W})$ is a lower bound on the optimal solution of the initial problem, the value $[Q^0]_l(\dot{X}, \dot{V}, \dot{W}) - [Q^0]_l(\hat{X}, \hat{V}, \hat{W})$ represents the optimality gap of the point $(\dot{X}, \dot{V}, \dot{W})$. If it is sufficiently close to zero, one can consider that both $(\hat{X}, \hat{V}, \hat{W})$ and $(\dot{X}, \dot{V}, \dot{W})$ lie on the optimal face of the linear relaxation and $(\dot{X}, \dot{V}, \dot{W})$ is one representative of the set of optimal solutions that satisfies the contraints of (1).

2.3 Variable selection for branching

In [2] the selection of the variable for branching is based on the worst violation of (2) and (3) at point $(\hat{X}, \hat{V}, \hat{W})$.

However, given our new stopping condition, we propose a selection of the variables among those involved in constraints that $(\dot{X}, \dot{V}, \dot{W})$ does not satisfy. Hence, at every nodes in the subtree a better evaluation of one of the variable implied in a violated constraint is improved. This tends to decrease the infeasibility of the next incumbent solutions.

2.4 Value selection for branching

In [2], the authors propose to branch on a value α based on the minimization of a certain error on the approximation of v_i and w_{ij}. However, those evaluations imply a division by factors that are very close to 0 when the bounds on the variable x_i are tight. Hence, the more accurate the bounds on a variable are, the less stable is the evaluation of α.

In order to improve accuracy, more stable values for α are proposed. For example, using the value of x_i in the solution of the linear relaxation is shown to be as efficient on instances with few variables and limits the errors due to numerical instability.

3. Preliminary numerical results

A set of problems, taken from [10] is used to emphasize the performance of the new branching scheme in terms of CPU time and solution accuracy.

Table 1. Numerical results

	Perron [10]			New branching scheme		
Ex	*z*	*CPU(s)*	*max violation*	*z*	*CPU(s)*	*max violation*
ex_5_20	-400	0.01	$< 10^{-10}$	-400	0.02	$< 10^{-10}$
ex_5_21	10126.60	0.04	$< 10^{-10}$	10126.60	0.03	$< 10^{-10}$
ex_5_22	17.014	4.89	$5\ 10^{-6}$	17.014	1.06	$< 10^{-10}$
ex_5_23	-5450.75	1.69	$7\ 10^{-6}$	-5450.75	0.09	$< 10^{-10}$
ex_5_26	156.22	144.98	$1\ 10^{-5}$	156.22	1.24	$< 10^{-10}$

The new branching scheme leads to solution that do not violate the constraints (within a tolerance beyond the machine precision). Moreover, the propagation techniques based on interval arithmetics enables significant gains in terms of CPU time.

References

[1] F. Al-Khayyal, C. Larsen and T. Van Voorhis, A relaxation method for nonconvex quadratically constrained quadratic programs, Journal of Global Optimisation, Vol 6, No 3, pages 215–230 , 1995.

[2] C. Audet, P. Hansen, B. Jaumard and G. Savard, A branch and cut algorithm for non-convex quadratically constrained quadratic programming, Mathematical Programming, Vol 87, No 1, pages 131–152 , 2000.

[3] X. Bao, N. V. Sahinidis and M. Tawarmalani, Multiterm polyhedral relaxations for nonconvex, quadratically constrained quadratic programs, Optimization Methods and Software, Vol 24, No 4-5, pages 485-504 , 2009.

[4] L. H. de Figueiredo and J. Stolfi, Self-Validated Numerical Methods and Applications, , 1997.

[5] C. A. Floudas, Deterministic global optimization: theory, methods and applications, , No 37 , 1999.

[6] R. Horst and H. Tuy, Global Optimization, , 1996.

[7] J. B. Lasserre, Global optimization with polynomials and the problem of moments, SIAM Journal on Optimization, Vol 11, pages 796–817 , 2001.

[8] J. Linderoth, A simplicial branch-and-bound algorithm for solving quadratically constrained quadratic programs, Mathematical Programming, Vol 103, pages 251-282 , 2005.

[9] F. Messine, Deterministic global optimization using interval constraint propagation techniques, RAIRO, Oper. Res., Vol 38, No 4, pages 277–293 , 2004.

[10] S. Perron, Applications jointes de l'optimisation combinatoire et globale, Thèse de Doctorat, École Polytechnique de Montréal, 2004.

[11] H. D. Sherali and C. H. Tuncbilek, A reformulation-convexification approach for solving nonconvex quadratic programming problems, Journal of Global Optimisation, Vol 7, No 1, pages 1–31 , 1995.

[12] M. Tawarmalani and N. V. Sahinidis, A polyhedral branch-and-cut approach to global optimization, Math. Program, Vol 103, No 2, pages 225–249 , 2005.

[13] H. Tuy and N. T. Hoai-Phuong, A robust algorithm for quadratic optimization under quadratic constraints, Journal of Global Optimisation, Vol 37(4), pages 557-569 , 2007.

On obtaining global solutions of practical problems via Stochastic Dynamic Programming*

Eligius M.T. Hendrix,[1,2] Rene Haijema[2] and Inmaculada García,[1]

[1] *Computer Architecture, Málaga University, Spain* {eligius,igarciaf}@uma.es

[2] *Operations Research and Logistics, Wageningen University, The Netherlands* rene.haijema@wur.nl

Abstract Dynamic programming (DP) is a conceptual way to look at optimization problems that can be written in a step-wise separable form. Several textbooks exist that introduce us to theory and show it can solve nonlinear programming problems obtaining global optimal solutions. On the application side, no standard way nor software exists to convert optimization problems to DP problems. This makes application for practical engineering and economic problems a challenge. We use several cases to illustrate the process of modelling, implementing and solving practical problems.

Keywords: Dynamic Programming, application, Value function

1. Introduction

Dynamic Programming is the name for a specific framework and a class solution methods for solving optimization problems that can be decomposed into subproblems that are to be solved step-by-step, or in DP terms stage-by-stage. The concepts like stage, state, and Bellman equation can be obtained from classical books like [2], and [6].The ground idea is to split the optimization problem into stages $t \in \{1, 2, \dots, T\}$ and to define per stage a (vector of) state(s) X_t and decision variable(s) Q_t. A decision Q_t is to be found for every possible state in stage t that minimizes or maximizes some objective function.

Although the DP framework as sketched above is more general, we will focus in this paper on problems with a time component. The length of a stage can be arbitrary small and is not necessarily the same for all stages. Any uncertainty involved, e.g. in events that may happen during a stage, can be modeled through a random variable ξ_t, yielding Stochastic DP (SDP). The dynamics of a system from state X_t to X_{t+1} is given by transformation function $F_t(Q, X, \xi)$ and the contribution to the objective function in stage t is function $g_t(Q, X, \xi)$.

As well the state space as decision space can be discrete or continuous. The horizon T in the objective $\sum_0^T E[g_t(Q_t, X_{t-1}, \xi)]$ can be finite or infinite. In the first case, one wants to determine the optimal strategy $Q_t^*(X)$ that specifies what to do in step t at a state X. In a stationary system situation, an optimal strategy $Q^*(X)$ tells the decision maker what to do in which situation. In both cases, the optimum strategy should fulfil the so-called Bellman equation; there exists a so-called value function $V_t(X)$, such that the Bellman equation [2] applies

$$V_t(X) = \min_Q \left(E[g(X, Q, \xi_t)] + E[V_{t+1}(F(X, Q, \xi_t))] \right). \tag{1}$$

Usually Equation (1) is determined backwards from an end valuation V_T or by so-called value iteration in a stationary system. Each step requires the solution of optimization problem

*This work has been funded by grants from the Spanish Ministry of Science and Innovation (TIN2008-01117), Junta de Andalucía (P11-TIC-7176), in part financed by the European Regional Development Fund (ERDF). Eligius Hendrix is a fellow of the Spanish "Ramón y Cajal" contract program, co-financed by the European Social Fund.

(1) for each grid point of X. That problem may be a global optimization problem as illustrated in Figure 1. It shows the optimization with an increasing variance of random variable ξ_t.

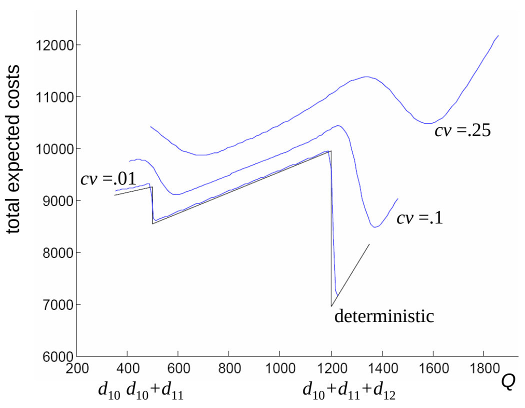


Figure 1. Value function inv. control, varying Q

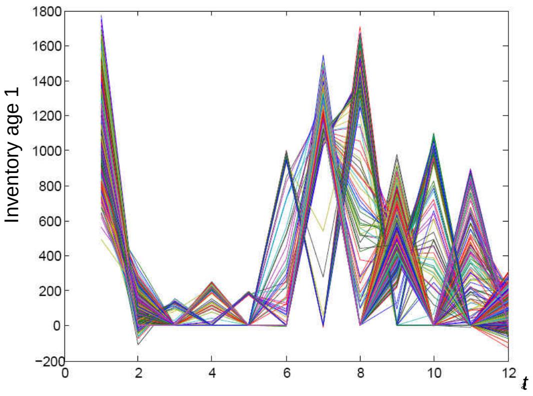


Figure 2. Simulated inventory level, SDP strategy

The theoretical result of optimality can be found in aforementioned handbooks. However, when studying this approach for a practical problem, one learns that function V and corresponding strategy Q^* can only be derived analytically for very specific cases, mostly in optimal control. Despite state space and decision space may be continuous, discretisation and truncation is commonly used for both spaces.

Moreover, the outcome space of the random variable ξ has to be dealt with and also interpolation is used to valuate the second term in (1). In the literature this technicality is either avoided by presenting a state-to-state transition matrix for each possible action, or, if used at all, it is not reported and certainly not well explained in the literature. The way to get the DP concept at work for practical problems enhances many questions. No standard software is available. Choices have to be made on the discretisation of states, truncating the state and decision space, on interpolation methods, on optimisation methods etc.

In the presentation, we show how strategies for many interesting problems can be derived: When and how much to order of perishable inventory products? [7, 5], When to release water in reservoir or lake management? [1], How should the European Union revise fish quota? [10], When to switch a traffic lights from colour? [4], Which actions to take consecutively in an agro-logistics supply chain? [3], Does price variation influence deforestation in Latin America? [8], and Will pollution of several industries in Australia lead to closure of some? [9]

In this paper, we share the experiences for the first three cases in Sections 2 to 4. We conclude in Section 5.

2. Inventory control for perishable products

In the management of inventory of fresh food one uses fixed expiry or use-by dates, after which the product cannot be sold anymore. Traditional inventory control balances inventory holding and ordering cost. Dealing with perishable products requires also keeping track of items of various ages to account for the waste due to outdating. In [7], a finite horizon stochastic programming model with a service level constraint has been described for a practical planning problem of a Dutch food producer. Chance constraints are known to lead to challenging (global) optimization problems. In that paper two static-dynamic approaches are evaluated: the static (discrete) order moments are determined as well as the continuous fixed order-up-to levels that dynamically set an order quantity.

Such strategy is not completely optimal, as it fixes the order moments whereas in an optimal strategy found by SDP this decision depends on the actual inventory levels. [5] provides a description of all challenges to approximate $Q_t^*(X)$: the dimension of the state space is the number of ages to keep track of, in principle the boundaries of the space depend on the demand functions, discretisation and interpolation has to be chosen to solve (1) and as shown in Figure 1, (1) is a global optimization problem that could be handled by a combination of grid search and local search. Notice that the number of optima is limited by the maximum age of the product. Figure 2 shows the resulting simulated inventory levels of the most fresh items when the optimum strategy is followed. One can notice that negative levels are reached in part of the samples due to the service level constraint.

3. Reservoir management

For the derivation of release operating tables in reservoir management, SDP can be used. We got involved in a study of an engineering office to design a two pump system for lake Amstelveense Poel. Their idea was to establish levels β_j under which one or two pumps are started to bring diphosphated water into the lake, see Figure 3. The idea was to determine the levels by black-box global optimization using simulated and real weather data over the past 25 years. However, this would imply a kind of stationary strategy $Q(X)$, whereas it is known that rainfall varies over the year and actually a non-stationary strategy $Q_t^*(X)$ is required. In our study based on weather data, periods of 10 days (decades) were chosen and an operating table derived by SDP in [1]. Notice that one challenge is that we are not dealing with a finite horizon concept; the valuation of the water level in the lake at the end of the year is starting point in following (1) for the 36 decades of which the outcome has to be repeated again. This process is called value iteration. The discretisation of the water level is easy and the optimization can be done in a simple spreadsheet.

Figure 3. Varying water level during the year in lake de Amstelveense Poel

4. Determination of fish quota

A team of economists was asked to evaluate what is the consequence for the fishery sector if the EU is not revising their quota on a yearly basis with all costs involved to do so. In our co-operation, we acknowledged that in fact we are dealing with a bi-level problem that potentially can be hard to solve, also called two-stage game in economics. On the first level, the EU sets quota and on the second level, fishermen react on that by deciding on harvest intensity and investment in their equipment. The resulting dynamic model has stochastic fish stock and capital (fishery equipment) as state variables and the quota is a continuous variable. The optimal value can be found by using nonlinear optimization for fish levels that are high enough to make harvesting profitable. The whole process is extremely sensitive for deriving the right bounds and discretisation of state and random outcome space. Moreover,

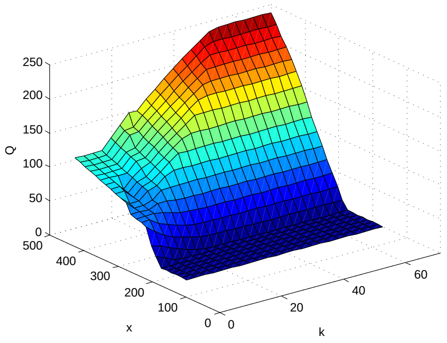


Figure 4. Quotum found by value iteration

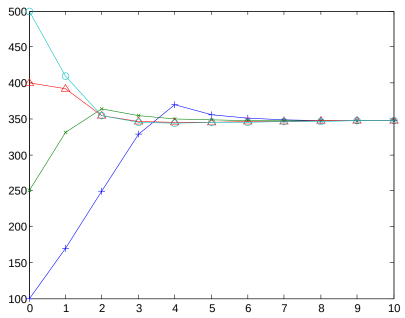


Figure 5. Stock convergence to steady state, det

we require value iteration repeating (1) many times to find the optimal stationary rule $Q^*(x,k)$ as depicted in Figure 4. The fish stock dynamics seem to be stable leading to convergence to a steady state with small random effects as sketched in Figure 5.

5. Summary

Stochastic Dynamic Programming is a concept to obtain global optimal strategies for many practical planning problems. The choice of state space and its bounding and discretisation is a big challenge in a tailor made environment, as there is no general purpose optimization tool.

References

[1] M. Bardoel. Ontwerp van een pompsysteem voor de Amstelveense Poel. Master's thesis, Wageningen University, Wageningen, 1995.

[2] R. Bellman. *Dynamic Programming*. Princeton University Press, 1957.

[3] J.K. Gigler, E.M.T. Hendrix, and G. Meerdink. On optimisation of agri chains by dynamic programming. *European Journal of Operational Research*, 139:613–625, 2002.

[4] R. Haijema and J. van der Wal. An MDP decomposition approach for traffic control at isolated signalized intersections. *Probability in the Engineering and Informational Sciences*, 22(4):587–602, 2008.

[5] E.M.T. Hendrix, R. Haijema, R. Rossi, and K. G. J. Pauls-Worm. On solving a stochastic programming model for perishable inventory control. In Ana Maria Rocha, editor, *Proceedings of Optimization Techniques and Applications 2012*, Salvador de Bahia, Brasil, 2012.

[6] R. A. Howard. *Dynamic Programming and Markov Processes*. MIT Press, Cambridge University, 1960.

[7] K. G. J. Pauls-Worm, R. Rossi, R. Haijema, E. M. T. Hendrix, and J. G. A. J. van der Vorst. Non-stationary inventory control for a perishable product. *OR Spektrum*, 2012.

[8] P. Roebeling and E.M.T. Hendrix. Land speculation and interest rate subsidies as a cause of deforestation: The role of cattle ranching in Costa Rica. *Land Use Policy*, 27:489–496, 2010.

[9] P. Roebeling, E.M.T. Hendrix, and M. van Grieken. Exploring industry specific social welfare maximizing rates of water pollution abatement in linked terrestrial and marine ecosystems. *Journal of Coastal Research*, 2:1681–1685, 2009.

[10] Diana van Dijk, Eligius M. T. Hendrix, Rene Haijema, Rolf Groeneveld, and Ekko van Ierland. On solving a bio-economic sdp model for analysing fishery policies. *Journal of Economic Dynamics and Control*, 2012.

On the Solution of Linear Programming Problems with Linear Complementarity Constraints

Joaquim Judice[1] and Isabel Ribeiro[2]

[1] *Instituto de Telecomunicações, Portugal,* Joaquim.Judice@co.it.pt

[2] *Universidade do Porto, Portugal,* iribeiro@fe.up.pt

A Linear Program with Linear Complementarity Constraints (LPLCC) can be stated as follows

$$\begin{array}{lll} Minimize & c^T x + d^T y & (1) \\ subject\ to & Ew = q + Mx + Ny & \\ & x \geq 0, w \geq 0 & (2) \\ & x^T w = 0 & \\ & Ay = b,\ y \geq 0 & \end{array}$$

where $E \in R^{m\times n}$, $M \in R^{n\times n}$, $N \in R^{m\times p}$, $A \in R^{t\times p}$, with $rank(A) = t < p$, $c \in R^n$, $d \in R^p$, $q \in R^m$ and $b \in R^t$ are given. Note that $x^T w = 0$ and $x \geq 0$ and $w \geq 0$ implies $x_i = 0$ or $w_i = 0$ for each $i = 1, \cdots, n$. Such variables x_i and w_i are named complementary.
The constraint set of an LPLCC is known as a General Linear Complementarity Problem (GLCP) and can even take a more general form allowing some x_i variables to be unrestricted in sign and their complementary w_i to be zero [1].
The LPLCC has found many applications in science, engineering, economics and finance [2]. The problem has also been shown useful as a tool for computing global minima of some important nonconvex programming problems, such as bilevel, bilinear, quadratic and absolute value programs [1]. Furthermore the well-known Linear Complementarity Problem (LCP) and the estimation of condition number of a nonsigular matrix in the l_1-norm can also be formulated as LPLCCs.
A (feasible) solution for the constraint set of an LPLCC, that is, a solution of a GLCP (2), can be found by solving the quadratic program

$$\begin{array}{lll} Minimize & x^T w & (3) \\ subject\ to & \text{linear constraints of GLCP (2)} & \end{array}$$

Then one of the following cases should occur:
(i) QP is infeasible (linear constraints are inconsistent) and GLCP is infeasible and has no solution.
(ii) QP is feasible and has a global minimum with a positive function value and GLCP is feasible and not solvable.
(iii) QP is feasible and has a global minimum with a zero function value and GLCP is solvable.
If $E = I_n$ and M is a PSD matrix, then case (ii) cannot occur, that is, GLCP is either infeasible or solvable. Furthermore if M is also an S-matrix (i.e, there exists $x \geq 0$ such that $Mx > 0$), then GLCP is solvable for each vector q and $\Omega = \{y \in R^p : Ay = b,\ y \geq 0\} \neq \emptyset$. This special type of LPLCC occurs quite often in applications namely on the solution of bilevel and bilinear programs by exploiting their LPLCC formulations [1].
In this talk we discuss the problem of finding a global minimum of an LPLCC when $E = I_n$

and $M \in PSD$. In this case, any stationary point of QP (3) is a solution of the GLCP (2). We show that if $M \in S$ and $c = 0$, then a global minimum for LPLCC can be found by solving a linear program and an LCP with $M \in PSD \cap S$, which should be considered as easy tasks. For a general PSD matrix M and $c \neq 0$, LPLCC reduces to the Parametric GLCP (PGLCP), which consists of finding the smallest value of λ such that

$$GLCP, \quad c^T x + d^T y \leq \lambda \tag{4}$$

is solvable. A possible way of solving such a PGLCP consists of reducing the parameter λ according to some rule and finding the corresponding solutions of GLCP(λ) (4). This is the idea of the so-called Sequential Linear Complementarity Problem (SLCP) algorithm [1]. Unfortunately, each GLCP(λ) is NP-hard and, in general, only an enumerative method can solve it [1]. Alternatively, GLCP(λ) can be stated as the following Linear Integer Programming (LIP)

$$\begin{array}{rl} Maximize & (\lambda + \theta)\alpha - c^T u - c^T v \\ subject\ to & \gamma = Mu + Nv + q\alpha \\ & 0 = Av - b\alpha \\ & c^T u + d^T v \leq \lambda\alpha \\ & \gamma \geq 0,\ u \geq 0,\ v \geq 0,\ 0 \leq \alpha \leq 1 \\ & \gamma \leq z \\ & u \leq e - z \\ & z \in \{0,1\}^n \end{array} \tag{5}$$

where $\theta \geq 1$ is a fixed real number.
Consider the Linear Program (LP):

$$\begin{array}{rl} Minimize & d^T y \\ subject\ to & Ay = b, Ny = -q \\ & y \geq 0 \end{array} \tag{6}$$

Then either LP (6) is unbounded and LPLCC is unbounded or LP (6) is infeasible or it has an optimal solution $\bar{y}$. Let

$$\bar{\lambda} = \begin{cases} d^T \bar{y} & \text{if LP is feasible} \\ +\infty & \text{if LP is infeasible} \end{cases}$$

Then we show that for $\lambda < \bar{\lambda}$, $(\bar{\gamma}, \bar{u}, \bar{v}, \bar{\alpha})$ is a mixed-integer feasible solution of LIP (5) with $\bar{\alpha} > 0$ if and only if ($\bar{w} = \frac{\bar{\gamma}}{\bar{\alpha}}, \bar{x} = \frac{\bar{u}}{\bar{\alpha}}, \bar{y} = \frac{\bar{v}}{\bar{\alpha}}$) is a solution of GLCP($\lambda$) (4). Hence this GLCP($\lambda$) has no solution if and only if the optimal value of LIP (5) is equal to zero.
A Strongly Stationary Point (SSP) $(\bar{x}, \bar{y}, \bar{w})$ of LPLCC is a solution of GLCP (2) such that there exists $(\alpha, \lambda, \beta, \theta, \gamma)$ satisfying the following conditions

$$\begin{array}{l} 0 = E^T \lambda + \alpha \\ c = -M^T \lambda + \beta \\ d = -N^T \lambda + A^T \theta + \gamma \\ \gamma \geq 0, \gamma^T \bar{y} = 0 \\ \beta_i \bar{x}_i = \alpha_i \bar{w} = 0,\ i = 1, \cdots, n \\ \beta_i \geq 0,\ \alpha_i \geq 0 \text{ for all } i \in I_x \cap I_w \end{array}$$

where $I_x = \{i : \bar{x}_i = 0\}$ and $I_w = \{i : \bar{w}_i = 0\}$. It is possible to show that any global minimum $(\bar{x}, \bar{y}, \bar{w})$ of LPLCC is an SSP if strictly complementarity holds, i.e., $\bar{x}_i + \bar{w}_i > 0$ for all $i = 1, \cdots, n$. Furthermore an SSP can be found by a Complementarity Active-Set (CASET) algorithm, which maintains complementarity (solutions of GLCP (2)) in each iteration. Furthermore, an algorithm that finds strongly stationary points in a systematic way and in a finite

number of iterations has been designed to compute a strongly stationary point (SSP) which is a global minimum for the LPLCC or close to one.
The incorporation of CASET, the finite SSP algorithm and the 0-1 integer program (5) in the sequential algorithm is discussed. Some comments about the benefits of using these techniques in practice and some topics for future research are presented in the last part of the talk.

Keywords: Global Optimization, Complementarity Problems, Nonlinear Programming.

References

[1] J Júdice. Algorithms for linear programming with linear complementarity constraints. *Top - Journal of the Spanish Society of Statistics and Operations Research*, 20:4–25, 2012.

[2] J. S. Pang Z. Q. Luo and D. Ralph. *Mathematical Programs with Equilibrium Constraints*. Cambridge University Press, 1996.

MILP Formulation for the Software Clustering Problem

Viviane Köhler,[1] Marcia Fampa,[1] Olinto Araújo,[2]

[1] *Universidade Federal do Rio de Janeiro, COPPE, C.P. 68530, 21945-970, Rio de Janeiro, Brazil,* vivianeck@gmail.com, fampa@cos.ufrj.br

[2] *Universidade Federal de Santa Maria, CTISM, Santa Maria, Rio Grande do Sul, Brazil,* olinto@ctism.ufsm.br

Abstract We present a Mixed Integer Linear Programming (MILP) formulation for the Software Clustering Problem (SCP), where we divide the modules of a software system into groups or clusters, to facilitate tha work of the software maintainers. We discuss a preprocessing that reduces the size of the instances of the SCP and introduce some valid inequalities that have been shown to be very effective in tightening the MILP formulation. Numerical results presented compare the results obtained with the formulation proposed with the solutions obtained by the exhaustive algorithm supported by the freely available Bunch clustering tool, for benchmark problems.

Keywords: MILP formulation, module dependency graph, software clustering problem

1. Introduction

The clustering problem has an important application in Software Engineering, which usually deals with large software systems with complex structures. A procedure used to facilitate the work of the software maintainers, is to group classes of the software, to help them make the correct identification of the snippets of code that need alteration. To separate the software components into groups, the Software Engineering has created a representation of the software system as a directed graph where the modules are represented by nodes and the relationships between the modules are represented by weighted directed edges that connect the nodes. This graph is referred in the literature as the Modular Dependent Graph (MDG).

The problem of finding a good partition of an MDG is called the Software Clustering Problem (SCP). The SCP consists in clustering the nodes of the MDG in such a way that the groups or clusters formed contain highly-interdependent modules and the independent modules are placed in different groups. The measure that has been used recently in the literature to analyze the quality of the partition is called Turbo Modularization Quality (TurboMQ), which is the objective function of the SCP [1, 3, 4].

Formally, the TurboMQ measurement for an MDG partitioned into K clusters is calculated by summing the Cluster Factor (CF_k) for each cluster k of the partitioned MDG, as it follows:

$$\text{TurboMQ } = \sum_{k=1:k\neq\emptyset}^{K} CF_k, \text{ where } CF_k = \frac{\mu_k}{\mu_k + \frac{1}{2}\epsilon_k}. \tag{1}$$

In (1) μ_k is the sum of the weights of intra-edges, i.e., edges for which the source and target nodes lie inside cluster k, and ϵ_k is the sum of the weighs of inter-edges, i.e., all edges that originate or terminate in cluster k.

The SCP considered in this paper consists in finding the partition of a given MDG that maximizes the objective function TurboMQ.

2. MILP formulation for the SCP

Consider as input for the SCP, a given MDG=$\bar{G}(\bar{V},\bar{E})$ and the edge weight $\bar{c}_{uv}$, for each $(u,v) \in \bar{E}$. Now, define the undirected graph $G = (V, E)$ such that $V = \bar{V}$ and $E = \{(u,v) \in \bar{E} | u < v)$ and let $c_{uv} = \bar{c}_{uv} + \bar{c}_{vu}$, for each $(u,v) \in E$ (if $(v,u) \notin \bar{E}$, consider $\bar{c}_{vu} = 0$.) Let $\bar{K} = |V|$ be the maximum number of clusters in the optimal solution of the SCP and $K = \{1, \ldots, \bar{K}\}$.

Define for $u \in V$, $k \in K$, the variable x_{uk}, which is equal to 1 if node u is assigned to the cluster k, and 0 otherwise. Also define for $(u,v) \in E$, $k \in K$, the variable x_{uvk}, which is equal to 1, if nodes u and v are assigned to the cluster k, and 0 otherwise.

The Cluster Factor of cluster k (CF_k), defined in (1), can then be represented by the variable r_k, defined for all $k \in K$, as

$$r_k = \frac{2\sum_{(u,v)\in E} c_{uv} x_{uvk}}{\sum_{(u,v)\in E} c_{uv}(x_{uk} + x_{vk})},$$

if $k \neq \emptyset$, and 0, otherwise, and the objective function TurboMQ can be expressed as $\sum_{k\in K} r_k$. Note that the definition of r_k can be modeled by the constraints

$$r_k \leq \sum_{u\in V} x_{uk}, \quad \text{and} \quad r_k \left(\sum_{(u,v)\in E} c_{uv}(x_{uk} + x_{vk}) \right) = 2 \sum_{(u,v)\in E} c_{uv} x_{uvk}, \text{ for } k \in K.$$

Considering $s_{uk} = r_k x_{uk}$, we finally formulate the SCP as the following MILP problem:

$$(MILP) \text{ Maximize } \sum_{k\in K} r_k \tag{2}$$

subject to

$$\sum_{k\in K} x_{uk} = 1, \qquad \forall u \in V \tag{3}$$
$$r_k \leq \sum_{u\in V} x_{uk}, \qquad \forall k \in K \tag{4}$$
$$\sum_{(u,v)\in E} c_{uv}(s_{uk} + s_{vk}) = 2 \sum_{(u,v)\in E} c_{uv} x_{uvk}, \qquad \forall k \in K \tag{5}$$
$$x_{uvk} \leq x_{uk}, \qquad \forall (u,v) \in E, \forall k \in K \tag{6}$$
$$x_{uvk} \leq x_{vk}, \qquad \forall (u,v) \in E, \forall k \in K \tag{7}$$
$$x_{uvk} \geq x_{uk} + x_{vk} - 1, \qquad \forall (u,v) \in E, \forall k \in K \tag{8}$$
$$s_{uk} \leq r_k, \qquad \forall u \in V, \forall k \in K \tag{9}$$
$$s_{uk} \leq x_{uk}, \qquad \forall u \in V, \forall k \in K \tag{10}$$
$$s_{uk} \geq r_k + x_{uk} - 1, \qquad \forall u \in V, \forall k \in K \tag{11}$$
$$0 \leq r_k \leq 1, \qquad \forall k \in K \tag{12}$$
$$0 \leq s_{uk} \leq 1, \qquad \forall u \in V, \forall k \in K \tag{13}$$
$$0 \leq x_{uvk} \leq 1, \qquad \forall (u,v) \in E, \forall k \in K \tag{14}$$
$$x_{uk} \in \{0,1\}, \qquad \forall u \in V, \forall k \in K \tag{15}$$

Constraint (3) assures that each node is assigned to exactly one cluster. Constraints (4-5) define r_k. Constraints (6-8) imply the definition of x_{uvk}. Constraints (9-11) assure the identity between s_{uk} and the product $r_k x_{uk}$.Constraints (12-15), together with the other constraints in the model, assure that $r_k \in [0,1]$ and all other variables assume only values 0 and 1.

3. Preprocessing and Valid Inequalities

The following theorem supports a preprocessing for the SCP, which reduces the size of the MDG.

Theorem 1. *Let $G = (V, E)$ be the undirected weighted graph given as input for the SCP, as discussed in Section 2. Let $u \in V$ be a node with degree equal to one and $v \in V$ be adjacent to u. Then in the optimal solution of the SCP, u and v are assigned to the same cluster.*

We now discuss valid inequalities that were added to $MILP$ to tight its relaxation. The first set of valid inequalities are the symmetry cuts that force each node to be assigned to a cluster whose index is not greater than the index of the node, which is formulated as $\sum_{k=1}^{u} x_{uk} = 1$, for $u \in V$.

In the second set of symetry cuts, the inequalities force the lowest indexed node in each cluster to be equal to the cluster index, which are formulated as $x_{uk} \leq x_{kk}$, for $u \in \{3, \ldots, |V|\}$, $k \in \{2, \ldots, u-1\}$.

Our next set of valid inequalities places an upper bound for the variable r_k in formulation $MILP$. The upper bound for r_k can be obtained by solving a problem P_k, where we search for a partition of the MDG into at most two clusters (indexed by 1 and k) that maximizes the value of r_k.

Theorem 2. *Let the optimal solution value of P_k be represented as $z^*(P_k)$. Then*

$$r_k \leq z^*(P_k), \ \forall k \in K,$$

are valid inequalities for formulation $MILP_2$.

4. Numerical results

We present computational results for the MILP formulation proposed for the SCP and compare them with the results of the exhaustive search procedure supported by the Bunch clusterization tool. The exhaustive procedure of Bunch explicitly enumerates all possible partitions of the MDG and was the only algorithm that we found in the literature to obtain the optimal solution of the SCP. Our code was implemented in C++ and all runs were conducted on a 24GB Ram, 2.67GHz Intel Xeon processor running under Linux. The solver CPLEX, v12.2 [2], was used to solve the MILP problems. In all tests we limited in 11000 seconds the CPU time to solve the problems. The set of 15 test problems considered on the computational experiments is a subset of the instances from [5]. In the first five columns of Table 1 we present, for each instance, the name of the instance (Inst), and the number of nodes ($n = |V|$) and number of edges ($m = |E|$) on the MDG, before and after the transformation of the original directed MDG (MDG1) into the undirected and preprocessed MDG (MDG2). In the other columns of the table we present results that show the performance of formulation $MILP^+$ and Bunch. $MILP^+$ was obtained by the addition of some symmetry cuts and valid inequalities to $MILP$. The statistics presented in the table are the objective function value at the solution obtained (z), the duality gap (Gap (%)) and the CPU seconds (Time) for solving the problems. When the gap is greater than zero or when Bunch doesn't converge to the optimal solution, we report the best solution found. In this case the symbol "-" represents that the CPU time is the limited 11000 seconds and the symbol "*" represents that memory overflow has interrupted the running before the time limit.

5. Summary

In this work, we present a new MILP formulation for the Software Clustering Problem (SCP). We also discuss a preprocessing that reduces the size of the instances of the problem and introduce some valid inequalities that have been shown to be very effective in tightening the MILP formulation. Using the solver CPLEX, we obtain the optimal solution of instances from the literature with up to 26 modules in the Module Dependency Graph (MDG). This is the first time to our knowledge that mathematical programming is applied to the SCP and that optimal solutions are reported for instances with more than 15 modules.

Table 1. Numerical Results

	MDG1		MDG2		$MILP^+$			Exhaustive	
Inst	n	m	n	m	z	Gap	Time	z	Time
small	6	5	3	5	1.8333	0.0	0.01	1.8333	0.17
compiler	13	32	13	32	1.5065	0.0	1.00	1.5065	14.00
lab4	15	18	10	14	3.4000	0.0	0.30	3.4000	556.84
nos	16	52	15	50	1.6775	0.0	4.01	1.6775	4029.02
lslayout	17	43	17	43	1.8613	0.0	1.15	1.7417	-
boxer	18	29	12	29	3.1011	0.0	0.25	2.7083	-
mtunis	20	57	20	57	2.3145	0.0	32.93	2.0821	-
SPDB	21	33	7	8	5.5897	0.0	0.02	3.4566	-
bunch	23	62	15	45	2.4060	0.0	1.28	1.8867	-
ispell	24	103	23	97	2.3639	0.0	352.36	1.3964	-
ciald	26	64	22	62	2.8513	0.0	12.56	1.8111	-
Modulizer	26	66	18	57	2.7579	0.0	5.68	1.5708	-
rcs	29	163	28	155	2.2775	24.3	-	1.2600	-
star	36	89	36	89	3.8321	3.0	-	1.8606	-
bison	37	179	36	167	2.7002	97.5	*	1.2620	-
Tot. time							411.55		4600.03
Av. Gap						8.3			

References

[1] Doval, D., Mancoridis, S., Mitchell, B.S.: Automatic Clustering of Software Systems using a Genetic Algorithm. Proceedings of 1999 IEEE International Conference on Software Tools and Engineering Practice (STEP'99), Pittsburgh, PA, 73-81 (1999).

[2] Gay, D.M.: IBM ILOG CPLEX, User's Manual for CPLEX v12.1. (2009).

[3] Mamaghani, A.S., Meybodi, M.R.: Clustering of Software Systems Using New Hybrid Algorithms. Proceedings of 2009 IEEE International Conference on Computer and Information Technology (CIT 2009), Xiamen, China, October, 20-25 (2009).

[4] Mancoridis, S., Mitchell, B.S., Chen, Y., Gansner, E.R.: Bunch: A Clustering Tool for the Recovery and Maintenance of Software System Structures. Proceedings of 1999 IEEE of International Conference on Software Maintenance (ICSM'99), Oxford, UK, 50-59 (1999).

[5] Mitchell, B.S.: A Heuristic Search Approach to Solving the Software Clustering Problem. Ph.D Thesis, Drexel University, Philadelphia (2002).

A Classical-Quantum algorithm for Continuous Global Optimization Problems

Pedro C. S. Lara,[1] Carlile Lavor,[2] and Renato Portugal[3]

[1] *Laboratório Nacional de Computação Científica, Petrópolis, Brazil,* pcslara@lncc.br

[2] *IMECC/Unicamp, Campinas, Brazil,* clavor@ime.unicamp.br

[3] *Laboratório Nacional de Computação Científica, Petrópolis, Brazil,* portugal@lncc.br

Abstract Grover's algorithm can be employed in global optimization methods providing, in some cases, a quadratic speedup over classical algorithms. This paper describes a hybrid algorithm for continuous global optimization problems that uses a classical algorithm for finding a local minimum and Grover's algorithm to escape from this local minimum. Simulations and comparisons with algorithms from the literature are presented.

Keywords: global optimization, continuous functions, Grover's algorithm, quantum computing

1. Introduction

Global optimization algorithms play an important role in many practical problems, such as protein structure prediction and molecular dynamics simulations [8]. The goal of this work is to present a new hybrid method that uses an efficient classical algorithm for finding a local minimum and a quantum algorithm to escape from that, in order to reach the global minimum.

Recently, some papers addressed the problem of finding the global minimum of discrete [2, 6, 7] and continuous functions [9, 10], using quantum algorithms where the method used in the discrete case is an extension of Dürr and Høyer's (DH) algorithm [3], which in turn is based on the quantum search algorithm developed by Boyer et. al. (BBHT) [1]. This was made possible after Lov Grover has discovered the seminal algorithm for searching one item in an unsorted database with N distinct elements [4]. Grover's algorithm finds the position of the desired element querying the database $O(\sqrt{N})$ times, which is a quadratic improvement with respect to the number of times classical algorithms query the database.

The article is organized as follows. In Sec. 2, we describe the relationship between quantum algorithms and global optimization problems. In Sec. 3, we review DH and Baritompa et. al (BBW) algorithms, and describe the new method. In Sec. 4, we present the simulations and discuss the results of this work. Finally, in Sec. 5, we present our conclusions.

2. Global Optimization Problems and Quantum Search

We consider the global optimization problem of minimizing a real continuous function $f : [a, b] \to \mathbb{R}$, where $a, b \in \mathbb{R}$.

First, we describe the results of Grover and Boyer et. al. (BBHT) algorithms without giving details on the quantum part. The algorithms address the following problem: find $x_0 \in \{0, \cdots, N-1\}$ by querying $f : \{0, \cdots, N-1\} \to \{0, 1\}$, where $N = 2^n$ and n is some positive integer, such that

$$f(x) = \begin{cases} 1, & \text{if } x \in M; \\ 0, & \text{otherwise,} \end{cases} \tag{1}$$

where $M \subset \{0, \cdots, N-1\}$. Grover's algorithm finds x_0 with probability greater than or equal to $1 - 1/N$ by querying f around $\pi\sqrt{N}/4$ times. Internally, the algorithm uses a vector in a Hilbert space that undergoes $\pi\sqrt{N}/4$ rotations of small angles θ, such $\sin(\theta/2) = 1/\sqrt{N}$. In each rotation, function f is queried one time.

BBHT algorithm generalizes Grover's algorithm in two directions. Firstly, it is considered the case $|M| > 1$ and showed that the number of rotations required to find one element in M with probability greater than or equal to $1 - 1/N$ is

$$\frac{\pi}{4}\sqrt{\frac{N}{|M|}}.$$

Secondly, it is addressed the problem of finding one element in M without knowing a priori the number of elements of M. The main problem in this case is to know what is the best number of rotations. If the algorithm performs too few or too many rotations, the probability to find the correct results becomes small. The algorithm can be generalized to a global optimization algorithm and can be put in the following form using pseudo code language (see [6]):

Global Optimization Algorithm (GOA)

```
1. Generate $x_0$ uniformly in $\{0, \cdots, N-1\}$ and set $c = 1$, $y_0 = f(x_0)$, and $M_c = \{x | f(x) < y_{c-1}\}$.
2. For $i = 1, 2, \cdots$, do
   (a) Choose an integer $r$ by some method
   (b) Apply Grover's search with $r$ rotations, and output an $x \in \{0, \cdots, N-1\}$.
   (c) If $x \in M_c$ then
       Set $x_c = x$, $y_c = f(x_c)$, and increment $c$.
```

GOA is the basis for the quantum algorithm used in this work.

3. DH, BBW, and the New Method

The first quantum algorithm for finding the minimum value of a finite set of numbers is the Dürr-Høyer (DH) algorithm [3]. This algorithm can be seen as finding the minimum value of a function $f : \{0, \cdots, N-1\} \mapsto \mathbb{R}$, where $N = 2^n$ and n is a positive integer. The goal of the algorithm is to find the point in the domain corresponding to the minimum value of f by querying f the least number of times. This algorithm was improved in Ref. [5] by decreasing the number of intermediate measurements from $\log^2 N$ to $\log N$.

DH algorithm is an example of a global optimization algorithm (GOA), described in the last section. The integer r which specifies the rotation number is selected randomly in the range $\{0, \cdots, \lceil m-1 \rceil\}$, where m is a parameter that increases as $m = \lambda\, m$ ($\lambda = 1.34$ as suggested in the BBH algorithm) at each round that had improved the best minimum current value. The details can be found in Refs. [3, 2]. BBH algorithm improved the DH algorithm by carefully choosing a sequence of rotation numbers using a deterministic approach. The details can be found in Refs. [2, 6].

The new method is a hybrid algorithm that employs a classical algorithm to descend to a local minimum and DH algorithm to escape from that towards another better candidate. The classical algorithm uses initially neighboring points at distance $\Delta x_0 = 1$ which increases as $\Delta x_k = \gamma \Delta x_{k-1}$. We tune parameter γ for the kind of functions that we are using.

New Method

```
1. Generate $x'$ uniformly in $\{0, \cdots, N-1\}$ and set $y' = f(x')$.
2. Use the classical descent method using $x'$ with $\Delta x_0 = 1$, output $x_0$ and set $y_0 = f(x_0)$.
   Define $M_i = \{x | f(x) < y_{i-1}\}$
3. For $i = 1, 2, \cdots$, do
   (a) Choose a random rotation number $r_i$ uniformly distributed on $\{0, \cdots, \lceil m-1 \rceil\}$.
   (b) Apply Grover's search with $r_i$ rotations and output an $x' \in \{0, \cdots, N-1\}$.
   (c) If $x' \in M_i$ then
         Use the classical descent method using $x'$ with $\Delta x_i = \gamma \Delta x_{i-1}$, output $x'$.
         Set $x_i = x'$, $y_i = f(x_i)$, and $m = 1$
       else
         Set $x_i = x_{i-1}$, $y_i = f(x_i)$, and $m = \lambda\, m$.
```

4. Computational Results

We have implemented DH, BBW, and the new method in the C language. For the simulations, we use a variation of the Shekel's objective function suggested in Ref. [11], given by

$$f(x) = \sum_{i=1}^{8} \mathbf{r}_i \cdot g(x - i \cdot 128), \text{ where } \quad g(x) = -\sum_{i=1}^{16} \frac{1}{\mathbf{k}_i \left(\frac{x}{16} - \mathbf{a}_i\right)^2 + \mathbf{c}_i},$$

and $\mathbf{r}$ is a vector in $\mathbb{R}^8$ with entries uniformly distributed in the range $(0, 10)$. Vectors $\mathbf{k}$, $\mathbf{a}$, and $\mathbf{c}$ are described in Ref. [11]. This class of functions has many local minima and maxima. Figure 1 shows an example of a function in this class.

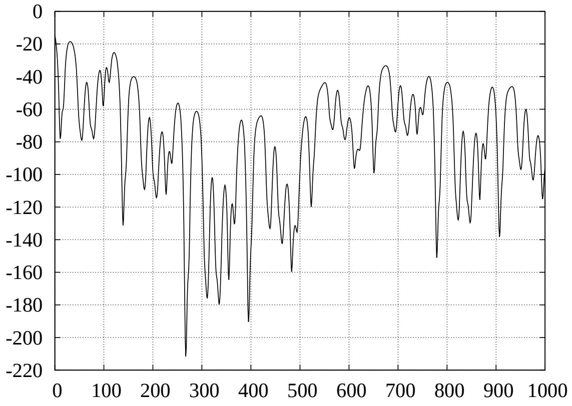


Figure 1. An example of the behavior of the objective function $f(x)$.

To compare the algorithms, we have tuned parameter γ for the classical algorithm and have set $\gamma = 1.2$. We first create a sample with 1024 function values taking $\{0, 1, \ldots, 1023\}$ as the domain set. With this sample, we apply each of the three global optimization algorithms. In each algorithm and in each round, we compute the success probability and the total effort (number of times that the function f is evaluated plus the increment). We average out this process 100,000 times for each method. In each round, we generate a new vector $\mathbf{r}$, creating a new objective function f. We analyze two case studies in order to compare with the results of Baritompa et. al. [2]. The first case explores 1% of the best of all elements and the second explores 0.2% (Figure 2).

The new method is better than the previous methods in both tests. Using the search rate 0.2%, the new method has required approximately 10 units of effort less than BBW algorithm and 35 units of effort less than DH algorithm, with 90% of success probability.

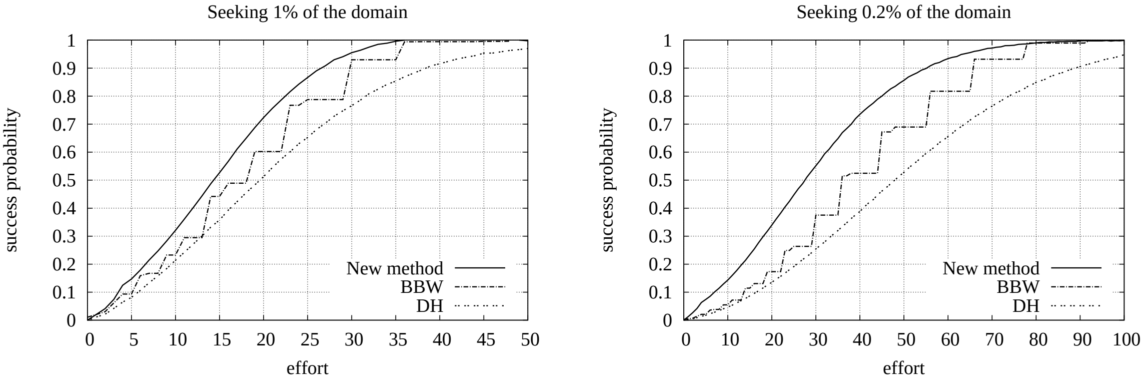


Figure 2. Performance graphs comparing the new method with Baritompa et. al. (BBW) and Dürr-Hoyer (DH)

5. Summary

This paper has proposed a new hybrid algorithm for global optimization of continuous functions using DH algorithm (quantum part) and a classical method that finds efficiently a local minimum. Our numerical simulations show that the DH, BBW, and the new method have very similar asymptotic behavior. However, the new method is more efficient than the other algorithms in the cases presented.

The new method can be generalized considering functions of many variables. This generalization and the application of the new method in other problems are in progress.

Acknowledgments

The authors would like to thank FAPESP and CNPq for their financial support.

References

[1] M. Boyer, G. Brassard, P. Høyer, and A. Tapp. Tight bounds on quantum searching. Fortschritte der Physik, 46:493–506, 1998.

[2] W.P. Baritompa, D.W. Bulger, and G.R. Wood. Grover's quantum algorithm applied to global optimization. SIAM Journal of Optimizaton, 15:1170–1184, 2005.

[3] C. Dürr and P. Høyer. A quantum algorithm for finding the minimum, http://lanl.arxiv.org/abs/quant-ph/9607014, 1999.

[4] L.K. Grover. Quantum Mechanics Helps in Searching for a Needle in a Haystack. Physical Review Letters, 79:325–328, 1997.

[5] L.A.B. Kowada, C. Lavor, R. Portugal, and C.H. Figueiredo. A new quantum algorithm to solve the minimum searching problem. International Journal of Quantum Information, 6:427–436, 2008.

[6] Y. Liu and G.J. Koehler. Using Modifications to Grover's Search Algorithm for Quantum Global Optimization. European Journal of Operational Research, 207;620-632, 2010.

[7] Y. Liu and G.J. Koehler. A Hybrid Method for Quantum Global Optimization. Journal of Global Optimization, 52:607–626, 2011.

[8] P. Pardalos, D. Shalloway, and G. Xue. Optimization methods for computing global minima of nonconvex potential energy functions. Journal of Global Optimization, 4:117–133, 1994.

[9] V. Protopopescu and J. Barhen. Solving a Class of Continuous Global Optimization Problems using Quantum Algorithms. Physics Letters A, 296:9–14, 2002.

[10] V. Protopopescu and J. Barhen. Quantum Algorithm for Continuous Global Optimization. Qi Liqun (ed.) et al., Optimization and control with applications. New York, Springer, 2005.

[11] A. Shpak, Global optimization in one-dimensional case using analytically defined derivatives of objective function. Computer Science Journal of Moldova, 3:168–184, 1995.

A hybrid algorithm between branch-and-bound and outer approximation for Mixed Integer Nonlinear Programming

Wendel A. X. Melo[1], Marcia H. C. Fampa[2], and Fernanda M. P. Raupp[3]

[1] *Universidade Federal do Rio de Janeiro, Rio de Janeiro, Brazil,* wendelmelo@cos.ufrj.br

[2] *Universidade Federal do Rio de Janeiro, Rio de Janeiro, Brazil,* fampa@cos.ufrj.br

[3] *Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro, Brazil,* fraupp@puc-rio.br

Abstract In this work, we present a new hybrid algorithm for convex Mixed Integer Nonlinear Programming combining branch-and-bound and outer approximation algorithms in an effective and efficient way.

Keywords: mixed integer nonlinear programming, branch-and-bound, outer approximation, hybrid algorithm

1. Introduction

Mixed Integer Nonlinear Programming (MINLP) problems are characterized by the presence of nonlinear functions of continuous and discrete variables. The MINLP problem addressed in this work can be algebraically represented in the following way:

$$\begin{array}{lll} (P) & \text{minimize}_{x,y} & f(x,y) \\ & \text{s. t.:} & g(x,y) \ \leq \ 0 \\ & & x \in X, \ y \in Y \cap \mathbb{Z}^{n_y} \end{array} \tag{1}$$

where X and Y are polyhedral subsets of $\mathbb{R}^{n_x}$ and $\mathbb{R}^{n_y}$, respectively, and Y is bounded. The functions $f : X \times Y \rightarrow \mathbb{R}$ and $g : X \times Y \rightarrow \mathbb{R}^m$ are convex and twice continuously differentiable. We call problem (1) by P and its continuous relaxation by $\tilde{P}$.

Algorithms in two distinct methodological classes have been employed to solve P: outer approximation algorithms [2] and branch-and-bound algorithms (the reader interested in MINLP algorithms can see [3, 4]). In [1], a hybrid approach combining algorithms in these two classes was introduced. In this work, we propose a new hybrid algorithm combining also the two cited methodologies in a more effective way than [1]. Our main goal is to potentialize the particular advantages of each class and remediate theirs drawbacks. In Section 2, we present an outer approximation algorithm and in Section 3, we show the proposed hybrid approach.

2. Outer Approximation

Proposed by Duran and Grossmann in [2], the Outer Approximation (OA) algorithm alternates between solving a Mixed Integer Linear Programming problem (MILP) and one or two NonLinear Programming problems (NLP). Its main idea is to approximate P by the following MILP problem that is built using linearization of functions in P on a set T of t linearization points, i.e., $T = \{(x^0, y^0), (x^1, y^1), \ldots, (x^t, y^t)\}$:

$$\begin{array}{lll} \left(P^{OA}(T)\right) & \min_{\alpha,x,y} & \alpha \\ & \text{s. t.:} & \nabla f(x^k, y^k)^T \begin{pmatrix} x - x^k \\ y - y^k \end{pmatrix} + f(x^k, y^k) \;\leq\; \alpha, \;\; \forall (x^k, y^k) \in T \\ & & \nabla g(x^k, y^k)^T \begin{pmatrix} x - x^k \\ y - y^k \end{pmatrix} + g(x^k, y^k) \;\leq\; 0, \;\; \forall (x^k, y^k) \in T \\ & & x \in X, \;\; y \in Y \cap \mathbb{Z}^{n_y}. \end{array} \tag{2}$$

As P is convex, we notice that problem (2) is a relaxation of P, which provides valid lower bounds to P. The baseline of OA algorithm is showed in Algorithm 1. New linearization points are added to set T, as the algorithm evolves. This strengthens the relaxation given by (2) and generates a non-decreasing sequence of lower bounds to P.

Let $(\hat{x}, \hat{y})$ be an optimal solution of an instance of problem (2). The integer variable values $\hat{y}$ are used to build the following NLP problem from P:

$$\begin{array}{lll} (P_{\hat{y}}) & \text{minimize}_x & f(x, \hat{y}) \\ & \text{s. t.:} & g(x, \hat{y}) \;\leq\; 0 \\ & & x \in X. \end{array} \tag{3}$$

Suppose problem (3) is feasible and let $\bar{x}$ be an optimal solution. So, the point $(\bar{x}, \hat{y})$ provides an upper bound to P. Thus, OA algorithm adds this point to the set of linearization points T and starts a new iteration using as stopping rule the annulment of the optimality gap.

In the case problem (3) is infeasible, OA algorithm solves the following feasibility problem:

$$\begin{array}{lll} (P_{\hat{y}}^V) & \text{minimize}_{u,x} & \sum_{i=1}^m u_i \\ & \text{s. t.:} & g(x, \hat{y}) \;\leq\; u \\ & & x \in X, \;\; u \in (\mathbb{R}^+)^m \end{array} \tag{4}$$

Let $(\check{u}, \check{x})$ be an optimal solution of (4) in the described context. Then, the point $(\check{x}, \hat{y})$ is added to the set T. Conforming demonstrated in [2], if the KKT conditions are satisfied at the optimal solutions of (3) and (4), OA algorithm converges in a finite number of iterations.

```
ALGORITHM 1: Outer approximation ;
INPUT: $P$: Problem (1), $T_0$: initial set of linearization points (it can be empty) ;
OUTPUT: $(x^*, y^*)$: optimal solution of $P$ ;

  $z^U = +\infty$ ; $z^L = -\infty$ ;
  Let $(x^0, y^0)$ be an optimal solution of $\tilde{P}$ ;
  $T = \{T_0 \cup (x^0, y^0)\}$ ; $k = 1$ ;
  WHILE $z^U - z^L > 0$ AND $P^{OA}(T)$ is feasible
  {
     Let $(\hat{\alpha}, \hat{x}, \hat{y})$ be an optimal solution of $P^{OA}(T)$ ;
     $z^L = \hat{\alpha}$ ; $y^k = \hat{y}$ ;
     IF $P_{\hat{y}}$ is feasible
     {
        Let $x^k$ be an optimal solution of $P_{\hat{y}}$ ;
        IF $f(x^k, y^k) < z^U$
        {
           $z^U = f(x^k, y^k)$ ;
           $(x^*, y^*) = (x^k, y^k)$ ;
        }
     }
     ELSE
     {
        Let $x^k$ be an optimal solution of $P_{\hat{y}}^V$ ;
     }
     $T = T \cup (x^k, y^k)$ ;
     $k = k + 1$ ;
  }
```

3. Our hybrid algorithm

```
ALGORITHM 2: Our hybrid algorithm ;
INPUT: $P$: Problem (1), $OA(\bar{P}, T^I, z^U, time)$: OA procedure that address $\bar{P}$, with initial linearization points set $T^I$, upper bound $z^U$ with time limited to $time$ seconds. **OA procedure returns:** $status$: status of OA application, $(\bar{x}, \bar{y})$: best obtained solution of $\bar{P}$, $T^F$: final set of linearization points, $\bar{z}^L$: lower bound to $\bar{P}$ ;
OUTPUT: $(x^*, y^*)$: optimal solution of $P$ ;

  $z^U = \infty$ ; Let $(x^0, y^0)$ be an optimal solution of $\tilde{P}$ ;
  $T^P = (x^0, y^0)$ \\ Initial linearization points to $P$ ;
  $[status, (\bar{x}, \bar{y}), T^F, \bar{z}^L] = OA(\tilde{P}, T^P, z^U, OA\_time)$ ;
  IF $status$ = "optimal solution" OR $status$ = "feasible solution"
  {
    $(x^*, y^*) = (\bar{x}, \bar{y})$ ; $z^U = f(\bar{x}, \bar{y})$ ;
  }
  IF $status$ = "optimal solution", THEN RETURN ;
  Choose a variable $y_j$ with fractional value $y_j^0$ ;
  $Y^1 = Y \cap \{y \in \mathbb{R}^{n_y} : y_j \leq \lfloor y_j \rfloor\}$ ; $Y^2 = Y \cap \{y \in \mathbb{R}^{n_y} : y_j \geq \lceil y_j \rceil\}$ ;
  Let $L^i$ be a lower bound to node $i$ ; $L^1 = L^2 = \max\{f(x^0, y^0), \bar{z}^L\}$ ;
  Let $N = \{1, 2\}$ be the initial list of open nodes ;
  $i = 2$ ; $iter = 0$ ; $T^P = T^P \cup T^F$ ;
BBLOOP:
  WHILE $N \neq \emptyset$
  {
    Choose a node $k$ of $N$ ; $N = N \setminus \{k\}$ ; $iter = iter + 1$ ;
    Let $(x^k, y^k)$ be an optimal solution of $\tilde{P}_{Y^k}$ ;
    IF $f(x^k, y^k) < z^U$
    {
      IF $y^k$ is integer
      {
        $z^U = f(x^k, y^k)$ ; $(x^*, y^*) = (x^k, y^k)$ ; $T^P = T^P \cup \{(x^k, y^k)\}$ ;
        $N = N \setminus \{j : L^j \geq z^U\}$); \\ Pruning
      }
      ELSE
      {
        $\bar{L} = f(x^k, y^k)$ ;
        IF $iter \equiv 0$ (mod $freq\_OA\_subprob$) \\ Applying OA to subproblem
        {
          $T^S = \{(x^k, y^k)\}$ ;
          $[status, (\bar{x}, \bar{y}), T^F, \bar{z}^L] = OA(P_{Y^k}, T^S, z^U, OA\_time)$ ;
          IF $status$ = "optimal solution" OR $status$ = "feasible solution"
          {
            $z^U = f(\bar{x}, \bar{y})$ ; $(x^*, y^*) = (\bar{x}, \bar{y})$ ; $N = N \setminus \{j : L^j \geq z^U\}$ \\ Pruning
            $T^P = T^P \cup \{(\bar{x}, \bar{y})\}$ ;
          }
          IF $status$ = "optimal solution" OR $status$ = "infeasible problem"
          {
            GO TO BBLOOP;
          }
          $\bar{L} = \max\{\bar{L}, \bar{z}^L\}$ ;
        }
        Choose a variable $y_j$ with fractional value $y_j^k$ ; \\ Branching
        $Y^{i+1} = Y^k \cap \{y \in \mathbb{R}^{n_y} : y_j \leq \lfloor y_j \rfloor\}$ ; $Y^{i+2} = Y^k \cap \{y \in \mathbb{R}^{n_y} : y_j \geq \lceil y_j \rceil\}$ ;
        $L^{i+1} = L^{i+2} = \bar{L}$ ; $N = N \cup \{i + 1, i + 2\}$ ; $i = i + 2$ ;
      }
    }
    IF $iter \equiv 0$ (mod $freq\_OA\_prob$)
    {
      $[status, (\bar{x}, \bar{y}), T^F, \bar{z}^L] = OA(P, T^P, z^U, OA\_time)$ ; \\Applying OA to $P$
      IF $status$ = "optimal solution"
      {
        $(x^*, y^*) = (\bar{x}, \bar{y})$ ; RETURN ;
      }
      IF $status$ = "feasible solution"
      {
        $z^U = f(\bar{x}, \bar{y})$ ; $(x^*, y^*) = (\bar{x}, \bar{y})$ ; $N = N \setminus \{j : L^j \geq z^U\}$ ; \\ Pruning
      }
      $T^P = T^P \cup T^F$ ; \\ Accumulating linearization points
    }
  }
```

Here, we propose a new hybrid algorithm combining branch-and-bound and outer approximation algorithms, showed in Algorithm 2. The inspiration to develop this algorithm comes from the hybrid algorithm proposed by Bonami *et al.* in [1]. Let us first define the subproblem addressed at each node of branch-and-bound tree in a given partition $\bar{Y} \subset Y$ as:

$$\begin{array}{lll} (P_{\bar{Y}}) & \text{minimize}_{x,y} & f(x,y) \\ & \text{s. t.:} & g(x,y) \ \leq \ 0 \\ & & x \in X, \ y \in \bar{Y} \cap \mathbb{Z}^{n_y}. \end{array} \tag{5}$$

The general idea behind the proposed approach is very simple: it makes space partitioning in the NLP branch-and-bound tree, and, then, applies outer approximation algorithm to some of the partitions Y^k, i.e., applies OA to some subproblems P_{Y^k}, with a time limit to spend. Bonami *et al.* adopt this strategy only once in their algorithm before beginning the space partitioning, i.e., OA is applied to solve the original MINLP problem in the root node, as they use the OA based branch-and-cut [5]. Here, we adopt this strategy in the root node and also in some generated subproblems along the evolution of the algorithm, resulting in several calls to OA procedure. During enumeration, the proposed algorithm comes back to the original MINLP problem considered in the root node to make new OA iterations with limited time. Integer solutions found in the branch-and-bound tree are used as linearization points when we apply again OA algorithm to solve P. On the other hand, OA algorithm collaborates with enumeration scheme providing stronger lower bounds to the addressed subtrees and integer solutions that improve the upper bound to P.

At every $freq\_OA\_subprob$ branch-and-bound iterations (e.g. 50), OA algorithm is applied to the current subproblem P_{Y^k} and at every $freq\_OA\_prob$ iterations (e.g. 200), OA algorithm is applied to the original problem P. We observe that if we interrupt the OA algorithm at the end of a given iteration, saving the set of linearization points, and later restart it by using this same set as input, OA algorithm continues as the same way as if it has never been interrupted, i.e., when we save the set of the current linearization points at the end of an iteration, we are saving the current state of algorithm as a whole. In this way, considering the calls to OA procedure at every $freq\_OA\_prob$ branch-and-bound iterations, this would be, in principle, like solving P using OA algorithm by making "pauses" in its execution. During these "pauses", the proposed algorithm performs $freq\_OA\_prob$ branch-and-bound iterations. Actually, it does not happen precisely in this way because during these "pauses", we can add new linearization points from integer solutions found by branch-and-bound algorithm. In this sense, we hope to speed up the performance of OA algorithm, i.e., saving some of its regular iterations.

The results of preliminary computational tests show that the proposed hybrid algorithm has better performance in comparison to pure outer approximation, pure branch-and-bound and hybrid from [1] algorithms on several instances of MINLP. The proposed hybrid algorithm will be available in the new open MINLP solver under development called *Muriqui*.

References

[1] P. Bonami, L. T. Biegler, A. R. Conn, G. Cornuéjols, I. E. Grossmann, C. D. Laird, J. Lee, A. Lodi, F. Margot, N. Sawaya, A. Wächter. An Algorithmic Framework for Convex Mixed Integer Nonlinear Programs. *Discrete Optimization 5*, pages 186–204, 2008.

[2] M. Duran, I. E. Grossmann. An outer-approximation algorithm for a class of mixed-integer nonlinear programs. *Mathematical Programming 36*, pages 307–339, 1986.

[3] I. E. Grossmann. Review of nonlinear mixed-integer and disjunctive programming techniques. *Optimization and Engineering 3*, pages 227–252, 2002.

[4] W. Melo. *Algorithms for Mixed Integer Nonlinear Programming*. Master thesis, Universidade Federal do Rio de Janeiro (Federal University of Rio de Janeiro), 2012 (printed in portuguese).

[5] I. Quesada, I.E. Grossmann. An LP/NLP based branch and bound algorithm for convex MINLP optimization problems. *Computers & Chemical Engineering 16 10-11*, pages 937–947, 1992.

A Stochastic Mixture Surrogate Model Algorithm for Computationally Expensive Black-Box Global Optimization Problems

Juliane Müller,[1,2] Christine A. Shoemaker,[2] and Robert Piché[1]

[1] *Tampere University of Technology, Department of Mathematics, P.O. Box 553, 33101 Tampere, Finland*
juliane.mueller@tut.fi, robert.piche@tut.fi

[2] *Cornell University, Center of Applied Mathematics 220 Hollister Hall, Ithaca, NY 14853-3501*
cas12@cornell.edu

Abstract A focus of research in derivative-free algorithms for computationally expensive global optimization problems is on the use of surrogate models with the goal to reduce the computational effort for finding (near) optimal solutions by using computationally inexpensive approximations of the objective function. Various algorithms employing different surrogate model types have been developed, but in practice it is unclear which algorithm should be applied to a certain problem. This paper examines the influence of the surrogate model type, and the strategy of selecting decision variable points at which the computationally expensive objective function is to be evaluated. Within this scope a stochastic mixture surrogate model algorithm (SO-MMS) is introduced that converges to the global optimum in probability. The efficiency of SO-MMS is compared to other widely used surrogate model algorithms on 19 global optimization test problems from the literature, an application problem dealing with groundwater bioremediation, and an application problem arising from energy generation using kites. The results show that random sampling strategies are more successful than 'more sophisticated' sampling strategies that are based on the optimization of some auxiliary function. As a result of this study, a modularized surrogate model algorithm toolbox for Matlab has been developed that allows the user to choose between mixture surrogate models.

Keywords: mixture surrogate model, global optimization, radial basis function, Kriging, derivative-free

1. Surrogate Model Algorithms

Application problems arising in engineering and management often require solving optimization problems with computationally expensive objective functions [2, 3, 5, 9, 15]. For example, finding the optimal shape of a structure may require a nonlinear finite element analysis, or determining a strategy for cleaning up contaminated groundwater at minimal cost may require the solution of a system of partial differential equations. To keep the computation times low, finding (near) optimal solutions should therefore require as few objective function evaluations as possible. Algorithms using surrogate models have been developed to achieve this goal [6–8, 10, 12, 13].

Surrogate models are approximations of the true objective function [1], and during the optimization process the information from the surrogate model is used to guide the search for the global optimum. Surrogate model algorithms consist in general of four major steps. At first, an initial experimental design is generated, and the computationally expensive objective function is evaluated at these points. Secondly, the parameters of the chosen surrogate model are computed using the data from the initial experimental design. Third, some criterion is used to determine the next point in the variable domain where the true objective function is evaluated, and fourth, given the new data point, the parameters of the surrogate model are updated. The algorithm iterates through steps three and four until a given stopping criterion is met.

Several options can be used in each of these steps. The initial experimental design can be created, for example, by Latin hypercube sampling or orthogonal arrays. The surrogate model may be chosen to be interpolating (e.g. Kriging [8] or radial basis functions (RBF) [6, 12, 13]) or non-interpolating (e.g. multivariate adaptive regression splines (MARS) [4] or polynomial regression models [11]), whereas mixtures of surrogate models [10] may either be interpolating or non-interpolating depending on the individual models in the mixture. For determining the next sample site in the third step of the algorithm an auxiliary (global) optimization problem may be solved [6, 8] or a random sampling strategy [12] may be applied. Further differences such as the optimization strategy used to find the optimum of the auxiliary problem exist. Stopping criteria for the algorithm may be, for example, a given maximal number of allowed function evaluations, or the algorithm may be stopped if no improvements have been found within several consecutive function evaluations. In general, it is a priori unclear which of these various options should be used to obtain the best result for a given problem, and trying different algorithm implementations is in practice computationally infeasible.

The goal of this paper is, on the one hand, to study the influence of the surrogate model and the strategy of determining the next sample point for doing the expensive function evaluation on the results. On the other hand, an improvement of the mixture surrogate model algorithm [10] by using a random sampling strategy is presented, and numerical experiments show how crucial the choice of the sampling strategy is. The improved algorithm, SO-MMS (Surrogate Optimization - Mixture Model Stochastic), converges to the global optimum in probability. Within the scope of this study, a Matlab toolbox has been developed that allows the user to choose between different (mixture) surrogate model algorithms, initial experimental design strategies, and methods for determining the sample points.

2. Numerical Experiments

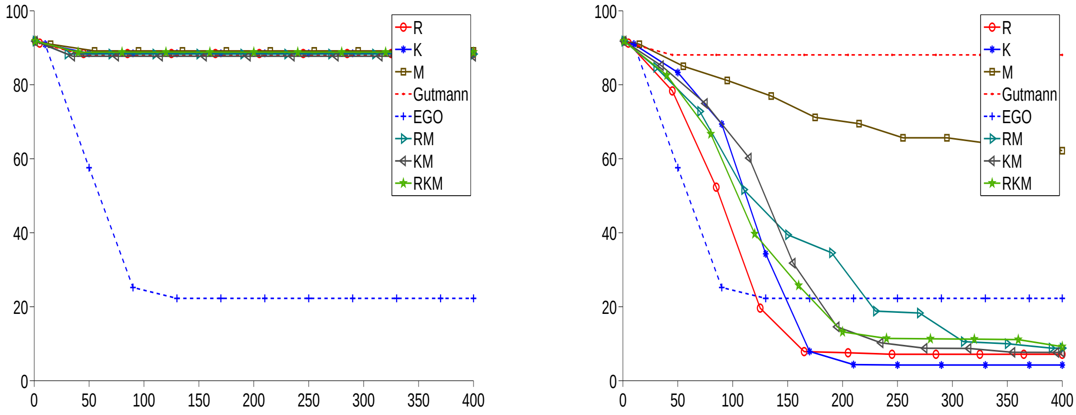
In the numerical experiments mixtures of different surrogate models have been used to examine how the performance of 'bad' individual models (models that deliver significantly worse results than the best model) can be improved by building a mixture with a 'good' model. The improved mixture surrogate model algorithm SO-MMS, has been compared to the efficient global optimization algorithm (EGO [8], implementation from [3]), Gutmann's algorithm [6], and an improved version of the G-MSRBF algorithm by Regis and Shoemaker [12] on 19 literature test problems with two to 30 dimensions, a 12-dimensional groundwater bioremediation application problem, and a 13-dimensional application problem arising from energy generation using kites. The problems are unconstrained and of the general form

$$\min\ f(\mathbf{x}), \quad \text{s.t. } \mathbf{x}^l \leq \mathbf{x} \leq \mathbf{x}^u, \tag{1}$$

where $\mathbf{x}^l$ and $\mathbf{x}^u$ are the lower and upper variable bounds, respectively, and $\mathbf{x} \in \mathbb{R}^d$, where d is the problem dimension. 30 trials have been made for every test problem and every algorithm. All algorithms start from the same initial experimental design, and the maximum number of allowed function evaluations was limited to 400. The algorithms are compared with respect to the best average function value found over 30 trials after an equal number of function evaluations.

Figures 1(a) and 1(b) show the average objective function value versus the number of function evaluations for a subset of the examined algorithms for the ten-dimensional Schoen [14] test function. Figure 1(a) shows the results of SO-MMS when the minimum point of the response surface is used as criterion for determining the next sample site in every iteration. Clearly, EGO performs best, whereas SO-MMS and Gutmann's method perform in comparison about equally bad.

Figure 1(b) illustrates the results of the same SO-MMS algorithms when using a random sampling strategy. The figure shows that when a random sampling strategy is used, Gutmann's method is clearly outperformed by EGO and all SO-MMS algorithms. EGO performs better than SO-MMS up to about 100 function evaluations, but is then outperformed by almost all SO-MMS algorithms (except when only MARS is used as surrogate model). It can be seen that for this test problem using MARS as individual surrogate model would be the worst choice, and using Kriging or RBF would be the best choice. However, if it is a priori unknown which surrogate model will perform best, then clearly a mixture model should be favored. The figure shows that building a mixture model of MARS and RBF and/or Kriging (RM, KM, RKM) significantly improves the performance than if only MARS is used. This example clearly shows how much influence the strategy of choosing the next sample site has on the performance of the algorithms, and how using a mixture surrogate model can prevent choosing the worst individual surrogate model if the performance of individual models is a priori unknown.



(a) SO-MMS with sampling at minimum of response surface (solid graphs), EGO and Gutmann's method

(b) SO-MMS with random sampling strategy (solid graphs), EGO and Gutmann's method

Figure 1. Average objective function value vs. number of function evaluations for ten-dimensional Schoen function. Illustrated are SO-MMS algorithms using RBF (R), Kriging (K), MARS (M), a mixture of RBF and MARS (RM), a mixture of Kriging and MARS (KM), and a mixture of RBF, Kriging and MARS (RKM).

Acknowledgements

The first author thanks the Finnish Academy of Science and Letters, Vilho, Yrjö and Kalle Väisälä Foundation for the financial support. The second author has been supported by NSF CISE: AF 1116298 and CPA-CSA-T 0811729.

References

[1] A.J. Booker, J.E. Dennis Jr, P.D. Frank, D.B. Serafini, V. Torczon and M.W. Trosset. A Rigorous Framework for Optimization of Expensive Functions by Surrogates. *Structural Multidisciplinary Optimization*, 17: 1–13, 1999.

[2] T. Dede, S. Bekiroğlu and Y. Ayvaz. Weight Minimization of Trusses with Genetic Algorithm. *Applied Soft Computing*, 11: 2565–2575, 2011.

[3] A. Forrester, A. Sóbester and A. Keane. Engineering Design via Surrogate Modelling - A Practical Guide. *Wiley*, 2008.

[4] J.H. Friedman. Multivariate Adaptive Regression Splines. *The Annals of Statistics*, 19: 1–141, 1991.

[5] B. Glaz, P.P. Friedmann and L. Liu. Surrogate Based Optimization of Helicopter Rotor Blades for Vibration Reduction in Forward Flight. *Structural and Multidisciplinary Optimization*, 35: 341–363, 2008.

[6] H.-M. Gutmann. A Radial Basis Function Method for Global Optimization. *Journal of Global Optimization*, 19: 201–227, 2001.

[7] D.R. Jones. A Taxonomy of Global Optimization Methods Based on Response Surfaces. *Journal of Global Optimization*, 21: 345–383, 2001.

[8] D.R. Jones, M. Schonlau and W.J. Welch. Efficient Global Optimization of Expensive Black-Box Functions. *Journal of Global Optimization*, 13: 455–492, 1998.

[9] J.-C. Jouhaud, P. Sagaut, M. Montagnac and J. Laurenceau. A Surrogate-model Based Multidisciplinary Shape Optimization Method with Application to a 2D Subsonic Airfoil. *Computers & Fluids*, 36: 520–529, 2007.

[10] J. Müller and R. Piché. Mixture Surrogate Models Based on Dempster-Shafer Theory for Global Optimization Problems. *Journal of Global Optimization*, 51: 79–104, 2010.

[11] R.H. Myers and D.C. Montgomery. Response Surface Methodology, Process and Product Optimization using Designed Experiments. *Wiley-Interscience Publication*, John Wiley & Sons, Inc., 1995.

[12] R.G. Regis and C.A. Shoemaker. A Stochastic Radial Basis Function Method for the Global Optimization of Expensive Functions. *INFORMS Journal on Computing*, 19: 497–509, 2007.

[13] R.G. Regis and C.A. Shoemaker. Improved Strategies for Radial Basis Function Methods for Global Optimization. *Journal of Global Optimization*, 37: 113–135, 2007.

[14] F. Schoen. A Wide Class of Test Functions for Global Optimization. *Journal of Global Optimization*, 3: 133–137, 1993.

[15] B.A. Tolson and C.A. Shoemaker. Dynamically Dimensioned Search Algorithm for Computationally Efficient Watershed Model Calibration. *Water Resources Research*, 43: W01413, 16 pages, 2007.

An Algorithm for Signomial Geometric Programming

Roberto Quirino do Nascimento

Informatics Department, Federal University of Paraiba, Paraiba,Brazil, quirino@de.ufpb.br

Abstract In this paper an interior point methodology for solving geometric programming signomial is proposed, the classical methodology used to solve this problem is condensation, this technique approximates an sum of positive terms by a product of such terms associated with a set of weights whose sum is 1, here this condition is relaxed and used as stop criterion for the algorithm, the method is an adaptation of the primal-dual posinomial, it is then used to solve this new problem, whose set of feasible solutions contains the set of feasible solutions of the original problem and it is a convex set, the algorithm is implemented in matlab, where we present the computational results in solving obtained some problems existing in the literature.

Keywords: Global Optimization, Signomial Geometric Programming, Interior Point.

1. Introduction

Geometric Programming is a mathematical programming technique often applied to minimize a class of generalized polynomial functions called signomial functions. The technique was deve-loped in 60's by R.J. Duffin and E. L. Peterson focusing on posynomial geometric programming problems, i.e, strictly positive signomial functions. Nevertheless, nowadays the technique is still strongly applied as a method for solving Signomial Geometric Problems (SGP), Quadratic Problems with quadratic constraints, Allocation and Financial Problems, and many others. To these kind of problems, the original approach by [3] did not seem to be totally useful and an alternative technique, named condensation, was proposed. To condense posynomial functions means to approximate them using the inequality between arithmetic mean and geometric mean or harmonic mean. This approach can give a solution which is just either a stationary point or a local minimum [5]. Other techniques applied to the same purpose are the global optimization techniques based on cutting planes and linearization [9, 6, 7].

1.1 Signomial Geometric Programming

A Signomial Geometric Programming Problem (SGP) also known in the literature as Generalized Geometric Programmig is an optimization problem stated as follows:

$$\begin{array}{llll} & \text{Minimize} & g_0(t) & & (1)\\ \text{SGP} & \text{Subject to} & g_k(t) \leq 1 & k = 1, \ldots, q, & (2)\\ & & g_k(t) \geq 1 & k = q+1, \ldots, p, & (3)\\ & & t_j > 0 & j = 1, \ldots, m. & (4) \end{array}$$

so that

$$g_k(t) = \sum_{i \in J[k]} c_i \prod_{j=1}^{m} t_j^{a_{ij}} \quad k = 0, 1, \ldots, p, \tag{5}$$

$$J[k] = \{m_k, m_{k+1}, \ldots, n_k\} \qquad k = 0, 1, \ldots, p, \tag{6}$$

$$m_0 = 1, \quad m_1 = n_0 + 1, m_2 = n_1 + 1, \ldots, m_p = n_{p-1} + 1, \quad n_p = n.$$

Exponents a_{ij} are arbitrary constants, coefficients c_i are positive, functions g_k are called posy-nomials, terms $c_i \prod_{j=1}^{m} t_j^{a_{ij}}$ are called existing posynomials terms of the problem and variables t_j are primal variables. Signomial geometric programs where constraints $g_k, k = q+1, \ldots, p$ are ausents, are called posinomial programs, such problems have the property that every locally optimal solution is global, while for signomial problems we obtain only local solutions when using traditional methodologies.

The aim of this paper is to present an approach to solve the signomial geometric programming problem a little different from the classical approach that uses condensation (see [1, 5])and the more recent approaches that use global optimization techniques based on cutting planes and branch and bound methods (see [6, 7, 9]),). The idea is to look solutions in a convex set greater than the feasible region of the problem (SGP) and generate a sequence of solutions that converges to a solution viable nearest the global solution of the problem obtained in the initial convex set.

Consider the following extended version of the problem SGP:

$$\text{PGP} \qquad \begin{array}{rcll} \text{Minimize} \quad g_0(t) & & & \\ \text{Subject to} \quad g_k(t) & \leq & 1 & k = 1, \ldots, q, \quad (7) \\ \dfrac{w_i}{c_i} \prod_{j=1}^{m} t_j^{-a_{ij}} & \leq & 1 & i \in J[k], k = q+1, \ldots, p, \quad (8) \\ \prod_{i \in J[k]} \left(\dfrac{w_i}{p_i}\right)^{-p_i} & \leq & \beta_l & l = 1, \ldots, p-q-1, \quad (9) \\ \sum_{i \in J[k]} w_i & \leq & 1 & k = q+1, \ldots, p, \quad (10) \\ t_j > 0, \quad w_i & > & 0 & j = 1, \ldots, m. \quad (11) \end{array}$$

Where p_i are weights such that:

$$\sum_{i \in J[k]} p_i = 1 \tag{12}$$

$\beta_l \geq 1, l = 1, \ldots, p-q-1$ problem *PGP* is a posinomial geometric programming problem, therefore, any local solution, if exists is also a global solution.

Are

$$S_{SGP} = \{t \in R^m : \quad \text{constraints } (2) - (4) \text{ are satisfied}\} \quad \text{and}$$

$$S_{PGP} = \left\{(t, w) \in R^m \times R^{p-q-1} : \text{constraints } (7) - (10) \text{ are satisfied}\right\}$$

Hypothesis:

- $S_{SGP} \neq \emptyset$;
- PGP Problem has optimal solution.

Clearly we have $S_{SGP} \subseteq S_{PGP}$, in addition:

- If $\beta_l = 1$, for all $\quad l = 1, \ldots, p-q-1$, e $(\bar{t}, \bar{w}) \in S_{PGP}$ then $\bar{t} \in S_{SGP}$
- $min\,\{g_0(t) : (t, w) \in S_{PGP}\} \leq min\,\{g_0(t) : t \in S_{SGP}\}$

Let $(t^*, w^*) \in S_{PGP}$ an optimal solution of the PGP problem, and $f_{t^*} : R^m_{++} \to R$ defined by:

$$f_{t^*}(t) = \frac{1}{m}\sum_{i=1}^{m}\frac{2}{3}\left(\frac{1}{2}\left(\frac{t_i}{t_i^*}\right)^2 + \left(\frac{t_i}{t_i^*}\right)^{-1}\right) \tag{13}$$

- f_{t^*} is a convex function and t^* is the unique unconstrained minimum of this function
- f_{t^*} is a posinomial function.

Consider the parametes $\beta_l^0 > 1, \quad l = 1, \dots, p-q-1$ and the weights p_i such that $\sum_{i \in J[k]} p_i = 1$ as problem data, PGP can be solved using several methods we will use primal-dual interior point methods (see [4, 5]),solved this problem, let $(t^*, w^*) \in S_{PGP}$ a optimal solution, we can determine from w^* new weights p^0 satisfying $\sum_{i \in J[k]} p_i^0 = 1$, to generate sequences β^s andp^s so that $1 \le \beta_l^{s+1} < \beta_l^s < \beta_l^0, \;\; \sum_{i \in J[k]} p_i^s = 1$ and formulate now the following problem:

$$\begin{array}{llcll}
 & \text{Minimize} \;\; g_0(t) + P^s f_{t^*}(t) & & & (14) \\
 & \text{Subject to} \;\; g_k(t) & \le & 1 \qquad k = 1, \dots, q, & (15) \\
 & \dfrac{w_i}{c_i}\displaystyle\prod_{j=1}^{m} t_j^{-a_{ij}} & \le & 1 \qquad i \in J[k], k = q+1, \dots, p, & (16) \\
\text{PGP-s} & \displaystyle\prod_{i \in J[k]} \left(\frac{w_i}{p_i^s}\right)^{-p_i^s} & \le & \beta_l^s \qquad l = 1, \dots, p-q-1, & (17) \\
 & \displaystyle\sum_{i \in J[k]} w_i & \le & 1 \qquad k = q+1, \dots, p, & (18) \\
 & t_j > 0, \quad w_i & > & 0 \qquad j = 1, \dots, m. &
\end{array}$$

where $P > 0$ is a penalty, $s \in N$, the objective function given in (**??**) will be minimized by giving preference to solutions close to t^*, furthermore the constraints that will be modified at the s-th iteration will not affect the original problem.

2. Algorithm and Computational Results

Given a SGP problem, and a weight vector satisfying (12), $\beta_l^{ini} > 1 \quad l = 1, \dots, p-q-1$ e $\epsilon > 0$, our algorithm will be formulated as:
Algorithm 1

```
SGP_Algorithm {% Step 0
  Solve PGP problem and call $t^0, w^0$ the solution
    $\beta_l^0 := \prod_{l=1}^{p-q-1} \max(1, \beta_l^{ini})^{\frac{1}{p-q}}; p_i^0 = \frac{w_i^0}{\sum_{i \in J[k]} w_i^0} \quad i \in J[k];$
 % Step 1
  Use weights $p^0$, parameters $\beta_l^0$, the solution $t^0$ and build the PGP-s problem ; s := 0;
 % Step 2
   { While $\beta_l^s > 1 + \epsilon$
       $p_i^{s+1} = \frac{w_i^s}{\sum_{i \in J[k]} w_i^s} \quad i \in J[k]; \qquad \beta_l^{s+1} = \prod_{l=1}^{p-q-1} \max(1, \beta_l^s)^{\frac{1}{p-q}}$
       $p^0 = p^{s+1}, \beta_l^0 = \beta_l^{s+1}, s = s + 1$ .
         Solve problem PGP-s }     }
```

This algorithm was implemented in Matlab, PGP and PGP-s problems were solved using the code GGPLAB. (see [2]).

Example This is a classical case, where a SGP solution obtained with condensation technique is a local solution only (see [1] pag. 358), the optimal solution obtained is:

$\bar{t_1} = 3.822875662810514$ $\bar{t_2} = 4.822875645827997$ when we set $t_{ini} = (4, 4.5)$ as initial point, see [1] pg. 358.

$$\begin{array}{ll} \text{Minimize} & t_1 \\ \text{Subject to} & .25t_1 + 0.5t_2 - \frac{1}{16}t_1^2 - \frac{1}{16}t_2^2 \leq 1; \quad \dfrac{1}{16}t_1^2 t_2^{-1} + \dfrac{1}{16}t_2 + \dfrac{7}{3}t_2^{-1} - t_1 t_2^{-1} \leq 1 \\ & 1 \leq t_1 \leq 100; \qquad 1 \leq t_2 \leq 100 \end{array}$$

The solution obtained in the presente work is $t_1^* = 1.177124318528388$ $t_2^* = 2.177124308566496$ was obtained without initial point, in relationship with originals variables.

3. Summary

In this work, we solved signomial geometric programming problems, relaxing the feasible region to obtain a larger convex set , where the objective function has global solution, then we approach the original feasible region iteratively, penalizing the distance between the viable solutions and the optimal solution of the initial problem, the algorithm was implemented and tests using examples of the literature were carried out and they confirmed the effectiveness of our algorithm.

Acknowledgments

The authors want to thank the Coordination of Improvement of Higher Education - CAPES research agency of the Brazilian government, for their financial support to this research.

References

[1] Beightler, C.S.and Phillips,D.T., *Applied Geometric Programming*, John Wiley & Sons, New York,1976.

[2] Boyd, S. and Jean Kim, S. and Vandenberghe L. and Hassibi Arash,*A Tutorial on geometric programming* Optim Eng.(2007)8:67-127

[3] Duffin, R.J., Peterson, E.L. and Zener, C., (1967), *Geometric Programming – Theory and Applications*, John Wiley & Sons, New York, 1967.

[4] Kortanek, K.O., Xu, X. and Ye, Y., An Infeasible Interior Point Method for Solving Primal and Dual Geometric Programs, *Mathematical Programming* 76, 155-181,1995.

[5] Yang, H. S, Bricker D. L.(1997) *Investigation of Path-Following algorithms for Signomial Geometric Programming Problems* European Journal of Operations Research, 103 pag 230-241.

[6] Yanjun Wang and Zhian Liang, *A deterministic global optimization algorithm for generalized geometric programming*, Applied Mathematics and Computation, Volume 168, Issue 1, 1 September 2005, Pages 722-737

[7] Costas D. Maranas and Christodoulos A. Floudas, *Global optimization in generalized geometric programming* Computers & Chemical Engineering, Volume 21, Issue 4, 20 December 1997, Pages 351-369

[8] Peiping Shen and Hongwei Jiao, *A new rectangle branch-and-pruning approach for generalized geometric programming*, Applied Mathematics and Computation, Volume 183, Issue 2, 15 December 2006, Pages 1027-1038

[9] Peiping Shen and Hongwei Jiao, *Global optimization of signomial geometric programming using linear relaxation*, Applied Mathematics and Computation, Volume 150, Issue 1, 27 February 2004, Pages 99-114.

XOR as MILP: alternative modelings

Cássia F. Novello[1], Shimon Cohen[2], Nelson Maculan[1], Felipe M. G. França [1], Gadi Pinkas[2], Priscila M. V. Lima[3]

[1]*Universidade Federal do Rio de Janeiro, PESC–COPPE, Rio de Janeiro, Brazil* cassiafn@cos.ufrj.br, maculan@cos.ufrj.br, felipe@cos.ufrj.br

[2]*Center for Academic Studies, Israel* shamon51@gmail.com, gadip@mla.ac.il

[3]*Universidade Federal Rural do Rio de Janeiro, Seropédica, Brazil* priscilamvl@ufrrj.br

Abstract SATyrus is a MaxSAT based environment to model binary optimization problems, having SATish as modeling language. SATish uses propositional logic to declaratively specify the constraints that define viable solutions to a target problem; the objective function is specified in the same way. This allows one to describe larger problems by modeling and combining their subproblems. XOR and 2-dimensional XOR are very common subproblems of larger ones, such as N-Queens and TSP. We investigate four approaches for modeling 2-XOR using binary variables: CrossWTA, LogWTA, CrossLogWTA and TreeWTA. Also, a comparison of the four approaches concerning number of variables and number of constraints is offered.

Keywords: binary integer programming, exclusive-Or constraints, SATyrus, weighted MaxSAT

1. Introduction

The classic exclusive-or (XOR) problem may be described as choosing exactly one out of n variables. The two-dimensional exclusive-Or (2-XOR) consists in choosing exactly n out of $(n \times n)$ variables, in such a way that no two variables share the same value for n dimensions. Another way to see it is placing n pieces on a $(n \times n)$ grid, so that they dont́ share any rows or columns. There are $n!$ distinct viable solutions for this problem, since there are n possibilities for the first choice, $n-1$ for the second, $n-2$ for the third and so on, among the $2^{(n \times n)}$ unrestricted possibilities. XOR and 2-XOR are very common subproblems of larger ones. The N-Queens problem, for instance, can be based on 2-XOR by adding diagonals' constraints to it. The TSP problem may also be considered a variant of 2-XOR that associate costs to paths between two neighboring cities. A viable solution is defined by an association of a position to each and every (distinct) city in the tour.

In this work, we investigate four 2-XOR modeling alternatives: CrossWTA [4] [8], LogWTA [7], TreeWTA [8] and CrossLogWTA, the first herein named and the latter introduced here. The four models are expressed as propositional constraints in SATish, the input language of the SATyrus compiler [6], which produces an algebraic expression that can be solved by a MILP solver.

2. SATyrus

SATyrus is a satisfiability-based environment to model optimization problems, having SATish as modeling language. The process begins by fully specifying the set of viable candidate solutions to a problem and its cost function using well-formed formulae in propositional logic. Then, the SATyrus environment reads the SATish model and acts as a compiler.

The compilation step maps logical operators into algebraic expressions, and propositional variables into 0–1 variables of a single energy function [8] [5]. The energy function combines all constraints and the cost function; solving it means finding its minimum. SATyrus generates either AMPL [1] or Mosel [9] as output to be fed to solvers. SATyrus2 [3] and BonMin solver [2] have been used. The following rules, that map logical formulae into algebraic function H, constitute the basis of the composition of the Energy function to be minimized [5] [6] [8]:

- $H(true) = 1$
- $H(false) = 0$
- $H(\neg p) = 1 - H(p)$ (when p is an atomic proposition)
- $H(p \wedge q) = H(p) * H(q)$
- $H(p \vee q) = H(p) + H(q) - H(p \wedge q)$

A SATish model has three elements: structures, constraints and penalty levels. Structures allow identifiers to be indexed. Constraints are well-formed formulas in propositional logic that use the elements of structures as their propositional variables. Each constraint is associated to a penalty level in order to guarantee that smaller energy values represent better solutions to the problem.

The language has two operators that allow concise modeling for similar constraints: *forall* and *exists*. Both of them make use of indeces intervals in order to express which constraints should be valid. *Forall* indicates that constraints must be generated by varying all indexes specified in the interval(s). *Exists* indicates that at least one constraint is valid among the ones resulting by the varying the indeces in the specified interval(s).

3. XOR models

Conventional WTA (Winner-Takes-All) constraints are modeled as $F_i \rightarrow \neg F_j$, where $i \neq j$, and F is any disjunction of propositional literals. It means that, we need $n(n-1)/2$ WTA constraints for a n-variable XOR problem, and $n^3 - n^2$ constraints for a n-variable 2-XOR problem. The idea is to reduce that by introducing new kinds of constraints, other than WTA. This may benefit large instances of problems that have XOR and 2-XOR as a subproblem, including TSP.

3.1 CrossWTA

Let $N_{n \times n}$ be a matrix of possible solutions, where rows represent each city and columns represent each position. The chosen element is valued 1 and the not-chosen are valued 0. Our goal is to find a solution where each row and each column have only one 1-valued element.

The constraints are: (i) avoid null solutions, which is, 0 in all positions of the solution matrix (n^2 constraints); (ii) each city must have only one position associated to it (n^3); (iii) each position must have one city associated to it (n^3). Contrainst of type (iii) are written in SATish as:

```
intgroup wta:
  forall{i,j,k} where i in (1,n), j in (1,n), k in (1,n) and i < k:
    pos[i][j] -> not pos[k][j];
```

CrossWTA has space complexity of $O(n^2)$, and $O(n^3)$ constraints. This is the most intuitive approach to solving this problem.

3.2 LogWTA

In this approach, each position is associated to a binary number. This way, the structure of propositional variables is $N_{(n \times \log n)}$. In case $n < 2^{\log n}$, the unused binary representations must be set to non-viable solutions. The possibility of the second index, which is binarily represented, be 0, must also be set as non-viable solution since 0 is not a possible position in the path.

The constraints are: (i) if one city is associated to one binary representation, no other city must be associated to it ($n^2 * n$); (ii) unused binary representations must not be associated to any city ($n * (2^{\lfloor(\log n)+1\rfloor} - n)$). Contraints of types (i) and (ii) are written in SATish as:

```
intgroup int1:
  forall {i,j} where i in (1,n), j in (1,n) and i!=j:
    (not logpos[i][3] and not logpos[i][2] and logpos[i][1]) ->
     not (not logpos[j][3] and not logpos[j][2] and logpos[j][1]);

intgroup int1:
  forall {i} where i in (1,num):
    not (not logpos[i][3] and not logpos[i][2] and not logpos[i][1]);
```

Note that, in the example above, we assume that $n \leq 2^3$. We only presented the constraints associated to the binary representation values of 0 and 1 to the second index. The full model must include all unused binary representations (0 and $n+1$ to $2^{\lfloor(\log n)+1\rfloor}$) and all used binary representations (1 to n). LogWTA has space complexity of $n \log n$ and $n^3 + n * (2^{\lfloor(\log n)+1\rfloor} - n)$ constraints.

3.3 CrossLogWTA

CrossLogWTA associates both positions and cities to binary representations. The structure of possible solutions is $N_{(2 \times n \times \log n)}$. The first index distinguishes cities (1) from positions (2).

The constraints are: (i) when the first index value is 1 (referring to city), and the second is fixed to, say, z, the value composed by varying the third index consists of the binary representation of the position of city z in the tour; likewise, when the first index value is 2 (referring to position), and the second is fixed to, say, w, the value composed by varying the third index consists of the binary representation of the city that occupies position w in the tour ($2n^2$); (ii) unused binary representations must not be associated to any city ($2n * (2^{\lfloor(\log n)+1\rfloor} - n)$).

Contraints of type (i), for $n < 2^3$, and only the ones associated to the pair (city 1, position 2), are written in SATish as:

```
intgroup int1:
  not logpos[1][2][3] and not logpos[1][2][2] and logpos[1][2][1] ->
  (not logpos[2][1][3] and logpos[2][1][2] and not logpos[2][1][1]);
```

The full model must include all the other combinations. Unused binary representations are the same as in the LogWTA approach. CrossLogWTA has space complexity of $2n \log n$ and $n^2 + n * (2^{\lfloor(\log n)+1\rfloor} - n)$ constraints.

3.4 TreeWTA

This approach models 1-XOR, associating the possible choices to a binary tree. The root represents all possible positions, from 1 to n. The second level splits the choices in two groups: from 1 to $n/2$, and from $n/2 + 1$ to n. The third level splits the choices in four groups, and so on, until the leaves level, where every node represents a single choice. WTA constraints ensure that just one node is activated at each level.

For 2-XOR, one must use n trees and associated WTA constraints. Let $N_{n,n*2-1}$ be the structure of possible solutions. The first index represents cities and the second index represents the binary tree, which will make sure that only one position is selected for each city.

The constraints are: (i) each activated (1 valued) parent has to have one activated child (n^2); (ii) the parent of an activated child has also to be activated (n^2); (iii) only one child of an activated parent is allowed to be activated (siblings WTA, $2n^2$); (iv) each city must have only one position associated to it (WTA, n^3).

Constraints of types (i) and (ii) are written in SATish as:

```
intgroup int1:
  forall {i,j} where i in (1,num), j in (1,num-1):
    p[i][j] -> p[i][j*2] or p[i][(j*2)+1];

intgroup int1:
  forall {i,j} where i in (1,num), j in (1,num-1):
    p[i][j*2] or p[i][(j*2)+1] ->  p[i][j];
```

TreeWTA has a structure size of $n * (2n - 1)$ and $(3n^3 + 5n^2 - 6n)/2$ associated constraints.

4. Discussion

Table 1 summarizes the space costs of the four 2-XOR approaches. CrossLogWTA is the approach with the smallest number of constraints and it's structure size is still smaller than CrossWTA, though twice the size of LogWTA. TreeWTA needs more space than the other approaches, but the number of constraints is still competitive with CrossWTA.

Ongoing work uses BonMin to evaluate the influence of each approach in the search space. At the present moment, only small instances were tested. Quantitative results on solving time are expected to follow the costs shown in Table 1.

Table 1. Comparing 2-XOR modeling approaches

Approach	*Number of variables*	*Number of constraints*
CrossWTA	n^2	$n^3 - n^2$
LogWTA	$n \log n$	$n^3 + n * (2^{\lfloor(\log n)+1\rfloor} - n)$
CrossLogWTA	$2n \log n$	$n^2 + 2n * (2^{\lfloor(\log n)+1\rfloor} - n)$
TreeWTA	$2n^2 - n$	$(3n^3 + 5n^2 - 6n)/2$

References

[1] AMPL Modeling Language for Mathematical Programming, http://www.ampl.com/.

[2] Basic Open-source Nonlinear Mixed INteger programming, http://www.coin-or.org/Bonmin/

[3] Monteiro, Bruno França. (2010) "SATyrus2: Compiling Logic Reasoning Specifications," M.Sc. Thesis, COPPE–Universidade Federal do Rio de Janeiro.

[4] Hopfield, John J., and Tank, David W. (1985) "Neural Computation of Decisions in Optimization Problems", Biological Cybernetics, **52**, 141-152.

[5] Lima, Priscila M. V., França, Felipe M. G. and Morveli-Espinosa, M. Mariela M. (2007). "Logic as Energy: A SAT-Based Approach", Proc. of BVAI 2007, LNCS, **4729**, 458-467.

[6] Lima, Priscila M. V. et al (2008) "Logical Reasoning via Satisfiability Mapped into Energy Functions", International Journal of Pattern Recognition and Artificial Intelligence, **22**(5), 1031-1043.

[7] Pinkas, Gadi, Lima, Priscila M. V. and Cohen, Shimon (2011) "Compact Crossbar Variable Binding for Neuro-Symbolic Computation" (position paper), Proc. of NeSy'11, Barcelona.

[8] Pinkas, Gadi. (1991) "Symmetric neural networks and propositional logic satisfiability", Neural Computation **3**(2), 282-291.

[9] Xpress Optimization Suite, http://www.fico.com/en/Products/DMTools/Pages/FICO-Xpress-Optimization-Suite.aspx.

A new method for approximating the Pareto-front of multiobjective optimization problems: application to a planar bi-objective facility location problem*

Juana L. Redondo[1], José Fernández[2], José D. Álvarez [3], Aránzazu G. Arrondo[2] and Pilar M. Ortigosa[4]

[1] *Dpt. of Computer Architecture and Technology, University of Granada, Granada, Spain*
jlredondo@atc.ugr.es

[2] *Dpt. of Statistics and Operations Research, University of Murcia, Murcia, Spain*
agarrondo@um.es, josefdez@um.es

[3] *Dpt. of Computer Sciences and Languages, University of Almería, ceiA3, Almería, Spain*
jhervas@ual.es

[4] *Dpt. of Computer Architecture and Electronics, University of Almería, ceiA3, Almería, Spain*
ortigosa@ual.es

Abstract A bi-objective competitive facility location and design problem is considered. It has been previously tackled through exact general methods, but they require high computational effort. In this work, we propose a new multi-objetive optimization heuristic algorithm, which deals with the problem at hand in a fast and efficient way. It combines ideas from different multi-objective and single-objective optimization evolutionary algorithms, although it also incorporates new devices which help to reduce the computational requirements, and also to improve the quality of the provided solutions. A comprehensive computational study shows that the heuristic method is competitive, being able to reduce, in average, the computing time of the exact method by approximately 99%, and offering good quality in the final solutions.

Keywords: Bi-objective location problem, multi-objective optimization algorithm, Pareto front, Efficient set.

1. Dealing with inner and outer competition: a planar bi-objective location problem

Competitive location deals with the problem of locating facilities to provide a service (or goods) to the customers (or consumers) of a given geographical area where other competing facilities offering the same service are already present or will enter the market in the near future. Many competitive location models are available in literature. However, the literature on multi-objective competitive location models is rather scarce. This is in part due to the fact that single-objective competitive location problems are difficult to solve, and considering more than one objective makes the problem nearly intractable.

In this paper, we revisit the bi-objective problem described in [2]. A franchise wants to increase its presence in a given geographical region by opening one new facility. Both the franchisor (the owner of the franchise) and the franchisee (the actual owner of the new facility

*This work has been funded by grants from the Spanish Ministry of Science and Innovation (TIN2008-01117, ECO2011-24927), Junta de Andalucía (P08-TIC-3518 and P10-TIC-6002) and Fundación Séneca (The Agency of Science and Technology of the Region of Murcia, 00003/CS/10 and 15254/PI/10), in part financed by the European Regional Development Fund (ERDF). Juana López Redondo is a fellow of the Spanish 'Juan de la Cierva' contract program.
Corresponding author: P.M. Ortigosa.

to be opened) have the same objective: maximizing their own profit. However, the maximization of the profit obtained by the franchisor is in conflict with the maximization of the profit obtained by the franchisee.

In that paper, an interval branch-and-bound method was proposed to solve the corresponding bi-objetive problem, but it was time consuming and had large memory requirements. (Meta)heuristic algorithms have proved to be good tools to overcome those drawbacks, while obtaining good approximations of the final solutions. In this paper, we present a new Fast and Efficienct Multi-Objective Evolutionary Algorithm (FEMOEA) whose aim is to obtain a good approximation of the Pareto front as fast as possible. In the computational studies we compare it with the inteval branch-and-bound method iB&B (see [2]).

2. A new method for approximating the Pareto-front

FEMOEA is an evolutionary algorithm initially devised to solve any multi-objective optimization problem. Its main aim is to provide a set of well-distributed and non-dominated solutions as fast as possible.

The most important concept in FEMOEA is that of subpopulation. A subpopulation is defined by a center and a radius. The center is a solution and the radius is a positive number, which determines the subregion of the search space covered by that subpopulation. The radius of a subpopulation is given by a decreasing exponential function, varying from R_1 to R_L (input parameters), which are the given largest and smallest radii, respectively. For a detailed description on how to compute the radius see [4].

Apart from the center and the radius, a subpopulation has two attributes which are related to the objective space: the non-domination rank (d_{rank}) and the crowding distance (c_{dist}) (see [1] for an in-detail description of these values). The former indicates the number of subpopulations which dominates that particular subpopulation, whereas the latter is an estimation of the density of solutions surrounding a particular solution in a population.

During the process, two lists of subpopulations are kept by FEMOEA, each with a maximum size M, (another input parameter). M refers to the desired number of solutions in the final Pareto front. The first list, named *population_list*, is composed of M diverse subpopulations with different attributes, i.e. various radii, non-domination ranks and crowding distances. FEMOEA is in fact a method for managing this list (i.e. creating, deleting and improving subpopulations). The second list, called *external_list*, can be understood as a deposit to keep non-dominated solutions. This external archive is also used in other algorithms described in literature [3].

Initially, a set of diverse subpopulations is created in the initialization phase. After this procedure, the FEMOEA main loop starts, which basically consists of three procedures: creating, improving and selecting subpopulations. This loop is executed until a stopping condition is fulfilled. For the problem at hand, the algorithm stops if either a considerable improvement has not been obtained among consecutive Pareto fronts (placed in *external_list*) or a maximum number of levels (cycles or generations) L (an input paramter) is achieved.

3. Computational studies

All the computational studies in this paper have been run in the supercomputer Ben Arabi of the Supercomputing Center of Murcia, Spain, in particular, in Arabi, which is a Blade Cluster with 816 cores, organized in 32 nodes with 16GB of memory each, and 70 nodes with 8GB (102 nodes altogether). Each node has 8 cores, divided into 2 Intel Xeon Quad Core (E5450) to 3.0 GHz.

In order to have an overall view of the performance of the algorithms, different types of problems have been generated, varying the number n of demand points, the number m of existing facilities and the number k of those facilities belonging to the chain. For $n = 25, 50$ demand points the settings used were $(m = 2, k = 1)$, $(m = 5, k = 1, 2)$ and $(m = 10, k = 2, 4)$. For every setting, 10 instances were generated by randomly choosing the parameters of the

Table 1. Hypervolume and computing time for problems with setting 25-5-2.

T	eps	$Av(T)_{200}$	$Av(T)_{400}$	$[lowH, uppH]$	$Av(Hyper)_{200}$	$Av(Hyper)_{400}$
209		262	537	[146.317,146.532]	146.320	146.332
489197	(.05)	347	1144	[1.326,1.328]	1.323	1.326
508028		273	596	[3.157,3.159]	3.146	3.158
97404		310	730	[112.931,113.380]	112.728	112.956
551537	(.05)	308	910	[1.751,1.764]	1.754	1.759
4536		274	562	[553.147,554.340]	553.151	553.163
569547		309	756	[2.282,2.283]	2.278	2.282
81687		266	596	[428.519,429.229]	427.964	428.800
389738	(.04)	338	1072	[1.342,1.344]	1.340	1.343
281464	(.04)	339	989	[1.762,1.763]	1.758	1.761
297340		302.6	789.2	[125.253,125.512]	125.176	125.288

problems uniformly within pre-defined intervals (see [5]). The searching space proposed in [5] has also been considered here for every problem.

As a general rule, the algorithm iB&B has been executed considering a tolerance of $eps = 0.03$ (the maximum width of a box in the solution list), which is not a negligible value. Even so, the algorithm ran out of memory when trying to solve several instances. In each of those cases, the value of eps was progressively increased until the algorithm was able to solve that particular problem. Regarding FEMOEA, and after extensive experiments, we found that a good parameter setting to deal with the current multi-objetive optimization problem is: $L = 30$ and $R_L = 5e - 03$. The parameter R_1 coincides with the diameter of search space. Furthermore, FEMOEA has been executed twice varying the number of points in the Pareto front, $M = 200$ or 400.

To measure the performance of FEMOEA, two main aspects are under consideration, that of effectiveness and that of efficiency. We start by saying that for stochastic algorithms, performance indicator values are also stochastic. For each random indicator, we approximate the expected value by taking the average over 5 runs.

As an effectiveness metric, we check whether the heuristic algorithm has successfully found an approximation of the Pareto front. We say so when both the objective function values of the points in the *external_list* are included in the corresponding intervals provided by the iB&B method, and the points themselves are included in the corresponding solution boxes offered by iB&B. Additionally, for measuring the goodness of an approximation to the Pareto front, the so-called hypervolume measure $Hyper$ has also been computed [6].

To measure the efficiency of the algorithms, one tries to compute the effort used to obtain the final result. Here, we measure the computing time to reach the result for iB&B (T), and the average computing time in the five runs $Av(T)$ for the algorithm FEMOEA.

Table 1 summarizes the obtained results for a set of 10 instances with settings 25-5-2. At the end of the table, average values for the 10 problems have been computed. In that table, the computing time and the hypervolume metric are shown for both iB&B and FEMOEA. The results for the rest of settings are not shown due to the lack of space, but similar conclusions can be inferred.

As can be observed in Table 1, the iB&B algorithm is very erratic regarding computing time. Additionally, it is difficult for a given problem to determine the tolerance that, a priori, will be suitable to execute the algorithm. Of course, a large value of eps can be considered, but it will affect the quality of the obtained solution (the larger the value of eps, the greater the intervals containing the exact Pareto front). For the cases where a different value of $eps = 0.03$ has been considered, this has been noted on the table, including the new value after the computing time spent by iB&B in brackets (see column eps). On the contrary, the evolutive algorithm seems to be more regular, it always spends similar computing times for instances with the same settings. In average, FEMOEA has reduced the computing time by more than 99% for problems with settings 25-5-2, and this considering a number $M = 400$ points in the Pareto front.

It is worth mentioning that FEMOEA approximates the Pareto front with 100% success for all the problems (for both $M = 200$ and $M = 400$), i.e. its solutions are always included in the intervals provided by the iB&B algorithm. Notice that FEMOEA increases its $Hyper$ value as the number of points in the Pareto front increases. In this sense, we should keep track of the number of generated points M in the set approximating the Pareto front, since theoretically speaking, higher values of $Hyper$ represent better approximations. As can be observed in Table 1, the hypervolume covered by the heuristic with 400 points is always included in the interval $[lowH, uppH]$, obtained from the lower-left and upper-right hand corners of the boxes provided by iB&B. On the other hand, with 200 points in the Pareto front, the hypervolume is smaller than the lower limit for 3 out of 10, but notice the number of points used to compute the hypervolume with iB&B (i.e., the number of boxes on the solution list) is usually much larger than M.

4. Conclusions and lines for future research

In this work, a new multi-objective optimization algorithm, FEMOEA, has been proposed. Addtionally, it has been compared to an exact iB&B method when solving a particular problem. More precisely, a bi-objective competitive location and design problem has been selected as a way to link both methods. Results have shown that FEMOEA use of the computing resources is more efficient. Moreover, the provided solutions by FEMOEA are competitive with respect to the ones given by iB&B: on the one hand, the solutions obtained by the heuristic algorithm are always included in the iB&B intervals and, on the other hand, with $M = 400$ they can cover practically all the area of the optimum Pareto front (the hypervolumes obtained by FEMOEA are always included in the hypervolume intervals of iB&B).

In the future, we plan to compare the FEMOEA algorithm with other heuristic algorithms devised to cope with multi-objective problems. In particular, we will compare it with two standards widely referenced in literature, i.e. NSGA-II and SPEA2.

References

[1] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, 2002.

[2] J. Fernández and B. Tóth. Obtaining the efficient set of nonlinear biobjective optimization problems via interval branch-and-bound method. *Computational Optimization and Applications*, 42(3):393–419, 2009.

[3] A.J. Nebro, F. Luna, E. Alba, B. Dorronsoro, J.J. Durillo, and A. Beham. AbYSS: Adapting scatter search to multiobjective optimization. *IEEE Transactions on Evolutionary Computation*, 12(4):439–457, 2008.

[4] P.M. Ortigosa, I. García, and M. Jelasity. Reliability and performance of UEGO, a clustering-based global optimizer. *Journal of Global Optimization*, 19(3):265–289, 2001.

[5] J.L. Redondo, J. Fernández, I. García, and P.M. Ortigosa. A robust and efficient global optimization algorithm for planar competitive location problems. *Annals of Operations Research*, 167(1):87–106, 2009.

[6] E. Zitzler. Evolutionary algorithms for multiobjective optimization:methods and applications. Isbn 3-8265-6831-1, Swiss Federal Institute of Technology (ETH) Zurich, Shaker Verlag, Germany, 1999.

AFSFilter: Artificial Fish Swarm Filter-Based Algorithm for Global Optimization*

Ana Maria A.C. Rocha,[1] M. Fernanda P. Costa,[2] and Edite M.G.P. Fernandes[1]

[1] *Algoritmi R&D Centre, U Minho, 4710-057 Braga, Portugal,* {arocha,emgpf}@dps.uminho.pt

[2] *Mathematics R&D Centre, U Minho, Braga, Portugal,* mfc@math.uminho.pt

Abstract A fish swarm intelligence algorithm based on the filter set concept to accept, at each iteration, a population of trial solutions whenever they improve constraint violation or objective function, relative to the current solutions, is proposed for constrained global continuous optimization problems. Preliminary numerical results are provided.

Keywords: Global optimization, Fish Swarm Intelligence, Filter method

1. Introduction

The problem to be addressed in this paper has the form

$$\min_{x\in\Omega} f(x)\,,\ \text{subject to}\ g_j(x) \leq 0\,, j = 1,\ldots,m \tag{1}$$

where at least one of the functions $f, g_j : \mathbf{R}^n \to \mathbf{R}$, is nonlinear and $\Omega = \{x \in \mathbf{R}^n : l_k \leq x_k \leq u_k, k = 1,\ldots,n\}$. Problems with equality constraints can be reformulated in the above form using a small tolerance. When convexity is not assumed, problem (1) may have multiple optimal solutions in Ω. This paper aims at proposing a stochastic method to compute a global solution of (1). From the class of stochastic methods, swarm intelligence algorithms have shown to be effective in reaching a global solution. Recent studies involving the artificial fish swarm (AFS) algorithm show that highly accurate solutions may be obtained with reduced computational costs [6, 7]. Although penalty function methods are probably the most known constraint handling techniques, a penalty function depends, in general, on a penalty parameter. Unfortunately, the performance of these algorithms depends strongly on the values set to the penalty parameter throughout the iterative procedure. Adaptive penalties [9] and augmented Lagrangian methodologies [2, 8] are just recent strategies to overcome partially this issue. The separate use of objective function and constraint violation with the nondominance concept from multiobjective programming, for example in [1], avoids the use of penalty parameters. Fletcher and Leyffer [4] proposed a filter method as an alternative to penalty functions to guarantee convergence to optimizers in nonlinear constrained optimization. This technique incorporates the concept of nondominance to build a filter set that is able to accept solutions if they improve either the objective function or the constraint violation, instead of a linear combination of those two measures.

In this paper, an artificial fish swarm filter-based algorithm, hereafter denoted by AFSFilter, for nonlinear constrained global optimization problems is proposed. Results from preliminary numerical experiments are provided.

*The financial support from FEDER COMPETE (Programa Operacional Fatores de Competitividade / Operational Programme Thematic Factors of Competitiveness) and FCT (Fundação para a Ciência e a Tecnologia / Portuguese Foundation for Science and Technology) Project FCOMP-01-0124-FEDER-022674 is gratefully acknowledged.

2. Artificial Fish Swarm Algorithm

Here is some notation used in the paper. Constraint violation of a point x is measured by the function

$$\theta(x) = \sum_{j=1}^{m} \max\{0, g_j(x)\} + \sum_{k=1}^{n} \left(\max\{0, x_k - u_k\} + \max\{0, l_k - x_k\}\right), \tag{2}$$

$x^i \in \mathbf{R}^n$ represents the ith point of a population of size p, and x^{best} is the best point in the population. Pairwise comparisons in the population use the following concept: between two points x^i and x^j, x^i is better than x^j if the following condition holds:

$$\theta(x^i) < \theta(x^j) \text{ or } \left(\theta(x^i) = \theta(x^j) \text{ and } f(x^i) < f(x^j)\right). \tag{3}$$

In the AFS algorithm, the initial population of p points is randomly generated inside the set Ω. A crucial quantity of the algorithm is the 'visual scope' of a point, say x^i. This is defined as the closed neighborhood with center x^i and radius equal to a positive quantity $v = \varsigma \max_{k\in\{1,\dots,n\}}(u_k - l_k)$, where ς is a positive visual parameter. Let n^i be the number of points in its 'visual scope' ($n^i < p$). If the condition $n^i/p \leq \kappa$ holds, where $\kappa \in (0, 1]$ is the crowd parameter, the 'visual scope' of x^i is said to be not crowded. Depending on the relative position of the points in the population, one of the following three situations occurs.

1. When $n^i = 0$, the 'visual scope' is empty, and the point x^i, with no other points in its neighborhood to follow, has a random behavior. Here, a point is randomly generated in the search space, x^r, and a movement is tried along the direction defined by $d = x^r - x^i$.
2. When the 'visual scope' is crowded, the point has some difficulty in following any particular point, and starts by following a searching behavior. A point inside the 'visual scope' is randomly generated, x^s, and a movement towards it is carried out if x^s is better than x^i (see condition (3)); otherwise, x^i moves according to a random behavior.
3. When the 'visual scope' is not crowded, the point firstly tries the chasing behavior moving towards the best point inside the 'visual scope', denoted by $x^{\min}$, if this is better than x^i, thus being the direction of movement $d = x^{\min} - x^i$. Otherwise, the point tries to follow the swarming behavior moving towards the central point, c, of the 'visual scope'. However, if c is not better than x^i, the point tries to follow a searching behavior; and if that randomly generated point x^s is not better than x^i, the point follows a random behavior.

The algorithm also implements an elitism procedure in the sense that the best point of the population is not moved and is maintained throughout the iterative process. For each current point x^i of the population, the trial point y^i is generated according to a direction d and a step size $\alpha \in (0, 1]$

$$y^i = x^i + \alpha\, d, \; i = 1, \dots, p \text{ and } i \neq best. \tag{4}$$

The procedure that decides if the trial solution is to be accepted and replaces the current point is a filter method combined with a backtracking line search, as described in the next section. The algorithm terminates with a successful run if the stopping conditions

$$\left| f(x^{best}) - f^* \right| \leq \epsilon_1 |f^*| + \epsilon_2 \text{ and } \theta(x^{best}) \leq \epsilon_2$$

are satisfied, for small positive tolerances ϵ_1, ϵ_2; otherwise $x^i \leftarrow y^i$ for all $i \neq best$ and the procedure is repeated until a maximum number of iterations is reached, where f^* is the best global solution of the problem available in the literature.

3. The Implemented Filter Methodology

This section briefly describes the filter methodology that aims at deciding which trial solution is to be accepted in the sequence of Eq. (4). The herein proposed AFSFilter method uses the filter set concept, as outlined in [3, 4], with the ability of exploring both feasible and infeasible regions, and building a filter set that is able to accept a trial point if it improves either the objective function or the constraint violation, relative to the current point. Filter-based algorithms treat the optimization problem as a biobjective problem aiming to minimize both the objective function and the nonnegative constraint violation function (2).

After a search direction d has been computed, A decreasing sequence of step sizes $\{\alpha_j\}$ with $\lim_j \alpha_j = 0$ is tried, until a set of acceptance conditions are satisfied. This j denotes the iteration counter for the inner loop. A trial step size α_j might be accepted if the corresponding trial point $y^i = x^i + \alpha_j\, d$ is acceptable by the filter. We only require an improvement in θ or in f, relative to the current point x^i, to consider the trial point y^i, in Eq. (4), to be acceptable, as shown:

$$\theta(y^i) < \theta(x^i) \ \text{ or } \ f(y^i) < f(x^i). \tag{5}$$

However, when x^i is (almost) feasible, the trial point y^i has to satisfy only the condition of simple reduction on f:

$$f(y^i) < f(x^i) \tag{6}$$

to be acceptable. To prevent cycling between points that improve either θ or f, at each iteration, the algorithm maintains a filter $\mathcal{F}$ which is defined as a finite set of entries $(\theta(x^j), f(x^j))$ that correspond to a collection of infeasible solutions x^j such that no filter entry is dominated by any of the others in the filter. During the backtracking line search procedure, the y^i is acceptable only if $(\theta(y^i), f(y^i)) \notin \mathcal{F}$. (Only solutions that are not dominated by any entry in the filter might be accepted.)

The filter is initialized with entries (θ, f) that satisfy $\theta \geq \theta_{\max}$, where $\theta_{\max} > 0$ is the upper bound on θ. Furthermore, the filter is augmented whenever y^i is accepted because condition (5) is satisfied. When it is not possible to find a point y^i with a step size $\alpha_j > \alpha_{\min} > 0$ that satisfies one of the conditions (5) or (6), a restoration phase is invoked. In this phase, the algorithm performs a coordinate random local search around the best point, with length $10^{-3} \max_k\{u_k - l_k\}$, to find a point inside $[l, u]$ that is acceptable to the filter. If no such point is found, the algorithm maintains the current point to the next iteration.

4. Preliminary Results

Table 1 contains the numerical results of our preliminary experiments with the AFSFilter method. Three well-known engineering design problems are used in the comparison with the results obtained by the Filter Simulated Annealing (SA) method proposed in [5]. The welded 'beam' design problem has four design variables and seven inequality constraints, the tension/compression 'spring' design problem has three continuous variables and four inequality constraints and the cylindrical 'vessel' design problem has four design variables (two of them are multiples of 0.0625) and four inequality constraints [5]. The size of the population is set to $p = 5n$ and the algorithm was allowed to run for a maximum of 200 iterations. A comparison with the pattern search hybrid GA from MatLab$^{\text{TM}}$ (with the tournament selection option to handle constraints) is also provided. A set of four small but difficult problems, with $n = 2$, selected from [2] and a technical report from the same authors[1], is also tested and the results are reported in the table for comparison with GA. The size of the population is set to $p = 20$ and a maximum of 50 iterations is allowed.

Each problem was run 30 times and the results reported in the table are: 'f^{best}', the best solution obtained during all the runs and 'avg.n.f.e.', the average number of function evaluations (from the 30 runs). Other parameter values are set as follows: $\epsilon_1 = 10^{-6}, \epsilon_2 = 10^{-8}$, $\theta_{\max} = 10^4$, $\alpha_{\min} = 10^{-3}$ and ς is set to one and is reduced every p iterations until it reaches 0.1. We may conclude that the proposed AFSFilter method is effective in reaching a global optimal solution with reasonable computational costs. New developments with further experimentation will follow.

[1] Technical report MCDO-051015 (2005) in http://www.ime.usp.br/∼egbirgin/.

Table 1. Comparison of AFSFilter with Filter SA in [5] and Hybrid GA in MatLab.

Prob.		AFSFilter		Filter SA in [5]		Hybrid GA from MatLab	
	f^*	f^{best}	avg.n.f.e.	f^{best}	avg.n.f.e.	f^{best}	avg.n.f.e.
beam	2.38081	2.3866641	65 687	2.381065	56 243	2.5526753	168 119
spring	0.012664	0.0126653	35 929	0.0126653	49 531	0.0126663	3 480
vessel	5854.738	5868.974	45 283	5868.765	108 883	5859.977	21 289
Example 1 in [2]	-1.0000000	-0.9983634	7 906	n.a.	n.a.	-0.9999910	4 027
Example 3 in [2]	1.0000000	1.0000013	10 849	n.a.	n.a.	0.9999987	5 366
Example 5 in [2]	-2.0000000	-2.0000330	8 180	n.a.	n.a.	-2.0000043	51 067
Problem 6†	-9.4772944	-9.4772842	9 458	n.a.	n.a.	-3.2883714‡	5 108

† Technical report MCDO-051015 (2005) in http://www.ime.usp.br/∼egbirgin/, pre-print to [2]
‡ This is a local optimal solution

References

[1] Ali, M.M., Golalikhani, M. (2010). An electromagnetism-like method for nonlinearly constrained global optimization. Computers and Mathematics with Applications, 60, 2279–2285

[2] Andreani, R., Birgin, E.G., Martinez, J.M., Schuverdt, M.L. (2007). On augmented Lagrangian methods with general lower-level constraints. SIAM Journal on Optimization, 18(4), 1286–1309

[3] Audet, C., Dennis, Jr., J.E. (2004). A pattern search filter method for nonlinear programming without derivatives. SIAM Journal on Optimization, 14(4) 980–1010

[4] Fletcher, R., Leyffer, S. (2002). Nonlinear programming without a penalty function. Mathematical Programming, 91, 239–269

[5] Hedar, A.-R., Fukushima, M. (2006). Derivative-free filter simulated annealing method for constrained continuous global optimization. Journal of Global Optimization, 35, 521–549

[6] Gao, X.Z., Wu, Y., Zenger, K., Huang, X. (2010). A knowledge-based artificial fish-swarm algorithm. 13th IEEE International Conference on Computational Science and Engineering, 327–332

[7] Rocha, A.M.A.C., Fernandes, E.M.G.P., Martins, T.F.M.C. (2011). Novel fish swarm heuristics for bound constrained global optimzation problems. ICCSA 2011, Part III, Lecture Notes in Computer Science 6784, B. Murgante et al. (Eds.) 185–199

[8] Rocha, A.M.A.C., Martins, T.F.M.C., Fernandes, E.M.G.P. (2011). An augmented Lagrangian fish swarm based method for global optimization. Journal of Computational and Applied Mathematics, 235(16) 4611–4620

[9] Silva, E.K., Barbosa, H.J.C., Lemonge, A.C.C. (2011). An adaptive constraint handling technique for differential evolution with dynamic use of variants in engineering optimization. Optimization and Engineering, 12, 31–54

Global and local optimization approaches for channel assignment in wireless networks*

Rosiane de F. Rodrigues,[1,2] Bruno Raphael C. Dias,[2] Aldemir M. Oliveira,[3] and Nelson Maculan[3]

[1] *Instituto de Computação, UFAM, Brazil* rosiane@icomp.ufam.edu.br

[2] *Programa de Pós-Graduação em Informática - PPGI/IComp, UFAM, Brazil,* bruno.dias@icomp.ufam.edu.br

[3] *Programa de Engenharia de Sistemas e Computação - PESC/COPPE, UFRJ, Brazil,* maculan@cos.ufrj.br

Abstract This paper presents a correlation between channel assignment in wireless networks and coloring problems in graphs and scheduling problems in parallel machines, resulting in more representative models. Integer linear programming formulations, and algorithmic strategies based on global and local optimization are applied, such as a branch-and-cut method and heuristics involving local search. Benchmark instances from the literature are used to validate the models and methods proposed, as well as generated instances simulating specific scenarios.

Keywords: algorithms, branch-and-cut, combinatorial optimization, global and local search, graph theory, integer linear programming

1. Introduction

The fact that wireless communication is growing at exponential scale, makes necessary rigorous planning of how to increase the capacity of existing networks, since the availability of channels for use by operators is regulated by government agencies that conduct auctions of frequency bands for private use. Optimize the use according to the current scenario, propose new scenarios for distribution and manipulation of frequency bands, as well as adopt new technologies and models, are part of the process of research and development in this area. In the current scenario of cellular wireless networks, broadcast frequencies are grouped into bands and are usually discretized into a set of channels. Thus, each base station receives a share of the total number of channels available to the entire system. Therefore, it is important to establish a strategy to reduce the total use of the available frequencies (for example, to be able to further expand in the future). Unfortunately, when it comes to wireless communication, there is the problem of interference, a problem that represents the superposition of two or more electromagnetic waves at one point. And this happens when the transmitters are close to each other, causing failures in the communications established by them [1] [4] [3].

This paper presents a correlation between such problems and problems in graphs and scheduling, resulting in more robust integer linear programming models, and so, algorithmic strategies to global and local optimization are applied, such as a branch-and-cut method by CPLEX tool and heuristics involving local search.

*Partially supported by "Conselho Nacional de Desenvolvimento Científico e Tecnológico" (CNPq), "Coordenação de Aperfeiçoamento de Pessoal de Nível Superior" (CAPES), and "Fundação Carlos Chagas Filho de Amparo à Pesquisa do Estado do Rio de Janeiro" (FAPERJ), brazilian agencies.

2. A MinMax Channel Assignment Problem

Given a set of n antennas (base stations) receiving channels, where each base station i has a demand of c_i channels. The k-th channel assignment to the i-th base station is denoted by $f_k i$. With this, it follows that for all pairs (i, j) of base stations, there is a value of d_{ij} (the distance between channels quoted above) where the following condition must be respected: $|f_{ki} - f_{mj}| \geq d_{ij}$. We want to minimize the largest channel used (to minimize the span of channels), a MinMax CAP (Channel Assignment Problem) [4]. This variant of channel allocation problem can be modeled as an IP model in bi-index binary variables x_{if}, where $x_{if} = 1$ if the channel $f \in F_v$ is assigned to the station i, and $x_{if} = 0$ otherwise. Moreover, binary variables y_f, where $y_f = 1$ if the channel $f \in F$ is allocated to a station, and $y_f = 0$ otherwise. And finally, z_{max} and z_{min}, represent the biggest and the lowest channels used in the frequency spectrum available. So, the formulation will be:

Minimizar $z_{max} - z_{min}$

Sujeito a:

$$\sum_{f \in F_i} x_{if} = c_i \quad (\forall i = 1, 2, ..., n) \tag{1}$$

$$x_{if} + x_{jg} = 1 \quad (\forall i, j = 1, 2, ..., n;\ f \in F_i;\ g \in F_j : |f - g| < d_{ij}) \tag{2}$$

$$x_{if} \leq y_f \quad (\forall i = 1, 2, ..., n;\ f \in F_i) \tag{3}$$

$$z_{max} \geq y_f \quad (\forall f \in F) \tag{4}$$

$$z_{min} \leq f_{max} - (f_{max} - f) y_f \quad (\forall f \in F) \tag{5}$$

$$x_{if} \in \{0, 1\} \ (\forall i = 1, 2, ..., n;\ f \in F_i) \tag{6}$$

$$y_f \in \{0, 1\} \quad (\forall f \in F) \tag{7}$$

An example is showed in Figure 1. In this case, the 6 base stations need only one channel each. The link between two base stations indicates that a base station may cause interference with each other if use close channels. The value of d_{ij} in each range indicates the distance that should exist between the channels of i and j stations. If is considered the distance $d_{ij} = 1$ for all values of i and j, only the stations that may interfere with each other should be different allocations of channels without need for an specific amount of distance. In this example, assuming then $d_{ij} = 1$, as previously mentioned, there is an optimal solution as shown in figure 1, where the number above the antenna indicates the channel allocated to it. It is observed that, even with 6 base stations, three channels are sufficient for the proposed network can run without noise. For each ERB i, there exists a set $f_i \subseteq N$ called the *domain frequency of* i containing the list of possible channels that can take i. For example, if $F_1 = \{1, 3, 4, 7\}$, is that the channels allocated for base station 1 may be $1, 3, 4$ or 7. Is denoted by F, then the set $F_1 \cup F_2 \cup ... \cup F_n$.

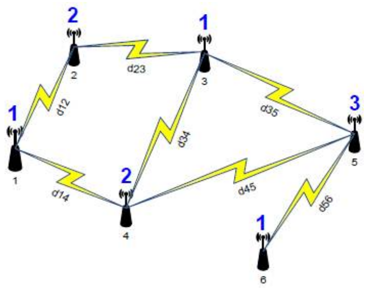


Figure 1. Example for a MinMax CAP with 6 base stations where 3 channels are sufficient.

In the next sections, the general problem of channel allocation is correlated with a special coloring graph problems in graphs, and a job scheduling problem with time window.

2.1 MinMax CAP as a special vertex coloring problem

We present the minmax channel assignment problem as a special coloring problem in graphs, as a mixed list coloring, multicoloring, and weighted coloring problems, that is special variations of the classical coloring problem in graphs ([2], [7]). A cellular network is model as an undirected graph whose edges indicate the proximity between base stations and the weight of the edge, the distance to be respected. Each channel that can be assigned to a base station corresponds to a color that can be assigned to a vertex (which, in turn, will have a natural number representing the same). There are a list of colors for each vertex (channels for a station), and each one can receive one or more colors (channels), been multicolored.

An example is given below. Figure 2 shows a cellular network comprising 7 base stations (vertices), each one with a frequency domain. The distance between the channels of base stations, when there is interference potential (ie an edge) is 2 if such a need and 0 otherwise. All stations have demand for only one channel. For each edge (i, j) this graph, there is the set $\{0, 1, ..., d_i j\}$. As all distances are equal to 2, for all edges of the graph, $\{0, 1, 2\}$. Figure 2 shows an optimal solution to the minimization of span of frequencies for this example, which can be seen as minimizing the number of colors used in a *weighted list coloring problem*.

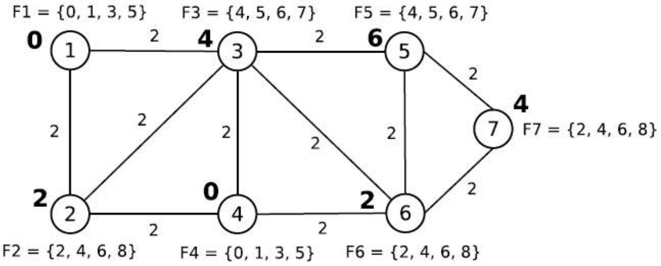


Figure 2. Optimal solution for minmax CAP instance as a weighted list coloring graph coloring.

2.2 MinMax CAP as a job scheduling problem in parallel machines

In general, scheduling problems involving allocation of a set of tasks (jobs) in a particular set of machines in order to satisfy certain conditions, in function of time [8] [5]. Some of the scheduling models for CAP proposed in this work include the situation in which each station needs only one channel, all channel distances are equals for all pairs of base stations and it is desired to minimize the maximum channel used, and the frequency domain of all antennas is the set of natural numbers. This is equivalent to the problem of job scheduling that minimizes the makespan, $P|p_j = p|C_{max}$, where $p_j = 1$ indicates that the processing times of the corresponding jobs are equal. This model can also be extended to other situations, such as when the frequency domain is defined for all base stations ($P|p_j = p, r_j\overline{d_i}|C_{max}$).

Returning to the example of Figure 2, where the CAP is modeled as a special coloring in graphs, can be solved also using the scheduling model above. First, processing times are used to representing possible channels in a station, ie, $p_j = 2$ for all jobs. Each set f_j is treated as the number of possible starting time of jobs. Finally, the objective function is the minimization of makespan. The scheduling problem is then: $P|p_j = 2|C_{max}$. One can see graphically the solution to the model of scheduling by Gantt chart (Figure 3) [5].

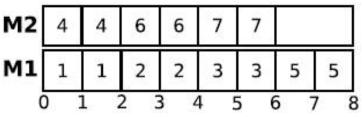


Figure 3. Gantt chart showing the job scheduling solution for Figure 2.

2.3 Algorithmic Strategy and Computational Experiments

The minmax channel assignment problem was solved by an algorithmic strategy adapting IP mathematical models for CAP, to minimize the maximum number of channels by CPLEX tool, which has an implementation of the branch-and-cut exact method and functions of code optimization, among others. We use Concert Technology (for C) library, running on a machine with AMD Turion X2 II of 2.30 GHz and 4GB of DDR3 memory, and operating system Ubuntu

Linux 64-bit 11.10. As instances, we show in this paper only results based on job scheduling model on CALM/CELAR benchmark. For comparison, in Table 1, the instance CELAR05 was modified to make all distances equal to 1 and all operators as $>$, which is to say that for any edge (i, j) of the graph, we have that $|f_i - f_j| = 1$. This artificial instance was called *CELAR*05*_UNIT*. All computational experiments realized are available and they are presented in a full version of this paper.

Table 1. Important results applying our local and global algorithmic strategy.

Instância	**Min**	**ObjF**	**MaxChannel**	**Bench**	**Time**
CELAR04	#Channels	46	-	46	9,98 seg.
CELAR05	Span	776	792	792	87,69 seg.
CELAR05_Unit	Span	160	512	-	5135,86 seg.

3. Summary

This paper addressed a minmax channel allocation problem in cellular wireless networks, correlating the variations of problems as special vertex coloring in graphs, as well as job scheduling problems with time windows.

Mathematical models of Integer Programming were adapted from literature and global/local search algorithms have been developed, with a primal heuristic based on a local search heuristic for scheduling problems ([6]) and a exact method of branch-and-cut. Computational experiments were conducted using benchmark instances from the literature. It was found that an interesting point search is to apply other kinds of special coloring problems in graphs, and job scheduling problems, that better represent the features specified channel allocation problem. Also, the proposition of new mathematical models and the development of exact and approximate methods of solution, are a focus of interest for future research. Finally, we want to explore, as well as issues related to mobile telephone networks, other scenarios involved in mobile wireless networks such as high-demand multimedia application and non-conventional as well as the new model research focus recently, which is the model based on cognitive radio networks.

References

[1] Audhya, G., and Sinha, K., and Ghosh, S., and Sinha, B. (2011). "A survey on the channel assignment problem in wireless networks", Wireless Communications and Mobile Computing.

[2] Chartrand, G., and Zhang, P. (2008). "Chromatic Graph Theory". Chapman Hall/CRC.

[3] Hale, W. (1980). "Frequency assignment: Theory and applications". Proceedings of the IEEE, vol. 68, pp. 1497-1514.

[4] Koster, A. (1999). "Frequency assignment: models and algorithms". Universiteit Maastricht publisher.

[5] Leung, J., ed. (2004). "Handbook of Scheduling: Algorithms, Models, and Performance Analysis". Chapman and Hall/CRC.

[6] Pessoa, A., and Poggi de Aragão, M., and Rodrigues, R., and Uchoa, E. (2010). "Exact algorithm over an arc-time-indexed formulation for parallel machine scheduling problems". Mathematical Programming Computation.

[7] McDiarmid, C., and Reed, B. (2000). "Channel Assignment and Weighted Coloring". John Wiley Sons, Inc.

[8] Rodrigues, R. de F. (2009). *"Caracterizações e Algoritmos para Problemas Clśsicos de Escalonamento"*. Phd Thesis, PESC/COPPE – Universidade Federal do Rio de Janeiro, Brazil.

A column generation heuristic for microdata protection

Éverton Santi,[1] Daniel Aloise,[1] Pierre Hansen,[2] and Caroline Rocha[1]

[1] *Universidade Federal do Rio Grande do Norte, Campus Universitário s/n, 59072-970, Natal-RN Brazil*
santi.everton@gmail.com, aloise@dca.ufrn.br, caroline.rocha@ect.ufrn.br

[2] *GERAD and HEC Montréal, 3000, Chemin de la Côte-Sainte-Catherine, Montréal, Québec, H3T 2A7, Canada*
pierre.hansen@gerad.ca

Abstract The biggest challenge when disclosing private data is to share information contained in databases while protecting people from being individually identified. Microaggregation is a family of methods for statistical disclosure control. The principle of microaggregation is that confidentiality rules permit the publication of individual records if they are partitioned into groups of g or more data, where none is more representative than the others in the same group. The application of such rules leads to replacing individual values by those computed from small groups (microaggregates), before data publication. This work proposes a column generation heuristic for numerical microdata. Computational experiments show that the proposed method finds the best results for a set of benchmark instances in the literature.

Keywords: microaggregation, column generation, heuristic

1. Introduction

The objective in the discipline of Statistical Disclosure Control (SDC) is to allow data to be mined while securing private information [6]. Indeed the best form for data protection is through its encryption. However, encrypted data has no utility for data mining techniques. SDC acts in the tradeoff between the maximum utility of using raw data and the maximum protection provided by encryption. SDC techniques often results in data being modified before made publicly available.

Microaggregation is a class of perturbative SDC methods for microdata (individual records) which has been extensively studied recently. The principle of microaggregation is that individual records can be replaced by those computed from small homogeneous groups (microaggregates), before data publication. Since the protected dataset contains only the masked data, its disclosure is less likely to violate individual privacy.

Domingo-Ferrer and Mateo-Sanz defined in [2] a mathematical programming model for microaggregation over numerical microdata. In their model, microdata is clustered with the minimum sum-of-squares criterion into groups of size larger or equal to a parameter g using as many clusters k as needed. It is expressed as follows:

$$SSE = \min_{x,y,k} \quad \sum_{i=1}^{n}\sum_{j=1}^{k} x_{ij}\|p_i - y_j\|^2 \tag{1}$$

subject to

$$\sum_{j=1}^{k} x_{ij} = 1, \qquad \forall i = 1,\ldots,n \tag{2}$$

$$\sum_{i=1}^{n} x_{ij} \geq g, \qquad \forall j = 1,\ldots,k \tag{3}$$

$$x_{ij} \in \{0,1\}, \qquad \forall i = 1,\ldots,n; \forall j = 1,\ldots,k \tag{4}$$

$$y_j \in \mathbb{R}^s \qquad \forall j = 1,\ldots,k. \tag{5}$$

The numerical microdata of n individual records are represented by points $p_i = (p_i^r, r = 1,\ldots,s)$ in $\mathbb{R}^s$ for $i = 1,\ldots,n$; k cluster centers must be located at unknown points $y_j \in \mathbb{R}^s$ for $j = 1,\ldots,k$; the norm $\|\cdot\|$ denotes the Euclidean distance between the two points in its argument in the s-dimensional space under consideration. The binary decision variables x_{ij} express the assignment of the point p_i to the cluster j. The set of constraints (2) assure that every point p_i, $i = 1,\ldots,n$, is assigned to a cluster. Constraints (3) define that the size of each cluster is greater or equal to g. Othewise, the trivial optimal solution would consist of singleton clusters.

2. A column generation formulation

Problem (1)-(5) correspond to a partitioning problem where the number of parts is to be determined. Let us consider any cluster C_t, with $|C_t| \geq g$, for which

$$a_{it} = \begin{cases} 1 & \text{if entity } o_i \text{ belongs to cluster } C_t \\ 0 & \text{otherwise,} \end{cases}$$

and let us denote by y_t the centroid of points p_i such that $a_{it} = 1$. Thus, the cost c_t of cluster C_t can be written as $c_t = \sum_{i=1}^{n} \|p_i - y_t\|^2 a_{it}$.
An alternative formulation for the microaggrgation problem (1)-(5) is then given by

$$\begin{array}{l} \min\limits_{z} \sum\limits_{t\in T} c_t z_t \\ \text{subject to} \\ \sum\limits_{t\in T} a_{it} z_t = 1, \quad \forall i = 1,\ldots,n \\ z_t \in \{0,1\} \quad \forall t \in T, \end{array} \tag{6}$$

where $T = \{1,\ldots,2^n-1\}$. The z_t variables are equal to 1 if cluster C_t is in the optimal partition, and to 0 otherwise. The constraints state that each entity belongs to one cluster.

3. Pricing problem

The dual of the formulation (6) is expressed by

$$\begin{array}{ll} \max & \sum\limits_{i=1}^{n} \lambda_i \\ \text{subject to} & \\ \sum\limits_{i=1}^{n} a_{it}\lambda_i \leq c_t & \forall t \in T \\ \lambda_i \text{ free} & i = 1,\ldots,n \end{array} \tag{7}$$

where the λ_i for $i = 1,\ldots,n$ are dual variables associated with the covering constraints.

Problem (7) is solved using a cutting plane method, starting with a relaxation and adding constraints as necessary. Given dual values λ, σ, a violated cut is searched to be added to the relaxed dual problem. The slack, or when negative, the violation π_t of a constraint is given by $\pi_t = c_t - \sum_{i=1}^n \lambda_i a_{it}$. Since we are interested in finding violated constraints $\pi_t < 0$, the auxiliary problem is given by $\pi^* = \min_t \pi_t$. Although the enumeration of π_t for all $t \in T$ is too expensive, the value of π^* can be found by solving

$$\begin{array}{ll} \min \sum\limits_{i=1}^{n} (\|p_i - y_v\|^2 - \lambda_i) v_i & \\ \sum\limits_{i=1}^{n} v_i \geq g & \\ v_i \in \{0, 1\}, & \forall i = 1, \ldots, n \\ y_v \in \mathbb{R}^s & \end{array} \tag{8}$$

with y_v denoting the centroid of points p_i for which $v_i = 1$. If $\pi^* < 0$, then the optimal solution v^* to (8) is added as a cut to the relaxed dual problem (in the primal, this is equivalent to adding a column to the restricted master problem together with its associated primal variable). Otherwise, problem (7) (or equivalently, problem (6)) is solved optimally.

The objective function of the pricing problem (8) can be viewed as minimizing the sum of functions equal to squared distances from the cluster center y_v to each of the entities, but with a limit on each of the distances, after which the corresponding function does not increase anymore. Clearly, for a given location y_v, v_i is equal to 1 if $\|p_i - y_v\|^2 \leq \lambda_i$, and to 0 otherwise. Geometrically, in the plane, this is equivalent to the condition that $v_i = 1$ if y_v belongs to a disc $D_i = \{y \mid \|p_i - y\|^2 \leq \lambda_i\}$ (i.e., a disc with radius $\sqrt{\lambda_i}$ centered at p_i), and 0 otherwise. However, due to constraint $\sum_{i=1}^n v_i \geq g$, a variable v_i may be forced to 1, in order to compose a valid cluster/column with g or more elements. Consequently, decomposition approaches as that developed in [1] cannot be used here to solve the pricing problem.

Indeed, we need to find just one negative cost solution (v, y_v) for (8) in order to add the corresponding column to the restricted master problem. Proposition 1 shows that, if it exists, such solution (v, y_v) can be searched among those with $\sum_{i=1}^n v_i = g$.

Proposition 1. *If a negative cost solution (v', y') such that $\sum_{i=1}^n v'_i \geq g + 1$ exists for (8) then a negative cost solution (v'', y'') with $\sum_{i=1}^n v''_i = g$ also exists.*

Proof. Without loss of generality, let us suppose that (v', y') is a negative cost solution for (8) with $\sum_{i=1}^n v'_i = g + 1$. Define S as the set of v' components for which $v'_i = 1$. Let us suppose now that there is no solution (v'', y'') for (8) with $\sum_{i=1}^n v''_i = g$. I.e., it is impossible to find a y'' for which $\sum_{i=1}^n (\|p_i - y''\|^2 - \lambda_i) v''_i$ is negative using only g components of v'' equal to 1. Consequently, there exists at least one element v_i* in S for which $\|p_{i^*} - y'\|^2 - \lambda_{i^*} > 0$. Hence, if v'_{i^*} is made equal to 0, a solution (v^*, y') with only three components equal to 1 is produced, having negative cost smaller than that of (v', y'), which is a contradiction. □

Corollary 2. *The optimal solution of the linear relaxation of problem (6) uses only variables whose columns have g non-zero elements.*

Proof. From Proposition 1, negative reduced cost variables associated to columns with $k > g$ non-zero elements only exist if at least one negative reduced cost variable associated to a column with g non-zero elements also exists. Consequently, a column algorithm for solving (6) is convergent if it adds at each iteration only variables associated to columns with exactly g non-zero elements. □

Our column generation algorithm uses the previous propositions to restrict the search in the pricing problem. In particular, microaggregation problems have been always benchmarked in the literature using instances with $g = 3, 5$ and 10, which is advantageous for our approach due to two main reasons: (i) column generation has usually good performance when columns are *small* (i.e., with few entries equal to 1) [4], and (ii) by using Proposition 1, the enumeration of pricing solutions for g small is not an expensive task for moderate n.

4. Preliminary computational results

We tested our algorithm in the following benchmark data sets from [2]: (i) TARRAGONA with $n = 834$ and $s = 13$, (ii) CENSUS with $n = 1080$ and $s = 13$, and (iii) EIA with $n = 4092$ and $s = 11$, for $g = 3$. The tests were performed in a AMD Phenom II with a 800 Mhz clock and 8 Gb of RAM memory. The restricted master problem was iteratively solved by CPLEX 12. Pricing problems (8) are solved by a specialized branch-and-bound based on Proposition 1. Finally, our column generation heuristic solves a MIP with all generated columns in order to obtain an approximate solution for the microaggregation problem. Table 1 presents the computational results for each one of the tested instances. First column shows the number of column generation iterations. Second and third columns report lower bounds (LB) and upper bounds (UB), while the fourth column (gap) presents their relative difference calculated as (UB-LB)/LB. Finally, the last column reports the total CPU time of our column generaton heuristic (CG), stabilized [3] with a solution provided by MDAV [2].

Table 1. Bounds and CPU times obtained by the column generation heuristic

Data set	CG				
	iter.	LB	UB	$gap(\%)$	CPU time(s)
TARRAGONA	4034	1502.42	1569.56	4.46	233.43
CENSUS	5226	652.30	655.60	0.50	338.02
EIA	11564	1.2533e+13	1.3301e+13	6.12	7026.7

Lower bounds for problem (6) were obtained for all instances. Moreover, the heuristic solutions obtained (column UB) are the best known in the literature for $g = 3$. In particular, the best upper bound solution for EIA with $g = 3$ was improved in more than 50%.

Acknowledgments

This work is supported by CNPq (Brazil) project 474231/2010-0 and by HEC Montreal Data Mining Chair (Canada).

References

[1] Aloise, D., Hansen, P. and Liberti, L. (2012). “An improved column generation algorithm for minimum sum-of-squares clustering”, Mathematical Programming, v. 131, 195–220.

[2] Domingo-Ferrer, J. and Mateo-Sanz, J.M. (2002). “Practical data-oriented microaggregation for statistical disclosure control”. IEEE Transactions on Knowledge and Data Engineering, 14:189–201.

[3] du Merle O., Villeneuve D., Desrosiers J., Hansen P. (1999). “Stabilized column generation”. Discrete Mathematics, v. 194, 229–237.

[4] Elhallaoui I., Villeneuve D., Soumis, F., Desaulniers, G. (2005). “Dynamic aggregation of set-partitioning constraints in column generation”. Operations Research, v. 53, 632–645.

[5] Hansen, P. and Mladenović, N. (2001). “Variable neighborhood search: Principles and applications”, European Journal of Operational Research, v. 130, 449–467.

[6] Willenborg, L. and DeWaal, T. (2001). “Elements of Statistical Disclosure Control”. Springer, New York.

Modeling the bi-objective Diameter Minimum Spanning Tree Problem

Andréa C. Santos[1], Diego R. Lima[2] and Dario J. Aloise[2]

[1]*ICD-LOSI, Université de Technologie de Troyes*
12, rue Marie Curie, BP 2060, 10010, Troyes CEDEX, France.
andrea.duhamel@utt.fr

[2]*Universidade do Estado do Rio Grande do Norte, UERN*
Rua Almino Afonso, 478, CEP 59.610-210, Mossoró, RN, Brasil.
{drlima7, aloisedj}@gmail.com

Abstract In this work, we propose formulations for the bi-objective Diameter Minimum Spanning Tree problem which consists in finding a spanning tree with minimum total cost and minimum diameter.

Keywords: Spanning trees, Mixed integer linear programming, Bi-objective

1. Introduction

The bi-objective Diameter Minimum Spanning Tree problem (bi-DMST) extends the Bounded Diameter Minimum Spanning Tree problem (BDMST) [4, 8] and the Minimum Diameter Spanning Tree problem (MDST) [1, 6]. The bi-DMST is also referred in the literature as the Minimum Diameter Minimum Cost Spanning Tree problem [7]. The diameter of a spanning tree is the number of edges in the longest path between any pair of nodes. The BDMST consists in finding a spanning tree with minimum total cost where the diameter does not exceed a given positive integer value. The MDST looks for a spanning tree (not necessarily with minimum total cost) where the diameter is minimized. The bi-DMST consists in finding a spanning tree with minimum total cost and minimum diameter. We propose mathematical formulations for the bi-DMST using different strategies to deal with the two objectives.

Formally, the bi-DMST is defined as follows. Let $G = (V, E)$ be a connected and undirected graph with a set V of vertices and a set E of edges. A cost $c_{ij} \geq 0$ is associated to each edge $[i, j] \in E$, with $i < j$. Let $\mathcal{T}$ be a spanning tree of G. Thus, there is a unique path $\mathcal{P}_{ij}$ in $\mathcal{T}$ linking any pair of node $i, j \in V$. Let d_{ij} be the number of edges in $\mathcal{P}_{ij}$. Then, the diameter D of $\mathcal{T}$ is defined as $D = max\{d_{ij} : i, j \in V\}$. A Minimum Spanning Tree (MST) of G is a spanning tree $\mathcal{T}$ with minimum total cost. The bi-DMST consists in defining a MST with minimum diameter. The bi-DMST addresses network design and transportation logistic applications. In network design, the diameter refers to quality of service (QoS) requirements where small diameters reduce delays and improve reliability. Another application appears on high speed trains where one looks for a MST backbone and minimum diameters reduce the transportation time between any pairs of cities and improves QoS [5].

The BDMST is NP-hard [2] when $4 \leq D < |V| - 1$, and several formulations [3, 9], exact and heuristics methods [4, 8] are found in the literature. The MDST has received less attention, distributed and efficient algorithms are presented in [1, 6]. The bi-DMST is NP-hard [7] and as far as we know, this is the first work dedicated to propose formulations for the bi-DMST. A multicommodity flow formulation and an optimization in two phases are respectively presented in Sections 2 and 3. Preliminary results and conclusions are given in Section 4.

2. A general multicommodity flow formulation

The formulation (1)-(10) is inspired on the work of Gouveia and Magnanti for the BDMST [3]. It makes use of an undirected graph $G = (V, E)$ and the diameter $D = max\{d_{pq} : p, q \in V\}$. Let x_{ij} be the decision variables on the choice of edge $[i, j]$. If edge $[i, j]$ belongs to the solution $x_{ij} = 1$, otherwise $x_{ij} = 0$. The directed flow variables y_{ij}^{pq} specify if the path from $p \in V$ to $q \in V$, with $i \neq q$ and $j \neq p$, passes through edge $[i, j]$, i.e. $y_{ij}^{pq} = 1$, otherwise $y_{ij}^{pq} = 0$. Let variables d_{pq} be the number of edges in the path from p to q.

$$z_1(x) = \min \sum_{[i,j] \in E} c_{ij} \cdot x_{ij} \tag{1}$$

$$z_2 = \min \ D \qquad \text{st} \tag{2}$$

$$\sum_{[i,j] \in E} x_{ij} = |V| - 1 \tag{3}$$

$$\sum_{j:[i,j] \in E} y_{ij}^{pq} - \sum_{j:[j,i] \in E} y_{ji}^{pq} = \begin{cases} 1, & \text{if } i = p \\ 0, & i \neq p \text{ and } i \neq q \\ -1, & \text{if } i = q \end{cases} \quad \forall i \in V, \forall p, q \in V \tag{4}$$

$$y_{ij}^{pq} + y_{ji}^{pq} \leq x_{ij} \qquad \forall [i, j] \in E, \forall p, q \in V \tag{5}$$

$$\sum_{[i,j] \in E} \left(y_{ij}^{pq} + y_{ji}^{pq} \right) \leq d_{pq} \qquad \forall p, q \in V \tag{6}$$

$$D \geq d_{pq} \qquad \forall p, q \in V \tag{7}$$

$$y_{ij}^{pq} \in \{0, 1\} \qquad \forall [i, j] \in E, \forall p, q \in V, i \neq q, j \neq p \tag{8}$$

$$x_{ij} \in \{0, 1\} \qquad \forall [i, j] \in E \tag{9}$$

$$d_{pq} \geq 1 \qquad \forall p, q \in V \tag{10}$$

The two objectives are given in equations (1) and (2) and they aim respectively at minimizing the total cost and the diameter. Restriction (3) ensures the spanning tree has at most $|V| - 1$ edges. Restrictions (4) are the classic multiflow conservation constraints. Inequalities (5) state no flow passes through edge $[i, j]$ whenever edge $[i, j]$ does not belong to the solution, i.e. $x_{ij} = 0$. Constraints (6) compute the number of edges in a path from p to q. Restrictions (7) together with the objective (2) minimize the diameter. Variables are defined from (8) to (10).

3. Optimization in two phases for the bi-DMST

Using an optimization in two phases, the objectives are optimized in a priority order. The diameter is considered as the priority objective. The second optimization phase seeks a MST where the diameter is bounded to the value found in the first optimization phase. Such a strategy is interesting for applications where small diameters are needed. Moreover, one can use well-known formulation for the BDMST in the second optimization phase.

We use the formulation (2) to (10) for the first optimization phase. As pointed out in [3, 9], once the diameter is fixed, single multiflow formulation can be applied for the BDMST. Then, in the second optimization phase, we considered the formulations introduced in [9]. These formulations rely on the property that whenever D is even, the MST has a central vertex i (resp. a central edge $e = [i, j]$ when D is odd) such that no other vertex is more than $D/2$ edges away from i (resp. for odd D values, no more than $(D - 1)/2$ edges away from one extremity of e). A digraph $G' = (V', A')$ is obtained from $G = (V, E)$ as follows: an artificial vertex r is introduced in V. Thus, $V' = V \cup \{r\}$ and $A' = A \cup \{(r, 1), \ldots, (r, |V|)\}$, and for every edge $[i, j] \in E$, with $i < j$, arcs (i, j) and $(j, i) \in A'$ are added with costs $c_{ij} = c_{ji}$.

Both odd and even D formulations use the decision variables x_{ij} on the choice of arc $(i, j) \in A'$, and non-negative variables u_i which specify the number of arcs in a path from r to $i \in V'$. When D is odd, the formulation also uses binaries variables z_{ij} that define whenever edge $[i, j] \in E$ is selected as the central spanning tree edge $z_{ij} = 1$, or not $z_{ij} = 0$. Let D^* be the optimal diameter found in the first optimization phase. Thus, $L = D^*/2$ when D^* is even and $L = (D^* - 1)/2$ when D^* is odd. For D^* even, the second optimization phase is given as :

$$\min \sum_{(i,j)\in A} c_{ij} \cdot x_{ij} \quad \text{st} \tag{11}$$

$$\sum_{j\in V} x_{rj} = 1 \tag{12}$$

$$\sum_{(i,j)\in A'} x_{ij} = 1 \quad \forall j \in V \tag{13}$$

$$u_i - u_j + (L+1)x_{ij} + (L-1)x_{ji} \leq L \quad \forall (i,j) \in A' \tag{14}$$

$$u_i \leq L + 1 \quad \forall i \in V' \tag{15}$$

$$x_{ij} \in \{0,1\} \quad \forall (i,j) \in A' \tag{16}$$

$$u_i \geq 0 \quad \forall i \in V' \tag{17}$$

The objective (11) minimizes the total cost. Restriction (12) states the artificial vertex r is connected to only one vertex in V. Constraints (13) ensure that only one arc must be incident to each vertex of V. Inequalities (14) and (15) establish that paths from the artificial vertex r to each vertex $i \in V$ have at most $L + 1$ arcs. Variables are defined in (16) and (17). When D is odd, the formulation takes into account the MST center is an edge as follows:

$$\min \sum_{(i,j)\in A} c_{ij} \cdot x_{ij} + \sum_{[i,j]\in E} c_{ij} \cdot z_{ij} \quad \text{st} \tag{18}$$

$$\sum_{j\in V} x_{rj} = 2 \tag{19}$$

$$\sum_{[i,j]\in E} z_{ij} = 1 \tag{20}$$

$$z_{ij} = x_{ri} \cdot x_{rj} \quad \forall [i,j] \in E \tag{21}$$

$$z_{ij} \in \{0,1\} \quad \forall [i,j] \in E \tag{22}$$

Constraints (13) to (17)

The objective function (18) computes the total cost including the center edge cost. Restriction (19) ensures the artificial central vertex r is connected to exactly two vertices of V. Constraints (20) and (21) guarantee only one central edge is selected. Restrictions (21) are non-linear, but they can be easily linearized as shown in [9]. Variables z_{ij} are defined in (22). Constraints (13) to (17) have already being defined. Readers are referred to [9] for further details.

4. Preliminary results and concluding remarks

The experiments were performed on an Intel Core i7 with 2.7 GHz clock and 8Gb of RAM memory, using CPLEX 12 under default parameters. Preliminary results are reported for the optimization in two phases given in Section 3. We have developed two sets of sparse instances. In the first test set, an arbitrary $Hamiltonien\ cycle$ is built to ensure the graph connectivity. The remaining edges are randomly added accordingly to the graph density set to {0.2, 0.3, 0.4}. In the second test set, graph connectivity is ensured by an arbitrary $Hamiltonien\ path$. The remaining edges are randomly added and the graphs have density {0.1, 0.09, 0.08}. Table 1 presents a subset of results. "c" and "p" stand respectively for the first and the second test sets. The number $|V|$ of vertices, the graph density d are given, followed by the results for the first and the second optimization phases. The optimal diameter and optimal cost are respectively depicted in columns D^* and C^*. "RL", "time" and "nodes" columns contain respectively the linear relaxation, the running time in seconds and the number of nodes visited in the branch and bound tree for the corresponding optimization phases.

Table 1. Results for the optimization in two phases.

Instances data			First optimization phase				Second optimization phase			
test sets	$\|V\|$	d	D*	RL	time	nodes	C*	RL	time	nodes
c	15	0.2	7	4.12	76.9	2,375	497	398.3	0.16	171
	15	0.3	4	4.00	74.6	83	635	392.3	1.37	10,659
	15	0.4	4	3.00	30,862.2	20,250	465	341.0	0.81	2,972
	20	0.2	5	4.00	29,123.0	14,215	980	748.2	0.97	3,072
	20	0.3	5	3.02	326,655.7	72,378	618	436.7	1.67	4,412
p	30	0.08	12	9.00	312.5	920	1,858	1,822.3	0.16	550
	30	0.09	9	7.00	49,340.0	32,321	1,407	1,161.1	2.10	10,127
	30	0.1	9	7.00	271,062.5	43,341	1,315	1,088.0	3.90	23,736

Results indicate the proposed test sets are difficult to be solved. They show an expected behavior : the tree diameter reduces when the graph density increases, and the problem becomes harder to be solved. In this work, the multiflow formulation presented in [3] has been adapted for the bi-DMST. Optimal values and lower bounds have been computed for sparse instances. The preliminary results motivate further investigation on heuristics and metaheuristics for the bi-DMST. We are currently working on heuristics and metaheuristics (NSGAII) that make use of Pareto function and Lorenz dominance for the general multiflow formulation given in Section 2.

References

[1] M. Bui, F. Butelle, and C. Lavault. A distributed algorithm for constructing the minimum spanning tree problem. *Journal of Parallel and Distributed Computing archive*, 64:571–577, 2004.

[2] M.R. Garey and D.S. Johnson. *Computers and intractability: A guide to the theory of NP-completeness*. W. H. Freeman, New York, 1979.

[3] L. Gouveia and T. L. Magnanti. Network flow models for designing diameter-constrained minimum-spanning and steiner trees. *Networks*, 41:159–173, 2003.

[4] L. Gouveia, L. Simonetti, and E. Uchoa. Modeling hop-constrained and diameter-constrained minimum spanning tree problems as steiner tree problems over layered graphs. *Mathematical Programming*, 128:123–148, 2011.

[5] B. W. Thomas H. Chen, A. M. Campbell. Network design for time-constrained delivery. *Naval Research Logistics*, 55:493–515, 2008.

[6] R. Hassin and A. Tamir. On the minimum diameter spanning tree problem. *Information Processing Letters*, 53(2):109–111, 1995.

[7] J.-M. Ho, D. T. Lee, C.-H. Chang, and K. Wong. Minimum diameter spanning trees and related problems. *SIAM Journal on Computing*, 20(5):987–997, 1991.

[8] A. Lucena, C. Ribeiro, and A. C. Santos. A hybrid heuristic for the diameter constrained minimum spanning tree problem. *Journal of Global Optimization*, 46:363–381, 2010.

[9] A. C. Santos, A. Lucena, and C. C. Ribeiro. Solving diameter constrained minimum spanning tree problem in dense graphs. *Lecture Notes in Computer Science*, 3059:458–467, 2004.

Scalability Analysis of a Parallel Coupled Simulated Annealing Implementation

Kayo Gonçalves e Silva, Samuel Xavier de Souza, and Daniel Aloise

Universidade Federal do Rio Grande do Norte, Natal, Brasil, {kayokgs, samuel, aloise}@dca.ufrn.br

Abstract We evaluated the parallel performance of the Coupled Simulated Annealing (CSA) for unconstrained optimization. The CSA is characterized by a set of SA processes coupled via their acceptance probabilities in order to exploit information from each SA process. This paper seeks to identify the particularities of the CSA to determine its parallel scalability. We evaluate the scalability of the CSA using performance metrics for parallel systems. Experiments were performed fixing the problem dimension and fixing the number of processing cores. In a synthesized data set, the results show that the CSA behaves as a scalable algorithm, allowing a significant improvement in efficiency when the dimension of the problem increases.

Keywords: coupled simulated annealing, heuristics, parallel performance, parallel scalability

1. Introduction

In recent years, we have seen an increasing number of computing devices that use processors with many cores. The emergence of multicore processors became a necessity because it is more difficult to cool singlecore processors with increasing speed. Moreover, the recent increase in the availability of computers with multicore processor and powerful graphics processing units (GPU), as well as the demand for high performance and low energy [5], motivated new research in parallel computing. Its consequence is the tendency to manufacture multicore processors with lower frequency, in addition to finding that parallel processing has emerged as the main motivator for contemporary computing. This trend has been called "The Multicore Era" [1, 2].

Owing to the larger computing power, several algorithms can be reformulated in an attempt to exploit its parallel potential. However, a more refined analysis of multicore chips is necessary to verify the leverage of the parallel performance and its scalability.

To improve the solution quality and reduce processing time, Xavier-de-Souza *et. al* [3] defined a class of optimization methods based on Simulated Annealing (SA) that can be used to solve unconstrained nonconvex problems. Coupled Simulated Annealing (CSA) is a parallel algorithm that couples the acceptance probabilities of the various SA processes. The coupling term is a function of the current energy of all SA processes. Through this term, performance indicators can be controlled, allowing the extract of more information when deciding to accept less favourable solutions and to improving the final quality of the solutions [3]. Despite being effective in improving solutions, no analyses was made to demonstrate its parallel scalability.

In this paper, we present an analysis of the parallel potential of the CSA applying the performance metrics for parallel systems to demonstrate its scalability.

2. Coupled Simulated Annealing

The CSA is a stochastic algorithm. Consist of several SA processes jointly minimizing a cost-function. Each SA process is composed of two stochastic processes: one is responsible for generation and the other for the acceptance. These processes are controlled by a generation temperature T and a acceptance temperature T^{ac}.

The CSA considers all current solutions in a set Θ and accept a probe solution y_i based not only on the current solution x_i, but also on the coupling term γ, which depends on the energy of all elements of Θ.

The acceptance probability function $A_\Theta : \Theta \to \mathbb{R}$ defined in [3] was used in this work:

$$A_\Theta\left(\gamma, x_i \to y_i\right) = \frac{\exp\left(\frac{E(x_i) - max(E(x_i))_{x_i \in \Theta}}{T^{ac}}\right)}{\gamma} \quad (1)$$

$$\gamma = \sum_{\forall x \in \Theta} \exp\left(\frac{E(x) - max\left(E\left(x_i\right)\right)_{x_i \in \Theta}}{T^{ac}}\right) \quad (2)$$

High values of T^{ac} implies large γ, hence a lower acceptance probability. Processes with low acceptance probability are more likely to accept less favorable solutions. The opposite happens for low values of T^{ac}.

It is possible to control the variance of the acceptance probabilities to follow a desired value specified, acting directly on the T^{ac} schedule, eliminating the necessity of fixing a scheduling and the initial value of T^{ac}. The acceptance temperature affects the probabilities and thus its control alter the program flow in the search for optimal solutions.

In [3], it is shown that the variance of the acceptance probabilities is within the limits $0 \leq \sigma^2 \leq \frac{m-1}{m^2}$, where m is the number of processes. Experiments showed that the performance increases with the variance around 99% of its maximum value. A simple control rule, such that if $\sigma^2 < \sigma_D^2$ then $T^{ac} = T^{ac}\left(1 - \alpha\right)$, otherwise $T^{ac} = T^{ac}\left(1 + \alpha\right)$ is used, where σ_D^2 is the desired variance and α the depreciation rate of temperature, usually a value between $(0, 0.1]$ [3]. The variance for A_Θ assumes the form $\sigma^2 = \frac{1}{m}\sum_{\forall x_i \in \Theta} A_\Theta^2 - \frac{1}{m}$

In Figure 1, the implementation of the algorithm is given. A parallel region is created, each corresponding to a SA process. Each SA process generates and analyses probing solutions, accepting them or not. Then, each SA process will try to enter a critical region at each iteration, otherwise it will increase an internal variable K_i, which counts the number of iterations of process i, and evaluate the stopping criterion. To enter a critical parallel region, the process must perform a $test\_lock$. The $test\_lock$ works similarly to a $lock$, but it does not force the calling process to wait until the specified lock is available. If a process manage to enter a critical region, it performs the control of global iteration K_{global}, the control of temperature T and T^{ac}; after that it leaves the critical region, increase K_i and evaluate the stopping criterion. If the criterion of any SA processes is met, the respective SA process ends; if not, a new iteration cycle starts, continuing until all SA processes are finished.

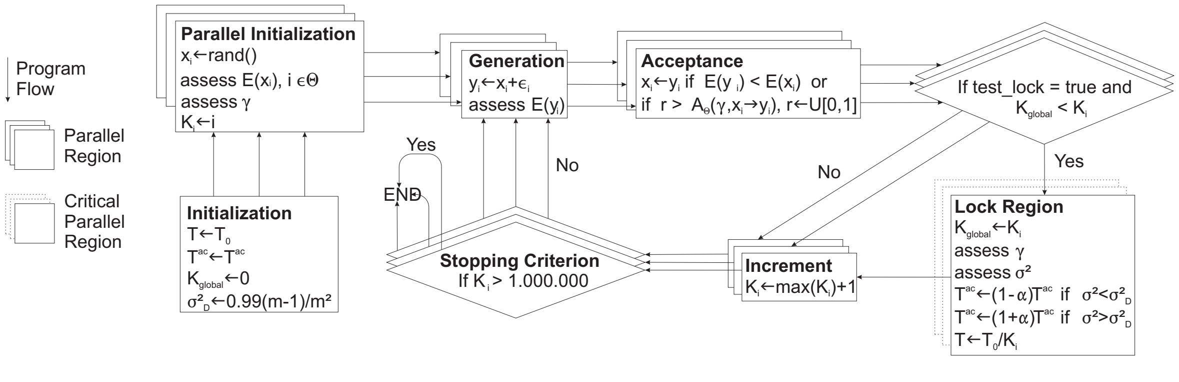

Program Flow
Parallel Region
Critical Parallel Region
Parallel Initialization
$x_i \leftarrow rand()$
assess $E(x_i)$, $i \in \Theta$
assess γ
$K_i \leftarrow i$
Initialization
$T \leftarrow T_0$
$T^{ac} \leftarrow T^{ac}$
$K_{global} \leftarrow 0$
$\sigma^2_D \leftarrow 0.99(m-1)/m^2$
Generation
$y_i \leftarrow x_i + \epsilon_i$
assess $E(y_i)$
Acceptance
$x_i \leftarrow y_i$ if $E(y_i) < E(x_i)$ or
if $r > A_\Theta(\gamma, x_i \to y_i)$, $r \leftarrow U[0,1]$
If test_lock = true and $K_{global} < K_i$
Yes
No
Lock Region
$K_{global} \leftarrow K_i$
assess γ
assess σ^2
$T^{ac} \leftarrow (1-\alpha)T^{ac}$ if $\sigma^2 < \sigma^2_D$
$T^{ac} \leftarrow (1+\alpha)T^{ac}$ if $\sigma^2 > \sigma^2_D$
$T \leftarrow T_0/K_i$
Increment
$K_i \leftarrow max(K_i)+1$
Stopping Criterion
If $K_i >$ 1.000.000
END
Yes
No

Figure 1. A Parallel Coupled Simulated Annealing Implementation

3. Parallel Performance Metrics

Although simplistic, Amdahl's Law [4] states that the sequential fraction of code severely limits the scalability of the algorithm when increasing the number of available processors. Its involvement will be felt in the analysis of speedup S, defined as the division of the serial processing time T_s by the parallel processing time T_p, and the efficiency E, defined as the division of the speedup by the number of processors p.

$$S = \frac{T_s}{T_p} \tag{3}$$

$$E = \frac{S}{p} \tag{4}$$

The efficiency expresses the percentage of the speedup achieved by the algorithm in relation to a linear speedup. Values above 1 indicate super-linear speedups; below, express infra-linearity, and equal to 1 show linear speedup. The goal of parallel algorithms is to achieve the linear speedup. For a scalable algorithm, it is expected that this linearity is achieved when the problem increases in size. This occurs because increasing the size of the problem in a scalable algorithm increases more the parallel fraction of code than the serial fraction. Thus, an indicative of good scalability is an increasing efficiency with a growing problem size.

4. Results

We conducted two performance analyses for our Coupled Simulated Annealing implementation. The first is the analysis of the speedup of the problem when the its dimension is constant, varying the number of processing cores. The second is the examination of the efficiency of the CSA when we fix the number of processing cores and vary the dimension of the problem.

The objective function used in our tests was $\sum_{i=1}^{Dim} C(x_i)$, where $C(x_i) = sen(x_i)$ for i even, $C(x_i) = cos(x_i)$ for i odd and Dim is the dimension of the problem.

We wrote the code in C with OpenMP. We used 1,000,000 iterations as stopping criterion. We measured the runtime for the serial version of the CSA algorithm and for the parallel code using p cores, with $2 \leq p \leq 24$. We repeated the tests increasing the problem dimension for a fixed p.

We calculated the speedup and efficiency for each test situation, according to (3) and (4). The tests were performed on a computer with two AMD Opteron 6172 processors, each one with 12 cores.

4.1 Fixing the Dimension of Problem

For this analysis, the problem dimension was fixed and experiments were performed by increasing the number of cores.

In this case, for a given dimension, the parallel portion is fractionated and then executed by different cores. As the serial fraction is not drastically reduced, it was expected that the speedup diverges from the ideal speedup when the number of cores increases, as shown in Figure 2. This experiment demonstrates the typical behavior of parallel systems.

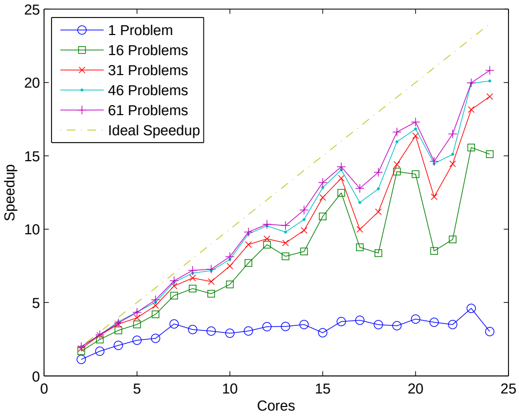


Figure 2. Number of Cores vs Speedup.

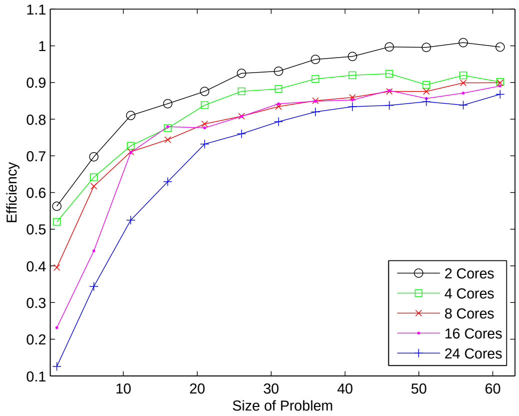


Figure 3. Dimension of Problem vs Efficiency.

4.2 Fixing the Number of Cores

We fixed the number of cores and increased the dimension of problem. For scalable parallel systems, it is expected that the efficiency approaches the unity. Thus, with the improvement in efficiency, shown in Figure 3, showing the typical characteristic of scalable algorithms, we can confirm that the scalability of our implementation for higher problem dimensions.

5. Summary

We have presented a scalability analysis of a parallel Coupled Simulated Annealing implementation. To study the performance of the CSA, two analyses were made in which the CSA is thoroughly tested. First, we analysed the algorithm and its behavior as a common parallel algorithm. In this analysis, for a fixed dimension problem, we increased the number of processing cores. The result showed that our implementation has a typical characteristic of parallel systems, i.e., the speedup diverges from the ideal speedup when the number of the cores increases. In the second analysis, to demonstrate its scalability, we fixed the number of processing cores and increased the problem dimension. There was an upward trend in the efficiency, confirming its good scalability. The results for the scalability analysis, therefore, confirm that our parallel Coupled Simulated Annealing implementation has good scalability.

Acknowledgments

The authors wish to thank MCT, CNPq and FAPERN for the financial support.

References

[1] G. Koch, "Discovering Multi-Core: Extending the Benefits of Moore's Law," in Techno-logy@intel Magazine. Technical report, Intel Corporation, Jul 2005.

[2] S. Borkar, "Thousand core chips-a technology perspective," in 2007 44th ACM/IEEE Design Automation Conference, pages 746-749, 2007.

[3] Xavier-de-Souza, S., Suykens, J. A. K., Vandewalle, J. e BollÃl', D. (2010), "Coupled Simulated Annealing," IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics, 40(2), 320-335.

[4] G. M. Amdahl. "Validity of the single processor approach to achieving large scale computing capabilities," Proc. AFIPS 1967 Spring Joint Computer Conf. 30 (April), Atlantic City, N.J., pages 483-485, 1967.

[5] Xuejun Yang, Yong Dou, Qing-Feng Hu. Progress and Challenges in High Performance Computer Technology. J. Comput. Sci. Technol., 2006: 674 681

Biased random-key genetic algorithm for bound-constrained global optimization

R. M. A. Silva[1], M. G. C. Resende[2], P. M. Pardalos[3], J. F. Gonçalves[4]

[1]*Centro de Informática (CIn), Universidade Federal de Pernambuco, Recife, PE, Brazil,* rmas@cin.ufpe.br

[2]*AT&T Labs Research, Florham Park, NJ, USA,* mgcr@research.att.com

[3]*Department of Industrial and Systems Engineering, University of Florida, Gainesville, FL, USA,* pardalos@ufl.edu

[5]*LIAAD, Faculdade de Economia do Porto, Universidade do Porto, Porto, Portugal,* fgoncal@fep.up.pt

Abstract Global optimization seeks a minimum or maximum of a multimodal function over a discrete or continuous domain. In this paper, we propose a biased random-key genetic algorithm for finding approximate solutions for continuous global optimization problems subject to box constraints. Experimental results illustrate its effectiveness on the robot kinematics problem, a challenging problem according to [7].

Keywords: random-key genetic algorithms, global optimization, metaheuristics.

1. Introduction

Global minimization optimization seeks a solution $x^* \in S \subseteq R^n$ such that $f(x^*) \leq f(x)$, $\forall\, x \in S$, where S is some region of R^n and the objective function f is defined by $f : S \to R$. In this paper, we present the BRKGA heuristic for solving continuous global optimization problems subject to box constraints. Without loss of generality, we take the domain S as the hyper-rectangle $S = \{x = (x_1, \ldots, x_n) \in R^n : \ell \leq x \leq u\}$, where $\ell \in R^n$ and $u \in R^n$ such that $u_i \geq l_i$, for $i = 1, \ldots, n$. Therefore, the minimization problem considered in this paper consists in finding $x^* = \text{argmin}\{f(x) \mid \ell \leq x \leq u\}$, where $f : R^n \to R$, and $\ell, x, u \in R^n$.

Genetic algorithms with random keys, or *random-key genetic algorithms* (RKGA), were first introduced by [1] for solving combinatorial optimization problems involving sequencing. In a RKGA, chromosomes are represented as vectors of randomly generated real numbers in the interval $[0, 1]$. A deterministic algorithm, called a *decoder*, takes as input a solution vector and associates with it a solution of the combinatorial optimization problem for which an objective value or fitness can be computed.

A RKGA evolves a population of random-key vectors over a number of iterations, called *generations*. The initial population is made up of p vectors of random-keys. Each component of the solution vector is generated independently at random in the real interval $[0, 1]$. After the fitness of each individual is computed by the decoder in generation k, the population is partitioned into two groups of individuals: a small group of p_e *elite* individuals, i.e. those with the best fitness values, and the remaining set of $p - p_e$ *non-elite* individuals. To evolve the population, a new generation of individuals must be produced. All elite individual of the population of generation k are copied without modification to the population of generation $k + 1$. RKGAs implement mutation by introducing *mutants* into the population. A mutant is simply a vector of random keys generated in the same way that an element of the initial population is generated. At each generation, a small number p_m of mutants is introduced into the population. With the p_e elite individuals and the p_m mutants accounted for in population $k + 1$, $p - p_e - p_m$ additional individuals need to be produced to complete the p individuals that make up the new population. This is done by producing $p - p_e - p_m$ offspring through the process of mating or crossover.

A *biased random-key genetic algorithm*, or BRKGA [4], differs from a RKGA in the way parents are selected for mating. While in a RKGA, [1] selects two parents at random from the entire population; in a BRKGA, each element is generated combining one element selected at random from the elite individuals set in the current population and one from the non-elite individuals set. In some cases, the second parent is selected from the entire population. Repetition in the selection of a mate is allowed and therefore an individual can produce more than one offspring. Since we require that $p_e < p - p_e$, the probability that an elite individual is selected for mating is greater than that of a non-elite individual and therefore the elite individual has a higher likelihood to pass on its characteristics to future generations. Another factor contributing to this end is *parameterized uniform crossover* [5], the mechanism used to implement mating in BRKGAs. Let $\rho_e > 0.5$ be the probability that an offspring inherits the vector component of its elite parent. Let n denote the number of components in the solution vector of an individual. For $i = 1, \ldots, n$, the i-th component $c(i)$ of the offspring c takes on the value of the i-th component $e(i)$ of the elite parent e with probability ρ_e and the value of the i-th component $\bar{e}(i)$ of the non-elite parent $\bar{e}$ with probability $1 - \rho_e$.

When the next population is complete, i.e. when it has p individuals, fitness values are computed for all of the newly created random-key vectors and the population is partitioned into elite and non-elite individuals to start a new generation.

To describe a BRKGA for continuous global optimization problems subject to box constraints, one needs only to show how solutions are encoded as vectors of random keys and how these vectors are decoded to feasible solutions of the problem:

- *Encoding a solution to a vector of random keys.* A solution is encoded as a vector $\chi = (\chi_1, ..., \chi_n)$ of size n, where χ_i is a random number in the interval $[0, 1]$, for $i = 1, \ldots, n$. The i-th component of χ corresponds to the i-th dimension of hyper-rectangle S.

- *Decoding a solution from a vector of random keys.* A decoder takes as input the vector of random keys χ and returns a solution $x \in S$ with $x_i = l_i + \chi_i \cdot (u_i - l_i)$, for $i = 1, \ldots, n$. During all decoder process, the solutions fitness are calculated by the objective function $f : S \to R$ of global optimization problem.

2. Experimental results

All experiments with BRKGA were done on a quad core Intel Core i7 processor (1.60 GHz) with Turbo Boost up to (2.80 GHz) and 16 Gb of memory, running Ubuntu 10.04 LTS released in April 2010. BRKGA heuristic was implemented in C++ and compiled with gcc version 4.4.3. The algorithm used for random-number generation is an implementation of the Mersenne Twister algorithm introduced by [6].

In this paper, we consider a problem from robot kinematics ([7–9]). We are given a 6-revolute manipulator (rigid-bodies, or links, connected together by joints, with each link connected to no more than two others), with the first link designated the base, and the last link designated the hand of the robot. The problem is to determine the possible positions of the hand, given that the joints are movable. In [9], this problem is reduced to solving a system of eight nonlinear equations $f_1(x), \ldots, f_8(x)$ in eight unknowns $x = \{x_1, \ldots, x_8\} \in [-1, 1]^8$:

$$f_1(x) = 4.731 \cdot 10^{-3} x_1 x_3 - 0.3578 x_2 x_3 - 0.1238 x_1 + x_7 - 1.637 \cdot 10^{-3} x_2 - 0.9338 x_4 - 0.3571 = 0$$

$$f_2(x) = 0.2238 x_1 x_3 + 0.7623 x_2 x_3 + 0.2638 x_1 - x_7 - 0.07745 x_2 - 0.6734 x_4 - 0.6022 = 0$$

$$f_3(x) = x_6 x_8 + 0.3578 x_1 + 4.731 \cdot 10^{-3} x_2 = 0$$

$$f_4(x) = -0.7623 x_1 + 0.2238 x_2 + 0.3461 = 0$$

$$f_5(x) = x_1^2 + x_2^2 - 1 = 0$$

$$f_6(x) = x_3^2 + x_4^2 - 1 = 0$$

$$f_7(x) = x_5^2 + x_6^2 - 1 = 0$$

Table 1. Roots of system in $[-1, 1]^8$ found by running BRKGA with seed=270001. For each root, the time in seconds and the value of objetive function $F(\cdot)$ are shown in the first collumn, as so as the components (in parenthesis) of known roots described in [7, 8].

time(sec.) $F(x)(10^{-5})$	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8
4.95	0.1658	−0.9851	0.7153	−0.6950	−0.9975	0.0638	−0.5251	−0.8557
9.79321	(0.1644)	(−0.9864)	(0.7185)	(−0.6956)	(−0.9980)	(0.0638)	(−0.5278)	(−0.8494)
7.5	0.1619	−0.9851	0.7182	−0.6946	−0.9979	−0.0616	−0.5232	0.8503
7.19678	(0.1644)	(−0.9864)	(0.7185)	(−0.6956)	(−0.9980)	(−0.0638)	(−0.5278)	(0.8494)
13.19	0.1731	−0.9827	0.7181	−0.6946	0.9973	−0.0686	−0.5195	0.8544
9.54526	(0.1644)	(−0.9864)	(0.7185)	(−0.6956)	(0.9980)	(−0.0638)	(−0.5278)	(0.8494)
5.95	0.6729	0.7394	−0.6480	−0.7641	−0.9623	−0.2706	−0.4333	0.9027
9.76283	(0.6716)	(0.7410)	(−0.6516)	(−0.7586)	(−0.9625)	(−0.2711)	(−0.4376)	(0.8992)
6.86	0.6736	0.7383	−0.6505	−0.7553	0.9634	0.2696	−0.4333	−0.9010
6.49664	(0.6716)	(0.7410)	(−0.6516)	(−0.7586)	(0.9625)	(0.2711)	(−0.4376)	(−0.8992)
6.53	0.6792	0.7328	−0.6555	−0.7553	0.9612	−0.27613	−0.4343	0.9027
9.23596	(0.6716)	(0.7410)	(−0.6516)	(−0.7586)	(0.9625)	(−0.2711)	(−0.4376)	(0.8992)
11.05	0.6768	0.7358	0.9502	−0.3132	−0.9623	0.2708	0.4002	−0.9162
9.68334	(0.6716)	(0.7410)	(0.9519)	(−0.3064)	(−0.9638)	(0.2666)	(0.4046)	(−0.9145)
15.24	0.6674	0.7427	0.9508	−0.3132	0.9661	−0.2620	0.4002	0.9156
9.81702	(0.6716)	(0.7410)	(0.9519)	(−0.3064)	(0.9638)	(−0.2666)	(0.4046)	(0.9145)
9.16	0.6792	0.7362	−0.6564	−0.7553	−0.9617	0.2745	−0.4343	−0.9008
9.1171	(0.6716)	(0.7410)	(−0.6516)	(−0.7586)	(−0.9625)	(0.2711)	(−0.4376)	(−0.8992)
98.98	0.6707	0.7462	0.9530	−0.3041	0.9644	0.2631	0.4079	−0.9107
8.55693	(0.6716)	(0.7410)	(0.9519)	(−0.3064)	(0.9638)	(0.2666)	(0.4046)	(−0.9145)
135.02	0.6646	0.7490	0.9551	−0.3015	−0.9652	−0.2625	0.4101	0.9114
9.82556	(0.6716)	(0.7410)	(0.9519)	(−0.3064)	(−0.9638)	(−0.2666)	(0.4046)	(0.9145)
354.76	0.1604	−0.9891	−0.9505	−0.3167	−0.9979	−0.0581	0.4111	0.9090
9.32723	(0.1644)	(−0.9864)	(−0.9471)	(−0.3210)	(−0.9982)	(−0.0594)	(0.4110)	(0.9116)
360.76	0.1680	−0.9844	−0.9514	−0.3167	0.9998	−0.0602	0.4098	0.9124
9.70348	(0.1644)	(−0.9864)	(−0.9471)	(−0.3210)	(0.9982)	(−0.0594)	(0.4110)	(0.9116)
409.27	0.1606	−0.9855	−0.9481	−0.3183	−0.9976	0.0554	0.4138	−0.9076
7.28536	(0.1644)	(−0.9864)	(−0.9471)	(−0.3210)	(−0.9982)	(0.0594)	(0.4110)	(−0.9116)
1204.24	0.1712	−0.9850	−0.9427	−0.3275	0.9976	0.0621	0.4052	−0.9143
8.21721	(0.1644)	(−0.9864)	(−0.9471)	(−0.3210)	(0.9982)	(0.0594)	(0.4110)	(−0.9116)
1369.81	0.1718	−0.9837	0.7178	−0.6947	0.9943	0.0687	−0.5246	−0.8519
8.63659	(0.1644)	(−0.9864)	(0.7185)	(−0.6956)	(0.9980)	(0.0638)	(−0.5278)	(−0.8494)

$$f_8(x) = x_7^2 + x_8^2 - 1 = 0$$

With this system, we form the optimization problem

$$\text{Find } x^* = \text{argmin}\{F(x) = \textstyle\sum_{i=1}^{8} f_i^2(x) \mid x \in [-1, 1]^8\}. \tag{1}$$

Since $F(x) \geq 0$ for all $x \in [-1, 1]^8$, it is easy to see that $F(x) = 0 \iff f_i(x) = 0$ for all $i \in \{1, \ldots, 8\}$. Hence, we have the following: $\exists\ x^* \in [-1, 1]^8 \ni F(x^*) = 0 \implies x^*$ is a global minimizer of problem (1) and x^* is a root of the system of equations $f_1(x), \ldots, f_8(x)$. From [7, 8], in the given domain, there are 16 known roots to this system. However, solving problem (1) 16 times using BRKGA (or any heuristic) with different starting solutions gives no guarantee of finding all 16 roots. It is entirely possible that some of the roots would be found multiple times, while others would not be found at all.

To avoid this, we modified the objective function $F(x)$, such as proposed by [3]. Suppose that heuristic has just found the k-th root (roots are denoted $x^1, \ldots, x^k$). Then BRKGA will restart, with the modified objective function given by

$$F(x) = \sum_{i=1}^{8} f_i^2(x) + \beta \sum_{j=1}^{k} e^{-\|x - x^j\|} \chi_\rho(\|x - x^j\|), \tag{2}$$

where

$$\chi_\rho(\delta) = 1, if \delta \leq \rho; 0, otherwise,$$

β is a large constant, and ρ is a small constant. This has the effect of creating an area of repulsion near solutions that have already been found by the heuristic.

For this problem, we ran BRKGA five times (a different starting random number seed for each run from 270001 to 270005) with $n = 8$, $p = 10$, $p_e = 0.2p$, $p_m = 0.1p$, $\rho_e = 0.7$, $\rho = 1$, and $\beta = 10^{10}$. At any time during a run, we define the optimality gap by $GAP = |F(x) - F(x^*)|$, where x is the current best solution found by the heuristic and x^* is the known global minimum solution. We then say that the heuristic has solved the problem if $GAP \leq \epsilon$ with $\epsilon = 0.0001$. In each case, the heuristic was able to find all 16 known roots. The average CPU time needed to find the 16 roots was 3623.27 seconds. The Table 1 illustrates one of these solutions: the 16 roots found in 4013.27 seconds by running BRKGA heuristic with seed=270001.

3. Concluding remarks

In this paper, we present the BRKGA heuristic for finding approximate solutions for continuous global optimization problems subject to box constraints. We illustrate the approach using a challenging problem with real-world applications, the robot kinematics, which nonlinear system was solved through a corresponding adaptively modified global optimization problem multiple times, each time using BRKGA with areas of repulsion around roots that have already been found. The promising results shown here illustrate the potential of BRKGA for global optimization problems.

References

[1] J.C. Bean. Genetic algorithms and random keys for sequencing and optimization. *ORSA J. on Computing*, 6: 154–160, 1994.

[2] C. A. Floudas and P. M. Pardalos. A collection of test problems for constrained global optimization algorithms. In G. Goods and J. Hartmanis, editors, *Lecture Notes in Computer Science*, volume 455. Springer Verlag, Berlin, 1990.

[3] M.J. Hirsch, C.N. Meneses, P.M. Pardalos, and M.G.C. Resende. Global optimization by continuous GRASP. In *Optimization Letters*, vol. 1, pages 201–212, 2007.

[4] J.F. Gonçalves and M.G.C. Resende. An evolutionary algorithm for manufacturing cell formation. In *Computers and Industrial Engineering*, 47:247–273, 2004.

[5] W.M. Spears and K.A. DeJong. On the virtues of parameterized uniform crossover. In *Proceedings of the Fourth International Conference on Genetic Algorithms*, pages 230–236, 1991.

[6] M. Matsumoto and T. Nishimura. Mersenne Twister: A 623-Dimensionally Equidistributed Uniform Pseudo-Random Number Generator. In *ACM Transactions on Modeling and Computer Simulation*, pages 3–30, 1998.

[7] C. A. Floudas, P. M. Pardalos, C. Adjiman, W. Esposito, Z. Gumus, S. Harding, J. Klepeis, C. Meyer and C. Schweiger. Handbook of Test Problems in Local and Global Optimization. In *Kluwer Academic Publishers, Dordrecht*, 1999.

[8] R. B. Kearfott. Some Tests of Generalized Bisection. In *ACM Transactions on Mathematical Software*, 13(3): 197–220, 1987.

[9] L. W. Tsai and A. P. Morgan. Solving the Kinematics of the Most General Six- and Five-Degree-of-Freedom Manipulators by Continuation Methods. In *Journal of Mechanisms, Transmissions, and Automation in Design*, 107:189–200, 1985.

Author Index

Topic Index