GloMIQO

Christodoulos A. Floudas, `floudas@titan.princeton.edu`**; Computer-Aided Systems Laboratory; Department of Chemical and Biological Engineering; Princeton University**

Ruth Misener, `r.misener@imperial.ac.uk`**; Centre for Process Systems Engineering; Imperial College London**

09 November 2012: GloMIQO 2.1

Contents

1 Introduction . . . 175
- 1.1 Licensing and software requirements . . . 176
- 1.2 Running GAMS/GloMIQO . . . 176

2 GAMS/GloMIQO Output . . . 176

3 Summary of GLOMIQO Options . . . 178
- 3.1 General Options . . . 178
- 3.2 Options for Solving the MILP Relaxations . . . 178
- 3.3 Options for Finding Feasible Solutions . . . 179
- 3.4 Options for Branching . . . 179
- 3.5 Options for Bounding . . . 179
- 3.6 Options for Logging to the Console . . . 179
- 3.7 Options for Addressing Special Structure . . . 180

4 Detailed Descriptions of GLOMIQO Options . . . 180

5 GloMIQO Algorithmic Features . . . 184
- 5.1 Reformulating Model Input . . . 184
- 5.2 Elucidating Special Structure . . . 185
- 5.3 Branch-and-Bound Global Optimization . . . 185

1 Introduction

The Global Mixed-Integer Quadratic Optimizer, GloMIQO (*Gló-me-ko*), considers Mixed-Integer Quadratically-Constrained Quadratic Programs (MIQCQP) of the form [44, 45]:

$$\begin{aligned} \min \quad & x^T \cdot Q_0 \cdot x + a_0 \cdot x \\ \text{s.t.} \quad & b_m^{\text{LO}} \leq x^T \cdot Q_m \cdot x + a_m \cdot x \leq b_m^{\text{UP}} \quad \forall\, m \in \{1, \ldots, M\} \\ & x \in \mathbb{R}^C \times \{0, 1\}^B \times \mathbb{Z}^I \end{aligned} \tag{MIQCQP}$$

where C, B, I, and M represent the number of continuous variables, binary variables, integer variables, and constraints, respectively. Note that this model can address quadratic continuous and/or integer terms, as well as bilinear terms of continuous-continuous, integer-continuous, and integer-integer type. We assume that it is possible to infer finite bounds $[x_i^L, x_i^U]$ on the variables participating in nonlinear terms.

Major applications of MIQCQP include quality blending in process networks, separating objects in computational geometry, and portfolio optimization in finance. Specific instantiations of MIQCQP in process networks optimization problems include: pooling problems [1, 4, 7, 13, 20, 27, 28, 29, 36, 41, 42, 43, 46, 47, 50, 57, 58], distillation sequences [2, 22, 25], wastewater treatment and total water systems [3, 5, 10, 14, 19, 26, 30, 32, 51, 52], hybrid energy systems [11, 12, 18], heat exchanger networks [15, 24], reactor-separator-recycle systems [33, 34], separation systems [56], data reconciliation [55], batch processes [39], and crude oil scheduling [35, 37, 38, 48, 49]. Computational geometry problems formulated as MIQCQP include: point packing [6, 16], cutting convex shapes from rectangles [31, 53], maximizing the area of a convex polygon [9, 8], and chip layout and compaction [17]. Portfolio optimization in financial engineering can also be formulated as MIQCQP [40, 54].

As illustrated in Figure 8.1, GloMIQO responds dynamically to elucidate and exploit special structure within user-defined MIQCQP. GloMIQO falls broadly into the category of branch-and-bound global optimization because it: generates and solves convex relaxations of the nonconvex MIQCQP that rigorously bound the global solution, finds feasible solutions via local optimization, and divides and conquers the feasible set to generate a sequence of convex relaxations converging to the global optimum [21, 23].

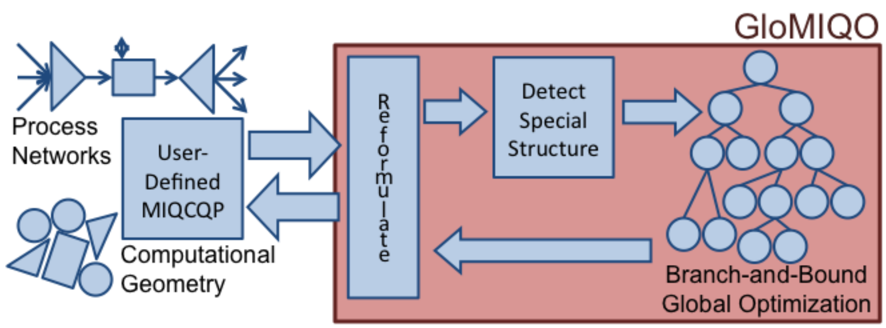


Figure 8.1: Given an MIQCQP optimization problem, GloMIQO reformulates the model, detects special structure in the reformulated MIQCQP, solves the optimization problem, and returns the model with respect to the original problem variables

1.1 Licensing and software requirements

Using GAMS/GloMIQO requires (1) a GloMIQO license, (2) a CPLEX license, and (3) a CONOPT or SNOPT license.

1.2 Running GAMS/GloMIQO

GAMS/GloMIQO solves `MIQCP`, `RMIQCP`, and `QCP` models. If GAMS/GloMIQO is not the default solver for these models, it can be called using the following command before the `solve` statement:

```
option miqcp=glomiqo, rmiqcp=glomiqo, qcp=glomiqo;
```

2 GAMS/GloMIQO Output

The log output shown below is generated using the MIQCP model `waste.gms` from the MINLPLib (http://www.gamsworld.org/minlp/minlplib/waste.htm).

```
--------------------------------------------------------------------------------
GloMIQO: Global Mixed-Integer Quadratic Optimizer; Version 2.1

         Ruth Misener and Christodoulos A. Floudas

         Computer-Aided Systems Laboratory (CASL)
         Department of Chemical & Biological Engineering; Princeton University
--------------------------------------------------------------------------------

Before Pre-processing:
         2484 Variables
                2084 Continuous
                 400 Binary
         1992 Equations

After Pre-processing:
         1036 Variables
                 636 Continuous
                 400 Binary
         1977 Equations
                 455 Linear
                1522 Nonconvex nonlinear
         1284 Bilinear/Quadratic Terms
           72 Possible Reformulation Linearization Technique (RLT) equations
             72 RLT Equations Added Outright to Formulation

Constituent Libraries:
            CPLEX  Solving relaxations
            CONOPT Finding feasible points
            LAPACK Addressing linear systems
            Boost  Bounding Intervals

--------------------------------------------------------------------------------
Time (s) Nodes explored Nodes remaining Best possible   Best found Relative Gap
--------------------------------------------------------------------------------

       7              1               1    +3.319e+02   +6.863e+02   +5.163e-01
      13              1               1    +5.934e+02   +5.989e+02   +9.186e-03
      22              1               1    +5.950e+02   +5.989e+02   +6.569e-03
      39              1               1    +5.950e+02   +5.989e+02   +6.569e-03
      56              1               1    +5.952e+02   +5.989e+02   +6.250e-03
      70              1               1    +5.962e+02   +5.989e+02   +4.497e-03
      80              1               1    +5.962e+02   +5.989e+02   +4.497e-03
      91              1               1    +5.962e+02   +5.989e+02   +4.497e-03
      99              2               3    +5.962e+02   +5.989e+02   +4.497e-03
     114              5               2    +5.962e+02   +5.989e+02   +4.497e-03
     119              9               2    +5.962e+02   +5.989e+02   +4.497e-03
     133             13               2    +5.977e+02   +5.989e+02   +2.117e-03
     137             18               1    +5.977e+02   +5.989e+02   +2.117e-03
     138             19               0    +5.989e+02   +5.989e+02   +1.000e-06

--------------------------------------------------------------------------------
Termination Status : Global minimum
Best Feasible Point: +5.989192e+02
Best Possible Point: +5.989186e+02
       Relative Gap: +1.000000e-06
```

```
Algorithm analysis :

                 19 Nodes explored
                  0 Nodes remaining

                  9 Maximum tree depth

             138.29 Total time (CPU s)
                      1.05 Pre-processing
                     52.69 Solving MILP relaxations
                     37.85 Searching for feasible solutions
                     44.22 Variable bounds tightening
                            33.01 OBBT
                            29.43 FBBT (2.93 EC; 0.06 RLT; 0.57 Factoring)
                      6.68 Branching
                             5.07 Reliability branching
-------------------------------------------------------------------------------
```

3 Summary of GLOMIQO Options

3.1 General Options

abs_opt_tol
: absolute stopping tolerance

dumpsolutions
: name of solutions index gdx file for writing alternate solutions

max_number_nodes
: node limit

max_time
: resource limit

readparams
: read secondary option file in GloMIQO syntax

rel_opt_tol
: relative stopping tolerance

trydual
: call CONOPT or SNOPT to produce duals

3.2 Options for Solving the MILP Relaxations

cplex_optfile
: read a secondary GAMS/CPLEX options file that will be applied to every LP and MILP subsolve

cut_generation_epsilon
: absolute violation threshold for separating hyperplanes

nominal_time_limit
: nominal time limit for solving MILP subproblems

populate_solution_pool
: emphasis on generating starting points

3.3 Options for Finding Feasible Solutions

feas_soln_time_limit
: time limit (s) for an NLP solve

feas_tolerance
: absolute feasibility tolerance

nlp_solver
: use CONOPT or SNOPT to find feasible solutions

3.4 Options for Branching

branching_bounds_push_away
: branch a minimum fraction away from the variable bounds

branching_weight
: branch on a convex combination of midpoint and solution

num_reliability_tests
: number of strong branching initialization tests

reliability_branching
: heuristic choice for building reliable pseudocosts

reliability_branching_mu
: score parameter for building reliability

use_reliability_branching
: use reliability branching?

3.5 Options for Bounding

fbbt_improvement_bound
: bounds reduction improvement threshold needed to exit FBBT loop

max_fbbt_iterations
: maximum number of FBBT iterations

max_obbt_iterations
: maximum number of OBBT iterations

max_time_each_obbt
: time limit (s) for each OBBT LP

obbt_improvement_bound
: bounds reduction improvement threshold

use_obbt
: use optimality-based bounds tightening?

3.6 Options for Logging to the Console

logging_freq
: how often should we log progress to the console?

logging_level
: logging information level

print_options
: print the option parameter choices used in a single run?

3.7 Options for Addressing Special Structure

adaptive_add_rlt
: use the dynamic approach to adaptively determine deep RLT cuts?

adaptive_add_rlt_tree_depth
: tree depth for heuristic that adaptively determines deep RLT cuts

add_bilinear_terms
: allow addition of nonconvex bilinear terms to generate deep RLT cuts

convexity_cuts
: derive convexity-based separating cuts for multivariable terms?

dominant_ec_only
: add only the low-dimension edge-concave aggregations introducing dominant cuts into relaxations?

eigenvector_projection_partitioning
: allow partitioning on eigenvector projections?

eigenvector_projections
: use eigenvector projections as additional cuts?

low_dim_edge_concave_agg
: use low-dimension edge-concave aggregations?

max_partitioned_quantities
: number of partitioned quantities

max_rlt_cuts
: maximum number of violated RLT cuts to add before resolving the relaxation?

naive_add_ec
: naively integrate all low-dimension edge-concave aggregations into relaxations?

naive_add_rlt
: naively add all RLT cuts to the relaxations?

number_of_partitions
: how many partitions per variable?

partitioning_scheme
: Partitioning scheme can be linear or logarithmic

piecewise_linear_partitions
: use piecewise-linear partitioning?

rlt
: find RLT variable/equation and equation/equation pairs?

use_alpha_bb
: apply globally-valid alphaBB cuts to tighten a node relaxation

use_edge_concave_dynamic
: apply locally-valid edge-concave cuts to tighten a node relaxation

4 Detailed Descriptions of GLOMIQO Options

abs_opt_tol (*real*) absolute stopping tolerance

(default = GAMS optca)

adaptive_add_rlt (*integer*) use the dynamic approach to adaptively determine deep RLT cuts?

In the first few levels of the branch-and-bound tree, query the RLT equations after solving an initial relaxation. Add violated equations to the relaxation and resolve. Track the most commonly-violated equations and include those cuts in later nodes.

(default = 1)

adaptive_add_rlt_tree_depth (*integer)* tree depth for heuristic that adaptively determines deep RLT cuts

To the specified tree depth, solve the relaxation of a node twice if RLT equations are violated. After this depth, automatically add the most commonly violated cuts to the solution of each node

Range: [1,100]

(default = 3)

add_bilinear_terms (*integer)* allow addition of nonconvex bilinear terms to generate deep RLT cuts

(default = 1)

branching_bounds_push_away (*real)* branch a minimum fraction away from the variable bounds

Range: [0,0.5]

(default = 0.1)

branching_weight (*real)* branch on a convex combination of midpoint and solution

The branching weight specifies the emphasis on the midpoint of a variable, so larger branching weights imply branching closer to the center of a variable range.

Range: [0,1]

(default = 0.25)

convexity_cuts (*integer)* derive convexity-based separating cuts for multivariable terms?

(default = 1)

cplex_optfile (*string)* read a secondary GAMS/CPLEX options file that will be applied to every LP and MILP subsolve

Gain direct access to the GAMS/CPLEX options. Specifying an options file allows, for example, the possibility of running the CPLEX subsolver with multiple threads. The value of the string should match the name of the GAMS/CPLEX options file.

cut_generation_epsilon (*real)* absolute violation threshold for separating hyperplanes

Absolute violation threshold to generate separating hyperplanes for convex multivariable terms

Range: [1e-7,10]

(default = 1e-4)

dominant_ec_only (*integer)* add only the low-dimension edge-concave aggregations introducing dominant cuts into relaxations?

(default = 1)

dumpsolutions (*string)* name of solutions index gdx file for writing alternate solutions

The GDX file specified by this option will contain a set call `index` that contains the names of GDX files with the individual solutions. For details see example model `dumpsol` in the GAMS Test Library.

eigenvector_projection_partitioning (*integer)* allow partitioning on eigenvector projections?

(default = 1)

eigenvector_projections (*integer)* use eigenvector projections as additional cuts?

(default = 1)

fbbt_improvement_bound (*real)* bounds reduction improvement threshold needed to exit FBBT loop

Range: [0,1]

(default = 0.999)

feas_soln_time_limit (*real)* time limit (s) for an NLP solve

(default = 30)

feas_tolerance (*real*) absolute feasibility tolerance

(default = 1e-6)

logging_freq (*real*) how often should we log progress to the console?

Wait at least the specified time in seconds before next output to the console

(default = 5)

logging_level (*integer*) logging information level

Log to the console at the specified level (-1: default; 0: minimal logging; 3: extensive logging)

Range: [-1,3]

(default = -1)

-1 minimal plus warnings

0 minimal

1 entering info

2 updating info

3 includes Cplex updates

low_dim_edge_concave_agg (*integer*) use low-dimension edge-concave aggregations?

(default = 1)

max_fbbt_iterations (*integer*) maximum number of FBBT iterations

Range: [1,100]

(default = 50)

max_number_nodes (*integer*) node limit

(default = GAMS nodlim)

max_obbt_iterations (*integer*) maximum number of OBBT iterations

Range: [1,100]

(default = 30)

max_partitioned_quantities (*integer*) number of partitioned quantities

Range: [0,50]

(default = 0)

max_rlt_cuts (*integer*) maximum number of violated RLT cuts to add before resolving the relaxation?

Range: [1,1000]

(default = 100)

max_time (*real*) resource limit

(default = GAMS reslim)

max_time_each_obbt (*real*) time limit (s) for each OBBT LP

Range: [1,100]

(default = 10)

naive_add_ec (*integer*) naively integrate all low-dimension edge-concave aggregations into relaxations?

(default = 0)

naive_add_rlt (*integer*) naively add all RLT cuts to the relaxations?

(default = 0)

nlp_solver (*string*) use CONOPT or SNOPT to find feasible solutions

(default = conopt)

conopt Conopt

snopt Snopt

nominal_time_limit (*real*) nominal time limit for solving MILP subproblems

Nominal time limit for solving MILP subproblems. Terminate long-running MILP subproblems over this time limit once they reach an integer feasible point

Range: [0.1,1000]

(default = 100)

num_reliability_tests (*integer*) number of strong branching initialization tests

Range: [1,100]

(default = 8)

number_of_partitions (*integer*) how many partitions per variable?

Range: [0,16]

(default = 1)

obbt_improvement_bound (*real*) bounds reduction improvement threshold

Bounds reduction improvement threshold needed to exit OBBT loop This parameter also determines whether to continue obbt in child; if the parent bound improvement is less than this threshold, then child node won't try OBBT

Range: [0,1]

(default = 0.95)

partitioning_scheme (*string*) Partitioning scheme can be linear or logarithmic

Linear partitioning uses a number of binary variables linear in the number of partitions while logarithmic partitioning uses a number of binary variables logarithmic in the number of breakpoints. Linear partitioning tends to be numerically favorable for a few breakpoints while logarithmic partitioning is better for a larger number of breakpoints.

(default = linear)

linear Linear partitioning

logarithmic Logarithmic partitioning

piecewise_linear_partitions (*integer*) use piecewise-linear partitioning?

(default = 0)

populate_solution_pool (*integer*) emphasis on generating starting points

Emphasis on generating many starting points for NLP solves using the CPLEX solution pool feature. Larger number implies more starting points.

Range: [0,4]

(default = 3)

print_options (*integer*) print the option parameter choices used in a single run?

(default = 1)

readparams (*string*) read secondary option file in GloMIQO syntax

rel_opt_tol (*real*) relative stopping tolerance

(default = GAMS optcr)

reliability_branching (*string*) heuristic choice for building reliable pseudocosts

(default = error)

error Max Error Branching
forward Forward branching
reverse Reverse branching

reliability_branching_mu (*real*) score parameter for building reliability

Range: [0,1]

(default = 0.15)

rlt (*integer*) find RLT variable/equation and equation/equation pairs?

(default = 1)

trydual (*real*) call CONOPT or SNOPT to produce duals

Spend the specified amount of time in seconds or less in producing a dual solution by calling CONOPT or SNOPT.

Range: [0,maxdouble]

(default = 5)

use_alpha_bb (*integer*) apply globally-valid alphaBB cuts to tighten a node relaxation

(default = 1)

use_edge_concave_dynamic (*integer*) apply locally-valid edge-concave cuts to tighten a node relaxation

(default = 1)

use_obbt (*integer*) use optimality-based bounds tightening?

(default = 1)

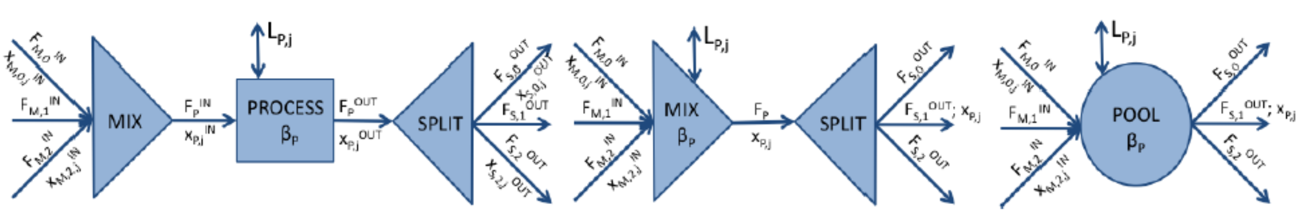
use_reliability_branching (*integer*) use reliability branching?

(default = 1)

5 GloMIQO Algorithmic Features

As illustrated in Figure 8.1, the primary algorithmic features in GloMIQO are reformulating model input (§5.1), elucidating special structure (§5.2), and branch-and-bound global optimization (§5.3) [44, 45].

5.1 Reformulating Model Input



(a) Original Model (b) Variable Elimination (c) Disaggregation

Figure 8.2: (a) Process networks problems are typically defined as a series of modular units. (b) The GloMIQO variable elimination steps transform the user model. (c) The subsequent bilinear term disaggregation further reformulates the model. The entire process is seamless and unseen by the modeler; GloMIQO reverses all transformations after solving the problem and reports results with respect to the original model in (a).

While the transformation steps illustrated in Figure 8.2 are implemented generically and applied universally, the reformulations are specifically targeted at enhancing the performance of GloMIQO on process networks problems. GloMIQO effectively transforms modular process networks problems into generalized pooling problems [42, 45]. GloMIQO may also add nonconvex bilinear terms to the model formulation to generate tight Reformulation-Linearization Technique cuts.

5.2 Elucidating Special Structure

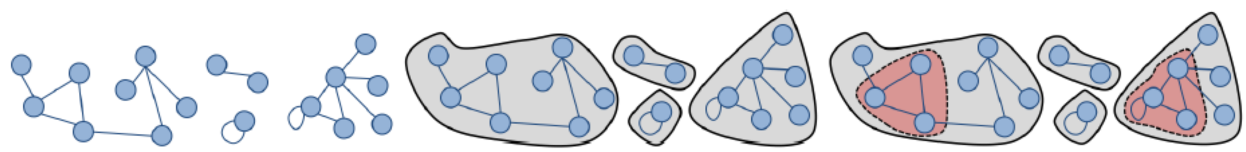
GloMIQO automatically detects: (a) Reformulation-Linearization Technique (RLT) equations that do not add nonlinear terms to MIQCQP and (b) special structure in separable multivariable terms [45].

GloMIQO considers equation/variable and equation/equation products for generating cuts and improving variable bounding. These RLT equations are updated at every node of the branch-and-bound tree:

Equation/Variable: Products of variable x_i with linear equation m $\quad(e.g., \left[a_m \cdot x - b_m^{\mathrm{UP}}\right] \cdot \left[x_i - x_i^{\mathrm{LO}}\right] \leq 0)$

Equation/Equation: Products of two linear equations m, n $\quad(e.g., -1 \cdot \left[a_m \cdot x - b_m^{\mathrm{UP}}\right] \cdot \left[a_n \cdot x - b_n^{\mathrm{UP}}\right] \leq 0)$

Observe in Section 2 that the GloMIQO preprocessor will add particularly strong RLT cuts outright the the model formulation. Modelers will significantly improve the performance of GloMIQO by writing linear constraints that can be multiplied together without increasing the number of nonlinear terms.



(a) Undirected Graph Representation (b) Separable Multivariable Terms (c) Low-Dimension Edge-Concave Aggregations

Figure 8.3: (a) Nonlinear equation m is an undirected graph with nodes representing variables and edges representing nonzero coefficients $Q_{m,i,j}$. (b) The equation is divided into separable multivariable terms by detecting disjoint vertex sets. (c) Separable multivariable terms are sum decomposable, so all high-order cuts and every bounding strategy operates on a specific multivariable term. For example, detecting three-dimensional edge-concave aggregations is illustrated in red.

As depicted in Figure 8.3, GloMIQO generates an undirected graph representation of each individual nonlinear equation m, partitions the equation into separable multivariable terms, and detects special structure including convexity and edge-concavity in the individual multivariable terms [45].

5.3 Branch-and-Bound Global Optimization

GloMIQO falls broadly into the category of branch-and-bound global optimization because it: generates and solves convex relaxations of the nonconvex MIQCQP that rigorously guarantee lower bounds on the global solution, finds feasible solutions via local optimization to bound the global solution from above, and divides and conquers the feasible set to generate a sequence of convex relaxations converging to the global optimum [21, 23].

GloMIQO **generates convex relaxations** using: termwise McCormick envelopes, low-dimensional edge-concave relaxations, eigenvector projections, piecewise-linear underestimators, outer approximation cuts for convex terms, and an adaptive implementation of the Reformulation-Linearization Technique (RLT) [27, 43, 44, 45, 46, 47].

GloMIQO **dynamically tightens convex relaxations** with cutting planes derived from edge-concave aggregations, αBB underestimators, and convex terms. Cuts are based on both individual equations and the collection of bilinear terms in MIQCQP. The branch-and-cut strategies differentiate globally-valid αBB and convex cuts from locally-valid edge-concave cuts. Previously-generated cuts are saved in a pool and applied as appropriate in the branch-and-bound tree.

GloMIQO **searches for feasible solutions** by multistarting an NLP solver.

GloMIQO **reduces the search space** using reliability branching, feasibility-based bounds tightening, optimality-based bounds tightening, RLT-based bounds tightening, and bounds tightening based on all higher-order cuts [44, 45].

GloMIQO References

[1] Adhya, N., Tawarmalani, M., Sahinidis, N.V.: A Lagrangian approach to the pooling problem. Ind. Eng. Chem. Res. **38**(5), 1965 – 1972 (1999)

[2] Aggarwal, A., Floudas, C.A.: Synthesis of general distillation sequences - nonsharp separations. Comput. Chem. Eng. **14**(6), 631–653 (1990)

[3] Ahmetović, E., Grossmann, I.E.: Global superstructure optimization for the design of integrated process water networks. AIChE Journal **57**(2), 434–457 (2011)

[4] Almutairi, H., Elhedhli, S.: A new Lagrangean approach to the pooling problem. J. Global Optim. **45**, 237 – 257 (2009)

[5] Alva-Argáez, A., Kokossis, A.C., Smith, R.: Wastewater minimisation of industrial systems using an integrated approach. Comput. Chem. Eng. **22**, S741 – S744 (1998)

[6] Anstreicher, K.M.: Semidefinite programming versus the reformulation-linearization technique for nonconvex quadratically constrained quadratic programming. J. Global Optim. **43**(2-3), 471 – 484 (2009)

[7] Audet, C., Brimberg, J., Hansen, P., Le Digabel, S., Mladenovic, N.: Pooling problem: Alternate formulations and solution methods. Manage. Sci. **50**(6), 761 – 776 (2004)

[8] Audet, C., Hansen, P., Messine, F.: The small octagon with longest perimeter. Journal of Combinatorial Theory, Series A **114**(1), 135 – 150 (2007)

[9] Audet, C., Hansen, P., Messine, F., Xiong, J.: The largest small octagon. Journal of Combinatorial Theory, Series A **98**(1), 46 – 59 (2002)

[10] Bagajewicz, M.: A review of recent design procedures for water networks in refineries and process plants. Comput. Chem. Eng. **24**, 2093 – 2113 (2000)

[11] Baliban, R.C., Elia, J.A., Floudas, C.A.: Toward novel hybrid biomass, coal, and natural gas processes for satisfying current transportation fuel demands, 1: Process alternatives, gasification modeling, process simulation, and economic analysis. Ind. Eng. Chem. Res. **49**(16), 7343–7370 (2010)

[12] Baliban, R.C., Elia, J.A., Floudas, C.A.: Optimization framework for the simultaneous process synthesis, heat and power integration of a thermochemical hybrid biomass, coal, and natural gas facility. Comput. Chem. Eng. **35**(9), 1647 – 1690 (2011)

[13] Ben-Tal, A., Eiger, G., Gershovitz, V.: Global minimization by reducing the duality gap. Math. Program. **63**, 193 – 212 (1994)

[14] Bergamini, M.L., Grossmann, I., Scenna, N., Aguirre, P.: An improved piecewise outer-approximation algorithm for the global optimization of MINLP models involving concave and bilinear terms. Comput. Chem. Eng. **32**(3), 477 – 493 (2008)

[15] Ciric, A.R., Floudas, C.A.: A retrofit approach for heat exchanger networks. Comput. Chem. Eng. **13**(6), 703 – 715 (1989)

[16] Costa, A., Hansen, P., Liberti, L.: On the impact of symmetry-breaking constraints on spatial branch-and-bound for circle packing in a square (2011). http://www.optimization-online.org/DB_HTML/2011/04/3013.html

[17] Dorneich, M.C., Sahinidis, N.V.: Global optimization algorithms for chip layout and compaction. Engineering Optimization **25**, 131–154 (1995)

[18] Elia, J.A., Baliban, R.C., Floudas, C.A.: Toward novel hybrid biomass, coal, and natural gas processes for satisfying current transportation fuel demands, 2: Simultaneous heat and power integration. Ind. Eng. Chem. Res. **49**(16), 7371–7388 (2010)

[19] Faria, D.C., Bagajewicz, M.J.: On the appropriate modeling of process plant water systems. AIChE J. **56**(3), 668 – 689 (2010)

[20] Floudas, C.A., Aggarwal, A.: A decomposition strategy for global optimum search in the pooling problem. ORSA J. Comput. **2**, 225 – 235 (1990)

[21] Floudas, C.A., Akrotirianakis, I.G., Caratzoulas, S., Meyer, C.A., Kallrath, J.: Global optimization in the 21st century: Advances and challenges. Comput. Chem. Eng. **29**, 1185 – 1202 (2005)

[22] Floudas, C.A., Anastasiadis, S.H.: Synthesis of distillation sequences with several multicomponent feed and product streams. Chem. Eng. Sci. **43**(9), 2407–2419 (1988)

[23] Floudas, C.A., Gounaris, C.E.: A review of recent advances in global optimization. J. Global Optim. **45**(1), 3 – 38 (2009)

[24] Floudas, C.A., Grossmann, I.E.: Synthesis of flexible heat-exchanger networks with uncertain flowrates and temperatures. Comput. Chem. Eng. **11**(4), 319–336 (1987)

[25] Floudas, C.A., Paules, G.E.: A mixed-integer nonlinear programming formulation for the synthesis of heat-integrated distillation sequences. Comput. Chem. Eng. **12**(6), 531 – 546 (1988)

[26] Galan, B., Grossmann, I.E.: Optimal design of distributed wastewater treatment networks. Ind. Eng. Chem. Res. **37**(10), 4036 – 4048 (1998)

[27] Gounaris, C.E., Misener, R., Floudas, C.A.: Computational comparison of piecewise-linear relaxations for pooling problems. Ind. Eng. Chem. Res. **48**(12), 5742 – 5766 (2009)

[28] Hasan, M.M.F., Karimi, I.A.: Piecewise linear relaxation of bilinear programs using bivariate partitioning. AIChE J. **56**(7), 1880 – 1893 (2010)

[29] Haverly, C.A.: Studies of the behavior of recursion for the pooling problem. ACM SIGMAP Bulletin **25**, 19 – 28 (1978)

[30] Jeżowski, J.: Review of water network design methods with literature annotations. Ind. Eng. Chem. Res. **49**(10), 4475 – 4516 (2010)

[31] Kallrath, J.: Cutting circles and polygons from area-minimizing rectangles. J. of Glob. Optim. **43**, 299 – 328 (2009)

[32] Karuppiah, R., Grossmann, I.E.: Global optimization for the synthesis of integrated water systems in chemical processes. Comput. Chem. Eng. **30**, 650 – 673 (2006)

[33] Kokossis, A.C., Floudas, C.A.: Synthesis of isothermal reactor–separator–recycle systems. Chem. Eng. Sci. **46**(5 - 6), 1361 – 1383 (1991)

[34] Kokossis, A.C., Floudas, C.A.: Optimization of complex reactor networks–II. nonisothermal operation. Chem. Eng. Sci. **49**(7), 1037 – 1051 (1994)

[35] Lee, H., Pinto, J.M., Grossmann, I.E., Park, S.: Mixed-integer linear programming model for refinery short-term scheduling of crude oil unloading with inventory management. Ind. Eng. Chem. Res. **35**(5), 1630–1641 (1996)

[36] Lee, S., Grossmann, I.E.: Global optimization of nonlinear generalized disjunctive programming with bilinear equality constraints: applications to process networks. Comput. Chem. Eng. **27**(11), 1557 – 1575 (2003)

[37] Li, J., Li, A., Karimi, I.A., Srinivasan, R.: Improving the robustness and efficiency of crude scheduling algorithms. AIChE J. **53**(10), 2659–2680 (2007)

[38] Li, J., Misener, R., Floudas, C.A.: Continuous-time modeling and global optimization approach for scheduling of crude oil operations. AIChE Journal (2011). In Press (DOI: 10.1002/aic.12623)

[39] Lin, X., Floudas, C.A.: Design, synthesis and scheduling of multipurpose batch plants via an effective continuous-time formulation. Comput. Chem. Eng. **25**(4 - 6), 665 – 674 (2001)

[40] Maranas, C.D., Androulakis, I.P., Floudas, C.A., Berger, A.J., Mulvey, J.M.: Solving long-term financial planning problems via global optimization. Journal of Economic Dynamics and Control **21**(8-9), 1405 – 1425 (1997)

[41] Meyer, C.A., Floudas, C.A.: Global optimization of a combinatorially complex generalized pooling problem. AIChE J. **52**(3), 1027 – 1037 (2006)

[42] Misener, R., Floudas, C.A.: Advances for the pooling problem: Modeling, global optimization, and computational studies. Applied and Computational Mathematics **8**(1), 3 – 22 (2009)

[43] Misener, R., Floudas, C.A.: Global optimization of large-scale pooling problems: Quadratically constrained MINLP models. Ind. Eng. Chem. Res. **49**(11), 5424 – 5438 (2010)

[44] Misener, R., Floudas, C.A.: Global optimization of mixed-integer quadratically-constrained quadratic programs (MIQCQP) through piecewise-linear and edge-concave relaxations. Math. Program. B (2011). Accepted for Publication; http://www.optimization-online.org/DB_HTML/2011/11/3240.html

[45] Misener, R., Floudas, C.A.: GloMIQO: Global Mixed-Integer Quadratic Optimizer (2011). Submitted for Publication

[46] Misener, R., Gounaris, C.E., Floudas, C.A.: Mathematical modeling and global optimization of large-scale extended pooling problems with the (EPA) complex emissions constraints. Comput. Chem. Eng. **34**(9), 1432 – 1456 (2010)

[47] Misener, R., Thompson, J.P., Floudas, C.A.: APOGEE: Global optimization of standard, generalized, and extended pooling problems via linear and logarithmic partitioning schemes. Comput. Chem. Eng. **35**(5), 876–892 (2011)

[48] Mouret, S., Grossmann, I.E., Pestiaux, P.: A new Lagrangian decomposition approach applied to the integration of refinery planning and crude-oil scheduling. Comput. Chem. Eng. DOI 10.1016/j.compchemeng.2011.03.026

[49] Mouret, S., Grossmann, I.E., Pestiaux, P.: A novel priority-slot based continuous-time formulation for crude-oil scheduling problems. Ind. Eng. Chem. Res. **48**(18), 8515–8528 (2009)

[50] Pham, V., Laird, C., El-Halwagi, M.: Convex hull discretization approach to the global optimization of pooling problems. Ind. Eng. Chem. Res. **48**, 1973 – 1979 (2009)

[51] Ponce-Ortega, J.M., El-Halwagi, M.M., Jiménez-Gutiérrez, A.: Global optimization for the synthesis of property-based recycle and reuse networks including environmental constraints. Comput. Chem. Eng. **34**(3), 318 – 330 (2010)

[52] Quesada, I., Grossmann, I.E.: Global optimization of bilinear process networks with multicomponent flows. Comput. Chem. Eng. **19**, 1219 – 1242 (1995)

[53] Rebennack, S., Kallrath, J., Pardalos, P.M.: Column enumeration based decomposition techniques for a class of non-convex MINLP problems. J. Glob. Optim. **43**(2-3), 277–297 (2009)

[54] Rios, L., Sahinidis, N.V.: Portfolio optimization for wealth-dependent risk preferences. Annals of Operations Research **177**, 63–90 (2010)

[55] Ruiz, J.P., Grossmann, I.E.: Exploiting vector space properties to strengthen the relaxation of bilinear programs arising in the global optimization of process networks. Optimization Letters **5**, 1–11 (2011)

[56] Saif, Y., Elkamel, A., Pritzker, M.: Global optimization of reverse osmosis network for wastewater treatment and minimization. Ind. Eng. Chem. Res. **47**(9), 3060 – 3070 (2008)

[57] Visweswaran, V.: MINLP: Applications in blending and pooling. In: C.A. Floudas, P.M. Pardalos (eds.) Encyclopedia of Optimization, 2 edn., pp. 2114 – 2121. Springer Science (2009)

[58] Wicaksono, D.S., Karimi, I.A.: Piecewise MILP under-and overestimators for global optimization of bilinear programs. AIChE J. **54**(4), 991 – 1008 (2008)