Projected Perspective Reformulations for MIQP problems

Antonio Frangioni
Claudio Gentile
Enrico Grande
Andrea Pacifici

Abstract. The *Perspective Relaxation* (PR) is a general approach for constructing tight approximations to Mixed-Integer NonLinear Problems with semicontinuous variables. The PR of a MINLP can be formulated either as a Mixed-Integer Second-Order Cone Program, provided that the original objective function is SOCP-representable, or as a Semi-Infinite MINLP. We show that under some further assumptions—rather restrictive, but satisfied in several practical applications—the PR of Mixed-Integer Quadratic Programs can also be reformulated as a piecewise linear-quadratic problem of roughly the same size of the standard continuous relaxation. Furthermore, if the original problem has some exploitable structure, this is typically preserved in the reformulation, allowing to construct specialized approaches for solving the PR. We report on implementing these ideas on two MIQPs with appropriate structure: a sensor placement problem and a Quadratic-cost (single-commodity) network design problem.

1. **Introduction**

Semi-continuous variables are very often found in models of real-world problems such as distribution and production planning problems [7, 10], financial trading and planning problems [8], and many others [1, 11, 12]. These are variables which are constrained to either assume the value 0, or to lie in some given polyhedron $\mathcal{P}$; when 0 belongs to $\mathcal{P}$, one incurs in a *fixed cost* to allow the variable to have a nonzero value. We will consider Mixed-Integer NonLinear Programs (MINLP) with n semi-continuous variables $x_i \in \mathbb{R}^{m_i}$ for each $i \in N = \{1, \dots, n\}$. Assuming that each $\mathcal{P}_i = \{x_i : A_i x_i \leq b_i\}$ is compact, and therefore $\{x_i : A_i x_i \leq 0\} = \{0\}$, each x_i can be modeled by using an associated binary variable y_i, leading to problems of the form

$$\min \qquad g(z) + \textstyle\sum_{i \in N} f_i(x_i) + c_i y_i \tag{1.1}$$

$$A_i x_i \leq b_i y_i \qquad i \in N \tag{1.2}$$

$$(x, y, z) \in \mathcal{O} \quad , \quad y \in \{0,1\}^n \quad , \quad x \in \mathbb{R}^m \quad , \quad z \in \mathbb{R}^q \tag{1.3}$$

where all f_i and g are closed convex functions, z is the vector of all the "other" variables, and $\mathcal{O} \subseteq \mathbb{R}^{m+n+q}$ (with $m = \sum_{i \in N} m_i$) represents all the "other" constraints of the problem.

It is known that the convex hull of a (possibly disconnected) domain such as $\{0\} \cup \mathcal{P}$ can be conveniently represented in a higher-dimensional space, which allows to derive *disjunctive cuts* for the problem [14]; this leads to defining the *Perspective Reformulation* of (1.1)—(1.3) [5, 7]

$$\min \left\{ \; g(z) + \textstyle\sum_{i \in N} y_i f_i(x_i / y_i) + c_i y_i \; : \; (1.2) \; , \; (1.3) \; \right\} \tag{1.4}$$

whose continuous relaxation is significantly stronger than that of (1.1)—(1.3), and that therefore is a more convenient starting point to develop exact and approximate solution algorithms [7, 8, 10, 12]. However, an issue with (1.4) is the high nonlinearity in the objective function due to the added fractional term. Two alternative reformulations of (1.4) have been proposed: one as a Mixed-Integer Second-Order Cone Program [15, 3, 12] (provided that the original objective function is SOCP-representable), and the other as a Semi-Infinite MILP [7]. In several cases, the latter outperforms the former in the context of exact or approximate enumerative solution approaches [9], basically due to the much higher reoptimization efficiency of active-set (simplex-like) methods for Linear and Quadratic Programs w.r.t. the available Interior Point methods for Conic Programs. However, both reformulations of (1.4) require the solution of substantially more complex continuous relaxations than the original formulation of (1.1)—(1.3); furthermore, they

Keywords: Mixed-Integer Quadratic Problems, Perspective Relaxation.

may spoil the valuable structure of the problem, such as the presence of network constraints. We show that, under some further assumptions, the PR of a Mixed-Integer *Quadratic* Program can also be reformulated as a piecewise linear-quadratic problem of roughly the same size of the standard continuous relaxation; this new reformulation is obtained by projecting each pair of variables (x_i, y_i) onto the subspace of the variables x_i, as discussed in Section 2. Moreover, if the original problem has some exploitable structure, then this structure is preserved in the reformulation, thus allowing to construct specialized approaches for solving the PR. We apply this approach to a Sensor Placement problem (Section 3) and to a Quadratic-cost (single-commodity) network design problem (Section 4), reporting numerical experiments comparing state-of-the-art, off-the-shelf MIQP solvers with the new specialized solution approach (Section 5).

2. A piecewise description of the convex envelope

Here we analyze the properties of the Perspective Reformulation under three further assumptions on the data of the original problem (1.1)—(1.3):

A1) each x_i is a *single variable* (i.e., $m_i = 1$) and each $\mathcal{P}_i$ is a bounded real interval $[0, u_i]$;

A2) the variables y_i *only* appear each in the corresponding constraint (1.2), i.e., the "other" constraints $\mathcal{O}$ do not concern the y_i;

A3) all functions are *quadratic*, i.e., $f_i(x_i) = a_i x_i^2 + b_i x_i$ (and since they are convex, $a_i \geq 0$).

While these assumptions are indeed restricting, they are in fact satisfied by most of the applications of the PR reported so far [7, 8, 11, 3, 12]. Since in this paragraph we will only work with *one* block at a time, to simplify the notation in the following we will drop the index "i". We will therefore consider the (fragment of) Mixed-Integer Quadratic Program (MIQP)

$$\min \left\{ \, ax^2 + bx + cy \; : \; 0 \leq x \leq uy \; , \; y \in \{0,1\} \, \right\} \tag{2.1}$$

and its Perspective Relaxation

$$\min \left\{ \, f(x,y) = (1/y)ax^2 + bx + cy \; : \; 0 \leq x \leq uy \; , \; y \in \{0,1\} \, \right\} \quad . \tag{2.2}$$

The basic idea behind the approach is to recast (2.2) as the minimization over $x \in [0, u]$ of

$$z(x) = \min_y f(x,y) = bx + \min_y \left\{ \, (1/y)ax^2 + cy \; : \; 0 \leq x \leq uy \; , \; y \in [0,1] \, \right\} \quad . \tag{2.3}$$

It is well-known that $z(x)$ (partial minimization of a convex function) is convex; furthermore, due to the specific structure of the problem $z(x)$ can be algebraically characterized. In particular, due to convexity of $f(x,y)$, the optimal solution $y^*(x)$ of the inner optimization problem in (2.3) is easily obtained by the solution $\tilde{y} = \tilde{y}(x)$ (if any) of the first-order optimality conditions of the unconstrained version of the problem, i.e., $\partial f(x,y)/\partial y = c - ax^2/y^2 = 0$. In fact, if $\tilde{y}$ is feasible for the problem, then it is optimal ($y^*(x) = \tilde{y}$); otherwise, $y^*(x)$ is the projection of $\tilde{y}$ over the feasible region, i.e., the extreme of the interval nearer to $\tilde{y}$ (this is where assumption A1 is used). Thus, by developing the different cases, one can construct an explicit algebraic description of $z(x) = f(x, y^*(x))$.

We start by rewriting the constraints in (2.3) as

$$(0 \leq)\; x/u \leq y \leq 1 \tag{2.4}$$

(since $u \geq x \geq 0 \Rightarrow x/u \geq 0$). We must now proceed by cases:

1) If $c \leq 0$, then $\tilde{y}$ is undefined: the derivative is always negative. Thus, there is no global minima of the unconstrained problem, and therefore $y^*(x) = 1$, yielding

$$z(x) = ax^2 + bx + c \tag{2.5}$$

2) Instead, if $c > 0$ then $\tilde{y} = x\sqrt{a/c}$ (note that we have used $x \geq 0$, $c > 0, a \geq 0$). In general, two cases can arise:

   2.1) $\tilde{y} \leq x/u \Leftrightarrow u \leq \sqrt{c/a} \Leftrightarrow y^*(x) = x/u \Rightarrow$

$$z(x) = \left(\, b + au + c/u \, \right) x \tag{2.6}$$

   2.2) $0 \geq \tilde{y} \geq x/u \Leftrightarrow u \geq \sqrt{c/a}$. This gives two further subcases

   * $(u \geq)\; x \geq \sqrt{c/a}\; (\geq 0) \Rightarrow \tilde{y} \geq 1 \Rightarrow y^*(x) = 1$,
   * $0 \leq x \leq \sqrt{c/a}\; (\leq u) \Rightarrow \tilde{y} \leq 1 \Rightarrow y^*(x) = \tilde{y}$,

finally showing that $z(x)$ is the piecewise linear-quadratic function

$$z(x) = \begin{cases} \left(b + 2\sqrt{ac} \right) x & \text{if } 0 \leq x \leq \sqrt{c/a} \\ ax^2 + bx + c & \text{if } \sqrt{c/a} \leq x \leq u. \end{cases} \tag{2.7}$$

Note that (2.7) is continuous and differentiable even at the (potential) breakpoint $x = \sqrt{c/a}$, and therefore convex (as expected).

In all the cases, $z(x)$ is a convex differentiable piecewise-quadratic function with at most 2 pieces.

3. **A sensor placement problem**

Consider the problem of optimally placing a set $N = \{1, \ldots, n\}$ of sensors to cover a given area, where deploying one sensor has a fixed cost plus a cost that is quadratic in the radius of the surface covered [1]. The problem, which is shown to be $\mathcal{NP}$-hard in [2], can be written as

$$\min \left\{ \sum_{i \in N} c_i y_i + \sum_{i \in N} a_i x_i^2 \ : \ \sum_{i \in N} x_i = 1 \ , \ 0 \leq x_i \leq y_i \ , \ y_i \in \{0,1\} \quad i \in N \right\} \tag{3.1}$$

Since we can assume $c_i > 0$ (for otherwise y_i can surely be fixed to 1), in the continuous relaxation of this problem the "design" variables y_i can be *projected* onto the x_i; that is, the y_i variables can be eliminated since at optimality $y_i = x_i$. Such a problem can be solved in $O(n \log n)$ by Lagrangian relaxation [1]; however, the bound can be weak, yielding to a large number of nodes in the enumeration tree and to a large computational time. We can improve on the bound by using the convex envelope of the single blocks of the objective function; as outlined in Section 2, we can compute this bound by means of a *single* minimization involving the piecewise-linear-quadratic functions (2.6)-(2.7). Hence, we can rewrite the problem in the form

$$\min \left\{ \sum_{j=1}^{m} b_j \chi_j + \sum_{j=1}^{m} d_j \chi_j^2 \ : \ \sum_{j=1}^{m} \chi_j = 1 \ , \ \chi_j \in [0, \alpha_j] \quad j = 1, \ldots, m \right\} \tag{3.2}$$

where $m \leq 2n$ and the coefficients b_j and d_j are as follows:

- if $\sqrt{c_i/a_i} \geq 1$ then only one new variable χ_j is generated with coefficients $b_j = a_i u_i + c_i/u_i$, $d_j = 0$, and $\alpha_j = u_i$;
- if $\sqrt{c_i/a_i} < 1$ then two new variables χ_{j_1} and χ_{j_2} are generated such that $x_i = \chi_{j_1} + \chi_{j_2}$ with $b_{j_1} = 2\sqrt{a_i c_i}$, $d_{j_1} = 0$, $\alpha_{j_1} = \sqrt{c_i/a_i}$ for the first variable and $b_{j_1} = 2\sqrt{a_i c_i}$, $d_{j_1} = a_i$, $\alpha_{j_1} = 1 - \sqrt{c_i/a_i}$ for the second variable.

This problem can be easily solved in $O(m \log m) = O(n \log n)$ with the same algorithm mentioned for the continuous relaxation of (3.1).

4. **Quadratic-cost network design**

A directed graph $G = (N, A)$ is given; for each node $i \in N$ a deficit $b_i \in \mathbb{R}$ is given indicating the amount of flow that the node demands (negative deficits indicate source nodes). Each arc $(i, j) \in A$ can be used up to a given maximum capacity u_{ij} paying a fixed cost c_{ij}. Otherwise, no cost is due if (i, j) is not installed but flow cannot pass through the arc. Additionally, if x_{ij} units of flow are sent through an installed arc (i, j), a quadratic flow cost $b_{ij} x_{ij} + a_{ij} x_{ij}^2$ is also incurred. The problem is to decide which arcs to install and how to route the flow in such a way that demands are satisfied and the total (installing + routing) cost is minimized. The problem can be written as

$$\begin{aligned} \min \quad & \sum_{(i,j) \in A} c_{ij} y_{ij} + b_{ij} x_{ij} + a_{ij} x_{ij}^2 \\ & \sum_{(j,i) \in A} x_{ji} - \sum_{(i,j) \in A} x_{ij} = b_i \quad i \in N \\ & 0 \leq x_{ij} \leq u_{ij} y_{ij} \ , \ y_{ij} \in \{0,1\} \quad (i,j) \in A \end{aligned} \tag{4.1}$$

This network design problem is $\mathcal{NP}$-hard, since it is a generalization of the sensor placement problem described in Section 3. A recent application of this general model in a Facility Location setting is given in [11, 12].

Again, since $c_{ij} > 0$ (for otherwise y_{ij} can surely be fixed to 1), in the continuous relaxation of (4.1) the design variables y_{ij} can be projected onto the x_{ij}; that is, at optimality $y_{ij} = x_{ij}/u_{ij}$.

The resulting problem can be efficiently solved by means of (convex) Quadratic Min-Cost Flow (QMCF) algorithms; however, the bound provided by the continuous relaxation is usually weak.

Applying the results of Section 2 to (4.1), a *Separable Convex-cost NonLinear MCF* problem is obtained, where the flow cost function on each arc is a piecewise linear-quadratic convex cost function. In turn, this can be rewritten as a QMCF problem

$$\begin{array}{lll} \min & \sum_{(i,j)\in A'} b'_{ij}\chi_{ij} + a'_{ij}\chi_{ij}^2 & \\ & \sum_{(j,i)\in A'} \chi_{ji} - \sum_{(i,j)\in A'} \chi_{ij} = b_i & i \in N \\ & 0 \leq \chi_{ij} \leq u'_{ij} & (i,j) \in A' \end{array} \tag{4.2}$$

on a graph $G' = (N, A')$ with the same node set and *at most* 2 times the number of arcs. For each of the original arcs (i, j), at most two "parallel" copies are constructed. If $u_{ij} \leq \sqrt{c_{ij}/a_{ij}}$ (case 2.1), then only one representative of (i, j) is constructed in G', with $b'_{ij} = b_{ij} + a_{ij}u_{ij} + c_{ij}/u_{ij}$, $a'_{ij} = 0$ and $u'_{ij} = u_{ij}$. Instead, if $u_{ij} < \sqrt{c_{ij}/a_{ij}}$ (case 2.2) then two parallel copies of the arc (i, j) have to be constructed in G': the first has $b'_{ij} = b_{ij} + 2\sqrt{a_{ij}c_{ij}}$, $a'_{ij} = 0$, and $u'_{ij} = \sqrt{c_{ij}/a_{ij}}$, while the second has $b'_{ij} = b_{ij} + 2\sqrt{a_{ij}c_{ij}}$, $a'_{ij} = a_{ij}$, and $u'_{ij} = u_{ij} - \sqrt{c_{ij}/a_{ij}}$. For this kind of "partitioned" NonLinear MCF problems—where some of the arcs have strictly convex cost functions, while the other have linear cost functions—specialized algorithms have been proposed [6]. In general, any algorithm for Convex (Quadratic) MCF problems (e.g., [4]) can be used. While codes implementing these algorithms are either not available or not very efficient in practice, the off-the-shelf solver `Cplex` turns out to be quite efficient in solving these convex QMCFs.

5. Computational Results

In order to assess the behaviour of the Projected Perspective Reformulation technique we implemented it on the two problems discussed in sections 3 and 4 within a specialized B&B where the perspective relaxation is solved by computing the projection $z(p)$ as in (2.6)-(2.7). We considered the reformulations (3.2) and (4.2) and, for their solution, we applied the specialized $O(n \log n)$ algorithm for the Sensor Placement problem and the `Cplex` quadratic solver, respectively. We compared the new approach (denoted as P^2/R) against the following ones:

- a B&C on the PR (2.2) using the Semi-Infinite MILP formulation (denoted as P/C for Perspective Cut method);
- a B&C on the PR (2.2) using the MI-SOCP formulation (denoted as CPLEX-SOCP);
- a standard B&C on the continuous relaxation (2.1) (denoted as CPLEX).

These three alternative methods have all been implemented by means of `Cplex` B&C solver. In particular, the P/C method has been coded with a `cut-callback` function. All the algorithms have been coded in `C++`, compiled with GNU `g++ 4.0.1` (with -O3 optimization option) and ran on an Opteron 246 (2 GHz) computer with 2 Gb of RAM, under Linux Fedora Core 3.

We generated 180 random instances of the Sensor Placement problem, grouped in 6 classes with 30 instances each. The first 4 classes contain instances with either 2000 or 3000 sensors and have either high or low quadratic costs. In the former ("h") , fixed costs are uniformly chosen in the interval $[1, n]$ while quadratic costs are uniformly chosen in the interval $[n, C_{max}]$, where $C_{max} \in \{10n, 20n, 30n\}$. In the latter ("l"), fixed costs are randomly generated in the interval $[n, B_{max}]$, where $B_{max} \in \{10n, 20n, 30n\}$, while quadratic costs are randomly generated in the interval $[1, n]$. The last two classes are generated starting from random instances of the Partition problem, according to the NP-hardness proof for the Sensor Placement problem in [2]. We considered 2000 and 3000 Partition items ranging in the intervals [100,1000], [500,1000], [1,100000]. Table 1 reports the obtained results.

For the Network Design Problem we generated 360 problems, grouped into 12 classes with 30 instances each, as follows:

- the underlying flow networks with 1000, 2000, or 3000 nodes have been generated by `netgen` [13], where: (i) the minimum arc cost is 1 and the maximum is randomly generated between 10 and 100, (ii) the total supply b_s is randomly generated between 100 and 1000,

name	P²/R			CPLEX			
	time	nodes	av. t/n	time	nodes	av. t/n	gap
2000-h	0.39	1	0.39	1020.51	223293	0.01	4.03
2000-l	0.09	1	0.09	101.58	3713	0.03	0.00
3000-h	0.92	1	0.92	1057.09	144406	0.01	7.18
3000-l	0.21	1	0.21	270.49	5724	0.05	0.00
PTN-2000	0.43	1	0.43	1018.13	4149	0.25	2.98
PTN-3000	1.02	1	1.02	1008.42	568	1.79	3.14
name	P/C			CPLEX - SOCP			
	time	nodes	av. t/n	time	nodes	av. t/n	gap
2000-h	47.74	924	30.43	1066.02	507	2.11	207.04
2000-l	17.02	1	17.02	49.32	38	7.60	0.00
3000-h	91.24	88	74.09	1069.73	332	3.24	412.54
3000-l	40.27	1	40.27	135.95	72	12.08	0.00
PTN-2000	94.30	6	56.93	23.79	1	23.80	0.00
PTN-3000	202.63	6	114.72	53.74	1	53.74	0.00

TABLE 1. Results for the Sensor Placement problem

and (*iii*) the minimum arc capacity is $0.05b_s$ and the maximum arc capacity is randomly generated in the interval $[0.2b_s, 0.4b_s]$;

- the fixed costs which are either low or high with respect to the linear costs generated by `netgen`, i.e., c_{ij} is uniformly generated either in $[0.5b_{ij}, b_{ij}]$ ("l") or in $[3b_{ij}, 10b_{ij}]$ ("h");
- the quadratic costs which are either low or high with respect to the linear costs generated by `netgen`, i.e., a_{ij} is uniformly generated either in $[3b_{ij}, 10b_{ij}]$ ("l") or in $[100b_{ij}, 1000b_{ij}]$ ("h").

Table 2 reports the obtained results.

name	P²/R			CPLEX			
	time	nodes	av. t/n	time	nodes	av. t/n	gap
1000-h-h	0.05	1	0.05	108.80	35630	0.28	0.00
1000-h-l	0.31	5	0.05	1037.63	324447	0.01	0.02
1000-l-h	0.05	1	0.05	163.67	46685	0.18	0.00
1000-l-l	0.32	5	0.05	1046.89	304305	0.01	0.01
2000-h-h	0.10	1	0.10	690.09	101868	0.11	0.00
2000-h-l	45.42	278	1.10	1031.75	141485	0.01	0.06
2000-l-h	0.09	1	0.09	858.22	131954	0.03	0.00
2000-l-l	8.78	63	0.10	1036.79	140877	0.01	0.04
3000-h-h	0.15	1	0.15	1041.96	88541	0.01	0.00
3000-h-l	71.02	269	0.17	1051.93	73591	0.01	0.12
3000-l-h	0.15	1	0.15	988.74	89209	0.12	0.00
3000-l-l	19.05	79	0.16	1062.45	85878	0.01	0.04
name	P/C			CPLEX - SOCP			
	time	nodes	av. t/n	time	nodes	av. t/n	gap
1000-h-h	17.03	3	10.14	967.30	26	62.86	0.01
1000-h-l	5.89	25	0.38	79.17	46	16.98	0.00
1000-l-h	8.89	4	4.60	620.77	21	38.62	0.00
1000-l-l	4.68	22	0.33	30.46	63	17.37	0.00
2000-h-h	57.09	7	13.84	895.70	8	207.60	0.01
2000-h-l	51.60	348	0.72	252.98	36	27.65	0.00
2000-l-h	42.3	6	16.57	525.35	9	63.35	0.00
2000-l-l	20.60	131	0.51	252.82	193	40.02	0.00
3000-h-h	117.30	11	18.90	564.41	2	407.97	0.01
3000-h-l	140.47	584	1.39	366.95	27	36.76	0.00
3000-l-h	101.18	12	12.01	372.16	4	89.53	0.01
3000-l-l	45.43	153	0.89	292.41	83	62.39	0.00

TABLE 2. Results for Network Design problems

For our experiments we fixed a time limit of 1000 seconds. All problems where solved at optimality within this time limit with the P^2/R and the P/C methods, therefore we do not report the gap at termination for them. For all methods, we report the running time in seconds, the number of B&B nodes and the average time for node. As expected from previous results [7, 9], the P/C method overcomes CPLEX B&C algorithm both with standard and SOCP formulations. However, the newly proposed P^2/R approach significantly overcomes the P/C method. This is mainly because of the much faster specialized solution methods used for the relaxations, which significantly reduces the effort required at each node. Furthermore, P/C approximates the true perspective relaxations by means of a finite number of cutting planes, thereby introducing some (small) approximation errors; these seem to cause the generation of more B&C nodes w.r.t. the "exact" solutions provided by P^2/R.

Bibliography

[1] A. Agnetis, E. Grande, P.B. Mirchandani, and A. Pacifici. Covering a line segment with variable radius discs. *Computers & Operations Research*, 36(5):1423–1436, 2009.

[2] A. Agnetis, E. Grande, and A. Pacifici. Demand allocation with latency cost functions. *CoRR*, abs/0810.1650, 2008.

[3] S. Aktürk, A. Atamtürk, and S. Gürel. A strong conic quadratic reformulation for machine-job assignment with controllable processing times. *Operations Research Letters*, 37(3):187–191, 2009.

[4] J. Castro and N. Nabona. An Implementation of Linear and Nonlinear Multicommodity Network Flows. *Europena J. of Operational Research*, 92:37–53, 1996.

[5] S. Ceria and J. Soares. Convex programming for disjunctive convex optimization. *Mathematical Programming*, 86:595–614, 1999.

[6] R. De Leone, R.R. Meyer, and A. Zakarian. A Partitioned ε-Relaxation Algorithm for Separable Convex Network Flow Problems. *Computational Optimization and Applications*, 12:107–126, 1999.

[7] A. Frangioni and C. Gentile. Perspective Cuts for 0-1 Mixed Integer Programs. *Mathematical Programming*, 106(2):225–236, 2006.

[8] A. Frangioni and C. Gentile. SDP Diagonalizations and Perspective Cuts for a Class of Nonseparable MIQP. *Operations Research Letters*, 35(2):181 – 185, 2007.

[9] A. Frangioni and C. Gentile. A Computational Comparison of Reformulations of the Perspective Relaxation: SOCP vs. Cutting Planes. *Operations Research Letters*, 37(3):206–210, 2009.

[10] A. Frangioni, C. Gentile, and F. Lacalandra. Tighter Appro-ximated MILP Formulations for Unit Commitment Problems. *IEEE Transactions on Power Systems*, 24(1):105–113, 2009.

[11] O. Günlük, J. Lee, and R. Weismantel. MINLP Strengthening for Separable Convex Quadratic Transportation-Cost UFL. IBM Research Report RC24213, IBM Research Division, 2007.

[12] O. Günlük and J. Linderoth. Perspective relaxation of MINLPs with indicator variables. In A. Lodi, A. Panconesi, and G. Rinaldi, editors, *Proceedings 13^{th} IPCO*, volume 5035 of *Lect. N. Comp. Sc.*, pages 1–16, 2008.

[13] D. Klingman, A. Napier, and J. Stutz. NETGEN: A program for generating large scale capacitated assignment, transportation, and minimum cost flow network problems. *Management Science*, pages 814–821, 1974.

[14] R.A. Stubbs and S. Mehrotra. A branch-and-cut method for 0-1 mixed convex programming. *Mathematical Programming*, 86:515–532, 1999.

[15] M. Tawarmalani and N.V. Sahinidis. Convex extensions and envelopes of lower semi-continuous functions. *Mathematical Programming*, 93:515–532, 2002.

Antonio Frangioni
Dipartimento di Informatica, Università di Pisa
Polo Universitario della Spezia, Via dei Colli 90,
19121 La Spezia, Italy
frangio@di.unipi.it

Claudio Gentile
IASI-CNR
Viale Manzoni 30,
00185 Roma, Italy
gentile@iasi.cnr.it

Enrico Grande
Dipartimento di Ingegneria dell'Impresa,
Università degli Studi di Roma "Tor Vergata"
via del Politecnico 1, 00133 Rome – Italy
grande@disp.uniroma2.it

Andrea Pacifici
Dipartimento di Ingegneria dell'Impresa,
Università degli Studi di Roma "Tor Vergata"
via del Politecnico 1, 00133 Rome – Italy
pacifici@disp.uniroma2.it