Relaxations of multilinear convex envelopes: dual is better than primal

Alberto Costa and Leo Liberti

LIX, Ecole Polytechnique, 91128 Palaiseau, France
{costa,liberti}@lix.polytechnique.fr

Abstract. Bilinear, trilinear, quadrilinear and general multilinear terms arise naturally in several important applications and yield nonconvex mathematical programs, which are customarily solved using the spatial Branch-and-Bound algorithm. This requires a convex relaxation of the original problem, obtained by replacing each multilinear term by appropriately tight convex relaxations. Convex envelopes are known explicitly for the bilinear case, the trilinear case, and some instances of the quadrilinear case. We show that the natural relaxation obtained using duality performs more efficiently than the traditional method.
Keywords: Global optimization, MINLP, mathematical programming.

1 Introduction

The general multilinear term is given by:

$$w(x) = x_1 \cdots x_k \tag{1}$$

for some $k \in \mathbb{N}$, and is possibly the most common nonlinear term occurring naturally in Mathematical Programming (MP) applications. As the need arises, we might also write (1) as $w(x) = x_{j_1} \cdots x_{j_k}$ with $J = \{j_1, \ldots, j_k\}$, and let $W_J = \{(x, w_J) \mid w_J = \prod_{j \in J} x_j \wedge x \in [x^L, x^U]\}$. The bilinear case is shown in Fig. 1. We let P be the set of vertices of the hyperrectangle $[x^L, x^U]$ and P_W be the lifting of P in the space spanned by (x, w_J), where, for each point $\bar{x} \in P$, the corresponding point in P_W is obtained by setting $w_J = w(\bar{x})$.

Convex envelopes for multilinear terms are available explicitly in function of x^L, x^U for $k = 2, 3$ and partly $k = 4$. Such envelopes consist of sets of constraints to be adjoined to the MP formulation. We argue in this paper that formulations obtained this way are larger and less accurate than those obtained using a dual representation of such envelopes, i.e. the convex combination of points in P_W. One further advantage of these dual envelopes is that they are the same for all k and need no special case-by-case treatment.

1.1 Contributions

The relaxations for multilinear MPs proposed in this paper, which are based on the dual, are a simple application of ideas which have been present in LP and

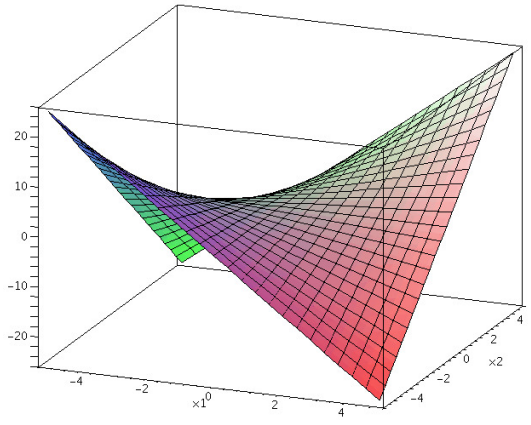

Fig. 1. The bilinear surface $w(x_1, x_2) = x_1 x_2$.

MILP theory for a long time. They constitute an original contribution insofar as they have never been tested computationally in the context of multilinear terms. This contribution would be negligible, had we not found empirically that dual relaxations provide a far superior way of relaxing multilinear MPs than "traditional relaxations". The fundamental purpose of this paper is to convey an important message: *it is possible that, until now, multilinear terms have been relaxed in the wrong way.* On the other hand, we think that the compact and elegant formulation of dual relaxations might provide a successful tool for future theoretical research: the current situation involves remarkably different formulations for each value of k and it is difficult to see how it can be exploited in a uniform way.

We remark that duality has been used in the context of multilinear relaxations in [6]. The authors exploited in several ways the same dual relaxations we propose here. The term-wise computational comparison we perform, however, which we feel is so important to convey the message above simply, clearly and unequivocally, is absent from their treatment.

1.2 Applications

Several applications involve bilinear products between binary and continuous variables that model situations where a continuous variable takes different values depending on whether a certain boolean variable is 0 or 1 [36]. In pooling and blending problems [18, 14, 1, 4, 30, 11], bilinear products ($k = 2$) occur whenever x_1 indicates a percentage and x_2 an oil flow in a pipe. The Hartree-Fock Problem [25] minimizes a quartic energy expression (involving quadrilinear terms) subject to some orthogonality constraints (involving bilinear terms). The Molecular Distance Geometry Problem [24] involves bilinear or quadrilinear terms depending on which formulation is used. General multilinear terms involving

continuous variables occur in multilinear least-squares problems [31]. In general, such products occur over bounded variables: most applications require the variables $x = (x_1, \ldots, x_k)$ to be bounded to the hyperrectangle $[x^L, x^U]$, where $x^L = (x_1^L, \ldots, x_k^L)$ and $x^U = (x_1^U, \ldots, x_k^U)$. We remark, however, that there exists an application from code debugging [26, 16] exhibiting bilinear terms $x_1 x_2$ where $x_1 \in \{0, 1\}$ and x_2 must be unbounded for the model to be correct (such variables are used to ensure that loops terminate whenever no upper bound is explicitly known for the loop counter).

1.3 Exact linearizations

It was observed in [13, 17] that if $k = 2$ and $x_1, x_2 \in \{0, 1\}$, then $w(x)$ can be replaced by an added variable $w_{12} \in [0, 1]$ whilst the *Fortet inequalities* are adjoined to the model:

$$w_{12} \leq x_1, \quad w_{12} \leq x_2, \quad w_{12} \geq x_1 + x_2 - 1. \tag{2}$$

It is easy to show that this reformulation is an exact linearization [22, 23] of the original bilinear program.

Whenever $x \in [x^L, x^U]$ and at least $k - 1$ variables out of k are constrained to be integer, the corresponding multilinear term can be linearized exactly. Each general integer variable is replaced by an aggregation of binary variables (for example choosing the value taken by the original integer variable), and the original multilinear term $w(x)$ is replaced by a sum of multilinear terms with at least $k-1$ binary variables. A sequence of $k-1$ Fortet's linearizations will then yield a Mixed-Integer Linear Programming (MILP) formulation of the original multilinear term.

1.4 Products of continuous variables

Whenever at least 2 variables in a multilinear term are continuous, exact linearizations are in general no longer possible, and one must resort to solution techniques for nonconvex programs, such as the spatial Branch-and-Bound (sBB) algorithm [12, 33, 2, 34, 21, 8]. This involves repeatedly solving the original problem and a convex relaxation thereof over appropriate sets of ranges $[x^L, x^U]$. The relaxation is obtained by replacing each multilinear term with an added variable w_J and adjoining some constraints to the formulation which define a convex relaxation of W_J. In general, the tighter these relaxations are, the more efficient the sBB will be. This has spawned a growing interest in finding constraints which define the convex and concave *envelopes* $\hat{w}(x)$ and $\check{w}(x)$ of multilinear terms. By definition, the set

$$\check{W}_J = \{(x, w_J) \mid w_J \geq \hat{w}(x) \wedge w_J \leq \check{w}(x) \wedge x \in [x^L, x^U]\} \tag{3}$$

is the *convex hull* of the set W_J. With a slight abuse of notation, the constraints on w_J appearing in the definition of $\check{W}_J$ are also called *convex envelopes* of the multilinear terms.

2 Convex envelopes of multilinear terms

It was shown in [32] that the convex envelopes of multilinear terms are *vertex polyhedral* [35], i.e. $\check{W}_J$ is a polyhedron having P_W as vertex set. This makes it possible to write the convex envelopes of multilinear terms by means of linear constraints.

2.1 McCormick's inequalities

Figure 2 shows the lower convex (left) and upper concave envelopes for the bilinear term x_1x_2, each consisting of two linear constraints. The corresponding

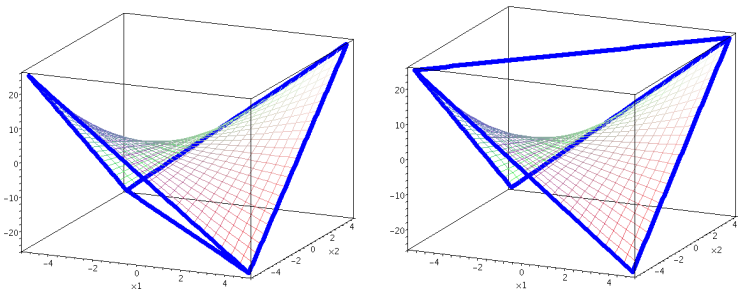

Fig. 2. Lower convex (left) and upper concave (right) envelopes for the bilinear term.

constraints:

$$w_{12} \geq x_1^L x_2 + x_2^L x_1 - x_1^L x_2^L \tag{4}$$
$$w_{12} \geq x_1^U x_2 + x_2^U x_1 - x_1^U x_2^U \tag{5}$$
$$w_{12} \leq x_1^L x_2 + x_2^U x_1 - x_1^L x_2^U \tag{6}$$
$$w_{12} \leq x_1^U x_2 + x_2^L x_1 - x_1^U x_2^L, \tag{7}$$

called *McCormick inequalities*, were first described in [27] and later shown to be envelopes in [3].

The McCormick inequalities are expressed explicitly in terms of x^L, x^U, and are therefore referred to as *explicit envelopes*. By contrast, there exists software, such as PORTA [10], which, given specific values for x^L, x^U, is able to write the corresponding constraints for the convex envelopes of the points in P_W. Finding the explicit envelopes of the multilinear term for each k is of practical interest because calling PORTA to relax each multilinear term would inefficient; and ever since McCormick's seminal paper, it has been a long-standing open question. The matter is settled in general for the case where $[x^L, x^U] = [0, 1]$ [32]; but since the use of such envelopes in the sBB algorithm implies that the bounds change at each node, this result may at best be useful only at the root node.

2.2 Meyer-Floudas inequalities

Significant progress was made with Meyer and Floudas' work [28, 29], which were able to write the explicit envelopes for the trilinear term $w(x) = x_1x_2x_3$. Their exact form depends on the relative sign of the variable bounds x^L, x^U. The paper [28] discusses 6 cases where the bound signs are equal (each case giving rise to 12 inequalities), whereas the other 9 cases are discussed in [29]. Several of these cases also involve checking nontrivial bound relations. Although Meyer and Floudas' results are conceptually simple to apply (it suffices to establish which is the case at hand, and adjoin the corresponding inequalities to the MP), the inequalities themselves are way more involved than McCormick's, and it is very easy to make mistakes when itegrating them in a computer program.

Worst of all, however, is the fact that some coefficients appearing in Meyer-Floudas inequalities involve nontrivial floating point operations. For example, the coefficient of x_1 in [29, Case 3.5, p. 133] is $\frac{x_1^U x_2^U x_3^L - x_1^L x_2^L x_3^L - x_1^U x_2^U x_3^U + x_1^U x_2^L x_3^U}{x_1^U - x_1^L}$. As is well known, floating point additions and subtractions are error-prone [20, 4.2.1]. This will yield an inaccurate constraint representation of $\breve{W}_J$; to make things worse, the simplex method will identify optimal solutions at the vertices of the polyhedron rather than at the interior, which implies that this inaccuracy will impact the optimal solution. In particular, if variables are constrained to be integer, a feasible integer solution on or near the vertex of the polyhedron might be deemed infeasible.

By contrast, each coefficient of the the McCormick inequalities ($k = 2$) only involves floating point multiplication, which is a much safer operation.

2.3 Quadrilinear terms

One of us (LL) has often heard Prof. C. Floudas state, at various Global Optimization conferences, that "the explicit envelopes of the quadrilinear terms haven't been found yet" to entice research in that direction. Accordingly, we undertook some effort in that direction in the past few years; although we failed to settle the question for $k = 4$, we managed to show how to choose the associative expression for $x_1x_2x_3x_4$ yielding the tightest convex relaxation [9, 7], and we extended this result to associative expressions of general sequences of functions.

Very recently, Ms. S. Balram of the National University of Singapore (supervised by Prof. Karimi) continued the "race" towards multilinear envelopes for higher k: her M.Sc. thesis [5] includes 44 inequalities for the simplest of the quadrilinear cases (all bounds in the nonnegative orthant). The thesis does not mention how many cases there will be in total for $k = 4$, but several coefficients of this simplest case involve even more floating point additions and subtractions than the Meyer-Floudas' inequalities, and are therefore expected to yield inaccurate formulations. As for the trilinear case, when integer variables are involved, some feasible solutions might be incorrectly deemed infeasible.

2.4 Beyond

At the moment, general envelopes for multilinear terms are computed on a per-instance basis using software such as PORTA [10] or `cdd` [15]. Since the resulting inequalities change in function of the bounds, the use of this software within the sBB algorithm, where the bounds change at each node, would be prohibitive.

2.5 Critique

From the cases $k = 3$ and $k = 4$ it appears clear that the explicit form of the inequalities describing $\breve{W}_J$, in function of x^L, x^U, considerable increases in complexity (from the point of view of floating point additions and subtractions) as k increases, thereby causing numerical instability. But this is not all: the number of such inequalities, even when they are found explicitly with PORTA, also increases, thereby yielding ever more sizable formulations. While it is known that this number increases as $O(2^k)$, the first column of Table 1 suggests that the increase is more like $O(k2^k)$. Lastly, finding explicit envelopes of multilinear terms for each separate value of k lacks elegance. The Meyer-Floudas inequalities required two papers and 15 separate cases, each with its own proof. A full treatment of the quadrilinear term along the same lines might make for a dull paper indeed. Their one redeeming feature is that they only involve the primal variables of the original formulation.

In the remainder of this paper, we shall propose *dual envelopes*: these are derived in an extremely natural way using well-known duality theory, they hold for each k, and yield more compact, accurate and numerically stable formulations. We shall henceforth refer to the convex envelopes presented in this section as *primal envelopes*.

3 Dual envelopes

The fact that the envelopes of multilinear terms are vertex polyhedral immediately suggests the following dual approach: express a point in $\breve{W}_J$ as the convex combination of the set P_W of extreme points of $\breve{W}_J$. We look for a vector λ of 2^k nonnegative Lagrange multipliers such that:

$$x = \sum_{i \leq 2^k} \lambda_i p_i \quad \wedge \quad \sum_{i \leq 2^k} \lambda_i = 1,$$

where $P_W = \{p_1, \ldots, p_{2^k}\} \subseteq R^{k+1}$. Now all that remains to do, in order to make (3) explicit envelopes, is to express the p_i's in function of x^L, x^U. To this aim, we define two parameter sequences $d \in \{0,1\}^{k2^k}$ and $b : \{0,1\}^k \to P_W$. Each d_{ij} is either 0 or 1 according as to whether the j-th component of p_i is a lower or upper bound, and $b_j(d_{ij})$ returns the correct component:

$$\forall i \leq 2^k \quad d_i = \left(\left\lfloor \frac{i-1}{2^{k-j}} \right\rfloor \bmod 2 \mid j \leq k \right) \tag{8}$$

$$\forall j \leq k \quad b_j(0) = x_j^L \ \wedge \ b_j(1) = x_j^U. \tag{9}$$

We relax the k-linear term $w(x) = x_1 \cdots x_k$ as follows. We add 2^k new nonnegatively constrained variables $\lambda_i \geq 0$ (for $i \leq 2^k$) and $k+1$ new constraints:

$$\forall j \leq k \quad x_j = \sum_{i \leq 2^k} \lambda_i b_j(d_{ij}) \tag{10}$$

$$w = \sum_{i \leq 2^k} \lambda_i \prod_{j \leq k} b_j(d_{ij}) \tag{11}$$

$$\sum_{i \leq 2^k} \lambda_i = 1. \tag{12}$$

Let $\bar{W}_J = \{(x, w, \lambda) \mid (10) - (12) \wedge \lambda \geq 0\}$. It is well known that the projection of $\bar{W}_J$ on the (x, w) variables is precisely $\breve{W}_J$.

The dual envelope adds exactly 2^k new nonnegative variables and $k + 1$ new constraints to the formulation. Table 1 reports the size increases for the cases $k \in \{2, 3, 4, 5\}$. Cases $k \in \{2, 3, 4\}$ refer to the McCormick, Meyer-Floudas and Balram [5] inequalities. The statistic for $k = 5$ is taken from [5], but devised computationally using a method similar to PORTA.

k	*Primal*	*Dual*
2	4	7
3	12	12
4	44	21
5	130	38

Table 1. Per-multilinear-term size increase (new constraints and variables) for primal and dual envelopes.

3.1 Relaxations

Given a multilinear MP P, a relaxation can be obtained by replacing each multilinear term with its corresponding primal or dual envelope. This term-wise fashion of construction relaxations was initially proposed in [27], refined and exploited in a sBB in [34], and further improved in [8]. We shall call relaxations constructed with primal envelopes *primal relaxations* and those constructed with dual envelopes *dual relaxations*.

4 Computational results

Our tests, carried out on an Intel Xeon CPU at 2.66GHz with 24GB RAM, show that dual relaxations can be solved faster (as the formulation size increases) than primal relaxations, and are also more stable. We measure *speed* by simply solving the primal and dual relaxations for the same original problem using the CPLEX

12.2 [19] simplex solver, and comparing CPU times. We define a method *stable* when its CPU time increase looks empirically proportional to the increase in formulation size. We measure stability by enforcing integrality constraints on some of the problem variables: this yields a *dual MILP relaxation* and a *primal MILP relaxation*. Both are solved with the CPLEX 12.2 MILP solver, and the CPU times are recorded and compared. This is meant to simulate the behaviour of these relaxations in a Branch-and-Bound setting. It turns out that the running times of the MILP solver on the dual MILP relaxation is proportional to the relaxation size, whereas it varies wildly for the primal MILP relaxation.

We generated 2500 random multilinear nonseparable MINLPs, involving linear, bilinear and trilinear terms. For each such MINLP P, we generated the primal relaxation R_P, the dual relaxation Λ_P, the primal MILP relaxation R'_P and the dual MILP relaxation Λ'_P. We let n (the number of original variables) vary in $\{10, 20\}$. For $n = 10$ we let the number of bilinear terms β vary in $\{0, 10, 13, 17, 21, 25, 29, 33\}$ and of trilinear terms τ in $\{0, 10, 22, 34, 36, 58, 71, 83\}$ (of course the case $\beta + \tau = 0$ is excluded). For $n = 20$, we let β vary in $\{0, 20, 38, 57, 76, 95, 114, 133\}$ and τ in $\{0, 20, 144, 268, 393, 517, 642, 766\}$. For each combination of the triplet (n, β, τ) we generated 16 random instances. The variable bounds, chosen at random, were all of magnitude $\pm 1 \times 10^6$.

The CPU time results comparing R_P, Λ_P are given in Fig. 3-4. The horizontal axis is marked by the instance ID. Each recognizable "block" corresponds to a fixed value of β. Since bilinear terms give rise to fewer relaxation variables/constraints than trilinear ones, the formulation size is strongly proportional to τ and weakly proportional to β. Although for $n = 10$ (Fig. 3) the CPU

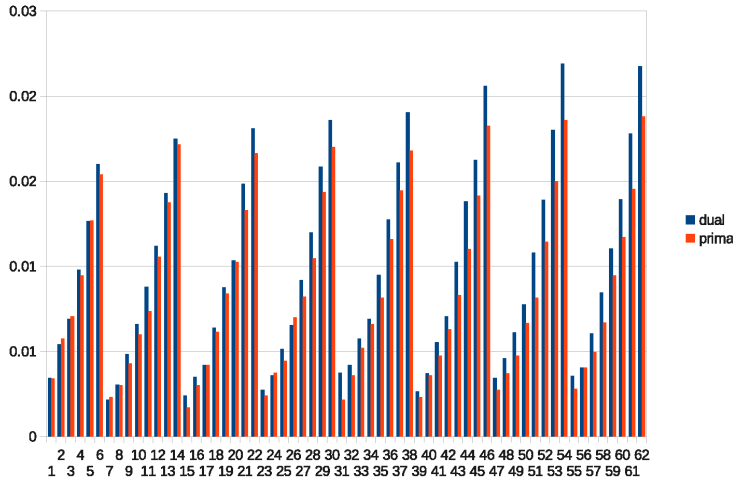

Fig. 3. CPU time averages over each 16-instance set with given (n, β, τ) with $n = 10$.

time is very slightly in favour of the primal relaxation, the situation changes visibly for $n = 20$ (Fig. 4). Although the CPU times differ, we cannot infer much

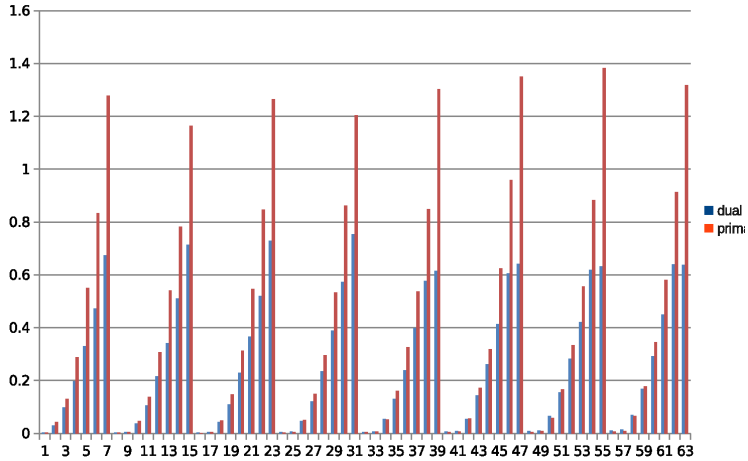

Fig. 4. CPU time averages over each 16-instance set with given (n, β, τ) with $n = 20$.

on the comparative stability of the two methods.

The CPU time results comparing R'_P, Λ'_P are given in Fig. 5-6. The CPU differences are decidedly striking in the case $n = 10$ and even excessively so for the case $n = 20$. The CPU time taken to solve primal relaxations is far from proportional to formulation size, whereas the stability associated to the dual relaxation is remarkable.

References

1. N. Adhya, M. Tawarmalani, and N.V. Sahinidis. A Lagrangian approach to the pooling problem. *Industrial and Engineering Chemistry Research*, 38:1956–1972, 1999.
2. C.S. Adjiman, S. Dallwig, C.A. Floudas, and A. Neumaier. A global optimization method, αBB, for general twice-differentiable constrained NLPs: I. Theoretical advances. *Computers & Chemical Engineering*, 22(9):1137–1158, 1998.
3. F.A. Al-Khayyal and J.E. Falk. Jointly constrained biconvex programming. *Mathematics of Operations Research*, 8(2):273–286, 1983.
4. C. Audet, J. Brimberg, P. Hansen, S. Le Digabel, and N. Mladenović. Pooling problem: Alternate formulations and solution methods. *Management Science*, 50(6):761–776, 2004.
5. S. Balram. Crude transshipment via floating, production, storage and offloading platforms. Master's thesis, Dept. of Chemical and Biomolecular Engineering, National University of Singapore, 2010.

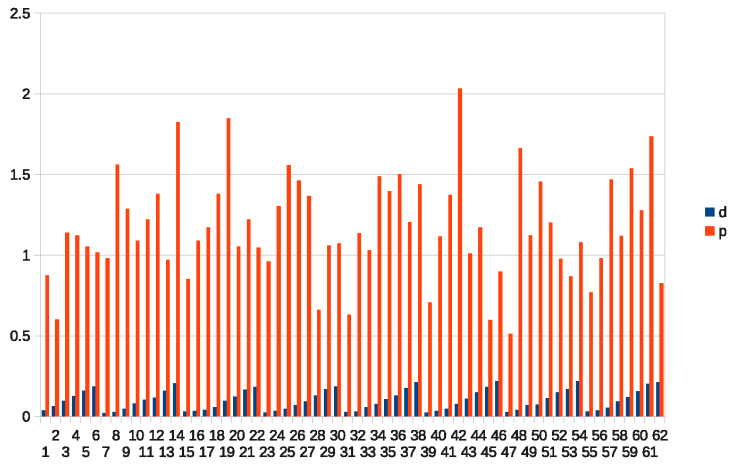

Fig. 5. CPU time averages over each 16-instance set with given (n, β, τ) with $n = 20$.

6. Xiaowei Bao, Nikolaos Sahinidis, and Mohit Tawarmalani. Multiterm polyhedral relaxations for nonconvex, quadratically constrained quadratic programs. *Optimization Methods and Software*, 24(4-5):485–504, 2009.
7. P. Belotti, S. Cafieri, J. Lee, L. Liberti, and A. Miller. On the composition of convex envelopes for quadrilinear terms. In P. Pardalos et al., editor, *Optimization and Optimal Control*, Nonconvex Optimization and Its Application. Springer, New York, submitted.
8. 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.
9. S. Cafieri, J. Lee, and L. Liberti. On convex relaxations of quadrilinear terms. *Journal of Global Optimization*, 47:661–685, 2010.
10. T. Christof and A. Löbel. The porta manual page. Technical Report v. 1.4.0, ZIB, Berlin, 1997.
11. C. D'Ambrosio, J. Linderoth, and J. Luedtke. Valid inequalities for the pooling porblem with binary variables. In O. Günluk and G. Woeginger, editors, *IPCO*, volume 6655 of *LNCS*, pages 117–129, Heidelberg, 2011. Springer.
12. J.E. Falk and R.M. Soland. An algorithm for separable nonconvex programming problems. *Management Science*, 15:550–569, 1969.
13. R. Fortet. Applications de l'algèbre de Boole en recherche opérationelle. *Revue Française de Recherche Opérationelle*, 4:17–26, 1960.
14. L.R. Foulds, D. Haughland, and K. Jornsten. A bilinear approach to the pooling problem. *Optimization*, 24:165–180, 1992.
15. K. Fukuda and A. Prodon. Double description method revisited. In M. Deza, R. Euler, and Y. Manoussakis, editors, *8th Franco-Japanese and 4th Franco-Chinese Conference on Combinatorics and Computer Science*, volume 1120 of *LNCS*, pages 91–111, London, 1995. Springer.

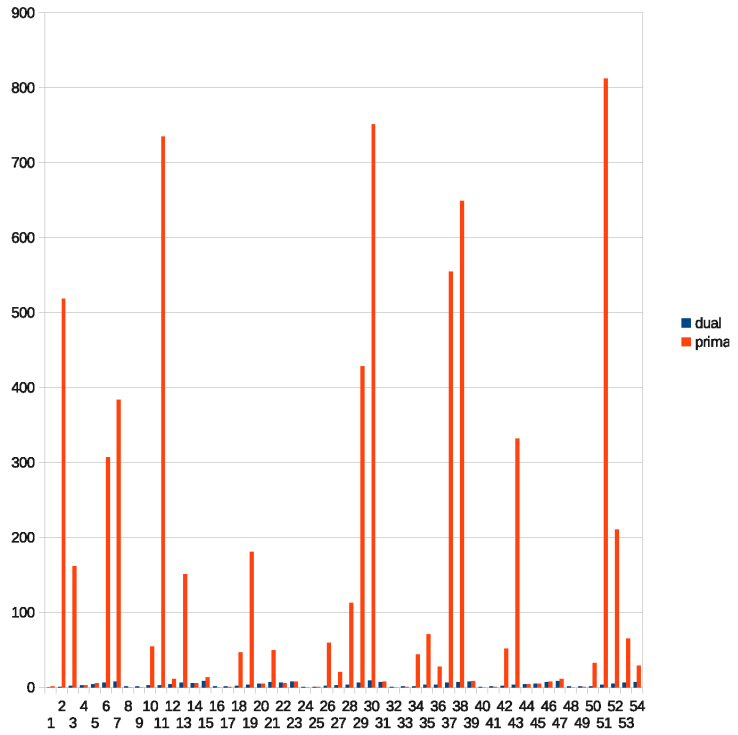

Fig. 6. CPU time averages over each 16-instance set with given (n, β, τ) with $n = 20$.

16. E. Goubault, S. Le Roux, J. Leconte, L. Liberti, and F. Marinelli. Static analysis by abstract interpretation: a mathematical programming approach. In A. Miné and E. Rodriguez-Carbonell, editors, *Proceeding of the Second International Workshop on Numerical and Symbolic Abstract Domains*, volume 267(1) of *Electronic Notes in Theoretical Computer Science*, pages 73–87. Elsevier, 2010.
17. P.L. Hammer and S. Rudeanu. *Boolean Methods in Operations Research and Related Areas*. Springer, Berlin, 1968.
18. C.A. Haverly. Studies of the behaviour of recursion for the pooling problem. *ACM SIGMAP Bulletin*, 25:19–28, 1978.
19. IBM. *ILOG CPLEX 12.2 User's Manual*. IBM, 2010.
20. D.E. Knuth. *The Art of Computer Programming, Part II: Seminumerical Algorithms*. Addison-Wesley, Reading, MA, 1981.
21. L. Liberti. Writing global optimization software. In L. Liberti and N. Maculan, editors, *Global Optimization: from Theory to Implementation*, pages 211–262. Springer, Berlin, 2006.
22. L. Liberti. Reformulations in mathematical programming: Definitions and systematics. *RAIRO-RO*, 43(1):55–86, 2009.
23. L. Liberti, S. Cafieri, and F. Tarissan. Reformulations in mathematical programming: A computational approach. In A. Abraham, A.-E. Hassanien, P. Siarry, and

A. Engelbrecht, editors, *Foundations of Computational Intelligence Vol. 3*, number 203 in Studies in Computational Intelligence, pages 153–234. Springer, Berlin, 2009.
24. L. Liberti, C. Lavor, A. Mucherino, and N. Maculan. Molecular distance geometry methods: from continuous to discrete. *International Transactions in Operational Research*, 18:33–51, 2010.
25. L. Liberti, C. Lavor, M.A. Chaer Nascimento, and N. Maculan. Reformulation in mathematical programming: an application to quantum chemistry. *Discrete Applied Mathematics*, 157:1309–1318, 2009.
26. L. Liberti, S. Le Roux, J. Leconte, and F. Marinelli. Mathematical programming based debugging. In R. Mahjoub, editor, *Proceedings of the International Symposium on Combinatorial Optimization*, volume 36 of *Electronic Notes in Discrete Mathematics*, pages 1311–1318, Amsterdam, 2010. Elsevier.
27. G.P. McCormick. Computability of global solutions to factorable nonconvex programs: Part I — Convex underestimating problems. *Mathematical Programming*, 10:146–175, 1976.
28. C.A. Meyer and C.A. Floudas. Trilinear monomials with positive or negative domains: Facet s of the convex and concave envelopes. In C.A. Floudas and P.M. Pardalos, editors, *Frontiers in Global Optimization*, pages 327–352. Kluwer Academic Publishers, Amsterdam, 2003.
29. C.A. Meyer and C.A. Floudas. Trilinear monomials with mixed sign domains: Facets of the convex and concave envelopes. *Journal of Global Optimization*, 29:125–155, 2004.
30. R. Misener and C. Floudas. Global optimization of large-scale generalized pooling problems: quadratically constrained minlp models. *Industrial Engineering and Chemical Research*, 49:5424–5438, 2010.
31. P. Paatero. The multilinear engine: A table-driven, least squares program for solving multilinear problems, including the n-way parallel factor analysis model. *Journal of Computational and Graphical Statistics*, 8(4):854–888, 1999.
32. A. Rikun. A convex envelope formula for multilinear functions. *Journal of Global Optimization*, 10(4):425–437, 1997.
33. H.S. Ryoo and N.V. Sahinidis. Global optimization of nonconvex NLPs and MINLPs with applications in process design. *Computers & Chemical Engineering*, 19(5):551–566, 1995.
34. E. Smith and C. Pantelides. A symbolic reformulation/spatial branch-and-bound algorithm for the global optimisation of nonconvex MINLPs. *Computers & Chemical Engineering*, 23:457–478, 1999.
35. F. Tardella. Existence and sum decomposition of vertex polyhedral convex envelopes. Technical report, Facoltà di Economia e Commercio, Università di Roma "La Sapienza", 2007.
36. H.P. Williams. *Model Building in Mathematical Programming*. Wiley, Chichester, 4th edition, 1999.