Feasibility check for the distance geometry problem: an application to molecular conformations *

Agostinho Agra[1], Rosa Figueiredo[2], Carlile Lavor[3], Nelson Maculan[4], António Pereira[5], and Cristina Requejo[6]

[1] *Department of Mathematics, University of Aveiro, 3810-193 Aveiro, Portugal.* aagra@ua.pt
[2] *LIA, University of Avignon, 84911, Avignon, France.* rosa.figueiredo@univ-avignon.fr
[3] *Department of Applied Mathematics, University of Campinas, 13081-970 Campinas, Brazil.* clavor@ime.unicamp.br
[4] *Federal University of Rio de Janeiro, C.P. 68511, 21945-970 Rio de Janeiro, Brazil.* maculan@cos.ufrj.br
[5] *Department of Mathematics, University of Aveiro, 3810-193 Aveiro, Portugal.* antoniop@ua.pt
[6] *Department of Mathematics, University of Aveiro, 3810-193 Aveiro, Portugal.* crequejo@ua.pt

Abstract

The Distance Geometry Problem (DGP) consists in finding an embedding in a metric space of a given weighted undirected graph such that for each edge in the graph, the corresponding distance in the embedding belongs to a given distance interval. We discuss the relationship between the existence of a graph embedding in a Euclidean space and the existence of a graph embedding in a lattice.

Different approaches, including two integer programming models (IP) and a constraint programming (CP) approach are presented to test feasibility of the DGP. The two IP models are improved with the inclusion of valid inequalities and the CP approach is improved with the use of an algorithm to perform a domain reduction.

The main motivation to this work is to derive new pruning devices within branch and prune algorithms for instances occurring in real applications related to determination of molecular conformations, which is a particular case of the DGP. A computational study based on a set of small sized instances from molecular conformations is reported. This study compares the running times of the different approaches to check feasibility.

***Keywords*:** Distance Geometry Problem; Graph Embedding; Integer Programming; Constraint Programming.

1 Introduction

The Distance Geometry Problem arises is many practical applications. One of the best known ones is the protein structure calculation, which is a major problem in computational biology [4]. The Molecular Distance Geometry Problem (MDGP) arises in nuclear magnetic resonance (NMR) spectroscopy analysis, which provides a set of inter-atomic distances for certain pairs of atoms of a given protein [22]. The question is how to use this set of distances in order to calculate

*Published in *International Transactions in Operational Research*, Volume 24, Issue 5, Pages 1023–1040, 2017. doi: 10.1111/itor.12283

the spatial positions of the atoms forming the molecule [3]. A very recent review about distance geometry problems and applications is given in [13].

A simple weighted undirected graph $G = (V, E, d)$ can be associated to the MDGP, where V represents the set of atoms, E models the set of atom pairs for which a Euclidean distance is available, and the function $d : E \to \mathbb{R}^+$ assigns distance values to each atom pair. The MDGP can then be formally defined as the following: given a weighted simple undirected graph $G = (V, E, d)$, find a function $x : V \to \mathbb{R}^3$ such that

$$||x_u - x_v|| = d_{uv} \quad \forall \{u, v\} \in E. \tag{1}$$

When G is a complete graph (all the distances are given), a unique three-dimensional structure can be determined by a linear time algorithm [5]. Otherwise, MDGP is NP-hard [18].

The MDGP can be naturally formulated as a nonlinear global minimization problem, where the objective function is given by

$$f(x_1, \ldots, x_n) = \sum_{\{u,v\} \in E} (||x_u - x_v||^2 - d_{uv}^2)^2.$$

Assuming that all the distances are correctly given, $x_1, \ldots, x_n \in \mathbb{R}^3$ solve the problem if and only if $f(x_1, \ldots, x_n) = 0$.

For MDGP instances provided by NMR experiments, G is not complete and there may exist errors in the given distances. Therefore, a more practical definition of the MDGP (denoted as iMDGP in the next sections) is to replace conditions (1) by

$$d_{uv}^L \leq ||x_u - x_v|| \leq d_{uv}^U \quad \forall \{u, v\} \in E, \tag{2}$$

where d_{uv}^L and d_{uv}^U are, respectively, lower and upper bounds on the distance between atom u and atom v.

Several algorithms have been proposed for the solution of the MDGP, most of them based on a search in a continuous space [12]. By exploring some rigidity properties of the graph G, the search space can be discretized and an efficient algorithm called Branch and Prune ($B\&P$) can be used, particularly when the given distances are precise [9].

The main idea behind the discretization is that the intersection among three spheres in the three-dimensional space can produce at most two points in the hypothesis their centers are not aligned. Consider four atoms u_1, u_2, u_3, and v. If the coordinates for u_1, u_2, u_3 are known, as well as the distances $d_{u_1v}, d_{u_2v}, d_{u_3v}$, then three spheres centered at u_i with radius $d_{u_i,v}, i = 1, 2, 3$ can be defined and their intersection provides at most two possible positions for the atom v. The definition of an ordering on the atoms of the protein satisfying such conditions suggests a recursive search on a binary tree containing the potential coordinates for the atoms of the molecule [10]. The binary tree of possible solutions is explored starting from its top, where the first three atoms are positioned, and by placing one node per time. At each step, at most two possible positions for the current node v are computed, and two new branches are added to the tree. As a consequence, the size of the binary tree can get very large quite quickly, but the presence of additional distances provided by NMR can help in verifying the feasibility of the computed positions. As soon as a position is found to be infeasible, the corresponding branch can be pruned and the search can be backtracked.

When all the distances are exact, each node of the tree refers to one atomic position. However, when one of such distances is represented by an interval, a curve in the three-dimensional space is associated to the corresponding node in the tree [10]. In this case, it is possible to choose some distances from the available interval that are associated to atomic positions on the curves. While this strategy was proved to work well for relatively small-sized instances [10], it does not have the same efficiency as the strategy applied to the case with exact distances [9].

In this paper we consider a lattice representation of the molecule. This lattice is a discretization of $\mathbb{R}^3$. Lattice models have been used in the past for related protein problems, see [19]. In order to handle with errors induced by the discretization we assume interval distances, although we report computational results for the cases of exact and interval distances. Notice that the knowledge of an exact distance d_{uv} is equivalent to considering $d^L_{uv} = d^U_{uv} = d_{uv}$ in expression (2). Our goal is to construct and test discrete optimization models (integer programming models [21]) to decide whether there is an embedding of a graph on a lattice satisfying a set of interval distances.

Let V' represent a neighborhood of a node of the original graph G and let $G' = G[V']$ be the subgraph of the original graph (corresponding to the entire molecule) induced by V'. The main idea is to locally use discrete optimization models in order to decide whether there is an embedding of $G' = (V', E', d')$ in the lattice. These models can then be used within branch and prune algorithms to solve the MDGP. While in the previous strategies, for exact distances [9] and for interval distances [10], the pruning is done using only the known distances of a current node to the previous fixed nodes, with our strategy we aim to prune the search tree by using all relevant information on the current node. In particular, by using a neighborhood of a node, it is possible to incorporate distance information on nodes that have not been explored yet, that is, to incorporate the distances to nodes that will be explored in deeper stages of the search tree. Doing so, it may be possible to identify infeasibility sooner and therefore prune the search tree in early stages.

The paper has the following contributions. (i) Propose and compare integer programming models, and one constraint programming model, to decide whether there is a feasible embedding of graph G in a lattice. This study can be applied to any embedding problem and not only for the MDGP. (ii) Provide several improvements on these models such as domain reduction and inclusion of valid inequalities. (iii) Include a new pruning test based on implicit distance information between atoms when the NMR provides no information that relates these atoms.

The paper is organized as follows. Definitions, notations to be used throughout the paper, and relation between embedding problems, are given in Section 2. In Section 3, several integer programming models are introduced and discussed. Section 4 presents a constraint programming model and, in Section 5, several improvements on the models introduced are discussed. In Section 6, we report the computational experiments conducted to compare the models. Additionally, we show that implicit distances from NMR can be used to reduce the size of the search tree of $B\&P$ algorithms for the MDGP. Finally, some conclusions and future directions of research are presented in Section 7.

2 Embedding problems and their relation

Consider an undirected graph $G = (V, E)$ with node set V and edge set E. To each edge $e = \{u, v\} \in E$ associate a positive interval $I_e = [d^L_{uv}, d^U_{uv}], 0 < d^L_{uv} \leq d^U_{uv}$, and denote by $I(E)$ the set of positive intervals $I(E) = \{I_e : e \in E\}$.

An embedding of G in a set $R \subseteq \mathbb{R}^3$, $x : V \to R$, is called feasible if

$$\forall \{u, v\} \in E, d^L_{uv} \leq \|x_u - x_v\| \leq d^U_{uv}. \tag{3}$$

The iMDGP can be reformulated as follows.

Problem 1 (iMDGP)**.** *Given a simple undirected graph $G = (V, E)$ and a set $I(E)$ of positive intervals, is there a feasible embedding of G in $\mathbb{R}^3$?*

The intervals (lower and upper bounds) are defined according to the NMR experiments. These experiments give distance information (intervals) for atoms that are close enough. In general, distances related to covalent bonds and covalent angles are considered fixed and defined by real numbers [22]. The iMDGP has been considered before in [6, 10, 15].

A very closely related problem occurs when we ask whether there exists a feasible embedding of G in a discrete set $B \subseteq \mathbb{R}^3$, which is a grid (lattice) limited by a box centered at the origin $(0, 0, 0)$. Let $\alpha > 0$ be the distance between consecutive lines of the grid. For a given grid dimension Δ, define $K = \lceil \frac{\Delta}{\alpha} \rceil$ as the number of lines in each positive/negative coordinate. Each side of the box has length $2K \times \alpha$, and the set of points in the box, denoted by B, is defined by: $B = L \times L \times L$ where $L = \{\alpha i : i \in Q\}$ and $Q = \{-K, -(K-1), \ldots, -2, -1, 0, 1, 2, \ldots, (K-1), K\}$.

Now, we seek an embedding of G in B defined by placing each node $v \in V$ on the nearest point $n(x_v)$ of the grid B when v is placed on $x_v \in \mathbb{R}^3$, see Figure 1. This new problem can be defined as follows.

Problem 2 (iMDGP-grid)**.** *Given a simple undirected graph $G = (V, E)$ and a set $I(E)$ of positive intervals, is there a feasible embedding of G in B?*

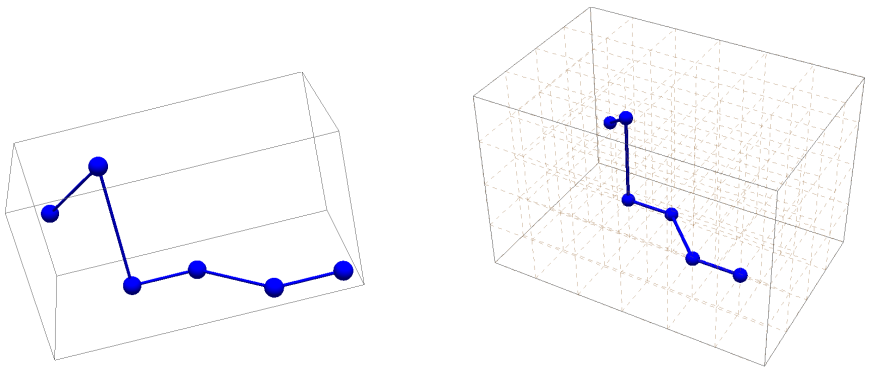

Figure 1: (a) A simple representation of a Molecule. (b) The Molecule representation embedded in the grid.

Next, we discuss some relations between the two problems. Clearly, even if there is a feasible embedding of $G = (V, E)$ in $\mathbb{R}^3$, it may be impossible to find a feasible embedding of G in B considering the same set of intervals $I(E)$. In order to ensure that any embedding of G in $\mathbb{R}^3$ corresponds to an embedding of G in B, we can measure the error induced by the discretization when each $v \in V$ placed in $x_v = (x_{v1}, x_{v2}, x_{v3}) \in \mathbb{R}^3$ is placed in the nearest point $n(x_v)$ in the grid B given by,

$$n(x_v) = \arg\min_{y \in B} \|x_v - y\| = (\alpha \times \text{round}(x_{v1}/\alpha), \alpha \times \text{round}(x_{v2}/\alpha), \alpha \times \text{round}(x_{v3}/\alpha)).$$

As $x_{vi} - \alpha \times \text{round}(x_{vi}/\alpha) \leq \frac{\alpha}{2}, i = 1, 2, 3$, then

$$\|x_v - n(x_v)\| = \sqrt{\sum_{i=1}^{3} (x_{vi} - \alpha \times \text{round}(x_{vi}/\alpha))^2} \leq \sqrt{3(\frac{\alpha}{2})^2} = \frac{\sqrt{3}}{2}\alpha.$$

So, given a feasible embedding x, the maximum distance error resulting from the placement of u and v to the nearest positions, respectively, $n(x_u)$ and $n(x_v)$ in B, is given by $|\|x_u - x_v\| - \|n(x_u) - n(x_v)\|| \leq \sqrt{3}\alpha$. As a consequence of the discussion above we can state the following trivial result as a remark.

Remark 1. *Consider a simple undirected graph $G = (V, E)$ and a set $I(E)$ of positive intervals. If x is a feasible embedding of G in $\mathbb{R}^3$, then $n(x_v)$, $v \in V$, is a feasible embedding of G in B, for the set $\bar{I}(E)$ where $\bar{I}_e = [\bar{d}^L_{uv}, \bar{d}^U_{uv}]$ is given by*

$$\bar{d}^L_{uv} = d^L_{uv} - \sqrt{3}\alpha > 0, \tag{4}$$

$$\bar{d}^U_{uv} = d^U_{uv} + \sqrt{3}\alpha. \tag{5}$$

Remark 1 implies that if there is no feasible embedding of G in B for the set $\bar{I}(E)$, then there is no feasible embedding of G in $\mathbb{R}^3$ for the set $I(E)$.

The size of the iMDGP-grid instances depend on the size of the grid, $|B|$, and on the number of nodes of G, $|V|$. For a given Δ (which defines the box size), as α increases, the number of points of the grid decreases and the same happens to the size of the instances of iMDGP-grid. However, as α increases the tolerance given for each interval (in order to ensure that no feasible solution is lost when moving from iMDGP to iMDGP-grid), denoted by ϵ, also increases. Hence, when increasing α, an instance of iMDGP for which there is no feasible embedding can be converted into an instance of iMDGP-grid having a feasible embedding. Thus, when there is no feasible embedding for an instance of iMDGP, the ideal choice of α is the largest α that leads to an instance of the iMDGP-grid problem having no feasible embedding. Therefore, the value of parameter α plays a key role in the feasibility check approaches discussed in this paper. We assume $\alpha = \frac{\epsilon}{\sqrt{3}}$, where ϵ is a tolerance satisfying $0 < \epsilon < \min_{\{u,v\} \in E} \{d^L_{uv}\}$. The last inequality prevents the lower bounds to collapse to zero.

3 Integer linear programming models

In this section, we present two Integer Programming (IP) models to describe an embedding of a graph $G = (V, E)$ in a grid for a given set $I(E)$ of interval distances.

The objective of the first model is to minimize an error distance function that is the sum of the interval distances violation. A feasible embedding corresponds to a feasible solution whose objective function value is zero. The second model is a feasibility model that checks whether a set of inequalities has feasible binary solutions. The first model always provides a solution (that may or may not correspond to a feasible embedding, depending on whether the objective value is zero or not), while the second model either provides one feasible solution corresponding to a feasible embedding or shows that there is no feasible embedding.

3.1 A discrete model to find an embedding that minimizes an error function

For each $\{u, v\} \in E$, the following model minimizes the distance (weight) error when assigning node u in the graph to position p in the grid and node v in the graph to position q in the grid. We use binary variables w_p^u indicating whether node $u \in V$ is assigned to position $p \in B$, i.e. $w_p^u = 1$, or not, i.e. $w_p^u = 0$. Let D_{pq} denote the Euclidean distance between positions p and q, $p, q \in B$.

$$\min \sum_{p,q \in B,\ \{u,v\} \in E,\ D_{pq} < d_{uv}^L} (d_{uv}^L - D_{pq}) w_p^u w_q^v \ + \sum_{p,q \in B,\ \{u,v\} \in E,\ D_{pq} > d_{uv}^U} (D_{pq} - d_{uv}^U) w_p^u w_q^v \quad (6)$$

subject to

$$\sum_{p \in B} w_p^u = 1, \qquad u \in V, \quad (7)$$

$$\sum_{u \in V} w_p^u \leq 1, \qquad p \in B, \quad (8)$$

$$w_p^u \in \{0, 1\}, \qquad p \in B,\ u \in V. \quad (9)$$

Constraints (7) establish that each node $u \in V$ must be assigned to exactly one point in the grid. Constraints (8) establish that at most one node of the graph is assigned to each point $p \in B$. Constraints (9) impose binary restrictions to the variables, while the objective function (6) minimizes the error associated with the embedding of G into B. Thus, an optimal solution has an optimal value equal to zero if and only if this solution defines a feasible embedding for the iMDGP-grid.

The quadratic objective function can be linearized by introducing two sets of new variables. For each edge $e = \{u, v\} \in E$ and each pair of grid points $p, q \in B$, we define a continuous variable ε_e, as the mapping error of assigning u, v to positions $p, q \in B$, as follows:

$$\varepsilon_e = \max\{0, d_{uv}^L - D_{pq}, D_{pq} - d_{uv}^U\}.$$

Also, for each pair of positions $p, q \in B$ and each edge $e = \{u, v\} \in E$, let y_{pq}^{uv} be the binary variable defined by:

$$y_{pq}^{uv} = \begin{cases} 1, & \text{if nodes } u \text{ and } v \text{ are assigned, respectively, to positions } p \text{ and } q \\ 0, & \text{otherwise.} \end{cases}$$

The integer linear formulation is as follows:

$$\min \quad \sum_{e \in E} \varepsilon_e \tag{10}$$

subject to (7), (8), (9),

$$y_{pq}^{uv} \geq (w_p^u + w_q^v - 1), \qquad p,q \in B,\ \{u,v\} \in E, \tag{11}$$

$$y_{pq}^{uv} \leq w_p^u, \qquad p,q \in B,\ \{u,v\} \in E, \tag{12}$$

$$y_{pq}^{uv} \leq w_q^v, \qquad p,q \in B,\ \{u,v\} \in E, \tag{13}$$

$$\varepsilon_{\{u,v\}} \geq (d_{uv}^L - D_{pq}) y_{pq}^{uv}, \qquad p,q \in B,\ \{u,v\} \in E:\ D_{pq} < d_{uv}^L, \tag{14}$$

$$\varepsilon_{\{u,v\}} \geq (D_{pq} - d_{uv}^U) y_{pq}^{uv}, \qquad p,q \in B,\ \{u,v\} \in E:\ D_{pq} > d_{uv}^U, \tag{15}$$

$$0 \leq y_{pq}^{uv} \leq 1, \qquad p,q \in B,\ \{u,v\} \in E, \tag{16}$$

$$\varepsilon_e \geq 0, \qquad e \in E. \tag{17}$$

Constraints (11) - (13) and (16) define the y_{pq}^{uv} variables and correspond to the classical linear reformulation of the quadratic term $w_p^u w_q^v$, that is, $y_{pq}^{uv} = 1$ iff $w_p^u w_q^v = 1$, which happens only if $w_p^u = w_q^v = 1$. Constraints (14), (15), and (17) model the ε_e variables. The objective function (10) is to minimize the violated distances. Let us refer to this formulation as MEA (Minimizing Error Assignment).

3.2 Discrete feasibility model

Deciding whether there exists a feasible embedding of G in B can be modeled as an assignment problem using the variables w_p^u defined above, while answering if there exists a feasible assignment is equivalent to solving the following feasibility model:

$$\sum_{p \in B} w_p^u = 1,\ u \in V, \tag{18}$$

$$\sum_{u \in V} w_p^u \leq 1,\ p \in B, \tag{19}$$

$$w_p^u + w_q^v \leq 1,\ p,q \in B,\ \{u,v\} \in E : D_{pq} \notin [d_{uv}^L, d_{uv}^U], \tag{20}$$

$$w_p^u \in \{0,1\},\ u \in V, p \in B. \tag{21}$$

Constrains (18), (19), (21) are the same as (7), (8), (9). Constraints (20) ensure that u and v cannot be simultaneously assigned, respectively, to p and q when the Euclidean distance D_{pq} is not in the interval distance $[d_{uv}^L, d_{uv}^U]$ related to edge $e = \{u,v\}$. These constraints are weak and will be strengthened in Section 5.

Let us refer to this formulation as FM (Feasibility Model). Each feasible solution of FM is equivalent to a valid embedding of G in B. This feasibility problem can be converted into the following optimization problem:

$$\max \quad \sum_{u \in V} \sum_{p \in B} w_p^u \tag{22}$$

$$\text{subject to} \sum_{p \in B} w_p^u \leq 1, u \in V, \tag{23}$$

$$(19), (20), (21),$$

where the objective is to maximize the number of nodes in V that can be mapped onto positions of B while satisfying the distance constraints. Constraints (18) are replaced by constraints (23) imposing that each node is assigned to at most one position in B. Hence, FM has a feasible solution iff the optimization problem has value equal to $|V|$.

The model given by (19), (20), (21), (22) and (23) can be seen as finding the maximum independent set on a conflict graph $G_C = (V_C, E_C)$ defined as follows: the set of nodes V_C is given by node-position pairs, $V_C = \{(u, p) : u \in V, p \in B\}$; there is an edge between two nodes $(u, p), (v, q)$ if the two pairs are incompatible, which occurs when i) $u = v$; or ii) $p = q$; or iii) $D_{pq} \notin [d_{uv}^L, d_{uv}^U]$. This model will be denoted by CG model.

Another approach to convert FM into an optimization model is to use the Farkas Lemma. Instead of using its variant for integer problems, we will apply the classical Farkas Lemma [21] to the linear relaxation of FM.

Associate dual variables $\alpha^u, u \in V$, with constraints (18), dual variables $\beta_p, p \in B$, with constraints (19), and dual variables $\gamma_{pq}^{uv}, \{u, v\} \in E, p, q \in B, p \neq q, D_{pq} \notin [d_{uv}^L, d_{uv}^U]$ with constraints (20).

The dual problem is given as follows:

$$h(\alpha, \beta, \gamma) = \min \sum_{u \in V} \alpha^u + \sum_{p \in B} \beta_p + \sum_{\{u,v\} \in E} \sum_{p \in B} \sum_{q \in B} \gamma_{pq}^{uv} \tag{24}$$

$$\text{s. t.} \quad \alpha^u + \beta_p + \sum_{v \in V} \sum_{q \in B} \gamma_{pq}^{uv} \geq 0, u \in V, p \in B, \tag{25}$$

$$\beta_p \geq 0, p \in B, \tag{26}$$

$$\gamma_{pq}^{uv} \geq 0, \{u, v\} \in E, p, q \in B, p \neq q, D_{pq} \notin [d_{uv}^L, d_{uv}^U]. \tag{27}$$

We assume in the model that each variable γ_{pq}^{uv} not defined is equal to zero.

The following theorem follows from the Farkas' Lemma.

Proposition 1. *Let X denote the set of solutions to the linear relaxation of the CG model. Then, either X is nonempty or $h(\alpha, \beta, \gamma) < 0$.*

As a result of Proposition 1, if $h(\alpha, \beta, \gamma) < 0$ (it suffices to find a dual solution for the dual problem with negative value) then the linear relaxation of the CG model is infeasible and, therefore, the integer problem is also infeasible. In that case no feasible embedding of G in B exists. The model (24)-(27) will be denoted by LFM (Linear Feasibility Model). We can easily see that the free variables α^u can be bounded to be non-positive.

Observe that when $h(\alpha, \beta, \gamma) \geq 0$ nothing can be concluded regarding the feasibility of the integer problem. Hence $h(\alpha, \beta, \gamma) < 0$ is just a sufficient condition for infeasibility. On the other hand, $h(\alpha, \beta, \gamma) < 0$ can be checked in polynomial time.

4 Constraint Programming

Constraint Programming (CP) is a technique that has proved to be competitive when compared against (mixed) integer programming (MIP) for certain combinatorial problems (such as scheduling problems) where the linear relaxations of the MIP models are weak; that is the case of the models discussed in Section 3. We refer a reader not familiar with the CP technique to [14].

We consider a very simple CP model. For each node $u \in V$, let $X(u)$ be a set variable with domain $B_u \subseteq B$ (an algorithm to compute such restricted domain B_u is described in the next section). We consider additionally the following constraints for the domain reduction:

$$d^L_{uv} \leq ||X(u) - X(v)|| \leq d^U_{uv}, \quad \forall (u,v) \in E. \tag{28}$$

We used the Xpress-Kalis [20] solver in the implementation of the CP model. Constraints (28) were encoded with the following *GenericBinaryConstraint* command to propagate the distance constraint:

$$\textit{GenericBinaryConstraint(}X(u)\textit{,}X(v)\textit{,“Valid Distance Positions”)},$$

where *“Valid Distance Positions”* is a binary function with inputs $p, q \in B$ that returns 1 if the distance between p and q lies in the interval $[d^L_{uv}, d^U_{uv}]$, and 0 otherwise.

5 Improvements

All the models discussed in the previous sections can be improved in different directions.

Domain reduction: When some atoms are fixed, which is the case of many instances tested, a preprocessing phase can be done in order to reduce the domain of each atom, that is: for each node $v \in V$, the set of grid points that can be assigned to v is reduced by checking its known distances to the fixed nodes. Let V^F be the set of fixed nodes. For each $u \in V^F$, let PosFix(u) denote the fixed grid position of node u in the grid. Also, for a grid point $p \in B$, (p_x, p_y, p_z) represents the corresponding grid coordinates.

Algorithm 1 is used to set the initial domains of the non-fixed nodes by computing the intersection of several spherical shells on the grid. Given a grid point $p \in B$ and two distances L and U, the function BallPoints defined in Algorithm 2 computes the spherical shell of grid points defined by $\{q \in B : L \leq \|p - q\| \leq U\}$.

Algorithm 1 allows us to define a restricted domain for each node $u \in V$, denoted by B_u. Set B can also be replaced by the restricted set $\overline{B} = \bigcup_{u \in V} B_u$. Using the restricted domains we set to zero all the variables y^{uv}_{pq} and w^u_p in the model MEA, and variables w^u_p in the models FM and CG, that do not belong to the corresponding domain. It also allows us to remove many constraints in the model MEA and to reduce the domain of the CP variables.

Strengthening model FM: The linearization proposed in models FM is very weak. Following the ideas from the Adam and Johnson linearization technique [1] for the quadratic assignment problem (which can not be applied directly here) the following set of equations can be added.

Algorithm 1 Domain reduction.

for all $u \in V^F, v \in V \setminus V^F$ **do**
  if $\{u, v\} \in E$ **then**
    if $D(v) = \emptyset$ **then**
      $D(v) \leftarrow \textsc{BallPoints}(PosFix(u), d^L_{uv}, d^U_{uv})$
    else
      $D(v) \leftarrow D(v) \cap \textsc{BallPoints}(PosFix(u), d^L_{uv}, d^U_{uv})$
    end if
  end if
end for

Algorithm 2 Ball Points: Given a grid point $p \in B$ and the distances L and U, computes all the grid points $q \in B$ such that $L \leq \|p - q\| \leq U$

function BallPoints(p, L,U)
  $X \leftarrow \emptyset$
  for $i = 0$ .. $\lfloor U/\alpha \rfloor$ **do**
    for $j = 0$ .. $\left\lfloor \sqrt{U^2/\alpha^2 - i^2} \right\rfloor$ **do**
      for $k = \left\lceil \sqrt{\max\{0, L^2/\alpha^2 - i^2 - j^2\}} \right\rceil \ldots \left\lfloor \sqrt{U^2/\alpha^2 - i^2 - j^2} \right\rfloor$ **do**

$$X \leftarrow X \cup \Big\{ (p_x + i\alpha, p_y + j\alpha, p_z + k\alpha), (p_x + i\alpha, p_y + j\alpha, p_z - k\alpha), (p_x + i\alpha, p_y - j\alpha, p_z + k\alpha), (p_x + i\alpha, p_y - j\alpha, p_z + k\alpha), (p_x - i\alpha, p_y + j\alpha, p_z + k\alpha), (p_x - i\alpha, p_y + j\alpha, p_z - k\alpha), (p_x - i\alpha, p_y - j\alpha, p_z + k\alpha), (p_x - i\alpha, p_y - j\alpha, p_z - k\alpha) \Big\}$$

      end for
    end for
  end for
  return X
end function

$$\sum_{q \in B} y_{pq}^{uv} = w_p^u, \{u,v\} \in E, p \in B \tag{29}$$

$$\sum_{p \in B} y_{pq}^{uv} = w_q^v, \{u,v\} \in E, q \in B \tag{30}$$

Equations (29) state that if w_p^u is one then y_{pq}^{uv} must be one for some $q \in B$. Similarly for equations (30). These equations are not necessary to define the model but they improve the lower bound.

Strengthening model CG: The model CG can be seen as finding a maximum cardinality independent set in a given graph. As a consequence, all the polyhedral theory known for the independent set problem [16] can be used here. In particular, since a non maximal clique inequality does not define a facet of the associated polytope, the defining inequalities of CG can be strengthened: each inequality (20) can be replaced by a stronger clique inequality that includes the nodes in a clique C of the conflict graph G_C:

$$\sum_{(u,p) \in C} w_p^u \leq 1. \tag{31}$$

Finding maximal cliques in G_C that allow us to derive inequalities of type (31) dominating inequalities (20) can be done efficiently using a greedy algorithm. Consider an edge $\{u,v\} \in E$ and a pair $p,q \in B$, such that $D_{pq} \notin [d_{uv}^L, d_{uv}^U]$. The first inequality states that if $D_{pq} > d_{uv}^U$, that is, if nodes u,v can not be placed in p,q, respectively (because the distance between p and q is greater than the maximum distance between u and v), then the same holds for every pair of points whose distance is greater than D_{pq}. Let π denote the vector $q-p$. The hyperplane defined by point p and normal to π is given by $(x-p)\cdot\pi = 0$. Similarly, the hyperplane defined by point q and normal to π is given by $(x-q)\cdot\pi = 0$. Each one of these hyperplanes defines two half-spaces. Let $H_u = \{x \in B \ : \ (x-p)\cdot\pi \leq 0\}$ and $H_v = \{x \in B \ : \ (x-q)\cdot\pi \geq 0\}$. It is easy to verify that when $D_{pq} > d_{uv}^U$, every point in H_u has distance at least d_{uv}^U from every point from H_v. Hence, if $D_{pq} > d_{uv}^U$, then the inequality

$$\sum_{\overline{p} \in H_u} w_{\overline{p}}^u + \sum_{\overline{q} \in H_v} w_{\overline{q}}^v \leq 1$$

can be added.

Define $S(\ell) = \{x \in B \ : \ \|x - \ell\| \leq d_{uv}^L\}, \ \ell \in B$. If $D_{pq} < d_{uv}^L$, then the following inequalities can be added:

$$\sum_{\overline{p} \in S(q)} w_{\overline{p}}^u + w_q^v \leq 1, \qquad q \in B, \ \{u,v\} \in E, D_{pq} \notin [d_{uv}^L, d_{uv}^U],$$

$$w_p^u + \sum_{\overline{q} \in S(p)} w_{\overline{q}}^v \leq 1, \qquad p \in B, \ \{u,v\} \in E, D_{pq} \notin [d_{uv}^L, d_{uv}^U].$$

When $D_{pq} < d_{uv}^L$, another possible lifting can be obtained by taking $x^* = \frac{1}{2}p + \frac{1}{2}q$. Then the inequality

$$\sum_{\overline{p} \in S(x^*)} w_{\overline{p}}^u + \sum_{\overline{q} \in S(x^*)} w_{\overline{q}}^v \leq 1$$

is valid for the set of feasible solutions.

Optimization strategy: The last improvement is related to the approach used to solve the models MEA and CG. Both are too large to be used as stated is Section 3, therefore a relaxation of each model obtained with the elimination of the larger set of constraints can be considered. The relaxation is solved and if the obtained solution violates one of the relaxed constraints, then those violated constraints are added to the model and the model is re-optimized. For Model MEA inequalities (11)-(15) are added dynamically as well as the variables that only appear in the constraints that are introduced. For model CG inequalities (20) are added dynamically. In both cases, the separation amounts to identifying pairs of nodes $u, v \in V$ whose distance in the current solution does not lie in the interval $I_e,\ e = \{u, v\}, e \in E$.

6 Computational experiments

In this section, we report some computational experiments to compare the IP models and the CP model, testing the effectiveness of these models in checking embedding feasibility and the improvements discussed in Section 5. We also show that the inclusion of a distance bound for those pairs of nodes for which the distance is not known, can be effective in solving the MDGP instances.

The computations were performed using the optimization software Xpress-Optimizer, Version 23.01.03 with Xpress Mosel Version 3.4.0 [20], on a computer with processor Intel Core 2, 2.2 GHz and with 2 GB RAM.

The domain reduction revealed to be crucial in reducing the size of the models considered. Most of the reported instances could not be solved without this reduction, therefore all the tests include the domain reduction as a pre-processing step.

First, we compare the IP models with and without dynamic inclusion of constraints and the CP model. Then, using the approach with the best performance (that is, the IP approach based on the CG model) we test the effectiveness of the strengthening of cuts discussed in Section 5. Finally, we report the improvements in solving the MDGP with the inclusion of implicit inequalities.

Model comparison

We compared the *Branch and Bound* using the IP models MEA and CG discussed in Section 3, with and without the improvements discussed in Section 5, against the CP model. The first set of instances is generated as follows. First we generate a molecule following the procedure described in [7]. Then we consider small sized instances with 5 or 6 nodes resulting from neighborhoods of atoms of the molecule. Two atoms are considered neighbors if their distance is less than $5.5\mathring{A}$. The position of the atoms is free. The size of the box is defined by taking $\Delta = 20\mathring{A}$. Three possible values for α, corresponding to different tolerance values (see Section 2) are considered: $\frac{2}{\sqrt{3}}$, $\frac{2.5}{\sqrt{3}}$, and $\frac{3}{\sqrt{3}}$.

Table 1 reports the results obtained. Column **#atoms** indicates the number of nodes. Column named α gives the value of α used in the discretization. Column **#NGrid** gives the number $|\overline{B}|$ of the points in the grid after the domain reduction using the algorithm described in Section 5. Columns **C Time** (fourth and sixth columns) indicate the running time in seconds using the complete model, that is, using the corresponding model with all the model constraints, and with no addition of valid inequalities. Columns **S Time** (fifth and seventh columns) give the

running time in seconds when all the improvements discussed in Section 5 are used, namely the *optimization strategy* is followed for both models (the larger sets of constraints are relaxed and these constraints are added dynamically); for model MEA we also include equations (29) and (30); and for model CG the lifted inequalities are included at the root node. Finally, column **Time** gives the running time in seconds using the CP model.

Table 1: Performance of the three models MEA, CG and CP.

			MEA		**CG**		**CP**
#atoms	α	**#NGrid**	**C Time**	**S Time**	**C Time**	**S Time**	**Time**
5	$\frac{3}{\sqrt{3}}$	82	0.05	0.26	0.03	**0.02**	0.09
5	$\frac{3}{\sqrt{3}}$	113	0.08	0.14	0.05	**0.03**	0.09
5	$\frac{2.5}{\sqrt{3}}$	205	0.31	0.29	0.05	**0.02**	0.08
5	$\frac{2.5}{\sqrt{3}}$	261	0.52	0.31	0.42	**0.05**	0.11
5	$\frac{2}{\sqrt{3}}$	524	2.12	0.53	4.03	0.09	**0.03**
5	$\frac{2}{\sqrt{3}}$	984	8.08	0.68	35.88	**0.08**	0.09
5	$\frac{2}{\sqrt{3}}$	1082	11.56	1.46	68.69	**0.09**	0.10
5	$\frac{2}{\sqrt{3}}$	1193	15.33	1.23	123.29	0.28	**0.11**
5	$\frac{2}{\sqrt{3}}$	1202	14.85	1.24	131.23	**0.11**	**0.11**
6	$\frac{3}{\sqrt{3}}$	89	0.30	1.03	0.08	**0.05**	0.09
6	$\frac{2.5}{\sqrt{3}}$	140	0.25	1.02	0.17	0.02	**0.01**
6	$\frac{2}{\sqrt{3}}$	186	1.15	0.36	0.30	**0.09**	**0.09**
6	$\frac{2.5}{\sqrt{3}}$	305	42.25	1.79	3.48	**0.03**	0.10
6	$\frac{2.5}{\sqrt{3}}$	353	4.21	3.14	5.20	**0.08**	0.09
6	$\frac{2.5}{\sqrt{3}}$	444	3.78	1.37	5.65	**0.03**	0.11
6	$\frac{2.5}{\sqrt{3}}$	504	3.42	2.42	5.90	**0.03**	0.11
6	$\frac{2}{\sqrt{3}}$	751	8.72	7.37	25.01	**0.12**	0.13

We can see that when the constraints are added dynamically, the *Branch and Bound* based on the CG model is the strategy that had the best performance. However the CP model was better in three instances.

Impact of using lifted inequalities

Considering only the best approach (the *Branch and Bound* method based on the CG model) we tested the effectiveness of the use of the strengthened inequalities. For these tests we considered a new set of larger instances based on a set of benchmark instances[1] extracted from Protein Data Bank[2].

As explained before the main purpose of this work is to provide feasibility tests that can be embedded in Branch and Prune algorithms where the binary tree of possible solutions is explored starting from its top, where the first three atoms are positioned, and by placing one node at a

[1] Available in `www.antoniomucherino.it/en/mdjeep.php/tests1.tar.gz`.

[2] `www.resb.org/pdb`

time following an ordering on the atoms of the protein. For each possible position of the selected node a new branch in the search tree is created. For each branch, a new instance is created based on the selected node and its neighbors. Thus, for each molecule, identified in Table 2 in column **Name** a set of instances of the iMDGP-grid is generated by considering neighborhoods of atoms. Each of these instances represents a set of atoms that were already fixed by the Branch and Prune ($B\&P$) algorithm [11] and a set of atoms that must be fixed in the next iterations of this algorithm. From the initial assumptions, the distances between three consecutive nodes are known. Hence, at least the selected node and the two previous ones (which are neighbors of the selected one) are considered in the instance and have fixed positions.

The main idea is to test if this approach enables us to check feasibility of an embedding of a set of atoms (which could be used within the $B\&P$ algorithm for the iMDGP). Although the original data comes from exact distances, we considered $\Delta = 20\mathring{A}$ and $\alpha = 0.1$; therefore the exact distances were converted into interval distances by using (4) and (5).

The first five columns are related to information about the instances. Column **#NInst** indicates the number of instances considered from the corresponding molecule identified in column **Name**, column **#atoms** gives the average number of atoms considered, column **#fixed** indicates the average number of fixed atoms, and column **#NGrid** gives the average number of the points in the grid after the elimination procedure described in Section 5. Columns **Time** give the average time, in seconds, and columns **#Uns** give the number of instances that couldn't be solved within the time limit of 1500 seconds of the corresponding model: CG is for the model CG with constraints added dynamically as described in *Optimization strategies*, Section 5, and CG+SI is for the same model with the addition at the root node of the lifted inequalities, described in the same section. We can see that the average running time is lower when the strengthened inequalities are added and the total number of unsolved instances drops from 23 to 9.

Table 2: Performance of the *Branch & Bound* algorithm based on the CG model with and without the strengthened inequalities.

Instance					**CG**		**CG+SI**	
Name	**#NInst**	**#atoms**	**#fixed**	**#NGrid**	**Time**	**#Uns**	**Time**	**#Uns**
1a70	18	9.2	4.4	28396	566	3	257	1
1bpm	43	9.0	4.2	23305	579	5	382	2
1fs3	46	9.4	4.5	22569	518	8	357	4
1jk2	29	9.4	4.6	26738	458	3	565	1
1m40	10	9.7	4.4	23570	614	3	438	1
1mbn	2	8.0	4.0	21086	55	0	76	0
1n4w	11	8.6	4.4	22271	259	1	180	0
All	159	9	4.4	23990.7	436	23	322	9

Among the 159 instances tested, 9 were proved to be infeasible. We ran the LFM for those infeasible instances and in none of them the LFM was able to prove infeasibility. Although this is a negative result, it also indicates that the linear relaxation of the CG model is weak, since for those instances the IP model is infeasible but its linear relaxation is feasible.

Use of implicit information from NMR

The main purpose of this work is to propose and compare feasibility checking models that can be used to improve the performance of a solution approach to the MDGP (as the $B\&P$ algorithm described in [9]). Additionally, in this section, we discuss a very simple improvement that uses implicit information from the NMR spectroscopy. The NMR spectroscopy analysis provides only few inter-atomic distances. The reason is that only short distances are detected (distances of less than $5.5\mathring{A}$). Hence, we use the fact that for each $(u, v) \notin E, u, v \in V$, the distance $d_{uv} \geq 5.5$. To the best of our knowledge, such information has not been used before in the $B\&P$ algorithm applied to protein structure calculation (see [8] for some related questions).

Based on a set of benchmark instances[3] with exact distances extracted from the Protein Data Bank[4], we compared the performance of the $B\&P$ algorithm described in [9] in both cases: with and without the implicit distance $d_{uv} \geq 5$ (a tolerance of $0.5\mathring{A}$ was considered). Table 3 reports the computational results. The first three columns characterize the instances. The columns *Name*, *#atoms* and #edges indicate the name of the instance, the number of nodes and the number of edges, respectively. Columns 4-8 report information of the $B\&P$ algorithm when it is run to find only one feasible embedding in $\mathbb{R}^3$, while the remaining columns refer to the case where the $B\&P$ is run until all feasible embeddings are found. Columns 4-5 and 9-11 give information for the pure $B\&P$, as described in [9], while columns 6-8 and 12-15 give information for the $B\&P$ where the implicit distances are tested at each tree node. Columns *#Nodes* indicate the number of the tree nodes, columns *Pruned* indicate the number of pruned nodes, and *%NR* indicate the percentage of the reduction of tree nodes by including the implicit distance constraints. Columns *#Sol* give the number of feasible embeddings (solutions) found.

We can see that the use of the implicit distance allows for a reduction on the number of nodes on average by 10% to obtain one solution and by 12% to obtain all feasible solutions. Considering the implicit distances, we can also eliminate feasible solutions found by the B&P algorithm in instances 1mbn, 1mqq, 3b34. Each solution that has been excluded corresponds to a feasible embedding that satisfies all the distance constraints for all pair $\{u, v\} \in E$ but does not satisfy the new implicit distances. However, it should be clear that the excluded solutions cannot correspond to the real conformation of the molecule.

7 Conclusions

We introduced different approaches for finding an embedding of a graph on a lattice that satisfies a set of distance intervals. We tested two integer programming models and one constraint programming model. We improved these models and showed that an integer programming model based on a conflict graph, strengthened with clique inequalities and based on an efficient generation of the domain of each variable had the best performance. This model allowed us to solve 150 out of the 159 tested instances within the specified time limit. These results show that this approach can be used to find a feasible embedding of small sized graphs in a lattice while satisfying a set of interval distances associated to the edges of the graph.

As a future line of research, we aim to use the best approach within a Branch and Prune

[3] Available in `www.antoniomucherino.it/en/mdjeep.php/tests1.tar.gz`.

[4] `www.resb.org/pdb`

Table 3: Performance of the *B&P* algorithm with and without the implicit distance constraints.

			1 Solution					All Solutions						
	Instance		BP		BP + dist			BP			BP + dist			
Name	#atoms	#edges	#Nodes	Pruned	#Nodes	%NR	Pruned	#Sol	#Nodes	Pruned	#Sol	#Nodes	%NR	Pruned
1a70	291	1628	534	182	534	0	182	2	1494	746	2	1494	0	746
1bpm	1443	9303	2334	705	2120	9	598	2	6674	3336	2	6126	8	3062
1crn	138	846	230	95	230	0	95	2	538	268	2	538	0	268
1fs3	372	2209	1382	612	734	47	288	2	4314	2156	2	2438	43	1218
1hoe	222	1259	333	114	333	0	114	2	874	436	2	874	0	436
1jk2	270	1816	620	262	488	21	196	8	4358	2172	8	3830	12	1908
1m40	1224	13823	2029	735	1945	4	693	2	5230	2614	2	5062	3	2530
1mbn	459	3200	833	354	596	28	131	8	3894	1936	2	1854	52	924
1mqq	2032	13016	2779	743	2779	0	743	8	11706	5846	4	10210	13	5102
1n4w	1610	10920	2291	668	2273	1	659	2	6518	3258	2	6482	1	3240
1pht	249	1448	661	275	485	27	187	2	2126	1062	2	1422	33	710
1poa	331	2201	767	331	767	0	331	2	1754	876	2	1754	0	876
1ppt	108	660	138	33	138	0	33	2	418	208	2	418	0	208
1ptq	150	829	373	156	355	5	147	2	1466	732	2	1394	5	696
1rgs	792	4936	4466	2096	4270	4	1998	8	21766	10876	8	17650	19	8818
1rwh	2265	14057	5676	2384	3780	33	1436	2	13190	6594	2	9386	29	4692
2e7z	2907	27706	6470	2610	6454	0	2602	2	16506	8252	2	16474	0	8236
2erl	120	763	213	91	213	0	91	2	486	242	2	486	0	242
3b34	2790	19563	4877	1909	4842	1	1906	4	12098	6046	2	11762	3	5880
Average	935	6852	1948	756	1755	10	654	3	6074	3035	3	5245	12	2621

($B\&P$) algorithm for the iMDGP, in order to identify infeasible assignments in early stages of the $B\&P$ search tree that, otherwise, could only be identified after a deep search. Since the running times of the branch and bound ($B\&B$) algorithm based on the CG model with strengthened inequalities are still high, this approach can only be useful when it identifies infeasibility quickly and allows to cut many search tree nodes. Currently, we are studying graph topologies where the characteristics of the subgraph corresponding to a neighborhood of atoms in the molecule are such that the $B\&B$ runs fast over instances that include nodes corresponding to atoms that appear in early stages of the search tree and others that appear only in latter stages of the tree. In this paper we have focused only in iMDGP instances. However, for general DGP, it would be interesting to investigate the behaviour of the three modelling approaches on large size instances. Also some of these approaches may be improved. For instance, it may be worthy of investigation other linearization techniques such as t-linearization, see [17] for the MEA model.

Acknowledgements

The authors would like to thank FAPESP, CAPES and CNPq for their financial support.

The research of authors $1, 2, 5$ and 6 was partially supported by Portuguese funds through the *Center for Research and Development in Mathematics and Applications (CIDMA)* and FCT, the Portuguese Foundation for Science and Technology, within project UID/MAT/04106/2013.

References

[1] W. P. Adams and T. A. Johnson, Improved linear programming-based lower bounds for the quadratic assignment problem, DIMACS Series in Discrete Mathematics and Theoretical Computer Science, American Mathematical Society, 16 (1994), 43–75.

[2] B. Berger, J. Kleinberg, and T. Leighton, Reconstructing a three-dimensional model with arbitrary errors, Journal of the ACM 46 (1999), 212–235.

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

[4] B. Donald, Algorithms in Structural Molecular Biology, MIT Press, Boston, 2011.

[5] Q. Dong and Z. Wu, A linear-time algorithm for solving the molecular distance geometry problem with exact inter-atomic distances, Journal of Global Optimization 22 (2002), 365–375.

[6] C. Lavor, R. Alves, W. Figueiredo, A. Petraglia, and N. Maculan, Clifford Algebra and the Discretizable Molecular Distance Geometry Problem, Advances in Applied Clifford Algebra, 25 (2015), 925–942.

[7] C. Lavor, On generating instances for the molecular distance geometry problem, Global Optimization: From Theory to Implementation, L. Liberti and N.Maculan, eds., Springer, Berlin (2006), 405–414.

[8] C. Lavor, A. Mucherino, L. Liberti, and N. Maculan, On the computation of protein backbones by using artificial backbones of hydrogens, Journal of Global Optimization, 50 (2011), 329–344.

[9] C. Lavor, L. Liberti, N. Maculan, and A. Mucherino, The discretizable molecular distance geometry problem, Computational Optimization and Applications, 52 (2012), 115–146.

[10] C. Lavor, L. Liberti, and A. Mucherino, The interval branch-and-prune algorithm for the discretizable molecular distance geometry problem with inexact distances, Journal of Global Optimization, 56 (2013), 855–871.

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

[12] L. Liberti, C. Lavor, A. Mucherino, and N. Maculan, Molecular distance geometry methods: from continuous to discrete, International Transactions in Operational Research, 18 (2010), 33–51.

[13] L. Liberti, C. Lavor, N. Maculan, and A. Mucherino, Euclidean distance geometry and applications, SIAM Review, 56 (2014), 3–69.

[14] K. Marriott, P. Stuckey, Programming with Constraints: An Introduction, The MIT Press, 1998.

[15] A. Mucherino, C. Lavor, L. Liberti, and N. Maculan (eds), Distance Geometry: Theory, Methods and Applications, Springer, New York, 2013.

[16] S. Rebennack, Stable set problem: branch & cut algorithms, Encyclopedia of optimization, Springer, (2008), 3676–3688.

[17] C. D. Rodrigues, D. Quadri, P. Michelon, and S. Gueye, 0-1 Quadratic Knapsack Problems: An Exact Approach Based on a t-Linearization, SIAM Journal on Optimization, 22 (2012), 1449–1468.

[18] J. Saxe, Embeddability of weighted graphs in k-space is strongly NP-hard, in Proc. of 17th Allerton Conference in Communications, Control, and Computing (1979), 480–489.

[19] H. Yoon, Optimization Approaches to Protein Folding, PhD thesis, Georgia Institute of Technology, 2006.

[20] FICO Xpress Optimization Suite, (2009).

[21] L. A. Wolsey, Integer Programming, Wiley-Interscience, 1998.

[22] K. Wütrich, Protein structure determination in solution by nuclear magnetic resonance spectroscopy, Science 243 (1989), 45-50.