An integer programming approach for the search of discretization orders in distance geometry problems

Jérémy Omer, Douglas S Gonçalves

An integer programming approach for the search of discretization orders in distance geometry problems

Jérémy Omer[1] and Douglas S. Gonçalves[2]

[1]INSA Rennes & Institut de Recherche Mathématiques de Rennes, France
[2]Universidade Federal de Santa Catarina

May 2, 2017

Abstract

Discretizable distance geometry problems (DDGPs) constitute a class of graph realization problems where the vertices can be ordered in such a way that the search space of possible positions becomes discrete, usually represented by a binary tree. Finding such vertex order is an essential step to identify and solve DDGPs. Here we look for discretization orders that minimize an indicator of the size of the search tree. This paper sets the ground for exact solution of the discretization order problem by proposing two integer programming formulations and a constraint generation procedure for an extended formulation. We discuss some theoretical aspects of the problem and numerical experiments on protein-like instances of DDGP are also reported.

Keywords Distance geometry, Integer programming, Discretization order, Cutting plane

I. Introduction

I.1. The distance geometry problem

Let K be a positive integer and $G = (V, E, d)$ be a simple weighted undirected graph, where edges are weighted by $d : E \to \mathbb{R}_+$, and $\{i, j\} \in E$ if the pairwise distance between $i, j \in V$ is available. The Distance Geometry Problem (DGP) asks whether there exists a realization $\mathbf{x} : V \to \mathbb{R}^K$ such that the constraints

$$\|\mathbf{x}(i) - \mathbf{x}(j)\| = d(\{i, j\}), \quad \forall \{i, j\} \in E, \tag{1}$$

are satisfied.

If the graph is complete, a realization (or the non-existence) may be found in polynomial time by matrix decomposition methods [8]. However, when only some pairwise distances are available, the problem becomes NP-Hard [12].

I.2. Discretization for branch-and-prune

For a class of DGPs where the vertices can be ordered in such a way that the initial K vertices induce a clique and the following vertices have at least K adjacent predecessors, it is possible to employ sequential geometric build-up methods [6].

Formally, we say that a DGP instance is a Discretizable DGP (DDGP) if there exists a bijection $\sigma : V \to \{1, \ldots, |V|\}$ such that:

(a) $G[\{\sigma^{-1}(1), \ldots, \sigma^{-1}(K)\}]$ is a clique;

(b) $\forall i \in V$ such that $\sigma(i) \in \{K+1, \ldots, |V|\}$: $|U(i)| \geq K$ holds, where

$$U(i) = \{j \in V \,:\, \{j, i\} \in E,\, \sigma(j) < \sigma(i)\}.$$

The vertex order defined by σ is called *discretization order*. The first K vertices may be positioned by matrix decomposition, for example, due to assumption (a). Then, for each $\sigma(i) > K$, assumption (b) ensures that i has at least K adjacent predecessors. Using the distances from i to K reference vertices in $U(i)$, at most two candidate positions for i are found by intersecting K spheres (assuming the references are affinely independent) [7]. This procedure leads to a binary search tree which can be explored by a Branch-and-Prune (BP) algorithm [9]. In case the spheres intersection is non-empty and $|U(i)| > K$, the additional information may be used to prune infeasible candidates [7].

Extensive numerical experiments show that the BP algorithm is efficient in solving practical problems that can be modelled as a DDGP [7]. Perhaps the most successful application is related to protein conformation.

I.3. Contributions and outline

From the previous section it is clear that an essential preprocessing step for solving a DDGP is to find a discretization order. We propose two formalizations of the discretization order problem (DOP) based on simple estimations of the quality of a discretization.

The focus of the article is then on the exact solution of the problem with integer programming (IP). In this first study, we only solve the simpler version of the DOP. We develop two IP formulations and describe a cutting plane algorithm that achieves a significant speed-up of the solution time. The different approaches are tested numerically on a set of instances based on real protein conformations obtained from the Protein Data Bank (PDB)[1].

Previous mathematical programming formulations were proposed in [6] where the problem is called Discretization Vertex Ordering Problem (DVOP), and in [2] where the adjacent predecessors should be contiguous, given raise to the Contiguous Trilateration Ordering problem (CTOP). Both problems differ from ours because they are only feasibility problems.

This paper is organized as follows. Section II defines indicators of the size of the BP tree and Section III describes the integer programming formulations for DOP and a constraint generation approach. Numerical experiments and a comparison between the models is presented in Section IV. Section V contains some final remarks and directions for future research.

II. Definition of the discretization order search

We are searching for a bijection $\sigma : V \to \{1, \ldots, |V|\}$ describing a valid discretization order of the DGP, i.e. satisfying the assumptions (a) and (b). The image of $i \in V$ through σ is the *rank* of i.

Using σ, the BP algorithm can then be executed to solve the DGP, by successively computing a finite number of potential realizations for the vertices with rank 1 to $|V|$. Recall that in order to avoid congruent solutions by rotations and translations, the realizations of the K first vertices are fixed. At each following iteration of the branch-and-prune:

- branching on the nodes of a vertex with K adjacent predecessors induces at most two children nodes;

- branching on the nodes of a vertex with more than K adjacent predecessors induces at most one child node.

The difficulty is that the potential realizations of the vertices are not computed during the search for a discretization order. Therefore, the exact number of nodes in the BP tree cannot be known before executing the BP.

To circumvent this issue, we define two indicators of the size of the BP tree for a given discretization order σ. For this, we define a *double vertex* as a vertex that has exactly K adjacent predecessors for σ. In contrast, a vertex with more than K adjacent predecessors is a *single vertex*. Since the double vertices are responsible for the growth of the BP tree, the first approach is to minimize their number, $N(\sigma)$. A more refined approximation is obtained by defining the following sequence:

$$\begin{aligned}&\text{For } k = 1, \dots, K : n_k(\sigma) = 1,\\ &\text{For } K+1 \leq k \leq |V| : n_{k+1}(\sigma) = \begin{cases} 2 \times n_k(\sigma) & \text{if } \sigma^{-1}(k) \text{ is double} \\ n_k(\sigma) & \text{otherwise.} \end{cases}\end{aligned}$$

Notice that $n_k(\sigma)$ corresponds to the number of nodes at level k of the BP tree. The maximum number of nodes in the BP tree is then given by $M(\sigma) = \sum_{k=1}^{|V|} n_k(\sigma)$.

The search for a valid discretization order minimizing $N(\sigma)$ is denoted MIN DOUBLE and the minimization of $M(\sigma)$ is denoted MIN NODES (as a reference to the nodes of the BP tree). As a first study of IP formulations for the DOP, we restrict our focus to MIN DOUBLE. This problem is likely to give us more insight into the polyhedral structure of the DGP discretization order constraints, because it does not include the nonlinearities that appear in the computation of $M(\sigma)$.

III. Integer programming formulations of MIN DOUBLE

The problem is essentially a vertex ordering problem so it has strong connections with the linear ordering problem (LOP) which is defined as follows. Given a finite set S and a cost function $c : S^2 \to \mathbb{R}$, find a bijection $\sigma : S \to \{1, \dots, |S|\}$ that minimizes $\sum_{\sigma(u)<\sigma(v)} c(u, v)$. Similarly to the LOP, MIN DOUBLE requires information about precedence between pairs of vertices. Moreover, there is a need to include constraints to guarantee that this relation of precedence corresponds to a vertex ordering. The main (and very significant) difference is implied by the cardinality constraints on the adjacent predecessors of each vertex. First, we describe two different sets of constraints specifying the total order relation. Then, we deal with the number of predecessors of each vertex. Finally, we describe a constraint generation procedure for an extended formulation.

III.1. Vertex ordering constraints

The binary variables that indicate precedence are defined as $x_{ij} = 1$ if and only if $\sigma(i) < \sigma(j)$, $\forall (i, j) \in V^2, i \neq j$. The first model we will consider shares some constraints with the classical formulation of the LOP [4, 5]. These constraints state that a total order relation is defined by totality and transitivity relations:

$$x_{ij} + x_{ji} = 1, \ \forall (i, j) \in V^2, i \neq j, \tag{2a}$$

$$x_{ij} + x_{jk} + x_{ki} \leq 2, \ \forall (i, j, k) \in V^3, i \neq j, i \neq k, j \neq k. \tag{2b}$$

From a graph theory point of view, a vertex ordering is a spanning acyclic tournament, and the above two constraints can be seen as 2-cycle and 3-cycle constraints.

This formulation requires $\mathcal{O}(|V|^3)$ constraints. Contrary to the LOP, MIN DOUBLE only needs the relation of precedence between vertices that are connected by an edge to count the number of adjacent predecessors. In other words, we need to determine an acyclic orientation of G, instead of a spanning acyclic tournament. Denoting $\mathcal{C}$ the set of elementary cycles of G, this is ensured by

$$\sum_{\{i,j\}\in E(C)} x_{ij} \leq |C| - 1, \; \forall C \in \mathcal{C}. \tag{3}$$

This extended formulation can then be used to reduce the number of constraints when the graph is sparse.

Proposition 1 *The value of x describes an acyclic orientation of G if:*

$$\forall (i,j) \in V^2 : x_{ij} + x_{ji} = 1 \tag{4a}$$

$$\forall \{i,j\} \in E, k \in V, k \neq i,j : x_{ij} + x_{jk} + x_{ki} \leq 2 \tag{4b}$$

Proof 1 *Let $(i_1, \ldots, i_p)$ be a p-cycle of G with $p \geq 4$. Using the 2-cycle constraints* (4a)*, the cycle constraint $x_{i_1,i_2} + \cdots + x_{i_p,i_1} \leq p-1$ is implied by the 3-cycle $x_{i_1,i_2} + x_{i_2,i_3} + x_{i_3,i_1} \leq 2$ and the $p-1$-cycle $x_{i_1,i_3} + x_{i_3,i_4} + \cdots + x_{p-1,p} + x_{p,1} \leq p-2$. By induction, we then show that any dicycle constraint of G is implied by a set of 3-cycle constraints that contain at least one edge of G.*

Considering constraints (4a)-(4b), we get a sufficient formulation for the vertex ordering with $|V|^2$ binary variables and $\mathcal{O}(|V|\,|E|)$ constraints. Moreover, the discretization conditions imply that there is at least $K|V| - K(K+1)/2$ edges, and we observed during preliminary tests that the instances are often solved trivially when the number of edges goes over $(K+2)|V|$. As a consequence, this formulation should allow for a quadratic number of constraints with respect to $|V|$ for the difficult instances.

A different approach relies on the observation that the travelling salesman problem (TSP) can also be seen as a vertex ordering problem. This allows to benefit from the extensive literature on IP formulations of the TSP (see e.g. [3]). In particular, we can adapt the Miller-Tucker-Zemlin [10] formulation to replace the cycle constraints (4) by

$$\forall \{i,j\} \in E : |V| \times x_{ij} + \sigma_i - \sigma_j \leq |V| - 1, \tag{5}$$

where $\sigma_i, \sigma_j \in \{1, \ldots, |V|\}$ refer to the ranks of vertices i and j in the order. This formulation involves $|E|$ constraints instead of the $\mathcal{O}(|V|\,|E|)$ cycle constraints (4), and it requires only the variables x_{ij} for $\{i,j\} \in E$ instead of V^2. The price to pay is the addition of $|V|$ integer variables, and a potential loss in the quality of the linear relaxation, as suggested by the following result.

Proposition 2 *The feasible set of* (4) *is included in the projection of the feasible set of* (5) *on $\{x_{ij}\}_{\{i,j\}\in E}$.*

Proof 2 *Let x be a solution of* (4) *and $\sigma_i = \sum_{j\neq i} x_{ij}, \forall i \in V$. For a given edge $\{i,j\} \in E$, we obtain the corresponding constraint* (5) *by summing the 3-cycle constraints $x_{ij} + x_{jk} + x_{ki} \leq 2$ for $k \in V$. Hence x satisfies* (5)*.*

III.2. Discretization constraints

To solve MIN DOUBLE, we need to count the double vertices and set the vertices belonging to the initial clique. For this, we define the binary variables $\delta_i = 1$ if and only if vertex $i \in V$ is double, and $\rho_i^k = 1$ if and only if $\sigma(i) = k$, $\forall i \in V, k = 1 \ldots, K$.

The constraints then state that the first K ranks must be occupied by exactly one vertex, and that no more than one rank is assigned to each vertex:

$$\forall k \in \{1,\dots,K\} : \sum_{i \in V} \rho_i^k = 1 \qquad \text{and} \qquad \forall i \in V : \sum_{k=1}^{K} \rho_i^k \leq 1 \tag{6}$$

Finally, we need to state that the vertices outside the initial clique have at least K predecessors, and that they are double only if they have exactly K. For the k^{th} vertex of the initial clique, we must ensure that it has $k-1$ predecessors. Hence

$$\forall i \in V : \sum_{\{i,j\} \in E} x_{ij} + \sum_{k=1}^{K} (K-k+1) \times \rho_i^k \geq K + (1-\delta_i) \tag{7}$$

We thus add $(K+1)\,|V|$ variables and $2\,|V| + K$ constraints to express the discretization conditions.

One alternative approach has also been considered, because it appears that this formulation brings a lot of symmetry to the model. Indeed, for a given initial clique, any permutation of the vertices belonging to the clique provides a valid order. To counter this, we enumerate all the possible initial K-cliques, denoted $\mathcal{K}$. For this formulation, we introduce the binary variables $\kappa_c = 1$ if and only if c is the initial K-clique, $\forall c \in \mathcal{K}$. And the constraints (6)-(7) become

$$\sum_{c \in \mathcal{K}} \kappa_c = 1 \qquad \text{and} \qquad \forall i \in V : \sum_{\{i,j\} \in E} x_{ij} + \sum_{c \in \mathcal{K} : i \in c} (K - R_i^c + 1)\kappa_c \geq K + (1-\delta_i), \tag{8}$$

where R_i^c is the rank (from 1 to K) of i in c. In our tests, the observed size of $\mathcal{K}$ is linear in $|V|$. Moreover, $\mathcal{K}$ has been further reduced by keeping only the K-cliques that can be extended into a $(K+1)$-clique, which is a necessary condition for the existence of a discretization order.

To summarize, we consider two different formulations of MIN DOUBLE that include cycle constraints and rank variables, respectively. The two models are denoted ($\mathbb{CYCLES}$) and ($\mathbb{RANKS}$), and are given by:

$$(\mathbb{CYCLES}) : \begin{cases} \min\limits_{x,\delta,\kappa} & \sum\limits_{i \in V} \delta_i \\ \text{subject to} & (4), (8) \end{cases} \qquad (\mathbb{RANKS}) : \begin{cases} \min\limits_{x,\sigma,\delta,\kappa} & \sum\limits_{i \in V} \delta_i \\ \text{subject to} & (5), (8) \end{cases}$$

Assuming that there are $\mathcal{O}(|V|)$ potential initial cliques, ($\mathbb{CYCLES}$) includes $\mathcal{O}(|V|^2)$ binary variables and $\mathcal{O}(|V|\,|E|)$ constraints, whereas ($\mathbb{RANKS}$) includes $\mathcal{O}(|E|)$ binary variables, $\mathcal{O}(V)$ integer variables and $\mathcal{O}(|E|)$ constraints. These two models correspond to the formulations that provide the best results, but we will discuss the computational performance achieved with the discretization constraints (6)-(7) in Section IV.

III.3. A cycle constraint generation algorithm

Model ($\mathbb{CYCLES}$) includes a reduced number of 3-cycles, but the resulting constraints still involve x_{ij} variables for the pairs of vertices (i,j) that do not correspond to edges of G. In contrast, the extended formulation (3) focuses on the edges that belong to cycles of G. In preliminary numerical tests, we thus have implemented classical cutting planes methods that regularly add cuts after solving the linear relaxation. In these tests, we generated cycles from (3), k-fences and Möbius ladder inequalities using the methods described by Grötschel et al. [4] for the LOP. Unfortunately, the results have been disappointing for the DOP: we never observed any improvement in the optimal value of the linear relaxation after adding cuts at the root node, and the separation was

very time-consuming. On the other hand, further investigations showed that for many instances, the optimal solution does not change when a large part of the 3-cycles constraints (4b) are dropped. As a consequence, we developed an algorithm that generates constraints only when an integer solution that does not satisfy (4b) is found.

Let $p \geq 2$ be an arbitrary integer parameter. The essence of the algorithm is to start the solution algorithm with only the cycles of (3) whose length is at most p, and add new cycle constraints when a new incumbent containing a cycle is found during the solution. Algorithm 1 formalizes the procedure assuming that the IP is solved with a standard branch-and-cut algorithm as implemented e.g. in CPLEX. In the remainder of the article, this method will be denoted (CCG) for *cycle constraint generation*.

input: $p \in \mathbb{N}$
$\overline{\mathcal{C}} := \{C \in \mathcal{C} : |C| \leq p\}$
$(\mathbb{MIP}) := \min\{\sum_{i \in V} \delta_i$: subject to (8), $\forall C \in \overline{\mathcal{C}} : \sum_{\{i,j\} \in C} x_{ij} \leq |C| - 1\}$
optimal := false
while *True* **do**
  Make one iteration of branch-and-cut; **if** *Optimality is proved* **then**
    Stop and return the best incumbent
  if *A new incumbent is found* **then**
    $(\overline{x}, \overline{\delta})$:= the new incumbent
    if *No cycle constraint is violated by* $\overline{x}$ **then**
      Accept the new incumbent
    else
      Separate a cycle constraint C violated by $\overline{x}$
      Add the cycle cut C to $(\mathbb{MIP})$

Algorithm 1: Cycle cut generation procedure

To identify cycles in the digraph induced by $\overline{x}$, we search for a topological order using the depth-first search (DFS), as first described by Tarjan [13]. The DFS actually returns a topological order if and only if the digraph is acyclic. Otherwise, denoting $\overline{\sigma}$ the result of the DFS, there exists $\{i, j\} \in E$ such that $\overline{x}_{ij} = 1$ but $\overline{\sigma}(i) > \overline{\sigma}(j)$. A shortest path from j to i then provides a cycle containing (i, j) with a minimum number of arcs.

IV. Experimental comparison

IV.1. Greedy search for an upper bound

Since every formulation described above is to be solved with a standard branch-and-cut algorithm, it can be worth identifying a good upper bound before starting the solution. For this, we execute Algorithm 2 that has been first described by Lavor et al. [6] for one given initial clique. It is a greedy algorithm whose execution is in $\mathcal{O}(|E|\,|V| \log(|V|))$ when only one K-clique is treated. In essence, starting from a given initial clique, the algorithm iteratively assigns ranks $K + 1$ to $|V|$ to the vertex with the largest number of already ordered adjacent vertices. Lavor et al. [6] have shown that this algorithm finds a valid discretization order (for a given initial clique) if and only if one exists.

for $C \in \mathcal{K}$ `// treat every potential initial clique`
do
  Set the rank of the vertices of the K-clique to $1, \ldots, K$
  $O := C$, $k := K$
  while *a vertex has not been ordered* **do**
    Let i be a vertex of $V \setminus O$ with maximum number of adjacent vertices in O
    If i has less than K adjacent vertices in O, treat the next clique
    Assign rank $k+1$ to i
    $k := k+1$
  Update the best discretization order found so far.
If $|O| < |V|$ for all initial cliques, then no discretization order exists. Otherwise, return the discretization order with minimum $N(\sigma)$

Algorithm 2: Greedy algorithm

IV.2. Numerical results

The two compact formulations (CYCLES) and (RANKS), and the extended formulation (CCG) have been compared on a benchmark built from existing proteins whose three-dimensional conformations are available in the Protein Data Bank. The specific instances we used are distributed by Mucherino [11] together with a software for the solution of DDGPs using BP. Among the available instances, we picked the 19 instances available under the `test1` folder. These instances are also used in the numerical tests of Lavor et al. [7].

Besides, we modified the instances to study the impact of the numbers of vertices and edges of the distance graph. In his instances, Mucherino [11] numbered the vertices according to a valid discretization order, so the number of vertices could be modified by discarding the last vertices with the guarantee that a discretization order exists. For our tests, we kept 20 to 100 vertices in each instance. To introduce variations in the graph density we iteratively discarded randomly picked edges. In this deletion procedure, we had to run the greedy algorithm (Algorithm 2) every time an edge is picked to verify that a discretization order still exists after its removal. In the tests presented below, the density is expressed as the average vertex degree.

Every model is solved with CPLEX 12.6.3 on a single thread of an Intel(R) Core(TM)i7-3770 CPU @ 3.40GHz processor, and the time limit was set to 1000 seconds. The constraint generation procedure of (CCG) was implemented by declaring the n-cycle constraints, $n > p$, as CPLEX lazy constraints. The default maximum length of the cycles initially considered in (CCG) is set to 3 (i.e. $p = 3$).

We first study the performance of the three formulations with respect to the number of vertices. The results are presented for a distance graph including on average 3.5 edges per vertex. This value is chosen, because it is half way between K and $K+1$ edges per vertex. Moreover, the degree of each vertex being at least K is a necessary condition for the existence of a discretization order. The number of instances solved to optimality (over 19) and the relative gap at time limit are represented on Figure 1. We first observe that the performance of (CYCLES) and (RANKS) are very similar, but significantly worse than that of (CCG). Indeed, the constraint generation procedures allows to solve every instance for $|V| = 40$ and still solve four instances for $|V| = 90$, whereas the 19 instances are only solved for $|V| = 20$, and none is solved from $|V| = 70$ with the other two formulations. The reason is that the linear relaxation takes 10 to 100 times longer to solve for (CYCLES), and the number of CPLEX branch-and-bound nodes is 10 to 30 times larger when solving (RANKS). These observations were expected, considering the number of constraints and variables of the two compact formulations.

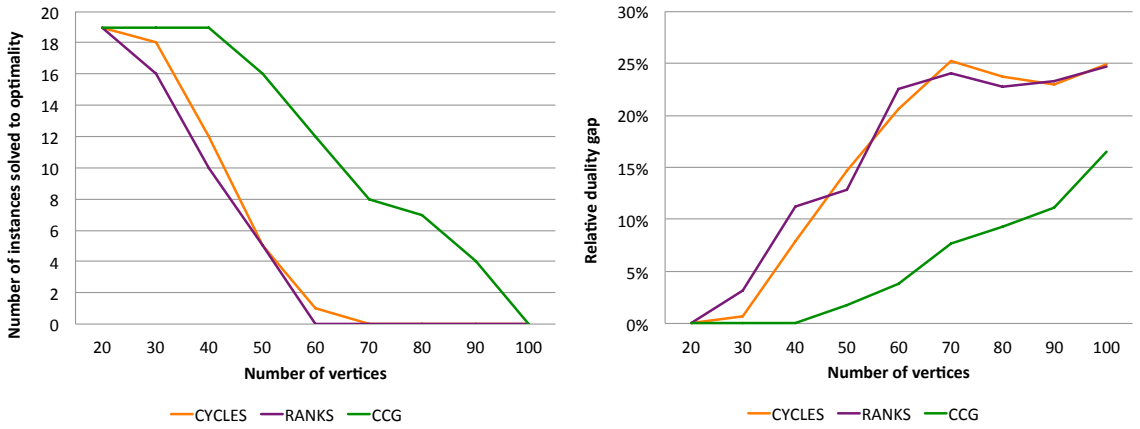

Figure 1: *Performance of the formulations with respect to the number of vertices*

We then draw our attention on the impact of the graph density. For this, we restrict the tests to (CCG), which has shown to be the best approach. Once, again we consider the number of instances solved to optimality and the relative duality gap on Figure 2. The average degree of the vertices varies from 3.1 to 5 and the number of vertices is set to 60. The numerical results clearly indicate that the solution is most difficult for intermediary degrees ranging from 3.8 to 4.4. For these instances, the linear relaxation is very weak; in particular, for densities larger than 4.4, there always exists a fractional solution with a zero objective value.

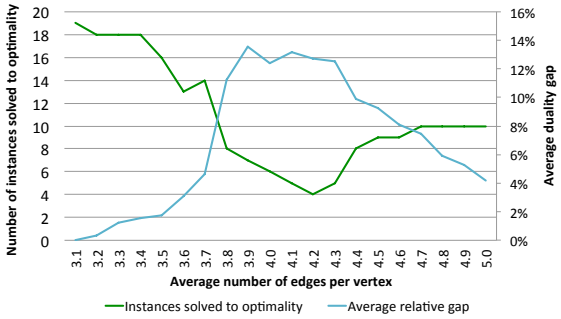

Figure 2: *Impact of the graph density on the performance of* (CCG)

Figure 3 presents the impact on (CCG) of the parameter p and of the enumeration of the potential initial cliques in the discretization constraints. The results of (CYCLES), in brown, have also been included as a reference to evaluate the importance of each improvement. As shown by the red curve, the enumeration of potential initial cliques is as important as the generation of cycle cuts to obtain the best results. Also, compared to the green ($p = 3$) and blue ($p = 4$) curves, the purple curves ($p = 2$) show that it is necessary to include at least the 3-cycles of G in the starting formulation of (CCG).

Finally, it has been observed that the greedy algorithm provides an excellent upper bound for every instance involved in our tests. To be specific, the average objective value of the greedy algorithm is inside 1% of the average best objective value found by solving (CCG).

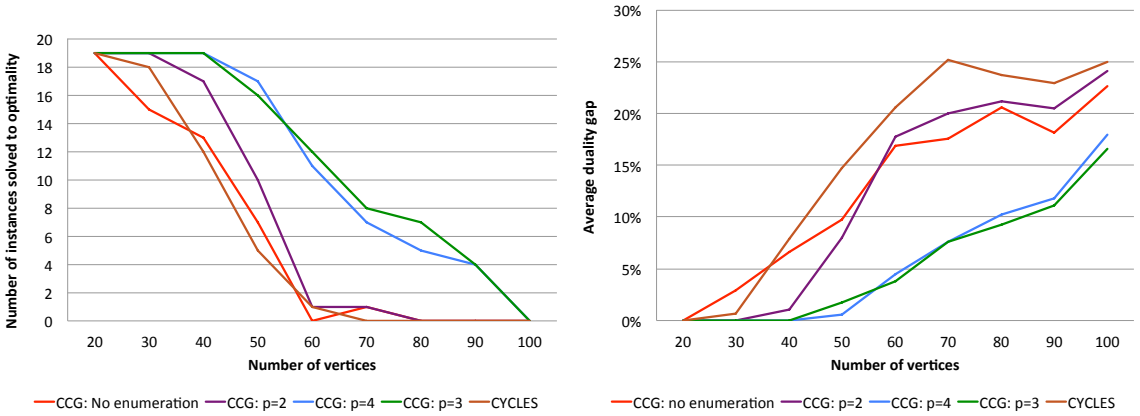

Figure 3: *Evaluation of the parameter p and of the enumeration of initial cliques*

V. Discussion

In this paper we proposed several compact and extended IP formulations for finding discretization orders with a minimum number of double vertices (responsible for the growth in the BP search tree). The formulations were tested on a benchmark of 19 protein conformation instances.

The results allow to highlight that the cutting plane generation algorithm provides better results than the compact IP formulations, but the instances that could be solved to optimality include only a small number of vertices compared to the actual number of atoms in the backbones of proteins.

Despite these limits, this study has several important contributions. First, it sets the ground for exact solution of the DOP, which could reveal essential for other applications. For instance, for the DGP with interval distances, one bad choice in the discretization order can lead to 10 to 100 more BP nodes. Second, it is very interesting to observe that the discretization constraints bring so much computational difficulties to the well-known LOP, whose linear relaxation has been observed to be very good in practice [4]. This could lead to important developments on the polyhedral structure of the problem. Finally, we could establish that the greedy algorithm presented by Lavor et al. [6] is able to provide sufficiently discretization orders that achieve a minimum number of double vertices for most protein conformation instances. This justifies the approach of the Lavor et al. [6] and Mucherino [11] (among others) who use this greedy algorithm as a preprocessing of BP methods for the solution of DDGPs.

Future works on the subject could treat the more relevant and complex discretization problem MIN NODES, which also considers the rank of the single vertices in the objective function. Moreover, the solution algorithms may be accelerated by searching symmetry in the instances and applying decomposition techniques.

References

[1] H. Berman, J. Westbrook, Z. Feng, G. Gilliland, T. Bhat, H. Weissig, I. Shindyalov, and P. Bourne. “The protein data bank”. In: *Nucleic Acids Research* 28 (2000), pp. 235–242.

[2] A. Cassioli, O. Gunluk, C. Lavor, and L. Liberti. “Discretization vertex orders in distance geometry”. In: *Discrete Applied Mathematics* 197 (2015), pp. 27–41.

[3] L. Gouveia and P. Pesneau. "On extended formulations for the precedence constrained asymmetric traveling salesman problem". In: *Networks* 48.2 (Sept. 2006), pp. 77–89.

[4] Martin Grötschel, Michael Jünger, and Gerhard Reinelt. "A Cutting Plane Algorithm for the Linear Ordering Problem". In: *Operations Research* 32.6 (Dec. 1984), pp. 1195–1220.

[5] Martin Grötschel, Michael Jünger, and Gerhard Reinelt. "Facets of the linear ordering polytope". In: *Mathematical Programming* 33.1 (Sept. 1985), pp. 43–60.

[6] Carlile Lavor, Jon Lee, Audrey Lee-St. John, Leo Liberti, Antonio Mucherino, and Maxim Sviridenko. "Discretization orders for distance geometry problems". In: *Optimization Letters* 6.4 (Apr. 2012), pp. 783–796.

[7] Carlile Lavor, Leo Liberti, Nelson Maculan, and Antonio Mucherino. "The discretizable molecular distance geometry problem". In: *Computational Optimization and Applications* 52.1 (2012), pp. 115–146.

[8] L. Liberti, C. Lavor, N. Maculan, and A. Mucherino. "Euclidean distance geometry and applications". In: *SIAM Review* 56 (2014), pp. 3–69.

[9] Leo Liberti, Carlile Lavor, and Nelson Maculan. "A Branch-and-Prune algorithm for the Molecular Distance Geometry Problem". In: *International Transactions in Operational Research* 15.1 (Jan. 2008), pp. 1–17.

[10] C. Miller, A. Tucker, and R. Zemlin. "Integer programming formulations and the travelling salesman problems". In: *Journal of the ACM* 7 (1960), pp. 326–329.

[11] Antonio Mucherino. *MD-jeep: a software for Distance Geometry*.

[12] J. B. Saxe. "Embeddability of Weighted Graphs in *k*-space is Strongly NP-hard". In: *Proceedings of* 17^{th} *Allerton Conference in Communications, Control and Computing*. Monticello, IL, 1979, pp. 480–489.

[13] Robert Endre Tarjan. "Edge-disjoint spanning trees and depth-first search". In: *Acta Informatica* 6.2 (1976), pp. 171–185.