A new algorithm for a class of Distance Geometry problems

Michael Souza[1], Douglas S. Gonçalves[2], Luiz M. Carvalho[3], Carlile Lavor[4], Leo Liberti[5]

[1] Federal University of Ceara, Brazil
[2] Federal University of Santa Catarina, Brazil
[3] Rio de Janeiro State University, Brazil
[4] University of Campinas, Brazil
[5] LIX CNRS, École Polytechnique, Institut Polytechnique de Paris, France

Abstract

We propose a new algorithm for Discretizable Molecular Distance Geometry problems (DMDGPs), a class of Distance Geometry problems (DGPs) whose search space can be discretized and represented by a binary tree. By efficiently exploiting the many interesting symmetry properties of DMDGP instances, the new algorithm solves a sequence of nested and overlapped DMDGP subproblems rather than exploring the binary tree in a depth first manner as the classic Branch-and-Prune (BP) algorithm. Computational results on artificially generated instances show that the new algorithm outperforms the classic BP algorithm in sparse DMDGPs.

Keywords : *Distance Geometry, Discretization, Symmetry, Partial reflections*

1 Introduction

The fundamental inverse problem in distance geometry is the one of finding positions from interpoint distances. More formally, given a simple undirected graph $G = (V, E)$, a weight function $d : E \to \mathbb{R}_+$ and an integer $K > 0$, the Distance Geometry Problem (DGP) aims to find a *realization* $x : V \to \mathbb{R}^K$ such that

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

where $\|.\|$ denotes the Euclidean norm, $x_u := x(u)$ and $d_{uv} := d(\{u, v\})$. A map $x : V \to \mathbb{R}^K$ satisfying (1) is called a *valid realization* [6]. Henceforth, we will consider that (1) admits a solution.

It is well-known that DGP is NP-Hard [10] but, recently, efficient methods have been developed for particular classes of DGPs [1, 4]. An important class of DGPs that arises in protein conformation problems ($K = 3$) is the one of Discretizable Molecular Distance Geometry problems (DMDGP) [4].

Definition 1 *We say that a DGP is a ^{K}DMDGP if there exists an order $(v_1, v_2, \ldots, v_n)$ for the vertices of G, where $n = |V|$, such that: 1) $G[\{v_1, \ldots, v_K\}]$ is a clique; 2a) for every $i > K$, v_i is adjacent to $v_{i-1}, v_{i-2}, \ldots, v_{i-K}$; 2b) $CM(v_{i-1}, \ldots, v_{i-K})^2 > 0$.*

In Definition 1, $G[.]$ denotes the induced subgraph and $CM(v_{i-1}, \ldots, v_{i-K})$ is the Cayley-Menger determinant (see [6, Sec. 2]) whose squared value is proportional to the $(K-1)$-volume of a realization for $v_{i-1}, \ldots, v_{i-K}$.

Although this definition considers an arbitrary dimension K, in this paper we focus on protein conformations where $K = 3$ and denote a ^{3}DMDGP simply by DMDGP. Sometimes we present figures considering $K = 2$ for easier visualization.

Properties 1 and 2 allows one to turn the search space for realizations into a discrete one in the following way. After fixing the positions of the first three vertices, for each new vertex v_i,

with $i > 3$, property 2(a) ensures that the possible positions x_i for v_i lie in the intersection of spheres centered at $x_{i-1}, x_{i-2}, x_{i-3}$ with radii $d_{i,i-1}, d_{i,i-2}, d_{i,i-3}$, respectively. Then, property 2(b) guarantees that there are at most two points, let us say $\{x_i^+, x_i^-\}$, in such intersection. Thus, following the vertex order, this process leads to a binary tree of possible positions: each path in this tree, from the root to a leaf node, corresponds to a possible realization for G.

However, notice that not all of these possible realizations (paths on the tree) are valid, because Definition 1 does not involve all edges of G. We will call the edges in Definition 1 *discretization edges* and the others that may be available *pruning edges*. The last ones can be used to validate partial realizations at certain levels of the search tree. Therefore, if we explore the binary tree in a depth first manner and validate the possible positions as soon as a pruning edge appears, then we arrive at the so called *Branch-and-Prune* (BP) algorithm [5]. It calls itself recursively in order to explore the search tree, pruning infeasible paths/branches by checking whether pruning edges $\{h, i\}$ are approximately satisfied: $|\|x_h - x_i\| - d_{hi}| \leq \varepsilon$, where $\varepsilon > 0$ is a prescribed tolerance.

Computational experiments in [4] showed that BP outperforms methods based on continuous optimization [9] and semidefinite programming [3] on instances of the DMDGP subclass.

In the last decade, some works [7, 8] studied the symmetries of DMDGPs, proving interesting results. For example, it is possible to count the number of solutions before solving the problem [8]. Furthermore, once a solution is found, all others may be built from it by considering *partial reflections* of that realization through its *symmetry planes* [7]. Naturally, other works (e.g., [2]) exploited such symmetry properties in order to solve the problem more efficiently.

However, none of these previous works *completely* exploited the symmetries for finding the *first* solution of a DMDGP instance. Here we address this issue by considering the whole problem as a sequence of nested subproblems, each one defined by a pruning edge $\{i, j\}$.

2 DMDGP symmetries and the new algorithm

Real life DMDGP instances, e.g., the ones from protein conformation problems, are often composed by many *nested or overlapped* DMDGP subproblems [7]. Notice that each pruning edge $\{i, j\}$ defines a sub-instance $G[v_i, \ldots, v_j]$ which, thanks to Definition 1, is itself a DMDGP instance.

Let us review some theoretical results from [7, 8] that are the base of our algorithm. Let X denote the set of all incongruent solutions of a DMDGP and $x \in X$ a particular realization. Given $x \in X$, for $i > 3$, let $R_x^i(y)$ be the reflection of $y \in \mathbb{R}^3$ through the plane containing $x_{i-1}, x_{i-2}, x_{i-3}$ with normal p_i. Let us also partition $E = E_D \cup E_P$ where E_D is the set of discretization edges and E_P the set of pruning edges and, for all $i > 3$, define *partial reflection* operators $g_i(x) = (x_1, x_2, \ldots, x_{i-1}, R_x^i(x_i), R_x^i(x_{i+1}), \ldots, R_x^i(x_n))$.

Proposition 1 *Let (G, d, K) be a feasible DMDGP instance with vertex order $(v_1, \ldots, v_n)$. With probability 1, for all $j > 3$ and $i < j-3$, the DMDGP subproblem defined by edge $\{v_i, v_j\}$ has two solutions.*

In Proposition 1, "with probability 1" means that the set of DMDGP instances for which the statements do not hold has Lebesgue measure zero in the set of all DMDGP instances [8]. The proof of Proposition 1 follows from a result in [7, Thm. 3.2]. Figure 1 illustrates the analogous of Proposition 1 for dimension $K = 2$.

The two solutions of subproblem $\{i, j\}$ are reflections of each other through the plane defined by x_i, x_{i+1}, x_{i+2}. Hence, once a subproblem is solved, if it is part of a larger DMDGP subproblem, it can be handled as a *rigid body*, meaning that we only need to store its first symmetry plane in order to compose with other partial reflections of the larger subproblem.

Another useful result for solving a DMDGP subproblem is the following.

Proposition 2 *Let x_i, x_k be such that $k > i+3$ and $y \in \mathbb{R}^3$ such that y is not in the hyperplanes containing the origin and normal to p_i, p_k. Then $R_x^i(R_x^k(y)) = R_{g_i(x)}^k R_x^i(y)$.*

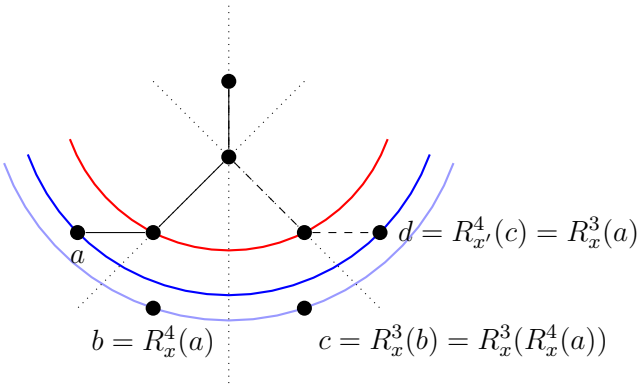

FIG. 1: The leftmost path/realization x is represented by a straight line whereas the rightmost x' by a dashed line. All 4 possible positions for the fourth vertex (denoted by a, b, c and d) can be generated by x and its induced reflections R^3_x and R^4_x. Illustration for $K = 2$. The concentric arcs centered at vertex 1 position have radii equal to the distance between vertex 1 and vertices 3 and 4.

Algorithm 1 SBP

SBP$(n, G(V, E), d)$
for each $\{i, j\} \in E_P$ **do**
  if $c(i) \neq c(j)$ **then**
    Initialize candidate positions x_k for $k = i, \ldots, j$
    Merge the clusters covered from i to j and update c
    Retrieve the P_{ij} necessary symmetry planes and the corresponding reflectors
    for each g of the $2^{P_{ij}}$ partial reflection compositions **do**
      if $\|g(x_j) - x_i\| = d_{ij}$ **then**
        Go to Step 12
      end if
    end for
    Apply the reflections composing g to all candidate positions from $j - 1$ to $i + 3$.
  end if
end for

Due to Propositions 1 and 2, we stress that in order to solve a DMDGP subproblem, we do not need to recompute all the positions every time a path ends up in a infeasible leaf node. As soon as we obtain a realization, *valid or not* (e.g., taking the leftmost path in the sub-tree), we can build all necessary symmetry planes and their corresponding reflection operators. Then, we can apply compositions of such partial reflections *only* to the *last* vertex in the sub-structure until we find its correct position, as illustrated in Figure 1 for the 2D case. Only after we find the correct composition for the last vertex, we apply it to all vertices in the subproblem to retrieve their position.

Since each pruning edge $\{i, j\}$, defines a DMDGP subproblem, we propose to handle the pruning edges, one at a time, following a specific order: they are sorted in increasing order of j, followed by a decreasing order of i. Such pruning edge order allows us to compute candidate vertex positions in the natural order $1, 2, \ldots, j$, when tackling edge $\{i, j\}$, which avoids the work of global alignment because all subproblems are solved in the same referential.

After the inner-most subproblems are solved, their structures/realizations can be merged in realizations of larger subproblems containing them and be treated as rigid bodies. In order to control which subproblems are already solved, we merge subproblems $\{u, v\}$ with $\{i, j\}$ when either $\{u, v\}$ is already solved and $i < u < v < j$, or $u < i < v < j$ and $v - i > 3$, giving rise to a "cluster" of the corresponding vertices. The information of which cluster each vertex i belongs is store in the variable $c(i)$. Initially, $c(i) = i, \forall v_i \in V$.

Algorithm 1 summarizes the above ideas.

			BP		SBP		
ID	$\|V\|$	$\|E\|$	time (s)	MDE	time (s)	MDE	Speed-up
1ADX	120	659	3.33E-04	6.41E-07	5.01E-05	2.53E-12	6.65E+00
1BDO	241	1345	3.22E-04	2.62E-08	8.49E-05	1.04E-11	3.79E+00
1ALL	480	3443	9.43E-04	3.91E-07	2.02E-04	1.27E-12	4.67E+00
1FHL	1002	6378	7.30E-03	4.02E-13	4.67E-04	1.17E-11	1.56E+01
6RN2	2052	13710	1.83E-02	6.02E-14	8.25E-04	9.35E-12	2.22E+01
1EPW	3861	23191	3.19E-01	4.21E-09	2.63E-03	9.78E-11	1.21E+02

TAB. 1: Computational results in some protein-like instances (cut-off: 5Å).

3 Computational results

We generate a set of protein-like instances whose data were extracted from the Protein Data Bank (PDB)[1], and compare the results with those of the classic BP [5, 4]. We consider only the protein backbone composed by the sequence of atoms $N - C_\alpha - C$ and add an edge either when the atoms are separated by at most three covalent bonds or the distance between atom pairs is smaller than 5 Å. The natural backbone order for instances generated in this way provides a vertex order satisfying the assumptions of Definition 1.

In Table 1, we present the results obtained by BP: the classic BP implementing a depth-first search [4, 5], where no symmetry is exploited at all and coordinates for the vertices are re-computed every time a backtracking occurs; and Algorithm 1, called SBP. The table shows the number of atoms $|V|$, number of edges (available distances) $|E|$, CPU time in seconds for the two algorithms and the normalized Mean Distance Deviation (MDE) [6]. Both algorithms were stopped as soon as the first solution is found.

From the numerical results we observe that the new algorithm may provide considerable speed-up with respect to the classic BP algorithm. It also seems that the CPU time for SBP varies linearly with the number of edges, but additional experiments and theoretical studies are necessary to support this claim.

References

[1] 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*, 22:365–375, 2002.

[2] F. Fidalgo, D. S. Gonçalves, C. Lavor, L. Liberti, and A. Mucherino. A symmetry-based splitting strategy for discretizable distance geometry problems. *Journal of Global Optimization*, 71:717–733, 2018.

[3] N. Krislock and H. Wolkowicz. Explicit sensor network localization using semidefinite representations and facial reductions. *SIAM Journal on Optimization*, 20:2679–2708, 2010.

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

[5] L. Liberti, C. Lavor, and N. Maculan. A branch-and-prune algorithm for the molecular distance geometry problem. *International Transactions in Operational Research*, 15:1–17, 2008.

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

[7] L. Liberti, C. Lavor, and A. Mucherino. The Discretizable Molecular Distance Geometry Problem seems easier on proteins. In Antonio Mucherino, Carlile Lavor, Leo Liberti, and Nelson Maculan, editors, *Distance Geometry*, pages 47–60. Springer New York, 2013.

[8] L. Liberti, B. Masson, J. Lee, C. Lavor, and A. Mucherino. On the number of realizations of certain Henneberg graphs arising in protein conformation. *Discrete Applied Mathematics*, 165:213–232, 2014.

[9] J. J. Moré and Z. Wu. Distance geometry optimization for protein structures. *Journal of Global Optimization*, 15:219–234, 1999.

[10] 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*, pages 480–489, Monticello, IL, 1979.

[1] https://www.rcsb.org/