Discretization vertex orders in distance geometry

Andrea Cassioli[1], Oktay Günlük[2], Carlile Lavor[3], Leo Liberti[2,4]

[1] *MOSEK ApS, Fruebjergvej 3 2100 Copenhagen, Denmark*
Email:andrea.cassioli@mosek.com

[2] *IBM Research, Yorktown Heights, 10598 NY, USA*
Email:{gunluk,leoliberti}@us.ibm.com

[3] *Dept. of Applied Math. (IMECC-UNICAMP), Univ. of Campinas, 13081-970, Campinas – SP, Brazil*
Email:clavor@ime.unicamp.br

[4] *CNRS LIX, École Polytechnique, F-91128 Palaiseau, France*
Email:liberti@lix.polytechnique.fr

September 15, 2014

Abstract

When a weighted graph is an instance of the Distance Geometry Problem (DGP), certain types of vertex orders (called discretization orders) allow the use of a very efficient, precise and robust discrete search algorithm (called Branch-and-Prune). Accordingly, finding such orders is critically important in order to solve DGPs in practice. We discuss three types of discretization orders, the complexity of determining their existence in a given graph, and the inclusion relations between the three order existence problems. We also give three mathematical programming formulations of some of these ordering problems.
Keywords: molecular conformation, protein, DDGP, DMDGP, re-order, Branch-and-Prune.

1 Introduction

The Distance Geometry Problem (DGP) is as follows: given a positive integer K and a simple, undirected, nonnegatively weighted graph $G = (V, E, d)$, where $d : E \to \mathbb{R}_+$, find a realization $x : V \to \mathbb{R}^K$ such that:

$$\forall \{u, v\} \in E \quad \|x_u - x_v\|_2 = d_{uv}. \tag{1}$$

If G is disconnected then realizing G is the same as realizing its connected components, so we assume G is connected.

Solution methods for the DGP generally involve a search in continuous space [19]. On the other hand, several applications of DGP supply some guarantees on the sparsity structure of the input graph, which may in turn imply graph rigidity. The solution set is then a finite subset of a Euclidean space, which allows for remarkable performance improvements of the solution algorithms. Many methods are iterative in nature: they assume a small subset of vertices have known positions, and try and infer the position of the rest of the vertices in some order. Thus, vertex orders play an important role. Trilateration orders, for example, guarantee that every vertex beyond the first $K + 1$ is adjacent to at least $K + 1$ predecessors [7]. This makes it possible to exactly triangulate the position of each next vertex. This implies a polynomial time algorithm and a unique solution modulo translations and rotations.

The main focus of this paper is to determine the worst-case complexity class of many vertex ordering problems used in algorithms for solving the DGP on certain rigid graphs. We also propose and test three Mixed-Integer Linear Programming (MILP) formulations for solving such vertex ordering problems, and empirically determine that they can only be useful for small-scale instances.

1.1 Some applications of vertex orders

Vertex orders in graphs are of paramount importance in all sorts of applications, from graph search to scheduling to graph drawing. The Depth-First Search (DFS) and Breadth-First Search (BFS) vertex orders occur whenever a graph search algorithm is deployed [1]. One of the most celebrated problems in combinatorial optimization, the Travelling Salesman Problem (TSP), asks for a certain order on the vertices of a complete weighted graph. Tasks and their precedences are often represented as graphs in scheduling, which yields partial or total vertex orders [9]. A vertex order is *balanced* if the neighbours of every vertex v are as evenly distributed to the left and right of v as possible [2]: balanced orders are used in graph drawing. Different vertex orders are also used to represent the adjacency matrix of graphs in a meaningful way [28]. Dynamic programming type algorithms can be used to solve a large class of vertex ordering optimization problems in "moderately exponential time" (i.e. the running time is asymptotically exponential, but faster than exhaustive search), as shown in [3].

1.2 Vertex orders in protein conformation

The function of proteins is strongly related to their chemical composition and their three-dimensional structure: proteins usually fold in space until they reach a stable configuration having low potential energy. Finding their 3D structure is therefore an important task in pharmaceutical research. Many approaches exist [31]: in this paper we adopt the point of view of distance geometry [4, 5, 27].

We represent a protein by means of a graph where vertices represent atoms and edges are present if the distance between two adjacent atoms is known. Atomic distances may be known for chemical or physical reasons, or because they were estimated using Nuclear Magnetic Resonance [32]. Proteins consists of chains of amino acids, which come in twenty different types. Amino acids consist of a common structure: a small chain starting with the amino group H_3N, followed by the α carbon C_α (linked to a hydrogen atom and to a side chain), followed by the carboxyl group COO^-. Amino acids only differ because of their side chains. The whole protein can then be seen as a backbone consisting of a chain formed by the repeated common structures of each amino acid, and many dangling side chains. To a large extent, the problem of finding a 3D realization of the protein can be decomposed into the subproblems of realizing the backbone and, separately, the side-chains; and then try to combine the partial realizations in a consistent way [30].

Protein backbones enforce an order on the atoms in the backbone. This order has some interesting properties: we know the distance of each atom v to its predecessor $v-1$, since covalent bond lengths are known for chemical reasons. Since covalent bond angles are also known, for every triangle of three consecutive atoms we know two of its side lengths and the angle between them: thus we can also compute the length of its third side, i.e. the distance between v and $v-2$. Moreover, NMR can estimate all distances up to a certain threshold (around 5.5Å), and it is known that distances between atom v and $v-3$ are always below this threshold, so the distance between v and $v-3$ is also known. This order makes the protein graph look like a chain of embedded cliques of size 4 (realized as 3-simplices), each sharing a face with the preceding one, plus some other scattered edges called *pruning edges*. Pruning edges are due to the fact that when a protein backbone folds in space, two atoms might come to be physically close even though they are be very distant in terms of their ranks in the backbone order. In particular, their Euclidean distance becomes known when it is below the NMR threshold. Orders were each vertex is adjacent to three predecessors have been shown to yield rigid structures in early 1900s [11].

1.3 The Branch-and-Prune algorithm

This order was instrumental in devising a discrete method called Branch-and-Prune (BP) for finding the 3D realization of protein backbones [18]. Although the BP was not the first discrete method for this problem [6], it was the first which could find all incongruent solutions to any given problem instance.

Most methods previously proposed in the literature, by contrast, were searches in continuous space (see [21] and references therein).

The principle behind the BP is that any 3D simplex on the vertices $\{v, v-1, \ldots, v-3\}$ (for some vertex v) has two distinct realizations modulo translations and rotations: supposing that the 3D position of vertices $v-1, v-2, v-3$ is known, vertex v can be reflected across the unique plane containing the points $v-1, v-2, v-3$. So if we suppose that the first three atoms have known positions, we can recursively place the remaining atoms by exploring each of the two possible positions at each step (branching step). Those positions which are inconsistent with the distances assigned to the pruning edges are pruned out (pruning step). This yields a method which is exponential in the worst case: if there are no pruning edges, BP yields a binary tree with 2^{n-3} nodes, where n is the number of atoms in the protein. It was recently shown in [20] that this order makes the BP a Fixed-Parameter Tractable (FPT) algorithm.

1.4 Discretization of distance geometry problems

We generalize the backbone order to define an order for K-dimensional spaces: each vertex is adjacent to at least K (rather than $K+1$) predecessors [19] (where, specifically for proteins in 3D space, $K = 3$). This number of adjacent predecessors (K) is critical: any fewer, and the solution set might be uncountable in general, since the graph may no longer be rigid [10]; any more, and the corresponding DGP subclass can be solved in polynomial time via trilateration [7].

As discussed above, in protein graphs the adjacent predecessors of any vertex v immediately precede v. This is an important feature: if v has K adjacent vertices that immediately precede v in the order, they are called *contiguous predecessors* of v (those which follow v are called *contiguous successors*). [19]. In summary:

1. the first K vertices in the order form a clique;
2. each vertex with rank greater than K is adjacent to at least K predecessors, exactly K of which are contiguous.

We call the class of DGP instances possessing these orders (and satisfying the strict triangular inequalities on the edge weights [19]) the DISCRETIZABLE MOLECULAR DISTANCE GEOMETRY PROBLEM in $\mathbb{R}^K$ (^{K}DMDGP), and the orders themselves *^{K}DMDGP orders*. In [14], the BP algorithm was extended to the ^{K}DMDGP. In [25], it was shown that the BP algorithm could also be used for a larger class of instances, the DISCRETIZABLE DISTANCE GEOMETRY PROBLEM (DDGP): the DDGP is the subclass of DGP instances for which an order exists (called *DDGP order*) such that Requirement 1 above holds, and a relaxation of Requirement 2 holds, where the K adjacent predecessors need not be contiguous. Both the DDGP and the ^{K}DMDGP are **NP**-hard problems [25, 14], by reductions from SUBSET-SUM. It was shown in [25] that ^{K}DMDGP $\subsetneq$ DDGP (problem P is included in problem Q if the two problems have the same input, and YES instances of P are also YES in Q).

Given a DDGP input graph, the problem of finding a DDGP order is at least as hard as CLIQUE (the problem of determining whether a graph has a clique of size K, which is itself **NP**-complete), since an initial clique of size K must be found for the order to start. Finding the rest of a DDGP order, however, is easier: there is an FPT algorithm to find DDGP orders whose time complexity depends exponentially on K but polynomially on the graph size [13]. Since K is fixed to the constant 3 for proteins in 3D space, this yields a polynomial time algorithm for finding DDGP orders on protein instances. One of the new results proved in the present paper is that establishing the existence of a ^{K}DMDGP order is an **NP**-complete problem, by reduction from HAMILTONIAN PATH. Moreover, in contrast to DDGP orders, we also prove that ^{K}DMDGP orders are hard to find for any fixed K.

Yet, the ^{K}DMDGP has two features that make it more convenient to handle with respect to the DDGP. For protein instances, the fact that the BP algorithm need only consider K contiguous predecessors makes it more likely that the distances between a vertex v and its K adjacent predecessors are well-scaled with respect to one another. This decreases the incidence of numerical floating point error whilst solving linear systems within the BP algorithm, which is a very desirable feature of the ^{K}DMDGP: we empirically found that computational errors when using BP on DDGP instances sometimes made their solution impossible. Moreover, in general, the symmetry structure of the ^{K}DMDGP is known [22], whereas the symmetry structure of the DDGP is still unknown. This motivates the search for orders involving contiguous adjacent predecessors, rather than just adjacent predecessors, even for DDGP instances. The compromise proposed in [16] (limited to $K = 3$) is to allow the repetition of some selected vertices in the order, so that at least K adjacent predecessors can always be chosen to be contiguous. Such orders are called *repetition orders* (or *re-orders*). We call the class of DGP instances that can be discretized this way Re-order Discretizable Distance Geometry Problem (RDDGP). Although hand-crafted periodic re-orders were used in [16] to discretize entire protein backbones, the relationship between DDGP, RDDGP, and ^{K}DMDGP was not investigated in depth.

1.5 Determination of vertex orders

Since these DGP variants only essentially differ in the vertex order they employ, we focus here on the inclusionwise relations between the corresponding order existence problems: respectively, the problem of finding a DDGP order, called[1] the Trilateration Ordering Problem (TOP); the Re-Order Problem (ReOP); and the problem of finding a ^{K}DMDGP order, called Contiguous TOP (CTOP). Specifically, we show that CTOP $\subsetneq$ ReOP $\subsetneq$ TOP. From this relationship, the corresponding relationship follows for ^{K}DMDGP, RDDGP, and DDGP.

1.6 Paper summary

The rest of this paper is organized as follows. In Sect. 2 we introduce some technical notation and definitions. In Sect. 3 we prove that finding DDGP orders is **NP**-complete, and finding ^{K}DMDGP orders and re-orders is **NP**-complete for any fixed K. The inclusion relationships between vertex discretization ordering problems is discussed in Sect. 4. In Sect. 5 we discuss the issue of solving these problems in practice, and propose three mathematical programming formulations for the CTOP.

2 Notation and definitions

Given a simple undirected graph $G = (V, E)$, and sets $E' \subseteq E$ and $V' \subseteq V$, we denote by $G[V']$ and $G[E']$ the sub-graphs of G induced by V' and E', respectively. For a graph $G = (V, E)$ and a subgraph $H = (U, F)$ of G, H is *spanning* if $U = V$.

We let $N_G(v)$ be the set of vertices adjacent to v in G, written $N(v)$ if there is no ambiguity. For a positive integer ℓ, we let $N_G^\ell(v)$ be the set of vertices in V whose shortest path to v has exactly ℓ edges (again, this is written $N^\ell(v)$ if there is no ambiguity). We extend N_G^ℓ to act on sets of vertices: let $W \subseteq V$, then $N_G^\ell(W)$ (also denoted $N^\ell(W)$ if no ambiguity arises) is the set of vertices $v \in V$ such that the minimum over all the shortest paths to $w \in W$ contains exactly ℓ edges.

For a given total order relation $<$ on V, we denote by $\gamma(v)$ the set of predecessors of v, and by $\rho(v) = |\gamma(v)| + 1$ the rank of v with respect of $<$ in V. To simplify notation, if $v \in V$ and $p \in \mathbb{N}$, we write $v \pm p$, $v > p$ and $v < p$ to mean, respectively, $\rho(v) \pm p$, $\rho(v) > p$ and $\rho(v) < p$. For each $v \in V$ we let $U(v) = N(v) \cap \gamma(v)$.

If β is a non-empty sequence of vertices in V containing a given vertex $v \in V$, then we may write β as

[1] This order is also called Discretizable Vertex Ordering Problem (DVOP) in [13, 19].

(β^1, v, β^2), where β^1 is the sequence of all predecessors of v in β, and β^2 is the sequence of all successors of v in β, in the same order as they appear in β.

2.1 Definition
Given a simple undirected graph $G = (V, E)$ and a positive integer K, a DDGP order for G is defined as follows:

1. *$G[\{v \in V \mid v \leq K\}]$ is a clique of size K;*
2. *for each $v \in V$ with $v > K$, $|U(v)| \geq K$.*

The problem of determining whether G has a DDGP order is known as TOP (see Sect. 1.5).

2.2 Definition
A ^{K}DMDGP order for G is defined as follows:

1. *$G[\{v \in V : v \leq K\}]$ is a clique of size K;*
2. *for each $v \in V$ with $v > K$, all vertices of rank $v - K \leq w \leq v - 1$ are adjacent to v.*

The problem of determining whether G has a ^{K}DMDGP order is known as CTOP (see Sect. 1.5).

2.3 Definition
A re-order for G is a sequence $\sigma_V = (v_1, v_2, \ldots, v_m)$ of m not necessarily distinct vertices from V such that:

1. *$G[\{v_1, \ldots, v_K\}]$ is a clique of size K;*
2. *for all $i \in \{K+1, \ldots, m\}$ and $j \in \{i-K+1, \ldots, i-1\}$, $\{v_j, v_i\} \in E$*
3. *for all $i \in \{K+1, \ldots, m\}$, either $\{v_{i-K}, v_i\} \in E$ or $v_{i-K} = v_i$*
4. *$m \geq |V|$*
5. *m is bounded by a polynomial in $|V|$*
6. *every vertex of V is present in the sequence σ_V.*

The problem of determining whether G has a re-order is known as ReOP *(see Sect. 1.5).*

Of course, not all graphs possess ^{K}DMDGP, DDGP orders or re-orders: hence the interest in solving the corresponding existence problems.

Some remarks are in order.

- The TOP, CTOP and ReOP are trivially in **NP**, since testing whether a vertex order has the required properties can obviously be done in polynomial time.
- The definition of re-order given in [16] is limited to $K = 3$ and lists one more condition (i.e. that $\{v_{i-K}, v_i\}$ might correspond to an interval of distance values instead of a single value) for practical purposes relating to the distribution of precise and imprecise distances in protein distance data. In this paper we only consider this more abstract definition of re-order.

2.1 Some structural properties of ^{K}DMDGP orders

^{K}DMDGP instances exhibit a remarkable structure. In [22], it was shown that the cardinality of the solution set of ^{K}DMDGP instances is almost certainly a power of two. [20] provides a description of the group structure of the partial reflection symmetries of ^{K}DMDGP instances; as a consequence, the BP algorithm is shown to be Fixed Parameter Tractable (FPT) on the ^{K}DMDGP. An algorithm to efficiently count the number of solutions of ^{K}DMDGP instances is provided in [17]. An improvement to the BP efficiency based on exploiting symmetries was described in [26]. In this section we make some observations on the structure of ^{K}DMDGP orders.

In any ^{K}DMDGP order, the first vertex is adjacent to at least K other vertices: $K-1$ because of the initial clique of size K, together with the vertex whose rank is $K+1$, by Condition 2. The second vertex is adjacent to at least $K+1$ other vertices: $K-1$ because of the initial clique, and then the $(K+1)$-st and $(K+2)$-nd vertices by Condition 2, and so on until the K-th vertex, which is adjacent to $2K-1$ other vertices. All vertices from the $(K+1)$-st to the $(n-K)$-th are adjacent to at least $2K$ other vertices, because of Condition 2 applied to themselves and their $K-1$ contiguous successors. The $(n-K+1)$-st vertex is adjacent to at least K contiguous predecessors and to all of its $K-1$ contiguous successors, for a total of at least $2K-1$ adjacencies. For similar reasons, the $(n-K+2)$-nd vertex is adjacent to at least $2K-2$ other vertices, and so on until the last vertex, which is adjacent to at least K predecessors by Condition 2. Thus, we have the following result.

2.4 Lemma
The following lower bounds on the adjacencies of the vertices of G hold for ^{K}DMDGP orders defined on a graph $G=(V,E)$:

1. *$\forall v \in V$ such that $1 \le v \le K$, $|N(v)| \ge K+v-1$;*
2. *$\forall v \in V$ such that $K < v \le n-K$, $|N(v)| \ge 2K$;*
3. *$\forall v \in V$ such that $n-K < v \le n$, $|N(v)| \ge K+n-v$.*

By Lemma 2.4, any graph with a vertex distribution that does not comply with these bounds is a NO instance of the CTOP problem, while the converse may not hold [23].

Next, we show that ^{K}DMDGP orders are symmetric.

2.5 Proposition
Let α be a ^{K}DMDGP order on $G=(V,E)$; the inverse order α^{-1} is also a ^{K}DMDGP order on G.

Proof. The last K vertices α are the first K vertices of α^{-1}. For any $u,v \in V$ such that $n-K+1 \le u < v \le n$, since $v-u \le K$ we have $u \in U(v)$, which means that $\{u,v\} \in E$ by Condition 2; thus $\{n-K+1,\ldots,n\}$ is a clique of size K. Now consider a vertex $v \in V$ such that $1 \le v \le n-K$. Because α is a ^{K}DMDGP order, for every $j \in \{1,\ldots,K\}$ the vertices ranked $v+j$ are also adjacent to v, either because $v \in U(v+j)$ or because v and $v+j$ are in the initial clique of size K of α. Hence v is also adjacent to its K contiguous successors, as claimed. □

3 Complexity of order existence problems

In this section we discuss the complexity of the three order existence problems defined above.

3.1 NP-completeness of TOP

In [13], it is mentioned that the TOP is **NP**-complete by reduction from Clique. Here we give a more detailed proof. We start with an instance (G, K) of Clique, where $G = (V, E)$. We transform G into a graph $G' = (V', E')$ by adding $K-1$ dummy vertices to V, so that each dummy vertex is adjacent to every vertex in V. Formally, let $U = \{u_1, \ldots, u_{K-1}\}$ be such that $U \cap V = \varnothing$, let $V' = V \cup U$, and let $E' = E \cup \{\{u, v\} \mid u \in U \wedge v \in V\}$. We aim to show that G is YES in Clique if and only if G' is YES in TOP.

3.1 Proposition
The TOP is **NP**-*complete.*

Proof. Let (G, K) be an instance of Clique, where $G = (V, E)$ with $V = \{v_1, \ldots, v_n\}$, and G' the corresponding instance of TOP. Assume G is a YES instance; after a suitable vertex relabelling, the clique of size K can be taken to be $C = \{v_1, \ldots, v_K\}$. We start the order with $(v_1, \ldots, v_K, u_1, \ldots, u_{K-1})$: each u_i is adjacent to all K vertices in C by construction. Let $W_0 = C$; next, we list (in any order) all the yet unlisted vertices in $N^1(W_0)$: by definition, each is adjacent to at least one vertex in W_0, which precedes it in the order, and all of the vertices in U, which also precede it in the order; and we set $W_1 = C \cup N^1(W_0)$. An easy induction on ℓ (appearing in the $N^\ell(\cdot)$ operator) shows that this procedure can be carried out for any ℓ as long as $N^\ell(W_{\ell-1}) \neq \varnothing$. We thus obtain a DDGP order on G'. Now let G be a NO instance. Suppose, to get a contradiction, that G' has a DDGP order $(v'_1, \ldots, v'_{n+K-1})$. Its initial clique C of size K cannot be such that $C \subseteq V$, for otherwise G would be trivially a YES instance, against the assumption. Also, C cannot contain more than one vertex in U, since U is by definition a stable set in G'. Hence C necessarily contains exactly one vertex in U and $K-1$ vertices in V, which implies that G has a $K-1$ clique C'. Now consider v'_{K+1}: since vertices in U are pairwise non-adjacent, and v'_{K+1} must be adjacent to all vertices in C, $v'_{K+1} \notin U$. So then $v'_{K+1} \in V$. But since v'_{K+1} is adjacent to all vertices in C then it is adjacent to all vertices in $C' \subseteq C$, which means that $C \cup \{v'_{K+1}\}$ is a clique of size K in V, against the assumption. So G' cannot contain a DDGP order, as claimed. □

The proof of Prop. 3.1 essentially states that the reason why TOP is hard has something to do with finding the initial clique of size K. This is confirmed by the fact, already mentioned above and in [13, Prop. 2], that if K is fixed then the TOP can be solved by a polynomial-time algorithm.

3.2 NP-completeness of CTOP

We show that the CTOP is **NP**-complete for every fixed K by reduction from Hamiltonian Path (HP). We start with the easy case $K = 1$, where a Hamiltonian path *is* also a ^{1}DMDGP order.

3.2 Proposition
The ^{1}*DMDGPO is* **NP**-*complete.*

Proof. Given $G = (V, E)$ with $n = |V|$, if G is a YES instance of HP, then there is a Hamiltonian path $\alpha = (v_1, \ldots, v_n)$, which also induces a total order on V. Since for all $v \in V$ such that $v > 1$, v is adjacent to its predecessor in the order, this order is also a ^{1}DMDGP order. Conversely, assume G is a NO instance of HP and suppose $\alpha = (v_1, \ldots, v_n)$ is a ^{1}DMDGP order. Then by definition α is also a Hamiltonian path in G, which is impossible. □

Next, we show that the CTOP is **NP**-complete for any fixed positive integer K. In order to prove that finding the order is hard, rather than simply finding the initial clique of size K, we consider suitable variants of Hamiltonian Path (HP) and of CTOP. More precisely, the restricted HP (rHP) problem contains HP instances paired with a given vertex s, which is required to be the starting vertex of the

Hamiltonian path if it exists. The RESTRICTED CTOP (rCTOP) contains CTOP instances paired with a given clique C_s of size K, which is required to be the initial clique of size K of the ^{K}DMDGP order if it exists. The fact that the rHP is **NP**-complete follows immediately from the **NP**-completeness HP, since solving the rHP on n vertices essentially requires solving a HP instance on $n-1$ vertices. And of course the rCTOP can be trivially reduced to a smaller CTOP (by removing the initial clique C_s).

The reduction from rHP to rCTOP replaces each vertex v of the rHP instance $G=(V,E)$ with a clique C_v of size K, and each edge $\{u,v\}$ with a clique C^{uv} of size K and two bicliques $(C_u,C^{uv}),(C^{uv},C_v)$ (see Fig. 1). More precisely, for a given graph $G=(V,E)$ and a given positive integer K, we define the graph $G'=(V',E')$ as follows. For each $v\in V$ let $C_v=\{u_{v1},\ldots,u_{vK}\}$ and $F_v=\{\{w,z\}\mid w\neq z\in C_v)\}$, so that C_v is a clique for each $v\in V$. For each $\{v,w\}\in E$ let $C^{vw}=\{u_1^{vw},\ldots,u_K^{vw}\}$, $H_1^{vw}=\{\{u_{vi},u_j^{vw}\}\mid i,j\leq K\}$ $H_2^{vw}=\{\{u_i^{vw},u_{wj}\}\mid i,j\leq K\}$ define the motif biclique-clique-biclique joining two cliques C_v,C_w. Now, we let:

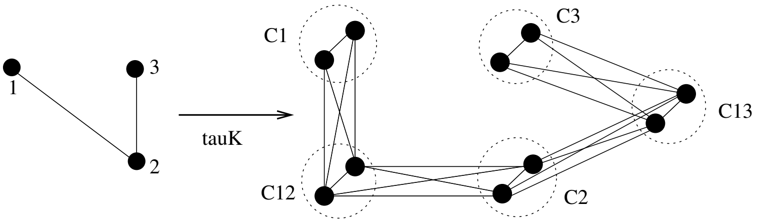

Figure 1: The reduction from rHP to rCTOP, for $K=2$.

$$\begin{aligned} V' &= \left(\bigcup_{v\in V} C_v\right)\cup\left(\bigcup_{\{v,w\}\in E} C^{vw}\right)\\ E' &= \left(\bigcup_{v\in V} F_v\right)\cup\left(\bigcup_{\substack{\{v,w\}\in E\\ i\in\{1,2\}}} H_i^{vw}\right). \end{aligned}$$

We let τ_K be the transformation taking G to G'. Notice that the transformation τ_K^{-1} is well-defined over the range of τ_K: simply contract each clique C_v to its originating vertex v (for $v\in V$), and contract each biclique-clique-biclique motif to the corresponding edge. This shows that $\tau_K^{-1}(\tau_K(G))=G$.

A ^{K}DMDGP order α in $G'=\tau_K(G)$ is *consecutive* if, for any clique C_v of size K in G' corresponding to a vertex v of G, all vertices of C_v are listed consecutively in α. First, we show that any Hamiltonian path in G can be transformed into a consecutive ^{K}DMDGP order in $\tau_K(G)$.

3.3 Lemma
Let $(G=(V,E),s)$ be a YES instance of rHP, and $K>1$. Then $G'=\tau_K(G)$ has a consecutive ^{K}DMDGP order starting with C_s.

Proof. Let $\beta=(v_1,\ldots,v_n)$ be a Hamiltonian path in G starting with $s=v_1$. Relabel each v_i to i, so that $\beta=(1,2,\ldots,n)$. Let

$$\alpha=(u_{11},\ldots,u_{1K},u_1^{12},\ldots,u_K^{12},u_{21},\ldots,u_{2K},\ldots,u_{n1},\ldots,u_{nK}).$$

Since $\{i-1,i\}\in E$ for each $i>1$, given any vertex u of a clique C of size K in G', at least K contiguous predecessors of u are adjacent to it by definition of τ_K, as claimed. □

Next, we show that $\tau_K(G)$ has no triangles involving vertices of three different cliques.

3.4 Lemma
Let C, C', C'' be three cliques of size K in $G' = \tau_K(G)$. There is no vertex triplet $u \in C$, $u' \in C'$, $u'' \in C''$ inducing a triangle in G'.

Proof. Cliques of size K in G' correspond to either vertices or edges in G: let us call the former "vertex cliques" and the latter "edge cliques". To reach a contradiction, assume a triangle induced by u, u', u'' exists in G': since all edges in $E(G')$ belong to bicliques, there must be bicliques (C, C'), (C', C''), (C'', C). Since there are only two types of cliques, there must be at least two cliques of size K in C, C', C'' having the same type, which results in bicliques on two vertex cliques or on two edge cliques. However, by definition of τ_K there is no biclique on pairs of vertex cliques, nor on pairs of edge cliques. □

Finally, we show that if $\tau_K(G)$ has a ^{K}DMDGP order, τ_K^{-1} maps it to a Hamiltonian path in G.

3.5 Lemma
Let $(G = (V, E), s)$ be an instance of rHP, $K > 1$, and α be a ^{K}DMDGP order in $G' = \tau_K(G)$. Then G is a YES instance.

Proof. If α is consecutive, then the vertices in all cliques C_v of size K in G' originating from vertices v of G are listed consecutively in G'. Therefore, by contracting C_v to v in $\tau_K^{-1}(\alpha)$ we obtain a Hamiltonian path in G starting with s. Now suppose α is not consecutive: then there must be a clique C of size K in G' such that α lists strictly fewer than K vertices from C consecutively. Let $u_1, \ldots, u_h$ be these vertices (with $h < K$). Let $u' \in C'$ be the predecessor of u_1, and $u'' \in C''$ be the successor of u_h, where C', C'' are cliques of size K in G' induced by the mapping τ_K. By definition of ^{K}DMDGP order, every vertex in α must be adjacent to at least K contiguous predecessors. Thus, for all $j \leq h$, u_j is adjacent to u', and u'' is adjacent to u_j. Moreover, since $h < K$, u'' is adjacent to u'. Thus $\{u', u_j\}, \{u_j, u''\}, \{u', u''\} \in E(G')$ for all $j \leq h$: this defines triangles in G' on vertex triplets belonging to three different cliques, which is impossible by Lemma 3.4. Hence α must be consecutive. □

3.6 Theorem
The rCTOP and CTOP are **NP***-complete for any fixed integer $K > 0$.*

Proof. The first part follows because, by Lemmata 3.3 and 3.5, τ_K maps YES instances of the rHP to YES instances of the rCTOP, and NO instances to NO instances. We reduce a rCTOP instance to a CTOP instance with K fewer vertices by removing the initial clique of size K. □

3.3 NP-completeness of ReOP

Because a re-order which never repeats any vertices is a ^{K}DMDGP order, the ReOP is also **NP**-complete for any fixed K by restriction to the CTOP.

4 Relations between discretization vertex order problems

In this section we discuss the inclusionwise relationship between the three order existence problems mentioned above.

4.1 CTOP $\subsetneq$ TOP

It follows by definition that CTOP $\subseteq$ TOP. Although the strict inclusion was shown in [13, Prop. 2] for a specific instance, we exhibit here a new infinite family of TOP instances that are not in CTOP.

For any pair of positive integers K, n with $n \geq K+3$, consider the graph $\mathcal{S}(K,n)$ whose vertices are $\{1,\ldots,K,\ldots,n\}$ and whose edges are $\{\{u,v\} \mid u < v \leq K\} \cup \{\{u,v\} \mid u \leq K \wedge v > K\}$. The first set in the union defines a clique of size K on $\{1,\ldots,K\}$, and the second set defines $n-K$ cliques on $\{1,\ldots,K,i\}$, each having size $(K+1)$, for all $i \in \{K+1,\ldots,n\}$ (see Fig. 2). It is trivial to verify that any vertex order starting with $1,\ldots,K$ is a DDGP order: simply take $\{1,\ldots,K\}$ as the initial clique of size K, and then the vertices $K+1,\ldots,n$ in any order: by definition of $\mathcal{S}(K,n)$, each such vertex is adjacent to the initial clique of size K.

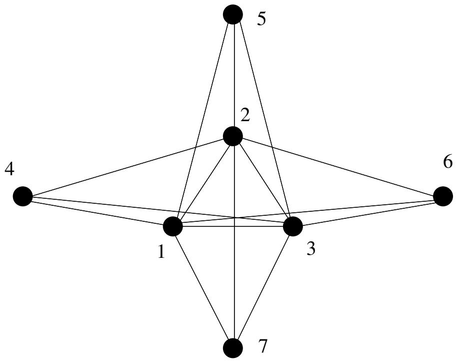

Figure 2: The graph $\mathcal{S}(3,7)$.

4.1 Lemma
For a positive integer K and $n > K+3$, no graph $\mathcal{S}(K,n)$ can have a ^{K}DMDGP order.

Proof. Suppose the contrary holds, and $v_1,\ldots,v_n$ is a ^{K}DMDGP order for $\mathcal{S}(K,n)$. By definition of $\mathcal{S}(K,n)$, either the initial clique of size K is $\{1,\ldots,K\}$ or not. In the former case, then v_{K+1} can be any vertex in $\{K+1,\ldots,n\}$, but since all of these are only adjacent to $\{1,\ldots,K\}$ by construction, and $\{1,\ldots,K\}$ have already been listed, no vertex v_{K+2} can be chosen compatible with the definition of ^{K}DMDGP order. In the latter case, the only possible initial cliques of size K have a vertex u in $\{K+1,\ldots,n\}$ and $K-1$ vertices in $\{1,\ldots,K\}$: say, without loss of generality, that $v_1 \in \{K+1,\ldots,n\}$ and $\{v_2,\ldots,v_K\} = \{1,\ldots,K-1\}$. Then the only possible choice for v_{K+1} is K, since none of the other vertices in $\{K+1,\ldots,n\} \smallsetminus \{v_1\}$ is adjacent to v_1, whereas by definition of the ^{K}DMDGP order, $v_1 \in N(v_{K+1})$. This implies that $v_{K+2} \in \{K+1,\ldots,n\} \smallsetminus \{v_1, v_{K+1}\}$. There can be no next vertex $v_{K+3} \in \{K+1,\ldots,n\} \smallsetminus \{v_1, v_{K+1}, v_{K+2}\}$ since none of them is adjacent to v_{K+2}. The result follows. □

4.2 CTOP $\subsetneq$ ReOP

In the ReOP definition (Defn. 2.3), if no pair of vertices $\{v_{i-K}, v_i\}$ is ever such that $v_{i-K} = v_i$, the order is trivially a ^{K}DMDGP one, thus it follows that CTOP $\subseteq$ ReOP. In Fig. 3 we exhibit a YES instance of ReOP which is NO in ^{2}DMDGPO. Let $G = (V,E)$ be the graph shown in Fig. 3, and let $K = 2$. First, it is easy to verify that $(1,2,1,3,1,4,1,5,1,6)$ is a re-order in G.

On the other hand, we claim that G does not have ^{2}DMDGP orders. Suppose one exists, say $\alpha = (v_1,\ldots,v_6)$: either $v_1 = 1$, or v_1 is one of the other vertices, say $v_1 = 2$ without loss of generality because

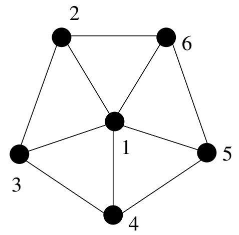

Figure 3: A graph with a re-order but no ^{2}DMDGP orders.

of the symmetry in G. In the former case, v_2 can only be one of the other vertices, say $v_2 = 2$ without loss of generality. Then $v_3 = 3$ or 6 (say $v_3 = 3$). This forces $v_4 = 4$, but this is impossible: in the order $1, 2, 3, 4$, $N(4) = \{1, 3\}$ but 1 is not a contiguous predecessor of 4. In the latter case, v_2 can only be in $\{1, 3, 6\}$. If $v_2 = 1$, then $v_3 = 3$ or 6, without loss of generality take $v_3 = 3$. This forces $v_4 = 4$, and $v_5 = 5$, but this is impossible: in the order $2, 1, 3, 4, 5$, the contiguous predecessors of 5 are $3, 4$, but $\{3, 5\} \notin E$. The remaining case where $v_2 = 6$ is symmetric.

Notice that the example in Fig. 3 can be generalized to any wheel graph with sufficiently many vertices.

4.3 ReOP $\subsetneq$ TOP

We first prove non-strict inclusion.

4.2 Proposition
All YES instances of ReOP *are YES instances of TOP.*

Proof. Let $G = (V, E)$ be a ReOP instance with a re-order $\alpha = (v_1, \ldots, v_m)$. Now construct an order β from α as follows. Scan α, and copy the first K vertices to β. Next, for every $i > K$, copy v_i to β only if it has not already appeared as v_j for some $j < i$. Now consider a vertex $w \in \beta$, and let U_w^α be the K contiguous adjacent predecessors of w in α. All vertices U_w^α appearing in α for the first time are also adjacent predecessors of w in β, by construction of β. Let $v_i \in U_w^\alpha$ such that there is $j < i$ with $v_j = v_i$, and let ℓ be the minimum such j. Then v_i does not appear in β; however, v_ℓ does. Also, since $v_\ell = v_i$ and $v_i \in U_w^\alpha$, v_ℓ is an adjacent predecessor of w in β. Thus β is a DDGP order for G, as claimed. □

It turns out that the family of graphs $\mathcal{S}(K, n)$ can also be used to separate DDGP from ReOP. For brevity, we only prove this for $\mathcal{S}(3, 7)$, shown in Fig. 2, by making use of the following lemma.

4.3 Lemma
If $U = \{u_1, \ldots, u_K\}$ is a clique of size K in a graph G, no minimally-sized re-order can have a subsequence listing all of U followed by a vertex u_i for some $i \leq K$.

Proof. If a minimally-sized re-order α had a subsequence $\alpha' = (u_1, \ldots, u_K, u_1)$ (without loss of generality) followed by a vertex v, this would mean that v is adjacent to all vertices in U, which implies that the re-order obtained by replacing α' in α by $(u_1, \ldots, u_K, v)$ is shorter than α, against minimality. □

We now attempt to build a minimally-sized re-order $(v_1, \ldots, v_m)$ for $\mathcal{S}(K, n)$. We either start from the initial clique $\{1, 2, 3\}$ or from one of the other vertices $\{4, 5, 6, 7\}$. Let us first look at the case where $v_1 = 1, v_2 = 2, v_3 = 3$: by Lemma 4.3, $v_4 \in \{4, 5, 6, 7\}$, say $v_4 = 4$ without loss of generality, which forces $v_5 = 2$. We know $v_6 \notin \{5, 6, 7\}$ because $v_4 = 4$ is not adjacent to any of these vertices, so $v_6 = 3$; for the same reason, $v_7 \notin \{5, 6, 7\}$, which forces $v_7 = 1$. Again by Lemma 4.3, $v_8 \in \{5, 6, 7\}$, say $v_8 = 5$;

this implies $v_9 = 3$, which forces $v_{10} = 1$. Now the only possible choice for v_{11} is v_8, but this would contradict Lemma 4.3. So now consider the case where $v_1 = 4$ and $v_2, v_3 \in \{1, 2, 3\}$, say $v_2 = 2, v_3 = 3$. Now $v_4 \in \{1, 4\}$, but Lemma 4.3 forces $v_4 = 1$, which implies $v_5 \in \{5, 6, 7\}$, say $v_5 = 5$. Now v_6 can only be 3, which forces $v_7 = 1$. Finally, $v_8 \notin \{6, 7\}$ because this would imply a non-existent edge $\{5, 6\}$ or $\{5, 7\}$ in the graph, $v_8 \notin \{1, 2, 3, 4, 8\}$ because repetitions are only allowed for pairs of vertices having order rank difference exactly K, so v_8 can only be 5, but this goes against Lemma 4.3. Thus $\mathcal{S}(3, 7)$ is a DDGP YES instance which does not have a re-order, as claimed.

5 Finding discretization orders in practice

The authors' main interest is in solving DGPs related to protein conformation [15] using the very efficient BP algorithm. In this setting, K is fixed to the constant 3. It was shown in [13] that the TOP can be solved in polynomial time by a greedy algorithm when K is fixed, and we do routinely solve fairly large TOP instances in practice [25], so the issue may appear to be closed.

Since the CTOP has nice mathematical properties, however, it would be more convenient to find ^{K}DMDGP orders than DDGP ones. We have shown above that ^{K}DMDGP orders are hard to find even for fixed K, and for a given initial clique. To circumvent this issue, we argued in [16] that certain re-orders which are hand-crafted for protein backbones could be profitably used instead. The fact that these re-orders are constructed by humans rather than automatically found by computers is a feature we exploit to accommodate many other biochemical requirements to do with proteins, so at this stage we have no motivation to solving ReOP algorithmically (though some efforts in this sense are foreshadowed in [24]). We do, however, have an interest in finding ^{K}DMDGP orders automatically, since these could be of use in other applications of the DGP.

In this section we propose and discuss three MILP formulations for the CTOP: the *vertex-rank* formulation, the *clique digraph* formulation, and a relaxation of the latter. The first one models the problem by assigning a unique rank to every vertex, and the second looks for a certain path in an auxiliary digraph whose nodes are cliques of size $(K + 1)$ in the input graph.

5.1 The vertex-rank formulation for CTOP

This is a "natural" formulation where binary decision variables x_{vi} decide whether vertex v should be the i-th in the order or not. Given a graph $G = (V, E)$ with $|V| = n$ and a positive integer K, the Integer Linear Programming (ILP) formulation below findsx a ^{K}DMDGP order in G if it exists, and is infeasible otherwise. For any $v \in V$ and $i \in \bar{n} = \{1, \ldots, n\}$, let x_{vi} be a binary variable, which will take value 1 if v is the i-th vertex in the order, or 0 otherwise. This is a pure feasibility problem:

1. each vertex has a unique rank:
$$\forall v \in V \quad \sum_{i \in \bar{n}} x_{vi} = 1;$$

2. each rank value is assigned a unique vertex:
$$\forall i \in \bar{n} \quad \sum_{v \in V} x_{vi} = 1;$$

3. there must be an initial clique of size K:
$$\forall v \in V, i \in \{2, \ldots, K\} \quad \sum_{u \in N(v)} \sum_{j < i} x_{uj} \geq (i - 1) x_{vi};$$

4. each vertex with rank $> K$ must have at least K contiguous predecessors
$$\forall v \in V, i > K \quad \sum_{u \in N(v)} \sum_{i-K \le j < i} x_{uj} \ge K x_{vi}.$$

5.2 The clique digraph formulation for CTOP

The following formulation is suggested by the complexity reduction used in Sect. 3.2 from the Hamiltonian Path problem: a ^{K}DMDGP order is a sequence of embedded cliques of size $(K+1)$, with successive pairs sharing K vertices and covering the whole of V. Consider an example with $K = 3$: if v's adjacent predecessors are $v-1, v-2, v-3$ then $(v+1)$'s are $v, v-1, v-2$. This means that both $\{v-3, v-2, v-1, v\}$ and $\{v-2, v-1, v, v+1\}$ must induce cliques of size 4 in the input graph $G = (V, E)$.

This suggests the use of a clique digraph $\mathscr{C} = (N, A)$ where N is the set of ordered cliques of G of size $(K+1)$ and $(\alpha, \beta) \in A$ if and only if $\alpha = \{u_1, \ldots, u_{K+1}\}$ and $\beta = \{u_2, \ldots, u_{K+2}\}$, for some $u_1, \ldots, u_{K+2} \in V$. For each $\nu \in N$ let $L(\nu)$ be the last vertex in the ordered clique ν of size $K+1$. A ^{K}DMDGP order in this setting is a path $P = (\nu_1, \ldots, \nu_{n-K})$ in $\mathscr{C}$, and $\nu_1 \cup \{L(\nu_i) \mid i > 1\} = V$, i.e. the last vertices of the cliques in P, together with ν_1, cover V. Notationwise, we stipulate that, since the cliques are ordered, ν_{jk} is the k-th vertex of the j-th clique.

Consider the following decision variables:

1. for $(i, j) \in A$, $x_{ij} \in \{0, 1\}$ takes value 1 iff (i, j) is an arc in P;
2. for $j \in N$, $\phi_j \in \{0, 1\}$ takes value 1 iff ν_j is the initial clique;
3. for $j \in N$, $\lambda_j \in \{0, 1\}$ takes value 1 iff ν_j is the last clique;
4. for $u, v \in V$, $y_{uv} \in \{0, 1\}$ takes value 1 iff u is a predecessor of v in the ^{K}DMDGP order.

The constraints of the problems are:

1. P has exactly one first clique and one last clique:
$$\begin{aligned} \sum_{j \in N} \phi_j &= 1 \\ \sum_{j \in N} \lambda_j &= 1 \end{aligned}$$
2. since P is a path, flow balance equations hold at each node aside from first and last:
$$\forall i \in N \quad \phi_i + \sum_{\substack{j \in N \\ (j,i) \in A}} x_{ji} = \lambda_i + \sum_{\substack{j \in N \\ (i,j) \in A}} x_{ij};$$
3. each clique in N has at most one successor (the fact that each clique also has at most one predecessor follows by the previous constraints):
$$\forall i \in N \quad \sum_{\substack{j \in N \\ (i,j) \in A}} x_{ij} \le 1;$$
4. the cliques in P cover the whole of V:
$$\forall v \in V \quad \sum_{\substack{j \in N \\ v \in \nu_j}} \phi_j + \sum_{\substack{(i,j) \in A \\ \nu_j \smallsetminus \nu_i = \{v\}}} x_{ij} = 1;$$

5. for each $u, v \in V$, either $u < v$ or $v < u$ but not both:
$$\forall u, v \in V \quad u \neq v \rightarrow y_{uv} + y_{vu} = 1;$$

6. linear ordering on vertex triplets:
$$\forall u, v, w \in V \quad u \neq v \wedge v \neq w \wedge w \neq u \rightarrow y_{uv} + y_{vw} + y_{wu} \leq 2;$$

7. enforce the fact that the cliques are ordered:
$$\begin{aligned} \forall (i,j) \in A, k \leq K, h = \nu_{ik}, \ell = \nu_{i,k+1} \quad w_{h\ell} &\geq x_{ij} & (2) \\ \forall (i,j) \in A, v \in \nu_j \smallsetminus \nu_i \ell = \nu_{i,k+1} \quad |\nu_j \smallsetminus \nu_i| = 1 \rightarrow w_{\ell v} &\geq x_{ij} & (3) \\ \forall j \in N, k \leq K, h = \nu_{jk}, \ell = \nu_{j,k+1} \quad w_{h\ell} &\geq \phi_j + \lambda_j. & (4) \end{aligned}$$

Since these constraints allow for the possibility that some more arcs in A might be selected besides the arcs on the path P, we impose the following objective function:
$$\min \sum_{(i,j) \in A} x_{ij}.$$

5.2.1 The unordered clique relaxation

One of the disadvantages of the clique digraph formulation is the size of N: in the worst case, it could have as many as $\binom{|V|}{K+1}(K+1)!$ nodes — for reasonably sparse graph the first term of the product, which denotes the number of cliques of size $K+1$ in the digraph is obviously smaller. By considering unordered cliques, however, we can reduce the worst-case size of N to the smaller term $\binom{|V|}{K+1}$. The number of arcs remains the same.

The unordered clique relaxation can be described as a sequence of changes to the clique digraph formulation of Sect. 5.2. First, we change the constraints in item 7 above, specifically (2)-4, to:
$$\forall (i,j) \in A, u \in \nu_i, v \in \nu_j \smallsetminus \nu_i \quad |\nu_j \smallsetminus \nu_i| = 1 \rightarrow w_{uv} \geq x_{ij}. \qquad (5)$$
Next, for each $j \in N$ we add variables $z_j \in \{0, 1\}$ taking value 1 iff the *unordered* clique ν_j is used in P. These depend on the activity of the adjacent arcs:
$$\begin{aligned} \forall (u,j) \in A \quad z_j &\geq x_{ij} \\ \forall j \in N \quad z_j &\geq \phi_j. \end{aligned}$$
These new variables are designed to help state the sentence:

every vertex $v \in V$ except the first K and the last K should appear in exactly $K+1$ cliques.

Unfortunately, expressing this sentence would involve inequalities depending on z but also on the rank that a vertex v has in the order. While this is certainly possible (using e.g. variables as in the vertex-rank formulation of Sect. 5.1), it would make the formulation unwieldy, and ultimately inefficient to solve. We therefore resort to a relaxation, and simply state that each vertex should appear in at most $K+1$ cliques:
$$\forall v \in V \quad \sum_{\substack{j \in N \\ v \in \nu_j}} z_j \leq K+1.$$

Whenever this relaxation is feasible, the solution should be verified, as it might be an invalid ^{K}DMDGP order. On the other hand, if the relaxation is infeasible then certainly the given graph does not have any ^{K}DMDGP orders. The results in Table 4 suggest that a wrong YES is a rare event. Moreover, the relaxation approach is also justified by the fact that the clique digraph formulation is usually much faster at proving NO than finding a solution proving YES.

5.3 Computational evaluation

We tested the formulations given above by means of AMPL [8] and CPLEX 12.6 [12] on three sets of instances, for $K \in \{1, 2, 3\}$ (note that testing for $K = 1$ reduces to a computationally expensive way of testing graph connectedness).

The first instance set, called `minimal`, contains some minimal ^{K}DMDGP instance graphs with varying numbers of discretization edges and no pruning edges. The second instance set, called `protein`, contains six graphs connected with the application to protein conformation, provided by A. Mucherino. The third instance set, called `random`, contains a fairly large number of randomly generated biconnected graphs with varying number of vertices and edge generation probability in $\{0.3, 0.5, 0.7\}$.

The tests were carried out on a two-core Intel i7 CPU at 2.0GHz with 8GB RAM, with CPLEX running in parallel mode and the running time of both AMPL and CPLEX limited to 300s of user time. This somewhat short time limit was set with a view to vertex ordering problems only being pre-processing steps to the actual problem of finding a realization of the graph in $\mathbb{R}^K$. The user time reported by the system, and reproduced in the results tables, is an estimation of the CPU time which a single core would have taken to complete the task.

In general, the clique digraph formulation and its unordered relaxation are better at proving that an instance is NO, but very poor at finding solutions of YES instances. This is partly due to the fact that for sparse instances there may not even be sufficiently many cliques of size $K + 1$ in the graph, in which case the clique digraph cannot even be constructed.

The vertex-rank formulation is somehow complementary, in that it is better at finding solutions for YES instances, and poorer at proving NO, but it turns out to be the best overall. Both approaches fall decidedly short of the current practical needs for this problem, which are in the hundreds and even thousands of vertices. In this sense the vertex ordering problem is still wide open.

5.3.1 Small instances

We first present results of all formulations on a subset of small instances, namely `minimal`, `protein` and instances in `random` with up to 30 vertices. Table 1 reports the number of vertices and the number of edges in each instance.

Instance	$\vert V\vert$	$\vert E\vert$
dmdgp_2_05	5	7
dmdgp_2_10	10	17
dmdgp_2_15	15	27
dmdgp_2_20	20	37
dmdgp_3_10	10	24
dmdgp_3_15	15	39
dmdgp_3_20	20	54
nodvop	6	9
nodvop-minimal	6	10
pb-decremental_test_NO	19	54
pbackbone-nobeta	19	52
pbackbone-withbeta	24	65
pbgraph	24	53

Instance	$\vert V\vert$	$\vert E\vert$
random-10_0.3	10	26
random-10_0.5	10	27
random-10_0.7	10	37
random-15_0.3	15	35
random-15_0.5	15	67
random-15_0.7	15	73
random-20_0.3	20	66
random-20_0.5	20	109
random-20_0.7	20	140
random-25_0.3	25	102
random-25_0.5	25	161
random-25_0.7	25	226
random-30_0.3	30	148
random-30_0.5	30	228
random-30_0.7	30	318

Table 1: Statistics for the "small instances" set: `minimal` is top left, `protein` is bottom left, and `random` is on the right.

Table 2 presents test results obtained with the vertex rank formulation.

Table 3 presents test results obtained with the clique digraph formulation.

	$K = 1$		$K = 2$		$K = 3$	
Instance	*Y/N*	*CPU*	*Y/N*	*CPU*	*Y/N*	*CPU*
minimal						
dmdgp_2_05	Yes	0.01	Yes	0.00	No	0.01
dmdgp_2_10	Yes	0.00	Yes	0.00	No	0.01
dmdgp_2_15	Yes	0.00	Yes	0.00	No	0.02
dmdgp_2_20	Yes	0.00	Yes	0.01	No	0.02
dmdgp_3_10	Yes	0.00	Yes	0.00	Yes	0.00
dmdgp_3_15	Yes	0.00	Yes	0.00	Yes	0.01
dmdgp_3_20	Yes	0.01	Yes	0.01	Yes	0.01
protein						
nodvop-minimal	Yes	0.00	Yes	0.00	No	0.01
nodvop	Yes	0.00	Yes	0.00	No	0.01
pb-decremental_test_NO	Yes	0.01	No	39.13	No	60.13
pbackbone-nobeta	Yes	0.01	No	101.96	No	50.12
pbackbone-withbeta	Yes	0.86	No	114.98	Limit	
pbgraph	Yes	31.81	No	20.24	No	0.03
random						
random-10_0.3	Yes	0.00	Yes	0.02	No	0.01
random-10_0.5	Yes	0.01	Yes	0.14	No	0.37
random-10_0.7	Yes	0.00	Yes	0.00	Yes	0.02
random-15_0.3	Yes	0.00	No	1.42	No	0.02
random-15_0.5	Yes	0.01	Yes	0.04	Yes	37.09
random-15_0.7	Yes	0.01	Yes	0.02	Yes	1.17
random-20_0.3	Yes	0.01	No	43.47	No	179.77
random-20_0.5	Yes	0.01	Yes	1.54	Limit	
random-20_0.7	Yes	0.01	Yes	0.10	Yes	1.12
random-25_0.3	Yes	0.01	Limit		Limit	
random-25_0.5	Yes	0.01	Yes	2.58	Limit	
random-25_0.7	Yes	0.02	Yes	0.33	Yes	1.86
random-30_0.3	Yes	0.01	Limit		Limit	
random-30_0.5	Yes	0.02	Yes	8.92	Limit	
random-30_0.7	Yes	0.02	Yes	0.39	Yes	9.83

Table 2: Results for the vertex rank formulation on the small instances; Limit indicates that the 300s time limit was exceeded.

	$K = 1$		$K = 2$		$K = 3$	
Instance	*Y/N*	*CPU*	*Y/N*	*CPU*	*Y/N*	*CPU*
minimal						
dmdgp_2_05	Yes	0.01	Yes	0.01	No	0.00
dmdgp_2_10	Yes	0.02	Yes	0.02	No	0.00
dmdgp_2_15	Yes	0.05	Yes	0.07	No	0.00
dmdgp_2_20	Yes	0.08	Yes	0.21	No	0.00
dmdgp_3_10	Yes	0.03	Yes	0.09	Yes	0.03
dmdgp_3_15	Yes	0.06	Yes	0.21	Yes	0.10
dmdgp_3_20	Yes	0.13	Yes	0.52	Yes	0.25
protein						
nodvop-minimal	Yes	0.01	Yes	0.01	No	0.00
nodvop	Yes	0.01	Yes	0.01	No	0.00
pb-decremental_test_NO	Yes	0.09	No	0.27	No	0.02
pbackbone-nobeta	Yes	0.09	No	0.41	No	0.08
pbackbone-withbeta	Yes	2.43	No	0.29	No	0.04
pbgraph	Yes	0.23	No	0.14	No	0.03
random						
random-10_0.3	Yes	0.04	Yes	0.12	No	0.01
random-10_0.5	Yes	0.03	Yes	0.07	No	0.01
random-10_0.7	Yes	0.06	Yes	0.30	Yes	20.37
random-15_0.3	Yes	0.06	No	0.01	No	0.00
random-15_0.5	Yes	0.10	Yes	20.63	Yes	9.52
random-15_0.7	Yes	0.17	Yes	26.35	Limit	
random-20_0.3	Yes	0.17	No	0.04	No	0.02
random-20_0.5	Yes	0.26	Yes	159.84	No	210.98
random-20_0.7	Yes	0.33	Yes		Limit	
random-25_0.3	Yes	0.29	No	5.19	No	0.03
random-25_0.5	Yes	0.43	Yes	602.21	Limit	
random-25_0.7	Yes	0.71	Limit		Limit	
random-30_0.3	Yes	0.56	Limit		No	0.05
random-30_0.5	Yes	0.82	Limit		Limit	
random-30_0.7	Yes	1.17	Limit		Limit	

Table 3: Results for the clique digraph formulation on the small instances; Limit indicates that the 300s time limit was exceeded.

Table 4 presents test results obtained with the unordered clique relaxation.

	K = 1		*K* = 2		*K* = 3	
Instance	*Y/N*	*CPU*	*Y/N*	*CPU*	*Y/N*	*CPU*
			minimal			
dmdgp_2_05	Yes	0.01	Fail		No	0.00
dmdgp_2_10	Yes	0.02	Yes	0.01	No	0.00
dmdgp_2_15	Yes	0.04	Yes	0.05	No	0.00
dmdgp_2_20	Yes	0.09	Yes	0.17	No	0.00
dmdgp_3_10	Yes	0.30	Yes	0.03	Yes	0.01
dmdgp_3_15	Yes	0.06	Yes	0.10	Yes	0.06
dmdgp_3_20	Yes	0.15	Yes	0.20	Yes	0.16
			protein			
nodvop-minimal	Yes	0.01	Yes	0.01	No	0.00
nodvop	Yes	0.01	Yes	0.01	No	0.00
pb-decremental_test_NO	Yes	5.33	No	17.98	No	0.00
pbackbone-nobeta	Yes	3.89	No	38.11	No	0.04
pbackbone-withbeta	Yes	1.13	No	7.64	No	0.00
pbgraph	Yes	7.83	No	0.11	No	0.00
			random			
random-10_0.3	Yes	0.52	Yes	0.04	No	0.00
random-10_0.5	Yes	0.06	Yes	0.44	No	0.00
random-10_0.7	Yes	0.40	Yes	0.17	Yes	1.97
random-15_0.3	Yes	1.28	No	0.00	No	0.00
random-15_0.5	Yes	3.54	Yes	10.50	Yes	20.05
random-15_0.7	Yes	5.03	Yes	24.52	Yes	47.87
random-20_0.3	Yes	13.10	No	0.07	No	0.00
random-20_0.5	Yes	36.18	Yes	127.04	Limit	
random-20_0.7	Yes	58.82	Yes	17.43	Limit	
random-25_0.3	Yes	79.91	No	545.11	No	0.00
random-25_0.5	Yes	101.74	Yes	395.88	Limit	
random-25_0.7	Yes	295.01	Limit		Limit	
random-30_0.3	Yes	347.66	Limit		No	0.00
random-30_0.5	Limit		Limit		Limit	
random-30_0.7	Limit		Limit		Limit	

Table 4: Results for the unordered clique relaxation on the small instances. Limit indicates that the 300s time limit was exceeded; Fail indicates a YES answer not corresponding to a valid order.

Since it is clear that the vertex rank formulation is the best overall, we focus the comparison over those instances where it takes more than 10s of CPU time; results are shown in Table 5.

				CPU	
Instance	*K*	*Y/N*	*v-r*	*c-d*	*u-c*
		protein			
pb-decremental_test_NO	2	No	39.13	**0.27**	17.98
pb-decremental_test_NO	3	No	60.13	0.02	**0**
pbackbone-nobeta	2	No	101.96	**0.41**	38.11
pbackbone-nobeta	3	No	50.12	0.08	**0.04**
pbackbone-withbeta	2	No	114.98	**0.29**	7.64
pbackbone-withbeta	3	No	Limit	0.04	**0**
pbgraph	1	Yes	31.81	0.23	**0.11**
pbgraph	2	No	20.24	0.14	**0**
		random			
random-20_0.3	2	No	43.47	**0.04**	0.07
random-20_0.3	3	No	179.77	0.02	**0**
random-20_0.5	3	No	Limit	**210.98**	Limit
random-25_0.3	2	No	Limit	**5.19**	545.11
random-25_0.3	3	No	Limit	0.03	**0**
random-25_0.5	3	?	Limit	Limit	Limit
random-30_0.3	2	?	Limit	Limit	Limit
random-30_0.3	3	No	Limit	0.05	**0**
random-30_0.5	3	?	Limit	Limit	Limit

Table 5: Comparative results over small instances where the vertex rank formulation takes more than 10s of user CPU time. In the three leftmost columns, *v-r* stands for vertex rank formulation, *c-d* for clique digraph formulation, and *u-c* for unordered clique relaxation. Best performances are in boldface.

5.3.2 Medium-sized random instances

We also tested the best formulation, i.e. the vertex rank formulation, on moderately larger instances, with $|V| \in \{35, 40, 45, \ldots, 95\}$ and edge sparsity ratios given in the instance names (see the results in Table 6. The same pattern emerged as for smaller instances, namely the vertex rank formulation is better at finding the order in YES instances than proving an instance NO. Indeed, this formulation was unable to prove any instance NO for this larger test set.

	$K=1$		$K=2$		$K=3$	
Instance	*Y/N*	*CPU*	*Y/N*	*CPU*	*Y/N*	*CPU*
`random-35_0.3`	Yes	0.01	Limit		Limit	
`random-35_0.5`	Yes	0.01	Yes	3.49	Limit	
`random-35_0.7`	Yes	0.01	Yes	0.37	Yes	10.01
`random-40_0.3`	Yes	0.01	Limit		Limit	
`random-40_0.5`	Yes	0.02	Yes	33.46	Limit	
`random-40_0.7`	Yes	0.02	Yes	0.66	Yes	12.07
`random-45_0.3`	Yes	0.02	Limit		Limit	
`random-45_0.5`	Yes	0.02	Yes	82.17	Limit	
`random-45_0.7`	Yes	0.03	Yes	1.35	Yes	23.71
`random-50_0.3`	Yes	0.02	Limit		Limit	
`random-50_0.5`	Yes	0.03	Yes	125.51	Limit	
`random-50_0.7`	Yes	0.04	Yes	2.03	Yes	80.77
`random-55_0.3`	Yes	0.03	Limit		Limit	
`random-55_0.5`	Yes	0.04	Yes		Limit	
`random-55_0.7`	Yes	0.04	Yes	3.76	Yes	58.81
`random-60_0.3`	Yes	0.03	Limit		Limit	
`random-60_0.5`	Yes	0.04	Yes	43.43	Limit	
`random-60_0.7`	Yes	0.05	Yes	5.05	Yes	63.45
`random-65_0.3`	Yes	0.04	Limit		Limit	
`random-65_0.5`	Yes	0.05	Limit		Limit	
`random-65_0.7`	Yes	0.06	Yes	6.71	Limit	
`random-70_0.3`	Yes	0.04	Limit		Limit	
`random-70_0.5`	Yes	0.05	Limit		Limit	
`random-70_0.7`	Yes	0.07	Yes	9.31	Yes	112.48
`random-75_0.3`	Yes	0.05	Limit		Limit	
`random-75_0.5`	Yes	0.07	Limit		Limit	
`random-75_0.7`	Yes	0.08	Yes	11.92	Limit	
`random-80_0.3`	Yes	0.06	Limit		Limit	
`random-80_0.5`	Yes	0.08	Limit		Limit	
`random-80_0.7`	Yes	0.11	Yes	17.32	Limit	
`random-85_0.3`	Yes	0.07	Limit		Limit	
`random-85_0.5`	Yes	0.11	Limit		Limit	
`random-85_0.7`	Yes	0.12	Yes	23.69	Limit	
`random-90_0.3`	Yes	0.09	Limit		Limit	
`random-90_0.5`	Yes	0.12	Limit		Limit	
`random-90_0.7`	Yes	0.13	Yes	28.52	Limit	
`random-95_0.3`	Yes	0.09	Limit		Limit	
`random-95_0.5`	Yes	0.12	Limit		Limit	
`random-95_0.7`	Yes	0.16	Yes	32.63	Limit	

Table 6: The vertex rank formulation tested on larger instances.

By contrast, some of the larger sparse instances could be proven to be NO for $K = 3$ by the clique-based formulations within 300 seconds of user CPU time, as shown in Table 7. It is worth mentioning that none of the other larger instances could be solved by either clique formulation.

			CPU		
Instance	*K*	*Y/N*	*v-r*	*c-d*	*u-c*
			`protein`		
`random-35_0.3`	3	No	Limit	0.09	**0.07**
`random-40_0.3`	3	No	Limit	0.13	**0.1**
`random-45_0.3`	3	No	Limit	**61.54**	875.27
`random-50_0.3`	3	No	Limit	**129.24**	Limit
`random-55_0.3`	3	No	Limit	**238.86**	Limit
`random-60_0.3`	3	No	Limit	Limit	Limit

Table 7: Proving larger instances NO by means of the cliques formulations. See the caption to Table 5 for a *legenda* of the columns.

6 Conclusion

The significance of discretization vertex orders concerns the possibility of using the BP algorithm for finding all incongruent solutions to certain DGP instances (e.g., related to protein conformation). In this paper we considered three such orders: ^{K}DMDGP orders, re-orders and DDGP orders. We proved the **NP**-completeness of their existence problems (respectively CTOP, ReOP and TOP), and showed specifically that, unlike the TOP, the CTOP and the ReOP problems are also **NP**-complete for any fixed K. We then discussed the inclusionwise relation of these problems, establishing that: CTOP $\subsetneq$ ReOP $\subsetneq$ TOP. Lastly, we proposed three MILP formulations for the CTOP.

Acknowledgments

We are grateful to Prof. Maxim Sviridenko for useful discussions. This work was partially supported by the Brazilian research agencies FAPESP, CNPq, CAPES, and by the French research agency ANR (project n. ANR-10-BINF-03-08 "Bip:Bip").

References

[1] A.V. Aho, J.E. Hopcroft, and J.D. Ullman. *Data Structures and Algorithms.* Addison-Wesley, Reading, MA, 1983.

[2] T. Biedl, T. Chan, Y. Ganjali, M. Taghi Hajiaghayi, and D. Wood. Balanced vertex-orderings of graphs. *Discrete Applied Mathematics*, 148:27–48, 2005.

[3] H. Bodlaender, F. Fomin, A. Koster, D. Kratsch, and D. Thilikos. A note on exact algorithms for vertex ordering problems on graphs. *Theory of Computing Systems*, 50:420–432, 2012.

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

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

[6] Q. Dong and Z. Wu. A geometric build-up algorithm for solving the molecular distance geometry problem with sparse distance data. *Journal of Global Optimization*, 26:321–333, 2003.

[7] T. Eren, D. Goldenberg, W. Whiteley, Y. Yang, A. Morse, B. Anderson, and P. Belhumeur. Rigidity, computation, and randomization in network localization. *IEEE Infocom Proceedings*, pages 2673–2684, 2004.

[8] R. Fourer and D. Gay. *The AMPL Book.* Duxbury Press, Pacific Grove, 2002.

[9] D. Gillies and J. Liu. Scheduling tasks with AND/OR precedence constraints. *SIAM Journal on Computing*, 24(4):797–810, 1995.

[10] J. Graver, B. Servatius, and H. Servatius. *Combinatorial Rigidity.* American Mathematical Society, 1993.

[11] L. Henneberg. *Die Graphische Statik der starren Systeme.* Teubner, Leipzig, 1911.

[12] IBM. *ILOG CPLEX 12.6 User's Manual.* IBM, 2014.

[13] C. Lavor, J. Lee, A. Lee-St. John, L. Liberti, A. Mucherino, and M. Sviridenko. Discretization orders for distance geometry problems. *Optimization Letters*, 6:783–796, 2012.

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

[15] C. Lavor, L. Liberti, N. Maculan, and A. Mucherino. Recent advances on the discretizable molecular distance geometry problem. *European Journal of Operational Research*, 219:698–706, 2012.

[16] 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*, 56:855–871, 2013.

[17] L. Liberti, C. Lavor, J. Alencar, and G. Abud. Counting the number of solutions of ^{k}DMDGP instances. In Nielsen and Barbaresco [29].

[18] 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.

[19] L. Liberti, C. Lavor, N. Maculan, and A. Mucherino. Euclidean distance geometry and applications. *SIAM Review*, 56(1):3–69, 2014.

[20] L. Liberti, C. Lavor, and A. Mucherino. The discretizable molecular distance geometry problem seems easier on proteins. In Mucherino et al. [27].

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

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

[23] A. Mucherino. On the identification of discretization orders for distance geometry with intervals. In Nielsen and Barbaresco [29].

[24] A. Mucherino. Molecular distance geometry and atomic orders. In *Proceedings of the IFORS Conference*, page 244, Barcelona, 2014. IFORS.

[25] A. Mucherino, C. Lavor, and L. Liberti. The discretizable distance geometry problem. *Optimization Letters*, 6:1671–1686, 2012.

[26] A. Mucherino, C. Lavor, and L. Liberti. Exploiting symmetry properties of the discretizable molecular distance geometry problem. *Journal of Bioinformatics and Computational Biology*, 10:1242009(1–15), 2012.

[27] A. Mucherino, C. Lavor, L. Liberti, and N. Maculan, editors. *Distance Geometry: Theory, Methods, and Applications.* Springer, New York, 2013.

[28] C. Mueller, B. Martin, and A. Lumsdaine. A comparison of vertex ordering algorithms for large graph visualization. In S.-H. Hong and K.-L. Ma, editors, *6th International Asia-Pacific Symposium on Visualization*, Sidney, 2007. IEEE.

[29] F. Nielsen and F. Barbaresco, editors. *Geometric Science of Information*, volume 8085 of *LNCS*, New York, 2013. Springer.

[30] R. Santana, P. Larra naga, and J.A. Lozano. Combining variable neighbourhood search and estimation of distribution algorithms in the protein side chain placement problem. *Journal of Heuristics*, 14:519–547, 2008.

[31] T. Schlick. *Molecular modelling and simulation: an interdisciplinary guide.* Springer, New York, 2002.

[32] K. Wüthrich, M. Billeter, and W. Braun. Pseudo-structures for the 20 common amino acids for use in studies of protein conformations by measurements of intramolecular proton-proton distance constraints with nuclear magnetic resonance. *Journal of Molecular Biology*, 169:949–961, 1983.