A Probabilistic Approach in the Search Space of the Molecular Distance Geometry Problem

Rômulo S. Marques,[†] Michael Souza,[‡] Fernando Baptista,[‡] Miguel Gonçalves,[‡] and Carlile Lavor[*,†]

[†]*Instituto de Matemática, Estatística e Computação Científica, Universidade Estadual de Campinas, Campinas, Brazil, 13083-859;*

[‡]*Departamento de Estatística e Matemática Aplicada, Centro de Ciências, Universidade Federal do Ceará, Fortaleza, Brazil, 60020-181.*

E-mail: clavor@unicamp.br

Abstract

The discovery of the three-dimensional shape of protein molecules using interatomic distance information from Nuclear Magnetic Resonance (NMR) can be modeled as a Discretizable Distance Geometry Problem. Due to its combinatorial characteristics, the problem is conventionally solved in the literature as a depth-first search in a binary tree. In this work, we introduce a new search strategy, which we call Frequency-Based Search (FBS), that for the first time utilizes geometric information contained in the Protein Data Bank (PDB). We encode the geometric configurations of all 14,134 molecules derived from NMR experiments present in the PDB into binary strings. The results obtained show that the sample space of the binary strings extracted from the PDB does not follow a uniform distribution. Furthermore, we compare the costs of FBS with the depth-first search in finding a solution in terms of the number of nodes of the binary tree that are explored and ascertain that FBS performs better in at least 70% of the cases. Computational results, comparing the two approaches, indicate that ???

Introduction

The determination of the three-dimensional structure of protein molecules represents a profoundly complex challenge within structural biochemistry. This task necessitates a multidisciplinary strategy that melds mathematical modeling, judicious use of computational resources, and experimental data.[?][?] Among the models developed to tackle this challenge, the Discretizable Distance Geometry Problem (DDGP) utilizes Nuclear Magnetic Resonance (NMR) data to compute the Cartesian coordinates of atoms within the molecule based on distance measurements provided by NMR techniques.[?]

In the ideal scenario where the distances between all pairs of atoms in a protein are known, the DDGP can be solved efficiently in linear time.[?] However, NMR experiments typically provide only partial and approximate distance data, represented as lower and upper limits, rather than precise values.[?] A common assumption in the literature is to fix bond lengths and bond angles of the protein molecule whose 3D structure we want to determine,[?] which despite simplifying the problem, allows us to use a mathematical model (the DDGP) that exploits the important combinatorial features of the problem, not considered in the continuous approach (see the subsequent section for further discussion).

The Branch-and-Prune (BP) method[?] has been widely adopted for its effectiveness in solving the DDGP. Based on the depth-first search (DFS) algorithm,[?] BP systematically progresses through the search tree of the DDGP. It prunes the solution space according to the satisfaction of distance constraints provided as inputs of the problem. While DFS is known for its low memory footprint, it does not incorporate biochemical information about proteins. In this paper, for the first time, we leverage data from the Protein Data Bank (PDB)[?] to propose an alternative search strategy to DFS.

The DDGP Search Tree

The DDGP is a specific subclass of the Molecular Distance Geometry Problem (MDGP),[?] defined as follows (for notational simplicity, we just write DDGP to represent the DDGP$_k$, for $k = 3$).

Given a weighted undirected graph $G = (V, E, d)$, where V represents the set of atoms in the molecule and E is the set of atom pairs with known distances, given by $d : E \to \mathbb{R}^+$, solving the MDGP involves finding a function $x : V \to \mathbb{R}^3$ such that

$$\|x(v_i) - x(v_j)\|_2 = d(v_i, v_j), \quad \forall \{v_i, v_j\} \in E.$$

The DDGP is an MDGP with a particular ordering of the vertices of G. Initially, a subset of V consisting of three vertices that form a clique (a fully connected subgraph) must be identified. For each vertex $v \in V$ outside this initial subset, there should be three vertices that precede v in the established order, connected to v (i.e., with known distances to v). Hence, the conditions for the ordering $v_1, \ldots, v_n \in V$ are as follows:

H$_1$: *The first three vertices are connected, i.e.,* $\{v_1, v_2\}, \{v_1, v_3\}, \{v_2, v_3\} \in E$.

H$_2$: *For any vertex v_i with $i > 3$, there are three other vertices v_j, v_k, v_l (referred to as reference vertices for v_i), with $j, k, l < i$, such that* $\{v_j, v_i\}, \{v_k, v_i\}, \{v_l, v_i\} \in E$.

A simplified notation is used where the vertices v_i are represented by their indices i only. Thus, x_i represents the coordinates of the vertex v_i and d_{ij} represents the distance between vertices v_i and v_j.

To eliminate solutions obtained merely by the rotations and translations of a given solution, the positions of the first three atoms can be fixed as follows:

$$x_1 = (0, 0, 0), \quad x_2 = (d_{1,2}, 0, 0), \quad x_3 = (d_{1,3}\cos(\theta_{2,3}), d_{1,3}\sin(\theta_{2,3}), 0),$$

forming the vertices of a triangle with sides $d_{1,2}, d_{1,3}, d_{2,3}$.

In an iterative construction approach to solving the DDGP, we begin by fixing the first three vertices. Subsequently, under Hypothesis H$_2$, with the reference vertices $\{j, k, l\}$ for the vertex $i > 3$ fixed, we need to find the solution to the following system:

$$\begin{aligned} \|x_i - x_j\|_2 &= d_{ij}, \\ \|x_i - x_k\|_2 &= d_{ik}, \\ \|x_i - x_l\|_2 &= d_{il}. \end{aligned} \tag{1}$$

Each of these constraints defines a sphere centered at one of the reference vertices, with a radius equal to the distance between this center and point x_i. Therefore, x_i must lie within the intersection of these three spheres. Assuming a solution exists for the DDGP and that points x_j, x_k, and x_l are not collinear, the intersection of these three spheres will consist of at most two points (see Figure **??**).

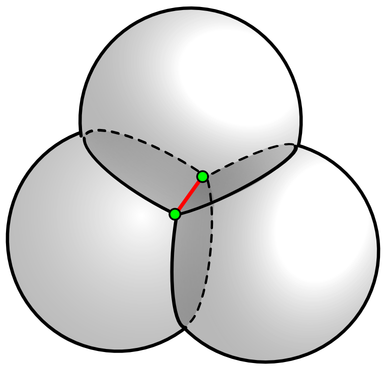


Figure 1: Two points (in green) in the intersection of three spheres.

Through this constructive procedure, once atoms $j < i$ are fixed, atom i will have two possible positions, x_i or x_i' in $\mathbb{R}^3$, obtained from the system (**??**). Once x_i is chosen from the two possibilities, we can continue the process by fixing point x_{i+1}.

A natural representation for all possible configurations $x = (x_1, x_2, \ldots, x_n) \in \mathbb{R}^{3\times n}$ is a binary tree, where x_1, x_2, x_3 are represented as a single root node (since they are fixed) and their two children represent the two possibilities for x_4, and for each of these, the two possibilities for x_5 and so on (see Figure **??**).

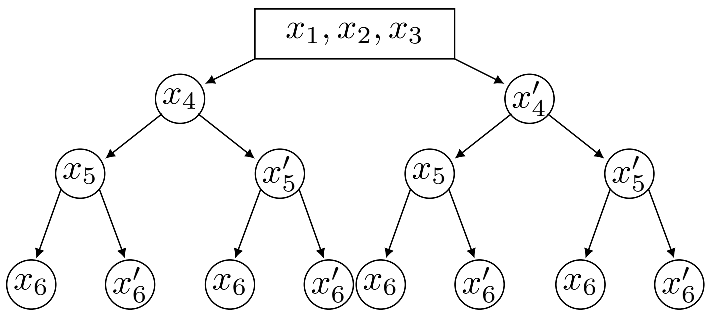


Figure 2: Binary tree associated with a DDGP instance with six atoms.

If we designate the left node as child 0 and the right one as child 1, we can also map the relative position of each realization of x_i with respect to the plane π_i defined by x_j, x_k, x_l, where j, k, l are the reference atoms of the i-th atom. Specifically, we can define the vectors

$$\vec{u} = x_j - x_l, \quad \vec{v} = x_k - x_l, \quad \vec{w} = x_i - x_l,$$

and assign orientation 0 if $\vec{w} \cdot (\vec{u} \times \vec{v}) < 0$, and 1 otherwise (see Figure **??**).

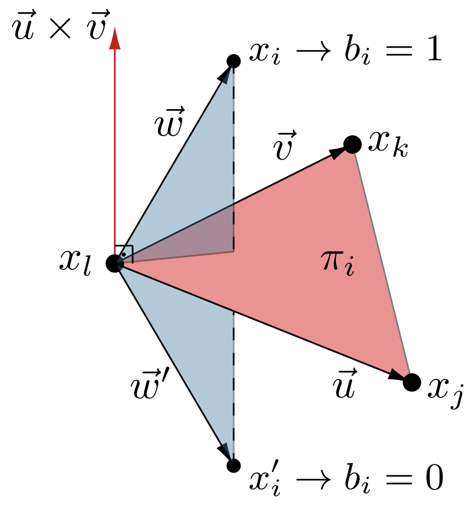


Figure 3: Plane π_i and the positions x_i and x'_i associated with orientations 0 and 1. Plane π_i is colored in light red. The perpendicular vector $\vec{u} \times \vec{v}$ is colored in red. The possible immersion points (in $\mathbb{R}^3$) for the vertex v_i are x_i and x'_i: the point "above" the plane π_i is x_i and has orientation $b_i = 1$; the point "below" π_i is x'_i and has orientation $b_i = 0$.

The search for solutions within the DDGP binary tree can be attempted using brute force. However, this approach becomes impractical for molecules with hundreds of atoms as the size of the tree grows exponentially with the number of atoms.

When additional distances, namely d_{ij} where j is not one of the reference atoms of atom i, are known, the viable configurations are reduced by the presence of the additional constraint $\|x_i - x_j\|_2 = d_{ij}$. These additional constraints are referred to as *pruning constraints*, as they render branches in the DDGP binary tree infeasible.

The BP algorithm was developed to solve the problem by intelligently traversing the DDGP search tree. Using the pruning constraints, it prunes branches that are identified as unfeasible, thus eliminating the need to explore the entire tree.

In its search for solutions to the DDGP, the BP algorithm explores paths in the associated binary tree using a DFS strategy that arbitrarily favors the 0 nodes of the binary tree. For example, in a binary tree of length two, the explored paths would be 00, 01, 10, 11.

DFS is a fundamental algorithm used in tree traversal,[?] characterized by exploring a branch as deeply as possible before backtracking to explore other branches. While it always finds solutions in finite trees (size dependent), DFS is notable for its memory efficiency, as it only needs to store a stack of nodes on the current path from the root node. However, it is important to note that DFS does not guarantee the shortest path to the solution.

In contrast to DFS, we propose a Best-First Search strategy, called *Frequency-Based Search* (FBS), an algorithm that traverses the tree selecting which path to follow based on an evaluation function that estimates which nodes are most likely to lead to a solution.

An FBS Approach Defined by the PDB

The Protein Data Bank (PDB) is a crucial resource for scientific advancement, containing over one terabyte of structural data for proteins, DNA, and RNA. The archive grows by nearly 10% per year and facilitates over 2 million structure data file downloads daily.[?] There are 184,318 protein-related entries in the PDB, of which 14,134 were obtained through NMR.[?]

To assemble our data repository, we selected all protein structures derived from NMR experiments present in the PDB, considering the relevance of such a technique to the scope of our research. From the first model of each selected PDB file, we extracted the following information for the backbone atoms of the protein in question: its unique index, its name (N, C_α, C, H, H_α), and its coordinates in $\mathbb{R}^3$, as well as the index and three-letter abbreviation of the residue it belongs to.

It is important to highlight that some PDB files do not completely describe the protein backbone. For example, there are files in which some residues are missing, and in others, some atoms are missing. We also chose to remove proline and glycine residues, as these amino acids exhibit unique geometric characteristics and are often studied separately in the literature.[?] [?] [?]

We refer to the stretches of the backbone formed by contiguous residues after the removal of prolines and glycines as *protein segments*. Thus, associated with each PDB file, we generated several files, one for each protein segment. The total number of protein segments obtained from all the NMR files in the PDB is 72,983.

An Example of a DDGP Order to Be Used in the FBS

To establish an FBS strategy to explore the DDGP search tree, inspired by the PDB data, one must first determine a DDGP order for the atoms in the backbone. We present a DDGP ordering that groups atoms in residues and utilizes the length of covalent bonds and bond angles, as well as the geometric properties of peptide planes.

In the context of protein geometry, it is considered that bond lengths and bond angles are fixed despite the natural internal motions of proteins. This assumption is known as the *rigid geometry hypothesis*.[?] Consequently, the distances between atoms connected by one or two covalent bonds are known. In addition to this distance information, it is well established in the biological literature that the atoms “around” a peptide bond belong to the same plane, implying that all distances between these atoms are also known.[?] Since the peptide bond connects the carboxyl carbon of the i-th residue (C^i) to the amine nitrogen of the (i+1)-th residue (N^{i+1}), the atoms in the i-th peptide plane are C^i_α, C^i, N^{i+1}, C^{i+1}_α (see Figure **??**). We also consider that the distances between H^i, H^i_α, and H^i_α, H^{i+1} can be detected by NMR.? ? ?

Based on these properties, we use the following DDGP order for atoms in the protein backbone:

$$\rho = (N^1, C^1_\alpha, H^1_\alpha, C^1, \ldots, H^i, N^i, C^i_\alpha, H^i_\alpha, C^i, \ldots, H^n, N^n, C^n_\alpha, H^n_\alpha, C^n). \tag{2}$$

Figure **??** illustrates the ordering ρ for a three-residue peptide. The numbers inside the circles indicate the index of the atoms in the ordering.

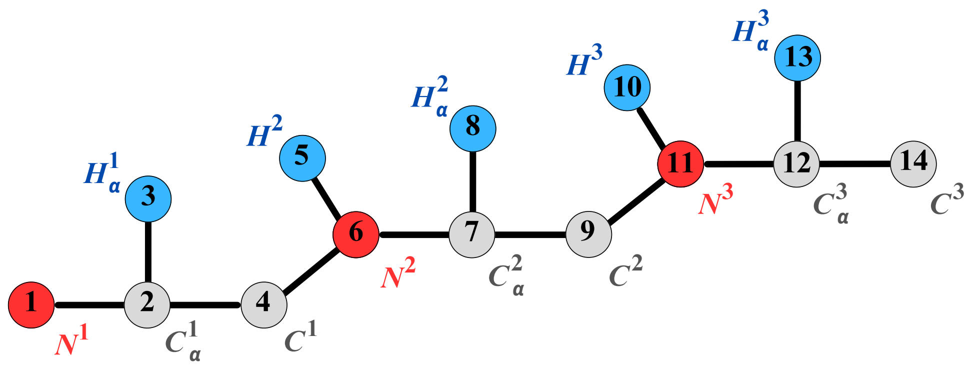


Figure 4: Order ρ of a 3-residue peptide backbone. The nitrogens are represented in red; the carbons in gray; and the hydrogens in blue. The number inside each circle represents the position of the respective atom in the order ρ.

Table **??** shows, for each element v in ρ, the three reference atoms of v, the coordinates and distances to v of which are used to immerse v in $\mathbb{R}^3$ (column "Predecessor Atoms"), as well as their positions in ρ (column "Predecessor Positions"). Note that all the residues consist of five atoms, except for the first, which is composed of four atoms.

Table 1: The DDGP order related to ρ. For each atom in ρ: first column displays its position; the second column displays its symbol; the third column displays the positions in ρ of its reference atoms; the fourth column displays the symbols of its reference atoms.

Order	**Atom**	**Predecessor Positions**			**Predecessor Atoms**		
1	N^1						
2	C_α^1						
3	H_α^1						
4	C^1	3	2	1	H_α^1	C_α^1	N^1
$\vdots$							
$5(i-1)$	H^i	$5(i-2)+4$	$5(i-2)+3$	$5(i-2)+2$	C^{i-1}	H_α^{i-1}	C_α^{i-1}
$5(i-1)+1$	N^i	$5(i-1)$	$5(i-2)+4$	$5(i-2)+2$	H^i	C^{i-1}	C_α^{i-1}
$5(i-1)+2$	C_α^i	$5(i-1)+1$	$5(i-1)$	$5(i-2)+4$	N^i	H^i	C^{i-1}
$5(i-1)+3$	H_α^i	$5(i-1)+2$	$5(i-1)+1$	$5(i-1)$	C_α^i	N^i	H^i
$5(i-1)+4$	C^i	$5(i-1)+3$	$5(i-1)+2$	$5(i-1)+1$	H_α^i	C_α^i	N^i
$\vdots$							
$5(n-1)$	H^n	$5(n-2)+4$	$5(n-2)+3$	$5(n-2)+2$	C^{n-1}	H_α^{n-1}	C_α^{n-1}
$5(n-1)+1$	N^n	$5(n-1)$	$5(n-2)+4$	$5(n-2)+2$	H^n	C^{n-1}	C_α^{n-1}
$5(n-1)+2$	C_α^n	$5(n-1)+1$	$5(n-1)$	$5(n-2)+4$	N^n	H^n	C^{n-1}
$5(n-1)+3$	H_α^n	$5(n-1)+2$	$5(n-1)+1$	$5(n-1)$	C_α^n	N^n	H^n
$5(n-1)+4$	C^n	$5(n-1)+3$	$5(n-1)+2$	$5(n-1)+1$	H_α^n	C_α^n	N^n

It should be emphasized that, for each element of ρ, except for the nitrogen atoms (N^i), the reference atoms are the three immediate predecessors (see Table **??**).

Binary Representation for the Protein Backbone

There are different representations for proteins. In principle, they can be represented by strings formed from 23 characters, each representing one of the possible amino acid residues. This is a unique representation but is not geometric. Another possible representation is a list of three-dimensional coordinates defining the location of each atom composing the protein.

Since the solution space of a DDGP can be organized as a binary tree, a solution can be represented as a binary string that incorporates geometric information. Formally, following the ordering ρ (given in (**??**)), each atom at position x_i can be associated with a bit b_i based on its relative position to the plane formed by its reference atoms (see Section **??**). Thus, the Cartesian coordinates of each protein segment can be mapped, in a one-to-one relationship, to a *binary sequence* $b = (b_1, \dots, b_n)$.

Given that the first three atoms of ρ are easily fixed in $\mathbb{R}^3$ and have no reference atoms (see Figure **??**), we consider $b_1 = b_2 = b_3 = 0$, without the loss of generality. It is noted that for the fourth atom of ρ, the two possible positions for immersing it in $\mathbb{R}^3$, resulting from the intersection of the spheres associated with x_1, x_2 and x_3, are always both feasible, as v_4 never has a fourth preceding neighbor. This implies that for every solution x where $b_4 = 0$, there exists another solution x', where $b_4 = 1$, obtainable by reflecting x with respect to plane π_4 that passes through points x_1, x_2, and x_3.

As a direct consequence of this unique characteristic of v_4, we know that the binary representation of x' is the total inversion of the bits of b from the fourth position onwards. Therefore, to reduce the representation of structures obtained in this manner, we normalize our dataset by inverting all binary representations where $b_4 = 1$. With this choice, the binary representations in our database have the format $b = (0, 0, 0, 0, b_5, \dots, b_n)$. We adopt the reduced representation $s = (b_5, \dots, b_n) \in \{0, 1\}^{n-4}$, removing the fixed bits.

Extracting Binary Subsequences

The known distances *a priori* relate either atoms within the same residue or the atoms of consecutive residues. In addition, it is known that small stretches of proteins can follow geometric patterns. Particularly, Ramachandran plots highlight some of these patterns for sequences of three residues.[?] To incorporate the geometric relations between adjacent residues, we extracted all *binary subsequences* associated with groups of up to five consecutive residues in a protein chain.

It should be noted that, except for the first residue in a protein segment, each residue is composed of five distinct atoms to be considered, namely, N, C_α, C, H, and H_α, as illustrated in Figure **??**. Consequently, the binary subsequences obtained vary in length with the number of consecutive residues considered, being 5, 10, 15, 20, and 25 bits, respectively.

Each extracted subsequence is related to the four bits immediately preceding it. As only the first subsequence is guaranteed to be normalized, i.e., the bit preceding it is defined as 0, it may be necessary to invert all bits of the analyzed subsequence to maintain the normalization of the strings in consideration. Therefore, in all cases, a check is made of the binary value of the atom immediately before the first atom of the binary subsequence in question. If this value is 1, all binary values in the subsequence are inverted.

For example, consider a five-residue segment with the following reduced binary sequence:

$$s = (1,0,0,1,1,0,1,1,1,0,0,0,1,1,0,1,0,1,0,1), \tag{3}$$

where the complete sequence $b = (0,0,0,0,s)$ includes the bits of the first residue.

The sections of s associated with each of its residues are

$$\begin{aligned} &\textit{Residue } 2: \quad (1,0,0,1,1,\\ &\textit{Residue } 3: \quad 0,1,1,1,0,\\ &\textit{Residue } 4: \quad 0,0,1,1,0,\\ &\textit{Residue } 5: \quad 1,0,1,0,1). \end{aligned} \tag{4}$$

Regarding the reduced sequence represented by s, we collect four subsequences from one residue, three subsequences from two residues, two subsequences from three residues, and one subsequence from four residues. Note that the sequence s is composed of bits associated with only four residues, which implies the absence of subsequences containing five residues.

The binary subsequences of three consecutive residues to be collected are those associated with the residue triplets (2, 3, 4) and (3, 4, 5), with $s_1 = (1,0,0,1,1,0,1,1,1,0,0,0,1,1,0)$ and $s_2 = (0,1,1,1,0,0,0,1,1,0,1,0,1,0,1)$, respectively. The binary value at the position in s immediately before the section associated with residues (3, 4, 5) is 1 (i.e., the binary of the last position of *Residue 2*). Therefore, all binaries of s_2 should be inverted: $s_2 \leftarrow (1,0,0,0,1,1,1,0,0,1,0,1,0,1,0)$.

Our complete dataset with its 72,983 three-dimensional configurations and the respective binary codings can be automatically generated by the Python script available in the repository `https://github.com/romulomarques/proteinGeometryData` (accessed on 25 November 2023). In this repository, we also provide a *.csv* file containing the PDB IDs of all the instances that we downloaded.

The FBS Evaluation Function

A solution to the DDGP is represented by a viable path from the root to a leaf in the binary tree of its solution space. In DFS, the search for such a path is performed in depth, preferring the node 0 at each level. In the new search strategy, FBS seeks to identify a viable path based on the frequency of binary subsequences of lengths 5, 10, 15, 20, and 25. More specifically, we group the subsequences by length, count the frequency of each occurrence, and order them in descending frequency within each group.

Although the DDGP tree may have a height greater than 25, we restrict our analysis to subsequences of lengths 5, 10, 15, 20, and 25 for two reasons: the first is that subsequences of lengths that are multiples of five represent residues with all five of their atoms; and the second is that the frequency of subsequences larger than 25 is relatively low compared to the number of possible subsequences of smaller lengths. In situations where the DDGP tree has a height greater than the considered subsequences, the FBS strategy can be employed to incrementally construct a preferential path by concatenating consecutive subsequences.

Ideally, the optimal tree search strategy is one that requires the fewest number of visited nodes. For DFS, assuming the binary solution is $b = (b_1, \ldots, b_n)$, the number of visited nodes is given by

$$DFS(b) = n + \sum_{i=0}^{n-1} b_{n-i}(2^{i+1} - 1), \tag{5}$$

where n is the number of nodes in the path representing the solution, and the term b_{n-i} indicates whether the subtree rooted in the node to the left of the node associated with b_{n-i} was visited (valued 1 in this case and 0 otherwise). When $b_{n-i} = 1$, the total number of nodes in the associated subtree must be added, which is given by the factor $(2^{i+1} - 1)$.

In FBS, paths associated with the most frequent sequences are tested first. Unlike DFS, FBS does not take advantage of already traveled paths. Instead, each alternative path is tested separately. Thus, if the position of b in the FBS order is k, then k paths of length n must be tested, totaling a cost of

$$FBS(b) = k \times n. \tag{6}$$

Figure **??** illustrates a binary tree of height four, composed of nodes numbered from 1 to 15, following the DFS visitation order. Below the tree representation, squares display the FBS ordering of the eight paths from the root to the leaves (ordering learned from the PDB). Suppose that the path $(1, 9, 13, 14)$, highlighted in blue, is the binary representation of the solution. Also, suppose that b is in the third position in the FBS order. With these assumptions, only node 15 would not be visited by DFS in the search for solution b, and the FBS algorithm would evaluate three paths of length four, involving a total of 12 nodes: $(1, 9, 10, 11), (1, 2, 6, 7)$, and $(1, 9, 13, 14)$.

Applying Equations (**??**) and (**??**), we obtain

$$DFS(b) = 4 + 0 \times (2^1 - 1) + 1 \times (2^2 - 1) + 1 \times (2^3 - 1) = 14$$

and

$$FBS(b) = 3 \times 4 = 12.$$

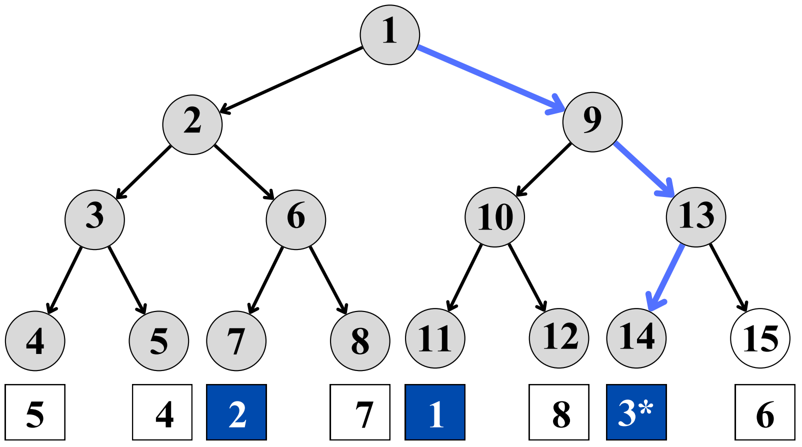


Figure 5: Binary tree nodes visited by the DFS and FBS. The number in each node represents the node position in the visiting order of the DFS. The number in a square at the bottom of each tree leaf represents the position of the respective root–leaf path in the FBS. The arcs in blue indicate the root–leaf path corresponding to the solution of the 3-DDGP. The blue squares indicate which root–leaf paths are explored by the FBS until it finds the solution (marked with *).

Computational Results

In this section, we present a descriptive statistical analysis of the binary subsequences extracted from the PDB. We highlight that the distribution of these binary subsequences is not uniform. Furthermore, we compare the performance of DFS and FBS in terms of the number of nodes visited in the search for solutions in the binary tree of the DDGP and ???.

In Table **??**, for each length, the column *Count* denotes the number of binary subsequences present in our dataset, the column k represents the number of distinct subsequences, and the column $k_{\max}$ indicates the maximum number of possible binary substrings, defined as $k_{\max} = 2^{length}$. The column $k/k_{\max}$ reveals the fraction of observed subsequences in relation to the universe of mathematically possible subsequences. Finally, the column $Count/k_{\max}$ is the ratio of the number of binary subsequences to the maximum number of possible subsequences.

Table 2: Frequency information of binary subsequences of each length: first column exhibits the binary subsequence lengths; second column exhibits the number of binary subsequences collected from PDB; third column exhibits the number of unique binary subsequences collected; fourth column exhibits the number of unique binary subsequences that are mathematically possible to exist. The fifth and sixth columns are ratios of the second and third columns to the fourth column, respectively.

Length	***Count***	k	k_{max}	k/k_{max}	$Count/k_{max}$
5	758,977	32	32	1.000	23,718.03
10	686,020	396	1024	0.387	669.94
15	613,063	4268	32,768	0.130	18.71
20	540,106	54,742	1,048,576	0.052	0.52
25	475,732	189,904	33,554,432	0.006	0.01

In Table **??**, except for length 5, the fraction of observed subsequences (k/k_{max}) remains well below 1. On the other hand, for lengths 10 and 15, the sample size (*count*) is significantly larger than the maximum number of possible subsequences (k_{max}). For example, for length 15, the sample is almost 19 times larger than k_{max}. These data, i.e., the low fraction of observed subsequences in a relatively large sample, suggest at least three possible interpretations: either we have an unrepresentative sample (even though we used all NMR data from the PDB), or "Nature" does not generate all mathematically possible binary subsequences, or some substrings have an extremely low probability of occurrence. However, this reasoning is not applicable to subsequences of length 20 and 25, as the sample size is considerably smaller than the number of possible subsequences: for length 20, the ratio of *count* to k_{max} is 0.52; for length 25, the ratio drops to just 0.01.

For the application of the FBS strategy, we assign indices to the binary subsequences, ordering them in descending order with respect to their occurrence probabilities. After this, we normalize the indices by dividing them by the total number of subsequences of each length. As a result, the normalized indices have values in the range $(0, 1]$. That is, if we have sequences s_1, s_2, and s_3 with occurrence probabilities of 0.1, 0.6, and 0.3, respectively, then the normalized indices would be 1/3 for s_2, 2/3 for s_3, and 1 for s_1.

Figure **??** shows the probability distribution (relative frequencies) for the binary subsequences of lengths 5, 10, 15, 20, and 25 identified by their normalized indices ($\mu = 10^{-6}$). Since the probability distributions show a considerable and monotonic decline, we can conclude that these distributions are not uniform. This reinforces that, in the PDB, some binary subsequences are favored.

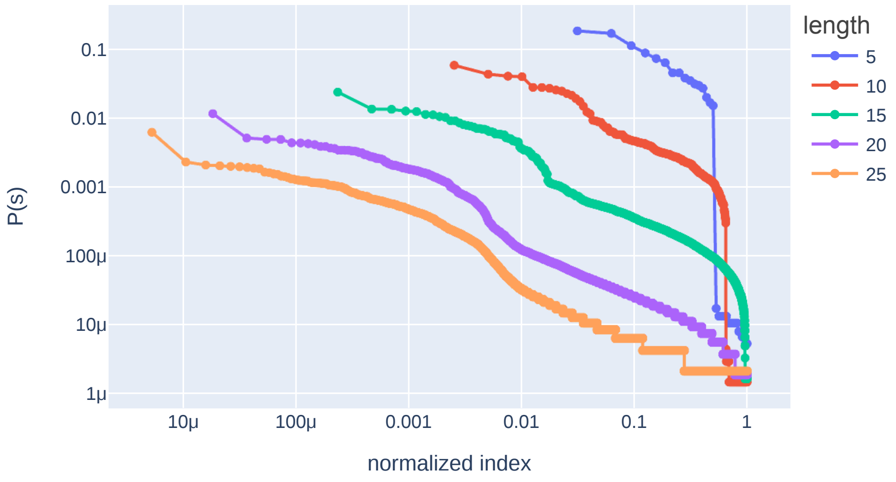


Figure 6: Probability, in logarithmic scale, of each binary subsequence configuration extracted from PDB, for each binary subsequence length. Each binary subsequence configuration is extracted from PDB, and is mapped, in a descending order of probabilities, to a real index in the interval $(0, 1]$. The indexes ('x' axis) are also in logarithmic scale.

Indeed, the most frequent subsequences have probabilities of orders of magnitude greater than the others. Still, in Figure **??**, the difference between the two highest probabilities for each length is clearly seen. In Figure **??**, the ratios between these probabilities are displayed. For example, for subsequences of length 20, the ratio between the two highest probabilities is 2.26, and for length 25, this ratio is 2.71.

Figure **??** presents the accumulated probabilities for subsequences with lengths of 5, 10, 15, 20, and 25, using the same normalized indices shown in Figure **??**. It is observable that, for all lengths, the first 10% of the normalized indices, which represent the most frequent subsequences, accumulate at least 50% of the probabilities. This observation highlights the differences in the distribution of the occurrence probabilities of the binary subsequences in the PDB.

In Figure **??**, the accumulated probability distribution is presented, describing the relationship between the costs associated with the execution of DFS and FBS in the search for all the binary subsequences contained in our dataset. The costs considered are the number of nodes visited during the exploration of the binary tree of the DDGP and are calculated using Equations (**??**) and (**??**). In this context, values greater than 1 indicate the superior performance of the FBS algorithm compared to DFS. As can be seen, the relative performance of FBS increases with the length of the subsequences. Furthermore, for all lengths, DFS performs better in, at most, 30% of the subsequences (that is, for every subsequence length, $P(DFS/FBS < 1) \leq 0.3$).

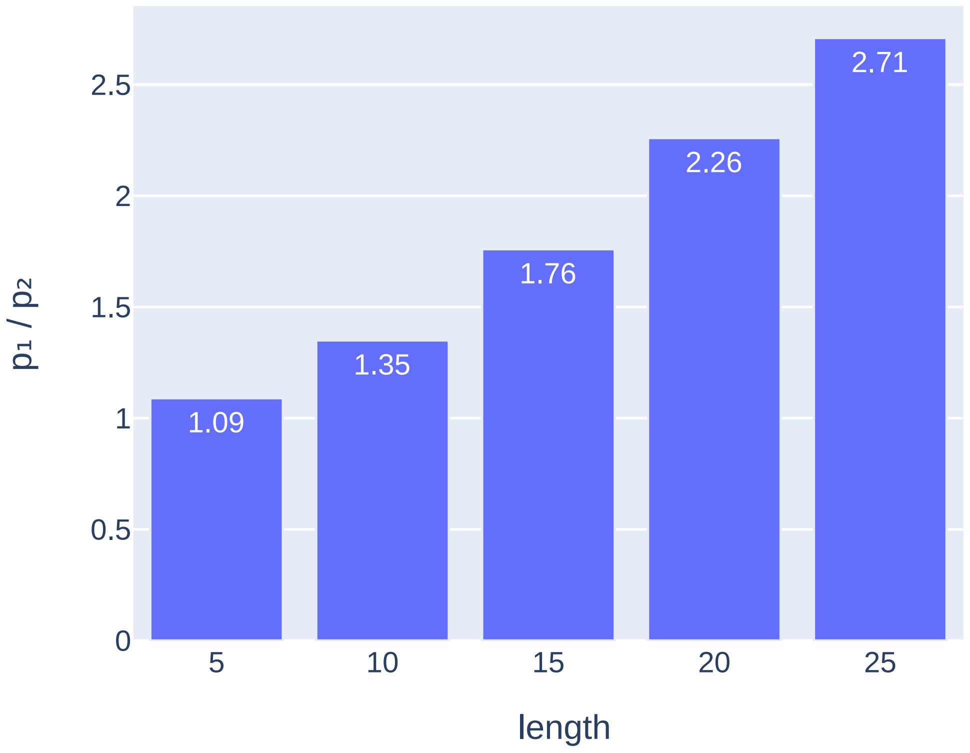


Figure 7: Ratio between the probabilities of the most frequent (first point of a color in Figure **??**) and the second most frequent (second point of a color in Figure **??**) binary subsequences, for each subsequence length.

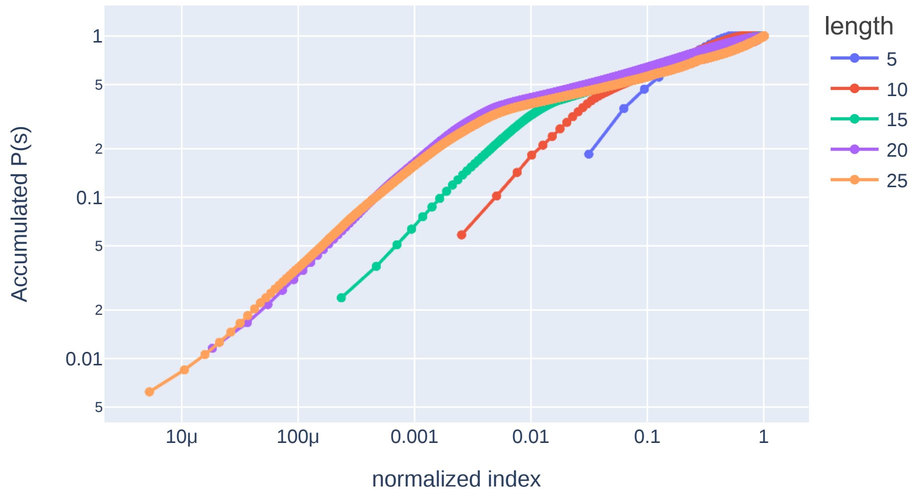


Figure 8: Accumulated probability of occurrence of the protein binary subsequences. Each binary subsequence configuration extracted from PDB is mapped, in a descending order of probabilities, to a real index in the interval $(0, 1]$. The indexes ('x' axis) and the accumulated probabilities ('y' axis) are in logarithmic scale.

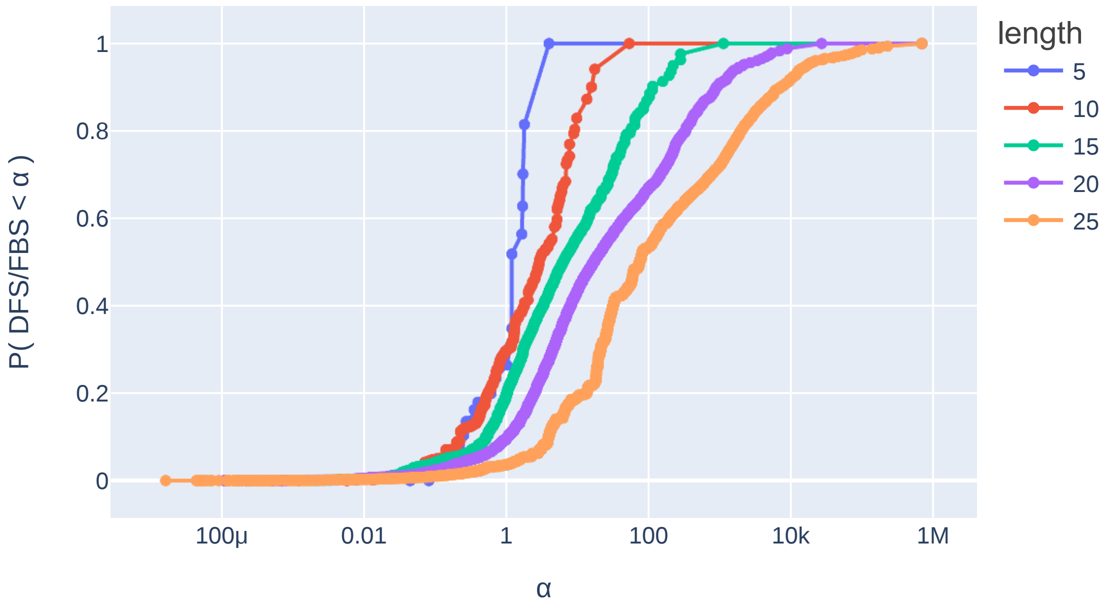


Figure 9: The probability that the relative cost (the ratio of DFS cost to FBS cost) for binary subsequences of different lengths being less than α, where α can go up to 1 million. If α is greater than 1, it means the FBS performs better than the DFS.

Figure **??** displays the fraction of instances in which the cost of the DFS algorithm is at least α times greater than the cost of the FBS algorithm. When $\alpha = 1.5$, this fraction is

50% in instances involving the subsequences of length 5. However, for the subsequences of length 25, FBS is less costly in more than 90% of the instances. Remarkably, for the larger instances, FBS is up to thousands of times more efficient than DFS ($\alpha = 1000$). These data suggest that the performance of FBS relative to DFS increases considerably with the length of the instances.

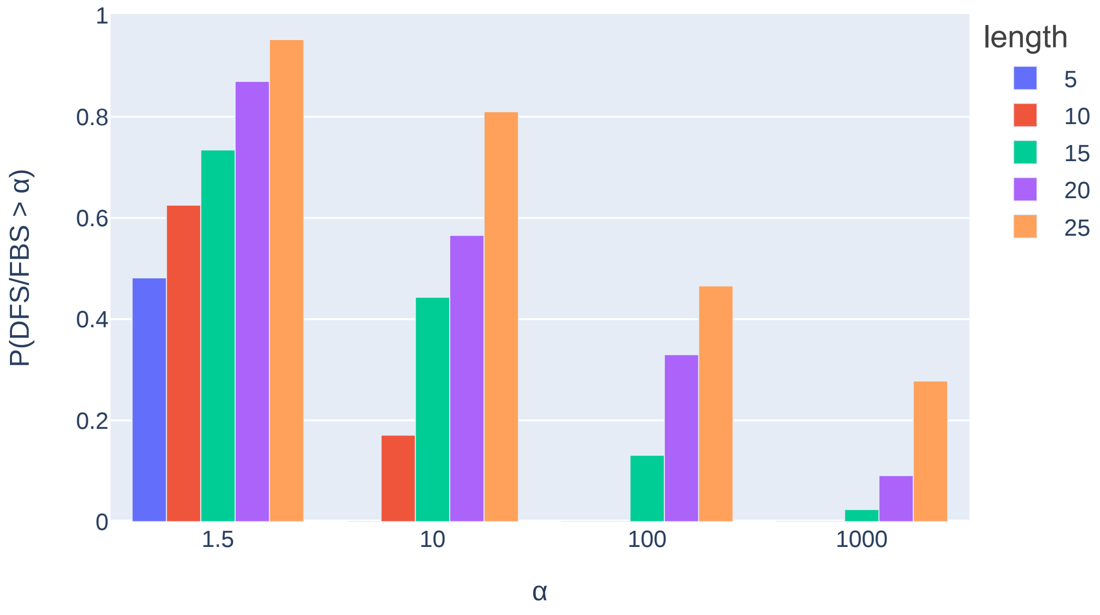


Figure 10: The likelihood of the relative cost (the quotient of DFS cost and FBS cost) exceeding 1.5, 10, 100, or 1000.

While our experiments demonstrate a discernible advantage of the FBS method over the traditional DFS, it is crucial to acknowledge a limitation in the scope of our data source. The binary subsequences that we have extracted and analyzed from the PDB represent only a specific subset of proteins, particularly those characterized using NMR techniques. Consequently, this subset may not comprehensively encompass the entire spectrum of possible protein subsequences. This limitation is significant because the NMR subset database of the RCSB PDB might display inherent biases, reflecting only a fraction of the diverse protein structures that exist in nature.

Therefore, while the results indicate the promising efficiency of FBS in the context of the analyzed data, we must consider the potential impact of these biases on our findings.

The dataset, primarily derived from NMR-based protein structures, might influence the frequency distribution patterns observed in our study, potentially skewing the generalizability of our results. Future research should aim to validate and extend our findings across a more diverse range of protein structures, including those determined through other methods like X-ray crystallography or cryo-electron microscopy, to ensure the broader and more unbiased application of the FBS method.

In the next section, we ???

In summary, while our current results with the FBS method are encouraging, further investigation using a more varied dataset is essential to fully understand its applicability and limitations in the wider context of structural biochemistry.

BP algorithm with FBS

???

Conclusions

This study represents a pioneering effort in integrating information from the Protein Data Bank (PDB) to solve the Discretizable Distance Geometry Problem (DDGP), a crucial model in determining protein structures via Nuclear Magnetic Resonance (NMR) data.

Our unique approach hinges on exploiting the combinatorial nature of the DDGP, the solution space of which is given by a binary tree. By developing a binary string representation for the coordinates of protein backbone atoms, we have observed distinctive patterns with non-uniform frequency distributions in these strings. This pivotal discovery led to the development of the Frequency Based Search (FBS) method, an innovative search technique within the DDGP solution space.

The efficacy of FBS over the conventional depth-first search method, as evidenced in our research, demonstrates at least 50% more efficiency in 70% of the cases tested. This efficiency is quantified by a reduced number of nodes visited in the search tree. Unlike the depth-first method that exhaustively explores the entire mathematical space, FBS strategically focuses on a specific subset of this space. This subset, informed by statistical data from the PDB, is more likely to yield viable and relevant solutions, thus significantly enhancing the search efficiency.

Expanding on the potential implications of this research, the FBS method opens up new possibilities in the realm of structural biochemistry. The efficient identification and analysis of protein structures using FBS can accelerate advancements in drug discovery, where understanding protein–ligand interactions is crucial. Additionally, this method can contribute significantly to enzyme research, aiding in the elucidation of enzyme mechanisms based on their structural configurations. In broader biochemical pathways, FBS can be instrumental in mapping out complex protein interactions, thereby providing deeper insights into cellular processes and molecular biology.

We incorporate FBS ??? into the Branch-and-Prune (BP) method represents a major advancement in computational biology. This adaptation, which effectively utilizes the geometric information available in the PDB, is set to significantly enhance the BP method's capability in protein structure determination. Consequently, our research not only marks a significant stride in computational biology but also sets the stage for transformative applications in structural biochemistry, enabling the more accurate, efficient, and comprehensive analysis of protein structures and their functions.

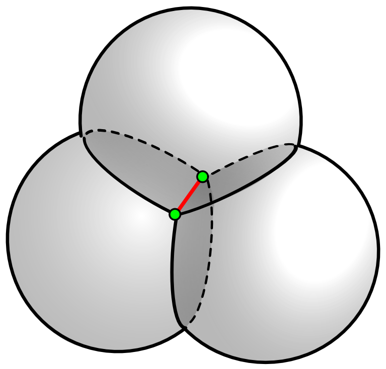
Acknowledgement

This research was funded by Brazilian research agencies FAPESP (grant numbers 2013/07375-0 and 2023/08706-1) and CNPq (grant number 305227/2022-0). We also extend our gratitude to the anonymous reviewers for their contributions to enhancing this paper.

Data and Software Availability

update here

The Jupyter Notebook file provided by the authors at `https://github.com/romulomarques/proteinGeometryData` (accessed on 15 March 2024) progressively generates all files and folders used in this research. However, since some of these folders occupy a significant amount of memory, such as the folder containing the files downloaded from the PDB, which amounts to 25 Gigabytes, the authors have decided to make available in the GitHub repository the folder containing the data of protein segments, which is the necessary information to actually reproduce the experiment. In addition, in the same repository, the authors provide a *.csv* file that lists the PDB IDs of all the downloaded 3D structures.



For Table of Contents Only