

Multiple Methods for Protein Side Chain Packing Using Maximum Weight Cliques

J.B. Brown¹

jbbrown@kuicr.kyoto-u.ac.jp

Dukka Bahadur K.C.²

dukka.kc@biology.gatech.edu

Etsuji Tomita³

tomita@ice.euc.ac.jp

Tatsuya Akutsu¹

takutsu@kuicr.kyoto-u.ac.jp

¹ Bioinformatics Center, Institute for Chemical Research, Kyoto University, Uji, 611-0011 Kyoto, Japan

² Center for the Study of Systems Biology, School of Biology, Georgia Institute of Technology, 250 14th Street, 30318 Atlanta GA, USA

³ Department of Information and Communication Engineering, The University of Electro-Communications, Chofu, 182-8585 Tokyo, Japan

Abstract

In this paper, we present several methods for computing a solution to the protein side chain packing problem, with all methods having a common solution approach of breaking the polymer into subpolymers and using maximum edge weight cliques to prune the search space for the optimal side chain packing. We characterize the graph sizes generated for each method and compare their prediction accuracies. These methods are demonstrated for computing proteins up to approximately 8000 residues. In addition, we update a result published previously.

Keywords: structural bioinformatics, side chain packing, protein structure prediction

1 Introduction

The protein side chain packing problem (SCPP) is an important problem in bioinformatics. The problem consists of, given a protein's backbone atoms (whether predetermined via throughput techniques or computational prediction), predicting the location of the resulting side chain atoms, concluding in the prediction of a protein's final conformation. Since a protein's structure affects its function, the ability to correctly predict a protein's structure affects the ability to correctly predict its corresponding function. Correct prediction of protein function paves the way to designing new chemicals and drugs in order to cure disease and illness.

The SCPP problem has been proven by Akutsu [1] to be NP-hard, so unless $P = NP$, methods to find the optimal solution to this problem in polynomial time are unlikely to exist. Hence, the focus of this paper and attention of researchers has been in how to develop other methods to find the optimal solution to the problem. These alternative schemes can be broadly classified into the two categories of heuristic and deterministic methods. For example, rapid tree decomposition [16] and integer programming approaches [7] are two deterministic methods, while SCAP [15] is one heuristic method to try and solve the problem.

In this paper, we examine the SCPP from the heuristic aspect of breaking it into subproblems, with each problem, when solved, contributing to the overall solution of the original prediction problem. The approach, first given by Bahadur *et al.* [8] in 2005 for linear subpolymers of even length, is generalized in this paper. Also presented in this work are several new methods for protein subdivision which are more realistic in considering a protein's spatiality.

Many of the methods developed for SCPP transform the problem into a graph [8, 16]. How the graphs are utilized is where the methods and results differ. As in the method of [8], the methods in

this paper also transform the problem into a graph in order to find a maximum weight clique, given an edge weighting function that reflects the frequency of contact pairs in a database of proteins. We include a characterization of each graph method so that methods can be compared. Compared with other complicated methods, the algorithms presented here are simple enough that they can be readily implemented for evaluation or usage.

The rest of this paper is organized as follows. The problem is concretely defined with an overall solution given in Section 2, along with a discussion of several methodologies for graph generation in Section 3. The results of those methodologies are presented and discussed in Section 4, including a revision of the results in [8]. In Section 5, we conclude with additional discussion of the results, including a fair assessment of the strengths and weaknesses of the approaches presented.

So as to not confuse the reader, we note that we interchangeably use the terms protein and polymer, with a subpolymer being a portion of the protein input. We also note that the main objective of this paper is not to correct [8], but more importantly to consider spatially realistic methods of protein subdivision for side chain packing.

2 Problem Definition

Computational protein structure prediction methods often are done in two phases. The first phase is threading, or predicting the location of only the backbone atoms. The second phase is side chain packing, or predicting the location of a protein’s side chain atoms based on its backbone atoms. Accurate chemical research is dependent on successful execution of both of these steps. Here, let us describe the overall framework for understanding the side chain packing problem, along with a general description of an algorithm for solving it.

2.1 Problem

Concretely, the SCPP can be phrased as this: given a protein of N amino acids and the atom positions for the backbone atoms in those acids, predict the physical location of each atom present in the side chains of the amino acids. Let us number the residues (amino acids) of the protein $a_1, a_2, a_3, \dots, a_N$. Each of these acids can be rotated into a particular conformation (rotamer) about the χ_1 axis, making a torsion angle between the C_α atom and the side chain. In [5, 16], rotamer libraries are used to define the rotation angles through which particular acids may rotate, though similar to [8], they are not used in this report. By excluding rotamer libraries, algorithm implementations need not be concerned with details of rotamer libraries, and additionally, the algorithm is applicable to proteins which are not covered by rotamer libraries. As will be shown, fears that an excessive number of rotamers are generated are quelled by eliminating many rotamers before graph generation.

In this paper and in [8], we wish to divide the protein P into p subpolymers P_1, P_2, \dots, P_p , where obviously $p < N$. For each subpolymer P_i , rotamers are generated for each amino acid $a_j \in P_i$, beginning at a random angle θ_j for each acid a_j , and generating a new rotamer at every $\frac{2\pi}{R}$ degrees, where R is the specified number of rotamers to generate per amino acid. Hence we generate rotamers $r_{u,j,v}$ for subpolymer u ’s j th residue rotated $\theta_j + (2\pi v/R)$ degrees. Note again that initial rotation angles θ_j are independently generated for each residue $a_1, \dots, a_N \in P$. Following the method of [10], the value of R is set to 20, but could be adjusted as needed for more or less precision.

Here, it becomes easy to understand why this combinatorial optimization problem is intractable. For a protein of N amino acids, we generate R rotamers, initially resulting in R^N configurations and, as to be explained, NR graph vertices to search. As the size of the protein increases to considerable size (e.g., the PDB Data Bank [4] structure *1xwl* has 580 residues), our search space grows rapidly, especially if we increase the polymer size and the number of rotations R simultaneously. Using the same example, this generation process immediately creates an initial search space of more than $3 \cdot 10^{754}$ possible configurations, and 11,600 graph vertices for the protein *1xwl*.

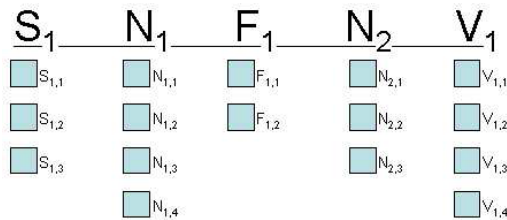


Figure 1: The side chain packing problem recast as a graph-theoretic problem. For each acid A_i , we generate up to R rotamers $A_{i,1}, \dots, A_{i,R}$. In this example $R = 4$, and those acids in with less than 4 rotamers have had some rotamers eliminated through consistency checks, as explained in Section 2.2.

We transform the problem into a graph theoretic-problem. This is done because there is a sense of spatial interconnectedness amongst the atoms in a protein, and we need our solution to reflect this. For each of the rotameric configurations we have generated, we let each individual configuration (of all atoms in an arbitrary rotamer) be represented by a vertex in a graph, as shown in Figure 1. To capture the interconnectedness of a real protein, we assign weights to the edges between vertices in our graph using a weighting function. The interested reader is directed to [12] for the details of the weighting function, but in short, the function uses a database of known protein structures and calculates an all-atom distance dependent probability function for assigning weights. Returning to our problem, we wish to find the maximum edge-weight clique in the graph (a clique is a connected graph, c.f. [6]) which represents the most likely final configuration of the protein being calculated. In order to ensure that the resulting clique is a spatially realistic polymer, we must check for consistency, as detailed below.

2.2 Consistency

After all rotamers are created in a sampling step, three checks are necessary to ensure that the rotamers and their respective atoms present in the resulting graphs are those that would not physically collide.

As a preprocessing step, we remove those rotamers generated for the acids glycine, alanine, and proline, following the method of [8]. These three acids typically do not have rotamers, and experiments show that this decreases the total number of rotamers to search by 15-25%.

The first check that is done is to eliminate those rotamers which collide with the fixed backbone atoms. Let $r_{u,j,v}$ be a rotamer in our search space. We check each side chain atom of $r_{u,j,v}$ against the backbone atoms of the entire input polymer P , and if any of the atoms collide, then we remove the rotamer from our list of candidate vertices. A rotamer is defined to collide with the backbone if the distance between any side chain atom of the rotamer and any atom of the backbone is less than 1\AA .

The second check that occurs is to verify the consistency of the rotamers in the subpolymer itself. In other words, we eliminate those rotamers which collide with other rotamers in the same subpolymer. We call this “internal consistency,” and in symbols we mean that if a rotamer $r_{u,j,v}$ collides with a rotamer $r_{u,k,x}$ for any k and x (except for $r_{u,j,v}$ itself), then the rotamers $r_{u,j,v}$ and $r_{u,k,x}$ are internally inconsistent. In a graph theory sense, this means setting the edge weight between the two edges to zero (“removing the edge”) so that the two vertices (rotamers) in question simultaneously cannot occur in the maximum clique for the subpolymer. Two rotamers are defined to collide if any one of the atoms from one rotamer is within 4\AA of the atoms from another rotamer.

The final check that must occur is that those rotamers which are internally consistent must also be externally consistent, or that they must not collide with rotamers from other subpolymers. Our distance constraint remains at 4\AA . Let $r_{u,j,v}$ be a rotamer that collides with a rotamer $r_{w,k,x}$ from another subpolymer. Then because $r_{u,j,v}$ creates a collision with a rotamer in another subpolymer, it

1. Load PDB.
2. Create all rotamers.
3. Trim redundant rotamers.
4. Trim backbone colliding rotamers.
5. Assign ID number to all remaining rotamers.
6. Create subpolymers and respective graphs.
7. Find cliques for subpolymer graphs.
8. Combine subpolymer clique solutions and output final solution (protein).

Figure 2: Algorithm for computing side chain packing of a protein.

cannot possibly be in the final solution, and hence, we remove all edges connected to rotamer $r_{u,j,v}$. Since we are considering only the subpolymer to which $r_{u,j,v}$ belongs to, we do not remove the edges connected to $r_{w,k,x}$, as those edges will be removed when we consider the subpolymer to which $r_{w,k,x}$ belongs to.

If a rotamer successfully passes these three checks, then it is a physically feasible rotamer. Since we are breaking the original polymer P into subpolymers P_i , we only weight the edges for which two rotamers are in the same subpolymer P_i (i.e., $r_{i,j,v}$ and $r_{i,k,x}$). Readers should see [12] for the details of the edge-weighting function.

2.3 Algorithm

As each subpolymer graph is made, the remaining step in the per-subpolymer algorithm is to find its maximum edge-weighted clique, and finally to use the union of subpolymer cliques as the solution to the original SCPP. We use the efficient WCQ weighted-clique algorithm developed by Suzuki *et al.* [13], which in turn makes use of the MCQ maximum clique algorithm developed by Tomita and Seki [14]. Interested readers can find the details in [13, 14], but we mention here that in our experiments a graph represented by a symmetric adjacency matrix of 3365x3365 entries (rotamers) had a maximum clique found in 1 second, with a maximum edge-weight clique found in 2 seconds.

Our overall algorithm to solve the SCPP is shown in Figure 2. We present a number of its details in order to accurately give a time complexity analysis on certain steps, leaving out constant time actions (that are typically program internal administrative duties).

Our analysis is as follows. Steps 1, 3, and 8 can be completed in $O(N)$ time, where N is the number of amino acids in the protein. Steps 2, 4, and 5 make use of at most R rotamers per acid, and can be done in $O(NR)$ time. Step 6, the crux of the paper, depends on the subpolymer creation method, and is analyzed in the Methods section below. Finding a maximum clique is known to be NP-hard [6].

3 Methods

In this section, we show three different methods for breaking a protein into subpolymers, and characterize the resulting graph sizes for each of their respective graph generation methods. Recall from Section 2.1 and Step 2 of Figure 2 that the initialization process automatically generates NR vertices, with some portion of them removed for redundancy in Step 3.

3.1 Linear Polymer

The most intuitive subpolymer is a linear subpolymer, shown in Figures 3 and 4. It is created by simply breaking up the amino acid sequence at specified residues and assigning the resulting fragments to subpolymers. As a toy example, if we have a protein with an amino acid sequence $P =$

SNFNVCRLPGTP, and we break it into 3 linear subpolymers, one possible assignment could give the resulting subpolymers $P_1 = SNFN$, $P_2 = VCRLPG$, $P_3 = TP$. The key to linear subpolymers is that their sequence order is preserved. Analysis of the graph size is then as follows: in the first subpolymer P_1 containing s_1 residues and hence a maximum of s_1R rotamers, it must check for consistency against itself and all remaining ($\leq NR$) rotamers in the original polymer P . As the external consistency check between subpolymers P_1 and P_2 has been done in this pass, there is no need for P_2 to check its external consistency again with P_1 , and hence, the second subpolymer only needs to check subpolymers P_2 through P_n , processing a total $s_2R * (N - s_1)R$ rotamers, where s_2 denotes the number of residues in subpolymer P_2 . Continuing in this fashion, the number of vertices processed over all subpolymers is

$$\begin{aligned} &\leq \left(s_1R(NR)\right) + \left(s_2R(N - s_1)R\right) + \left(s_3R(N - s_1 - s_2)R\right) + \cdots + \left(s_pR(s_pR)\right) \\ &= \left(\sum_{i=1}^p [s_iR] \left[(N - \sum_{j=1}^{i-1} s_j)R\right]\right). \end{aligned}$$

In this framework, if we consider the formulation in [8], the number of residues processed over all subpolymers of even size s_0 is

$$\begin{aligned} &= \left(\sum_{i=1}^p [s_0R] \left[(N - \sum_{j=1}^{i-1} s_0)R\right]\right) \\ &= \left(s_0R^2 \left[\sum_{i=1}^p N - \sum_{i=1}^p \sum_{j=1}^{i-1} s_0\right]\right) \\ &= \left(s_0R^2 \left[Np - \frac{s_0p^2}{2} + \frac{s_0p}{2}\right]\right), \end{aligned}$$

and since all polymers are of size $s_0 = \frac{N}{p}$, the total number of graph vertices generated over p even length subpolymers is

$$\leq \left(\frac{1}{2}N^2R^2 \left[\frac{p+1}{p}\right]\right).$$

We can expect the non-uniform length subpolymer graph generation method will create a similar number of vertices, using the fact that $N = s_0p = (s_1 + s_2 + \cdots + s_p)$.

As the number of subpolymers increases, the total amount of graph vertices generated is reduced, until we reach the limit

$$= \lim_{p \rightarrow \infty} \frac{1}{2}N^2R^2 \left[\frac{p+1}{p}\right] = \frac{1}{2}N^2R^2.$$

This clearly establishes the advantage of the linear subpolymers proposed here and in [8], which have a total number of vertices kN^2R^2 , where $\frac{1}{2} \leq k \leq 1$, over the technique in [9], which is always N^2R^2 .

A drawback of the algorithm in [8] is that by dividing a polymer into even length divisions, we fail to solve the problem in a biological sense. Instead, in order to improve the technique's applicability, one can divide a protein into linear subpolymers at biologically significant residues using data from the Uniprot [2], Pfam [3], and Interpro [11] databases. Results of computing side chain packing using linear subpolymers and this data are given in Section 4.

3.2 Centroid Subpolymers

Since a protein typically does not stay linear and flat, alternative subpolymer methods are also useful for solving the SCPP by decomposition. Our next approach to division by subpolymer division was done using the centroid of a protein, illustrated in Figure 5. We find the centroid of a protein by

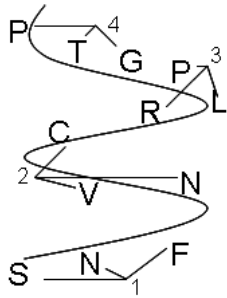


Figure 3: In this example, the sample protein $P = SNFNVCLPGTP$, which resembles a beta sheet, is divided into four even-length subpolymers, as in [8]. The numbers indicate resulting subpolymers.

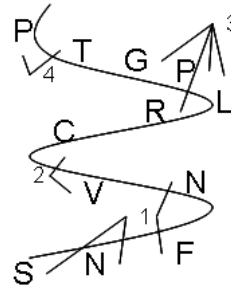


Figure 4: The same protein as in Figure 3, but broken into biologically significant clusters (subpolymers) where turns occur. Subpolymers are no longer of the same length.

adding the respective X, Y, and Z coordinates of every atom in the protein and dividing by the total number of atoms. We then determine how many residues are located on each side of the axes running through the centroid, and use the axis with the minimum residue difference as the axis to break the original polymer into two subpolymers P_{\leq} and $P_{>}$. Note here that the residues no longer need to preserve order as they did in a linear subpolymer. Here, the analysis is fairly simple. We have two subpolymers, of size s_1 and s_2 such that $s_1 + s_2 = N$, and we must check each polymer for internal consistency and external consistency against the other subpolymer. It is obvious that this requires $\leq (s_1^2 R^2 + s_1 s_2 R^2) + (s_2^2 R^2 + s_2 s_1 R^2)$ rotamer consistency checks for both graph generations, which is equivalent to the number of vertices as the $N^2 R^2$ method in [9], because the centroid method is equivalent to twice generating the first subpolymer of a linear subpolymer. A perfect centroid evenly divides the amino acids into two even-length subpolymers, though experiments rarely yielded this. Where this algorithm becomes useful is in its consideration of the spatial arrangement of atoms. Results from this type of graph generation are given in Section 4.

3.3 Distance Bin Subpolymers

Our final method of creating subpolymer graphs is by using distance bins. In this approach, we find the maximum and minimum X, Y, and Z coordinates among all atoms in the protein. We then take the axis with the maximum width (i.e., maximum of $x_{max} - x_{min}$, $y_{max} - y_{min}$, and $z_{max} - z_{min}$) and divide it amongst b bins of even length. We put each amino acid into the corresponding distance bin (thus we do not preserve order), using the centroid of that acid with respect to the axis being used for binning. The process is shown in Figure 6. For subpolymer graph generation, we need only consider internal consistency and the two adjacent subpolymers for external consistency, rather than the entire original polymer. In practice, we add fictitious empty subpolymers to both ends of the protein so that the subpolymer graph generation works like a sliding window on bins $1 \dots b$. For each call to the graph generation method on a subpolymer of size s_i residues with neighbors of size s_{i-1} and s_{i+1} , we need $\leq (s_i R(s_{i-1} R + s_i R + s_{i+1} R))$ consistency checks. Taken over b bins, we have (where s_0 represents our fictitious empty subpolymers)

$$\begin{aligned} &\leq s_1 R(s_0 R + s_1 R + s_2 R) + s_2 R(s_1 R + s_2 R + s_3 R) + \dots + s_b R(s_{b-1} R + s_b R + s_0 R) \\ &= R^2(s_1^2 + 2s_1 s_2 + s_2^2 + 2s_2 s_3 + \dots + 2s_{b-1} s_b + s_b^2) \end{aligned}$$

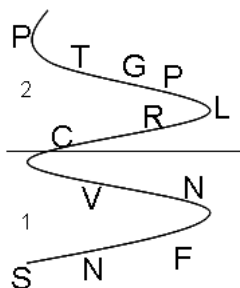


Figure 5: The example polymer is divided into two subpolymers at its centroid. In this two-dimensional example, the Y axis is used for doing the division work. Though the order of residues and their respective rotamers is preserved in this example, it would not be preserved if the X axis were used.

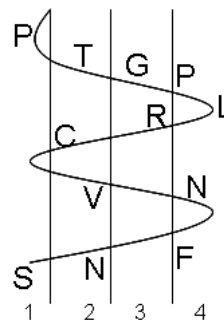


Figure 6: The example polymer $P = SNFNVCRLPGTP$ is broken into four distance bins ($P_1 = SP$, $P_2 = NVCT$, $P_3 = RG$, and $P_4 = FNLP$) along the X axis. The subpolymers have differing size and no longer preserve the original protein’s sequence order.

$$= R^2(N^2 - 2 \sum_{i=1}^{b-2} \sum_{j=i+2}^b s_i s_j)$$

total graph vertices to check over all subpolymers that are created. It is obvious that the distance bin method can create no more vertices than the linear subpolymer or centroid method (unless $b = \{1, 2\}$). A value of $b = 1$ or $b = 2$ agrees with the $N^2 R^2$ result of the linear and centroid methods, as it intuitively should.

Interestingly, when we observed the numbers of acids that fell into each distance bin, we generally found the values to follow a Gaussian distribution, though we make no further analysis based on this observation.

3.4 Completed Analysis

Combing the analysis from Section 2 with the methods in Section 3, it can be seen that the graph generation step is the most most demanding, except for the distance binning method (as long as $b \neq \{1, 2\}$). Even in the case of processing a whole protein as a single subpolymer (for linear and distance bin cases), the worst case number of rotamer comparisons is $N^2 R^2$.

4 Experimental Results

After Bahadur *et al.* reported in [8] RMSD scores for an even-length protein division method, a mistake in the code was discovered that affected the final results of the method. Fixing this code mistake, several polymers were re-evaluated, and the result is that several polymers no longer achieve maximum cliques. We display the results of the experiment using corrected code in Table 1, where “dnf” denotes that a maximum clique was not achievable using the specified number of even-length subpolymer divisions. This new result is not a setback but rather a confirmation that polymers cannot simply be broken apart at random locations without some biological consequence. Furthermore, this motivated the study presented here for breaking polymers into pieces based on biologically relevant locations or more realistic spatial considerations.

In order to evaluate the methods proposed in this paper, we used the same set of proteins as in [8, 10], and set a time limit of 5 minutes for finding a maximum clique. The results of the side chain

Table 1: Side chain packing results reported in [8] (in parentheses), and adjusted values after program code corrections. “dnf” means the protein did not finish computing in under 30 minutes, a limit set based on the packing runtimes in [8]. The column headings indicate the number of even-length subpolymers p the original protein was divided into. All calculations are RMSD in Å.

PDB	Residues	$p = 2$	$p = 3$	$p = 4$	$p = 5$
1xwl	580	(1.66)dnf	(1.56)dnf	(1.60)dnf	(1.74)dnf
1gof	639	(1.63)dnf	(1.46)dnf	(1.69)dnf	(1.70)dnf
1biy	689	(1.53)dnf	(1.64)dnf	(1.77)1.84	(1.72)1.87
1aa6	696	(1.58)1.87	(1.68)1.80	(1.61)1.83	(1.70)1.82
1a81	812	(1.56)dnf	(1.63)dnf	(1.63)dnf	(1.75)dnf
1lnh	836	(1.50)1.93	(1.63)dnf	(1.64)1.87	(1.55)dnf
1fiy	874	(1.42)dnf	(1.53)dnf	(1.63)dnf	(1.54)dnf

Table 2: The results, in RMSD Å, of computing the same proteins used in [8, 10], applying the methods described in this paper, where b is the number of distance bins used. “dnf” signals that a maximum weight clique was not found within a time limit of 5 minutes. “DBAve.” is the average result for the distance bin method using 5-20 bins.

PDB	Residues	Linear	Centroid	$b = 10$	$b = 11$	$b = 12$	$b = 13$	$b = 14$	$b = 15$
1xwl	580	2.036	2.055	1.897	1.887	1.804	1.818	1.826	1.873
1gof	639	1.876	dnf	dnf	dnf	dnf	1.742	dnf	dnf
1biy	689	dnf	dnf	1.850	dnf	dnf	1.738	1.721	1.806
1aa6	696	dnf	dnf	dnf	dnf	dnf	dnf	dnf	1.733
1a81	812	dnf	dnf	dnf	dnf	dnf	dnf	dnf	1.823
1lnh	836	dnf	dnf	dnf	1.806	1.855	1.840	dnf	1.764
1fiy	874	dnf	1.937	1.785	dnf	dnf	1.740	1.830	1.810
PDB	Residues	Linear	Centroid	$b = 16$	$b = 17$	$b = 18$	$b = 19$	$b = 20$	DBAve.
1bhe	376	1.832	1.802	1.546	1.421	1.559	1.629	1.581	1.540
1xwl	580	2.036	2.055	1.646	1.738	1.827	1.744	1.886	1.843
1gof	639	1.876	dnf	1.743	1.650	1.661	1.620	1.678	1.732
1t70	2040	dnf	dnf	1.747	1.838	1.822	1.870	1.819	1.804
1aon	8015	dnf	dnf	dnf	dnf	dnf	1.607	1.628	1.622

prediction are shown in Table 2. Overall, at least one of the methods succeeds in finding a solution to the SCPP for a protein, with many proteins being successfully calculated for multiple methods. Though not given here due to space considerations, the computation times were highest for the centroid method, which agrees with our analysis above, and gradually lowered as the number of distance bins increased. A number of proteins had improved accuracy as the number of bins increased, signifying that a particular subpolymer could realistically consider only its immediately adjacent neighbors to predict the side chain conformations.

To put this study into perspective on a realistic set of proteins for an application, we tested a small subset of proteins that were extracted by querying the Protein Data Bank [4] with the phrase “DNA excision repair.” We imposed the constraint that we compare our result against only the first model of a PDB, though we include all chains. The results of the proposed methods on the protein set are shown in Table 3. Note that many DNA excision repair proteins are smaller than the test sets used in [8, 10], and subsequently, several proteins could be calculated using all three techniques.

It is to note here that while many proteins did not finish within the 5 minute time limit, if we

Table 3: Evaluation of the methods in this paper on proteins with the common theme of DNA excision repair, where b is the number of bins used in the distance bin method. “dnf” signals that a maximum weight clique was not found within a time limit of 5 minutes. Results are in RMSD Å. “DBAve.” is the average result for the distance bin method using 5-20 bins.

PDB	Residues	Linear	Centroid	b=16	b=17	b=18	b=19	b=20	DBAve.
1d4u	111	dnf	dnf	1.293	1.441	1.257	1.507	1.523	1.341
1qze	214	2.358	dnf	1.507	1.467	1.488	1.464	1.427	1.560
1xsn	324	1.976	2.043	1.852	1.709	1.710	1.698	1.779	1.769
2uug	445	dnf	dnf	1.724	1.833	1.823	1.912	dnf	1.787
1xsl	1305	dnf	dnf	1.846	1.812	1.831	1.864	1.860	1.820

let the clique finder execute without a time restriction, it would eventually find and return a clique corresponding to the solution for a particular subpolymer. For the purposes of speed, we set our limit low.

5 Conclusion

The algorithms presented here offer simple ways to understand and generate solutions to the protein side chain packing problem. They can be implemented relatively quickly, and with good implementations can be very useful for prediction on older computers or devices with a minimum of memory and processor abilities. Many of the proteins evaluated in this study could have their side chains calculated using less than 64 MB of memory. Excluding the clique finding step, which has the demonstrated possibility of failing to find a solution in a given amount of time, the algorithms require no more than N^2R^2 consistency checks, which, as demonstrated via the distance bin method, is often not required.

While this time is not optimal, the algorithm is still useful, particularly for quick evaluation of small proteins with a large number (> 50) of rotamers. This is akin to consider whether to implement insertion sort or quick sort for a list of at most 10 numbers. While quick sort’s speed is an asset in the limit, insertion sort’s practicality on small datasets cannot be denied, and similarly, the algorithms given in this report can be very practical for some proteins, as opposed to some of the more complicated solutions to the SCPP.

Many extensions to this work are possible, including hybridized methods (e.g., first divide by centroid, then divide the resulting centroids by linear or distance bin subpolymers), recursive centroids, inclusion of rotamers augment the search space, inclusion of disulfide bridges and other physicochemical constraints, or a decomposition method specifying an upper limit to the number of residues in a subpolymer. It would be very interesting to set up a mathematical program to solve for the optimal number of bins in the distance bin method, in hopes of minimizing the overall number of rotamer consistency checks.

Acknowledgments

J.B. Brown is grateful to be supported by a scholarship from the Japanese Ministry of Education, Culture, Sports, Science and Technology.

References

- [1] Akutsu, T., NP-hardness results for protein side-chain packing, *Genome Inform.*, 8:180–186, 1997.

- [2] Bairoch, A., Apweiler, R., Wu, C.H., Barker, W.C., Boeckmann, B., Ferro, S., Gasteiger, E., Huang, H., Lopez, R., Magrane, M., Martin, M.J., Natale, D.A., O'Donovan, C., Redaschi, N., and Yeh, L.S., The Universal Protein Resource (UniProt), *Nucleic Acids Res.*, 33:D154–D159, 2005.
- [3] Bateman, A., Coin, L., Durbin, R., Finn, D., Hollich, V., Griffith-Jones, S., Khanna, A., Marshall, M., Moxon, S., Sonnhammer, E., Studholme, D., Yeats, C., and Eddy, S., The Pfam protein families database, *Nucleic Acids Res.*, 32:D138–D141, 2004.
- [4] Berman, H.M., Westbrook, J., Feng, Z., Gilliland, G., Bhat, T.N., Weissig, H., Shindyalov, I.N., and Bourne, P.E., The protein data bank, *Nucleic Acids Res.*, 28:235–242, 2000.
- [5] Canutescu, A.A., Shelenkov, A.A., and Dunbrack, R.L. Jr., A graph-theory algorithm for rapid protein side-chain prediction, *Protein Sci.*, 12:2001–2014, 2001.
- [6] Cormen, T., Leiserson, C., Rivest, R., and Stein, C., *Introduction to Algorithms, Second Edition*, The MIT Press, 2001.
- [7] Eriksoon, O., Zhou, Y., and Elofsson, A., Side chain-positioning as an integer programming problem, *Proc. the First International Workshop on Algorithms in Bioinformatics*, Springer-Verlag, 128–141, 2001.
- [8] Dukka Bahadur, K.C., Brown, J.B., Tomita, E., Suzuki, J., and Akutsu, T., Large scale protein side chain packing based on maximum edge-weight clique finding algorithm, *Proc. 2005 International Joint Conference of InCoB, AASBi and KSBI, BIOINFO 2005*, 228–233, 2005.
- [9] Dukka Bahadur, K.C., Tomita, E., Suzuki, J., and Akutsu, T., Protein side-chain packing problem: A maximum edge-weight clique algorithmic approach, *J. Bioinform. Comput. Biol.*, 3(1):103–126, 2005.
- [10] Dukka Bahadur, K.C., Ph.D. dissertation, Kyoto University, 2006.
- [11] Mulder, N.J., Apweiler, R., Attwood, T.K., Bairoch, A., Bateman, A., Binns, D., Bradley, P., Bork, P., Bucher, P., Cerutti, L., Copley, R., Courcelle, E., Das, U., Durbin, R., Fleischmann, W., Gough, J., Haft, D., Harte, N., Hulo, N., Kahn, D., Kanapin, A., Krestyaninova, M., Lonsdale, D., Lopez, R., Letunic, I., Madera, M., Maslen, J., McDowall, J., Mitchell, A., Nikolskaya, A.N., Orchard, S., Pagni, M., Ponting, C.P., Quevillon, E., Selengut, J., Sigrist, C.J., Silventoinen, V., Studholme, D.J., Vaughan, R., and Wu, C.H., InterPro, progress and status in 2005, *Nucleic Acids Res.*, 33:D201–D205, 2005.
- [12] Samudrala, R. and Moulton, J., An all-atom distance-dependent conditional probability discriminatory function for protein structure prediction, *J. Mol. Biol.*, 257:893–914, 1998.
- [13] Suzuki, J., Tomita, E., and Seki, T., An algorithm for finding a maximum clique with maximum edge-weight and computational experiments, *Technical Report MPS*, 45–48, The Information Processing Society of Japan, 2002.
- [14] Tomita, E. and Seki, T., An efficient branch-and-bound algorithm for finding a maximum clique, *Proc. DMTCS 2003*, Lecture Note in Computer Science, 2731:278–289, 2003.
- [15] Xiang, Z. and Honig, B., Extending the accuracy limits of prediction for side-chain conformations, *J. Mol. Biol.*, 311:421–430, 2001.
- [16] Xu, J., Rapid protein side-chain packing via tree decomposition, *Proc. 9th Annual International Conference on Research in Computational Molecular Biology (RECOMB 2005)*, 423–329, 2005.