An Efficient Algorithm for Polymer Sequence Design

Jon M. Kleinberg[1]

January, 1998

Abstract. *Polymer sequence design* is a natural inverse problem to protein structure prediction: given a *target structure* in three dimensions, we wish to design an amino acid sequence that will fold to it. A model of Sun, Brem, Chan, and Dill casts this problem as an optimization on a space of sequences of hydrophobic (H) and polar (P) monomers; the goal is to find a sequence which achieves a dense hydrophobic core with few solvent-exposed hydrophobic residues. Sun et al. developed a heuristic method to search the space of sequences, without a guarantee of optimality or near-optimality; Hart subsequently raised the computational tractability of constructing an optimal sequence in this model as an open question. Here we answer this question by providing an efficient algorithm to construct optimal sequences; the method has a polynomial running time, and performs very efficiently in practice. We illustrate the implementation of our method on structures drawn from the Protein Data Bank, and discuss some possible extensions of the model.

Keywords: inverse protein folding, sequence design problems, combinatorial optimization, network flow algorithms.

1 Introduction

Polymer sequence design. The computational aspects of *protein structure prediction* are centered around the development of algorithms capable of producing reliable information about the three-dimensional structure of a protein from its amino acid sequence [18]. There is a natural inverse problem to this, the object of several recent studies, that has the potential to provide new insights on the basic problems of this area [8, 19, 24, 22, 7, 21, 15]. This *polymer sequence design* problem, in its simplest statement, is the following: given a three-dimensional structure, find an amino acid sequence that will fold to it.

A number of concrete computational formulations of this problem have recently been advanced [24, 22, 7, 21]. At more or less explicit levels, these formulations attempt to capture the competing requirements of *positive design* — the designed sequence should have low free energy in the target structure — and *negative design* — there should be very few other "competing" structures in which the designed sequence has free energy comparable to that of the target structure [6, 24]. A designed sequence satisfying some version of both these requirements has the potential to fold to the target conformation. In most of the current formulations of the sequence design problem, these conditions are approached implicitly through the construction of a global optimization function on the space of possible sequences; the function is designed to reward features that a "good" sequence is presumed to possess [22, 7, 21, 15].

[1]Department of Computer Science, Cornell University, Ithaca NY 14853. Email: kleinber@cs.cornell.edu. Phone: +1 607 255 7316. Fax: +1 607 255 4428. Supported in part by an Alfred P. Sloan Research Fellowship and by NSF Faculty Early Career Development Award CCR-9701399.

In this paper, we focus primarily on one such approach to the sequence design problem proposed by Sun, Brem, Chan, and Dill [21], the *Grand Canonical* (GC) model. Their concern was with a restricted, more "robust" variant of the sequence design problem: rather than trying to assign an amino acid to each residue position in the target structure, they were interested in deciding, for each residue position, whether it should be occupied by a *hydrophobic* (H) or *polar* (P) residue. This can be viewed as sequence design with a *binary folding code*, in which there are only two amino acids, H and P. There has been extensive work (see e.g. [17, 6, 16]) suggesting that modeling a protein as a heteropolymer with only two amino acid types captures many of the qualitative aspects of protein structure.

Our aim in this paper is to provide, for the first time, an efficient algorithm for designing provably optimal sequences in the GC model. Such optimal sequences provide a means of assessing the effectiveness of the GC model as an approach to designing biologically plausible sequences.

The GC model of Sun et al. In order to fully define the GC model, we must specify the geometric representation of the target structure, and the optimization function on the set of possible sequences. Following Sun et al. [21], the target structures will be structures of proteins whose native conformations are known; this allows for the most informative test of the computational techniques, since there is a natural "true" sequence corresponding to each such target structure. A simplified geometric representation of such a structure is obtained by constructing a sphere of the appropriate radius at the location of each non-hydrogen backbone atom, and replacing the side chain of each residue with a single "side chain bead," of radius 2Å, at a distance of 3Å along the C_α-C_β bond vector [21]. In this way, the residues in the target structure are made "uniform." (Sun et al. position the side chain bead for a glycine residue at a distance of 3Å along a vector inferred from the tetrahedral geometry of its C_α; we, on the other hand, position the glycine side chain bead at the location of the C_α. We find that the final results are affected very little by this decision.) For each side chain bead, one also computes the area of its solvent-accessible *contact surface* with respect to a standard 1.4Å solvent probe [20]. (For this, we used the algorithm and implementation ASC of Eisenhaber and Argos [10, 11].)

We now define the optimization function in the GC model, for a given target structure with n residues. To design a sequence, we must specify which residues in the target structure will be H, and which will be P; thus, we say that a *polymer sequence* $\mathcal{S}$ is a sequence of n symbols, each of which is either H or P. We use $\mathcal{S}_H$ to denote the set of numbers i such that the i^{th} position in the sequence $\mathcal{S}$ is equal to H; we define $\mathcal{S}_P$ analogously. Now, the *potential* $\Phi(\mathcal{S})$ of a sequence $\mathcal{S}$, with respect to the target structure, is a scoring function motivated by the following (partially conflicting) requirements. We would like the H residues in $\mathcal{S}$ to have low solvent-accessible surface area; we would also like H residues to be close to one another in space, so as to form a compact *hydrophobic core.* Thus one defines Φ by

$$\Phi(\mathcal{S}) = \alpha \sum_{\substack{i<j-2 \\ i,j \in \mathcal{S}_H}} g(d_{ij}) + \beta \sum_{i \in \mathcal{S}_H} s_i.$$

Here s_i denotes the area of the solvent-accessible contact surface of the side chain for residue

i (in $\mathring{\mathrm{A}}^2$), and d_{ij} denotes the distance between the side chain centers of residues i and j (in $\mathring{\mathrm{A}}$). g is a sigmoidal function that rewards small distances; in [21] it is defined to be $\frac{1}{1+\exp(d_{ij}-6.5)}$ for $d_{ij} \leq 6.5\mathring{\mathrm{A}}$. Finally, $\alpha < 0$ and $\beta > 0$ are scaling parameters; in [21] they are given default values of $\alpha = -2$ and $\beta = \frac{1}{3}$.

Summary of Our Results. The goal of the GC model is to design a sequence of minimum potential (i.e. as negative as possible); we will call such a sequence *optimal.* Clearly this corresponds to constructing a sequence with many close-range H-H contacts, and very few solvent-exposed H's. Sun et al. [21] noted that there are 2^n possible amino acid sequences in the binary H/P model — too many to perform an exhaustive search — and developed a heuristic method to find low-potential sequences based on a genetic algorithm. Their method does not provide any measure of how close the final designed sequences are to the optimal sequence(s). Hart subsequently studied the optimization problem in the restricted discrete case of a chain in a three-dimensional cubic lattice [15]; he provided a polynomial-time algorithm to find an optimal sequence for one particular setting of the parameters in the potential function, within this lattice model. He raised several natural open questions [15] — specifically, whether there is an efficient algorithm to find optimal sequences for *any* choice of parameters in the lattice-based potential function, and whether there is an efficient algorithm to find optimal sequences in the full *geometric* model of Sun et al., where one can deal with real protein structures.

In this paper, we resolve these questions by providing an efficient algorithm to find optimal polymer sequences in the GC model, for any setting of parameters, in both the geometric and lattice settings. We use the term *efficient* here in two related senses. First, our algorithm has a running time that is bounded by a polynomial function of n, the length of the sequence; as such, it satisfies the standard theoretical definition of *efficiency* (e.g. [12, 4]). Second, we have implemented a version of the algorithm (on top of discrete optimization code of Cherkassky and Goldberg [3]), and it runs very efficiently on real data. We will discuss this in more detail below; see also the plot in Figure 1, which depicts the running time of our implementation (as a function of sequence length) on 25 target protein structures from the Protein Data Bank (PDB) [1].

Our efficient algorithm for GC sequence design is based on the efficient solvability of a well-studied problem in discrete optimization — that of finding the *minimum cut* between two vertices in a directed graph. This latter problem belongs to the general class of *network flow problems*, for which a range of efficient algorithms have been developed [9, 14, 4, 3]. When applied to the sequence design problem, these flow-based algorithmic techniques can be viewed as performing a type of *local search* on the space of possible sequences that converges to the optimum in a small number of steps; in this way, exponential brute-force search for an optimal sequence is avoided. It is hoped that the algorithmic methods developed in this paper will suggest promising uses for network flow techniques in other biomolecular structure problems as well.

Given our method for designing optimal sequences in the GC model, we are able to assess more accurately the limits of this model as a means for producing biologically realistic sequences for a target structure. Since we will be dealing with target structures obtained from real proteins, let us define the *natural H/P sequence* for a protein structure to be the one obtained by translating the protein's true amino acid sequence into an H/P sequence, according to a designation of each of the twenty amino acids as either hydrophobic or hydrophilic. (Following Sun et al., we map the amino acids A,C,F,I,L,M,V,W,Y to H, and the others to P.) Since the potential function Φ associated with the model is designed only to approximate the factors favoring the *natural sequence*, the natural sequence is likely to be *sub-optimal* when scored according to Φ with respect to its structure; correspondingly, the optimal sequence under Φ may differ non-trivially from the natural sequence. Sun et al. compared the sequences obtained by their heuristic algorithm with the natural sequences of 23 protein structures from the PDB, and determined the percentage agreement in each case. Because we are now able to solve the GC model to optimality, we are able to determine the percentage agreement between the optimal and natural sequences (Figure 2, in the column under "Basic Algorithm.") Overall, the results appear to be mixed; for some proteins, the natural sequence agrees significantly more with the optimal sequence than with the sequence obtained by the method of Sun et al.; for others, solving the GC model to optimality produces sequences that have less agreement than the sequences obtained by the earlier heuristic method. We will discuss these issues in detail below.

The remainder of the paper is organized as follows. In Section 2, we describe and analyze the basic algorithm for computing the optimal sequence for a given target structure in the GC model. Section 3 discusses the results of applying this algorithm to the 23 protein structures considered by Sun et al., and also indicates some extensions of the GC model that can also be efficiently solved to optimality. One of these extensions — the *Scaling Algorithm* — is discussed in Section 4; this is a method to adaptively set the parameters α and β in the potential function Φ so that the relative proportions of H and P residues approximately match the naturally occurring frequencies of hydrophobic and hydrophilic residues in proteins.

Throughout the paper, we discuss computational methods for the three-dimensional geometric version of the GC sequence design problem. However, our methods also yield efficient algorithms for the natural lattice versions of the GC model that have been studied [21, 15]; we do not go into the details of this, but it is a straightforward adaptation of the methods presented in Section 2.

2 Computational Methods

The Minimum Cut Problem. We now address the problem of computing an optimal sequence in the GC model. Our method involves some terminology from graph theory that we introduce here for the sake of completeness; we refer the reader to a text such as [4] for further details.

A *directed graph* G consists of a pair of sets: V (the *vertices*) and E (the *edges*). Each edge $e \in E$ is an ordered pair of vertices $e = (u, v)$; we call u the *tail* of e and v the *head*. We also assume that each edge has a given *capacity* c_e, which is either a positive number or ∞ (indicating that the edge has infinite capacity). Let s and t be two vertices of G. An *s-t cut* in G is a partition of V into two sets, X and Y, so that $s \in X$ and $t \in Y$; we denote such a cut by the pair (X, Y). Roughly, an s-t cut is a way of dividing the graph into two

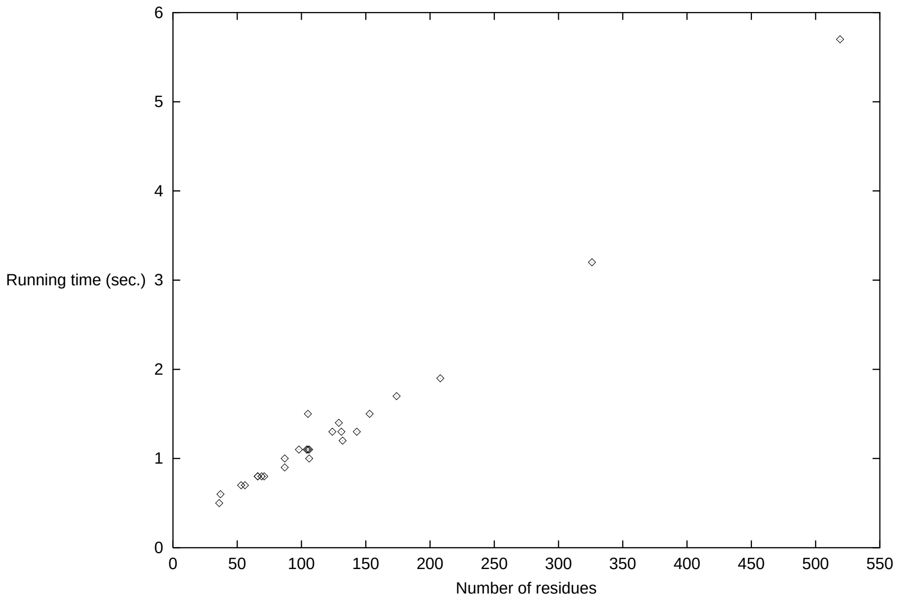

Figure 1: Running time as a function of sequence length

pieces so as to separate s from t. We say that an edge *crosses* a cut (X, Y) if it has its tail in X and its head in Y. The *capacity* of a cut (X, Y) is equal to the sum of the capacities of all edges which cross (X, Y); the capacity of (X, Y) is ∞ if and only if any of these edges has infinite capacity.

The *minimum s-t cut problem* asks, for a given graph G and vertices s and t, to find the s-t cut (X, Y) of minimum capacity. (This minimum will be equal to ∞ if and only if G has no s-t cut of finite capacity.) Although the details are beyond the scope of this paper, the minimum s-t cut problem can be solved, for an arbitrary graph with n vertices and m edges, by an algorithm with a running time proportional to $mn \log n$ [14], and efficient implementations exist for such algorithms (e.g. [3]).

Designing an Optimal Sequence. Let Φ be the potential function corresponding to a given target structure of length n. Recall that Φ includes a function g, which is applied to inter-residue distances, and which takes the value 0 when the distance is greater than a fixed cutoff γ. ($\gamma = 6.5\text{Å}$ in [21]). We also recall that the scaling parameters α and β are numbers that are negative and positive, respectively; aside from this, we make no assumptions about their values.

We define the following graph G based on Φ. The vertex set V of G consists of s, t, a vertex v_i for each of the residue positions $i = 1, 2, \ldots, n$ in the target structure, and a vertex u_{ij} for each pair of residue positions i, j for which $i < j - 2$ and the inter-residue distance d_{ij} is at most γ. (So the vertex u_{ij} does not exist for all pairs i, j.) The edge set E of G consists of an edge (s, u_{ij}) for each vertex u_{ij}, an edge (v_i, t) for each vertex v_i which has a non-zero solvent-exposed contact surface area s_i, and edges (u_{ij}, v_i) and (u_{ij}, v_j) for each vertex u_{ij}. We refer the reader to Figure 3 for an example of the directed graph constructed

Protein (PDB code)	sequence length	Sun et al. algorithm	Basic algorithm	Scaling algorithm
1aaj	105	66	73	72
1aba	87	81	75	70
1aps	98	72	78	77
1arr (mon)	53	62	62	64
1arr (dim)	106	73	71	74
1bba	36	58	58	58
1bbl	37	68	57	68
1bov	69	74	74	70
1brq	174	68	74	71
1cis	66	64	68	64
1cmb (mon)	104	62	63	64
1cmb (dim)	208	70	74	73
1hel	129	74	79	78
1ifb	131	76	69	70
1kba (mon)	66	72	68	76
1kba (dim)	132	73	77	74
2gb1	56	80	79	70
2hpr	87	78	78	78
2il8	71	77	72	79
256b	106	81	77	75
3cln	143	62	72	70
3rn3	124	81	69	69
3trx	105	80	77	81
Length-weighted average		72.1	72.6	72.2

Figure 2: Results for sequences

by this procedure from an artificial 9-residue structure. We now assign a capacity to each edge as follows. We assign a capacity of $|\alpha| g(d_{ij})$ to the edge (s, u_{ij}), a capacity of βs_i to the edge (v_i, t), and a capacity of ∞ to all edges of the form (u_{ij}, v_i) and (u_{ij}, v_j).

Note that G has an s-t cut of finite capacity — consider, for example, the cut (X, Y) in which $X = \{s\}$ and $Y = V - \{s\}$. We say that a set X of vertices is *closed* if (i) X contains s but not t, and (ii) for each $u_{ij} \in V$, X contains u_{ij} if and only if it contains both v_i and v_j. We now have the following fact.

(1) *If (X, Y) is a minimum s-t cut in G, then X is a closed set.*

To see why this is true, suppose that X contains a vertex u_{ij} but not the vertex v_i (the case of v_j is the same). Then the edge (u_{ij}, v_i) crosses (X, Y) and it has infinite capacity; this contradicts the assumption that (X, Y) is a minimum cut. Conversely, suppose that X contains some pair of vertices v_i and v_j, but not the vertex u_{ij}. Then the s-t cut $(X \cup \{u_{ij}\}, Y - \{u_{ij}\})$ would have smaller capacity than (X, Y), again a contradiction. This completes the proof of (1).

For an n-symbol H/P sequence $\mathcal{S}$, let Z denote the set of all vertices v_i for which position i in $\mathcal{S}$ is labeled H. Let $X(\mathcal{S})$ denote the closed set consisting of s, the vertices in Z, and all

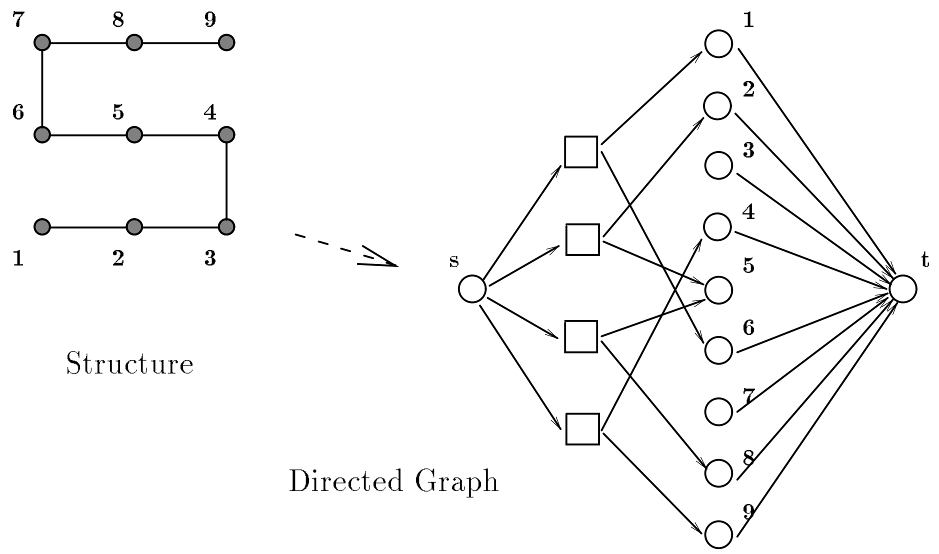

Figure 3: A small example of the construction of a directed graph from a target structure with four possible contacts (1-6, 2-5, 5-8, 4-9).

vertices u_{ij} for which $v_i, v_j \in Z$. Conversely, if X is a closed set, let $\mathcal{S}(X)$ denote the H/P sequence in which position i is labeled H if v_i belongs to X, and is labeled P if v_i does not belong to X. From these constructions, we see that

(2) *There is a one-to-one correspondence between n-symbol H/P sequences and closed sets in G.*

Now we come to the crucial fact about the graph G. Let B denote the quantity $\sum_{u_{ij} \in V} |\alpha| g(d_{ij})$.

(3) *Let X be a closed set and $\mathcal{S}(X)$ the corresponding H/P sequence. Then the capacity of the s-t cut (X, Y) is equal to $B + \Phi(\mathcal{S}(X))$.*

We can prove this fact as follows. From the definition of *closed set*, we know that the only edges crossing (X, Y) have the form (v_i, t), where $v_i \in X$, or (s, u_{ij}), where one of v_i or v_j does not belong to X. Thus,

$$\begin{aligned} c(X,Y) &= \sum_{\substack{u_{ij} \in V \\ \{v_i, v_j\} \not\subset X}} |\alpha| g(d_{ij}) + \sum_{v_i \in X} \beta s_i \\ &= B - \sum_{\substack{u_{ij} \in V \\ \{v_i, v_j\} \subset X}} |\alpha| g(d_{ij}) + \sum_{v_i \in X} \beta s_i \\ &= B + \alpha \sum_{\substack{i < j-2 \\ i,j \in \mathcal{S}(X)_H}} g(d_{ij}) + \beta \sum_{i \in \mathcal{S}(X)_H} s_i . \\ &= B + \Phi(\mathcal{S}(X)). \end{aligned}$$

This completes the proof of (3).

Amino Acid	Basic algorithm (% agreement)	Scaling algorithm (% agreement)
ala	42	47
arg	93	78
asn	90	85
asp	89	82
cys	73	87
gln	75	62
glu	89	86
gly	75	60
his	70	59
ile	61	68
leu	53	71
lys	92	87
met	42	56
phe	57	76
pro	69	54
ser	83	78
thr	84	74
trp	62	76
tyr	55	70
val	56	67

Figure 4: Results by amino acid type

Thus the potentials of H/P sequences for the target structure differ from capacity of the corresponding cut in G simply by the fixed additive constant B. Consequently, if (X, Y) is a minimum capacity s-t cut in G, then $\mathcal{S}(X)$ is an optimal sequence — so to find an optimal sequence for the target structure, we need only construct the graph G and compute a minimum capacity s-t cut in it. Let p denote the number of residue pairs in the target structure whose distance is at most the cut-off value γ; since the graph G has a number of vertices and edges proportional to $n + p$, we have

(4) *An optimal sequence in the GC model can be computed in time proportional to* $(n+p)^2 \log(n+p)$

Excluded-volume constraints in three dimensions imply that for each residue, only a small constant number of other residues will lie within the cut-off distance γ. Thus, p can be assumed to be proportional to n and hence the running time of the algorithm is proportional to $n^2 \log n$ — roughly quadratic in the length of the sequence, rather than exponential.

3 Computational Experiments, Discussion, Extensions

We implemented the algorithm of Section 2, making use of the highly efficient code of Cherkassky and Goldberg for computing the maximum flow in a network [3]. We tested the implementation on the 23 structures considered by Sun et al. [21], as well as on two larger protein structures — pepsin (326 residues) and pyruvate kinase (519 residues). The running

Protein (PDB code)	Basic alg., natural seq.	Basic alg., random seq.	Scaling alg., natural seq.	Scaling alg., random seq.
1aaj	0.72	0.60	0.73	0.59
1aba	0.82	0.60	0.83	0.58
1aps	0.83	0.59	0.83	0.58
1arr (mon)	0.68	0.61	0.73	0.63
1arr (dim)	0.77	0.62	0.82	0.60
1bba	0.59	0.61	0.64	0.65
1bbl	0.73	0.62	0.84	0.67
1bov	0.77	0.57	0.77	0.55
1brq	0.77	0.60	0.76	0.58
1cis	0.70	0.57	0.71	0.55
1cmb (mon)	0.73	0.61	0.72	0.58
1cmb (dim)	0.82	0.59	0.83	0.60
1hel	0.84	0.55	0.86	0.58
1ifb	0.78	0.64	0.85	0.66
1kba (mon)	0.78	0.61	0.79	0.61
1kba (dim)	0.81	0.60	0.80	0.59
256b	0.85	0.59	0.84	0.57
2gb1	0.79	0.60	0.82	0.58
2hpr	0.84	0.57	0.84	0.57
2il8	0.81	0.61	0.85	0.60
3cln	0.81	0.61	0.85	0.60
3rn3	0.77	0.58	0.76	0.56
3trx	0.82	0.61	0.83	0.60

Figure 5: Objective function scores of true and random HP sequences

times of the algorithm (in CPU seconds) on a Sun Sparc 10 are depicted in Figure 1; for the structures of lengths 36–208 considered in [21], the running times ranged from 0.5 to 1.9 seconds; for the largest structure (519 residues), the running time was 5.7 seconds.

In Figure 2, we compute the percentage agreement between the natural and optimal sequences for the 23 structures from [21] (in the column under "Basic Algorithm"); we also reproduce the numbers of Sun et al. for the sake of comparison. It is interesting to note the markedly varied way in which the percentages of overlap change, for different structures, as we move from heuristically designed sequences to the optimal sequences. This is in keeping with the observation above that *optimality* in the GC model is not the same as achieving *identity* with the natural sequence. Designed sequences for some structures, such as ribonuclease A (3rn3), showed significantly less agreement with the natural sequence when solved to optimality. For others, the percentage agreement increased. One case worth noting is calmodulin (3cln), whose percentage agreement jumped from 62% to 72% when solved to optimality. This is interesting, as Sun et al. had conjectured that the low level of agreement for calmodulin, relative to most of the other structures studied, was due to intrinsic aspects of its structure.

We can study the relation between the optimal and natural sequences at other levels as well. In Figure 4, we show the percentage agreement on the 23 sample structures, organized by amino acid type. That is — over all target structure positions occupied by a given amino acid, what was the percentage agreement between the natural and optimal structures? It is clear that the GC model produces much better agreement for some amino acids than for others — some notable patterns are that agreement is markedly better for polar residue positions than for non-polar positions, and best for acidic and basic residues such as arginine and lysine. Indeed, alanine and methionine positions were classified as polar more than half the time. One possible explanation for the lower level of agreement among non-polar residue positions, parallel to observations of several previous authors [21, 22], is the recurring presence of exposed non-polar residues on protein surfaces for reasons of biological function, something that the simple optimization function of the GC model does not take into account.

Finally, Figure 5 shows the score obtained by the natural sequence when plugged into the GC optimization function Φ. This is depicted in the second column, and compared with the average results for 100 randomly generated H/P sequences of the same composition in the third column. The scores are linearly normalized so that the *potential* of the optimal sequence is mapped to 1, while the "baseline" quantity $\beta\sum_{i=1}^{n} s_i$ is mapped to 0. For the ensembles of 100 random structures, the standard deviations for almost all are concentrated in the interval $[0.04, 0.065]$. Thus in all cases but one (the short bovine pancreatic polypeptide 1bba, for which the optimal sequence assumed all P values), the natural sequence scores significantly better than the ensemble of random H/P sequences.

One benefit of using an efficient optimal algorithm is that it allows us the flexibility to try a range of modifications to and variations on the basic method. Many of these variations brought about very little qualitative change in the results: for example, using a step function for g in the definition of Φ; changing the cut-off value γ; or replacing the surface areas s_i with discretized values. We now discuss a more significant extension of the model, in which it is still possible to design optimal sequences efficiently.

Modeling multi-residue interactions. Φ as defined above involves terms based only on single residues and pairs of residues. But we may want to consider the effect of k-wise interactions among residue positions, for $k > 2$ — for example to reward the proximity of three tightly packed hydrophobic residues, beyond the gain in Φ obtained from considering them as three separate pairs. We can formulate such a model as follows.

Let D_R denote the distance matrix for a set R of residues, and g^* denote a function that operates on arbitrary distance matrices. Finally, let $\mathcal{X}$ denote a collection of *candidate sets* — all sets for which we may want to include a term in the overall potential function. Thus in the basic GC model, g^* would simply be the function g, and $\mathcal{X}$ would be the set of all pairs of residue positions (i, j) with $i < j - 2$; now, more generally, $\mathcal{X}$ may contain sets of size greater than 2. Finally, we define the more general version of Φ by

$$\Phi^*(\mathcal{S}) = \alpha \sum_{\substack{K\in\mathcal{X}\\ K\subset\mathcal{S}_H}} g^*(D_K) + \beta \sum_{i\in\mathcal{S}_H} s_i.$$

Although we do not go into the details here, the methods of Section 2 can be used to efficiently minimize the potential Φ^*, in time proportional to $(n+p)^2\log(n+p)$, where p is now the number of sets K in $\mathcal{X}$ for which $g^*(D_K)$ may take a non-zero value. We have begun preliminary investigations of a potential function with terms for three-way interactions of neighboring residues.

4 Sequences of Approximately Fixed Composition

Certain of the optimal sequences constructed have a sharp imbalance in the ratio of H to P residues, and in most cases this *a fortiori* prevents them from having a high degree of agreement with the natural sequence. We now describe a extension of our basic algorithm — the *Scaling Algorithm* — which attempts to construct a sequence in which the ratio of H to P residues is roughly 2/3, matching the relative frequencies of the corresponding amino acids in naturally occurring polypeptide sequences [5].

The relative values of the parameters α and β in the potential Φ control the relative proportions of H and P residues in an optimal sequence. Qualitatively, one can see this as follows: as α is made increasingly negative, for fixed β, there is an increasing reward for hydrophobic contacts; as β is made increasingly positive, for fixed α, there is an increasing penalty for solvent-exposed hydrophobic residues. We are actually able to prove analytically that Φ as defined in the GC model satisfies a very clean *monotonicity condition* that captures this notion; we state the condition below but omit the proof from this version of the paper.

(5) *Let $\alpha' < \alpha < 0$; fix $\beta > 0$. Let Φ be the potential with parameters α and β, and Φ' the potential function with parameters α' and β. Then for every optimal sequence $\mathcal{S}$ with respect to Φ, there exists an optimal sequence $\mathcal{S}'$ with respect to Φ' such that the H-residue positions in $\mathcal{S}$ are a subset of the H-residue positions in $\mathcal{S}'$ ($\mathcal{S}_H \subset \mathcal{S}'_H$). That is, the H-residue positions in the optimal sequences form a "nested" progression as we lower α.*

The Scaling Algorithm simply uses repeated calls to the basic algorithm of Section 2 to determine the value of α (with β fixed at the default of $\frac{1}{3}$) for which the optimal sequence has approximately the appropriate fraction of H residues. We note that a formulation of the sequence design problem due to Shakhnovich and Gutin [22] directly imposes the constraint of a fixed ratio of H residues to P residues, leading to an optimization problem different from what we consider here. In our case, rather than performing an optimization with the value of this ratio imposed as an explicit constraint, we attempt to achieve the ratio indirectly by varying the parameters in the GC model.

The effect of the Scaling Algorithm on the overall agreement between optimal and natural sequences is summarized in Figures 2-5. Again, this effect was mixed. It is interesting to note the contrast in the results of the Scaling Algorithm on the two short sequences 1bba and 1bbl. Each had a very low ratio of H to P with the standard setting of the parameters (0/36 and 4/33 respectively); but the Scaling Algorithm produced a significant increase in agreement in the latter case while producing none in the former. We do not currently have a convincing structural explanation for this contrast.

5 Conclusion

We have provided an efficient algorithm to construct optimal sequences in the *Grand Canonical* sequence design model of Sun et al. [21], and investigated the performance of the algorithm on structures drawn from the Protein Data Bank. Our algorithm and its implementation are based on very efficient methods for solving *network flow problems* [3]; we believe such methods have the potential to provide efficient optimization algorithms in the context of further structure-related problems.

References

[1] F.C. Bernstein, T.F. Koetzle, G.J.B. Williams, E.F. Meyer, M.D. Brice, J.R. Rodgers, O. Kennard, T. Shimanouchi, M.Tasumi, *J. Molecular Bio.* 112(1977), pp. 535–542.

[2] H.S. Chan, K.A. Dill, *Proteins: Structure, Function, and Genetics* 24(1996), pp. 335–344.

[3] B. Cherkassky, A.V. Goldberg, "On implementing push-relabel method for the maximum flow problem," *Proc. MPS Symp. on Int. Prog. and Comb. Optimization* 1995, pp. 157–171. Implementation available at `http://www.neci.nj.nec.com/homepages/avg/soft/soft.html`.

[4] T. Cormen, C. Leiserson, R. Rivest. *Introduction to Algorithms.* McGraw-Hill, 1990.

[5] T.E. Creighton, *Proteins: Structure and Molecular Properties*, Freeman, 1993.

[6] K.A. Dill, S. Bromberg, K. Yue, K. Fiebig, D. Yee, P. Thomas, H.S. Chan, *Protein Science* 4(1995), pp. 561–602.

[7] J.M. Deutsch, T. Kurosky, *Phys. Rev. Letters* 76(1996), pp. 323–326.

[8] K.E. Drexler, *Proc. Nat. Acad. Sci.* 78(1981), pp. 5275–5278.

[9] J. Edmonds, R.M. Karp, *Journal of the ACM* 19(1972), pp. 248–264.

[10] F. Eisenhaber, P. Argos, *J. Computational Chemistry* 14(1993), pp. 1272–1280.

[11] F. Eisenhaber, P. Lijnzaad, P. Argos, C. Sander, M. Scharf, *J. Computational Chemistry* 16(1995), pp. 273–284.

[12] M. Garey and D. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W.H. Freeman, 1979.

[13] N. Gō, H. Taketomi, *Proc. Nat. Acad. Sci.* 75(1978), pp. 559–563.

[14] A.V. Goldberg, R.E. Tarjan, *Journal of the ACM* 35(1988), pp. 921–940.

[15] W. Hart, Proc. RECOMB Conf. on Computational Molecular Biology 1997, 128–136.

[16] S. Kamtekar, J. Schiffer, H. Xiong, J. Babik, M. Hecht, *Science* 262(1993), 1680–1685.

[17] K.F. Lau, K. Dill, *Macromolecules* 22(1989), pp. 3986–3997.

[18] K. Merz, S. LeGrand, Eds., *The Protein Folding Problem and Tertiary Structure Prediction*, Birkhäuser, 1994,

[19] J. Ponder, F.M. Richards, *J. Molecular Bio.* 193(1987), pp. 63–89.

[20] T.J. Richmond, F.M. Richards, *J. Molecular Bio.* 119(1978), pp. 537–555.

[21] S. Sun, R. Brem, H.S. Chan, K.A. Dill, *Protein Engineering* 8(1995), pp. 1205–1213.

[22] E.I. Shakhnovich, A.M. Gutin, *Protein Engineering* 6(1993), pp. 793–800.

[23] E.I. Shakhnovich, G. Farztdinov, A.M. Gutin, M. Karplus, *Phys. Rev. Letters* 67(1991), pp. 1665–1668.

[24] K. Yue, K.A. Dill, *Proc. Nat. Acad. Sci.* 89(1992), pp. 4163–4167.