JOURNAL OF COMPUTATIONAL BIOLOGY
Volume 14, Number 6, 2007

Pp. 777–790
DOI: 10.1089/cmb.2007.R016

Hypergraph Model of Multi-Residue Interactions in Proteins: Sequentially-Constrained Partitioning Algorithms for Optimization of Site-Directed Protein Recombination

XIAODUAN YE,[1] ALAN M. FRIEDMAN,[2] and CHRIS BAILEY-KELLOGG[1]

ABSTRACT

Relationships among amino acids determine stability and function and are also constrained by evolutionary history. We develop a probabilistic hypergraph model of residue relationships that generalizes traditional pairwise contact potentials to account for the statistics of multi-residue interactions. Using this model, we detected non-random associations in protein families and in the protein database. We also use this model in optimizing site-directed recombination experiments to preserve significant interactions and thereby increase the frequency of generating useful recombinants. We formulate the optimization as a sequentially-constrained hypergraph partitioning problem; the quality of recombinant libraries with respect to a set of breakpoints is characterized by the total perturbation to edge weights. We prove this problem to be NP-hard in general, but develop exact and heuristic polynomial-time algorithms for a number of important cases. Application to the beta-lactamase family demonstrates the utility of our algorithms in planning site-directed recombination.

Key words: directed evolution, dynamic programming, experiment planning, multi-order residue interactions, protein engineering.

1. INTRODUCTION

THE NON-RANDOM ASSOCIATION of amino acids, as expressed in pairwise potentials, has been usefully applied in a number of situations. Such pairwise contact potentials (Tanaka and Scheraga, 1976; Miyazawa and Jernigan, 1985) play a large role in evaluating the quality of models in protein structure prediction (Maiorov and Crippen, 1992; Simons et al., 1997; Kihara et al., 2001; Godzik, 2003). It has been suggested, however, that "it is unlikely that purely pairwise potentials are sufficient for structure prediction" (Betancourt and Thirumalai, 1999; Carter et al., 2001).

[1]Department of Computer Science, Dartmouth College, Hanover, New Hampshire.
[2]Markey Center for Structural Biology, Department of Biological Sciences and Purdue Cancer Center, Purdue University, West Lafayette, Indiana.

To better model evolutionary relationships that determine protein stability and functionality, it may be necessary to capture the higher-order interactions that are ignored in simple pairwise models (Fig. 1a). Researchers have begun to demonstrate the importance of accounting for higher-order terms. A statistical pseudo-potential based on four-body nearest neighbor interactions (as determined by Delaunay tessellations) has successfully predicted changes in free energy caused by hydrophobic core mutations (Carter et al., 2001). Similar formulations have been used to discriminate native from non-native protein conformations (Krishnamoorthy and Tropsha, 2003). Geometrically less restricted higher-order interactions have also been utilized for recognition of native-like protein structures (Simons et al., 1999). Recent work on correlated mutation analysis has moved from identifying pairwise correlations (Gobel et al., 1994) to determining clusters or cliques of mutually-dependent residues that identify subclasses within a protein family and provide mechanistic insights into function (Lockless and Ranganathan, 1999; Thomas et al., 2005).

This paper develops a rigorous basis for representing multi-order interactions within a protein family. We generalize the traditional representations of sequence information in terms of single-position conservation and structural interactions in terms of pairwise contacts. Instead, we define a hypergraph model in which edges represent pairwise and higher-order residue interactions, while edge weights represent the degree of "hyperconservation" of the interacting residues (see Section 2). Hyperconservation can reveal significant residue interactions both within members of the family (arising from structural and functional constraints) and generally common to all proteins (arising from general properties of the amino acids). We then combine family-specific and database-wide statistics with suitable weighting (see Section 2.1), ensure non-redundancy of the information in super- and sub-edges with a multi-order potential score (see Section 2.2), and derive edge weights by mean potential scores (see Section 2.3). Application of our approach to beta-lactamases (see Section 4) shows that the effect of non-redundant higher-order terms is significant and can be effectively handled by our model.

Protein recombination *in vitro* (Fig. 1b) enables the design of protein variants with favorable properties and novel enzymatic activities, as well as the exploration of protein sequence-structure-function relationships (Saftalov et al., 2006; Stemmer, 1994; Ostermeier et al., 1999; Lutz et al., 2001; Sieber et al., 2001; Voigt et al., 2002; O'Maille et al., 2002; Aguinaldo and Arnold, 2003; Coco, 2003; Castle et al., 2004). In this approach, libraries of hybrid proteins are generated either by stochastic enzymatic reactions or intentional selection of breakpoints. Hybrids with unusual properties can either be identified by large-scale genetic screening and selection, or many hybrids can be evaluated individually to determine detailed sequence-function relationships for understanding and/or rational engineering. We focus here on site-directed recombination, in which parent genes are recombined at specified breakpoint locations, yielding hybrids in which different sequence fragments (between the breakpoints) can come from different

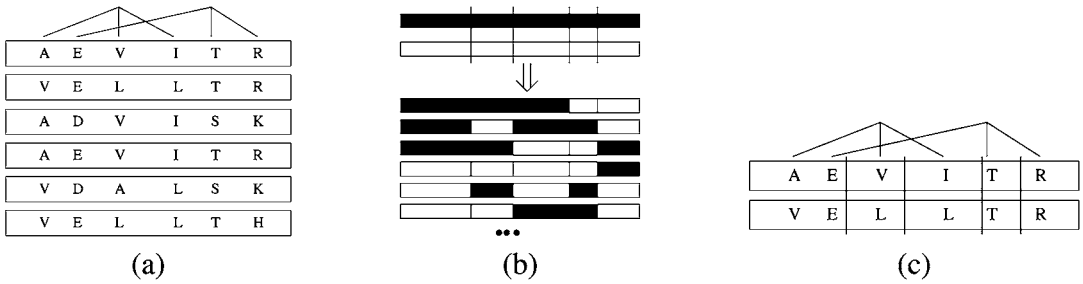

FIG. 1. Hypergraph model of evolutionary interactions, and effects of site-directed protein recombination. **(a)** Higher-order evolutionary interactions (here, order-3) determining protein stability and function are observed in the statistics of "hyperconservation" of mutually interacting positions. The left edge is dominated by Ala,Val,Ile and Val,Leu,Leu interactions, while the right is dominated by Glu,Thr,Arg and Asp,Ser,Lys ones. The interactions are modeled as edges in a hypergraph with weights evaluating the degree of hyperconservation of an interaction, both generally in the protein database and specific to a particular family. **(b)** Site-directed recombination experiments mix and match sequential fragments from homologous parents to construct a library of hybrids with the same basic structure but somewhat different sequences and thus different functions. **(c)** Site-directed recombination experiments perturb edges that cross one or more recombination breakpoints. The difference in edge weights derived for the parents and those derived for the hybrids indicates the effect of the perturbation on maintenance of the evolutionarily preserved interactions.

parents. Both screening/selection and individually evaluated experiments benefit from recombination that preserves the most essential structural and functional features while still allowing variation. In order to enhance the success of this approach, it is necessary to choose breakpoint locations that optimize preservation of these features.

The labs of Mayo and Arnold (Voigt et al., 2002; Meyer et al., 2003) have established criteria for non-disruption of contacting residue pairs and demonstrated the relationship between non-disruption and functional hybrids (Voigt et al., 2002). There is an on-going search for algorithms to select breakpoints for recombination based on non-disruption (Meyer et al., 2003; Endelman et al., 2004), although none has yet been experimentally validated. Optimizing retainment of multi-order interactions after recombination (Fig. 1c) should help identify the best recombinants and thus the best locations for breakpoints. In support of this optimization, we develop criteria to evaluate the quality of hybrid libraries by considering the effects of recombination on edge weights (see Section 2.4). We then formulate the optimal selection of breakpoint locations as a sequentially-constrained hypergraph partitioning problem (see Section 3), and prove it to be NP-hard in general (see Section 3.1). We develop exact and heuristic algorithms for a number of important cases (see Sections 3.2–3.5), and demonstrate their practical effectiveness in design of recombination experiments for members of the beta-lactamase family (see Section 4).

2. A HYPERGRAPH MODEL OF EVOLUTIONARY INTERACTIONS

In order to more completely model statistical interactions in a protein, it is necessary to generalize single-position sequence conservation and pairwise structural contact. We model a protein and its reference structure with a weighted hypergraph $G = (V, E, w)$, where vertices $V = \{v_1, v_2, \ldots, v_{|V|}\}$ represent residue positions in sequential order on the backbone, edges $E \subseteq 2^V$ represent mutually interacting sets of vertices, and weight function $w : E \to \mathbb{R}$ represents the relative significance of edges. We construct an order-c edge $e = \langle v_1, v_2, \ldots, v_c \rangle$ for each set of residues (listed in sequential order for convenience) that are in mutual contact; this construction can readily be extended to capture other forms of interaction, for example, long-range interaction of non-contacting residues due to electrostatics. Note that subsets of vertices associated with a higher-order edge form lower-order edges.

The definition of the edge weight is key to effective use of the hypergraph model. In the case where the protein is a member of a family with presumed similar structures, edge weights can be evaluated from the general database or a specific family. There are many observed residue values (across the family or database) for the vertices of any given edge. We thus build up to an edge weight by first estimating the probability of the residue values, then decomposing the probability to ensure non-redundant information among multi-order edges for the same positions. Finally, we determine the effect on the pattern of these values due to recombination according to a set of chosen breakpoint locations.

2.1. Distribution of hyperresidues in database and family

Let $R = \langle r_1, r_2, \ldots, r_c \rangle$ be a "hyperresidue," a c-tuple of amino acid types (e.g., $\langle$Ala, Val, Ile$\rangle$). Intuitively speaking, the more frequently a particular hyperresidue occurs in functional proteins, the more important it is expected to be for their folding and function. We can estimate the overall probability p of hyperresidues from their frequencies in the database $\mathcal{D}$ of protein sequences and corresponding structures:

$$p(R) = \frac{\#R \text{ in } \mathcal{D}}{|\mathcal{D}|}, \tag{1}$$

where $|\mathcal{D}|$ represents the total number of tuples of the same order in the database. When considering a specific protein family $\mathcal{F}$ with a multiple sequence alignment (MSA) and shared structure, we can estimate position-specific (i.e., for edge e) probability of a hyperresidue:

$$p_e(R) = \frac{\#R \text{ at } e \text{ in } \mathcal{F}}{|\mathcal{F}|}, \tag{2}$$

where $|\mathcal{F}|$ is the total number of tuples at positions forming edge e in the family MSA, i.e., the total number of sequences in the family MSA.

Estimation of probabilities from frequencies is valid only if the frequencies are large. Thus the general probability estimated from the whole database (Equation (1)) is more robust than the position-specific one from a single family (Equation (2)). However, family-specific information is more valuable as it captures the evolutionarily-preserved interactions in that family. To combine these two aspects, we adopt the treatment of sparse data sets proposed by Sippl (1990):

$$q_e(R) = \omega_1 \cdot p(R) + \omega_2 \cdot p_e(R), \tag{3}$$

but employing weights suitable for our problem:

$$\omega_1 = \frac{1}{1 + |\mathcal{F}|\rho} \quad \text{and} \quad \omega_2 = 1 - \omega_1, \tag{4}$$

where ρ is a user-specified parameter that determines the relative contributions of database and family. Note that when $\rho = 0$, $q_e(R) = p(R)$ and the family-specific information is ignored; whereas when $\rho = \infty$, $q_e(R) = p_e(R)$ and the database information is ignored. Using a suitable value of ρ, we will obtain a probability distribution that is close to the overall database distribution for a small family but approximates the family distribution for a large one.

2.2. Multi-order potential score for hyperresidues

Since we have multi-order edges, with lower-order subsets included alongside their higher-order supersets, we must ensure that these edges are not redundant. In other words, a higher-order edge should only include information not captured by its lower-order constituents. The inclusion-exclusion principle ensures non-redundancy in a probability expansion, as demonstrated in the case of protein structure prediction (Simons et al., 1999). We define an analogous multi-order potential score for hyperresidues at edges of orders 1, 2, and 3, respectively, as follows:

$$\phi_{v_i}(r_\alpha) = \log q_{v_i}(r_\alpha), \tag{5}$$

$$\phi_{v_i v_j}(r_\alpha r_\beta) = \log \frac{q_{v_i v_j}(r_\alpha r_\beta)}{q_{v_i}(r_\alpha) \cdot q_{v_j}(r_\beta)}, \tag{6}$$

$$\phi_{v_i v_j v_k}(r_\alpha r_\beta r_\gamma) = \log \frac{q_{v_i v_j v_k}(r_\alpha r_\beta r_\gamma) \cdot q_{v_i}(r_\alpha) \cdot q_{v_j}(r_\beta) \cdot q_{v_k}(r_\gamma)}{q_{v_i v_j}(r_\alpha r_\beta) \cdot q_{v_i v_k}(r_\alpha r_\gamma) \cdot q_{v_j v_k}(r_\beta r_\gamma)}. \tag{7}$$

Here, $\phi_{v_i}(r_\alpha)$ captures residue conservation at v_i; $\phi_{v_i v_j}(r_\alpha r_\beta)$ captures pairwise hyperconservation and is zero if v_i and v_j are not in contact or their residue types are completely independent; $\phi_{v_i v_j v_k}(r_\alpha r_\beta r_\gamma)$ captures 3-way hyperconservation and is zero if v_i, v_j, and v_k are not in mutual contact or their residue types are completely independent. The potential score of higher-order hyperresidues can be defined similarly. The potential score of a higher-order hyperresidue contains no information redundant with that of its lower-order constituents.

An alternative understanding of the hyperconservation score is as a measurement of over/underrepresentation of hyperresidues. Let $q'_e(R)$ be the probability of hyperresidue R at edge e assuming no order-$|R|$ conservation (there might be lower-order conservation). Then we can write the general definition of the hyperconservation score as follows,

$$\phi_e(R) = \log \frac{q_e(R)}{q'_e(R)}, \tag{8}$$

which includes Equations (5)–(7) as special cases.

2.3. Edge weights

In the hypergraph model, edge weights measure evolutionary optimization of higher-order interactions. For a protein or a set of proteins $\mathcal{S} \subseteq \mathcal{F}$, we can evaluate the significance of an edge as the average

potential score of the hyperresidues appearing at the positions forming the edge:

$$w(e) = \sum_R \frac{\#R \text{ at } e \text{ in } \mathcal{S}}{|\mathcal{S}|} \cdot \phi_e(R). \tag{9}$$

2.4. Edge weights for recombination

A particular form of edge weights serves as a guide for breakpoint selection in site-directed recombination. Suppose a set $\mathcal{S} \subseteq \mathcal{F}$ of parents is to be recombined at a set $X = \{x_1, x_2, \ldots, x_n\}$ of breakpoints, where $x_t = v_i$ indicates that breakpoint x_t is between residues v_i and v_{i+1}. We can view recombination as a two-step process: *decomposing* followed by *recombining*. In the decomposing step, each protein sequence is partitioned into $n+1$ intervals according to the breakpoints, and the hypergraph is partitioned into $n+1$ disjoint subgraphs by removing all edges spanning a breakpoint. The impact of this decomposition can be individually assessed for each edge, using Equation (9) for the parents $\mathcal{S}$.

In the recombining step, edges removed in the decomposing step are reconstructed with new sets of hyperresidues according to all combinations of parent fragments. The impact of this reconstruction can also be individually assessed for each edge, yielding a breakpoint-specific weight:

$$w(e, X) = \sum_R \frac{\#R \text{ at } e \text{ in } \mathcal{L}}{|\mathcal{L}|} \cdot \phi_e(R). \tag{10}$$

In this case, the potential score of hyperresidue R is weighted by the amount of its representation in the library $\mathcal{L}$. Note that we need not actually enumerate the set of hybrids (which can be combinatorially large) in order to determine the weight, as the frequencies of the residues at the positions are sufficient to compute the frequencies of the hyperresidues (see Section 3.6).

The combined effect of the two-step recombination process on an individual edge, the *edge perturbation*, is then defined as the change in edge weight:

$$\Delta w(e, X) = w(e) - w(e, X). \tag{11}$$

If all vertices of e are in one fragment, we have $w(e) = w(e, X)$ and $\Delta w(e, X) = 0$. The edge perturbation thus integrates essential information from the database, family, parent sequences, and breakpoint locations.

3. OPTIMIZATION OF BREAKPOINT LOCATIONS

Given parent sequences, a set of breakpoints determines a hybrid library. The quality of this hybrid library can be measured by the total perturbation to all edges due to the breakpoints. The hypothesis is that the lower the perturbation, the higher the representation of folded and functional hybrids in the library. We formulate the breakpoint selection problem as follows.

Problem 3.1. *c-RECOMB. Given $G_c = (V, E_c, w)$ and a positive integer n, choose a set of breakpoints $X = \{x_1, x_2, \ldots, x_n\}$ minimizing $\sum_{e \in E_c} \Delta w(e, X)$.*

where G_c represents a hypergraph with edge order uniformly c. Since lower-order edges can be regarded as a special kind of higher-order ones, G_c includes "virtual" lower-order edges.

This hypergraph partitioning problem is significantly more specific than general hypergraph partitioning, so it is interesting to consider its algorithmic difficulty. As we will see in Section 3.1, *c-RECOMB* is NP-hard for $c = 4$ (and thus also for $c > 4$), although we provide polynomial-time solutions for $c = 2$ in Section 3.2 and $c = 3$ in Section 3.4.

A special case of *c-RECOMB* with even more structure provides an efficient heuristic approach to minimize the overall perturbation. By minimizing the total weight of all edges removed in the decomposing step, fewer interactions need to be recovered in the recombining step.

Problem 3.2. *c-DECOMP. Given $G_c = (V, E_c, w)$ and a positive integer n, choose a set of breakpoints $X = \{x_1, x_2, \ldots, x_n\}$ minimizing $\sum_{e \in E_X} w(e)$, where $E_X \subseteq E_c$ is the set of edges spanning X.*

c-DECOMP could also be useful in identifying modular units in protein structures, in which case there is no recombining step.

3.1. NP-hardness of 4-RECOMB

4-RECOMB is combinatorial in the set X of breakpoints and the possible configurations they can take relative to each edge. The number of possible libraries could be huge even with a small number of breakpoints (e.g., choosing 7 breakpoints from 262 positions for beta-lactamase results in combinations at the order of 10^{13}). The choices made for breakpoints are reflected in whether or not there is a breakpoint between each pair of sequentially-ordered vertices of an edge, and thus in the perturbation to the edge. We first give a decision version of *4-RECOMB* as follows and then prove that it is NP-hard. Thus the related optimization problem is also NP-hard. Of course edges are not arbitrary in real protein structures; it remains interesting future work if the problem is still NP-hard in such "geometrically-constrained" situations.

Problem 3.3. 4-*RECOMB-DEC. Given* $G_4 = (V, E_4, w)$*, a positive integer* n*, and an integer* W*, does there exist a set of breakpoints* $X = \{x_1, x_2, \ldots, x_n\}$ *such that* $\sum_{e \in E_4} \Delta w(e, X) \leq W$.

Theorem 3.4. *4-RECOMB-DEC is NP-hard.*

Proof. We reduce from *3SAT*. Let $\phi = C_1 \wedge C_2 \wedge \cdots \wedge C_k$ be a boolean formula in 3-CNF with k clauses. We shall construct a hypergraph $G_4 = (V, E_4, w)$ such that ϕ is satisfiable iff there is a *4-RECOMB-DEC* solution for G_4 with $n = 3k$ breakpoints and $W = -|E_4|$ (Fig. 2). For clause $C_i = (l_{i,1} \vee l_{i,2} \vee l_{i,3})$ in ϕ, add to V four vertices in sequential order $v_{i,1}$, $v_{i,2}$, $v_{i,3}$, and $v_{i,4}$. Elongate V with $3k$ trivial vertices (v'_j in Fig. 2), where we can put trivial breakpoints that cause no perturbation. Let us define predicate $b(i, s, X) = v_{i,s} \in X$ for $s \in \{1, 2, 3\}$, indicating whether or not there is a breakpoint between $v_{i,s}$ and $v_{i,s+1}$. We also use indicator function I to convert a boolean value to 0 or 1. We construct E_4 with three kinds of edges: (1) For the 4-tuple of vertices for clause C_i, add an edge $e = \langle v_{i,1}, v_{i,2}, v_{i,3}, v_{i,4} \rangle$ with $\Delta w(e, X) = -I\{b(i, 1, X) \vee b(i, 2, X) \vee b(i, 3, X)\}$. (2) If two literals $l_{i,s}$ and $l_{j,t}$ are identical, add an edge $e = \langle v_{i,s}, v_{i,s+1}, v_{j,t}, v_{j,t+1} \rangle$ with $\Delta w(e, X) = -I\{b(i, s, X) = b(j, t, X)\}$. (3) If two literals $l_{i,s}$ and $l_{j,t}$ are complementary, add an edge $e = \langle v_{i,s}, v_{i,s+1}, v_{j,t}, v_{j,t+1} \rangle$ with $\Delta w(e, X) = -I\{b(i, s, X) \neq b(j, t, X)\}$.

There are $7k$ vertices and at most $k + 3\binom{k}{2} = O(k^2)$ edges, so the construction takes polynomial time. It is also a reduction. First, if ϕ has a satisfying assignment, choose breakpoints $X = \{v_{i,s} | l_{i,s}$ is TRUE$\}$ plus additional breakpoints between the trivial vertices to reach $3k$ total. Since each clause is satisfied, one of its literals is true, so there is a breakpoint in the corresponding edge e and its perturbation is -1. Since literals must be used consistently, type 2 and 3 edges also have -1 perturbation. Thus *4-RECOMB-DEC* is satisfied with $n = 3k$ and $W = -|E_4|$. Conversely, if there is a *4-RECOMB-DEC* solution with breakpoints X, then assign truth values to variables such that $l_{i,s} = b(i, s, X)$ for $s \in \{1, 2, 3\}$ and $i \in \{1, 2, \ldots, k\}$.

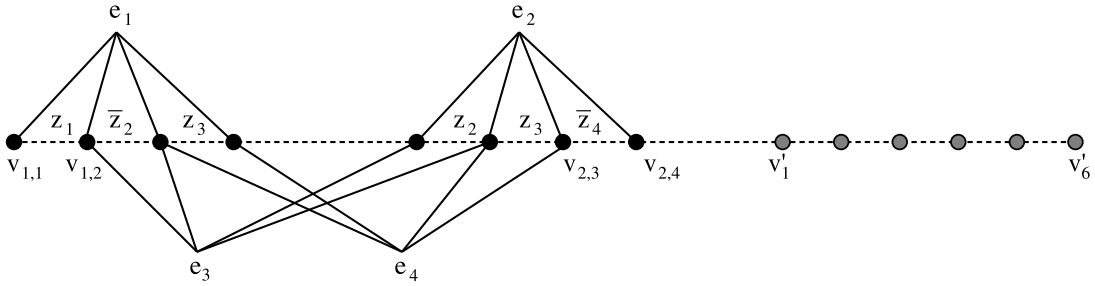

FIG. 2. Construction of hypergraph $G_4 = (V, E_4, w)$ from an instance of *3SAT* $\phi = (z_1 \vee \bar{z}_2 \vee z_3) \wedge (z_2 \vee z_3 \vee \bar{z}_4)$. Type 1 edges e_1 and e_2 ensure the satisfaction of clauses (-1 perturbation iff there is a breakpoint iff the literal is true and the clause is satisfied), while type 3 edge e_3 and type 2 edge e_4 ensure the consistent use of literals (-1 perturbation iff the breakpoints are identical or complementary iff the variable has a single value).

Since perturbation to type 1 edges is -1, there must be at least one breakpoint in each clause vertex tuple, and thus a true literal in the clause. Since perturbation to type 2 and 3 edges is -1, literals are used consistently. ■

We note that *4-RECOMB-DEC* is in NP, since given a set of breakpoints X for parents $\mathcal{S}$ we can compute $\Delta w(e, X)$ for all edges in polynomial time ($O(\mathcal{S}^4 E)$), and then must simply sum and compare to a provided threshold.

3.2. Dynamic programming framework

Despite the NP-hardness of the general sequentially-constrained hypergraph partitioning problem *c-RECOMB*, the structure of the problem (i.e., the sequential constraint) leads to efficient solutions for some important cases. Suppose we are adding breakpoints one by one from left to right (N- to C-terminal) in the sequence. Then the additional perturbation to an edge e caused by adding breakpoint x_t given previous breakpoints $X_{t-1} = \{x_1, x_2, \ldots, x_{t-1}\}$ can be written:

$$\Delta\Delta w(e, X_{t-1}, x_t) = \Delta w(e, X_t) - \Delta w(e, X_{t-1}), \tag{12}$$

where $X_0 = \emptyset$ and the additional perturbation caused by the first breakpoint is $\Delta\Delta w(e, X_0, x_1) = \Delta w(e, X_1)$. Reusing notation, we use $\Delta\Delta w(E, X_{t-1}, x_t)$ to indicate the total additional perturbation to all edges. Now, if the value of $\Delta\Delta w(E, X_{t-1}, x_t)$ can be determined by the positions of x_{t-1} and x_t, independent of previous breakpoints, then we can adopt the dynamic programming approach shown below. When the additional perturbation depends only on x_{t-1} and x_t, we write it as $\Delta\Delta w(E, x_{t-1}, x_t)$ to indicate the restricted dependence.

Let $d[t, \tau]$ be the minimum perturbation caused by t breakpoints with the rightmost at position τ. If, for simplicity, we regard the right end of the sequence as a trivial breakpoint that causes no perturbation, then $d[n+1, |V|]$ is the minimum perturbation caused by n breakpoints plus this trivial one, i.e. the objective function for Problem 3.1. We can compute d recursively:

$$d[t, \tau] = \begin{cases} \Delta w(E, \{\tau\}), & \text{if } t = 1; \\ \min\limits_{\lambda \le \tau - \delta} \{d[t-1, \lambda] + \Delta\Delta w(E, \lambda, \tau)\}, & \text{if } t \ge 2. \end{cases} \tag{13}$$

where δ is a user-specified minimum sequential distance between breakpoints. The recurrence can be efficiently computed bottom-up in a dynamic programming style, due to its optimal substructure. In the following, we instantiate this dynamic programming formulation with different forms of $\Delta\Delta w$ for different cases of *c-RECOMB* and *c-DECOMP*.

The special case of *2-DECOMP* (disruption of pairwise interactions) has been previously solved as a shortest path problem (Endelman et al., 2004). A complexity analysis accounting for both the edge weight calculation and dynamic programming shows that the total time is $O(\mathcal{S}E + VE + nV^2)$ (see Section 3.6).

The instantiation for *2-RECOMB* is as follows. Each order-2 edge $\langle v_i, v_j \rangle$ has two states: either there is breakpoint between v_i and v_j or not (Fig. 3). The state of e is changed by adding breakpoint x_t iff $x_{t-1} < v_i < x_t < v_j$. Thus the additional perturbation caused by adding x_t can be determined by the positions of x_{t-1} and x_t, and is independent of previous breakpoints. Our dynamic programming framework Equation (13) is therefore applicable to 2-*RECOMB*; the time complexity is $O(\mathcal{S}^2 E + VE + nV^2)$ (see Section 3.6).

3.3. Reduction from *c*-DECOMP to 2-DECOMP

A significant property of our multi-order potential score (see Section 2.2) is that the score of a higher-order edge captures only higher-order hyperconservation and contains no information about its lower-order constituents. Thus in the decomposition phase, a higher-order edge is broken if there is a breakpoint *anywhere* in the set of residue positions it spans. The lack of breakpoints between any adjacent pair of its vertices will be captured by the weight of the appropriate lower-order constituent edge. By this reasoning, we can reduce the *c-DECOMP* problem to the *2-DECOMP* problem: given hypergraph $G_c = (V_c, E_c, w_c)$,

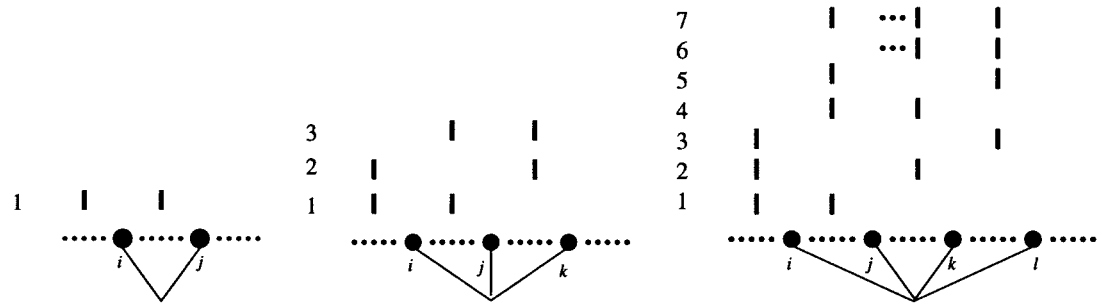

FIG. 3. All breakpoint configurations that cause additional perturbation to an edge as breakpoints are added one by one from left to right in the sequence. The dynamic programming formulation requires that we be able to distinguish these configurations from each other and from configurations with no additional perturbation. For an order-2 edge $\langle v_i, v_j \rangle$, there is additional perturbation if and only if the current breakpoint (right bar) is added between v_i and v_j and the previous breakpoint (left bar) is to the left of v_i. Similarly, the configurations on an order-3 edge $\langle v_i, v_j, v_k \rangle$ can be distinguished by the positions of the current breakpoint and the preceding one with respect to the intervals $[v_i, v_j]$ and $[v_j, v_k]$. However, for an order-4 edge, configurations 6 and 7 are ambiguous with respect to the intervals of $\langle v_i, v_j, v_k, v_l \rangle$. We cannot be certain about the (non-)existence of a breakpoint between v_i and v_j without potentially looking back at all previous breakpoints (dots).

construct graph $G_2 = (V_2, E_2, w_2)$ such that $V_2 = V_c$ and each edge $e_c = \langle v_1, v_2, \ldots, v_c \rangle \in E_c$ is mapped to an edge $e_2 = \langle v_1, v_c \rangle \in E_2$ connecting the first and last vertex of e_c, putting weight $w_c(e_c)$ on $w_2(e_2)$. There is a breakpoint decomposing e_c in G_c iff there is one decomposing e_2 in G_2. G_2 can be constructed in $O(V + E)$ time, and optimal solutions for *c-DECOMP* on G_c correspond to optimal solutions for *2-DECOMP* on G_2. Under this reduction (which adds only $O(E)$ computation), the total time complexity for *c-DECOMP* is $O(\mathcal{S}E + VE + nV^2)$ (see Section 3.6). Thus protein modules can be computed under *c-DECOMP* in polynomial time for any order of edge.

3.4. Dynamic programming for 3-RECOMB

We have seen that the *c-RECOMB* problem is NP-hard when $c \geq 4$ (see Section 3.1) and solvable in polynomial time when $c = 2$ (see Section 3.2). In this section, we instantiate our dynamic programming framework to give a polynomial-time solution when $c = 3$.

An order-3 edge has four possible states, according to whether or not there is at least one breakpoint between each pair of its vertices listed in sequential order. As Figure 3 illustrates, given only x_{t-1} and x_t, all breakpoint configurations that cause additional perturbation can be uniquely determined, and the additional perturbation can be computed as in Equation (12). This edge perturbation calculation meets the restriction required for our dynamic programming framework, and Equation (13) and be used to optimize *3-RECOMB* in $O(\mathcal{S}^3 E + VE + nV^2)$ time (see Section 3.6).

3.5. Stochastic dynamic programming for 4-RECOMB

Tetrahedra are natural building blocks of 3D structures, and Delaunay tetrahedra in the protein core have been shown to capture interactions important for protein folding (Carter et al., 2001). Our results below show significant information in general order-4 hyperconservation. In order to solve *4-RECOMB* problems, we develop here a heuristic approach based on stochastic dynamic programing. Unlike *2-RECOMB* and *3-RECOMB*, the additional perturbation of a breakpoint cannot always be determined by reference just to the current and previous breakpoint locations. As Figure 3 shows, given x_{t-1} and x_t, there is ambiguity only between configurations 6 and 7.

We can still employ the dynamic programming framework if we move from a deterministic version, in which both the additional perturbation and next state are known, to a stochastic version, in which they are predicted as expected values. In the ambiguous case of configurations 6 and 7 with $t \geq 2$, let us assume that breakpoints before x_{t-1} are uniformly distributed in the sequence. Then the probability of finding no

breakpoint between v_i and v_j (i.e., being in configuration 6 rather than 7) is

$$p = \left(1 - \frac{v_j - v_i}{x_{t-1}}\right)^{t-2}, \tag{14}$$

since $\frac{v_j - v_i}{x_{t-1}}$ is the probability of a breakpoint being located between v_i and v_j and $t-2$ is the number of breakpoints before position x_{t-1}. Therefore, for the ambiguous cases, the expected additional perturbation to e caused by adding x_t is

$$\Delta\Delta w(e, x_{t-1}, x_t, t) = p \cdot \Delta\Delta w_6(e, X_{t-1}, x_t) + (1-p) \cdot \Delta\Delta w_7(e, X_{t-1}, x_t), \tag{15}$$

where the subscript of $\Delta\Delta w$ indicates the configuration. Note that, unlike our previous formulations, the additional perturbation depends on the number of previous breakpoints. Thus, the time complexity of this stochastic dynamic programming is increased to $O(\mathcal{S}^4 E + nVE + nV^2)$ (see Section 3.6). This stochastic dynamic programming technique can also be applied to $c > 4$ *c-RECOMB* problems, but the effectiveness of the approximation is expected to decrease with an increasing number of ambiguous states.

3.6. Time complexity analysis

Since *c-DECOMP* is a special case of *c-RECOMB* and is always easier, we focus on the time complexity of *c-RECOMB*. The hyperresidue potential score $\phi_e(R)$ needs to be computed only once for each family and requires time polynomial in the size of the database and family. Thus we assume that $\phi_e(R)$ is precomputed before breakpoint selection. Then the time complexity of the dynamic programming algorithms includes three parts:

1. Computation of $\Delta w(e, X)$ for all edges. Although $w(e, X)$ is defined over the whole library, we do not really need to enumerate all hybrids because the number of hyperresidues is bounded by $O(\mathcal{S}^c)$, when all vertices in an order-c edge are separated by breakpoints and can combine freely. Since each combination shows up exactly the same number of times in the library, the time complexity of computing $w(e, X)$ is $O(\mathcal{S}^c)$. In fact, this bound can be improved if $|\mathcal{S}| > 20$. We first count the frequencies of residues at each position in $O(\mathcal{S})$ time, and then compute the frequencies of hyperresidues in $O(20^c)$ time, thereby improving the time complexity of computing $w(e, X)$ to a total of $O(\mathcal{S} + 20^c)$. Doing either of these computations for all edges results in total time of $O(\min\{\mathcal{S}^c E, (\mathcal{S} + 20^c)E\})$. For *2-DECOMP*, we only need to compute $w(e)$, which takes $O(\mathcal{S}E)$ time.
2. Computation of $\Delta\Delta w(E, \lambda, \tau)$ for $\lambda \in \{1, 2, \ldots, |V| - \delta\}$ and $\tau \in \{\delta + 1, \delta + 2, \ldots, |V|\}$. In a naive approach, we can check which edges are broken for each combination of λ and τ, and compute the additional perturbation; this takes time $O(V^2 E)$. This time complexity can be improved to $O(VE)$ as follows. For each position of λ, first compute $\Delta\Delta w(E, \lambda, \lambda + \delta)$, in $O(E)$ time. Then in sweeping from $\lambda + \delta$ to $|V|$, the state of an edge e will be changed at most $2^{c-1} - 1$ times. Thus $\Delta\Delta w(E, \lambda, \tau)$ for $\tau \in \{\lambda + \delta, \ldots, |V|\}$ can be computed in total $O(E)$ time in an incremental manner by adjusting it at each step for those edges whose weight changes from τ to $\tau + 1$. Thus the total time complexity is $O(VE)$ for *2-RECOMB* and *3-RECOMB*, and $O(nVE)$ for the stochastic dynamic programming of *4-RECOMB* since we need to compute the additional perturbation for $t \in \{2, 3, \ldots, n\}$.
3. Computation of $d[n+1, |V|]$ in Equation (13). We must compute $d[t, \tau]$ for $t \in \{1, 2, \ldots, n, n+1\}$ and $\tau \in \{\delta + 1, \delta + 2, \ldots, |V|\}$. Each evaluation takes $O(V)$ time to choose the minimum, so the time complexity is $O(nV^2)$.

Therefore, the time complexity of dynamic programming for *c-RECOMB* is $O(\mathcal{S}^c E + VE + nV^2)$ for $2 \le c \le 3$ and $O(\mathcal{S}^c E + nVE + nV^2)$ for $c = 4$, where the first term in each can be improved to $O((\mathcal{S} + 20^c)E)$ for large $\mathcal{S}$.

4. RESULTS

We demonstrate our hypergraph model and recombination planning algorithms in analysis of the beta-lactamase protein family, since previous site-directed recombination experiments have employed

beta-lactamase parents TEM-1 and PSE-4 (Meyer et al., 2003). We identified 136 beta-lactamases for $\mathcal{F}$, including TEM-1 and PSE-4, with no more than 80% sequence identity, and constructed a multiple sequence alignment with at most 20% gaps in any sequence. PDB file 1BTL was used as the representative family structure. Vertices were considered as located at the average position of non-hydrogen side-chain atoms (C^{α} atoms not included except for Glycines), and edges formed for sets of vertices whose positions were within 8 Å of each other.

For the database $\mathcal{D}$, we started with a subset of sequences culled from the protein data bank according to structure quality (R-factor less than 0.25) and mutual sequence identity (at most 60%) by *PISCES* (Wang and Dunbrack, 2003). To minimize the effect of structural errors on statistical results, chains with nonconsecutive residue numbers, gaps (C^{α}-C^{α} distance greater than 4.2 Å between consecutive residues), or incorrect atom composition of residues were excluded (Carter et al., 2001). This left 687 chains. Contact maps were constructed as with the family.

We first considered the information content in higher-order interactions. Figure 4 shows the distributions of hyperresidue potential scores in both the database and family, for increasing hyperresidue order. By the non-redundant decomposition, a higher-order potential score would be 0 if the lower-order terms were independent. Non-zero scores represent positive and negative correlation. The figure shows that there is clearly information in the higher-order constraints. Note that the family distribution is biased (μ not at zero) because many sets of amino acid types are not observed in the MSA. It is also more informative (larger σ, with those for higher-order interactions comparable to that for pairwise interactions). Dicysteine pairs are expected to be informative (i.e., cysteines are not independent), and they are clear outliers marked in the $c = 2$ database potential.

A limited amount of data is currently available for evaluating the experimental effectiveness of the hyperconservation score and a recombination plan. Here, we use the beta-lactamase hybrid libraries of Meyer et al. (2003) and Hiraga and Arnold (2003). For each hybrid in a library, we computed both the total potential score and the mutation level. The total potential score is the sum, over all edges up to order-4, of the edge potential (Equations (5)–(7)) for the residues in the hybrid sequence. The mutation level is the number of residues in the hybrid that are different from the closest parent. Hybrids with small

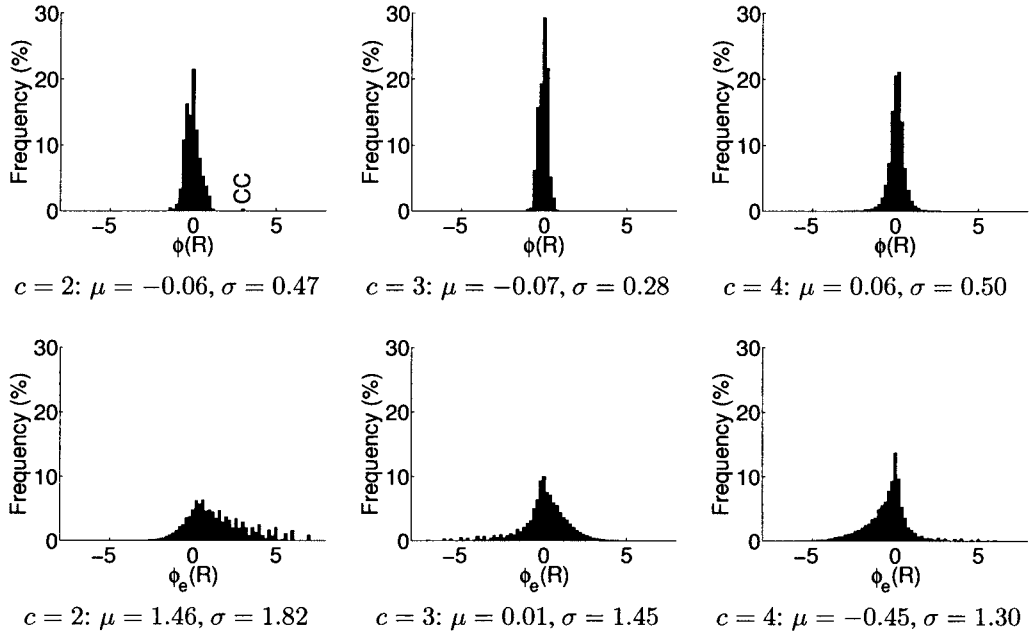

FIG. 4. Multi-order potential scores, derived from the database (top) and the beta-lactamase family (bottom). For each order c of hyperresidues, the distribution of potential scores among bins of size 0.2 is shown (pooled over all edges for the family version). Base 2 logarithm is used for computing potential scores.

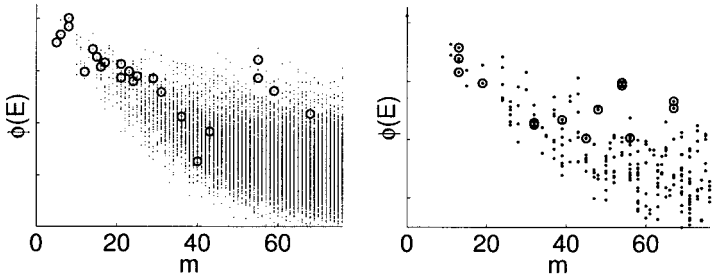

FIG. 5. Potential score $\phi(E)$ (sum over all interactions up to order-4) versus mutation level m (to the closest parent) for all hybrids in a beta-lactamase library with (left) 13 breakpoints (Meyer et al., 2003) and (right) 7 breakpoints (Hiraga and Arnold, 2003). Dots indicate hybrids, and circles those determined to be functional.

mutation levels are expected to be functional. Figure 5 shows that, especially for high mutation levels, the hybrids with better potential scores are enriched in measured functional activity.

Next we applied our dynamic programming algorithms to optimize 7-breakpoint sets for different beta-lactamase parents (Fig. 6), using minimum effective fragment length $\delta = 10$, database/family weight $\rho = 0.01$, and maximum order of edges $c = 3$. We found the results to be insensitive to ρ, beyond very small values placing all the emphasis on the database (data not shown). In the 1-parent case, the plan amounts to decomposing the protein (PDB file 1BTL as representative family structure) into modules preserving multi-order interactions. The 2-parent and 12-parent cases illustrated here would be useful in site-directed recombination experiments. We note that some locations can "float" due to parent sequence identity (e.g., in positions 17–20 with 2 parents). These all represent viable experiment plans, optimizing multi-order interactions according to sequence characteristics of different parents.

Finally, we considered the error that could be caused by the stochastic approximation in solving *4-RECOMB*. Figure 7 shows the distribution, over all order-4 edges, of differences in perturbations between the ambiguous states. The differences are expressed in terms of perturbation standard deviations $\varepsilon = \frac{|\Delta\Delta w_6 - \Delta\Delta w_7|}{(std(\Delta\Delta w_6) + std(\Delta\Delta w_7))/2}$. Edges with identical residues at v_i or v_j are excluded, since the perturbation is necessarily the same. Even so, in a majority of cases the heuristic would lead to no or very small error. Thus the stochastic dynamic programming will provide a near optimal solution, which makes it reasonable to include 4-way interactions in practice.

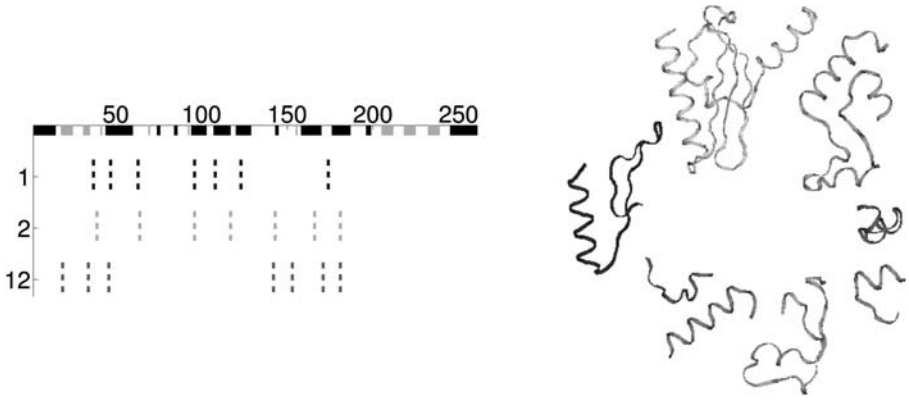

FIG. 6. **(Left)** Optimized breakpoint locations for beta-lactamase when planning with 1, 2, or 12 parents. The sequence is labeled with residue index, with helices in black and β-sheets in gray. **(Right)** Fragments of beta-lactamase in 3D structure (PDB id: 1BTL) according to optimized breakpoint locations for the 1-parent case.

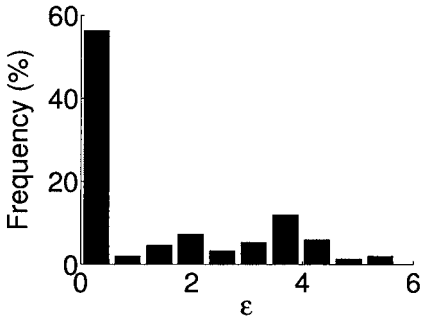

FIG. 7. Distribution of differences in edge perturbations in ambiguous *4-RECOMB* cases. The differences are expressed in terms of perturbation standard deviations ε.

5. DISCUSSION

We have developed a general hypergraph model of multi-order residue interactions in proteins, along with algorithms that use the model to optimize site-directed recombination experiments. Our multi-order potential provides a natural means for identifying and representing conservation across sets of residues. Our experiment planning algorithms take advantage of the structure of this potential, along with the sequential constraint imposed by recombination experiments, in order to efficiently determine optimal sets of breakpoints maintaining important interactions.

Our model of multi-order interactions may be productively applied to other problems, after suitable parameterization. For example, our multi-order potential generalizes the four-body interactions employed in prediction of ΔG° of unfolding (Carter et al., 2001), and may prove useful in prediction of stability mutagenesis. To apply our approach to functionality mutagenesis, it may be necessary to separate and appropriately weight the contributions to stability and functionality from the multi-residue interactions (e.g., accounting for relationships with known functional sites). The hyperedges and their weights are by no means limited to spatially proximate sets of residues, and non-contacting interactions may also be usefully incorporated for these contexts. Indeed, studies of pairwise residue coupling have identified many important non-contacting relationships (Lockless and Ranganathan, 1999; Thomas et al., 2005). Finally, as we illustrate in Figure 6, multi-order interactions may also be applied to identify modular units of protein structure (finer-grained than domains). Optimization of breakpoints for modularity may require a potential that appropriately balances intra-fragment interactions with inter-fragment interactions.

The design of an optimal library for recombination must account for and balance multiple criteria of optimality. Various approaches have been explored such as making trade-off between the diversity and activity of all hybrids and the best hybrid in a library (Ostermeier, 2003), or minimizing the average number of clashes per hybrid (Saraf et al., 2005). Our approach focuses on obtaining a high representation of folded and functional hybrids, by preserving significant interactions observed in the family and database. We take the diversity of hybrids into account only indirectly, by limiting the minimum effective fragment length. To more directly account for diversity, it may be helpful to weight our potential to provide varying amounts of freedom to maintain or perturb different interactions. Once we have learned the rules, we know how to break them. Alternatively, a planning algorithm may keep these aspects separate in a multi-dimensional optimization.

We have assumed that parent sequences are given and focused on breakpoint selection by minimizing hyperconservation perturbation. The influence of parent sequences on library quality has been studied by optimizing breakpoints for each possible set of parents (Endelman et al., 2004). Such an exhaustive approach will not scale beyond small problems. Since our potential score integrates essential information from the database, family, parent sequences, and breakpoint locations, it can naturally be exploited for parent sequence selection. Extended planning algorithms may simultaneously optimize parent sequences and breakpoint locations.

Finally, after the planned recombination experiments have been conducted, we may desire to improve the model according to consistency with experimental data. Some interactions determined to be important

from the database and family information may prove to be highly conserved in the folded, functional hybrids, while some may have more flexibility. An improved model can then be used in subsequent rounds of planning.

ACKNOWLEDGMENTS

We thank Dr. Bruce Craig (Statistics, Purdue), Shobha Potluri and John Thomas (CS, Dartmouth), and Michal Gajda (IIMCB, Poland) for stimulating discussion. This work was supported in part by an NSF Career Award (to C.B.K., IIS-0444544) and an NSF SEIII grant (to C.B.K., A.M.F., and Bruce Craig, IIS-0502801).

REFERENCES

Aguinaldo, A., and Arnold, F. 2003. Staggered extension process (StEP) in vitro recombination. *Methods Mol. Biol.* 231, 105–110.

Betancourt, M., and Thirumalai, D. 1999. Pair potentials for protein folding: choice of reference states and sensitivity of predictive native states to variations in the interaction schemes. *Protein Sci.* 8, 361–369.

Carter, Jr., C., LeFebvre, B., Cammer, S., et al. 2001. Four-body potentials reveal protein-specific correlations to stability changes caused by hydrophobic core mutations. *J. Mol. Biol.* 311, 621–638.

Castle, L., Siehl, D., Gorton, R., et al. 2004. Discovery and directed evolution of a glyphosate tolerance gene. *Science* 304, 1151–1154.

Coco, W. 2003. RACHITT: gene family shuffling by random chimeragenesis on transient templates. *Methods Mol. Biol.* 231, 111–127.

Endelman, J., Silberg, J., Wang, Z.-G., et al. 2004. Site-directed protein recombination as a shortest-path problem. *Protein Eng.* 17, 589–594.

Gobel, U., Sander, C., Schneider, R., et al. 1994. Correlated mutations and residue contacts in proteins. *Proteins* 18, 309–317.

Godzik, A. 2003. Fold recognition methods. *Methods Biochem. Anal.* 44, 525–546.

Hiraga, K., and Arnold, F. 2003. General method for sequence-independent site-directed chimeragenesis. *J. Mol. Biol.* 330, 287–296.

Kihara, D., Lu, H., Kolinski, A., et al. 2001. TOUCHSTONE: an *ab initio* protein structure prediction method that uses threading-based tertiary restraints. *Proc. Natl. Acad. Sci. USA* 98, 10125–10130.

Krishnamoorthy, B., and Tropsha, A. 2003. Development of a four-body statistical pseudo-potential to discriminate native from non-native protein conformations. *Bioinformatics* 19, 1540–1548.

Lockless, S., and Ranganathan, R. 1999. Evolutionarily conserved pathways of energetic connectivity in protein families. *Science* 286, 295–299.

Lutz, S., Ostermeier, M., Moore, G., et al. 2001. Creating multiple-crossover DNA libraries independent of sequence identity. *Proc. Natl. Acad. Sci. USA* 98, 11248–11253.

Maiorov, V., and Crippen, G. 1992. Contact potential that recognizes the correct folding of globular proteins. *J. Mol. Biol.* 227, 876–888.

Meyer, M., Silberg, J., Voigt, C., et al. 2003. Library analysis of SCHEMA-guided protein recombination. *Protein Sci.* 12, 1686–1693.

Miyazawa, S., and Jernigan, R. 1985. Estimation of effective interresidue contact energies from protein crystal structures: quasi-chemical approximation. *Macromolecules* 18, 531–552.

O'Maille, P., Bakhtina, M., and Tsai, M. 2002. Structure-based combinatorial protein engineering (SCOPE). *J. Mol. Biol.* 321, 677–691.

Ostermeier, M. 2003. Synthetic gene libraries: in search of the optimal diversity. *Trends Biotechnol.* 21, 244–247.

Ostermeier, M., Shim, J., and Benkovic, S. 1999. A combinatorial approach to hybrid enzymes independent of DNA homology. *Nature Biotechnol.* 17, 1205–1209.

Saftalov, L., Smith, P., Friedman, A., et al. 2006. Site-directed combinatorial construction of chimaeric genes: general method for optimizing assembly of gene fragments. *Proteins* 64, 629–642.

Saraf, M.C., Gupta, A., and Maranas, C.D. 2005. Design of combinatorial protein libraries of optimal size. *Proteins* 60, 769–777.

Sieber, V., Martinez, C., and Arnold, F. 2001. Libraries of hybrid proteins from distantly related sequences. *Nature Biotechnol.* 19, 456–460.

Simons, K., Ruczinski, I., Kooperberg, C., et al. 1999. Improved recognition of native-like protein structures using a combination of sequence-dependent and sequence-independent features of proteins. *Proteins* 34, 82–95.

Simons, K.T., Kooperberg, C., Huang, E., et al. 1997. Assembly of protein tertiary structures from fragments with similar local sequences using simulated annealing and Bayesian scoring functions. *J. Mol. Biol.* 268, 209–225.

Sippl, M. 1990. Calculation of conformational ensembles from potentials of mean force. an approach to the knowledge-based prediction of local structures in globular proteins. *J. Mol. Biol.* 213, 859–883.

Stemmer, W. 1994. Rapid evolution of a protein *in vitro* by DNA shuffling. *Nature* 370, 389–391.

Tanaka, S., and Scheraga, H. 1976. Medium and long range interaction parameters between amino acids for predicting three dimensional structures of proteins. *Macromolecules* 9, 945–950.

Thomas, J., Ramakrishnan, N., and Bailey-Kellogg, C. 2005. Graphical models of residue coupling in protein families. *5th ACM SIGKDD Workshop Data Mining Bioinform. (BIOKDD).*

Voigt, C., Martinez, C., Wang, Z., et al. 2002. Protein building blocks preserved by recombination. *Nature Struct. Biol.* 9, 553–558.

Wang, G., and Dunbrack Jr., R.L. 2003. Pisces: a protein sequence culling server. *Bioinformatics* 19, 1589–1591.

Address reprint requests to:
Dr. Chris Bailey-Kellogg
6211 Sudikoff Laboratory
Dartmouth College
Hanover, NH 03755

E-mail: cbk@cs.dartmouth.edu

or

Dr. Alan M. Friedman
Lilly Hall
Purdue University
West Lafayette, IN 47907

E-mail: afried@purdue.edu