# Mixed-Integer Linear Programming Algorithm for a Computational Protein Design Problem

**Yushan Zhu***

*Department of Chemical Engineering, Tsinghua University, Beijing 100084, P.R.China*

The computational protein design problem or the side-chain positioning problem is a central part in computational methods for predicting protein structure and designing protein sequences. However, the computational protein design problem is NP-complete, and it is also NP-complete to find a reasonable approximate solution to this problem. Here, we present a critical finding that the network flow structure embedded in the integer linear programming formulation of the computational protein design problem makes it equivalent to a mixed-integer linear programming formulation with fewer binary variables. This novel formulation effectively reduces the combinatorial difficulty of the computational protein design problem, thereby allowing the sequence selection of the global minimum energy conformation for large proteins to become tractable by using the standard optimization algorithms. Our preliminary calculation results for 20 core redesigned proteins show that the mixed-integer linear programming algorithm with a validated physical model is faster than its integer linear programming counterpart by up to 3 times. More importantly, the specific mathematical structure underlying the mixed-integer linear programming formulation paves the way for developing even faster combinatorial searching strategies by using stronger cutting planes for mixed-integer programming.



## Introduction

The computational protein design problem[1] involves finding an amino acid sequence compatible with a three-dimensional scaffold. In a common formulation of the problem, the backbone is fixed, side-chain types and conformations are chosen from a discrete rotamer library, and a pairwise energy function is optimized with the assumption that the global minimum energy sequence and conformation corresponds to the protein natural evolution for stability and activity. If all rotamers at each residue position come from a single amino acid, the computational protein design problem is simplified into a side-chain positioning problem.[2,3] The side-chain positioning problem plays a very important role in the study of the homology modeling and protein folding.[4,5] During the past decade, the computational protein design problem has become a fruitful field of endeavors, which is of particular interest to the biotechnology and pharmaceutical industries. Successful case studies include enhanced thermostability, catalytic activity, and binding of small molecules.[1,6−8] Recently, the computational protein design problem has stimulated much interest in the chemical engineering community as it provides great research challenges, opportunities, and synergism for both systems engineering and computational biology.[9−13]

Desmet et al.[14] proposed the first global optimization algorithm based on the dead-end elimination (DEE) theorem for the side-chain positioning problem; thereafter, Dahiyat and Mayo[15] extended this algorithm into solving the computational protein design problem. Despite its apparent simplicity, DEE is very efficient in practice.[1,6] Its efficiency deteriorates when more rotamers are encountered and more accurate but sophisticated energy functions are used. In order to improve its efficiency, a lot of variants are proposed, such as the Goldstein DEE,[16] the split DEE,[17,18] or its implementation within a branch-and-bound framework.[19,20] Nonetheless, the single DEE based combinatorial searching methods always cannot converge on the global minimum and this is when extended DEE for rotamer pairs has to be run.[13,14] The result is that the search time becomes too long even with parallel operations. Such typical difficulty of the computational protein design problem comes from its NP-hard complexity,[21] and it is NP-complete to find even a reasonable approximate solution to this problem by any approximate algorithm that guarantees that its solution is within some factor of optimality.[22] In order to tackle this problem, the computational protein design problem is reformulated into an integer linear programming problem (ILP)[23−25] with the hope that mature mathematical programming techniques for integer linear programming can handle large problem sizes.[26] To our surprise, Kingsford et al.[25] found that when positioning side chains on native and homologous backbones, optimal solutions of the ILP using a simple energy function can usually be found using its linear programming (LP) relaxation. On the other hand, the computational protein design problem cannot be solved using LP directly. It is mysterious why an ILP for a side-chain positioning problem can be solved by its LP relaxation problem. And, why can an ILP for a computational protein design problem with so many binary variables be pruned so efficiently? These questions strongly encouraged us to extricate this paradox, and if we can find the specific mathematical structure underlying the ILP, it is very likely that we can extend the mathematical programming techniques to computational protein design problems of large sizes.

In this paper, we present a critical finding that the network flow structure embedded in the ILP makes it equivalent to a mixed-integer linear programming (MILP) formulation with fewer binary variables. This novel formulation answers the aforementioned two questions since most of the binary variable in the ILP formulation can be relaxed into continuous ones, and this effectively reduces the combinatorial difficulty of the computational protein design problem, thereby allowing the sequence selection of the global minimum energy conformation for large proteins to become tractable by using standard optimization algorithms.

* To whom correspondence should be addressed. Tel.: +86-10-62784572 (O). Fax: +86-10-62770304 (O). E-mail: yszhu@tsinghua.edu.cn.

10.1021/ie0605985 CCC: $37.00 
Published on Web 12/21/2006

## Method

**Computational Protein Design Problem.** The computational protein design problem can be stated as follows:[14] Given a fixed template, which consists of the protein backbone and the side chains of residue positions not to be optimized, with $p$ optimized residue side chains, each optimized position $i$ is associated with a set of candidate rotamers $\{i_j\}$ coming from the same or different amino acids. Once a particular rotamer at each residue position has been chosen, the potential energy of a protein system can be written as[14]

$$E = E_0 + \sum_{i=1}^{p} E(i_j,i_j) + \sum_{i=1}^{p-1} \sum_{k=i+1}^{p} E(i_j,k_s) \tag{1}$$

where $E_0$ is the self-energy of the template, $E(i_j,i_j)$ is the energy resulting from the interaction between the template and the chosen rotamer $i_j$ at position $i$ as well as the intrinsic energy of rotamer $i_j$, and $E(i_j,k_s)$ stands for the pairwise interaction energy between chosen rotamers $i_j$ and $k_s$. In this discretized settings, the placement of each side chain is reduced to finding an assignment of rotamers to positions that achieves the global minimum energy conformation (GMEC) of the system.

**Dead-End Elimination Theorem.** The dead-end elimination theorem based combinatorial searching methods and their heuristics can greatly reduce the searching complexity of the computational protein design problem.[14,16,17] In this paper, the Goldstein DEE criterion[16] is run as a filter for the mixed-integer linear programming algorithm, which says that a rotamer $i_j \in V_i$, where $V_i$ includes all rotamers at position $i$, can be eliminated if there is another rotamer $i_u \in V_i$ such that[16]

$$E(i_j,i_j) - E(i_u,i_u) + \sum_{\substack{k=1 \\ k\neq i}}^{p} \min_{k_s \in V_k} \{E(i_j,k_s) - E(i_u,k_s)\} > 0 \tag{2}$$

This condition implies that the contribution to the total energy is always reduced by using an alternative rotamer $i_u$ to replace rotamer $i_j$. The efficiency of the Goldstein DEE filter is shown in Table 1 for 20 protein core redesigned problems.

**Integer Linear Programming Formulation.** The computational protein design problem can be reformulated into an integer linear programming problem by using graph-theoretical terms.[23,25] Let $G$ be an undirected $p$-partite graph with node set $V_1 \cup V_2 \cup \cdots V_p$ shown in Figure 1, where $V_i$ includes a node $i_j$ for each rotamer at position $i$ and the node subsets may have varying cardinalities, i.e., $n_i = |V_i|$. Each node $i_j$ of $V_i$ is assigned a weight $E(i_j,i_j)$; each pair of nodes $i_j \in V_i$ and $k_s \in V_k$ for $i \neq k$ is joined by an edge with a weight of $E(i_j,k_s)$. The global minimum energy conformation is achieved by picking one node among each subset to minimize the total weight of all edges and nodes in the induced subgraph. Let the node set of this graph be $V = V_1 \cup V_2 \cup \cdots \cup V_p$ and the edge set $D = \{(i_j,k_s): i_j \in V_i, k_s \in V_k, i \neq k\}$. Now, introduce a $\{0,1\}$ decision variable $x_{i_j,i_j}$ for each node $i_j$ in $V_i$ and a $\{0,1\}$ decision variable $y_{i_j,k_s}$ for each edge in $D$ so that setting $x_{i_j,i_j}$ to 1 corresponds to choosing rotamer $i_j$ and setting $y_{i_j,k_s}$ to 1 corresponds to the condition where the edge weight between rotamers $i_j$ and $k_s$ will be paid. The optimization framework for rotamer selection needs constraints to enforce the condition where exactly only one rotamer is chosen at each optimized position, and the interaction energy for edge $(i_j,k_s)$ is considered if and only if both rotamers $i_j$ and $k_s$ are chosen. The final integer programming formulation for computational protein design is presented as

$$(P0)\begin{cases} \text{minimize} & E = \displaystyle\sum_{i=1}^{p}\sum_{j=1}^{n_i} E(i_j,i_j)x_{i_j,i_j} + \sum_{i=1}^{p-1}\sum_{j=1}^{n_i}\sum_{k=i+1}^{p}\sum_{s=1}^{n_k} E(i_j,k_s)y_{i_j,k_s} \\ \text{subject to} & \\ & \displaystyle\sum_{j=1}^{n_i} x_{i_j,i_j} = 1 \quad \text{for } i = 1,\ldots,p \\ & \left.\begin{aligned} &\displaystyle\sum_{j=1}^{n_i} y_{i_j,k_s} = x_{k_s,k_s} \quad \text{for } s = 1,\ldots,n_k \\ &\displaystyle\sum_{s=1}^{n_k} y_{i_j,k_s} = x_{i_j,i_j} \quad \text{for } j = 1,\ldots,n_i \end{aligned}\right\} \text{for } i = 1,\ldots,p-1; k = i+1,\ldots,p \\ & x_{i_j,i_j} \in \{0,1\}, \text{ for } i = 1,\ldots,p; j = 1,\ldots,n_i \\ & y_{i_j,k_s} \in \{0,1\}, \text{ for } i = 1,\ldots,p-1; j = 1,\ldots,n_i; k = i+1,\ldots,p; s = 1,\ldots,n_k \end{cases}$$

Where, $p$ is the number of the optimized residue positions. The first set of constraints ensures that exactly one rotamer is chosen for each residue. The second set of constraints demands that the edge variable $x_{i_j,k_s}$ is set to 1 for edges that are in the subgraph induced by the chosen rotamers, i.e., if $x_{i_j,i_j} = 0$ (or $x_{k_s,k_s} = 0$), then no adjacent edges can be chosen and, if $x_{i_j,i_j} = 1$ (or $x_{k_s,k_s} = 1$), then exactly one adjacent edge is chosen for each vertex set. The number of the variables is given by $n = \sum_{i=1}^{p} n_i + \sum_{i=1}^{p-1}\sum_{k=i+1}^{p} (n_i n_k)$ and the number of the constraints $m = p + \sum_{i=1}^{p-1}\sum_{k=i+1}^{p} (n_i + n_k)$. Since all the variables are demanded to be binary, $P0$ is a huge combinatorial optimization problem. For example, assuming that we have 50 optimized positions and each position has 20 rotamers left after running some variant of the dead-end elimination filter,[14,16,17] the total number of the binary variables reaches 491 000, which is largely beyond the ability of the currently available optimization solvers for integer programming.[26,28] However, Kingsford et al.[25] reported that this ILP can be pruned quite efficiently by the standard branch-and-bound algorithm. So, we conjecture that

there must exist a special mathematical structure underlying this ILP such that the LP relaxation problem at the root node and its descendents must be very close to the ILP itself. And, the following section will provide a theorem to ensure that this conjecture is correct.

**Mixed-Integer Linear Programming Formulation.** The network flow problem[27] is represented by minimizing the transportation cost of moving materials through a network, $G = (V, E)$, to meet demands for material at various locations given sources of material at other locations. An important special case is when the set of nodes $V$ can be partitioned into two sets $S$ and $D$, $V = S \cup D$ and $S \cap D = 0$, such that every edge in $E$ has its tail in $S$ and its head in $D$. A network flow problem on such bipartite graphs is called a transportation problem. In the case where every supply node is connected to every demand node, the problem is called the Hitchcock transportation problem. If we denote the supplies at the supply nodes by $r_i$, $i \in S$, and the demands at the demand nodes by $s_j$, $j \in D$, then we can write the problem as

$$(\mathrm{P1}) \quad \begin{cases} \text{minimize} & \displaystyle\sum_{i=1}^{|S|}\sum_{j=1}^{|D|} c_{ij}x_{ij} \\ \text{subject to} & \displaystyle\sum_{j=1}^{|D|} x_{ij} = r_i \qquad i = 1,2,...,|S| \\ & \displaystyle\sum_{i=1}^{|S|} x_{ij} = s_j \qquad j = 1,2,...,|D| \\ & x_{ij} \geq 0 \qquad i = 1,2,...,|S|,\ j = 1,2,...,|D| \end{cases}$$

where $r_i$ is the supply units at node $i$, while $s_j$ is the demand units at node $j$.

There is an important property that the coefficient matrix $N$ of the network flow problem, such as ($P1$), satisfies, namely, that of total unimodularity. A matrix $N$ is said to be totally unimodular if every square submartix of $N$ has a determiant of $+1$, $-1$, or 0. For network flow problems with this property, the basic primal solutions are computed without any multiplications or divisions in the simplex algorithm, so we have the following integrality theorem, as follows.

**Theorem 1.**[26,27] *For network flow problems (P1) with integer data, every basic feasible solution and, in particular, every basic optimal solution assigns integer flow to every edge.*

By virtue of this theorem, the following theorem ensures that a mixed-integer linear programming formulation is equivalent to the integer programming formulation ($P0$) for the computational protein design problem, as follows.

**Theorem 2**. *The integer programming formulation (P0) of the computational protein design problem is equivalent to the following mixed-integer linear programming formulation, described by*

$$(P2) \quad \begin{cases} \text{minimize} & e = \displaystyle\sum_{i=1}^{p}\sum_{j=1}^{n_i} E(i_j,i_j)x_{i_j,i_j} + \sum_{i=1}^{p-1}\sum_{j=1}^{n_i}\sum_{k=i+1}^{p}\sum_{s=1}^{n_k} E(i_j,k_s)y_{i_j,k_s} \\ \text{subject to} & \\ & \displaystyle\sum_{j=1}^{n_i} x_{i_j,i_j} = 1 \qquad \text{for } i = 1,...,p \\ & \left.\begin{aligned} &\displaystyle\sum_{j=1}^{n_i} y_{i_j,k_s} = x_{k_s,k_s} \quad \text{for } s = 1,...,n_k \\ &\displaystyle\sum_{s=1}^{n_k} y_{i_j,k_s} = x_{i_j,i_j} \quad \text{for } j = 1,...,n_i \end{aligned}\right\} \text{for } i = 1,...,p-1; k = i+1,...,p \\ & x_{i_j,i_j} \in \{0,1\},\ \text{for } i = 1,...,p;\ j = 1,...,n_i \\ & 0 \leq y_{i_j,k_s} \leq 1, \quad \text{for } i = 1,...,p-1;\ j = 1,...,n_i;\ k = i+1,...,p;\ s = 1,...,n_k \end{cases}$$

**Proof.** Let the minimum values of problems ($P0$) and ($P2$) be $E^*$ and $e^*$. We have $e^* \leq E^*$ by observing that problem ($P2$) is a relaxation problem of problem ($P0$) since the integrality constraints on the edge variables are dropped. Assuming that $\bar{x}$ is any combination of 0 and 1 satisfying the first set of constraints in problem ($P2$), we have

$$(P3) \quad \begin{cases} \text{minimize} & \bar{e} = \displaystyle\sum_{i=1}^{p}\sum_{j=1}^{n_i} E(i_j,i_j)\bar{x}_{i_j,i_j} + \sum_{i=1}^{p-1}\sum_{j=1}^{n_i}\sum_{k=i+1}^{p}\sum_{s=1}^{n_k} E(i_j,k_s)y_{i_j,k_s} \\ \text{subject to} & \\ & \left.\begin{aligned} &\displaystyle\sum_{j=1}^{n_i} y_{i_j,k_s} = \bar{x}_{k_s,k_s} \quad \text{for } s = 1,...,n_k \\ &\displaystyle\sum_{s=1}^{n_k} y_{i_j,k_s} = \bar{x}_{i_j,i_j} \quad \text{for } j = 1,...,n_i \end{aligned}\right\} \text{for } i = 1,...,p-1; k = i+1,...,p \\ & 0 \leq y_{i_j,k_s} \leq 1, \quad \text{for } i = 1,...,p-1;\ j = 1,...,n_i;\ k = i+1,...,p;\ s = 1,...,n_k \end{cases}$$

where the first term of the objective function is constant. Since the edge variables at different pair residue positions $i$ and $k$ are separable, so problem ($P3$) can be decomposed into $p(p-1)/2$ Hitchcock transportation problems. It is also not difficult to confirm that the coefficient matrix of problem ($P3$) is totally unimodular by observing that each edge variable just appears in two different rows in the constraint matrix of problem ($P3$). According to the assumption on $\bar{x}$, the right-hand side of problem ($P3$) has integral data, and by virtue of theorem 1, the optimal solution of problem ($P3$) assigns integer to each edge variable. Since our choice of $\bar{x}$ represents all feasible combinations of 0 and 1 for problem ($P2$), we conclude that the minimum solution of problem ($P2$), i.e., $(x^*,y^*)$, is integral. So, $(x^*,y^*)$ is also a feasible solution to problem ($P1$). By virtue of $e^* \leq E^*$, we conclude that problem ($P2$) has the same optimal solution as that of problem ($P0$).

The essential difference between formulations ($P2$) and ($P0$) is that the integrality conditions on edge variables in ($P2$) have been dropped. Then, the number of the binary variables in problem ($P2$) is given by $n_B = \Sigma_{i=1}^{p} n_i$, which is a tractable number for the standard mixed-integer linear programming software, such as ILOG CPLEX,[28] with the number of continuous variables given by $n = \Sigma_{i=1}^{p-1} \Sigma_{k=i+1}^{p} (n_i n_k)$. For instance, the ($P2$) formulation of the above-mentioned example has only 1000 binary variables, which is far less than the 491 000 in its ($P0$) formulation and the left 490 000 variables have changed to be continuous ones. In particular, while the residues in position $i$ and $k$ are too far apart to have any rotamers that interact with each other, i.e., the pairwise energies between their rotamers are zero, we can remove all the variables $y_{i_j,k_s}$ with $i_j \in V_i$ and $k_s \in V_k$, and remove all the constraints concerning of those edge variables without affecting the optimal solution and destroying the partial network flow structure.

**Table 1. Computational Results for Core Redesigned Proteins[a]**

| ser. | prot. | no. opt | no.rot | size | time (s) MILP (ILP)[c] | MILP opt energy (LP) | MILP no. node (ILP) | no. var[d] | no. constr |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 1aac | 20 | 566(1860) | 29(39) | 93.8(89.3)[b] | −124.56(−125.99) | 2(4) | 150088 | 10774 |
| 2 | 1b9o | 23 | 605(2139) | 32(45) | 90.1(315.4) | −139.69(−149.61) | 33(122) | 174379 | 13333 |
| 3 | 1c5e | 18 | 369(1674) | 23(35) | 2.3(8.1) | −103.03(−104.11) | 2(2) | 63386 | 6291 |
| 4 | 1c9o | 12 | 258(1116) | 15(23) | 2.6(5.0) | −64.66(−65.73) | 3(2) | 29742 | 2850 |
| 5 | 1cc7 | 11 | 381(1023) | 17(21) | 34.7(56.7) | −68.67(−77.11) | 61(61) | 65942 | 3821 |
| 6 | 1cex | 50 | 1132(4650) | 67(98) | 113.2(158.9) | −262.26(−263.41) | 4(7) | 624474 | 55518 |
| 7 | 1cku | 11 | 129(1023) | 11(21) | 0.1(0.1) | −61.86(−61.86) | 0(0) | 7178 | 1301 |
| 8 | 1ctj | 17 | 321(1581) | 21(33) | 2.4(5.2) | −99.14(−99.14) | 0(0) | 47759 | 5153 |
| 9 | 1cz9 | 28 | 588(2604) | 37(55) | 28.7(73.5) | −147.22(−149.96) | 10(63) | 165154 | 15904 |
| 10 | 1czp | 18 | 401(1674) | 24(35) | 4.6(12.7) | −84.30(−85.83) | 2(1) | 75165 | 6835 |
| 11 | 1d4t | 20 | 623(1860) | 30(40) | 372.0(911.2) | −124.55(−140.72) | 114(1497) | 182739 | 11857 |
| 12 | 1igd | 10 | 203(930) | 13(19) | 1.6(1.22)[b] | −66.79(−70.21) | 4(0) | 17952 | 1837 |
| 13 | 1pga | 10 | 207(930) | 13(19) | 0.2(0.4) | −67.74(−67.74) | 0(0) | 18528 | 1873 |
| 14 | 1qq4 | 40 | 851(3720) | 53(78) | 100.8(133.7) | −209.39(−217.88) | 50(159) | 350901 | 33229 |
| 15 | 1qtn | 26 | 620(2418) | 35(51) | 73.7(233.9) | −162.60(−169.08) | 42(38) | 183195 | 15526 |
| 16 | 1ubq | 14 | 445(1302) | 21(27) | 146.4(145.2)[b] | −76.81(−85.45) | 48(40) | 91610 | 5799 |
| 17 | 2pth | 45 | 1062(4185) | 61(88) | 1710.4(4556.4) | −222.16(−235.59) | 345(2281) | 547726 | 46773 |
| 18 | 3lzt | 26 | 555(2418) | 34(51) | 57.0(272.8) | −142.01(−157.49) | 36(39) | 146311 | 13901 |
| 19 | 5p21 | 45 | 1227(4185) | 64(88) | 1499.9(1890.5) | −269.56(−283.15) | 124(335) | 732027 | 54033 |
| 20 | 7rsa | 15 | 254(1395) | 18(29) | 3.64(2.03)[b] | −89.45(−90.82) | 2(0) | 29069 | 3571 |

[a] The no. opt column gives the number of core positions that were allowed to be optimized. The no. rot column gives the rotamers left after running the Goldstein DEE filter, and the number in parenthesis gives the total number of rotamers. Size is the log10 of the search space left after the Goldstein DEE filter, and the number in the parenthesis gives the original search space. Time is the number of CPU seconds spent on solving the MILP, and the number in the parenthesis gives that on ILP. MILP opt energy gives the minimum energy of the optimal rotamer sequence, and the amount in parenthesis gives the solution to the LP relaxation of the MILP. MILP no. node gives the number of LP problems solved in finding the optimal rotamer sequence by using MILP, and the number in parenthesis gives that by using ILP. The no. var column gives the number of all variables in MILP, and note that the number of 0−1 variables is the same as the number of rotamers left after running DEE. The no. constr column gives the number of constraints in MILP. [b] The proposed MILP performs poorly because CPLEX uses different heuristics for MILP and ILP. [c] The calculation is performed on a personal computer with 2.8 GHz of CPU speed and 512 MB of RAM. [d] The number of binary variables is equal to the number of rotamers left after Goldstein DEE filtering, and the number of continuous variables can be calculated from the difference between the total number of variables (no. var column) and the total number of binary variables.

## Results and Discussion

A mixed-integer linear programming algorithm was developed in the PROtein Design Algorithmic/PRODA package written in C/C++, where the side-chain template and side-chain−side-chain interaction energies are computed using the parameter set 22 of the CHARMM force field.[29] This force field considers explicit positions of all hydrogen atoms and includes only nonbonded terms for van der Waals and electrostatic interactions in our protein core redesigned testing. The solvation effects are modeled by using approximate pairwise surface area decomposition to reward buried nonpolar surface areas with a parameter of 26 cal/mol Å$^2$ and to penalize exposed nonpolar surface areas with a parameter of 100 cal/mol Å$^2$,[30] and the solvent-accessible surface areas of the overlapping atoms are calculated by using the numerical surface calculation (NSC) algorithm.[31] The backbone-independent rotamer library of Xiang and Honig[5] with 984 rotamers was used for sequence selection, and the variable core residue identities were allowed to choose from among the hydrophobic amino acids A, V, L, I, F, Y, and W, each of which having 93 rotamers. The precalculated energy matrices for 20 proteins in our test set (25,20) were performed on a PC cluster with 24 dual Xeon 3.2 GHz of CPU speed and 2.0 GB of RAM, but the sequence selection using the mixed-integer linear programming algorithm for each protein was performed on a personal computer with 2.8 GHz of CPU speed and 512 MB of RAM. The Goldstein DEE preprocessor was not accounted for in the CPU time, and the dual simplex linear programming method of ILOG CPLEX[28] was chosen to solve the linear programming subproblems in the ILOG CPLEX mixed-integer linear programming solver in order to take advantage of the network flow structure embedded in the constraint matrix of problem ($P2$). For each of the 20 proteins in our test set shown in Table 1, only the residues which expose less than 10% of their surface areas to the solvent and belong to the above-defined hydrophobic amino acid set are allowed to assume any rotamer of any amino acid in this set, but the backbone and nonoptimized residues are held fixed at the coordinates obtained from the Protein Data Bank (PDB).[32] The

**Table 2. Best Scoring Sequence Selected by PRODA for Protein Cores**[a]

| Prot. | Sequence |
|---|---|
| 1aac | A3A, V23V, I25F, A26F, Y30F, V37V, V43L, W45W, I46I, V55V, F57F, L62L, A77A, Y78F, L80L, F82F, Y88F, Y90F, V102V, V104V |
| 1b9o | F3F, L8V, L12W, I15W, I21L, L23L, L26W, I27L, Y36W, Y50F, L52F, I55L, W60W, F80A, I85L, A92F, I95Y, L96L, I101F, W104L, L105L, W108F |
| 1c5e | L27L, A31L, L37L, L46Y, W49F, A57A, I60V, L61L, A65A, L72L, F74F, F80W, V85L, V91L, A100A, F101F, I106L, I108L |
| 1c9o | V6V, F9F, I18L, V26Y, V28I, A32A, I33F, L41L, V47L, F49F, A60A, V63A |
| 1cc7 | Y7L, F9F, V11I, V22V, L26A, L29W, V33L, V45W, V47A, I56F, I60L |
| 1cex | L23L, A32A, V34L, I35V, F36A, I37L, Y38F, A39A, L48A, L51W, I55L, A56W, L59L, V68I, I70W, V73V, A79F, A94A, I95L, L101W, F102F, A105A, A112A, L114L, I115V, A116A, A123A, A124A, L125L, A126A, A127A, A128A, I130L, L133L, I137W, I141V, A142L, V145V, L146W, F147F, Y149F, I159A, Y162F, F170F, V177V, A195A, A199A, F202F, L203L, V207W |
| 1cku | V7V, A14A, Y19F, A23Y, W60W, F66F, L70L, I71V, V73L, W76A, W80F |
| 1ctj | V10V, F11F, L32F, I37L, L41W, A49A, I50L, I54W, W64F, L68F, I73L, V76I, A77V, A78F, Y79F, V80L, A84A |
| 1cz9 | W61A, F65W, L77L, A78W, V79L, V81I, I88V, V89W, V90A, A100A, A101L, W105F, A108F, I109L, I118L, F126F, W134F, L135F, I140V, V156V, A159A, L163F, I167L, A171A, I180L, L188F, L189L, A192V |
| 1czp | V5L, L7L, I18A, V20V, I26A, L27W, A29A, A30F, L37L, A50A, L66L, Y75F, V76F, L77L, V80V, A81A, V87L, I89I |
| 1d4t | V4L, V6I, L20L, Y29A, L30L, L31L, V40W, L43L, V45L, L46L, V56L, W64W, A66W, I80W, L83A, I84L, F87L, I94L, L98L, V102W |
| 1igd | Y8F, L10A, I12W, A25A, A31A, F35F, A39A, V44L, F57F, V59F |
| 1pga | Y3F, L5A, L7F, A20A, A26A, F30F, A34A, V39L, F52F, V54F |
| 1qq4 | I3A, V4W, Y9A, I11A, V19L, F21F, V23A, F31W, V32A, A34V, A42W, A44F, I46L, V51A, F54V, F59F, A65A, W66Y, V67V, L69A, V79V, V86F, V88V, V100V, I114L, A121W, V128I, L131L, A136A, W147F, I148I, A154A, V157V, L180Y, F181F, L184L, I187L, L188L, Y191A, L193L |
| 1qtn | L237A, I238A, I239L, F244F, A247A, L254F, A267A, F274W, F279F, I281W, V289A, I292L, I295F, L296A, Y299W, F310F, I311L, I314V, L315F, I322W, I323I, Y324L, A331L, L336F, F343F, L348L |
| 1ubq | I3I, V5V, I13F, L15L, V17L, I23V, V26A, I30W, L43W, L50L, L56L, I61A, L67F, L69A |
| 2pth | I2W, L4A, I5W, V6F, L8L, A9A, A22L, A24A, F26F, V27V, L30L, A31A, F44W, V51A, L53L, V58V, L60L, L61F, V62I, A73A, V74L, A78L, I83V, L88A, L89A, V90V, A91A, L97W, A102A, F104W, L116W, I119I, I120V, F129F, L132F, I134L, I136F, V149W, I161V, A164A, I165I, A168A, A169W, A183A, L187L |
| 3lzt | F3F, L8L, A9A, A11W, L17F, L25L, W28F, V29V, A31A, A32A, F38W, Y53W, I55V, L56L, I58L, W63L, L83L, I88I, V92A, A95A, I98L, V99L, W108W, A110W, W111F, V1120Y |
| 5p21 | Y4F, L6L, V7V, V8L, V9F, A11W, V14V, A18W, L19L, L23F, I46L, L52W, L53F, I55L, L56W, A59A, Y64W, Y71W, F78F, L79L, V81L, F82F, A83A, I84I, F90F, I93V, Y96F, I100L, V103V, V109I, V112I, L113L, V114V, V125V, A130A, A134A, I139W, Y141F, A146A, V152V, A155A, F156L, L159A, V160W, I163W |
| 7rsa | F8F, A20A, Y25F, L35L, F46F, V47A, V54F, V57V, I81L, I106V, I107L, V108I, A109L, F120F, A122L |

[a] The numbers in the sequence column give the variable residues of each protein corresponding to those exposing less than 10% of their surface area to solvent, and their identities are given in one-letter amino acid form preceding those positions. The designed sequences by PRODA are given following the positions. The predicted identical positions are colored by red for clarity.

core residues are chosen in our test set because of two considerations: one is that Kuhlman and Baker[33] demonstrated, by using an extensive test set of protein backbones drawn from 7-fold families, that more than half of the core residues were predicted correctly simply by minimizing a folding free energy function in sequence space; the other is that the physical model described above is validated for core residues[15,33] so that it can avoid the misleading impressive optimization performance with an inappropriate physical model.[34]

The characteristic data and computational results for 20 redesigned proteins by using the mixed-integer linear programming algorithm are given in Table 1, and the best scoring sequences selected by PRODA for core positions are shown in Table 2. In the designed sequences, 41.4% (190/459) of the core residues are identical to the amino acids in the corresponding positions in the native sequence. This result is in good agreement with that of Kuhlman and Baker,[33] and it verifies that the physical model used here is valid to test the mixed-

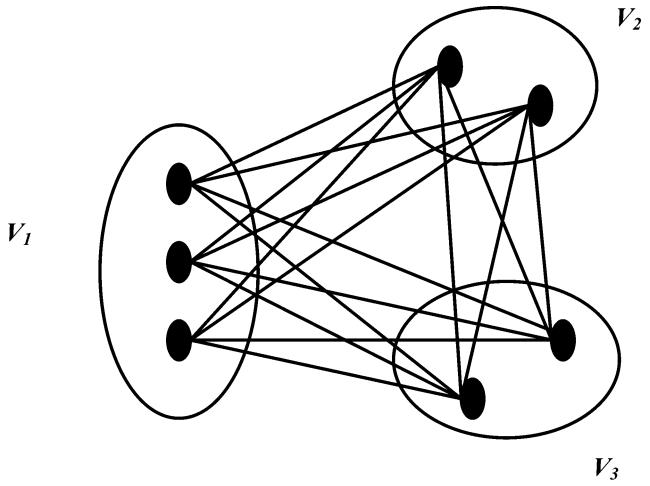


**Figure 1.** Graph for a computational protein design problem with three rotamers at position 1 and two rotamers at positions 2 and 3. Each open ellipsoid represents a backbone position; each filled ellipsoid node represents a rotamer alternative; each edge connecting two nodes is weighted by the pairwise energy. In this picture, the goal is to choose the one node within each open ellipsoid that minimizes the sum of the three edge weights connecting the nodes together with the weights of the three nodes.

integer linear programming algorithm. In Table 1, it is clear to see that many rotamers are still left after running Goldstein DEE and the residual searching spaces of the sequences still reach $10^{67}$ and $10^{64}$ for 1cex and 5p21, respectively. But, the standard ILOG CPLEX mixed-integer linear programming solver can identify their deterministic global optimal sequences in less than 30 CPU min on low-end personal computers. Only 3 out of 20 solutions of the linear programming relaxations have integral solutions. Among others, the optimal energies of the linear programming relaxation and the global minimum ones are much different for large proteins. This result agrees with that of Kingsford et al.[25] and shows the critical necessity of identifying the integral solution to the protein design problem. A plausible explanation for this phenomenon comes from the different characteristics of similar-sized side-chain positioning and protein design problems.[34] For the protein design problem, there are few positions with many rotamer choices, whereas, for repacking side chains on a backbone, there are many positions with few rotamer choices. This difference leads to the energy matrix that emphasizes off-diagonal elements contributed by rotamer−rotamer interaction relative to the diagonal elements contributed by rotamer−template interaction. This causes more coupling and means more difficult optimization.

For the CPU running time and searched nodes shown in Table 1 for different MILP and ILP formulations of 20 protein core redesigned problems, ILOG CPLEX prunes the MILP formulation faster than the ILP formulation by up to 3 times for most proteins with many less nodes searched. It can be explained easily by observing that ILOG CPLEX avoids the wrong branching on the edge variables that do not need integrality constraints at all according to theorem 2. Another reason comes from the different cutting planes ILOG CPLEX generated for MILP and ILP since ILOG CPLEX can produce Gomory mixed-integer cuts for MILP which are stronger than the Gomory fractional cuts for ILP.[26] More importantly, the efficiency of the mixed-integer linear programming algorithm can be further improved by exploiting the network flow structure embedded in the constraint matrix, such as the network flow simplex algorithm which can be used to solve the linear programming subproblems by using Bender's decomposition strategy.[26,35] Further, the well-known mixed-integer cuts for fixed-charge network flow problem[26] and the lift-and-project cutting planes[36,37] can be incorporated into ILOG CPLEX to further accelerate the branch-and-bound searching by improving the bounding operation at the root and leaf nodes.

## Conclusion

Here, we have proposed a mixed-integer linear programming algorithm for a computational protein design problem based on a critical finding that the network flow structure embedded in the integer linear programming formulation of the computational protein design problem makes it equivalent to a mixed-integer linear programming formulation with fewer binary variables. This novel formulation effectively reduces the combinatorial difficulty of the computational protein design problem, and the preliminary calculation results for 20 core redesigned proteins show that the mixed-integer linear programming algorithm with a validated physical model is faster than its integer linear programming counterpart by up to 3 times with many less nodes searched. More importantly, the specific mathematical structure underlying the mixed-integer linear programming formulation suggests that the mixed-integer linear programming algorithm should be a prospective technique to attack the computational protein design problem.

## Acknowledgment

This work was supported by the Scientific Research Foundation for the Returned Overseas Chinese Scholars, Ministry of Education, China, and the National Natural Science Foundation of China with Grant No. 20506013.

## Literature Cited

(1) Dahiyat, B. I.; Mayo, S. L. De novo protein design: fully automated sequence selection. *Science* **1997**, *278*, 82−87.

(2) Ponder, J. W.; Richards, F. M. Tertiary template for proteins: use of packing criteria in the enumeration of allowed sequences for different structural classes. *J. Mol. Biol.* **1987**, *193*, 775−791.

(3) Dunbrack, R. L., Jr.; Karplus, M. Backbone-dependent rotamer library for proteins: application to side-chain prediction. *J. Mol. Biol*. **1993**, *230*, 543−574.

(4) Bower, M. J.; Cohen, F. E.; Dunbrack, R. L., Jr. Prediction of protein side-chain rotamers from a backbone-dependent rotamer library: a homology modeling tool. *J. Mol. Biol*. **1997**, *267*, 1268−1282.

(5) Xiang, Z.; Honig, B. Extending the accuracy limits of prediction for side-chain conformations. *J. Mol. Biol.* **2001**, *311*, 421−430.

(6) Malakauskas, S. M.; Mayo, S. L. Design, structure and stability of a hyperthermophilic protein variant. *Nature Struct. Biol.* **1998**, *5*, 470−475.

(7) Looger, L. L.; Dwyer, M. A.; Smith, J. J.; Hellinga, H. W. Computational design of receptor and sensor proteins with novel functions. *Nature* **2003**, *423*, 185−190.

(8) Dwyer, M. A.; Looger, L. L.; Hellinga, H. W. Computational design of a biological active enzyme. *Science* **2004**, *304*, 1967−1971.

(9) Floudas, C. A. Research challenges, opportunities and synergism in systems engineering and computational biology. *AIChE J*. **2005,** *51*, 1872−1884.

(10) Floudas, C. A.; Fung, H. K.; McAllister, S. R.; Monnigmann, M.; Rajgaria, R. Advances in protein structure prediction and de novo protein design: A review. *Chem. Eng. Sci.* **2006**, *61*, 966−988.

(11) Fung, H. K.; Rao, S.; Floudas, C. A.; Prokopyev, O.; Pardalos, P. M.; Rendl, F. Computational comparison studies of quadratic assignment like formulations for the in silico sequence selection problem in de novo protein design. *J. Comb. Opt.* **2005**, *10*, 41−60.

(12) Morikis, D.; Soulika, A. M.; Mallik, B.; Klepeis, J. L.; Floudas, C. A.; Lambris, J. D. Improvement of the anti-C3 activity of compstain using rational and combinatorial approaches. *Biochem. Soc. Trans.* **2004**, *32*, 28−32.

(13) Klepeis, J. L.; Floudas, C. A.; Morikis, D.; Tsokos, C. G.; Argyropoulos, E.; Spruce, L.; Lambris, J. D. Integrated computational and experimental approach for lead optimization and design of compstatin variants with improved activity. *J. Am. Chem. Soc.* **2003**, *125*, 8422−8423.

(14) Desmet, J.; De Maeyer, M.; Hazers, B.; Lasters, I. The dead-end elimination theorem and its use in protein side-chain positioning. *Nature* **1992**, *356*, 539−542.

(15) Dahiyat, B. I.; Mayo, S. L. Protein design automation. *Protein Sci.* **1996**, *5*, 895−903.

(16) Goldstein, R. F. Efficient rotamer elimination applied to protein side chains and related spin glasses. *Biophys. J.* **1994**, *66*, 1335−1340.

(17) Pierce, N. A.; Spriet, J. A.; Desmet, J.; Mayo, S. L. Conformational splitting: A more powerful criterion for dead-end elimination. *J. Comp. Chem.* **1999**, *21*, 999−1009.

(18) Looger, L. L.; Hellinga, H. W. Generalized dead-end elimination algorithms make large-scale protein side-chain structure prediction tractable: implications for protein design and structural genomics. *J. Mol. Biol.* **2001**, *307*, 429−445.

(19) Gordon, D. B.; Mayo, S. L. Radical performance enhancements for combinatorial optimization algorithms based on the dead-end elimination theorem. *J. Comput. Chem.* **1998**, *19*, 1505−1514.

(20) Wernisch, L.; Hery, S.; Wodak, S. J. Automatic protein design with all atom force-fields by exact and heuristic optimization. *J. Mol. Biol.* **2000**, *301*, 713−736.

(21) Pierce, N. A.; Winfree, E. Protein design is NP-hard. *Protein Eng.* **2002**, *15*, 779−782.

(22) Chazelle, B.; Kingsford, C.; Singh, M. A semidefinite programming approach to side-chain positioning with new rounding strategies. *INFORMS J. Comput.* **2004**, *16*, 380−392.

(23) Althaus, E.; Kohlbacher, O.; Lenhof, H.-P.; Muller, P. A combinatorial approach to protein docking with flexible side chains. *J. Comput. Biol.* **2002**, *9*, 597−612.

(24) Klepeis, J. L.; Floudas, C. A.; Morikis, D.; Tsokos, C. G.; Lambris, J. D. Design of peptide analogues with improved activity using a novel de novo protein design approach. *Ind. Eng. Chem. Res.* **2004**, *43*, 3817−3826.

(25) Kingsford, C. L.; Chazelle, B.; Singh, M. Solving and analyzing side-chain positioning problems using linear and integer programming. *Bioinformatics* **2005**, *21*, 1028−1036.

(26) Nemhauser, G. L.; Wolsey, L. A. *Integer and combinatorial optimization*; John Wiley & Sons: New York, 1988.

(27) Ahuja, R. K.; Magnanti, T. L.; Orlin, J. B. *Network flows*; Prentice Hall: Upper Saddle River, NJ, 1993.

(28) *ILOG CPLEX*, version 9.0; 2003.

(29) MacKerell, A. D., Jr.; Bashford, D.; Bellott, M.; et al. All-atom empirical potential for molecular modeling and dynamics studies of proteins. *J. Phys. Chem. B* **1998**, *102*, 3586−3616.

(30) Street, A. G.; Mayo, S. L. Pairwise calculation of protein solvent accessible surface area. *Folding Des.* **1998**, *3*, 253−258.

(31) Eisenhaber, F.; Lijnzaad, P.; Argos, P.; Sander, C.; Scharf, M. The double cubic lattice method: efficient approaches to numerical integration of surface area and volume and to dot surface contouring of molecular assemblies. *J. Comput. Chem.* **1995**, *16*, 273−284.

(32) Berman, H. M.; Westbrook, J.; Feng, Z.; Gilliland, G.; Bhat, T. N.; Weissig, H.; Shindyalov, I. N.; Bourne, P. E. The Protein Data Bank. *Nucleic Acids Res.* **2000**, *28*, 235−242.

(33) Kuhlman, B.; Baker, D. Native protein sequence are close to optimal for their structures. *Proc. Natl. Acad. Sci. U.S.A.* **2000**, *97*, 10383−10388.

(34) Gordon, D. B.; Hom, G.; Mayo, S. L.; Pierce, N. Exact rotamer optimization for protein design. *J. Comput. Chem.* **2002**, *24*, 232−242.

(35) Zhu, Y.; Kuno, T. Global optimization of nonconvex MINLP by a hybrid branch-and-bound and revised general Benders decomposition method. *Ind. Eng. Chem. Res.* **2003**, *42*, 528−539.

(36) Balas, E.; Ceria, S.; Cornuejols, G. A lift-and-project cutting plane algorithm for mixed-integer 0−1 programs. *Math. Program.* **1993**, *58*, 295−324.

(37) Zhu, Y.; Kuno, T. A disjunctive cutting plane based branch-and-cut algorithm for 0−1 mixed-integer nonlinear programs. *Ind. Eng. Chem. Res.* **2006**, *45*, 187−196.

*Received for review* May 15, 2006
*Revised manuscript received* August 20, 2006
*Accepted* November 13, 2006

IE0605985