A new framework for computational protein design through cost function network optimization

Seydou Traore, David Allouche, Isabelle André, Simon de Givry, George Katsirelos, Thomas Schiex, Sophie Barbe

BIOINFORMATICS ORIGINAL PAPER

Vol. 29 no. 17 2013, pages 2129–2136
doi:10.1093/bioinformatics/btt374

Structural bioinformatics

Advance Access publication July 10, 2013

A new framework for computational protein design through cost function network optimization

Seydou Traoré[1,2,3,†], David Allouche[4,†], Isabelle André[1,2,3], Simon de Givry[4], George Katsirelos[4], Thomas Schiex[4,*] and Sophie Barbe[1,2,3,*]

[1]Université de Toulouse; INSA, UPS, INP; LISBP, 135 Avenue de Rangueil, F-31077 Toulouse, France, [2]INRA, UMR792, Ingénierie des Systèmes Biologiques et des Procédés, F-31400 Toulouse, France, [3]CNRS, UMR5504, F-31400 Toulouse, France and [4]Unité de Mathématiques et Informatique Appliquées, UR 875, INRA, F-31320 Castanet Tolosan, France

Associate Editor: Anna Tramontano

ABSTRACT

Motivation: The main challenge for structure-based computational protein design (CPD) remains the combinatorial nature of the search space. Even in its simplest fixed-backbone formulation, CPD encompasses a computationally difficult NP-hard problem that prevents the exact exploration of complex systems defining large sequence-conformation spaces.

Results: We present here a CPD framework, based on cost function network (CFN) solving, a recent exact combinatorial optimization technique, to efficiently handle highly complex combinatorial spaces encountered in various protein design problems. We show that the CFN-based approach is able to solve optimality a variety of complex designs that could often not be solved using a usual CPD-dedicated tool or state-of-the-art exact operations research tools. Beyond the identification of the optimal solution, the global minimum-energy conformation, the CFN-based method is also able to quickly enumerate large ensembles of suboptimal solutions of interest to rationally build experimental enzyme mutant libraries.

Availability: The combined pipeline used to generate energetic models (based on a patched version of the open source solver Osprey 2.0), the conversion to CFN models (based on Perl scripts) and CFN solving (based on the open source solver toulbar2) are all available at http://genoweb.toulouse.inra.fr/~tschiex/CPD

Contacts: thomas.schiex@toulouse.inra.fr or sophie.barbe@insa-toulouse.fr

Supplementary information: Supplementary data are available at *Bioinformatics* online.

Received on March 7, 2013; revised on May 29, 2013; accepted on June 21, 2013

1 INTRODUCTION

The engineering of tailored proteins with desired properties holds great interest for applications ranging from medicine, biotechnology (Nestl *et al.*, 2011) and synthetic biology to nanotechnologies (Grunwald *et al.*, 2009). Although, directed evolution techniques coupled with high-throughput automated procedures have met with some success, they do not provide structural design principles to guide the rational design of novel proteins. The development of generic and effective protein engineering methodologies, both experimental and computational, is thus of utmost interest to speedup the design of tailored proteins having the desired properties. Structure-based computational protein design (CPD) approaches have demonstrated their potential to adequately capture fundamental aspects of molecular recognition and interactions, which have already enabled the successful (re)design of several enzymes for various purposes (Dahiyat and Mayo, 1997; Hellinga and Richards, 1991). Despite these outstanding results, the efficiency, predictability and reliability of CPD methods have shown that they still need to mature.

CPD is faced with several challenges. The first lies in the exponential size of the conformational and protein sequence space that has to be explored, which rapidly grows out of reach of computational approaches. Another obstacle to overcome is the unsolved issue of accurate structure prediction for a given sequence. Therefore, the design problem is usually approached as an inverse folding problem (Pabo, 1983), to reduce the problem to the identification of an amino acid sequence that can fold into a target protein 3D-scaffold that matches the design objective. This paradigm typically assumes a fixed protein backbone and, for each type of amino acid considered at a given position, allows the side chains to move only among a set of discrete and low-energy conformations, called rotamers (Janin *et al.*, 1978). CPD is thus formulated as an optimization problem, which consists in choosing combinations of rotamers at designable specified positions such that the fold is stabilized and the desired property is achieved. To solve this problem, we need a computationally tractable energetic model to evaluate the energy of any combination of rotamers. We also require computational optimization techniques that can efficiently explore the sequence-conformation space to find the sequence-conformation model of global minimum energy (GMEC: global minimum-energy conformation) or an ensemble of low-energy sequence-conformation models. Indeed, several reasons can motivate the generation of multiple near-optimal solutions. First, the sequence-conformation model with the lowest predicted energy may not fold into the targeted protein scaffold owing to inaccuracies in the modeling of protein energetics. Secondly, the GMEC solution may be so stabilized that it can lack the flexibility required to operate the protein biological function (Arnold, 2001). Such suboptimal ensembles can then be analyzed to rationally guide the

*To whom correspondence should be addressed.
†The authors wish it to be known that, in their opinion, the first two authors should be regarded as joint First Authors.

experimental construction of protein libraries while enhancing the chances of success to identify a protein hit.

The protein design problem modeled with a rigid backbone, a discrete set of rotamers and pairwise energy functions, has been proven to be NP-hard (Pierce and Winfree, 2002). Hence several meta-heuristic methods have been applied to it, including Monte-Carlo with simulated annealing (Kuhlman and Baker, 2000; Voigt *et al*., 2000), genetic algorithms (Desjarlais and Handel, 1995; Raha *et al*., 2000) and other methods (Allen and Mayo, 2006; Desmet *et al*., 2002; Wernisch *et al*., 2000). These approaches can usually find a relatively low-energy model fairly quickly but without any guarantees of completeness or accuracy. Indeed, these stochastic optimization routines may end up trapped in local minima and miss the GMEC with no indication.

Conversely, there exist methods that solve the GMEC exactly, such as approaches based on the dead-end elimination (DEE) theorem (Desmet *et al*., 1992), on the branch-and-bound algorithm (Gordon and Mayo, 1999; Hong *et al*., 2009; Wernisch *et al*., 2000), on integer linear programming (Althaus *et al*., 2002; Kingsford *et al*., 2005) or on dynamic programming (Leaver-Fay *et al*., 2005). These exact methods offer several advantages. First, they ensure that discrepancies between CPD predictions and experimental results come exclusively from the inadequacies of the biophysical model and not from the algorithm. Next, because provable methods can determine that the optimum is reached, they may actually stop before meta-heuristic approaches. Finally, empirical studies on solving the GMEC problem reported that the accuracy of meta-heuristic approaches tend to degrade as the problem size increases (Voigt *et al*., 2000).

In this article, we modeled the CPD problem as either a binary cost function network (CFN) or an integer linear programming (ILP) problem (Section 2.4). We compared the performance of the open source CFN solver *toulbar2* and the IBM™ ILOG ILP solver *cplex* against that of the combined DEE/A* approach as implemented in the dedicated CPD software *Osprey* (Section 3.2), for design problems (Section 3.1). The CFN-based method outperformed by several orders of magnitude the other methods both in identifying the GMEC but also in producing a set of low-energy sequence-conformation models. This second step was not attainable in most of the study cases using DEE/A* (Section 3.3). Therefore, on the basis of the CFN approach, we propose a new CPD framework (Fig. 1). Our methodology, which we describe in Section 2, is well-adapted to solving exactly macromolecular design problems of large sequence-conformation spaces. It also has the potential to improve methods that integrate flexibility to a larger extent in protein design, as this considerably expands the size of the search space or may require solving a large number of GMEC instances (Hallen *et al*., 2013; Humphris and Kortemme, 2008). These aspects are highly relevant to CPD and we shall address them here.

2 METHODS

The CPD strategy introduced in this work (Fig. 1) is composed of five main stages discussed in more details in the following subsections. The whole CPD framework and the approaches used to handle the sequence-conformation combinatorial optimization problem were assessed for the design of more stable proteins and cofactor-bound proteins as well as protein–ligand and protein–protein interfaces.

2.1 Preparation of structural molecular systems

Three-dimensional models of proteins in free and complex states were derived from their respective crystallographic structures deposited in the protein data bank (PDB) (Bernstein *et al*., 1977) (Supplementary Table S1). Missing heavy atoms in crystal structures as well as hydrogen atoms were added using the *tleap* module of the *Amber 9* software package (Case *et al*., 2006). Cofactors as well as crystallographic water molecules specified in *SITE* and *LINK* records of PDB files were kept in structural models. Histidine protonation states and disulfide bonds were assigned using the *tleap* module. For multimeric proteins, the transformation matrix specified in the PDB file was applied to reproduce missing chains. Parameters for non-amino acid type ligands and cofactors were generated with the *Antechamber* module of *Amber 9* (Wang *et al*., 2006). The molecular all-atom *ff99SB* (Hornak *et al*., 2006), *Glycam06* (Kirschner *et al*., 2008) and *gaff* force fields (Wang *et al*., 2004) were used for the proteins, carbohydrates and other non-standard molecules, respectively. Each molecular system was then subjected to 500 steps of minimization with the *Sander* module of *Amber 9*, using the generalized born/surface area implicit solvent model (Hawkins *et al*., 1996).

2.2 Definition of sequence-conformation spaces

The residues of each protein were classified into three layers (labeled core, boundary or surface) according to their burial in the 3D-model (Supplementary Fig. S1). This burial of residues was defined by calculating their solvation radius from atomic solvation radii, as defined by (Archontis and Simonson, 2005). The salvation radius B_R of residue R is given as follows:

$$B_R = \frac{\sum_{i \in R} q_i^2}{\sum_{i \in R} \frac{q_i^2}{b_i}} \quad (1)$$

where b_i and q_i are the atomic solvation radius of atom i from residue R and its partial charge, respectively. From these calculations, three layers of decreasing residue-solvation radius from the core to the surface of the protein were defined. The set of amino acid types considered at each mutable residue (i.e. candidate positions for redesign) of the proteins (in their apo form or bound to a cofactor) depends on the layer to which the residue belongs to. Further details regarding the selection of designed positions and the allowed amino acids can be found in the Supplementary data.

2.3 Computation of pairwise energies

The total energy (E_{total}) of a sequence-conformation model defined by the selection of one specific amino acid associated with a given conformation (rotamer) for each variable amino acid type is assessed as follows:

$$E_{total} = E_c + \sum_i E(i_r) + \sum_{i,j} E(i_r, j_s) \quad (2)$$

where E_c is a constant energy contribution capturing interactions between fixed parts of the model, $E(i_r) = E_{self}(i_r) - E_{ref}(i_r)$ is the difference between the self energy of rotamer r at position i capturing internal interactions or interactions with fixed regions $E_{self}(i_r)$ and its reference energy $E_{ref}(i_r)$, which corresponds to the lowest computed intra-rotamer energy for each amino acid type by variable residue position (Lippow *et al*., 2007) and $E(i_r, j_s)$ is the pairwise interaction energy between rotamer r at position i and rotamer s at position j. In this formulation, the conformations (i.e. rotamers) of amino acid–type ligands are processed as rotamers of amino acid side chains.

All pairwise energy terms were pre-computed and stored using *Osprey 2.0* (Gainza *et al*., 2012). These calculations were based on the *Amber*

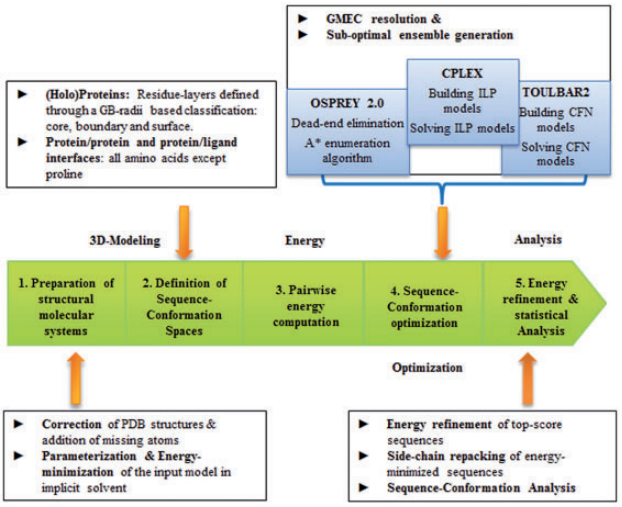

Fig. 1. Flow chart of the CPD framework

all-atom *ff94* parameters implemented in *Osprey 2.0* as well as on additional force field parameters generated from *Glycam06* and *gaff* force fields using the *Antechamber* module of *Amber 9*. These parameters were added in the parameter files of *Osprey 2.0* (Chen *et al*., 2009) and were used for modeling carbohydrates and other non-standard molecules. The energy functions consisted in the sum of the *Amber* electrostatic terms (with a distance-dependent dielectric constant), van der Waals and dihedral energy terms and the EEF1 implicit solvation energy term. No cutoff was used for non-bonded interactions.

2.4 Sequence-conformation optimization

The problem of finding the set of rotamers that will optimize the total energy (E_{total}) was modeled as either a binary CFN or an ILP problem. The complete interaction graph and large tree-width excluded the use of dynamic programming (Leaver-Fay *et al*., 2005). The performance of CFN and ILP was compared against that of the combined DEE/A* CPD-dedicated approach.

CFN model A *CFN*, or weighted constraint satisfaction problem P, is composed of a set of local cost functions, each involving a set of specific variables (Schiex *et al*., 1995). CFNs have been used as a modeling framework for representing and solving various combinatorial optimization problems in bioinformatics and resource allocation (Cabon *et al*., 1999; De Givry *et al*., 2006; Zytnicki *et al*., 2008).

Formally, a CFN P is a triple $P = (X, D, C)$, where $X = \{1, 2, \ldots, n\}$ is a set of n variables. Each variable $i \in X$ has a discrete domain $d_i \in D$. C is a set of local cost functions. Each cost function $c_s \in C$ is defined over a subset of variables $S \subseteq X$ (called its scope), has domain $\Pi_{i \in S} d_i$ and takes its values in $N \cup \{+\infty\}$. Cost functions must be non-negative but are otherwise totally arbitrary and are often described by cost tables. The infinite cost is used to represent hard constraints. An assignment A is a mapping from variables to values from their domains. The cost of an assignment A for a local cost function is the value of the cost function for the projection of A to the scope of the function. The global cost of A is the sum of the costs of A over all local cost functions. It is usually assumed that C contains one constant cost function, with an empty scope, denoted as c_0. A CFN P defines a joint cost distribution over all the variables X defined by the cost of the assignments. Because all cost functions in a CFN are non-negative, the constant cost function $c_0 \in C$ defines a lower bound on this joint cost distribution. Solving a CFN consists in finding an assignment that minimizes the joint cost distribution.

Modeling a CPD problem as a CFN is straightforward. Each mutable or flexible residue i is represented by a variable i. The set of rotamers available to the residue defines the domain d_i of the variable i. Finally, each interaction energy term in E_{total} can be represented as a cost function. The constant term E_c is captured as a constant cost function with empty scope. The terms $E(i_r)$, which depend on one residue only, are captured as unary cost functions involving one variable i each. Finally, interaction terms $E(i_r, j_s)$ can be captured as binary cost functions involving variables i and j. To enforce the non-negativity requirement on cost functions, one can simply subtract the minimum of every cost function from its cost table. The joint cost distribution defined by the corresponding CFN is then equal to the energy, up to a known constant shift. The optimal solution of the CFN is an assignment that corresponds to a GMEC for the CPD problem. When the maximum number of available rotamers over all residues is d, the resulting binary CFN takes space in $O(n^2 d^2)$.

Solving a CFN consists in finding a combination of values for all the variables in X that minimizes the joint cost distribution. Such an optimal solution defines a GMEC for the CPD problem. Existing exact algorithms for solving CFN are usually depth-first branch and bound (DFBB) algorithms integrating strong incrementally computed polynomial time lower bounds. The use of depth-first search algorithms avoids the worst-case exponential space of the A* algorithm used in the DEE/A* approach. The incremental lower bounds are produced by algorithms that enforce so-called local consistencies (Larrosa and Schiex, 2004; Schiex, 2000). Enforcing a local consistency on a given CFN P transforms it to an *equivalent* CFN P' (defining the same joint cost distribution) with a possibly increased constant cost function c_0, providing an improved lower bound. This lower bound is used to prune the search tree and to delete rotamers.

To identify the GMEC of all CPD instances, we used the open source *toulbar2* (Toulbar2 is an international collaborative CFN solver development. All sources are available on our software forge at http://mulcyber.toulouse.inra.fr/projects/toulbar2) solver (release 0.9.6.0) with options −d:−l = 3 -m and no initial upper bound. By default, *toulbar2* maintains existential directional arc consistency (De Givry *et al*., 2005) for incremental lower bounding, dynamic value ordering (based on minimum unary cost) and a variable ordering heuristics (based on the median energy of terms involving a given residue following preprocessing) combined with last conflict heuristics (Lecoutre *et al*., 2009). The counterpoint to the improved space complexity of a DFBB search instead of A* is that DFBB search cannot directly provide a sorted list of suboptimal solutions. To enumerate all suboptimal solutions within $E_{\text{cut}} = 2.0\,\text{kcal.mol}^{-1}$ of the GMEC, we first computed the GMEC and its associated energy E_{GMEC} as above and performed a second exhaustive search for all solutions with energy below $E_{GMEC} + E_{cut}$ (options –a and –ub in *toulbar2* to, respectively, produce all solutions and set a global upper bound).

ILP model We also modeled the CPD problem as an integer linear program (01LP) problem using the usual translation from CFN to ILP initially proposed in (Koster *et al*., 1999). An ILP is defined by a linear criteria and a set of linear constraints on integer variables. For every value/rotamer i_r of the variable/residue i, we introduced one Boolean variable $E_{cut}\, d_{i_r}$ that indicates whether the rotamer i_r is used $(d_{i_r} = 1)$or not $(d_{i_r} = 0)$. To enable the expression of the energy as a linear function of variables, we introduced an extra Boolean variable $P_{i_r j_s}$ for every pair of rotamers (i_r, j_s), capturing the fact that this pair of rotamers is used. The energy can then be expressed directly as the linear function to be minimized (the constant term can be ignored as it cannot change the optimal solution):

$$\sum_{i,r} E(i_r).d_{i_r} + \sum_{i,r,j,s} E(i_r, j_s).p_{i_r j_s} \quad (3)$$

Additional constraints enforce that exactly one rotamer is selected for each variable position and that a pair is used if and only if the

corresponding values are used. Then, finding a GMEC reduces to the following ILP:

$$\min \sum_{i,r} E(i_r).d_{i_r} + \sum_{i,r,j,s} E(i_r,j_s).p_{i_r j_s} \quad (4)$$

such that:

$$\sum_r d_{i_r} = 1(\forall i)$$
$$\sum_s p_{i_r j_s} = d_{i_r}(\forall i,r,j)$$

The resulting ILP contains $O(n^2d^2)$ variables and $O(n^2d)$ constraints. It is equivalent to the model proposed for CPD in (Kingsford *et al.*, 2005). Note that because the objective function is non-linear, it is fundamentally impossible to express it in ILP without introducing a quadratic number of variables. Hence, this ILP model cannot be improved significantly in size.

To identify the GMEC, we used the IBM™ ILOG ILP solver *cplex* (version 12.4.0.0) on this ILP with parameters EPAGAP, EPGAP and EPINT set to zero to avoid premature stop.

Dead-end-elimination / A combined approach (DEE/A*)* The DEE algorithm iteratively eliminates rotamers and pairs of rotamers that cannot possibly be part of the GMEC (Desmet *et al.*, 1992) by using a dominance criterion. The original DEE single elimination criterion removes a rotamer r at position i if there exists another rotamer u at the same position such that

$$E(i_r) - E(i_u) + \sum_{j \neq i} \min_s E(i_r,j_s) - \sum_{j \neq i} \max_s E(i_u,j_s) > 0 \quad (5)$$

In this case, r can be removed because any conformation using r can be transformed into a lower energy conformation by substituting u for r. The pruning criterion is applied until a single solution remains (i.e. the GMEC) or all solutions outside an energy window of E_{cut} have been pruned or otherwise when no more pruning is identified during a given round. The DEE pruning step is followed by an A* branch-and-bound like search, which uses the remaining rotamers (Leach *et al.*, 1998) to identify the GMEC or produces a sorted list of all models whose energy is within a specified energy E_{cut} of the GMEC energy. The A* algorithm is a worst-case exponential space and exponential time algorithm whose efficiency is tightly linked to the quality of the heuristic admissible evaluation function used to decide which node to explore next. Interestingly, the heuristic used in the A* approach applied in this study is equivalent to the CFN heuristic used in the PFC-DAC algorithm (Wallace, 1996). This lower bound as well as an improved variant of it (Larrosa *et al.*, 1998) has been obsoleted by the incremental local consistencies introduced in (Schiex, 2000).

In this study, we used *Osprey 2.0* to perform the DEE/A* procedure to find the GMEC and suboptimal models within a $E_{cut} = 2.0\,\text{kcal.mol}^{-1}$ of the GMEC energy (option *initEw* = 2). The procedure starts by extensive DEE (*algoOption* = 3, which enables simple Goldstein, Magic bullet pairs, 1- and 2-split positions, bounds and pairs pruning) and is followed by the A* search. We also optionally applied a procedure including a prefiltering step before the DEE, which eliminates rotamers i_r such that $E(i_r) > 30\,\text{kcal.mol}^{-1}$ and pairs (i_r,j_s) such that $E(i_r,j_s) > 100\,\text{kcal.mol}^{-1}$ (pruning and stericE parameters).

All computations (*toulbar2*, *cplex* and *Osprey*) were performed on one core of an AMD Opteron™ Processor 6176@2.3 GHz. We used 128 GB of RAM and a 100 h time-out.

2.5 Analysis of top-score models

For four design cases (PDB ids: 1TEN, 1UBI, 2PCY, 1CSK), the 3D structure of the best conformation of each unique sequence found within a $2\,\text{kcal mol}^{-1}$ window of the GMEC energy was built using *Osprey 2*. These 3D structure models were then subjected to 1000 steps of minimization with the *Sander* module of *Amber 9,* using the generalized born/surface area implicit solvent model. During these minimizations, heavy atom positions were restrained using a harmonic potential (force constant = $1\,\text{kcal.mol}^{-1}\,\text{Å}^{-2}$). The score of minimized structures was computed with *Osprey 2.0*. Finally, the conformational variability of the minimized models was assessed by carrying out an optimization step allowing all variable amino acids (mutable and flexible) to repack. This step was performed using the CFN-based approach with a E_{cut} of $0.2\,\text{kcal mol}^{-1}$ and involved the pre-computing of pairwise energy terms for each minimized model using *Osprey 2*.

3 RESULTS AND DISCUSSION

3.1 Benchmark set

A tailored benchmark set was produced to assess the performance of combinatorial optimization algorithms on sequence-conformation spaces of various sizes and complexity as well as the potential of the CPD methodology proposed herein (Fig. 1) for tackling the redesign of diverse structural systems involving free proteins or proteins bound to a cofactor, a ligand or a protein. The studied systems have all been extracted from previously published articles about protein engineering, *in silico* protein design or protein structural studies (see references Supplementary Table S1). A detailed description of our benchmark preparation protocol is given in Supplementary Data.

In our benchmark set, the number of mutable residues varies from 3 to 119 (Supplementary Table S1). They are located either in the core of (holo)proteins (except when data are available in the literature) or at the protein–protein or protein–ligand interfaces. We then defined a set of flexible residues (from 1 to 93) (Supplementary Table S1) that surrounds mutable positions and mainly occupies the core and the boundary regions of proteins. The resulting number of variable residues ranges then from 23 to 120 and, given the *penultimate* rotamer library used, from 3 to 194 amino acid rotamers were considered at each variable position. Our resulting benchmark set covers thus a wide range of combinatorial spaces (from 4.10^{26} to 2.10^{249}) and allows us to evaluate different combinatorial optimization problems of varying complexity.

3.2 GMEC-based design

First, we evaluated the performance of CFN and ILP methods for solving the GMEC identification problem exactly and compared them against the exact CPD-specific method DEE/A*. We compared the CFN solver *toulbar2*, the ILP solver *cplex* and the DEE/A* implemented in the *Osprey* software on the benchmark set of 35 design cases (Supplementary Table S1) and present the results in Table 1.

Out of the 35 design cases, the CFN solver *toulbar2* and the ILP solver cplex solved, respectively, 30 and 27 cases within the 100 h time-out, whereas the DEE/A* CPD-dedicated approach identified only 18 GMEC (Table 1).

The DEE/A* method managed to find the 18 GMEC with CPU times ranging from a few minutes to >1 h. Only four cases (1MJC, 1CSK, 1SHF and 1NXB) corresponding to the exploration of small combinatorial spaces on small proteins took <1 min to be solved. The DEE step converged to a single solution in only three instances (1MJC, 1NXB, 1CSE) (Supplementary Table S3). The A* search successfully identified

Table 1. CPU-time for solving the GMEC using DEE/A* (*osprey*), ILP (*cplex*) and CFN (*toulbar2*)

PDB	Sequence conformation space size	Times (s)		
		DEE/A*	ILP	CFN
1MJC	4.36e+26	4.57	3.94	0.08
1CSP	5.02e+30	200.00	360.00	0.84
1BK2	1.18e+32	93.20	303.00	0.65
1SHG	2.13e+32	138.00	42.30	0.25
1CSK	4.09e+32	41.70	24.90	0.15
1SHF	1.05e+34	44.30	11.10	0.17
1FYN	5.04e+36	622.00	2.26e+03	3.79
1PIN	5.32e+39	9.54e+03	3.00e+03	3.99
1NXB	2.61e+41	11.10	21.20	0.24
1TEN	6.17e+43	113.00	81.70	0.33
1POH	8.02e+43	77.90	31.80	0.45
2DRI	1.16e+47	T_{A*}	2.92e+5	24.5
1FNA	3.02e+47	3.31e+03	419.00	0.73
1UBI	2.43e+49	T_{A*}	704.00	2.14
1C9O	3.77e+49	2.31e+03	1.40e+03	2.20
1CTF	3.95e+51	T_{A*}	580.40	1.23
2PCY	2.34e+52	2.08e+03	76.70	0.51
1DKT	3.94e+58	5.42e+03	1.85e+03	3.95
2TRX	9.02e+59	487.00	1.34e+03	1.7
1PGB	5.10e+61	T_{DEE}	T	T
1CM1	3.73e+63	T_{A*}	2.65e+04	19.2
1BRS	1.67e+64	T_{A*}	2.39e+05	426.0
1ENH	6.65e+64	T_{DEE}	T	T
1CDL	5.68e+65	T_{A*}	T	191.1
1LZ1	1.04e+72	T_{A*}	1.25e+03	2.11
2CI2	7.26e+75	T_{DEE}	T	T
1GVP	1.51e+78	T_{A*}	T	593.6
1RIS	1.23e+80	T_{A*}	T	501.00
2RN2	3.68e+80	M_{A*}	1.14e+03	2.77
1CSE	8.35e+82	367.00	205.93	1.36
1HNG	3.70e+88	5.59e+03	4.15e+03	6.97
3CHY	2.36e+92	T_{A*}	2.91e+04	171.00
1L63	2.17e+94	M_{A*}	2.82e+03	6.41
3HHR	2.98e+171	T_{DEE}	M	T
1STN	2.00e+249	T_{DEE}	M	M
Number of cases solved in 10 min		11	13	30
Number of cases solved in 100 h		18	27	30

Note: A 'M' indicates an exceeded memory size (128 GB) and a 'T' indicates an exceeded computation time (100 h). For the DEE/A* approach, the A* and the DEE associated with M or T indicate the step during which occurred the exceeding of memory or computation time.

15 further GMEC using the subset of rotamers remaining after the DEE step. It failed for 12 instances owing to time limit (10 cases) or memory limit (2 cases). In these cases, the size of the combinatorial search space of non-pruned rotamers after the DEE step ranged $\sim 10^{19}$ to 10^{36}. This was not a sufficient reduction to enable A* to extract the GMEC from the subset of remaining rotamers with the available computational resources. Supplementary Table S3 gives fractions of times spent during DEE and A* for each instance. Using *toulbar2* instead of A* after the DEE step allowed to quickly identify the GMEC for the 12 DEE/A* unsolvable instances. A* is therefore the limiting step for these 12 instances. Finally, the DEE step failed to complete within the 100 h time-out for five tests covering initial combinatorial spaces from $\sim 10^{61}$ to 10^{249} and the highest number of variable residues at the surface of the proteins. One can expect the surface residues to be more flexible than the buried residues, and such flexibility may then lead to a high density of conformations with similar energy and a corresponding increase in the complexity of the combinatorial space to be explored. These results clearly underline the limits of the DEE-based approach to handle large and complex problems.

To improve the efficiency of the DEE-based approach, a preprocessing is usually applied to eliminate rotamers and pairs of rotamers of high energy, which are not expected to appear in the GMEC. We then performed such pre-filtering using a 30 kcal mol^{-1} threshold for rotamers and a 100 kcal mol^{-1} threshold for pairs of rotamers. Using these parameters in one instance (1CSE), the optimal solution after preprocessing did not match the GMEC obtained in the absence of this pre-processing step. However, when we increased the threshold for rotamers 30–50 kcal mol^{-1}, the optimal solution remained the same with and without preprocessing. Besides losing the guarantee of optimality, the preprocessing step did not improve the performance of DEE/A* in terms of the number of instances that were solved (18 GMEC identified out of the 35 cases) (Supplementary Table S3). Furthermore, seven instances required more CPU time for identifying the GMEC when preprocessing was used.

The ILP solver *cplex* solved nine additional instances from our benchmark set (2DRI, 1UBI, 1CTF, 1CM1, 1BRS, 1LZ1, 2RN2, 3CHY, 1L63) compared with DEE/A* (Table 1). In all these cases, DEE/A* timed out during the A* search. Among these nine design cases, only two (3CHY and 1L63) cover combinatorial spaces of greater size ($\sim 10^{92}$ and 10^{94}) than the largest combinatorial problem (1HNG $\sim 10^{88}$) solved by DEE/A* (Table 1). The other 18 instances solved by *cplex* are the ones that were successfully solved by DEE/A*. Although the ILP solver was faster in 13 of these 18 instances, the time is overall similar for both methods. Nevertheless, the total number of solved instances shows that the ILP solvers can be more efficient for several design cases, as previously reported (Allouche *et al.*, 2012). More concise quadratic programming (cplex QP solver) and partial weighted MaxSAT (akmaxsat, MaxHS and Bin-Core-Dis solvers) models were also tried, but they all failed on all instances.

The CFN solver *toulbar2* solved, respectively, 12 and 3 more cases compared with DEE/A* and *cplex* (Table 1). Therefore, CFN only failed on five instances (1PGB, 1ENH, 2CI2, 3HHR, 1STN) out of the 35 handled. These instances correspond to vast combinatorial spaces (from about 10^{61} to 10^{249}), which mostly include variable residues scattered over the three layers of the proteins (Supplementary Table S1). There are no cases solved by DEE/A* or ILP that CFN could not solve. Moreover, CFN outperformed the two other approaches by an impressive margin in terms of speed. Among the 30 instances solved by CFN and including large combinatorial spaces, 11 cases were solved in <1 s, 23 in <10 s and only five instances required a few minutes. Given its running time performance and its success rate for handling large protein design problems, the CFN approach appears as an appealing alternative to current exact

CPD-dedicated methods, especially for solving highly complex GMEC-based design problems.

There is no single explanation for the performance advantage of the CFN solver *toulbar2* over the ILP solver *cplex* and the DEE/A* implemented in the *Osprey*. Indeed, solvers are complex systems involving various mechanisms. The effect of their interactions during solving is hard to predict. Moreover, the IBM™ ILOG ILP solver cplex is a totally closed-source industrial black box solver.

Compared with the CFN solver *toulbar2*, *Osprey* uses a lower bound considered as obsolete in CFN (Wallace, 1996) instead of the recent incremental stronger lower bounds offered by soft local consistencies such as EDAC (Larrosa *et al*., 2005). This, together with the associated informed value ordering provided by these local consistencies, explains why the CFN approach outperforms the DEE/A* method. Considering ILP, it is known that the LP relaxation lower bound used in ILP is (by duality) the same as the Optimal Soft AC (Cooper *et al*., 2010) lower bound when no upper bounding occurs. Because OSAC dominates all other local consistencies at the arc level, this provides an explanation for the efficiency of *cplex* compared with *Osprey*. Finally, compared with ILP, the formulation of the deeply non-linear problem is more direct in CFN. This likely contributes, together with the upper bounding, variable and value ordering heuristics of *toulbar2*, to the efficiency of the CFN approach compared with ILP.

3.3 Suboptimal ensemble generation

We also performed computational design tests to assess the ability of the CFN method to generate an ensemble of provably near-optimal sequence-conformation models in addition to the GMEC. The performance of the CFN approach was compared against DEE/A*. For this purpose, the 35 design cases of the benchmark set (Supplementary Table S1) were again investigated using the CFN *toulbar2* solver and the DEE/A* implemented in *Osprey* software to access the set of sequence-conformation models comprised within an energy window of 2 kcal mol^{-1} of the GMEC energy.

Out of the 35 design cases, the CFN solver *toulbar2* managed to produce the suboptimal ensembles of solutions for 30 design cases, whereas the DEE/A* approach only successfully handled one instance (Table 2).

The DEE/A* approach failed for 34 instances owing to time (30 cases) or memory (4 cases) limits. It only identified the set of near-optimal models for one instance (1SHF) among the 18 successfully handled for the GMEC problem (Table 1). Although this solved instance corresponds to one of the smallest investigated combinatorial spaces ($\sim 10^{34}$) (Supplementary Table S1), ~37 h of computation were needed to find the ensemble of low-energy models compared with <1 s for the CFN approach. This running time is even >~7 h required by the CFN method in the worst case (1L63) including an important combinatorial space ($\sim 10^{94}$) and a large number of suboptimal solutions (8×10^8).

The CFN method only failed to identify the near-optimal ensemble on the five instances (1PGB, 1ENH, 2CI2, 3HHR, 1STN) for which the GMEC problem was also unsolved (Table 1). In addition to the high success rate achieved by CFN, the method was also efficient: 10 cases were solved by CFN in <1 min, 15 required several minutes and only five instances needed several hours (Table 2).

Table 2. CPU-time for generating the ensemble of suboptimal models ($E_{cut} = 2$ kcal.mol^{-1}) using DEE/A* (*osprey*) and CFN (*toulbar2*)

PDB	Times (s)		Number of sequence-conformation solutions	Number of different sequences
	DEE/A*	CFN		
1MJC	T_{A*}	42.73	2.11e+06	91
1CSP	M_{A*}	6.29	1.18e+05	794
1BK2	M_{A*}	6.75	3.18e+05	2.19e+03
1SHG	M_{A*}	13.58	6.46e+05	542
1CSK	M_{A*}	9.04	4.48e+05	199
1SHF	1.20e+05	0.99	3.07e+04	26
1FYN	T_{DEE}	14.13	4.05e+05	7.02e+03
1PIN	T_{DEE}	9.36	1.62e+04	310
1NXB	T_{A*}	997.46	4.67e+07	526
1TEN	T_{A*}	123.68	6.42e+06	294
1POH	T_{A*}	1.76e+04	7.56e+08	177
2DRI	T_{DEE}	226.3	4.94e+06	340
1FNA	T_{A*}	1.83e+03	9.87e+07	3.94e+03
1UBI	T_{A*}	12.07	3.04e+05	194
1C9O	T_{DEE}	76.42	3.37e+06	1.65e+03
1CTF	T_{A*}	421.4	1.96e+07	3.06e+04
2PCY	T_{A*}	6.28	2.28e+05	144
1DKT	T_{A*}	2.07e+04	1.00e+09	2.83e+04
2TRX	T_{A*}	2.27e+04	1.01e+09	132
1PGB	T_{DEE}	T	n.d	n.d
1CM1	T_{DEE}	113.53	3.01e+06	5.18e+03
1BRS	T_{DEE}	1.70e+03	2.10e+06	2.09e+04
1ENH	T_{DEE}	T	n.d	n.d
1CDL	T_{DEE}	922.6	1.22e+05	2.54e+03
1LZ1	T_{A*}	251.2	9.87e+06	546
2CI2	T_{DEE}	T	n.d	n.d
1GVP	T_{DEE}	1.50e+04	4.32e+08	3.13e+05
1RIS	T_{DEE}	3.24e+03	1.11e+08	1.24e+04
2RN2	T_{A*}	594.4	2.32e+07	4.00e+03
1CSE	T_{A*}	2.43e+03	8.00e+07	61
1HNG	T_{A*}	3.30e+03	1.08e+08	1.3e+03
3CHY	T_{A*}	385.1	3.52e+06	8.56e+03
1L63	T_{A*}	2.24e+04	8.00e+08	6.03e+03
3HHR	T_{DEE}	M	n.d	n.d
1STN	T_{DEE}	M	n.d	n.d

Note: A 'M' indicates an exceeded memory size (128 GB) and a 'T' indicates an exceeded computation time (100 h). For the DEE/A* approach, the A* and the DEE associated with M or T indicate the step during which occurred the exceeding of memory or computation time. 'n.d' indicates not determined.

While the task of finding a set of low-energy sequence-conformation models proved to be an insurmountable computational hurdle for DEE/A* as implemented in *Osprey*, the CFN solver *toulbar2* successfully solved most of the design cases tested. Moreover, the CFN approach gave access to sets of provably suboptimal solutions with outstanding running time performances.

The CFN approach efficiently uses the knowledge of the GMEC solution in the enumeration procedure of the near-optimal models. The GMEC defines an upper bound corresponding to the energy of the GMEC + 2 kcal mol^{-1}. In CFN, this upper

bound is systematically compared with the lower bound provided by local consistency enforcing. The DEE implemented in *Osprey* uses the same upper bound (parameter pruningE) but exploits a weaker lower bound. This likely explains the performance gap compared with the *toulbar2* CFN solver.

3.4 Suboptimal ensemble analysis

First, we analyzed the sequence and conformational variability of the near-optimal models obtained for four design cases (1CSK, 1TEN, 1UBI, 2PCY) of proteins adopting distinct structural folds (Supplementary Table S2). These instances include from 9 to 16 mutable residues and from 21 to 30 flexible residues (Supplementary Table S1).

Within a window of 2 kcal mol^{-1} of the GMEC, the CFN-based approach produced $>10^5$ sequence-conformation models for each of the four design cases (Table 2). The score of these models is lower by as much as $\sim$20 kcal mol^{-1} than that of the wild-type model (Supplementary Figs S2c–S5c). In these ensembles of models, 144, 194, 199 and 294 unique sequences were found, respectively, for 2PCY, 1UBI, 1CSK and 1TEN design cases.

Only few unique sequences were then generated compared with the high number of enumerated models within a small energy window of 2 kcal mol^{-1} of the GMEC energy. However, when the experimental construction of the protein library is considered, it is important to have access to a larger ensemble of distinct sequences. For this purpose, the outstanding performances of the CFN solver (Table 2) could be harnessed to provably predict suboptimal models distributed on a wider energy window of the GMEC and thus attempt to generate more diverse sequence ensembles.

For the four design cases, the wild-type amino acid was the most often either substituted by an amino acid of slightly larger size or conserved (Supplementary Figs S2a–S5a). Nevertheless, the entire wild-type sequence of the protein was never found within the suboptimal ensembles. It is noteworthy that the mutable residues of glycine type were not found substituted by another amino acid type (Supplementary Figs S2a, S3a and S4a). The number of conformations adopted by each sequence decreases gradually as the energy value of the sequence becomes more unfavorable (Supplementary Figs S2d–S5d). The superposition of the best conformation of each unique sequence showed that each flexible residue adopts almost the same rotamer in all best conformations (Supplementary Figs S2b–S5b). Moreover, the orientation of mutable residue side-chains is similar in all the best models regardless the assigned amino-acid type.

For the protein design problems studied here, we expected that mutations would favor the introduction of bulkier amino acids to fill up the free space available in the core of proteins. However, changes in amino acid sizes were subtle. A visual inspection of the 3D structures of mutants suggests that some slight adjustments of side chains and/or backbone of surrounding residues could enable accommodation of larger side chains, which were here assigned with high interaction energies. This lack of conformational relaxation seems also to be at the origin of the observed conservation of glycine amino acid types. Therefore, the sequence selection may be biased and restricted by the lack of flexibility of surrounding residues. These results highlight the key role of the local molecular flexibility to extend the accessible sequence space, as shown by previous work (Bordner and Abagyan, 2004). We then subjected each unique sequence of the four suboptimal ensembles to energy minimization to assess the effect of the relaxation of side chains and backbone freedom degrees on the energy ranking of the sequence ensembles. Overall, the minimization decreased the energy values of these models from $\sim$20 to 60 kcal mol^{-1} depending on the design case (Supplementary Figs S2c–S5c). Nonetheless, the superposition of the structures before and after minimization only showed slight rearrangements of protein side chains and backbone. This clearly indicates that slight geometrical adjustments can significantly lower model energies. The conformational variability of these low-energy sequences were further investigated by carrying out an optimization step (with a E_{cut} of 0.2 kcal mol^{-1}), which enables all variable amino acids (mutable and flexible) to be repacked. Despite an E_{cut} value, which is 10 times smaller, the number of conformations adopted by each unique sequence was found extremely high whatever the energy ranking of the sequence (Supplementary Figs S2d–S5d). Therefore, the significant differences observed among mutants before minimization step, is probably the result of the discretization of conformational freedom degrees. The minimization step thus allows us to extend the accessible conformational space. Current trends in CPD refine the exact DEE/A* approach along various directions, allowing, respectively, for continuous rotamers (Georgiev *et al*., 2008a,b; Gainza *et al*., 2012), for continuous (Georgiev and Donald, 2007) or discrete (Georgiev *et al*., 2008a,b) backbone conformation adjustments or both (Hallen *et al*., 2013). The CFN approach could further be extended to handle such flexibilities descriptions.

In addition to lowering the energy and increasing the conformational variability of models, the geometry relaxation step reranks the sequence ensemble. The GMEC obtained after minimization (refined GMEC) does not match with the original GMEC. The energy values of the minimized suboptimal models are spread within an energy window up to $\sim$6 kcal mol^{-1} of the refined GMEC (Supplementary Figs S2c–S5c). Therefore, with a E_{cut} of 2 kcal mol^{-1}, from 92 to 185 unique sequences depending on the design case would be removed of these minimized ensembles. Even on a small energy window, these results demonstrate the advantage of the post-minimization to re-rank and post-screen the most promising candidate sequences to evaluate experimentally.

4 CONCLUSION

In this article, we have formulated a novel open-source based computational framework to provably identify the GMEC as well as a set of low-energy protein sequences within the context of atomic protein design. This CFN-based approach provides remarkable speedups, allowing us to explore vast sequence-conformational spaces much more efficiently than the DEE/A* algorithm or state-of-the-art ILP algorithms. Despite the significant change in terms of problem complexity, it is surprising to see that this efficiency extends to the generation of gap-free sets of suboptimal solutions. This article and the companion open-source computational tools we offer will therefore facilitate the optimization of new CPD systems, without requiring expensive computational resources.

Ultimately, we hope that CFN technology will allow complex CPD problems, mixing optimization of flexible systems and

discrete integration (capturing entropic effects and affinity) to be directly tackled.

ACKNOWLEDGEMENTS

We thank the Computing Center of Region Midi-Pyrénées (CALMIP, Toulouse, France) and the GenoToul Bioinformatics Platform of INRA-Toulouse for providing computing resources and support.

Funding: Agence Nationale de la Recherche (ANR 10-BLA-0214, ANR-12-MONU-0015-03); INRA and the Region Midi-Pyrénées (to S.T.).

Conflict of Interest: none declared.

REFERENCES

Allen,B.D. and Mayo,S.L. (2006) Dramatic performance enhancements for the FASTER optimization algorithm. *J. Comput. Chem.*, **27**, 1071–1075.

Allouche,D. *et al.* (2012) Computational protein design as a cost function network optimization problem. In: *Proceedings of Principles and Practice of Constraint Programming–CP 2012, Québec City, QC, Canada*. pp. 840–849.

Althaus,E. *et al.* (2002) A combinatorial approach to protein docking with flexible side chains. *J. Comput. Biol.*, **9**, 597–612.

Archontis,G. and Simonson,T. (2005) A residue-pairwise generalized born scheme suitable for protein design calculations. *J. Phys. Chem. B*, **109**, 22667–22673.

Arnold,F.H. (2001) Combinatorial and computational challenges for biocatalyst design. *Nature*, **409**, 253–257.

Bernstein,F.C. *et al.* (1977) The Protein Data Bank. A computer-based archival file for macromolecular structures. *Eur. J. Biochem.*, **80**, 319–324.

Bordner,A.J. and Abagyan,R.A. (2004) Large-scale prediction of protein geometry and stability changes for arbitrary single point mutations. *Proteins*, **57**, 400–413.

Cabon,B. *et al.* (1999) Radio link frequency assignment. *Constraints*, **4**, 79–89.

Case,D.A. *et al.* (2006) *AMBER 9*. University of California, San Francisco, CA, USA.

Chen,C.Y. *et al.* (2009) Computational structure-based redesign of enzyme activity. *Proc. Natl Acad. Sci. USA*, **106**, 3764–3769.

Cooper,M.C. *et al.* (2010) Soft arc consistency revisited. *Artif. Intell*, **174**, 449–478.

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

De Givry,S. *et al.* (2005) Existential arc consistency: getting closer to full arc consistency in weighted CSPs. In: *Proceedings of 19th International Joint Conference on Artificial Intelligence*. San Francisco, CA, USA, pp. 84–89.

De Givry,S. *et al.* (2006) Mendelsoft: Mendelian error detection in complex pedigree using weighted constraint satisfaction techniques. In: *Proceedings of 8th World Congress on Genetics Applied to Livestock Production*. Vol. 2, Belo Horizonte, Brazil.

Desjarlais,J.R. and Handel,T.M. (1995) De novo design of the hydrophobic cores of proteins. *Protein Sci.*, **4**, 2006–2018.

Desmet,J. *et al.* (1992) The dead-end elimination theorem and its use in protein sidechain positioning. *Nature*, **356**, 539–542.

Desmet,J. *et al.* (2002) Fast and accurate side-chain topology and energy refinement (FASTER) as a new method for protein structure optimization. *Proteins*, **48**, 31–43.

Gainza,P. *et al.* (2012) Protein design using continuous rotamers. *PLoS Comput. Biol.*, **8**, e1002335.

Georgiev,I. and Donald,B.R. (2007) Dead-end elimination with backbone flexibility. *Bioinformatics*, **23**, i185–i194.

Georgiev,I. *et al.* (2008a) The minimized dead-end elimination criterion and its application to protein redesign in a hybrid scoring and search algorithm for computing partition functions over molecular ensembles. *J. Comput. Chem.*, **29**, 1527–1542.

Georgiev,I. *et al.* (2008b) Algorithm for backrub motions in protein design. *Bioinfmatics*, **24**, i196–i204.

Gordon,D.B. and Mayo,S.L. (1999) Branch-and-terminate: a combinatorial optimization algorithm for protein design. *Structure*, **7**, 1089–1098.

Grunwald,I. *et al.* (2009) Mimicking biopolymers on a molecular scale: nano(bio)technology based on engineered proteins. *Philos. Trans. A Math. Phys. Eng. Sci.*, **367**, 1727–1747.

Hallen,M.A. *et al.* (2013) Dead-end elimination with perturbations (DEEPer): a provable protein design algorithm with continuous sidechain and backbone flexibility. *Proteins*, **81**, 18–39.

Hawkins,G.D. *et al.* (1996) Parametrized models of aqueous free energies of solvation based on pairwise descreening of solute atomic charges from a dielectric medium. *J. Phys. Chem.*, **100**, 19824–19839.

Hellinga,H.W. and Richards,F.M. (1991) Construction of new ligand binding sites in proteins of known structure: I. Computer-aided modeling of sites with predefined geometry. *J. Mol. Biol.*, **222**, 763–785.

Hong,E.J. *et al.* (2009) Rotamer optimization for protein design through MAP estimation and problem-size reduction. *J. Comput. Chem.*, **30**, 1923–1945.

Hornak,V. *et al.* (2006) Comparison of multiple Amber force fields and development of improved protein backbone parameters. *Proteins*, **65**, 712–725.

Humphris,E.L. and Kortemme,T. (2008) Prediction of protein-protein interface sequence diversity using flexible backbone computational protein design. *Structure*, **16**, 1777–1788.

Janin,J. *et al.* (1978) Conformation of amino acid sidechains in proteins. *J. Mol. Biol.*, **125**, 357–386.

Kingsford,C.L. *et al.* (2005) Solving and analyzing side-chain positioning problems using linear and integer programming. *Bioinformatics*, **21**, 1028–1036.

Kirschner,K.N. *et al.* (2008) GLYCAM06: a generalizable biomolecular force field. Carbohydrates. *J. Comput. Chem.*, **29**, 622–655.

Koster,A.M.C. *et al.* (1999) Solving frequency assignment problems via tree-decomposition. *Electron. Notes Discrete Math.*, **3**, 102–105.

Kuhlman,B. and Baker,D. (2000) Native protein sequences are close to optimal for their structures. *Proc. Natl Acad. Sci. USA*, **97**, 10383–10388.

Larrosa,J. and Schiex,T. (2004) Solving weighted CSP by maintaining arc consistency. *Artif. Intell.*, **159**, 1–26.

Larrosa,J. *et al.* (1998) Reversible DAC and other improvements for solving Max-CSP. In: *Proceedings of the National Conference on Artificial Intelligence*. Madison, Wisconsin, USA, pp. 347–352.

Larrosa,J. *et al.* (2005) Existential arc consistency: getting closer to full arc consistency in weighted CSPs. In: *Proceedings of the 19th International Joint Conference on Artificial Intelligence*. Morgan Kaufmann Publishers Inc., Edinburgh, Scotland, UK, pp. 84–89.

Leach,A.R. *et al.* (1998) Exploring the conformational space of protein side chains using dead-end elimination and the A* algorithm. *Proteins*, **33**, 227–239.

Leaver-Fay,A. *et al.* (2005) An adaptive dynamic programming algorithm for the side chain placement problem. *Pac. Symp. Biocomput.*, 16–27.

Lecoutre,C. *et al.* (2009) Reasoning from last conflict (s) in constraint programming. *Artif. Intell.*, **173**, 1592–1614.

Lippow,S.M. *et al.* (2007) Computational design of antibody affinity improvement beyond in vitro maturation. *Nat. Biotechnol.*, **25**, 1171–1176.

Nestl,B.M. *et al.* (2011) Recent progress in industrial biocatalysis. *Curr. Opin. Chem. Biol.*, **15**, 187–193.

Pabo,C. (1983) Molecular technology: designing proteins and peptides. *Nature*, **301**, 200.

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

Raha,K. *et al.* (2000) Prediction of amino acid sequence from structure. *Protein Sci.*, **9**, 1106–1119.

Schiex,T. (2000) Arc consistency for soft constraints. In: *Proceedings of Principles and Practice of Constraint Programming–CP 2000, Singapore*. pp. 411–425.

Schiex,T. *et al.* (1995) Valued constraint satisfaction problems: hard and easy problems. *Int. Joint Conf. Artif. Intell.*, **14**, 631–639.

Voigt,C.A. *et al.* (2000) Trading accuracy for speed: a quantitative comparison of search algorithms in protein sequence design. *J. Mol. Biol.*, **299**, 789–803.

Wallace,R.J. (1996) Enhancements of branch and bound methods for the maximal constraint satisfaction problem. In: *Proceedings of the thirteenth national conference on Artificial Intelligence (AAAI-96)*. Portland, Oregon, pp. 188–195.

Wang,J. *et al.* (2004) Development and testing of a general AMBER force field. *J. Comp. Chem.*, **25**, 1157–1174.

Wang,J. *et al.* (2006) Automatic atom type and bond type perception in molecular mechanical calculations. *J. Mol. Graph. Model.*, **25**, 247260.

Wernisch,L. *et al.* (2000) Automatic protein design with all atom force fields by exact and heuristic optimization. *J. Mol. Biol.*, **301**, 713–736.

Zytnicki,M. *et al.* (2008) DARN! A weighted constraint solver for RNA motif localization. *Constraints*, **13**, 91–109.