Bioinformatics
doi.10.1093/bioinformatics/xxxxxx
Advance Access Publication Date: Day Month Year
Manuscript Category

OXFORD

Subject Section

POsitive Multistate Protein *d*esign

Jelena Vucinic[1,2], David Simoncini[1,3], Manon Ruffini [1,2], Sophie Barbe [1*] and Thomas Schiex [2*]

[1]LISBP, Université de Toulouse, CNRS, INRA, INSA, Toulouse, France.
[2] MIAT, Université de Toulouse, INRA, Auzeville-Tolosane, France.
[3] IRIT UMR 5505-CNRS, Université de Toulouse, 31042 Cedex 9, France.

*To whom correspondence should be addressed.

Associate Editor: XXXXXXX

Received on XXXXX; revised on XXXXX; accepted on June 7 2019

Abstract

Motivation: Structure-based Computational Protein design (CPD) plays a critical role in advancing the field of protein engineering. Using an all-atom energy function, CPD tries to identify amino acid sequences that fold into a target structure and ultimately perform a desired function. The usual approach considers a single rigid backbone as a target, which ignores backbone flexibility. Multistate design (MSD) allows instead to consider several backbone states simultaneously, defining challenging computational problems.
Results: We introduce efficient reductions of positive MSD problems to Cost Function Networks with two different fitness definitions and implement them in the Pomp^d (Positive Multistate Protein $d$esign) software. Pomp^d is able to identify guaranteed optimal sequences of positive multistate full protein redesign problems and exhaustively enumerate suboptimal sequences close to the MSD optimum. Applied to NMR and back-rubbed X-ray structures, we observe that the average energy fitness provides the best sequence recovery. Our method outperforms state-of-the-art guaranteed computational design approaches by orders of magnitudes and can solve MSD problems with sizes previously unreachable with guaranteed algorithms.
Availability: `https://forgemia.inra.fr/thomas.schiex/pompd` as documented Open Source.
Contact: `Thomas.Schiex@inra.fr` and `Sophie.Barbe@insa-toulouse.fr`
Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 Introduction

Computational Protein Design (CPD) seeks to identify sequences that adopt a desired tertiary structure with sufficient stability to ultimately perform a desired function. This requires an energy function that accurately reflects protein stability and a reliable search method to identify a sequence with a conformation of optimal stability (Global Minimum Energy Conformation or GMEC). Because of the intractable combination of the many degrees of freedom of a protein and the non-convex form of even the crudest energy functions, this problem has been simplified by several assumptions: the energy is supposed to be described as a pairwise decomposable function, the protein backbone degrees of freedom are fixed to an idealized target backbone and the side-chain of each amino acid is assumed to adopt one of a finite set of possible conformations or rotamers.

Despite these simplifications, the size of the search space remains exponentially large and the problem of searching for a sequence with a minimum energy conformation is known to be decision NP-complete (Pierce and Winfree, 2002). Because of this, most CPD approaches rely on stochastic optimization algorithms such as Monte Carlo Simulated Annealing or Genetic algorithms, which provide only asymptotic convergence guarantees. Recent progress in guaranteed discrete optimization techniques showed that such stochastic methods may durably fail to find or even get close to the GMEC when the problem becomes hard. Despite years of CPU-time, a tuned Simulated Annealing algorithm was unable to find the global energy optima that was identified and proved as optimal by Cost Function Networks (CFN) algorithms (Simoncini *et al.*, 2015; Allouche *et al.*, 2014). The recent design of the hyper-stable self-assembling β-propeller "Ika" by CFN technology (Noguchi *et al.*, 2019) indicates that guaranteed methods can also be useful in practice, combining efficiency with the assurance that optimization did not fail.

In this paper, we aim at combining the guarantees and efficiency of CFN algorithms with the idea of defining the target structure as an ensemble of backbone conformations instead of a single idealized

structure. Indeed, the traditional single state protein design (SSD) contrasts with the increasing evidence that proteins do not remain fixed in a unique conformational state but rather sample conformational ensembles. Compared to the usual SSD approach, multistate design (MSD) has shown to provide enhanced design capacities (Davey and Chica, 2012) to stabilize an ensemble of backbones (Allen *et al.*, 2010), to design conformational switches (Ambroggio and Kuhlman, 2006) or proteins with specific binding properties (Negron and Keating, 2013). In these cases, MSD seeks to identify a sequence that optimizes a function of its optimal energies on the different considered states. This function, or "fitness", is itself non trivial to compute, as it requires the computation of optimal conformations of the sequence on several backbone states. Many SSD optimization algorithms have been extended to MSD, with more or less general fitness functions, including Monte Carlo with simulated annealing (Ambroggio and Kuhlman, 2006), genetic algorithms (Pokala and Handel, 2005), the FASTER approach (Allen and Mayo, 2010), cluster expansion (Negron and Keating, 2013), and dead-end-elimination (Yanover *et al.*, 2007), also in combination with A* (Hallen and Donald, 2016).

The nature of the fitness function intimately depends on the design problem. The Boltzmann-weighted average of the energies in each state is ideal when the aim is to stabilize any of the backbone states. When instead, it is to design a sequence that fits several conformational states, the fitness will typically be the average of the energies on all states. These two cases are identified as *positive* multistate design. The sought sequence tries to maximize stability in all states: the fitness improves when the energy of the sequence improves in any state. However, for some design problems undesirable states are present and this property is violated: the fitness function may worsen when the energy of the sequence in an undesirable state improves. This can occur for the design of protein-ligand binding or oligomeric association specificity. These design problems involve negative design, against unwanted binding partners present in the medium. Specificity then arises from the preference for a given partner over the others. Thus, undesirable (negative) molecular states also have to be considered.

In this paper, we show that the type of the fitness function has a profound influence on the computational nature of the problem. The introduction of undesirable states makes the problem qualitatively more complex, shifting its complexity from NP-complete to the much harder NP^{NP}-complete category (Stockmeyer, 1976). This result has several implications. Negative MSD being qualitatively harder than SSD, optimization methods may become unable to reach good quality solutions sooner than in the SSD case. It also shows that positive MSD is an interesting target: it is "just" NP-complete while capturing some backbone flexibility and dynamics (Davey *et al.*, 2017). Hence, we leverage the polynomial equivalence of NP-complete problems by introducing efficient reductions of two variants of positive multistate design to Cost Function Networks. The first variant uses a minimum energy fitness and the second one a (weighted) average energy fitness. Beyond saving programming efforts, this approach directly benefits from the advanced CFN processing machinery (Cooper *et al.*, 2010; Hurley *et al.*, 2016).

On various positive MSD problems, we show that it is possible to identify an optimal MSD sequence with associated optimal conformations in reasonable time, on computationally extremely challenging design problems of a size far beyond what has been solved with existing state-of-the-art guaranteed multistate design methods (Hallen and Donald, 2016), including recent CFN based methods with dedicated algorithms (Karimi and Shen, 2018). Pomp^d is also natively able to exhaustively enumerate suboptimal sequences close to the MSD optimum, which is convenient for sequence library design. Contrarily to what has been previously described (Allen *et al.*, 2010), we observe that the use of an ensemble of NMR structures as a positive ensemble of backbones provides strong improvements in term of native sequence and sequence similarity recovery when an average energy criteria is used. We also show that this improvement is reduced but still present when a backrub generated ensemble derived from a single X-ray structure is used. These results show that Positive Multistate Design is essentially as hard to solve as Single State Design, both in theory and in practice. Given the significant improvement that the multistate approach brings, positive MSD should be considered as a default design approach when specificity is not the main target.

2 Methods

We use bold letters to denote sequences of objects (e.g. $\mathbf{a}$ for a sequence of amino acids or $\mathbf{c_a}$ for a sequence of conformations for the amino acid sequence $\mathbf{a}$). Each element of a sequence is denoted by a plain letter such as $a \in \mathbf{a}$. The element at position i in a sequence $\mathbf{s}$ is denoted as $\mathbf{s}[i]$.

2.1 Our definitions of multistate design

In discrete rigid MSD, we are given a set of positive backbones that represent the target structure and a set of negative backbones that are undesirable. In either the positive or negative case, these states have also been called "sub-states" (Karimi and Shen, 2018). The final fitness of a sequence is then defined as the difference of the fitness on the positive and negative states. Various definitions of the fitness can be considered:

- If the set of states represents "possible backbones" that the sequence can (de)stabilize, with no prior knowledge on which one will be adopted in practice, the Boltzmann-weighted energy over all the considered states (defined as the sum of $e^{-\beta E}E$, where $\beta = \frac{1}{k_B T}$), is an attractive criteria. Because this gives an exponential advantage to the backbone with lowest energy, it has been approximated by the minimum energy (Karimi and Shen, 2018). This becomes equivalent to what is called Multistate Analysis (MSA) (Davey and Chica, 2017).
- If instead the set of states represents structures that must be jointly (de)stabilized, as in conformational switches design for example, it is important that the energy of every state contributes to the fitness: optimizing the average energy is more adequate.

More formally, we are given a set of positive and negative rigid backbone states $\mathbf{B} = \mathbf{B}^+ \cup \mathbf{B}^-$, all with the same number ℓ of residues. At each position $1 \leq i \leq \ell$, we have a set S_i of possible amino acids. For each $a \in S_i$ and each state $B_j \in \mathbf{B}$, we are given a set $C^j_{i,a}$ of allowed conformations for the amino acid a at position i in state B_j. At position i, a pair $r = (a, c)$ where $a \in S_i$ and $c \in C^j_{i,a}$ is called a rotamer.

We also assume that the energy $E_b(\mathbf{a}, \mathbf{c})$ of a backbone B_b equipped with a given amino acid sequence $\mathbf{a} \in \prod_i S_i$ and conformations $\mathbf{c} \in \prod_i C^j_{i,a[i]}$ is described as a sum of terms that each involve at most two rotamers $r_i = (\mathbf{a}[i], \mathbf{c}[i])$ and $r_j = (\mathbf{a}[j], \mathbf{c}[j])$, for $1 \leq i, j \leq \ell$:

$$E_b(\mathbf{a}, \mathbf{c}) = \left[E_b() + \sum_{1 \leq i \leq \ell} E_b(r_i) + \sum_{1 \leq i < j \leq \ell} E_b(r_i, r_j) \right] \quad (1)$$

To capture the different criteria that have been used, such as minimum or (weighted) average energy, we imagine that a binary operator $\oplus$ is used to combine the energies of the backbones. Optimizing the minimum energy is obtained using $\oplus = \min$. This will be called min-MSD. Since the number of states is fixed, optimizing the average energy is obtained using $\oplus = +$. This will be called Σ-MSD.

More formally, the $\oplus$-MSD problem asks whether there exists a sequence $\mathbf{a} \in \prod_i S_i$ (the sequence design space) such that

$$\left(\bigoplus_{B_j \in \mathbf{B}^+} \min_{\mathbf{c} \in \prod_i C^j_{i,a[i]}} E_j(\mathbf{a}, \mathbf{c}) \right) - \left(\bigoplus_{B_j \in \mathbf{B}^-} \min_{\mathbf{c} \in \prod_i C^j_{i,a[i]}} E_j(\mathbf{a}, \mathbf{c}) \right) \leq k$$

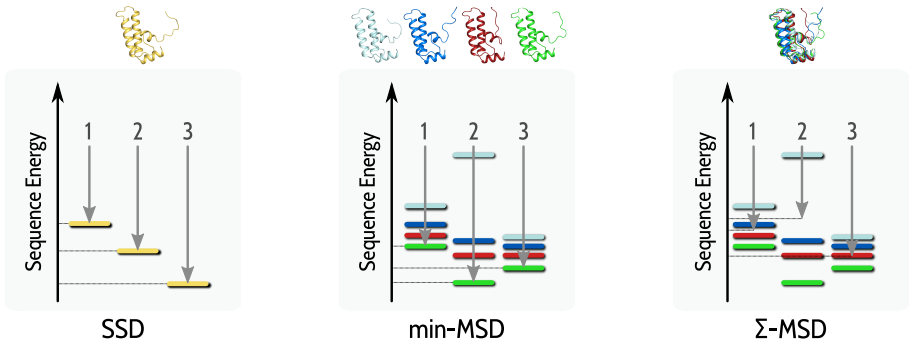

Fig. 1. In SSD (left), a single state (yellow) is used to score and rank sequences (1,2 and 3) according to their energy, which defines the sequence fitness (grey arrow): the best sequence is sequence 3. In $\oplus$-MSD, an ensemble of (here) four backbone states (cyan, blue, red and green) is used to score and rank sequences. The fitness of each sequence (grey arrow) can be computed using the $\min$ (center) or the (weighted) sum of the sequence energies in each state (right). Depending on the used operator, the ranking may change and different sequences can be selected. In $\min$-MSD sequence 2 is ranked first as it has the best energy on the green backbone. In Σ-MSD, it is ranked last because of its bad energy on the cyan backbone.

When the set $\mathbf{B}^-$ is empty, we say that this is a positive $\oplus$-MSD problem. The problem is to identify a sequence $\mathbf{a} \in \prod_i S_i$ (the sequence design space) such that:

$$\left(\bigoplus_{B_j \in \mathbf{B}^+} \min_{\mathbf{c} \in \prod_i C^j_{i,a[i]}} E_j(\mathbf{a}, \mathbf{c}) \right) \leq k$$

.

In this paper, we consider three types of design approaches: SSD, $\min$-MSD (equivalent to MultiState Analysis (Davey and Chica, 2017)) and Σ-MSD. These three approaches are described in Figure 1 showing how different backbones are used to score various sequences in each case.

2.2 Computational Complexity of Multistate Design

Since Pierce's seminal paper (Pierce and Winfree, 2002), we know that the SSD problem is decision NP-complete: given an arbitrary rigid backbone, and arbitrary pairwise decomposable energy function and rotamer library, deciding whether there exists a sequence and associated side-chain conformations with energy lower than a given threshold k is NP-complete. This result proves that the SSD problem is among the hardest of all the problems in its class: any other problem in NP can be reduced to it efficiently (in a time that grows as a polynomial in the size of the problem).

Theorem 1. *Assuming energies are represented as finite objects and that addition, comparison and $\oplus$ can be computed in a time polynomial in the length of their arguments, the positive $\oplus$-MSD problem is* NP*-complete and the general $\oplus$-MSD problem is* NP^{NP}*-complete (or Σ_2^p-complete).*

The proof is given as a Supplementary Information. This theorem says that even if we had an algorithm (also called oracle) that could solve any instance of an NP-complete problem (such as SSD) in *constant* time, then, in the worst case, solving the general MSD problem would still require an exponential number of calls to this oracle (assuming, as it is usual, that the Polynomial Hierarchy does not collapse (Stockmeyer, 1976)).

2.3 Cost function networks

Cost function networks are deterministic Graphical Models derived from Constraint Satisfaction Problems (Schiex *et al.*, 1995), introduced in Artificial Intelligence for automated reasoning (Rossi *et al.*, 2006).

Definition 1. A CFN (X, W, k) is defined by:

- a set X of variables $x_i \in X$ indexed by $I = \{1, \ldots, n\}$, each variable x_i takes its values in a finite domain D_i of maximum cardinality d.
- a set of numerical cost functions $w_S \in W$ each involving a subset $\{x_i \in X \mid i \in S\}$ of all variables.
- The cost k is a finite or infinite upper bound on costs: a cost of k or above is considered as forbidden.

The set $S \subset I$ of a cost function w_S is called the scope of the cost function. We denote by D^S the Cartesian product of the domains of all variables indexed in S: $D^S = \prod_{i \in S} D_i$.

The cost of an assignment t of all variables is defined as the sum $\sum_{w_S \in W} w_S(t[S])$ of all cost functions. If it is strictly less than k, it is said to be a solution. Notice that the upper bound k plays the role as an infinite cost: any assignment with cost k or above is considered as infeasible and is not a solution. The weighted constraint satisfaction problem (WCSP) is to identify a solution of guaranteed minimum cost over all $t \in D^X$.

2.4 Modeling SSD with Cost Function Networks

We model the rigid discrete SSD problem using a CFN (X, W, k) with one variable x_i per position i in the design. The domain of variable x_i is the set of rotamers $(a, c) \in S_i \times C_{i,a}$ available for design at position i and the set of functions W contains the terms of the pairwise decomposable energy functions: a constant term $E()$ for the rigid bodies, one-body terms $E(x_i)$ that capture internal side-chain energies and rotamer-backbone interactions at position i and two-bodies terms $E(x_i, x_j)$ which capture interactions between positions i and j. The objective is to find the combination of rotamers which minimizes the joint cost/energy of the backbone. This is the optimum solution of the WCSP (Allouche *et al.*, 2014; Traoré *et al.*, 2013; Simoncini *et al.*, 2015).

2.5 Positive min-MSD as a Cost Function Network

In positive $\min$-MSD, one seeks a sequence that best stabilizes one backbone among all backbones $B_i \in \mathbf{B}^+$ or equivalently that minimizes:

$$\min_{B_b \in \mathbf{B}^+} \min_{\mathbf{c} \in \prod_i C^j_{i,a[i]}} \left[E_b() + \sum_{1 \leq i \leq \ell} E_b(r_i) + \sum_{1 \leq i < j \leq \ell} E_b(r_i, r_j) \right]$$

where we use the decomposable form of the energy from Equation 1.

This problem can be tackled by solving the SSD problem on every backbone state $B_i \in \mathbf{B}^+$ and using the sequence of the backbone with minimum energy $E_{\min}$ as the solution. Given an energy gap of size $\Delta > 0$, a library of suboptimal sequences whose energy is less than $E_{min} + \Delta$ can be obtained by taking the union of the libraries obtained

with energy threshold Δ on every backbone $B_i \in \mathbf{B}^+$. Because it allows to consider each state independently, this approach is often referred as just a "Multistate Analysis" (MSA) (Davey and Chica, 2012).

Instead of solving as many SSD problems as there are states, the problem can be modelled as a single Cost Function Network whose optimal solution will define both the optimal sequence and the state on which the optimum is reached. This model exploits the fact that CFNs solvers can deal with terms involving more than two variables.

For a set $\mathbf{B}^+$ of n states, we start from a model with the same variables as in the SSD case: one variable x_i per position i, with a domain equal to the set of available rotamers at this position. We also introduce a variable x_B with a domain $\{1, \ldots, b\}$ that represents an index in the set of positive states. The CFN for SSD of any of those backbones involves zero, one and two-bodies terms. We introduce the new variable x_b in the scope of each of these terms so that:

- all the constant terms $E_b()$ for state $B_b \in \mathbf{B}^+$ are transformed in a one-body function depending on the state index x_b and equal to the constant term for this state $E(x_b) = E_{x_b}$.
- for every position i, all the one-body terms $E_b(x_i)$ for states $B_b \in \mathbf{B}^+$ are transformed in two-bodies terms $E(x_b, x_i) = E_b(x_i)$.
- for every pair of positions (i, j), all the two-bodies terms $E_b(x_i, x_j)$ for all states $B_b \in \mathbf{B}^+$ are transformed in three-bodies terms $E(x_b, x_i, x_j) = E_b(x_i, x_j)$.

A solution of the resulting CFN defines a state through its index x_b and a sequence-conformation for every position in x_i. The cost of the solution is, by definition of the terms above, equal to the energy of this sequence-conformation on this backbone. An optimal solution minimizes the energy over all possible choices of states and sequence-conformations and is therefore a solution of the positive $\min$-MSD problem.

This approach was tested but never found to outperform the simple approach where each backbone is solved independently. We therefore used this latter method. The reduction above has the advantage that it simplifies the construction of a sequence library: it suffices to enumerate all suboptimal sequences within Δ of the optimum of this Cost Function Network to directly build the joint library.

2.6 Positive Σ-MSD as a Cost Function Network

In positive Σ-MSD, one seeks a sequence that best simultaneously stabilizes all states $B_i \in \mathbf{B}^+$ or equivalently that minimizes:

$$\sum_{B_b \in \mathbf{B}^+} \min_{\mathbf{c} \in \prod_i C^j_{i,a[i]}} \left[E_b() + \sum_{1 \leq i \leq \ell} E_b(r_i) + \sum_{1 \leq i < j \leq \ell} E_b(r_i, r_j) \right]$$

This problem cannot be tackled by solving the SSD for every state $B_b \in \mathbf{B}^+$ and summing the energies because the optimal sequences for each SSD problem may differ. To avoid this issue, we exploit the capacity of CFNs to represent hard constraints using the cost "k". Contrarily to stochastic search algorithms (that could fail because of lack of ergodicity or require specific treatment to preserve it), CFN algorithms have the capacity to actively exploit these constraints to accelerate search by predicting inconsistent choices using local consistencies (Cooper *et al.*, 2010).

For each state $B_b \in \mathbf{B}^+$, we compute the SSD CFN defined in Section 2.4. We use a superscript for all variables in these CFNs to identify the state they correspond to: x_i^b is the variable representing position i in the SSD CFN of state B_b. We build a Σ-multistate CFN as follow:

- the set of variables of the multistate CFN is the union of all the sets of variables of each SSD CFN. For a positive Σ-MSD full redesign problem with n backbones of length ℓ, there will be $n\ell$ variables, each with the same domain as in the original SSD problems.
- the set of functions of the multistate CFN contains all the cost function $E_b(), E_b(x_i^b), E_b(x_i^b, x_j^b)$ of every SSD CFN plus a set of two-bodies functions $SS(x_i^b, x_i^{b'})$ which, for every position i and every pair of state B_b and $B_{b'} \in \mathbf{B}^+$, enforce that the rotamers used in the states B_b and $B_{b'}$ for position i should represent the same amino acid. $SS(x_i^b, x_i^{b'})$ is equal to zero if x_i^b and $x_i^{b'}$ represent the same amino acid and is equal to the upper bound k in Definition 1 otherwise.

A solution of the multistate CFN contains a solution defining a sequence and conformation for every state $B_b \in \mathbf{B}^+$. By definition, the cost of this solution is the sum of all energy terms over all states. Additionally, the $SS(x_i^b, x_i^{b'})$ functions impose that the same sequence is used in all states: an optimal solution defines a sequence that minimizes the sum of energies. It therefore solves the positive Σ-MSD problem.

This generates a CFN with $n\ell$ variables, $n\frac{\ell(\ell+1)}{2}$ energy terms and $\ell\frac{n(n-1)}{2}$ additional $SS(x_i^b, x_i^{b'})$ constraints. Since the $SS(x_i^b, x_i^{b'})$ constraints define an equivalence relation, transitivity implies that it is sufficient to only enforce this constraint for every pair $b, b+1$ of successive states. This requires $\ell(n-1)$ constraints instead of $\ell\frac{n(n-1)}{2}$.

This reduction is used in the rest of the paper to solve positive Σ-MSD: the MSD problem is transformed in a CFN and the CFN solved. The use of a single CFN also allows to easily generate suboptimal sequences using the dedicated SCP-branching strategy (Traoré *et al.*, 2016).

2.7 Benchmark Preparation

Two datasets have been prepared. The first one contains 15 NMR structures and the second one 15 X-ray structures (see Table S1) that have been extracted from the Protein Data Bank (PDB) (Berman *et al.*, 2000) and filtered with following criteria:

- monomeric proteins, no missing or nonstandard residues, no ligand
- maximum sequence length of 100 amino acid residues
- NMR resolved structures must contain at least 20 conformations
- X-ray structures must be resolved below 2 Å

The set of backbones in the NMR ensemble was submitted to RMSD-based hierarchical clustering using the Durandal software (Berenger *et al.*, 2012) in order to select the four most diverse conformations.

The X-ray ensembles have been generated by RosettaBackrub (Davis *et al.*, 2006) which uses the BackrubEnsemble method for flexible protein backbone modeling in Rosetta (Friedland *et al.*, 2009; Humphris and Kortemme, 2008). One hundred conformations were generated for each structure. This step was followed by the same RMSD-based hierarchical clustering as for the NMR ensembles in order to select the four most diverse among given conformations. Clustering distance thresholds were set to reach the desired number of clusters (see Table S2). The structures were relaxed using RosettaFastRelax with harmonic constraints, resulting in output structures which are typically within 2 Å RMSD of the initial structure. Pairwise energy matrices were computed with Dunbrack2010 rotamer library (Shapovalov and Dunbrack, 2011) and `beta_nov16` scoring function (Alford *et al.*, 2017), using PyRosetta 171 (Chaudhury *et al.*, 2010). These problems define huge search spaces (Table S3) with sizes that can exceed 10^{900} or 10^{540} if the effect of the SS constraints is taken into account.

We also used the 4 multistate problems provided with the multistate iCFN solver at `https://shen-lab.github.io/software/iCFN`. All 4 problems include eleven states of 3QDJ, a complex between TCR DMF5 and human Class I MHC HLA-A2 with a bound MART-1(27-35) nonameric peptide, produced by an MD simulation (Karimi and Shen, 2018). Each problem has a single residue to redesign (from 20 possible amino acids, with 7 protonation states for Asp, Glu and His), all close

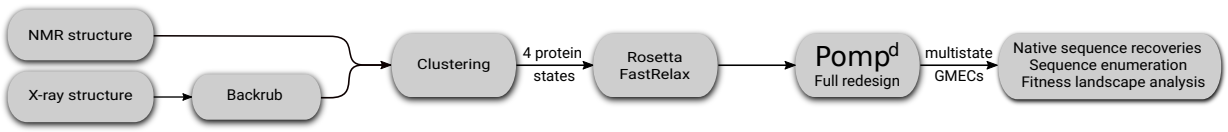

Fig. 2. Overall workflow for Xray and NMR structures.

residues are considered as flexible. Because of a dense rotamer library ($4,731$ rotamers), these problems define large search spaces (Table S4).

2.8 Solving SSD, min-MSD and Σ-MSD with Pomp^d

Thanks to the three reductions of SSD, positive min-MSD and Σ-MSD to CFNs, it is now possible to solve these problems using a CFN solver such as toulbar2 (`https://github.com/toulbar2/toulbar2`).

For our benchmarking NMR and X-ray instances, we downloaded toulbar2 from its repository, using its 'cpd' branch. All instances were solved using the "`-dee: -O=-3 -B=1 -A -cpd`" taken in a recent paper (Simoncini *et al.*, 2015). Compared to the default behavior, this command line deactivates Dead End Elimination and activates the exploitation of the interaction structure (treewidth) and the strong 'Virtual Arc Consistency' bounds (Cooper *et al.*, 2010). Computations were done on an Intel(R) Xeon(R) CPU E5-2630 at 2.30GHz with 24GB of RAM. The overall workflow is described in Figure 2.

3 Results and Discussion

3.1 Comparing SSD, min-MSD and Σ-MSD

Protein design problems can be modeled as SSD, min-MSD or Σ-MSD. SSD has the advantage of simplicity: there is only one backbone to design. MSD approaches have the advantage of accounting for protein backbone flexibility, with additional modeling and computing costs. As we already mentioned, min-MSD seems more suitable for situation of uncertainty: it is not known which, among all the available backbones, is the suitable one. Instead Σ-MSD seems more suitable when there is an explicit requirement that all states should be stabilized.

We assessed Pomp^d on our benchmark backbone conformational state ensembles, either extracted from NMR structures or generated by backrub motions from X-ray structures. Notice that our benchmark dataset represents a selection of full protein design problems for structures of size varying between 53 and 96 residues.

Σ-MSD outperforms SSD and min*-MSD in terms of sequence recovery* In order to compare the accuracy of these methods, we used the native sequence recovery (*nsr*) and native sequence similarity recovery (*nssr*), which have been used extensively to evaluate protein design methods (Havranek *et al.*, 2004; Humphris and Kortemme, 2007; Löffler *et al.*, 2017). Native sequence recovery is defined as the fraction of positions where the native and designed sequences are identical. Native sequence similarity is defined as the fraction of positions where the native and designed sequences have a positive similarity score in BLOSUM62 protein similarity matrix. For SSD, *nsr* and *nssr* have been computed as the average of the recovery for the four SSD conformations. The results of these comparisons are shown in Table 1. Σ-MSD achieves on average a *nsr* of 64.7% and 66.4% and a *nssr* of 74.4% and 73.9% for respectively back-rubbed X-ray and NMR structure datasets. For every protein design in the X-ray structure dataset and for 13 out of the 15 protein designs in the NMR structure dataset, Σ-MSD provides the best native sequence recovery (p-value when comparing to respectively average SSD and min-MSD over all proteins of $2.5\ 10^{-6}$ and $1.3\ 10^{-5}$, Wilcoxon signed rank test). For NMR structures, Σ-MSD performs 15.6% better on average

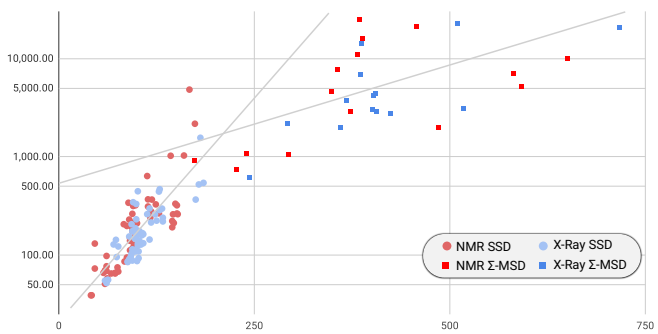

Fig. 3. CPU time in seconds (Y logscale axis) vs. problem size (MB) for SSD and Σ-MSD problems (X axis). Each point represents one instance, NMR structures are in red, X-ray in blue. SSD problems are represented as circles, Σ-MSD problems as squares. A related figure with the X axis showing protein size is available as Figure S1.

than SSD and 8% better for X-ray structures. min-MSD achieves native sequence recovery rates which can almost not be differentiated from those obtained by SSD (p-value of 0.6 on the 30 proteins, Wilcoxon signed rank test). While min-MSD and SSD achieve a better sequence recovery rate on the X-ray dataset than on the NMR structures ($7-9\%$ better on average), Σ-MSD is less sensitive to the dataset type (1% better on average on X-ray dataset).

We expected Σ-MSD to perform better on NMR, given that the NMR ensemble corresponds to likely states of the observed proteins and min-MSD to be more adapted to the back-rubbed X-ray structures that just define a set of possible states. Instead, Σ-MSD dominates even in the back-rubbed case. Instead, min-MSD is worse than SSD on NMR ensembles but at least improves over SSD on back-rubbed X-ray structures. It is possible that a set of 4 states is too small for min-MSD to have a chance to find a suitable backbone while the more consensual approach of Σ-MSD is able to extract local information from every backbone.

Analyzing the efficiency of Pomp^d *on SSD and* Σ*-MSD:* Because SSD and ⊕-MSD are NP-complete, we expect an exponential cpu-time growth as the sizes of the problems solved increase. We plotted the cpu-time taken by Pomp^d to solve the SSD and Σ-MSD problems against the problem size represented as the size (in bytes) of the compressed file that contains the description of the problem solved in `wcsp` format (see Figure 3). Empirically, we observe that for each class of problem (SSD and Σ-MSD), an exponential function fits the CPU-time reasonably well and that the Σ-MSD problems tend to be simpler to solve than the SSD problems, given their larger size. In the end, the relatively slow increase in CPU time as the size grows shows that full redesign problems using an SSD or a positive min-MSD and Σ-MSD approach can be solved on a standard computer for proteins of size less than 100 amino acids in reasonable time, with guarantee on the fitness of the produced sequence.

Comparing the computational efficiency of iCFN and Pomp^d We compared Pomp^d to the recent iCFN solver (Karimi and Shen, 2018). iCFN can solve min-MSD problems but not Σ-MSD problems. In our first comparison, we converted the 4 positive multistate problems available on the iCFN web site (see SI) to a format that we could process. We tackled

Table 1. Native sequence recoveries and similarity recoveries for SSD, min-MSD and Σ-MSD on both NMR structure (left) and X-ray structure (right) datasets. The protein sequences have length that vary from 53 to 96.

NMR structures				X-ray structures			
PBD ID	$\overline{SSD}$	min-MSD	Σ-MSD	PBD ID	$\overline{SSD}$	min-MSD	Σ-MSD
	Native sequence recoveries				Native sequence recoveries		
5vso	48.0%	44.0%	62.7%	1hyp	61.8%	67.5%	68.9%
5l7b	52.9%	53.6%	59.5%	1hoe	60.1%	59.4%	77.0%
5mmc	54.2%	60.0%	58.5%	1mjc	56.5%	53.6%	59.4%
1gb1	48.7%	46.4%	60.7%	1pga	52.7%	58.9%	73.2%
5t8a	39.7%	39.3%	37.7%	2b8i	48.4%	42.8%	57.1%
2l5t	58.4%	49.3%	83.1%	4y2k	62.7%	67.7%	70.8%
2n6r	53.9%	52.6%	67.1%	1f94	64.3%	65.1%	71.4%
1bmw	53.4%	56.4%	79.8%	1who	61.4%	62.7%	68.1%
5ix5	52.6%	48.5%	63.2%	1tud	45.8%	45.0%	48.3%
5x0s	42.9%	50.9%	58.5%	1yu5	64.1%	65.7%	68.6%
6ews	46.8%	46.1%	69.8%	1bxy	53.3%	50.0%	60.0%
6fwn	55.8%	54.1%	72.9%	1ctf	57.9%	50.7%	60.9%
6qf8	54.3%	51.3%	68.4%	1guu	53.4%	66.6%	66.6%
6jlt	54.9%	57.8%	65.7%	1wvn	41.9%	44.5%	50.0%
6hkc	45.0%	44.0%	66.6%	1ucs	66.1%	70.3%	70.3%
Average	50.8%	50.3%	66.4%	Average	56.7%	58.0%	64.7%
	Native sequence similarities				Native sequence similarities		
5vso	58.6%	54.6%	70.6%	1hyp	71.3%	75.7%	81.1%
5l7b	69.6%	65.2%	75.4%	1hoe	67.2%	68.9%	82.4%
5mmc	59.6%	63.1%	61.5%	1mjc	67.5%	66.7%	69.6%
1gb1	62.9%	60.7%	67.8%	1pga	62.9%	69.6%	80.3%
5t8a	52.5%	52.5%	49.2%	2b8i	63.3%	58.4%	71.4%
2l5t	71.1%	62.3%	90.9%	4y2k	66.1%	69.2%	75.4%
2n6r	64.5%	59.2%	73.7%	1f94	74.2%	73.0%	80.9%
1bmw	64.9%	64.9%	84.1%	1who	72.1%	73.4%	80.9%
5ix5	62.8%	58.8%	72.1%	1tud	55.0%	56.7%	55.0%
5x0s	51.9%	60.4%	75.5%	1yu5	72.4%	73.1%	74.6%
6ews	65.8%	73.0%	82.5%	1bxy	64.5%	58.3%	73.3%
6fwn	63.2%	61.1%	78.8%	1ctf	65.6%	59.4%	68.1%
6qf8	64.4%	64.4%	77.6%	1guu	69.6%	72.5%	82.4%
6jlt	62.5%	61.8%	71.1%	1wvn	57.1%	63.5%	62.2%
6hkc	59.3%	58.6%	78.6%	1ucs	73.0%	76.6%	78.1%
Average	62.2%	61.4%	73.9%	Average	66.8%	67.7%	74.4%

the min-MSD problem on these four instances with 11 states with iCFN and Pomp^d. Since sequence recovery was shown to be better on Σ-MSD, we also tried to solve Σ-MSD problem with Pomp^d only. This is also the criteria that COMETS (Hallen and Donald, 2016) uses.

The results are presented in Table 2. We observe that Pomp^d is much faster than iCFN, by a non constant factor that increases with problem size. Furthermore, the Σ-MSD variant can also always be solved in reasonable time by Pomp^d despite the vast search spaces (See Table S4). A possible explanation for this surprising capacity to explore vast spaces of size larger than 10^{440} is that the several backbones in each problem are sufficiently similar to define correlated regions of low energies that enable both quick identification of optimal sequences and fast optimality proof. To check if this intuition is true, we computed, for each protein in the benchmark set, the difference ΔE between the average energy of the SSD optimal sequences ($\overline{SSD}$) and the optimal average energy provided by Σ-MSD (see Table S5). With an average of respectively $19.2\ kcal.mol^{-1}$ and $10.5\ kcal.mol^{-1}$ for respectively NMR and back-rubbed X-ray structures, these exact differences show that the Σ-MSD sequences have higher energies than the SSD sequences: there is a non negligible frustration generated by trying to fit all backbones together. This frustration is also more important for NMR structures than X-ray structures (p-value = 0.03, Wilcoxon rank sum test) indicating that the back-rubbed structures are more compatible with each other energy-wise than the NMR structures.

Table 2. Comparison of the CPU-times (in seconds) for iCFN and Pomp^d for solving min-MSD and for Pomp^d to solve the corresponding Σ-MSD.

redesigned position	iCFN min-MSD	Pomp^d min-MSD	speedup	Pomp^d Σ-MSD
26	445.4	25.7	17.3	55.4
28	594.9	32.7	18.1	99.9
98	640.3	22.7	28.2	89.6
100	719.8	29.5	24.4	105.1

We also converted our benchmarking problems to a format suitable for iCFN min-MSD algorithm. After 65 hours of computing, none of the full-redesign min-MSD problems could be solved by iCFN. This was even the case for the smallest protein of our dataset (PDB id: 1pga) which is solved by Pomp^d in less than 20 minutes. We therefore prepared several design problems with increasingly smaller search spaces by decreasing the number of mutable amino acid residues, leaving non-mutable residues as flexible. With a number of mutable residues reduced to 5, iCFN was still unable to provide a solution after 24 hours. It's only after fixing all non-mutable residues in a rigid position that iCFN could finally produce a solution in 247 seconds. Pomp^d solves this problem in 14.59 seconds.

3.2 Sequence enumeration for min-MSD and Σ-MSD

In addition to the optimal sequence, Pomp^d can provide an exhaustive list of sub-optimal sequences within a given energy threshold of the MSD optimum. In order to characterize the energy landscape of the min and Σ-MSD approaches, we enumerated all sequences within a 1 $kcal.mol^{-1}$ of the optimum for the largest protein of our dataset (96 amino acid residues) whose structure has been solved by both NMR (1bmw) and X-ray crystallography (1who). As expected, Σ-MSD enumerations are computationally more costly than min-MSD enumerations (Table 3).

Table 3. Number of enumerated sequences and CPU-time taken for the enumeration for 1who and 1bmw

	min-MSD		Σ -MSD	
	# of seq.	CPU-time	# of seq.	CPU-time
1bmw	131,616	2' 50"	94,522	43'30"
1who	56,790	2'32"	143,457	67'16"

Different important features of the fitness landscape of SSD problems have already been studied in (Simoncini *et al.*, 2018). We used some of these features to analyze the landscapes of min-MSD and Σ -MSD. The distribution of the Hamming distances to the optimal sequence (number of substitutions compared to the optimum) shows a similar uni-modal distribution for both methods (Figure 4). However, Σ-MSD shows a narrower distribution, with more solutions close to the optimum (mode at distance 5 of the optimum instead of 7 and 10 respectively for 1bmw and 1who in min-MSD). These results are consistent with the *nsr* and *nnsr* computed for all enumerated sequences (average values shown in Table 4).

We also computed the local optima network defined by the enumerated sequences and a neighborhood at a Hamming distance of 1 (See Figure 5). For both proteins, the networks for the Σ-MSD landscapes are much more densely connected than the min-MSD networks. In min-MSD, the basin

Table 4. Average nsr and nssr over all enumerated sequences.

	nsr(%)		*nssr*(%)	
PBD ID	min-MSD	Σ-MSD	min-MSD	Σ-MSD
1who	62.9%	67.9%	74.7%	80.3%
1bmw	55.6%	79.5%	63.2%	84.3%

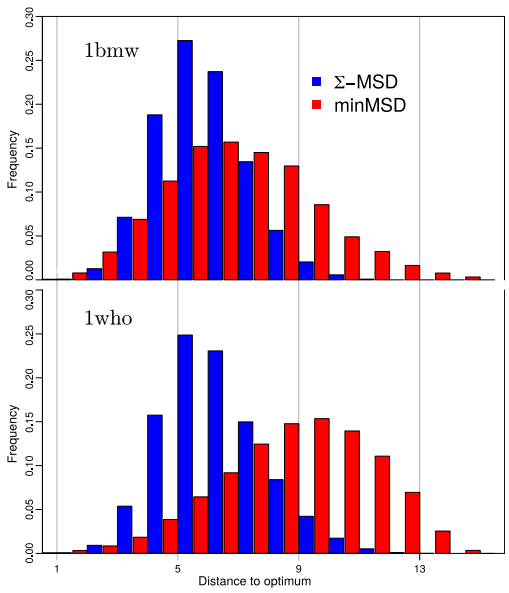

Fig. 4. Distribution of Hamming distances to GMEC for 1bmw (top) and 1who (bottom). min-MSD is shown in red and Σ-MSD in blue.

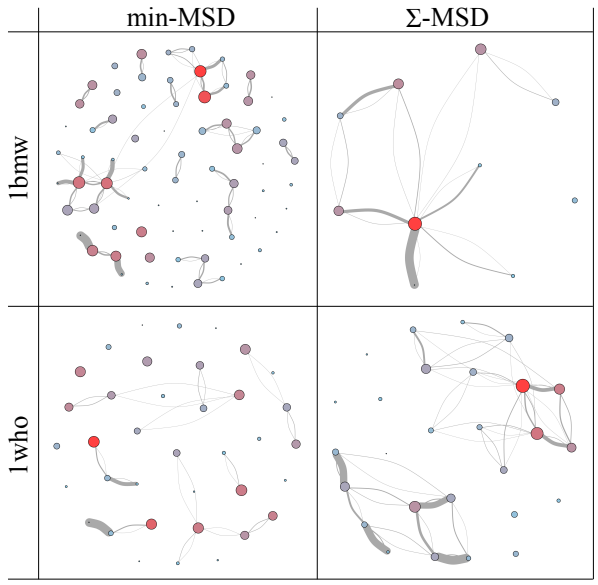

Fig. 5. 2D views of the local optima networks of 1bmw and 1who for min-MSD and Σ-MSD. The size of a node is proportional to the log size of the attraction basin of the local minima. The energy of the local minima is represented as a color gradient from blue (high energy) to red (low energy). Edge thickness is proportional to the probability of escaping a basin to another basin assuming that the probability to go from a solution to any of its neighbors is uniform. A 3D-view is available in the SI (Figure S2).

of the global optimum is often disconnected from most of the other basins. Instead, the Σ-MSD landscapes show far less wider basins which can be reached by all or a large fraction of the other basins. This may explain why, despite the frustration generated by the requirement of fitting several backbones, Σ-MSD are easier to solve given their size: they clearly more identify the globally optimal sequence.

From a biological perspective, the Σ-MSD fitness landscapes seem more relevant. Considering that evolution occurs by random mutations, one can interpret these networks as an abstract representation of the possible mutational paths that can be explored by evolution. The densely connected Σ-MSD local optima networks allow random mutations to easily escape local minima. By capturing the natural flexibility of proteins in a more realistic manner, Σ-MSD leads to more natural fitness landscapes.

4 Conclusion

In this paper, we have shown that multistate protein design problems can be intrinsically much harder than the usual NP-complete Single State Protein Design problem (Pierce and Winfree, 2002). This additional complexity can be precisely pinned down to the introduction of negative states. While negative states are crucial when the design target is to generate specificity, when the aim is just to stabilize an ensemble of backbones, or to design conformational switches, positive states suffice. The positive MSD problem is therefore a soft spot of multistate protein design, offering the ability to capture some of the flexibility of protein backbones while remaining "only" NP-complete, just as SSD.

To exploit this situation, we designed efficient reductions of the optimization problem defined by positive MSD problems to the generic discrete optimization framework of Cost Function Networks (Cooper *et al.*, 2010), a framework introduced in Artificial Intelligence that has already shown its efficiency on SSD problems (Traoré *et al.*, 2013; Simoncini *et al.*, 2015). On a mixture of RMN and back-rubbed X-ray structures, Pomp^d shows that the average energy criteria is clearly superior to the MSA approach in terms of native sequence recovery. In terms of efficiency, it also outperforms a very recent guaranteed multistate algorithm such as iCFN (Karimi and Shen, 2018), which is also restricted to the simple min-MSD (or MSA) problem. In our knowledge, this is the first time that it is possible to access guaranteed optimal average energy full multistate redesigns of proteins of size close to 100 amino acids, defining search space of size larger than 10^{500}. Because it just relies on a reduction to CFN, this approach also inherits all the capabilities of CFN solvers such as toulbar2, including the ability to exhaustively enumerate sequences within a threshold of the optimum to directly produce a sequence library.

References

Alford, R. F. *et al.* (2017). The rosetta all-atom energy function for macromolecular modeling and design. *Journal of chemical theory and computation*, **13**(6), 3031–3048.

Allen, B. D. and Mayo, S. L. (2010). An efficient algorithm for multistate protein design based on faster. *Journal of computational chemistry*, **31**(5), 904–916.

Allen, B. D. *et al.* (2010). Experimental library screening demonstrates the successful application of computational protein design to large structural ensembles. *Proceedings of the National Academy of Sciences*.

Allouche, D. *et al.* (2014). Computational protein design as an optimization problem. *Artif. Intell.*, **212**, 59–79.

Ambroggio, X. I. and Kuhlman, B. (2006). Computational design of a single amino acid sequence that can switch between two distinct protein folds. *Journal of the American Chemical Society*, **128**(4), 1154–1161.

Berenger, F. *et al.* (2012). Durandal: fast exact clustering of protein decoys. *Journal of computational chemistry*, **33**(4), 471–474.

Berman, H. M. *et al.* (2000). The protein data bank. *Nucleic acids research*, **28**(1), 235–242.

Chaudhury, S. *et al.* (2010). Pyrosetta: a script-based interface for implementing molecular modeling algorithms using rosetta. *Bioinformatics*, **26**(5), 689–691.

Cooper, M. C. *et al.* (2010). Soft arc consistency revisited. *Artificial Intelligence*, **174**(7-8), 449–478.

Davey, J. A. and Chica, R. A. (2012). Multistate approaches in computational protein design. *Protein Science*, **21**(9), 1241–1252.

Davey, J. A. and Chica, R. A. (2017). Multistate computational protein design with backbone ensembles. In *Computational Protein Design*, pages 161–179. Springer.

Davey, J. A. *et al.* (2017). Rational design of proteins that exchange on functional timescales. *Nature chemical biology*, **13**(12), 1280.

Davis, I. W. *et al.* (2006). The backrub motion: how protein backbone shrugs when a sidechain dances. *Structure*, **14**(2), 265–274.

Friedland, G. D. *et al.* (2009). A correspondence between solution-state dynamics of an individual protein and the sequence and conformational diversity of its family. *PLoS computational biology*, **5**(5), e1000393.

Hallen, M. A. and Donald, B. R. (2016). Comets (constrained optimization of multistate energies by tree search): A provable and efficient protein design algorithm to optimize binding affinity and specificity with respect to sequence. *Journal of Computational Biology*, **23**(5), 311–321.

Havranek, J. J. *et al.* (2004). A simple physical model for the prediction and design of protein–dna interactions. *Journal of molecular biology*, **344**(1), 59–70.

Humphris, E. L. and Kortemme, T. (2007). Design of multi-specificity in protein interfaces. *PLoS computational biology*, **3**(8), e164.

Humphris, E. L. and Kortemme, T. (2008). Prediction of protein-protein interface sequence diversity using flexible backbone computational protein design. *Structure*, **16**(12), 1777–1788.

Hurley, B. *et al.* (2016). Multi-language evaluation of exact solvers in graphical model discrete optimization. *Constraints*, **21**(3), 413–434.

Karimi, M. and Shen, Y. (2018). iCFN: an efficient exact algorithm for multistate protein design. *Bioinformatics*, **34**(17), i811–i820.

Löffler, P. *et al.* (2017). Rosetta: Msf: a modular framework for multi-state computational protein design. *PLoS computational biology*, **13**(6), e1005600.

Negron, C. and Keating, A. E. (2013). Multistate protein design using clever and classy. In *Methods in enzymology*, volume 523, pages 171–190. Elsevier.

Noguchi, H. *et al.* (2019). Computational design of symmetrical eight-bladed β-propeller proteins. *IUCrJ*, **6**(1).

Pierce, N. A. and Winfree, E. (2002). Protein design is np-hard. *Protein engineering*, **15**(10), 779–782.

Pokala, N. and Handel, T. M. (2005). Energy functions for protein design: adjustment with protein–protein complex affinities, models for the unfolded state, and negative design of solubility and specificity. *Journal of molecular biology*, **347**(1), 203–227.

Rossi, F. *et al.* (2006). *Handbook of constraint programming*. Elsevier.

Schiex, T. *et al.* (1995). Valued constraint satisfaction problems: hard and easy problems. In *Proc. of the 14th IJCAI*, pages 631–637, Montréal, Canada.

Shapovalov, M. V. and Dunbrack, R. L. (2011). A smoothed backbone-dependent rotamer library for proteins derived from adaptive kernel density estimates and regressions. *Structure*, **19**(6), 844–858.

Simoncini, D. *et al.* (2015). Guaranteed discrete energy optimization on large protein design problems. *Journal of chemical theory and computation*, **11**(12), 5980–5989.

Simoncini, D. *et al.* (2018). Fitness landscape analysis around the optimum in computational protein design. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 355–362. ACM.

Stockmeyer, L. J. (1976). The polynomial-time hierarchy. *Theoretical Computer Science*, **3**(1), 1–22.

Traoré, S. *et al.* (2013). A new framework for computational protein design through cost function network optimization. *Bioinformatics*, **29**(17), 2129–2136.

Traoré, S. *et al.* (2016). Fast search algorithms for computational protein design. *Journal of computational chemistry*, **37**(12), 1048–1058.

Yanover, C. *et al.* (2007). Dead-end elimination for multistate protein design. *Journal of computational chemistry*, **28**(13), 2122–2129.