Protein Design Based on Parallel Dimensional Reduction

Germán Moltó,[†] María Suárez,[‡,§] Pablo Tortosa,[‡] José M. Alonso,[†] Vicente Hernández,[†] and Alfonso Jaramillo*[,‡,§]

Departamento de Sistemas Informáticos y Computación, Universidad Politécnica de Valencia, 46022 Valencia, Spain, Epigenomics Project, Genopole-Université d'Évry Val d'Essonne-CNRS UPS 3201, 91034 Évry, France, and Laboratoire de Biochimie, École Polytechnique-CNRS UMR 7654, 91128, Palaiseau, France

Received December 17, 2008

The design of proteins with targeted properties is a computationally intensive task with large memory requirements. We have developed a novel approach that combines a dimensional reduction of the problem with a High Performance Computing platform to efficiently design large proteins. This tool overcomes the memory limits of the process, allowing the design of proteins whose requirements prevent them to be designed in traditional sequential platforms. We have applied our algorithm to the design of functional proteins, optimizing for both catalysis and stability. We have also studied the redesign of dimerization interfaces, taking simultaneously into account the stability of the subunits of the dimer. However, our methodology can be applied to any computational chemistry application requiring combinatorial optimization techniques.

INTRODUCTION

Computational protein design has achieved remarkable breakthroughs[1] by considering the inverse folding problem: the identification of sequences able to fold on a predetermined three-dimensional structure and that exhibit high activity or stability.[2] Furthermore, the use of physicochemical inspired models has advanced our knowledge on the biophysical interactions governing processes such as protein structure,[3−6] protein−protein interactions,[7,8] or DNA−protein interactions.[9] In addition, the computational design of new biocatalysts,[10−14] the design of biosensors for non-natural molecules,[15] the redesign of improved protein binding affinity,[16] and the redesign of protein binding specificity[17,18] have opened new avenues for biotechnological and biomedical applications.[19,20]

The common approach to the inverse folding problem[21−23] relies on the precomputation of the interaction energies among the amino acids, in their different conformations, forming the possible sequences. We may compute different energies and use them as scoring functions (i.e., folding free energies, binding free energies, etc.), according to the goal of the designing procedure: thermostability, ligand binding affinity, protein−protein interaction, or DNA−protein binding specificity. Once we have computed the energies, we collect them into energy matrices for their optimization through different techniques: Monte Carlo Simulated Annealing (MCSA),[24] Dead End Elimination,[25] Branch and Bound,[26] or Genetic Algorithms.[27] The best suited methods to treat combinatorial problems of an ever increasing size are those based in heuristic approaches such as MCSA.

Protein design methodologies rely on the assumption that we will be able to perform a suitable exploration of the space of sequences, through the use of adequate combinatorial optimization methods, to find the optimal sequences. Nevertheless, whenever we are confronting the problem of designing functional proteins, our optimization problem will have two objectives since we want our protein to be both stable and functional. As a result, we will have to consider at least two scoring functions: one representing stability and another scoring function to account for the desired functionality of the protein.

Multiobjective searches are thus required to treat more complex problems (specificity design, improvement of binding affinity, or introduction of a new enzymatic activity) while maintaining the overall foldability of the considered protein. The development of multiobjective algorithms,[28,29] able to consider more than one interaction matrix at the same time, has led to an increase of the computational memory requirements. Moreover, the CPU time requirements of the optimization phase grow accordingly with the size of the designed protein. Therefore, Grid based projects (folding@home, rosetta@home[30,31]) have been developed to face this computational complexity. Nevertheless, these approaches still have to deal with the fact that each node must have enough available memory to load the complete matrices in order to perform the optimization.

In this paper, we propose a High Performance Computing approach that can benefit the protein optimization procedure to overcome these two difficulties. First of all, the memory requirements are distributed among different processors, thus allowing the tackling of larger proteins which, at the moment, cannot be optimized in a sequential platform. In addition, multiple processes can collaboratively optimize a single protein to increase the exploration of the search space. Our approach enables each different process to optimize part of the global problem: each one works with all the protein

* Corresponding author e-mail: alfonso.jaramillo@polytechnique.edu.
[†] Universidad Politécnica de Valencia.
[‡] École Polytechnique.
[§] Epigenomics Project.

10.1021/ci8004594 CCC: $40.75
Published on Web 05/07/2009

positions but only with a subset of rotamers. Therefore, it is possible to distribute the energy matrix data among different processors.

To systematically check the performance of our optimization algorithm we have generated our own set of matrices of varying sizes, with similar characteristics to the ones derived from natural structures. Additionally, we have applied our optimization algorithm to the full redesign of thioredoxin from *Escherichia coli* (PDB code 2TRX, 1.5 Å resolution, 108 residues)[32] for stability and ligand binding affinity, which in this case is related to catalytic activity. To test our algorithm with a bigger protein we have considered a dimer, the 4-oxalocrotonate tautomerase from *E. coli* (PDB code 1GYX, 1.35 Å resolution, 76 positions at each chain),[33] and we have redesigned it for the stability of each chain (first objective) and to increase the stability of the complex (second objective).

METHODS

Protein Design. We expect that molecular dynamics computations will be able to predict the structure of a protein from its amino acid sequence. In addition, we could also address the inverse question, that is, which sequences would fold into a given three-dimensional structure. Computational protein design addresses the inverse folding problem to design proteins with targeted properties.[34]

Our automatic protein design software DESIGNER[5,17] is entirely based on physicochemical principles and is devoted to the resolution of the inverse folding problem. As such, it requires as initial data a high-resolution atomic protein structure, and then it analyzes possible sequences stabilizing the given fold. In the proposed sequences, we can allow either all the side chains or only a subset of them to vary.

For each considered sequence, DESIGNER computes the folding free by scoring the free energy of the folded state and of a reference state, the unfolded state. DESIGNER constructs detailed atomic models of both the folded and the unfolded states. In the folded state a side chain may adopt different conformations, and each conformation (rotamer) can be defined by the value of its inner dihedral angles. Some rotamers are more much more frequent than others in naturally occurring proteins, so we use a library that for each residue in a given backbone contains the most frequently appearing rotamers.[35] The molecular mechanics tasks (minimization and energy evaluation) are done using a standard molecular mechanics force field (CHARMM[36]). CHARMM uses empirical energy functions to describe the forces between atoms in molecules. The model for the reference or unfolded state is built assuming that the amino acids do not interact and form a distribution modeled by a gas of dipeptides. The folding free energy is the difference between the energies of these two states and is used as the energetic score for the stability objective.

Protein stability has to be considered throughout the designing process. However, to optimize the protein to achieve a particular functionality, we may describe this functionality in terms of some energy such as ligand−protein interaction energy or protein−protein interaction energy. Therefore, this new free energy will become a second scoring function for the optimization process.

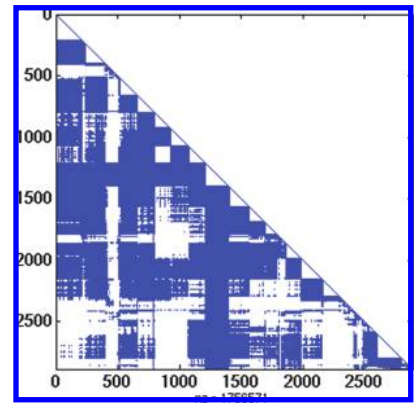


Figure 1. Example of a matrix of energetic interaction among rotamers, where the nonzero elements are shown. The protein consists of 101 positions and 2886 rotamers. The total number of elements in the matrix is 4165941, with 1756571 nonzero elements (a 42% considering the lower triangular part of the matrix).

Table 1. Abbreviations List

abbreviation	meaning
n_l	number of local iterations
N_G	number of global iterations
P	number of designed positions of the protein
N_{R_p}	total number of rotamers at position p
N_{r_p}	number of considered rotamers at position p
d	dimensional reduction factor $N_{r_p} = (N_{R_p})/(d)$
C	number of processors
R	$1.987(cal)/(K \cdot mol)$

The solvation energies are computed with an implicit solvation model based on atomic accessible surface areas, with coefficients taken from atomic hydration experimental data.[37] In the energetic computations, DESIGNER introduces a pair wise approximation so that at most two different residue conformations are considered at a time.[38] The solvation model based on accessible surface areas is specially suitable for the pair wise approximation. As each interacting pair can be evaluated independently, it is possible to perform a trivial parallelization to efficiently use multiprocessor systems. As a result, we obtain, for each objective, an energy matrix storing the interaction energy of the different pairs. In addition, the energy matrix contains the "single" energies in the diagonal. The single energy of a rotamer is computed considering only its interaction with the backbone of the protein and the nondesigned positions, assuming that the rest of the designed positions are devoid of their corresponding side chains. When we are considering stability, the single energy terms include the energy of the corresponding amino acid in the unfolded state, also called the reference energy.

Protein Optimization: Dimensional Reduction and Parallel Optimization. The energy matrices are square, symmetric matrices, whose dimension is the total number of considered rotamers (in the order of tens of thousands). These matrices are sparse: with a large number of zero elements corresponding to noninteracting rotamers (a cutoff is introduced so that the interaction energy of rotamers 15 Å away is not computed). An example is shown in Figure 1. As a result, efficient storage techniques can be employed to only assemble the nonzero elements,[39] thus reducing the amount of required memory. However, for large proteins, even using this memory-saving technique, the large amount of data may

overwhelm the capacity of a single PC. Our approach enables each different process to optimize only a part of the global problem, distributing the energy matrix data among different processors. Figure 2 summarizes the principal steps of the proposed optimization procedure, while Table 1 includes a list with the abbreviations employed.

In the initial phase, for a given position, p, each process randomly chooses N_{r_p} rotamers from the N_{R_p} allowed rotamers. Initially there is an equiprobable probability distribution, $\Pi^0_{G_p}(i)$ of having rotamer i in position p:

$$\Pi^0_{Gp}(i) = \frac{1}{N_{R_p}} \tag{1}$$

The memory consumption in each processor is thus limited by $\sum_{p=1}^{p} N_{r_p}$, since each process only loads from the disk the corresponding section of the energy matrix required to create the data structures employed during the optimization procedure.

After the initial phase, the following procedure is iterated N_G times.

i) Each processor performs a **local optimization**, using the assigned rotamers, following a MCSA and performing n_l iterations. We use the standard Metropolis algorithm[40] to accept/reject solutions following an exponential cooling schedule from an initial temperature T_0 to the final $T_f/R =$ 0.01 kcal/mol. The cost function we consider in the MCSA is the sum of the scoring functions for each objective. However, it is also possible to include an additional weight to give more preeminence to one objective with respect to the other.

Each processor α hosts a local vector (F^{α}_{Li}) containing information of the frequency of appearance of rotamer i. Each time a mutation is accepted in any of the n_l steps, the corresponding rotamer's counter is increased.

ii) The local frequency vectors from each of the C processors are collected into a global frequency vector F_{Gi}. Then, we transform F_{Gi} into a probability distribution, available to all processors.

$$\begin{aligned} F_{Gi} &= \sum_{\alpha=1}^{C} F^{\alpha}_{Li} \\ \sigma &= C \cdot n_l \\ \Pi_{i_p} &= \frac{F_{Gi}}{\sigma} \end{aligned} \tag{2}$$

Each rotamer is associated with a given position, and Π_{i_p} is the probability to select rotamer i associated with position p, arising from the local optimization just performed. This newly gathered information is combined with the global rotamer probability distribution obtained in the previous iteration ($\Pi^{prev}_{G_{i_p}}$):

$$\Pi^{new}_{Gi_p} = \frac{\Pi^{prev}_{G_{i_p}} + \Pi_{i_p}}{\sum_{i=1}^{N_{R_p}} (\Pi^{prev}_{G_{i_p}} + \Pi_{i_p})} = \frac{\Pi^{prev}_{G_{i_p}} + \Pi_{i_p}}{2} \tag{3}$$

Thus, after each iteration the new global probability for a given rotamer is the average between the old global probability and the new local probability. A different weighting factor for $\Pi^{prev}_{G_{i_p}}$ and Π_{i_p} could be used to perform this averaging. A too high weight given to the old global probability will increase the number of iterations needed to erase the effect of the random initial distribution. On the other hand, a too high weight given to the new local probability will risk destroying the long-run quality of the global probabilities.

iii) Each process selects N_{r_p} different rotamers for the p^{th} designed position according to the new rotamer probability distribution. The main problem arising in this phase consists of obtaining n different samples out of a population of size N, with $n \leq N$. To discard repeated samples while maintaining the integrity of the probability distribution, we use a modification of the DSS (Discrete Sequential Search)[41] algorithm. This algorithm involves modifying the probability of a chosen rotamer to 0, so that it does not get selected again.

Notice that the only communication among the processors is performed in the second step of each global iteration,

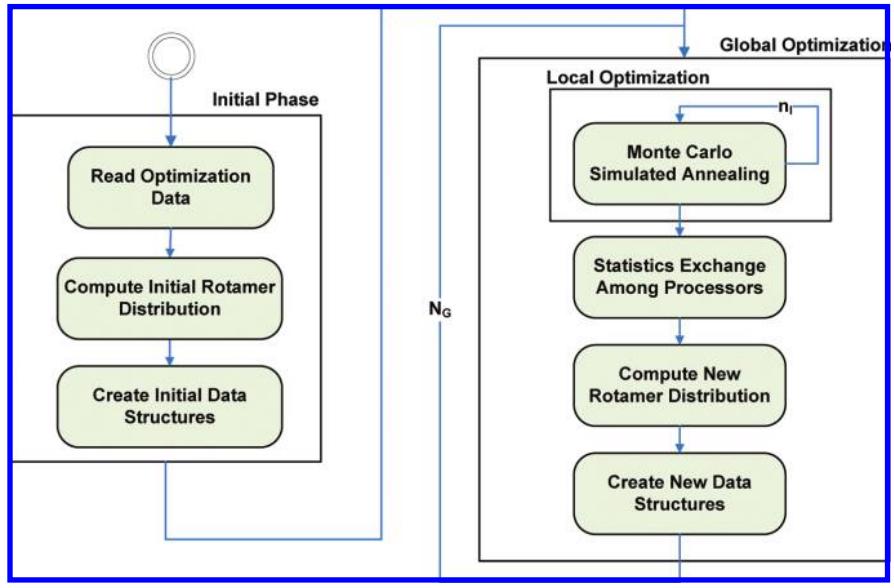


Figure 2. Algorithm of the proposed optimization procedure using dimensional reduction and parallel computing.

where the local frequency vectors are collected and used to refine the rotamers probability distribution. In addition, a collaborative approach is implemented, in contrast to a master-slave approach, where all the processors send the information to a chosen one, who performs the sum and later distributes the result. This enables one to take advantage of improved vendor-supplied multicast implementations of the Message Passing Interface[42] (MPI) operations employed in the algorithm implementation.

Arbitrary Size Matrices Generation. To test the optimization procedure we produced artificial energy matrices with structure and energy values similar to the ones for natural protein backbones. The advantage of this procedure was to save a huge amount of CPU time, since we did not need to perform the actual computation of physical interactions in the protein. In addition, we were able to construct matrices with an arbitrary number of positions (P) and rotamers at each position (N_{R_p}), with known minimum energy.

An initial analysis of natural protein scaffolds,[5] showed the following:

i) Sequences minimizing the total energy have pair energies distributed forming a wide Gaussian, centered at $\mu = -0.5$ kcal/mol. Their single energies are in the $(-4, -5)$ kcal/mol range.

ii) Rotamers not belonging to the optimal sequence have pair and single energies distributed in Gaussians centered at 0 and -2 kcal/mol, respectively.

We assumed a globular model for the 3D structure of the protein (volume proportional to P^3), and we classified every position into one of the following types: *core, surface*, or *intermediate*. According to our globular model, the number of *surface* positions was proportional to $P^{2/3}$, the number of positions in the *core* was 80% of the remaining positions, and all the rest belonged to the *intermediate* category. Each position interacted with 10/6/4 positions depending on its classification (core/intermediate/surface). We have assumed this distribution to mimic naturally occurring proteins. This distribution has been estimated assuming amino acids to be able to interact within a 5 Å sphere so that they may be at most 10 Å. In the core each residue is surrounded by others, so that our spheres model will allow for 10 amino acids to surround it. In the surface or in the intermediate region the residues are not completely surrounded so they have less interacting partners. Then, we constructed the energy matrices using the following rules:

- Rotamers belonging to the minimum energy sequence had a -4.0 kcal/mol single energy.
- Interaction energy between rotamers in the minimizing sequence was -1.0 kcal/mol.
- The single energies of rotamers not in the minimum energy sequence took random values following a Gaussian distribution with mean $\mu_s = -2.0$ and variance $\sigma_s = 0.4$, respectively.
- The energy of the pairs with at least one rotamer not from the minimum energy sequence randomly took values following a Gaussian distribution with $\mu_p = -0.0$ and $\sigma_p = 0.5$.

The μ and σ values were adjusted by optimizing small matrices generated using these rules with a previous implementation of heuristic optimization[26] and comparing the convergence with the results obtained for natural proteins[5] (data not shown). It is easy to see that increasing the value of σ_p and σ_s will worsen the convergence of the optimization. The chosen sequence minimizes the energy, as long as the number of rotamers is high enough to apply the central limit theorem.

RESULTS

Arbitrary Size Matrices. *Dimensional Reduction Performance.* In the first test we performed, we considered twice the same objective, and the goal was to obtain the sequences minimizing the energy of the proteins whose energy interactions are stored in the previously generated matrices. To obtain the number of considered rotamers at each position (N_{r_p}), we can take advantage of the fact that our matrices have the same total number of rotamers (N_{R_p}) at each position, so we divide N_{R_p} by a fixed factor, d, which represents the dimensional reduction factor.

i) Without dimensional reduction: in this case only one process was used that considered the whole ensemble of rotamers, performing thus a standard MCSA procedure, with $n_l = 10^5$. The initial temperature, $T_0/R = 1$ kcal/mol, decreased at each step following $T \rightarrow T((0.01)/(T_0))^{(1)/(n_l)}$. The matrices we used in this test had a uniform rotamer distribution: $(N_{r_p} = N_{R_p} = 100, \forall p)$.

ii) With dimensional reduction: the same set of parameters in the local optimization phases was used. In addition, 5-processor executions with 30 global iterations (N_G) were performed. We considered different values for the dimensional reduction factor (d).

In Figure 3 we compare the minima attained using both approaches with the energy of the minimizing sequence as it was obtained during the matrix construction process. Using the value $d = 2$, we obtained a very accurate optimization when compared to the case with no dimensional reduction. Notice that using a value of $d = 2$ means that each processor works with half the total number of rotamers at each position. If the value of d is increased, then the chance of achieving the global minimum by the algorithm is reduced.

Variation of Local Optimization Parameters. To adjust the different parameters involved in the optimization procedure, we have to consider the following constraints:

$$n_l \gg P \cdot N_{r_p} = P \cdot \frac{N_{R_p}}{d} \tag{4}$$

$$N_G \cdot C \gg d \tag{5}$$

Equation 4 guarantees that n_l, the number of iterations in each local optimization, is higher than the number of considered rotamers, so a suitable exploration of the chosen rotamers may be performed. In addition, to enforce a suitable exploration of the whole rotamers space, we might naively impose $N_G \cdot C \cdot n_l \gg P \cdot N_{R_p}$, but this expression will only be valid if no dimensional reduction were involved. Instead, eq 5 imposes that the global iterations will erase the effect of the distribution of rotamers between different process.

To further measure the effect of the variations of d in the convergence of our optimization method we considered a 500 positions protein (with a total of $5 \cdot 10^4$ rotamers) and used 10 processors. We considered eq 4 and fixed $n_l = 10 \cdot P \cdot (N_{R_p})/(d)$, thus obtaining the values shown in the table of Figure 4.

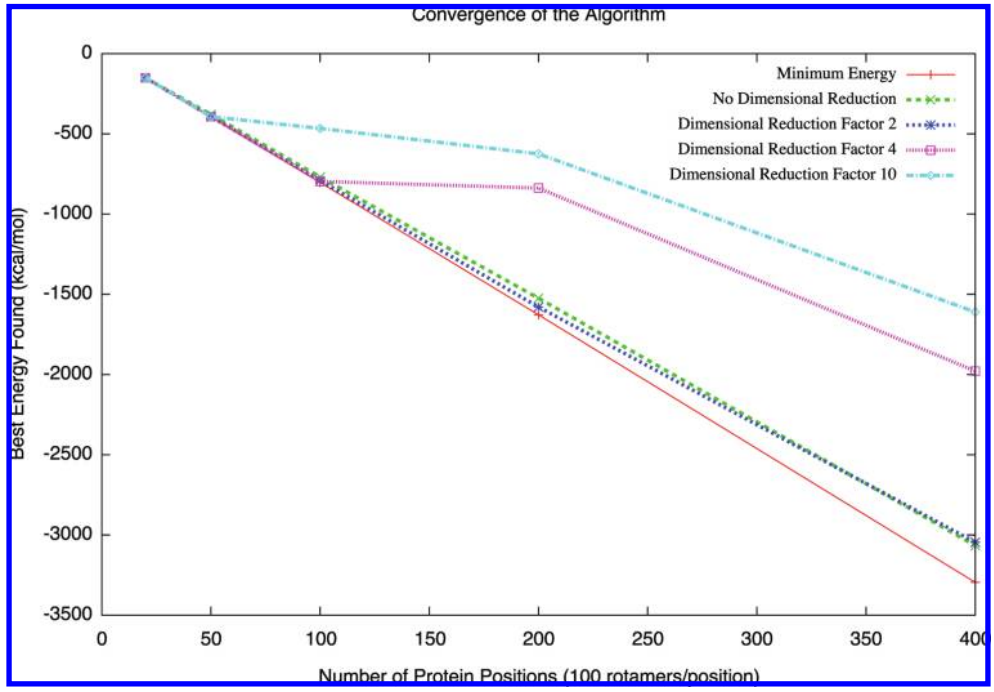
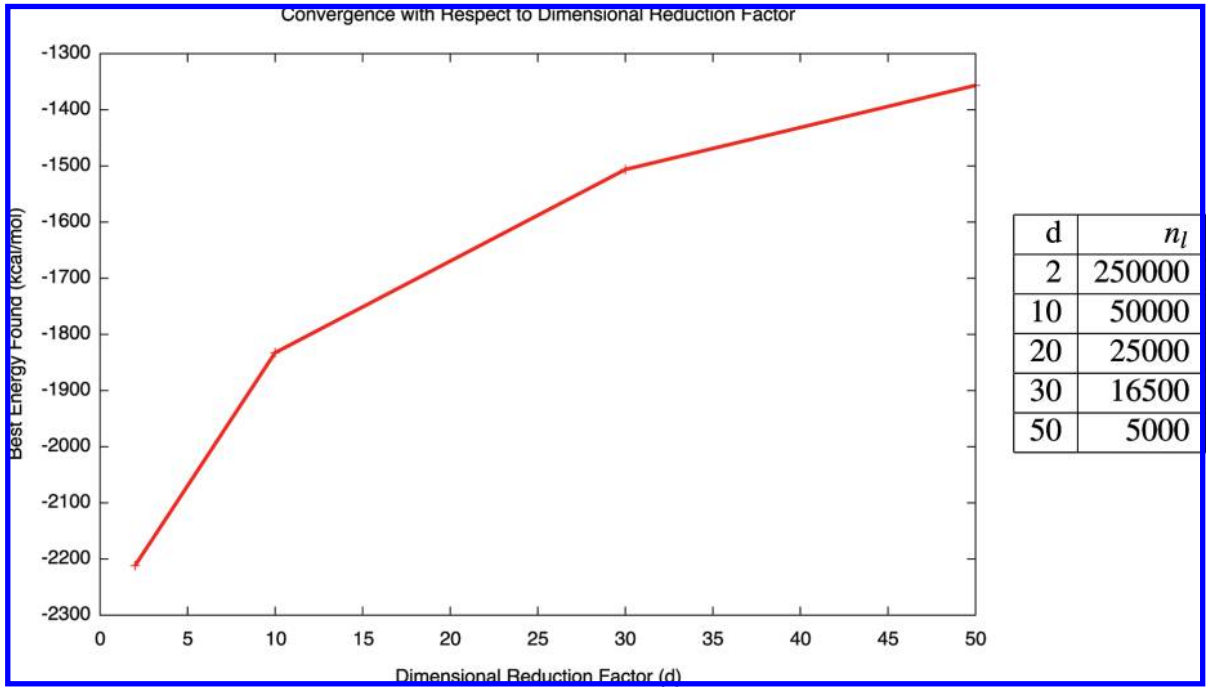


Figure 3. Convergence of the optimization procedure with and without dimensional reduction. In every case we used $n_l = 10^5$. We used 5 processors for the cases with dimensional reduction.



d	n_l
2	250000
10	50000
20	25000
30	16500
50	5000

Figure 4. Minimum energy obtained altering the dimensional reduction factor. The table shows the combination of dimensional reduction factor and number of local iterations executed.

We kept $N_G = 30$ fixed and analyzed the convergence of the method varying d. In this and the following studies, we were not interested in obtaining the minimum energy solution for this problem, which we knew to be −4137 kcal/mol. Therefore, the chosen optimization parameters are not adequate to obtain this solution, instead they were chosen to analyze their effect on the global performance of the method without an excessive CPU resource consumption.

Figure 4 shows the impact of modifying the dimensional reduction factor (d) in the minimum energy obtained during the optimization process. According to the results, increasing d reduces the ability of the algorithm to find a sequence of rotamers with reduced energy. In fact, the dimensional reduction factor enables the algorithm to work with a dynamic subset of the total rotamers. Therefore, this reduces the chance of obtaining the best sequence of rotamers that produce the minimum energy.

Figure 5 shows the effect of modifying the total number of global iterations in the convergence of the algorithm. Executions have been carried out with 10 processors, a 500-positions protein, 33000 local iterations, and a dimensional reduction factor of 30. Different optimizations have been performed with increasing values of the number of global iterations from 10 to 75. For low values of N_G the best energy

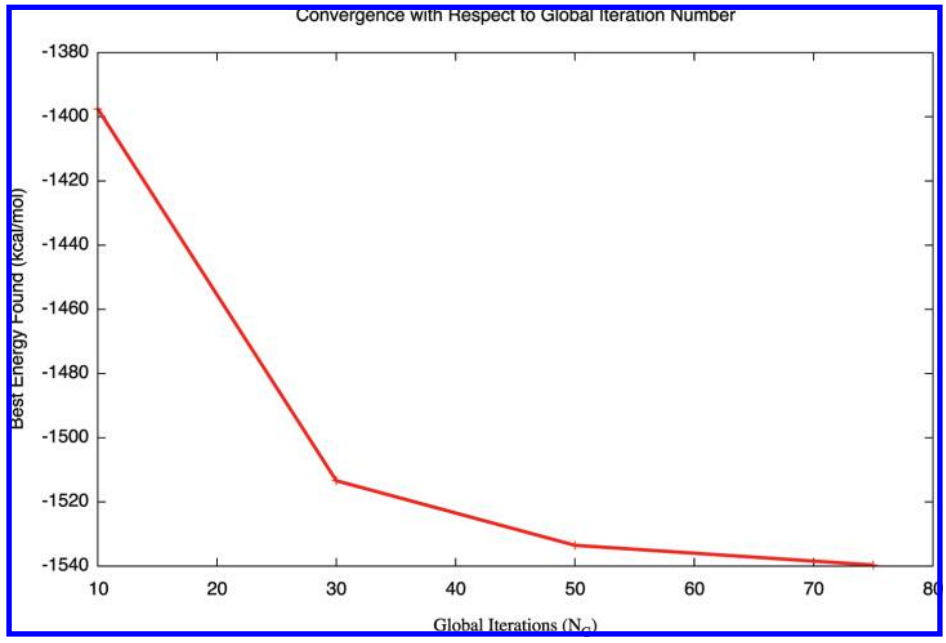


Figure 5. Minimum energy obtained altering the number of global iterations.

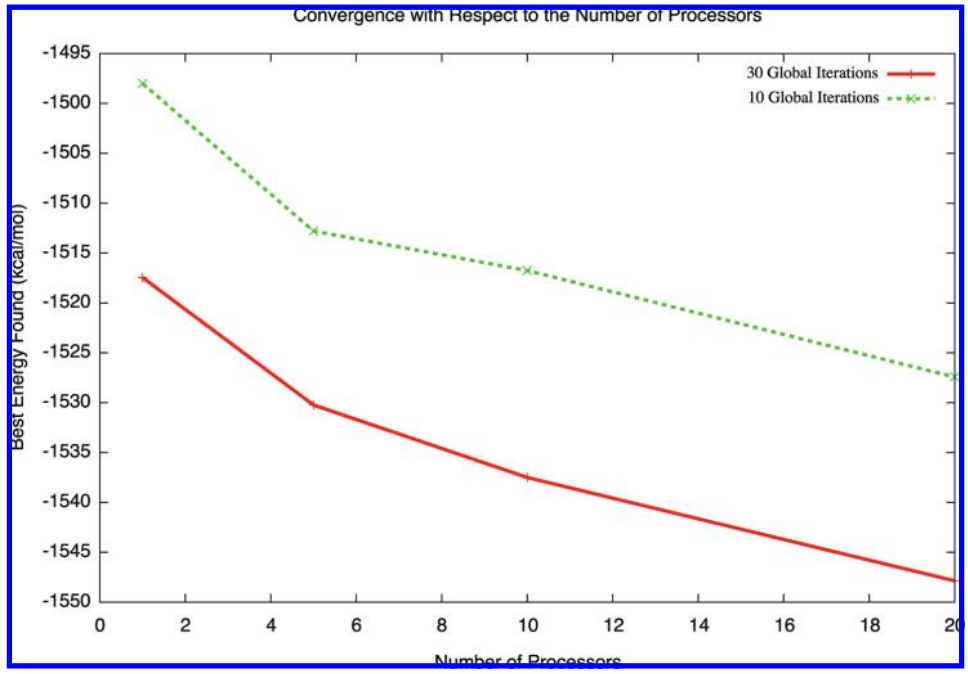


Figure 6. Minimum energy obtained altering the number of processors.

found rapidly decreases with N_G, while for higher N_G values it still decreases but less steeply. As the different processors optimize collaboratively the protein, the exchange of information after each global iteration allows for the increase in the chance to obtain a better optimized protein. However, the increased cost of a large number of global iteration should be considered when evaluating the benefits of this approach. For these particular executions, each global iteration requires 32.71 min on a cluster of Xeon 2.0 Ghz PCs with 1 GByte of RAM each.

Figure 6 shows the impact of varying the number of processors in the convergence of the algorithm. Executions were performed with the same configuration described for the previous figure ($N_{R_p} = 100$, $n_l = 33000$, $d = 30$), using both $N_G = 10$ and $N_G = 30$. It can be shown that increasing the number of processors allows the algorithm to find proteins with reduced energy. This is coherent with the algorithm strategy, as having more processors optimizing a single protein increases the chance of finding a better rotamer sequence. According to the results, the best strategy involves using a relatively large number of global iterations (30−40) as well as using a moderate number of processors (5−10). This way, a trade-off between computational efficiency and reduced energy of the design can be achieved.

Figures 3−6 point out the importance of the initial set of parameters chosen for the optimization. Different sets of parameters were applied to the optimization of existing proteins, and the results differ depending on these parameters. Equations 4 and 5 provide some constraints for the values of d, N_G, n_l and number of processors. Some of these

parameters like the dimensional reduction factor, d, will be limited by the size of the matrices and the available memory resources. In another case, CPU time consumption will be the main criterion guiding our decisions. Still for each case a detailed analysis will have to be performed since we lack a generalized method to fix all these parameters.

Thioredoxin Redesign for Enzymatic Activity. To test our algorithm with an existing protein, we chose the introduction of a new ligand binding site in a thioredoxin scaffold, a problem previously treated using MCSA.[28] The chosen ligand was p-nitrophenyl acetate (PNPA): increasing thioredoxin binding affinity toward this substrate introduces esterase catalytic activity having as substrate PNPA in the thioredoxin scaffold.[12,13] This is a two objective design problem where the score functions are i) the previously described approximation to the folding free energy of the protein and ii) the binding free energy of the complex PNPA-protein. The PNPA was treated as a generalized rotamer interacting with histidine in position 39. To measure the interaction energy between the thioredoxin and the PNPA we constructed an atomic model of the transition state of the reaction.

We mutated to all amino acids (but proline and glycine), all nonproline, nonglycine positions in the neighborhood of position 39. All the rest of the nonproline nonglycine positions were only allowed conformational changes. We also eliminated rotamers with single energy exceeding 30 kcal/mol, to avoid steric clashes. Finally, we considered 101 positions and 7453 rotamers. The distribution of rotamers per position was highly irregular: 70 positions had less than 10 rotamers each, but 18 positions with more than 150 rotamers each were present; there was even a single position containing 959 rotamers. This irregular distribution is due to the different treatment of positions surrounding the active site, where mutations were allowed and the rest where only conformational changes were permitted.

The binding and folding energy matrices occupied 38MB and 65MB, respectively. Both scoring functions were added with equal weights, since in this case folding and binding are equally important. A previous independent MCSA analysis found −461.768 kcal/mol as the minimum energy of the system. At each position, p, each of the 10 processors considered $N_{r_p} = N_{R_p}/d$ rotamers.

As rotamers with single energies higher than 30 kcal/mol have been erased from this matrix, we find that there are some positions with a very low number of rotamers (there are 74 positions with less than 20 rotamers and among them 63 have less than 10). On the other hand, we have a few positions with a large number of rotamers (21 positions with more than 200 rotamers each). The optimization algorithm described relies on the fact that multiple interactions erase the effect of not considering all rotamers at once, but for positions with a very small number of rotamers large statistical fluctuations will appear. Therefore, we considered at least Min_R rotamers at each position (whenever $N_{R_p} < Min_R$, we took all the N_{R_p} rotamers).

Table 2 summarizes our results. The behavior expected from Figure 3 is recovered in this case, as can be seen in Figure 7. As we have previously said, our optimization algorithm relies on the cooperative effect of multiple process independently minimizing the global protein to bypass the fact that in each process only a subset of rotamers at each position is considered. Then a subtle interplay is established between the number of independent processes and the dimensional reduction factor. Nevertheless, as there are cases when a given position has a very limited number of rotamers, we have introduced the Min_R parameter. Figure 8 shows the interplay between these two factors, d and Min_R. Of course setting too high a value of Min_R would mean that all rotamers are considered at all positions.

Table 2. Convergence of our Parallel Optimization Method in the Optimization of Thioredoxin for Stability and PNPA Binding Affinity[a]

energy	d	Min_R	N_G
−461.768	2	1	100
−461.768	5	25	100
−461.768	10	25	100
−461.364	15	25	100
−461.768	15	50	100
−461.768	20	25	100
−461.768	30	50	100
−461.364	2	25	10
−450.950	5	1	10
−457.934	5	5	10
−460.156	5	10	10
−460.910	5	25	10
−461.058	5	50	10
−444.310	10	1	10
−450.970	10	5	10
−459.886	10	50	10
−438.276	15	5	10
−452.500	15	25	10
−459.080	15	50	10
−435.538	20	5	10
−458.962	20	50	10

[a] The shown energy is the sum of both objectives. Independent MCSA optimization yielded a minimum energy of −461.768 kcal/mol. The optimization parameters are $T_0/R = 1$ kcal/mol, $n_l = 10^5$, and $C = 10$.

From Figure 8, we find that fixing $Min_R = 25$ and $d = 10$ results in good convergence properties. Setting $Min_R = 25$ and $d = 10$ means performing a conventional MCSA optimization in those positions with less than 25 rotamers: 75 positions that all together have 402 rotamers, 5.4% of the total number of rotamers. For those positions with $25 < N_{R_p} < 250$, setting $Min_R = 25$ implies that we will always consider 25 rotamers regardless of the d value. There are 15 positions with $25 < N_{R_p} < 250$, and they represent 25.6% of the total number of rotamers. This value of Min_R affects 89 out of the 101 positions, but it leaves 69% of the total number of rotamers unaffected, those that are enduring our optimization algorithm. Comparing the results obtained for $Min_R = 1$ and 25, we find that this slight modification of the algorithm greatly enhances its power.

Dimerization Interface Redesign. 4-Oxalocrotonate tautomerase is one of the smallest enzyme subunits known.[43] 4-Oxalocrotonate tautomerase from *E. coli* in solution forms either a hexamer or a dimer (PDB code 1GYX, 1.35 Å resolution[33]). We redesigned the dimerization interface to increase the probability of a dimer to be formed. As a result, we considered as one of the optimization objectives the folding free energy, to stabilize each unit. Our second scoring function was the interaction energy between the units. During the optimization we did not impose symmetry constraints, so that the final result is no longer a homodimer but a heterodimer with increased binding affinity across the

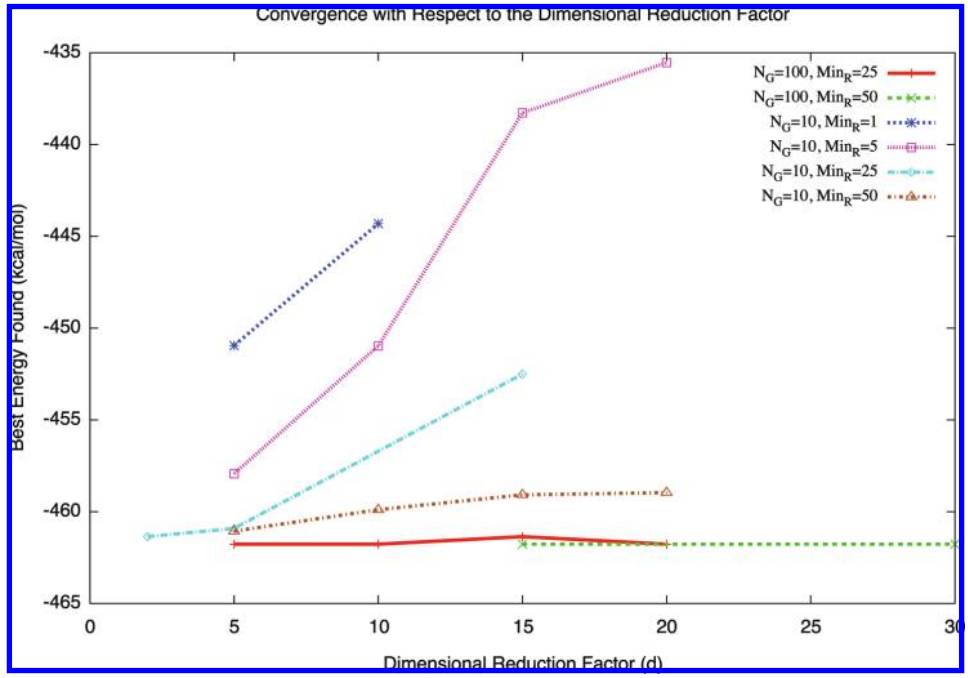


Figure 7. Effect of the dimensional reduction factor, d, on the convergence of the proposed optimization algorithm for the thioredoxin redesign, with $T_0/R = 1$ kcal/mol, $n_l = 10^5$, and $C = 10$.

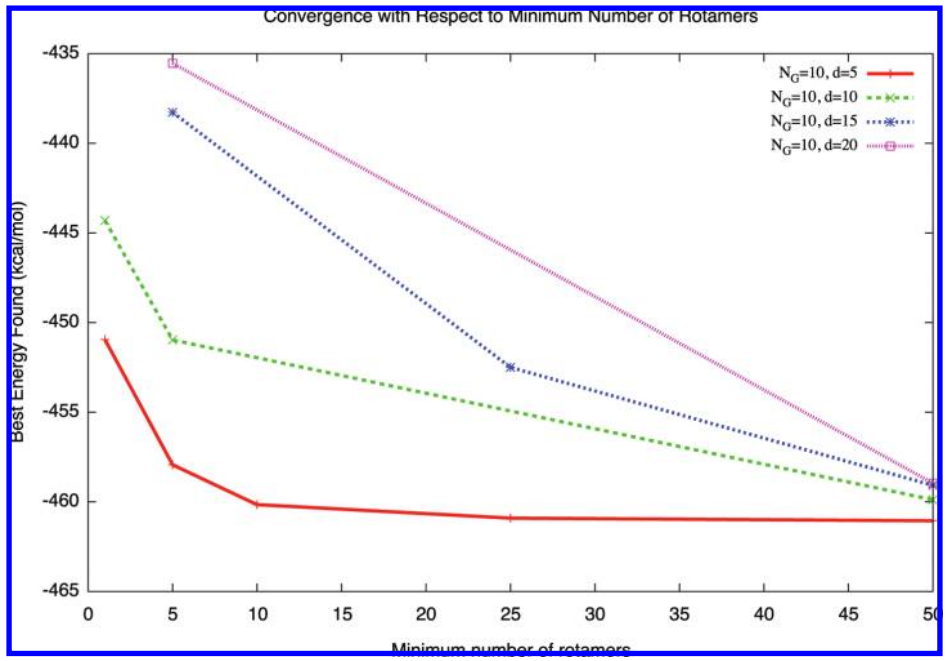


Figure 8. Effect of the minimum number of rotamers on the convergence of the proposed optimization algorithm for the thioredoxin redesign, with $T_0/R = 1$ kcal/mol, $n_l = 10^5$, and $C = 10$.

dimerization interface. The interaction energy between the units was computed as the sum of interaction energy among rotamers pairs one at each unit.

We mutated to all amino acids nonproline positions in the protein, also the proline in the first position was allowed to mutate, and we had 72 designed positions in each chain. As in the previous case, we avoided steric clashes by eliminating all rotamers with a single energy higher than 30 kcal/mol. We obtained a library with 144 positions and a total of 19144 rotamers. In this case the distribution of rotamers at each position was fairly uniform: among the 144 selected positions 142 had more than 100 rotamers and only 2 had less than 10. The matrix had sizes of 450MB and 400MB (folding and binding, respectively) which rendered the traditional optimization process without distributed memory a difficult task. In the optimization both scoring functions were added with equal weights, and we chose as initial temperature for the local MCSA $T_0/R = 1$ kcal/mol. At every position, each of the processors considered a number of rotamers equal to $N_{R_p}/5$ (and a minimum number of 10), $N_G = 100$, and $n_l = 10^6$.

In this case two different and competing objectives were simultaneously optimized. The result of a multiobjective optimization is not a single sequence but a set of nondominated sequences that form the approximation to the Pareto Set (PS) of sequences.[44] To construct the PS we considered

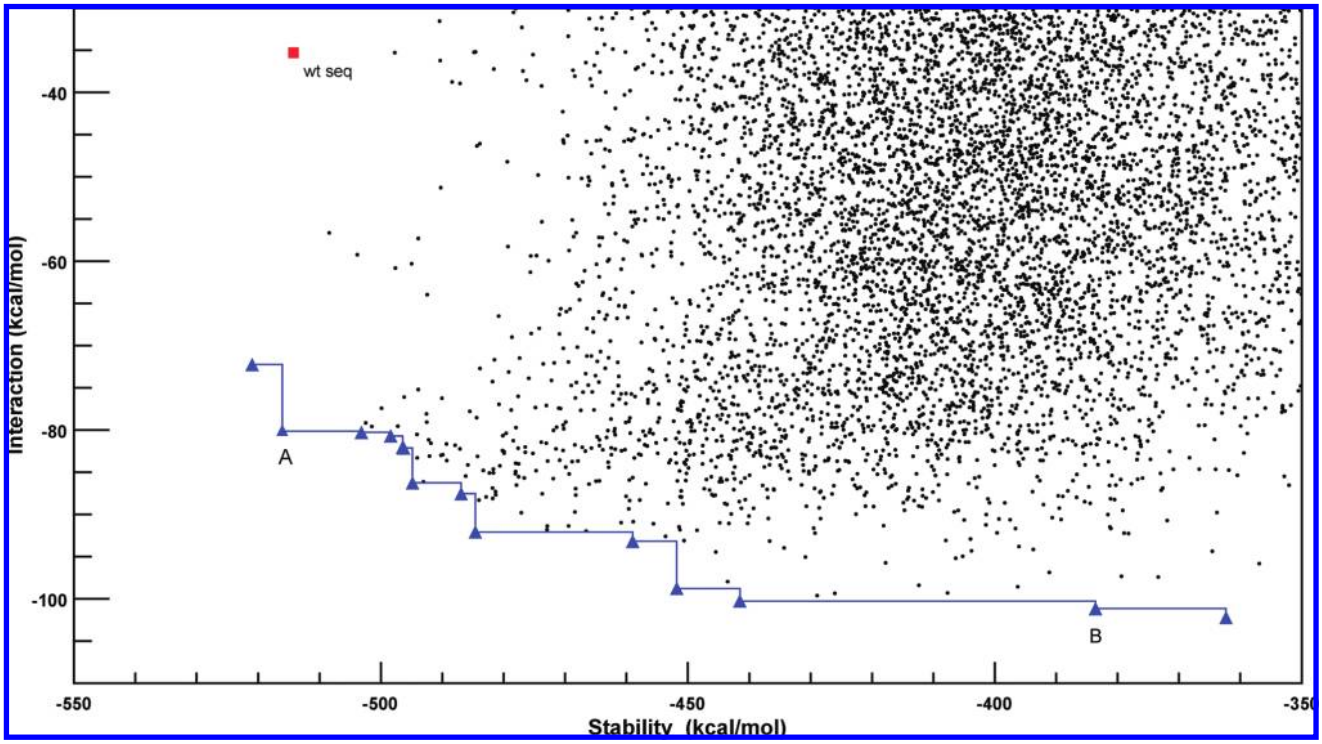


Figure 9. Dimerization interface redesign. Sequences explored in the optimization process. We have signaled (triangles) the nondominated sequences, and together with the frontiers of the regions they dominate (blue lines). These sequences form an approximation to the Pareto Set of sequences. The point (−514.24, −35.31) corresponds to the wild type sequence. Details of the structure of the marked sequences are shown in Figure 10. The complete structures of these sequences can be accessed in the Supporting Information.

the final optimized sequences and the intermediate lowest total energy solutions obtained at each iteration. Then we kept all nondominated ones, since they represent the best trade-off between the objectives. The intermediate sequences as well as the obtained PS are represented in Figure 9, together with the naturally occurring (wild type) sequence. We find that the optimization has increased the performance in both objectives, although the algorithm has been able to explore solutions exploring a broader range of binding affinity.

Sequences forming the PS show, in general, a decrease of the binding energy corresponding to interactions across the interface of about 40 kcal/mol with respect to the native sequence. On the other hand, the wild type sequence is among the most stable sequences obtained (note that the lower the folding energy the more stable a sequence is and the lower the binding energy the higher binding affinity a sequence has). Evolution has presumably optimized this protein for stability but also to form hexamers and homodimers. In our design, both units are no longer identical, and our designed sequences will not likely be able to form hexamers. The release of these two constraints may explain the fact that the wild type sequence is well away from the set of nondominated solutions.

Among the sequences in our PS, we have chosen to analyze two extreme cases, sequences A and B. Sequence A is one of the most stable sequences and has stability and interaction scores of −517.056 and −80.1374 kcal/mol, respectively. In Figure 10 A we can see a detail of the model of the structure corresponding to this sequence. We have focused on the dimerization interfaced formed by the two α helix. Almost no H-bonds involving side chains in this region have been produced in the designing process (those corresponding to the fixed backbone have been maintained, although they are not shown in the figures). On the other hand the side chain modeling has increased the stability of each of the monomers. As an example, the mutations K6D, F8Q, Q45D, and L12N have succeeded in constructing a triple H-bond stabilizing the interaction between the α-helix and the neighbor loop within a monomer. For sequence B (Figure 10 B) we have the opposite situation, where almost all the intradomain H-bonds involving only side chains have been lost, but the number of interactions across the dimerization interface has been greatly increased. This sequence has folding energy = −383.588 kcal/mol and binding energy = −101.165 kcal/mol. Mutations L32E (chain A) and L12K and I5K (in chain B) can be seen to increase the binding affinity of the monomers, through the formation of the interchain H-bonds. On the other hand, mutations L12K and I5K destabilize the protein, since they introduce polar groups into the protein core. These are two of the most extreme examples of the interplay between these competing objectives. Parts A and B show the α helix interface of the modeled structures, but the whole structures show a higher degree of interaction among the monomers than the wild type, as can be seen in the structures provided as Supporting Information.

CONCLUSIONS AND PERSPECTIVES

This paper describes the design of proteins with targeted properties using High Performance Computing. This approach allows a group of processors to collaborate in the optimization of the protein, thus increasing the quality of the obtained solutions through a more exhaustive exploration of the space of sequences and conformations. In addition,

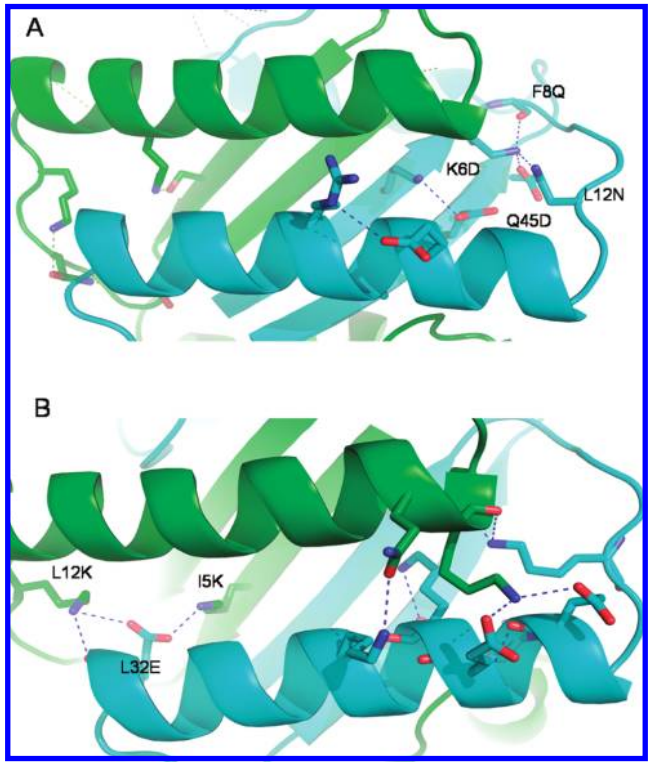


Figure 10. A) Dimerization interface of one of the most stable designed sequences. H-bonds involving side chains of residues in the same monomer have been marked as well as some selected mutations. B) Dimerization interface of one of the sequences with the highest interaction value, H-bonds across the interface have been marked. In both cases only H-bonds involving side chains have been depicted, since those only involving the main chain are kept fixed due to our fixed backbone assumption.

the proposed algorithm includes a dimensional reduction approach, that allows each processor to work with a subset of rotamers at each position of the protein. The subset of rotamers a given processor considers dynamically changes according to a probability distribution that considers the rate of appearance of rotamers in the partial solutions.

We have tested the algorithm with both virtual and real proteins, and we have detailed its performance under different parameter sets (i.e., dimensional reduction factor, number of processors, and number of global iterations). Our results show that it is possible to retrieve the global minimum with a reduced amount of memory. This paves the way to the optimization of larger proteins whose memory requirements represent a serious handicap when using a traditional computer.

In the near future, we plan to automatically compute the number of rotamers to be used for each processor. This value could be self-adjusted in order to fully take advantage of the available memory in the execution platform. This way, the optimization process would adapt its execution to the computational capacities available.

Additionally, the SA performed by each of the jobs to determine the rotamers probability distribution may be suboptimal since this distribution is constructed during the exploration of the different ranges of temperatures. Perhaps it would be well worth constructing this profile after the system has reached a fixed temperature T_{fp}. To construct this profile we would let the system evolve using SA from T_0 to T_{fp}, and then, keeping T fixed to T_{fp}, the algorithm could explore the space of solutions following a Metropolis algorithm to accept or reject solutions. This modification would mean inclusion of additional local iterations to explore this region.

A natural extension of our algorithm to tackle multiobjective optimization problems is the use of the Weighted Sums Method[28,44] to assign different weights to each objective and to analyze the trade-off between the objectives. Nowadays, our method is implemented in such a way that the MCSA accepts/rejects solutions considering only the sum of both objectives, but we could extend it to keep also the solutions that are found to be nondominated and thus candidates for the Pareto Set.

Finally, eq 3 averages the old global probability and the new local probability to obtain the new global probability. In this equation both the local and global probabilities are equally treated. Perhaps the best way to obtain the new global probability would be to introduce a varying weighting factor, in such a way that initially a higher weight is given to the local probability, to erase the effect of the random initial distribution and, as the algorithm proceeds, a higher weight is given to the global probability distribution to not endanger the long-run quality of the global probabilities.

ACKNOWLEDGMENT

A.J. acknowledges financial support from the EU grants BioModularH2 (FP6-NEST contract 043340) and EMERGENCE (FP6-NEST contract 043338) and the ATIGE Genopole/UEVE CR-A3405. A.J., G.M., V.H., and J.M.A. wish to thank the Spanish Ministry of Science and Technology for the financial support received to develop the project ngGrid: New Generation Components for the Efficient Exploitation of eScience Infrastructures (TIN2006-12890). This work has been partially supported by the Structural Funds of the European Regional Development Fund (ERDF).

Supporting Information Available: Files in the Protein Data Bank format containing the atomic coordinates corresponding to sequences A and B from Figure 10. This material is available free of charge via the Internet at http://pubs.acs.org.

REFERENCES AND NOTES

(1) Lippow, S. M.; Tidor, B. *Curr. Opin. Biotech.* **2007**, *18*, 305–311.
(2) Bowie, J. U.; Lthy, R.; Heisenberg, D. *Science* **1991**, *253*, 164–170.
(3) Kuhlman, B.; Dantas, G.; Ireton, G. C.; Varani, G.; Stoddard, B. L.; Baker, D. *Science* **2003**, *302*, 1364–1368.
(4) Dantas, G.; Kuhlman, B.; Callender, D.; Wong, M.; Baker, D. *J. Mol. Biol.* **2003**, *332*, 449–460.
(5) Jaramillo, A.; Wernisch, L.; Héry, S.; Wodak, S. J. *Proc. Natl. Acad. Sci. U. S. A.* **2002**, *99*, 13554–13559.
(6) López de la Paz, M.; Serrano, L. *Proc. Natl. Acad. Sci. U. S. A.* **2004**, *101* (1), 87–92.
(7) Joachimiak, L. A.; Kortemme, T.; Stoddard, B. L.; Baker, D. *J. Mol. Biol.* **2006**, *361*, 195–208.
(8) Bolon, D. N.; Grant, R. A.; Baker, T. A.; Sauer, R. T. *Proc. Natl. Acad. Sci. U. S. A.* **2005**, *102*, 12724–12729.
(9) Ashworth, J.; Havranek, J. J.; Duarte, C. M.; Sussman, D.; Monnat, R. J.; Stoddard, B. L.; Baker, D. *Nature* **2006**, *41*, 656–659.
(10) Jiang, L.; Althoff, E. A.; Clemente, F. R.; Doyle, L.; Rothlisberger, D.; Zanghellini, A.; Gallaher, J. L.; Betker, J. L.; Tanaka, F.; Barbas, C. F.; Hilvert, D.; Houk, K. N.; Stoddard, B. L.; Baker, D. *Science* **2008**, *319*, 1387–1391.

(11) Röthlisberger, D.; Khersonsky, O.; Wollacott, A. M.; Jiang, L.; DeChancie, J.; Betker, J.; Gallaher, J. L.; Althoff, E. A.; Zanghellini, A.; Dym, O.; Albeck, S.; Houk, K. N.; Tawfik, D. S.; Baker, D. *Nature* **2008**, *06879*, 1–6.
(12) Suárez, M.; Tortosa, P.; García-Mira, M. M.; Rodríguez-Larrea, D.; Godoy-Ruiz, R.; Ibarra-Moreno, B.; Sánchez Ruiz, J.; Jaramillo, A. manuscript in preparation, 2008.
(13) Bolon, D. N.; Mayo, S. L. *Proc. Natl. Acad. Sci. U. S. A.* **2001**, *98*, 14274–14279.
(14) Kaplan, J.; DeGrado, W. F. *Proc. Natl. Acad. Sci. U. S. A.* **2004**, *101*, 11566–11570.
(15) Looger, L. L.; Dwyer, M. A.; Smith, J. J.; Hellinga, H. W. *Nature* **2003**, *423*, 185–190.
(16) Lazar, G. A.; Dang, W.; Karki, S.; Vafa, O.; Peng, J. S.; Hyun, L.; Chan, C.; Chung, H. S.; Eivazi, A.; Yoder, S. C.; Vielmetter, J.; Carmichael, D. F.; Hayes, R. J.; Dahiyat, B. I. *Proc. Natl. Acad. Sci. U. S. A.* **2006**, *103*, 4005–4010.
(17) Ogata, K.; Jaramillo, A.; Cohen, W.; Briand, J. P.; Connan, F.; Choppin, J.; Muller, S.; Wodak, S. J. *J. Biol. Chem.* **2002**, *278*, 1281–1290.
(18) Cochran, F. V.; Wu, S. P.; Wang, W.; Nanda, V.; Saven, J. G.; Therien, M. J.; DeGrado, W. F. *J. Am. Chem. Soc.* **2005**, *127*, 1346–1347.
(19) Kazlauskas, R. J. *Curr. Opin. Chem. Biol.* **2005**, *9*, 195–201.
(20) Jürgens, C.; Strom, A.; Wegener, D.; Hettwer, S.; Wilmanns, M.; Sterner, R. *Proc. Natl. Acad. Sci. U. S. A.* **2000**, *97*, 9925–9930.
(21) Chowdry, A. B.; Reynolds, K. A.; Hanes, M. S.; Vorhies, M.; Pokala, N.; Handel, T. M. *J. Comput. Chem.* **2007**, *28* (14), 2378–2388.
(22) Archontis, G.; Simonson, T. *J. Phys. Chem. B* **2005**, 22667–22673.
(23) Huang, A.; Stultz, C. M. *Biophys. J.* **2007**, 34–45.
(24) Hellinga, H. W.; Richards, F. M. *Proc. Natl. Acad. Sci. U. S. A.* **1994**, *91*, 5803–5807.
(25) Dahiyat, B.; Mayo, S. L. *Science* **1997**, *278* (5335), 82–87.
(26) Wernisch, L.; Hery, S.; Wodak, S. J. *J. Mol. Biol.* **2000**, *301* (3), 713–736.
(27) Jones, D. T. *Protein Sci.* **1994**, *3*, 567–574.
(28) Suárez, M.; Tortosa, P.; Carrera, J.; Jaramillo, A. *J. Comput. Chem.* **2008**, *29*, 2704–2711.
(29) Humphris, E. L.; Kortemme, T. *PLoS Comput. Biol.* **2007**, *3*, e164+.
(30) Zagrovic, B.; Sorin, E. J.; Pande, V. *J. Mol. Biol.* **2001**, *313*, 151–169.
(31) Rohl, C. A.; Strauss, C. E.; Misura, K. M.; Baker, D. *Methods Enzymol.* **2004**, *383*, 66–93.
(32) Katti, S. K.; Le Master, D. M.; Eklund, H. *J. Mol. Biol.* **1990**, *212*, 167.
(33) Almrud, J. J.; Kern, A. D.; Wang, S. C.; Czerwinski, R. M.; Johnson, W. H., Jr.; Murzin, A. G.; Hackert, M. L.; Whitman, C. P. *Biochemistry* **2002**, *41*, 12010–1202.
(34) Suárez, M.; Jaramillo, A. J. R. *Soc. Interface*, in press, 2009.
(35) Dunbrack, R. L.; Karplus, M. *J. Mol. Biol.* **1993**, *230*, 543–547.
(36) Brooks, B. R.; Bruccoleri, R. E.; Olafson, B. D.; States, D. J.; Swaminathan, S.; Karplus, M. *J. Comput. Chem.* **1983**, *4*, 187–217.
(37) Ooi, T.; Oobatake, M.; Némethy, G.; Scheraga, H. A. *Proc. Natl. Acad. Sci. U. S. A.* **1987**, *84*, 3086–3090.
(38) Jaramillo, A.; Wodak, S. J. *Biophys. J.* **2005**, *88*, 156–171.
(39) Saad, Y. *Iterative methods for sparse linear systems*; Society for Industrial and Applied Mathematics (SIAM): Philadelphia, PA, U.S.A., 2003.
(40) Gardiner, V.; Hoffman, J. G.; Metropolis, N. *J. Natl. Cancer Inst.* **1956**, *17*, 175–188.
(41) Hörmann, W.; Leydold, J.; Derflinger, G. *Automatic Nonuniform Random Variate Generation*; Springer: Berlin, 2004.
(42) MPI Forum. *MPI: A Message-Passing Interface Standard*, *version 2.0*; 1995.
(43) Whitman, C. P *Arch. Biochem. Biophys.* **2002**, *402* (1), 1–13.
(44) Papalambros, P. Y.; Wilde, D. J. *Principles of Optimal Design*; Cambridge University Press: New York, 2000.

CI8004594