Growth algorithm for finding low energy configurations of simple lattice proteins

Wenqi Huang and Zhipeng Lü*
School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan, Hubei Province, 430074, China

He Shi
Academy of Mathematics and Systems Science in the Chinese Academy of Science, Beijing, 100080, China

(Received 20 December 2004; published 8 July 2005)

PERM and its new variant nPERMis have been developed to optimize the energy function of protein folding based on HP simple lattice model and were found to outperform all other previous fully blind general purpose algorithms. Using the concept of core-guiding and life-forecasting, we propose a new version of nPERMis, called nPERMh. A major difference with respect to nPERMis is that criteria for further growth of new residue are based on the species of current growing monomer and its position in the HP sequence. Seventeen sequences of length ranging from 46 to 124 residues were tested by nPERMh on the cubic lattice and our algorithm proved very efficient. It should be pointed out that our new version of nPERMis is exclusively designed for conformational search. We hope that similar methods will ultimately be useful for finding native states of more realistic protein models.

DOI: 10.1103/PhysRevE.72.016704 PACS number(s): 05.10.Ln, 87.14.Ee, 31.15.Ar, 02.70.Uu

I. INTRODUCTION

Prediction of three-dimensional protein structures has proven to be one of the most challenging problems in theoretical structural biology. In spite of decades of effort, it remains an amazingly difficult computational problem because not only is the folded three-dimensional structure of a protein extremely complicated, but also a general prediction strategy must not depend on the foreknowledge of any specific structural information about the proteins whose structures are to be predicted. In principle, it is possible to predict theoretically the three-dimensional structure of proteins based only on sequence information and without using templates.

Unfortunately, unless simplified models can be used, long simulations are far too costly to be practical. A popular model used in these studies is called the HP simple lattice model [1,2], in which only two types of monomers, hydrophobic (H) and polar (P) ones, are considered. The protein is confined as a self-avoiding path on a regular cubic lattice with attractive or repulsive interactions between neighboring nonbonded monomers. The energy of a lattice protein with a certain configuration can be calculated by summing over all possible pairwise energy between any two neighboring nonbonded hydrophobic monomers:

$$E = \sum_{\substack{i,j=1 \\ i<j-1}}^{N} -\sigma_{ij} \tag{1}$$

where $\sigma_{ij}=1$ if the ith and jth monomers, both being hydrophobic, are the nearest lattice neighbors; $\sigma_{ij}=0$ otherwise. Study indicates that the energy minimization strategies that are used for searching the conformational space can distinguish between native or native-like structures and non-native ones.

Despite its simplification, the corresponding protein folding problem based on the HP lattice model has proven to be NP complete [3]. In recent years, a wide variety of approximate computational strategies have been employed to simulate and analyze this model, including a genetic algorithm [4,5], a strong heuristic core-directed algorithm [6], conventional Monte Carlo schemes with various types of moves [7], an evolutionary Monte Carlo algorithm [8], a sequential importance sampling method [9], the pruned enriched Rosenbluth method (PERM) [10–13], various Monte Carlo simulations [14–16], an exact enumeration method [16], and others. These advances give hope that protein structure prediction is indeed a tractable problem.

Among the above various computational strategies, nPERMis [12] generally yields much better computational results than the other ones. It is a novel biased chain growth algorithm for the *ab initio* prediction of protein structures given only the amino acid sequences of the proteins. Especially, new lower energies for several two- and three-dimensional protein sequences were found by nPERMis for the first time. It is the purpose of this article to present a new improved nPERMis for solving the three-dimensional lattice protein folding problem.

II. ALGORITHM

A. Pruned enriched Rosenbluth method and nPERMis

The original PERM is built on the old idea of sequential sampling or chain growth [17]. From an empty configuration, the polymer chain grows by adding monomers one by one and a Boltzmann factor weight is given to a certain partial configuration containing n monomers. This weight can be represented as the product of the potential energy of adding the ith monomer ($W_n=\prod_{i=1}^{n} w_i$). Configurations with too

*E-mail address: zhipenglu@126.com

low weight are pruned at certain probability, and high weight configurations are cloned by splitting the branch into several identical independent ones, with the weight shared among them. In this case, the branches with higher weight would have more chances of survival and enrichment, while the *bad* ones are inclined to be *killed* before hitting the chain length. It can be viewed as a realization of a "*natural evolution*" strategy [10–13,17].

The main concern of PERM is how to decide on when to enrich or prune. The solution is given by choosing thresholds $W_n^>$ and $W_n^<$, depending on the estimate of the partition sums of n-monomer chains (Z_n):

$$Z_n = \frac{1}{M}\sum_{i=1}^{M} W_n^{(i)}, \tag{2}$$

where $i=1,2,\ldots,M$ denotes the already existing trials of branch of length n during the running process and these thresholds are continuously updated as the simulation progresses [17]. Current chain of length n is enriched by making identical copies when W_n is bigger than threshold $W_n^>$. For W_n being smaller than $W_n^<$, the chain is pruned at probability 1/2, otherwise it grows with doubling the weight. If W_n lives between the two thresholds, the chain is simply continued without pruning and enriching the partial configuration [12,18,19].

In each enrichment event, exact clones are involved. While this is efficient at high temperatures, it turns out to be untrue at very low temperatures because in this case all the copies will follow a single path and result in a loss of diversity. One way out is that we no longer make identical copies when enriching the high-weight branches, but enforce the branches chosen to be mutually different [12]. When deciding to grow a configuration with $n-1$ monomers, we first get an estimated weight W_n^{pred}. If $W_n^{pred} > W_n^>$, we choose k different positions to place the nth monomer, where

$$W_n^{pred} = W_{n-1}\sum_{\alpha} q_{\alpha} \tag{3}$$

[here W_{n-1} is the weight of partial configuration with $n-1$ monomers; q_α denotes the quality of the valid branch α of placing the nth monomer, see Eq. (A1) in the Appendix],

$$W_n^> = C(Z_n/Z_0)(C_n/C_0)^2, \tag{4}$$

$$W_n^< = 0.2W_n^>, \tag{5}$$

$$k = \min\left\{k_{free}, \left\lceil W_n^{pred}/W_n^>\right\rceil\right\}. \tag{6}$$

C, Z_0 and C_0 are constants; k_{free} denotes the total number of possible branches of placing the nth monomer [12]. This modification yields much better computational results than the previous version of PERM [11] and all other previous stochastic algorithms, because it not only guides the searching to the promising region, but also ensures the diversity of configurations [12].

B. Improvements in nPERMis

Since the lowest energy of a given protein sequence is our main focus, we define the weight of a partial configuration as $W_n = W_{n-1}\exp(-\Delta E_n/T)$ (with $W_1 = W_2 = 1$) [19], rather than $W_n = W_{n-1}k_{free}\exp(-\Delta E_n/T)$ [12]. We find that it is also a feasible approach to unify the calculation of weight when growing possible branches, i.e., neither doubling the weight when $W_n^{pred} < W_n^<$, nor sharing weight among the k different continuations when $W_n^{pred} > W_n^>$. Computational results indicate that with such modifications our algorithm can also reach the lowest energy states as easily as nPERMis for several difficult two-dimensional lattice protein sequences. Following will be the further improvements in these strategies.

W_n^{pred} is an essential factor affecting the efficiency of this growth algorithm, so the main purpose of this article is to redefine W_n^{pred} based on some heuristic ideas. Because the native state configuration for a certain protein sequence generally features a hydrophobic core, one may try to construct a hydrophobic core by means of some heuristic strategies to find the global optimum [6]. In this chain growth algorithm, when we place a hydrophobic monomer, it is guided to the *right* way due to the Boltzmann factor [12,19]. However, once a polar is placed, there is not as much powerful guidance as placing the hydrophobic monomer, so we cannot easily discriminate the *good* branches from the *bad* ones. Therefore, one of our modifications is to let polar monomers have more chances to enrich than the hydrophobic ones. We can put this idea into practice by multiplying the number of possible branches k_{free} and a given constant β [see Eqs. (7) and (8)]. Thus, with compulsively increasing the chances of poplars to enrich, especially for those with a great number of possible branches, W_n^{pred} is redefined as follows:

If nth monomer is hydrophobic,

$$W_n^{pred} = \alpha(n)W_{n-1}\exp(-\Delta\bar{E}_n/T). \tag{7}$$

If nth monomer is polar,

$$W_n^{pred} = \alpha(n)W_{n-1}k_{free}\beta, \tag{8}$$

where $\Delta\bar{E}_n$ is the average newly increased energy by the placement of the nth monomer.

Another modification in this article arises from the idea of life-forecasting. We now separate the growth running procedure into three phases. When a partial chain is short, without knowing whether it can grow into a *good* configuration or a *bad* one at a later time, we may give it considerable chances to make as many choices as possible to grow. This can be demonstrated in Eq. (4), where C_0 is a great constant from 10^4 to 10^6. On the other hand, once a protein chain grows near to its full length, we can readily determine whether this partial configuration is *good* or not, because much information concerning the minimized energy has been kept in the partial configurations. However, when a chain's length is between "*too short*" and "*very long*," its foreground cannot be easily foreseen. In this case, we cannot enrich it automatically as it is in *childhood*, nor can we determine whether to stop its growth decisively as it is *matured*. Several benchmark instances in Ref. [12], such as the sequences of $N=64$ and $N=88$, have shown that one may have to pass

TABLE I. Ten 48-monomer sequences and nPERMh's performance compared with previous versions of PERM (nPERMss and nPERMis) [12].

No.	HP sequence[a]	$E_{\min}$[b]	nPERMss[c]	nPERMis[d]	nPERMh[e]
48.1	$HPH_2P_2H_4PH_3P_2H_2P_2HPH_3PHPH_2P_2H_2P_3HP_8H_2$	−32	0.66	0.63	1.22
48.2	$H_4PH_2PH_5P_2HP_2H_2P_2HP_6HP_2HP_3HP_2H_2P_2H_3PH$	−34	4.79	3.89	1.45
48.3	$PH(PH_2)_2H_4P_2(HP)_2(PH)_2(HP)_3P_2H(P_2H_2)_2P_2(HP)_2PHP$	−34	3.94	1.99	0.37
48.4	$PHPH_2P_2HPH_3P_2H_2PH_2P_3H_5P_2HPH_2(PH)_2P_4HP_2(HP)_2$	−33	19.51	13.45	1.83
48.5	$P_2HP_3HPH_4P_2H_4PH_2PH_3P_2(HP)_3PHP_6(H_2P)_2H$	−32	6.88	5.08	1.78
48.6	$H_3P_3H_2PH(PH_2)_3PHP_7(HP)_2PHP_3HP_2H_6PH$	−32	9.48	6.60	0.58
48.7	$PHP_4HPH_3(PH)_2H_3(PH_2)_2P_3(HP)_2P_2H(H_2P_2)_3PH$	−32	7.65	5.37	0.50
48.8	$(PH_2)_2HPH_4P_2H_3P_6HPH_2P_2H_2PHP_3H_2(PH)_3HP_3$	−31	2.92	2.17	2.01
48.9	$(PH)_2P_4(HP)_3(PH)_2H_5P_2H_3PHP_2HPH_2P_2HPH_3P_4H$	−34	378.64	41.41	32.72
48.10	$PH_2P_6H_2P_3H_3PHP_2HPH_2(P_2H)_3HP_2H_7P_2H_2$	−33	0.89	0.47	0.34

[a]Ten 48-monomer sequences from Ref. [21].
[b]Ground state energies [21].
[c]CPU times (minutes) per independent ground state hit, on 167 MHz Sun ULTRA I work station [12].
[d]CPU times (minutes) per independent ground state hit, on 167 MHz Sun ULTRA I work station [12].
[e]CPU times (minutes) for the total rounds of hitting the ground state on AMD 1.84 GHz PC.

through *bad* partial configurations first to get *good* ones at a later time. Then, one possible way out is to purposely increase the chain's chance of growth by multiplying a constant C_1. $\alpha(n)$ in Eqs. (7) and (8) is defined as

$$\alpha(n)=\begin{cases}1, & n \leqslant 0.3N,\\ C_1, & 0.3N < n \leqslant 0.75N,\\ C_2, & n > 0.75N,\end{cases} \tag{9}$$

where $C_1 > C_2 > 1$ and $C_1 \in [30,45]$, $C_2 \in [5,10]$. While we try some other forms of $\alpha(n)$, such as the linear or quadratic function of n, they all give similar computational results.

C. Implementation of the algorithm

Being a growth algorithm, nPERMh can be described by giving the strategy of how to choose a branch or branches to place the nth monomer under any given situation where the first $n-1$ monomers have been placed (see the Appendix).

For the first configuration hitting length N, $W^{<}=0$ and $W^{>}=+\infty$ were employed, i.e., nPERMh neither pruned nor branched, by which the initial values of Z_n and C_n could be obtained ($Z_n=W_n$ and $C_n=1$). Then the above proposed recursive process was implemented until all configurations hit the total chain length N or the growth of the chain is stopped because of a "*dead end*" or pruning. We call such a complete searching of the configuration tree a round. After a present so-called round is finished, a new round starts from an empty configuration, with keeping Z_n of the last round without change and resetting $C_n=1$. Such process is implemented repeatedly until obtaining the given lowest energy or hitting the fixed maximal number of round. The CPU time we report below is the time spent on the whole rounds of algorithm running.

TABLE II. Several three-dimensional HP sequences from literature and computational performance comparison with nPERMis.

No.	Length	HP protein sequence[a]	E_{nPERMis}[b] (hours)	E_{nPERMh}[c] (hours)
1	46	$P_2H_3PH_3P_3HPH_2PH_2P_2HPH_4PHP_2H_5PHPH_2P_2H_2P$	−34[d]	−35(0.01)
2	58	$PH(PH_3)_2P_2H_2PH(PH_2)_2(HP)_3H_2P_2H_3P_2(HP)_2P(P_2H)_3(HP_2)_2H$	−44(0.19)	−44(1.10)
3	103	$P_2H_2P_5H_2P_2H_2PHP_2HP_7HP_3H_2PH_2P_6HP_2HPHP_2HP_5H_3P_4H_2PH_2P_5H_2P_4H_4PHP_8H_5P_2HP_2$	−54(3.12)	−55(0.25)
4	124	$P_3H_3PHP_4HP(P_4H_2)_2P_2H_2(P_4H)_2(P_2H)_2HP_3H_2PHPH_3P_4H_3P_6H_2(P_2H)_2PHP_2$ $HP_7HP_2H_3P_4HP_3H_5P_4H_2(PH)_4$	−71(12.3)	−71(1.19)
5	64	$(PH_2)_3H(P_2HPH)_2P_2H_3(PH_2)_2P(PH_2)_3H(P_2HPH)_2P_2H_3(PH_2)_2P$	−56(0.45)	−56(0.47)
6	67	$PH((PH_2)_2PHP_2H_3P_3H)_3(PH_2)_2PHP_2H_3P$	−56(1.10)	−56(0.33)
7	88	$PH(PH_2)_2PHP_2H_2(P_2H)_6H(P_2H_3)_4P_2H(PH_2)_2P(HP_2)_3H_2(P_2H)_3HP_2HP$	−69 (···)	−69(0.45)

[a]HP sequences from [6,22,24].
[b]Lowest energy found by nPERMis and CPU time (hours) per lowest energy configuration hit on 667 MHz DEC ALPHA 21264 [12].
[c]Lowest energy found by nPERMh and CPU time (hours) for the total rounds of hitting the lowest energy on AMD 1.84 GHz PC.
[d]The lowest energy in Ref. [23], No report by nPERMis.

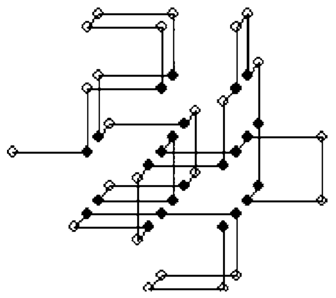

FIG. 1. Typical configuration with $E=-34$ of the 48.9# sequence. [Notes: For all figures, black and white beads represent H (hydrophobic) and P (polar) monomers, respectively.]

III. RESULTS

In previous work [19,20], we have applied the improved PERM, without the strategies of core-guiding and life-forecasting, to a two-dimensional lattice model with chain length from $N=36$ to $N=100$ and obtained the lowest energy for nine 2D HP protein sequences. In this article, we will focus ourselves on finding the lowest energy states for a set of three-dimensional protein sequences. Tables I and II show the sequences and the performance comparison between nPERMh and nPERMis.

(a) We first test our algorithm on ten designed three-dimensional instances with 48 monomers [21]. Using our new version of nPERMis, called nPERMh, we could reach lowest energy configurations for all of them within very short CPU times. As seen from Table I, our algorithm has comparable computational performance with nPERMis. For sequence 48.9, whose lowest energy was not hit by Ref. [6] and who has the lowest degeneracy among these ten sequences, Fig. 1 shows one of its typical lowest energy configurations. Numerical results indicate that our improvements concerning the heuristic ideas of core-guiding and life-forecasting are feasible and effective.

(b) Then, we test our algorithm on four sequences of $N=46$, 58, 103, and 124, which were proposed as models for real proteins [22]. $E=-34,-42,-49$, and -58 were reached in Ref. [23] respectively for these four HP chains. For the sequence of $N=46$, no results of this sequence are reported by nPERMis, but we could find configurations with new lower energy $E=-35$ than the CI method [23] did within 40 s CPU time (see Fig. 2), which has a highly compact hydrophobic core. For the sequences of $N=58$, 103, and 124, with nPERMis, much lower energies than those in Ref. [23] were found, which are supposed to be the putative ground states for these sequences (see Table II). With our new version of nPERMis, we could get the same low energy as nPERMis for the sequences of $N=58$ and $N=124$ with comparable CPU times. Figures 3 and 4 show the typical lowest energy configurations for these two sequences, which are the newly found configurations of lowest energy states missed by nPERMis. For the sequence of $N=103$, it is noteworthy that we could get lower energy than that of nPERMis, which can only get energy $E=-54$. With $\exp(1/T)=25$, we get 46 new lower energy configurations with $E=-55$ within 0.25 h on AMD 1.84 GHz PC. Figure 5 shows two typical configurations of these putative lowest energy states.

(c) Next, two HP sequences of $N=64$ and $N=67$ [24], both with $\boldsymbol{E}_{\min}=-56$, were studied. We can easily get the ground states for them (see Figs. 6 and 7). The $N=67$ sequence folds into a configuration resembling an α/β barrel, which has much low degeneracy. Finally, we test a sequence of $N=88$ and $\boldsymbol{E}_{\min}=-72$ given in Ref. [6], for which we can only get $E=-69$ as nPERMis did. The difficulties of PERM with this sequence are easily understood, because before its nucleus of the hydrophobic core is formed, a growth algorithm starting at either end has to pass through *bad* partial configurations first to get *good* ones at a later time [12].

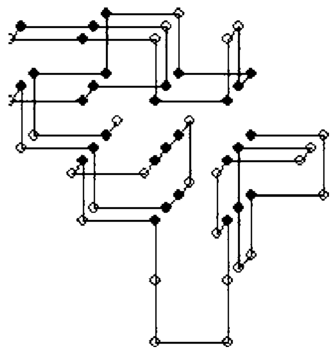

FIG. 3. Typical configuration with $E=-44$ of the sequence $N=58$.

IV. DISCUSSION AND CONCLUSION

In this article we present a new version of nPERMis which is applied to a three-dimensional HP lattice protein

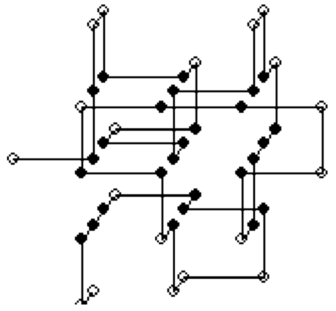

FIG. 2. Typical configuration with $E=-35$ of the sequence $N=46$

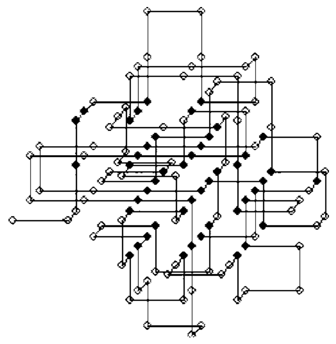

FIG. 4. Typical configuration with $E=-71$ of the sequence $N=124$.

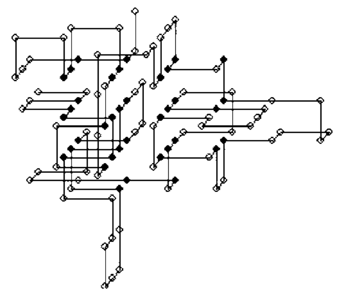
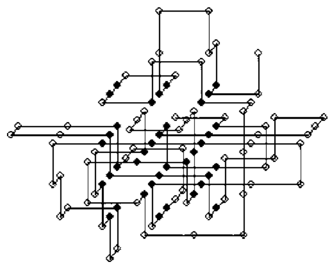

FIG. 5. Two typical configurations with $E=-55$ of the sequence $N=103$.

model. The main improvement is that we introduce a new definition of W_n^{pred} based on the heuristic ideas of core-guiding and life-forecasting. The W_n^{pred} is so modified that we choose different definitions according to whether the current growing monomer is hydrophobic or hydrophilic and which phase the current growing monomer locates in the whole HP sequence. Although we apply our algorithm only to a three-dimensional HP lattice protein model in this article, principally the same algorithm can also be general enough to be used for other lattice or off-lattice systems [25].

Using strong heuristic strategies, our algorithm nPERMh is here exclusively designed for conformational search. So,it may not be able to generate a correct Boltzmann distribution, which is different from the standard PERM and nPERMis [10–12]. Briefly, it is the only purpose of this article to present a more efficient algorithm for finding lower energy states of 3-D HP lattice proteins, rather than to explore the thermodynamic and kinetic behavior of model proteins.

Comparing our results with previous work, we see that we found the lowest energy states and had comparable CPU times for all the ten 48-monomer sequences. When it comes to other HP protein sequences up to $N=124$, our algorithm also performs efficiently in finding the lowest energy configurations, i.e., we can find the known lowest energy states but one. For sequences of $N=58$ and $N=124$, we found the

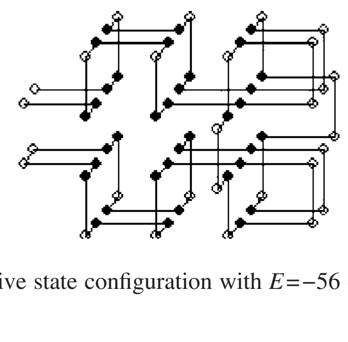

FIG. 6. Native state configuration with $E=-56$ of the $N=64$ HP sequence.

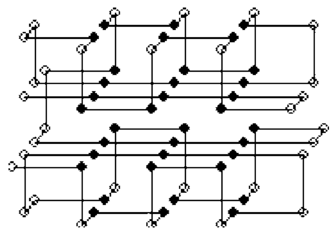

FIG. 7. Native state configuration with $E=-56$ of the $N=67$ HP sequence forms an α/β barrel

same low energy as nPERMis, but new configurations of the lowest states missed by nPERMis. Especially for the sequences of $N=46$ and $N=103$, we respectively found lower energy $E=-35$ and $E=-55$ than previous methods, while the lowest energy that the CI (contact interactions) method [23] and nPERMis can get is $E=-34$ and $E=-54$ for these two sequences, respectively. All these indicate that our improvements in nPERMis are successful. In future work, we hope that our improved algorithm will be helpful for other more realistic protein models, such as the off-lattice model [26] or even the all-atom model [27].

ACKNOWLEDGMENTS

This work was supported by National Natural Science Foundation of China (Grant No. 10471051) and partially supported by the NKBRPC (Grant No. G2004CB318000).

APPENDIX

The recursive procedure of placing the nth monomer when the previous $n-1$st monomer have been placed.

(1) Compute the qualities of all possible valid branches α of placing the nth monomer:

$$q_\alpha = (k_{free}^{(\alpha)} + 0.5)e^{-(\Delta E_n/T)}, \tag{A1}$$

where $k_{free}^{(\alpha)}$ is the number of free positions where the $n+1$th monomer can be placed after the nth monomer has been placed with action α.

(2) Compute the predicted weight W_n^{pred} and thresholds $W_n^<$, $W_n^>$ according to Eqs. (7) and (8) and Eqs. (4) and (5), respectively, where Z_n and C_n are the arithmetic average and the total number of W_n s that have already been generated, respectively.

(3) Compare W_n^{pred} with $W_n^<$ and $W_n^>$, and decide to enrich or prune:

(a) If $W_n^{pred} \in [W_n^<, W_n^>]$, choose a branch α at probability $q_\alpha/\Sigma_\alpha q_\alpha$ among the possible branches and grow it, then compute W_n, update Z_n [Eq. (2)], $C_n(C_n=C_n+1)$, and turn into a new situation, where $W_n=W_{n-1}\exp(-\Delta E_n^{(\alpha)}/T)$.

(b) If $W_n^{pred} < W_n^<$, draw r uniformly $\in(0,1]$; if $r<1/2$, the partial configuration is pruned; else choose a branch α at probability $q_\alpha/\Sigma_\alpha q_\alpha$ among the possible branches and implement it, then compute W_n, update Z_n [Eq. (2)], $C_n(C_n=C_n+1)$, and turn into a new situation, where $W_n=W_{n-1}\exp(-\Delta E_n^{(\alpha)}/T)$.

(c) If $W_n^{pred}>W_n^{>}$, choose a set of branches $A=\{\alpha_{\nu_1},\ldots,\alpha_{\nu_k}\}$ consisting of k [Eq. (6)] mutually different continuations α_{ν_j} at probability p_A and implement them one by one, compute W_n, update Z_n [Eq. (2)], $C_n(C_n=C_n+1)$, and turn into corresponding new situations independently [12]:

$$p_A=\frac{\sum_{\alpha\in A} q_\alpha}{\sum_{A'}\sum_{\alpha\in A'} q_\alpha}, \tag{A2}$$

where A' represents a subset containing k mutually different branches. Corresponding weights of the k partial configurations with different continuations are

$$W_{n,\alpha_{\nu_i}}=W_{n-1}e^{-(\Delta E_{n,\alpha_{\nu_i}}/T)},\quad i=1,\ldots,k \tag{A3}$$

[1] K. A. Dill, Biochemistry **24**, 1501 (1985).
[2] K. A. Dill, S. Bromberg, K. Yue, K. M. Fiebig, D. P. Yee, P. D. Thomas, and H. S Chan, Protein Sci. **4**, 561 (1995).
[3] R. Unger and J. Moult, Bull. Math. Biol. **55**, 1183 (1993).
[4] R. Unger and J. Moult, J. Mol. Biol. **231**, 75 (1993).
[5] R. König and T. Dandekar, Protein Eng. **14**, 329 (2001).
[6] T. C. Beutler and K. A. Dill, Protein Sci. **5**, 2037 (1996).
[7] G. Chikenji, M. Kikuchi, and Y. Iba, Phys. Rev. Lett. **83**, 1886 (1999).
[8] F. Liang and W. H. Wong, J. Chem. Phys. **115**(7),3374 (2001).
[9] J. L. Zhang and J. S. Liu, J. Chem. Phys. **117**(7), 3492 (2002).
[10] P. Grassberger, Phys. Rev. E **56**, 3682 (1997).
[11] H. Frauenkron, U. Bastolla, E. Gerstner, P. Grassberger, and W. Nadler, Phys. Rev. Lett. **80**, 3149 (1998).
[12] H. P. Hsu, V. Mehra, W. Nadler, and P. Grassberger, Phys. Rev. E **68**, 021113 (2003); H. P. Hsu, V. Mehra, W. Nadler, and P. Grassberger, J. Chem. Phys. **118**, 444 (2003).
[13] U. Bastolla, H. Frauenkron, E. Gerstner, P. Grassberger, and W. Nadler, Proteins: Struct., Funct., Genet. **32**, 52 (1998).
[14] A. Sali, E. Shakhnovich, and M. Karplus, J. Mol. Biol. **235**, 1614 (1994).
[15] N. D. Socci and J. N. Onuchic, J. Chem. Phys. **101**, 1519 (1994).
[16] D. K. Klimov and D. Thirumalai, Proteins: Struct., Funct., Genet. **26**, 411 (1996).
[17] M. N. Rosenbluth and A. W. Rosenbluth, J. Chem. Phys. **23**, 356 (1955).
[18] P. Grassberger, *NIC Symp. Proc*, Vol. 1 (2004)
[19] W. Q. Huang and Z. P. Lü, Chin. Sci. Bull. **49**, 2092 (2004).
[20] W. Q. Huang, N. H. Cheng, and Z. P. Lü, J. Huazhong Univ. Sci. Technol. **32**, 1 (2004) (in Chinese).
[21] K. Yue, K. M. Fiebig, P. D. Thomas, H. Chan, E. Shakhnovich, and K. A. Dill, Proc. Natl. Acad. Sci. U.S.A. **92**, 325 (1995).
[22] E. E. Lattman, K. M. Fiebig, and K. A. Dill, Biochemistry **33**, 6158 (1994).
[23] L. Toma and S. Toma, Protein Sci. **5**, 147 (1996).
[24] K. Yue and K. A. Dill, Proc. Natl. Acad. Sci. U.S.A. **92**, 146 (1995).
[25] H. P. Hsu, V. Mehra, and P. Grassberger, Phys. Rev. E **68**, 037703 (2003).
[26] F. H. Stillinger and T. Head-Gordon, Phys. Rev. E **52**, 2872 (1995).
[27] A. Irbäck, J. Phys.: Condens. Matter **15**, 1797 (2003).