J. Mol. Biol. (1993) **231**, 75–81

Genetic Algorithms for Protein Folding Simulations

Ron Unger[1,2] and John Moult[1]

[1]*Center for Advanced Research in Biotechnology*
Maryland Biotechnology Institute, University of Maryland
9600 Gudelsky Drive, Rockville, MD 20850, U.S.A.

[2]*Institute for Advanced Computer Studies*
University of Maryland, College Park, MD 20742, U.S.A.

(Received 14 July 1992; accepted 18 September 1992)

Genetic algorithms methods utilize the same optimization procedures as natural genetic evolution, in which a population is gradually improved by selection. We have developed a genetic algorithm search procedure suitable for use in protein folding simulations. A population of conformations of the polypeptide chain is maintained, and conformations are changed by mutation, in the form of conventional Monte Carlo steps, and crossovers in which parts of the polypeptide chain are interchanged between conformations. For folding on a simple two-dimensional lattice it is found that the genetic algorithm is dramatically superior to conventional Monte Carlo methods.

Keywords: protein folding simulations; genetic algorithms; lattice models; search methods; folding pathways

1. Introduction

Computing the functional conformation of a protein molecule from the amino acid sequence is difficult for two reasons: the contributions to free energy that stabilize the folded conformation are poorly understood (see review by Dill, 1990), and the space of possible conformations is very large and complex (Levinthal, 1968), making it difficult to search for the appropriate free energy minimum. While analyzing the first problem requires detailed models of protein structure, the second problem can be investigated on much simplified models. In this paper we address the search problem, and investigate the potential usefulness of a recently established search technique, genetic algorithms, for finding the functional conformation of proteins.

Genetic algorithm methods (GAs†) are so called because they utilize the same optimization procedures as natural genetic evolution: mutation, crossover and replication operating on strings (Holland, 1975; Goldberg, 1989). In the last few years these methods have begun to gain recognition as a valuable search technique (Goldberg, 1989; Davidor, 1990). In GAs a population of current solutions is maintained. The solutions evolve by mutations and crossovers. The latter process is the heart of the method. Technically, the operation consists of exchanging parts of strings between pairs of solutions, so as to yield new solutions. This has a large impact on the effectiveness of the search, since it allows exploration of regions of the search space not accessible to either of the two "parent" solutions. Through such interactions, good features from one solution can be transferred to the others and further explored. The population size is maintained by pruning, using criteria of fitness for each solution in such a way that (1) better solutions have a higher chance of reproducing; and (2) the diversity of the population is maintained to allow for a large sampling of individual solutions so that many combined features may emerge. Experience with other "co-operative" problem solving methods (Clearwater *et al.*, 1991; Huberman, 1990) has shown that this feature of exchange of information between solutions is often a powerful way of extending the effectiveness of a search.

Our application of GAs to the protein folding problem may be regarded as an extension of the more familiar Monte Carlo (MC) methods to include information exchange between a set of parallel simulations. A population of evolving conformations is maintained. Each conformation changes independently for some time by the Metropolis Monte Carlo procedure (Metropolis *et al.*, 1953) in

† Abbreviations used: GA, genetic algorithm; MC, Monte Carlo.

0022-2836/93/090075-07 $08.00/0

the usual manner, in a process equivalent to the accumulation of point mutations. Then selected polypeptide chains are cut and each rejoined to another chain cut at the same point (crossovers). Metropolis-type criteria are used to see if each newly generated conformation should be accepted. Those that are accepted enter the MC phase again, and the process is iterated. Here, we describe the details of the procedure and compare its effectiveness with Monte Carlo alone. We find that a simple GA can dramatically improve search effectiveness in a model of protein folding.

2. The Model

We wish to develop an implementation of a GA suitable for protein folding and compare it with the MC method. Thus, we seek to use the simplest model that still captures the essence of the important components of protein folding (Lau & Dill, 1990). The linear sequence is composed of "amino-acids" of only two types: hydrophobic (black) and hydrophilic (white). This sequence is "folded" on a two-dimensional square lattice on which at each point the chain can turn 90° left or right, or continue ahead. The energy function is simple: -1 for each direct contact (occupying neighboring non-diagonal lattice points) of non-bonded hydrophobic-hydrophobic amino acids. Figure 1 shows possible conformations of the 20 amino acid molecule *B-W-B-W-W-B-B-W-B-W-W-B-W-B-B-W-W-B-W-B*.

Under this energy function, low energy conformations are compact with a hydrophobic core: since hydrophobic–hydrophobic interactions are rewarded, the hydrophobic (black) residues tend to be on the inside of a low energy structure, while the hydrophilic (white) residues are forced to the surface. Because each residue can participate in only two contacts at most (3 for the terminal residues), the solvation of hydrophobic residues and the desolvation of hydrophilic residues are implicitly unfavorable without being directly penalized.

Lattice-based Monte Carlo simulations using a simplified description of the atomic structure have been the most successful folding methods so far (Covell & Jernigan, 1990; Skolnick & Kolinsky, 1990). We use additional simplifying assumptions here: a two-dimensional model is used, residues are represented by a single atom, internal electrostatic interactions are not considered, and the energies are very short range. Nevertheless, such simple models do exhibit many of the features of real folding (Lau & Dill, 1990; Crippen, 1991) and also permit a rigorous analysis of the search efficiency, as seen below.

The number of possible valid (i.e. self-avoiding) conformations for a L-long sequence on a two-dimensional square lattice approaches:

$$A\mu^L L^\gamma,$$

where $\mu \simeq 2{\cdot}63$, $\gamma \simeq 0{\cdot}333$ (Guttman *et al.*, 1968; Barber & Ninham, 1970). Thus, the number of

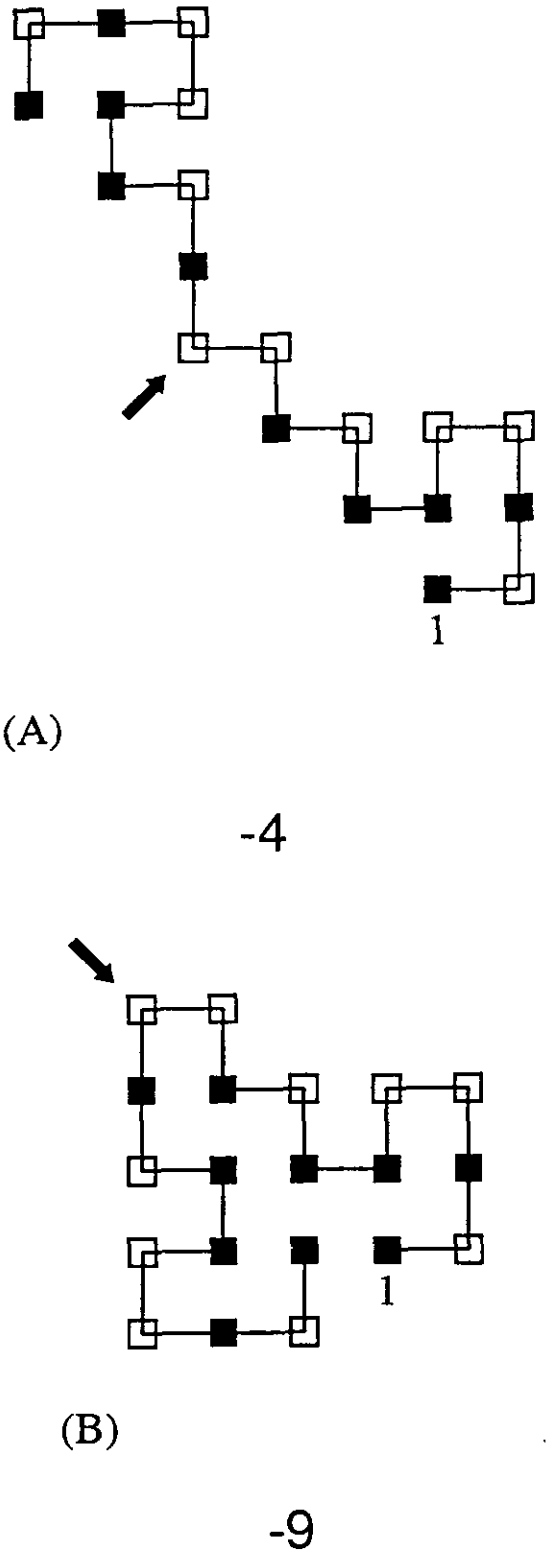

Figure 1. The Monte Carlo algorithm. The process starts with the structure in a fully extended conformation. Then, an amino acid is chosen at random and the C-terminal portion of the chain is rotated around that amino acid. In this example residue number 11 was chosen randomly as the pivot for a move. A 180° rotation brings the structure in (A) with an energy of -4 to the compact structure with lower energy -9 (B). The move is always accepted if it leads to a lower energy conformation, or non-deterministically accepted, according to the energy increase with the move.

Table 1
Energy level distribution

Energy level	No. of conformations
0	36,098,079
−1	31,656,934
−2	12,473,446
−3	2,943,974
−4	517,984
−5	77,080
−6	10,364
−7	1194
−8	96
−9	4
Total	83,779,155

A full enumeration was performed to evaluate the energy of all the self-avoiding confirmations possible for the sequence BWBW-WBBWBWWBWBBWWBWB. For each energy level we list the number of conformations with that energy. Note that the largest fractional decrease is between the number of conformations found in energy level −8 and the number of conformations with the lowest energy level −9.

possibilities is exponential in the length of the sequence. Our goal is to devise a search algorithm that can find a conformation with the lowest free energy value. For the sequence given above, the energies of all the 83,779,155 possible valid conformations were calculated (see Table 1). The number of conformations in each energy level decreases rapidly, with the largest fractional decrease in the final transition to the lowest energy level: there are four conformations with energy −9 *versus* 96 conformations with −8. (Similar behavior was observed for 24 residue long sequences.) Note that even for this very simple lattice model the precise arrangement of an optimal conformation is very rare and difficult to achieve. The infinitesimally small size of the optimal subset relative to the size of the conformational space (*only* $\simeq 0{\cdot}5 \times 10^{-7}$ of the conformations!) highlights the problem of designing an efficient search.

3. Monte Carlo Methods

The Monte Carlo (MC) method (Metropolis *et al.*, 1953; Kirkpatrick *et al.*, 1983; Aarts & Korst, 1989) for protein folding can be described in the following general algorithm. (1) Start from a random coil conformation. (2) From a conformation S_1 with energy E_1, make a single random change of the conformation to conformation S_2 and evaluate its energy E_2. (3) If $E_2 \leq E_1$, then accept the change to conformation S_2, otherwise decide, non-deterministically, whether to accept the change according to the energy increase with the change. Usually the criterion is of the form: accept if:

$$Rnd < \exp\left[\frac{E_1 - E_2}{c_k}\right],$$

where *Rnd* is a random number between 0 and 1 and c_k is gradually decreased (cooled) during the simulation to achieve convergence. If the change was not accepted, then retain the former conformation S_1. (4) If the stop criterion is not met, then repeat steps (2) to (4).

Theoretically, with the appropriate cooling scheme this algorithm is guaranteed to converge to the global minimum, but it must be remembered that the number of steps in such an "appropriate" scheme is strikingly large. It is actually larger than the exponential number of steps needed to enumerate the whole space! (The theoretical aspects of MC methods are discussed in Aarts & Korst (1989), chapter 3.) Practically, the selection of the cooling scheme is crucial for the success of the process. Usually, c_k is cooled linearly (i.e. $c_{k+1} = \alpha c_k$, where α is a constant smaller than but close to 1). As the minimum energy value is not known in advance and as the algorithm does not always converge to the lowest energy level it has encountered, the usual procedure is to run the algorithm as long as the computer resources permit, while decreasing c_k gradually and keeping track of the lowest energy solution found.

In our model the initial conformation is fully extended (i.e. a straight line). The random change is performed by randomly selecting an amino acid and rotating the C-terminal portion of the chain around that amino acid (see Fig. 1). For the 20 amino acid example above, the algorithm was run for 50,000,000 steps, about one half of which yielded valid (self-avoiding) conformations. When a valid conformation was encountered its energy was evaluated. The c_k was reduced very slowly from 2 to 0·15 (c_k was decreased by $\alpha = 0{\cdot}99$ every 200,000 steps), reducing the chance of accepting a move with a cost of +1 from 0·6 to 10^{-3}. The simulation was run five times. In these runs, an optimal conformation with energy −9 was found after 3,199,813, 8,823,199, 469,984, 292,443 and 7,367,375 energy evaluations, respectively.

4. Genetic Algorithms

In implementing a genetic algorithm, one has to choose the appropriate method of encoding the data, the size of the population, the specific manner of applying the genetic operators, and the population pruning scheme. Our implementation of the genetic algorithm is unique in that the solutions are not encoded as binary strings but rather are the conformations themselves which are treated directly in the spirit of genetic operators. The process starts with N extended structures. In each generation each structure is subject to a number of mutation steps. Each mutation is the same as a single MC step described above and is subject to similar acceptance criteria as in a MC process. At the end of this MC stage the crossover operation is performed. The chance $p(S_i)$ of a structure being selected for crossover is proportional to its energy value E_i, i.e.:

$$p(S_i) = \frac{E_i}{\sum_{j=1}^{N} E_j}.$$

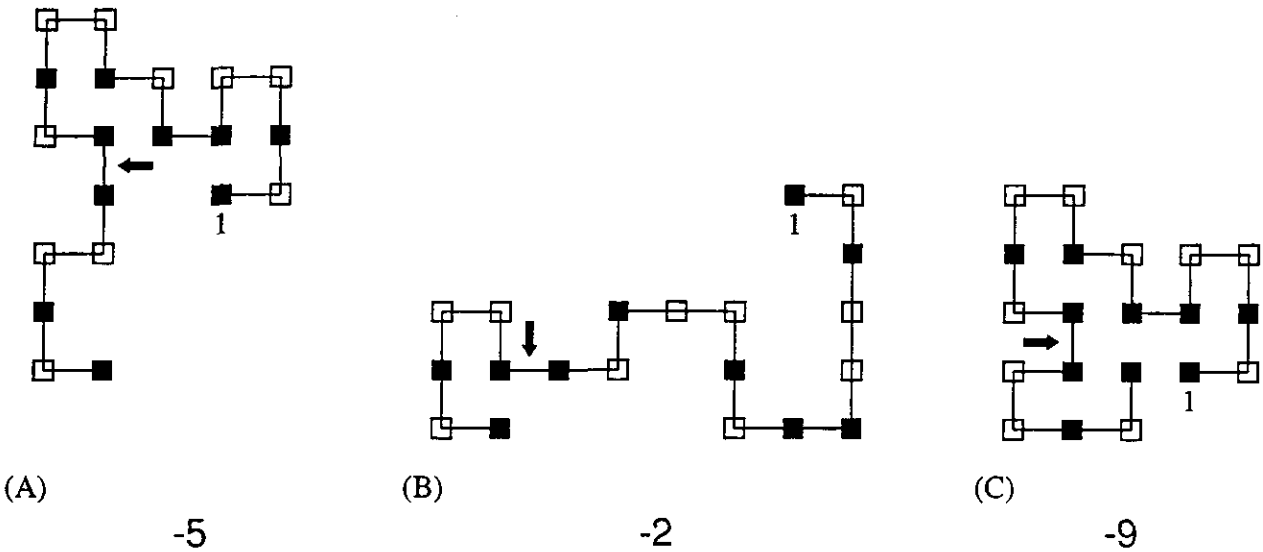

Figure 2. The genetic algorithm. The process starts with a population of fully extended structures. Each structure undergoes a MC stage followed by a crossover stage. In the crossover stage, pairs of structures are randomly (based on their energies) cut and pasted. In this example the cutpoint was randomly chosen to be after residue 14. Joining the first 14 residues of (A) with the last 6 residues of (B) and applying a randomly chosen 270° rotation at the joint achieves the compact structure in (C). In this case, the energy value of the hybrid (C) is −9, lower than the energies −5 and −2 of its "parents". The hybrid is always accepted if its energy is lower than the averaged energies of its parents, or non-deterministically accepted according to its energy increase.

Thus, the lower energy conformations have a higher chance of being selected. For a pair of selected structures a random point is chosen along the sequence and the N-terminal portion of the first structure is connected to the C-terminal portion of the second structure (see Fig. 2). As there are three ways to join the parts together (connecting the chains with angles of 0°, 90° or 270°), these possibilities are tested in a random order to find one that is valid (i.e. where no residue from one structure occupies a lattice point used by a residue from the other). If none of the three ways lead to a self-avoiding structure, then another pair of structures is selected. Once a valid structure S_k is created, its energy E_k is evaluated and compared to the averaged energy $\bar{E}_{ij} = (E_i + E_j)/2$ of its "parents". The structure is accepted if $E_k \leq \bar{E}_{ij}$, or if the energy will be increased based on the decision:

$$Rnd < \exp\left[\frac{\bar{E}_{ij} - E_k}{c_k}\right].$$

This crossover operation is repeated until $N-1$ new accepted hybrid structures have been constructed to constitute the population of the next generation. In addition, the lowest energy conformation in each generation is directly replicated to the next generation. We allow a higher acceptance rate for bad moves that increase the energy for mutation steps than for crossovers. This strategy maintains the diversity of the population and prevents premature convergence to a few low energy conformations.

For the case of the 20-residue long molecule described above, we performed a simulation with a population of 200 structures with 20 steps of individual mutations per structure between crossover stages. Five of the structures after the fifth and the tenth generations are shown in Figure 3. Each application of a genetic operator is counted as a step. Thus, a generation takes $20 \times 200 = 4000$ mutation steps plus the number of crossover trials it takes to get 200 new valid structures, typically around 900 steps. When a valid conformation is encountered, its energy is evaluated. The simulation was run for five times. The optimal conformation was found after 40,521, 32,708, 30,492, 36,026 and 68,868 energy evaluations, respectively. Note that for this example the GA runs found the solution much faster than the MC runs described above.

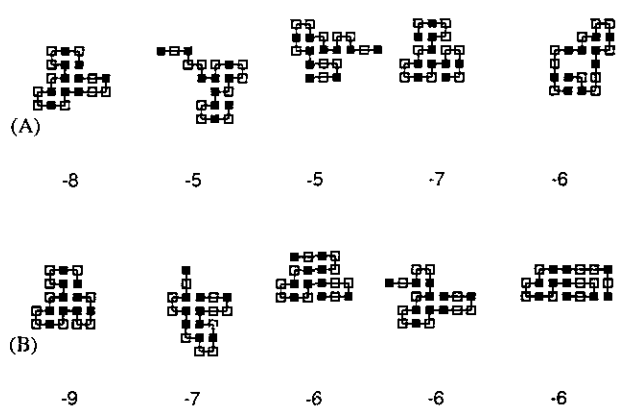

Figure 3. A snapshot during a GA run. (A) Five structures from the 5th generation of the run, and (B) 5 structures from the 10th generation. Note that the early structures are less compact and less organized than the ones achieved later in the run. This is reflected in lower energies for the later structures, including one with the lowest possible energy for this sequence: −9.

Table 2
Comparison of genetic algorithm (GA) and Monte Carlo (MC) folding simulations

Length[a]	Optimal energy[b]	GA[c]	MC[d]	Long MC[e]	Multiple MC[f]
20	−9	−9 (30,492)	−8	−9 (292,443)	−9 (41%)
24	−9	−9 (30,491)	−8	−9 (2,492,221)	−9 (19%)
25	−8	−8 (20,400)	−7	−8 (2,694,572)	−7 (100%)
36	−14	−14 (301,339)	−12	−13 (6,557,189)	−13 (5%)
48	−22	−22 (126,547)	−18	−20 (9,201,755)	−19 (3%)
50	−21	−21 (592,887)	−19	−21 (15,151,203)	−20 (1%)
60	−34	−34 (208,781)	−31	−33 (8,262,338)	−32 (7%)
64	−42	−37 (187,393)	−31	−35 (7,848,952)	−32 (2%)

Eight sequences were tested. For each, the GA method is compared with several MC variants. We check a MC that uses as many energy evaluations as the GA, a much longer MC and a multiple MC running 100 different simulations. The GA results are very superior to those from the comparable MC which failed to find the optimal solution in all cases, and are also better than the longer MC runs. In the 36, 48 and 60-residue long sequences, the lowest energy conformation was found by the GA and was not found in any of the MCs. All the methods failed to find the lowest energy conformation for the longest sequence.

[a] The following sequences were tested:

(20) BWBWWBBWBWWBWBBWWBWB;
(24) BBWWBWWBWWBWWBWWBWWBWWBB;
(25) WWBWWBBWWWWBBWWWWBBWWWWBB;
(36) WWWBBWWBBWWWWWBBBBBBBWWBBWWWWBBWWBWW;
(48) WWBWWBBWWBBWWWWWBBBBBBBBBBWWWWWBBWWBBWWBWWBBBBB;
(50) BBWBWBWBWBBBBWBWWWBWWWBWWWWBWWWBWWWBWBBBBWBWBWBWBB;
(60) WWBBBWBBBBBBBBWWWBBBBBBBBBBWBWWWBBBBBBBBBBBBWWWWBBBBBBWBBWBW;
(64) BBBBBBBBBBBBWBWBWWBBWWBBWWBWWBBWWBBWWBWWBBWWBBWWBWBWBBBBBBBBBBBB.

[b] The optimal energies were determined from the designed structures. For the first 2 sequences these energies were validated by full enumeration of the energies of all valid conformations.

[c] The GA was run with 200 structures for 300 generations. For the mutation stage the cooling scheme starts with $c_k = 2$ and is cooled by $c_k = 0{\cdot}97c_k$ every 5 generations. The crossover stage starts with $c_k = 0{\cdot}3$ and is cooled by $c_k = 0{\cdot}99c_k$ every 5 generations. For each sequence the simulation was run 5 times. For the most efficient run we report the lowest energy value achieved together with the number of conformations scanned before that value was found.

[d] The MC was run to scan the number of conformations given in the GA column. c_k starts as 2 and decreases as $c_k = 0{\cdot}95c_k$ every 1/50 of the number of conformations. The simulation was repeated 5 times, and the lowest energy value found during these simulations is listed.

[e] The long MC run performed 50,000,000 steps ($c_k = 2$; $c_k = 0{\cdot}99c_k$ every 200,000 steps). The simulation was run 5 times. The lowest energy found during the best run with the number of conformations that were scanned to find this value are reported.

[f] Each multiple MC simulation consisted of 100 parallel runs of 500,000 steps, starting with $c_k = 2$ and cooling by $c_k = 0{\cdot}95c_k$ every 10,000 steps. The best result found by any of the runs is reported together with the percentage of the runs that achieved this value.

5. A Comparison between the Methods

The simple model enables us to compare the performances of the two methods. The main factors to be compared are the number of energy evaluations needed to find one of the lowest energy conformations, and the lowest energy found for a given number of energy evaluations. The genetic algorithm is not significantly more costly per step than the regular Monte Carlo method: most of the genetic algorithm steps are mutations which are the same as the regular MC steps, and a crossover is not much more expensive. The overhead involved with the population book-keeping is not high and, in any case, the dominant factor is the energy calculation, identical for both methods and performed once for each valid conformation in each method.

The two methods were compared for a set of sequences which were designed to have particular low energy folds. Each genetic algorithm run had a population of 200 conformations for 300 generations. Each generation consisted of L mutation steps per structure (where L residues is the length of the sequence) followed by a crossover stage. The results are given in Table 2. In order to make a thorough comparison between the GA and MC methods, three variants of the latter were used. In the first variant, the number of energy evaluations was set to the minimum required to find the optimal solution with the GA. In the second variant, a much longer (50,000,000 steps) MC simulation was used. Third, in order to encourage the MC procedure to explore more of the conformational space, multiple MC runs were performed. For this purpose, 100 runs of 500,000 steps each (i.e. equivalent computer resources to the single long run) were carried out.

Table 2 summarizes the results obtained on eight different sequences ranging from 20 to 64 residues in length. The GA finds one of the lowest energy level conformations rapidly for all but the longest sequence, where it does not succeed with the allocated computing resoures. Single MC runs using the same number of energy evaluations in no case found a correct solution, and for the longer sequences, found only solutions with relatively high energies. The much longer single MC runs did find a conformation with the lowest energy level for four of the sequences (mostly the shorter ones), with the use of ten to 100 times as many steps. For four of the longer sequences the MC runs failed to find the lowest energy conformations. An example of one of the cases where the GA succeeds, and the long MC

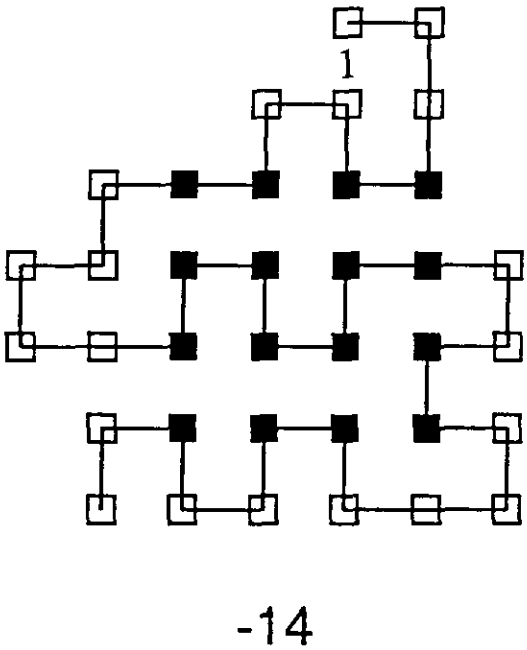

Figure 4. The lowest energy structure for a 36-residue long sequence. This structure was produced by our GA algorithm after 301,339 energy evaluations. The structure contains a hydrophobic core formed by a helical trace of the chain. Much longer MC (50,000,000 steps) runs were unable to find such a low energy conformation.

does not, is shown in Figure 4. The multiple short MC runs succeeded with only the two short sequences and failed for all others. Thus, at least for this type of model of the folding process, the GA method is very superior to the more traditional MC approach.

6. Discussion

Other workers have also noted severe limitations with the convergence of Monte Carlo simulations on a lattice. In a similar three-dimensional model (Shakhnovich *et al.*, 1991), it has been recently reported that a Monte Carlo algorithm, using a variant allowing very local single step changes in the structure, failed to find the global minimum in all but one of the 30 sequences (of length 27 residues) investigated. The failure of these Monte Carlo searches was attributed to the lack of specific folding pathways encoded in the sequences.

In choosing a general search technique we have tacitly assumed that we wish to find the lowest energy conformations available to the sequence. In fact, it is not known whether the functional conformation of a globular soluble protein is necessarily at the global free energy minimum. It has been reasoned that the number of possible conformations is so large that proteins must fold by following a sequence-encoded pathway from the unfolded to the folded state which, in some sense, guides them to the appropriate functional minimum (Levinthal, 1968). We have shown that protein folding, at least on a lattice, is a member of the class of NP-complete problems, and therefore there probably exists no general search algorithm that can be guaranteed to find the global free energy minimum for real proteins (Unger & Moult, 1993). The real folding process may thus end up in a functional conformation that is not the global minimum of free energy. As the model becomes more computationally demanding, the general search algorithms are more likely to fail. While genetic algorithms can be used as general search procedures, they have special properties that are compatible with the folding pathways hypothesis. As we will discuss below, genetic algorithms may be able to mimic the folding pathway rather than conducting a hopeless brute force search for the global minimum.

The pathway hypothesis implies that search algorithms may only be successful if they in some way mimic pathway behavior. Real protein folding pathways have usually been supposed to depend on local regions of the chain folding first (early folding units) and the rest of the structure forming around these by a combination of diffusion/collision or propagation processes (Wetlaufer, 1973; Karplus & Weaver, 1976; Moult & Unger, 1991). If this is the case, GAs may be particularly suited for reproducing pathway behavior, since they have the property of tending to preserve local favorable conformational features through successive generations. This property is based on the Schema concept (Holland, 1975). A schema is a pattern used to describe a feature common to many current solutions. Holland has pointed out that short patterns that have above average performance will receive increasing attention during a GA procedure, while below average patterns will be rapidly abandoned. In protein structure applications GAs may be able to produce a large number of local substructures, concentrate on the favorable ones, and then find the exact way in which these local substructures should be assembled to form the full structure. In this sense, genetic algorithms may be considered not as conducting a search on a population of structures, but as sampling points in the conformational space of a single molecule along the folding pathway. We will next apply the methods developed here to more realistic protein models to see if this is true.

This work was supported in part by NIH grant 41034 to J.M. We thank Hue Sun Chan for providing us with his enumeration data.

References

Aarts, E. & Korst, J. (1989). *Simulated Annealing and Boltzmann Machines*, John Wiley & Sons, New York.

Barber, M. N. & Ninham, B. W. (1970). *Random and Restricted Walks*, Gordon and Breach, New York.

Clearwater, S. H., Huberman, B. A. & Hogg, T. (1991). Cooperative solution of constaint satisfaction problems. *Science*, **254**, 1181–1183.

Covell, D. G. & Jernigan, R. L. (1990). Conformation of folded proteins in restricted spaces. *Biochemistry*, **29**, 3287–3294.

Crippen, G. M. (1991). Prediction of protein folding from

amino acid sequences over discrete conformation spaces. *Biochemistry*, **30**, 4232-4237.

Davidor, T. (1990). *Genetic Algorithms and Robotics: A Heuristic Strategy of Optimization*, World Scientific, New Jersey.

Dill, K. A. (1990). Dominant forces in protein folding. *Biochemistry*, **29**, 7133–7155.

Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, Reading, MA.

Guttman, A. J., Ninham, B. W. & Thompson, C. J. (1968). Determination of critical behavior in lattice statistics from series expansions. *Phys. Rev.* **172**, 554–558.

Holland, J. H. (1975). *Adaptation in Natural and Artificial Systems*, The University of Michigan Press, Ann Arbor.

Huberman, B. A. (1990). The performance of cooperative processes. *Phys. D.* **42**, 38–47.

Karplus, M. & Weaver, D. L. (1976). Protein folding dynamics. *Nature (London)*, **260**, 404–406.

Kirkpatrick, S., Gellat, C. D. & Vecchi, M. P. (1983). Optimization by simulated annealing. *Science*, **220**, 671–680.

Lau, K. F. & Dill, K. A. (1990). Theory for protein mutability and biogenesis. *Proc. Nat. Acad. Sci., U.S.A.* **87**, 638–642.

Levinthal, C. (1968). Are there pathways for protein folding? *J. Chem. Phys.* **65**, 44–45.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H. & Teller, E. (1953). Equation of state calculations by fast computing machines. *J. Chem. Phys.* **21**, 1087–1092.

Moult, J. & Unger, R. (1991). An analysis of protein folding pathways. *Biochemistry*, **30**, 3816–3824.

Shakhnovich, E., Farztdinov, G., Gutin, A. M. & Karplus, M. (1991). Protein folding bottlenecks: a lattice Monte Carlo simulation. *Phys. Rev. Letters*, **67**, 1665–1668.

Skolnick, J. & Kolinsky, A. (1990). Simulations of the folding of globular proteins. *Science*, **250**, 1121–1125.

Unger, R. & Moult, J. (1993). Finding the lowest free energy conformation of a protein is a NP-hard problem: proof and implications. *Bull. Math Biol.* in the press.

Wetlaufer, D. B. (1973). Nucleation, rapid folding, and globular interchain regions in proteins. *Proc. Nat. Acad. Sci., U.S.A.* **70**, 697–701.

Edited by F. Cohen