Journal of Computer Science 4 (7): 525-529, 2008
ISSN 1549-3636

Solving Protein Folding Problem using Elitism-Based Compact Genetic Algorithm

Amr Badr, Ibtehal M. Aref, Basma M. Hussien and Yosr Eman
Department of Computer Science, Faculty of Computers and Information,
Cairo University, Cairo, Egypt

Abstract: Proteins are vital components of living cells. A number of diseases such as Alzheimer's, Cystic fibrosis and Mad Cow diseases are shown to result from misfunctioning of proteins. **Problem statement:** Protein folding problem is the process of predicting the optimal 3D molecular structure of a protein, or tertiary structure, which is an indication of its proper function. **Approach:** An enhancement over persistent elitist compact genetic algorithm (pe-cGA) was made to minimize the energy of proteins indicating how far it is from its optimal 3D structure. Energy was calculated using the Empirical Conformational Energy Program for Peptides (ECEPP) package. **Results:** Experiments were performed on the Met-enkephalin protein. The enhanced algorithm reached an energy of -7.378 in 140,000 iterations surpassing the Distributed Genetic Algorithm (DGA) which reached the same energy in 700,000 iterations. A comparison was also made with the Breeder Genetic Algorithm (BGA) which did not reach this energy in the first place. **Conclusions/Recommendations:** Results show that the enhanced algorithm is superior to DGA and BGA and a computational alternative to costly laboratory methods and an efficient means for solving organic docking problems.

Keywords: Estimation of distribution algorithm, elitism-based compact genetic algorithm, persistent elitist compact genetic algorithm, non-persistent elitist compact genetic algorithm

INTRODUCTION

Proteins are fundamental components of all living cells. The bacteria that infect us, the plants and animals we eat, the hemoglobin that carries oxygen to our tissues, the insulin that signals our bodies to store excess sugar, the antibodies that fight infection, the actin and myosin that allow our muscles to contract, and the collagen that makes up our tendons and ligaments (and even much of our bones) are all examples of proteins. To make proteins, ribosomes string together amino acids into long, linear chains. Like shoelaces, these chains loop about each other in a variety of ways (i.e., they fold). But, as with a shoelace, only one of these many ways allows the protein to function properly. Yet lack of function is not always the worst scenario.

Recent discoveries have shown that some diseases (Alzheimer's disease, Cystic fibrosis, Mad Cow disease, and many cancer types) are the result of misfolded proteins. Also, protein misfolding is behind many of the unexpected difficulties biotechnology companies encounter when trying to produce human proteins in bacteria.

A misfolded protein can actually poison the cells around it, so misfolded protein could be worse than a normally folded one.

The prediction of molecular structure (polypeptide's native conformation) of a protein given only its amino acid sequence is not an easy task, but has numerous potential applications[1]. This structure prediction problem is commonly referred to as the protein folding problem. Efforts to solve it nearly always assume that the native conformation corresponds to the global minimum free energy state of the system. Given this assumption, a necessary step in solving the problem is the development of efficient global energy minimization techniques. This is a difficult optimization problem because of the non-linear and multi-modal nature of the energy function.

The motivation of this work is to find the optimal 3D structure of protein (angles of amino acids) to be used in the treatment by using Estimation of Distribution Algorithm (EDA)[2].

MATERIALS AND METHODS

The algorithm:

Estimation of Distribution Algorithm (EDA): Instead of using traditional recombination and mutation operators, Estimation of Distribution Algorithm (EDA)[1] generates offspring population according to the estimated probabilistic model of parent population.

Corresponding Author: Amr Badr, Department of Computer Science, Faculty of Computers and Information, Cairo University, Cairo, Egypt

Also, EDAs express the interrelations explicitly through the joint probability distribution associated with the individuals of variables selected at each generation. The probability distribution is calculated from a database of selected individuals of previous generation. Then offspring are generated from sampling this probability distribution. Neither crossover nor mutation is applied in EDAs. But the estimation of the joint probability distribution associated with the database containing the selected individuals is not an easy task. The flow chart of EDA is shown in the Fig. 1.

Different EDA approaches:

Independent variables: The easiest way to calculate the estimation of probability distribution is to consider all the variables in a problem as uni-variate (independent). In Uni-variate Marginal Distribution Algorithm (UMDA)[3], the joint probability distribution is factorized as a product of independent uni-variate marginal distribution.

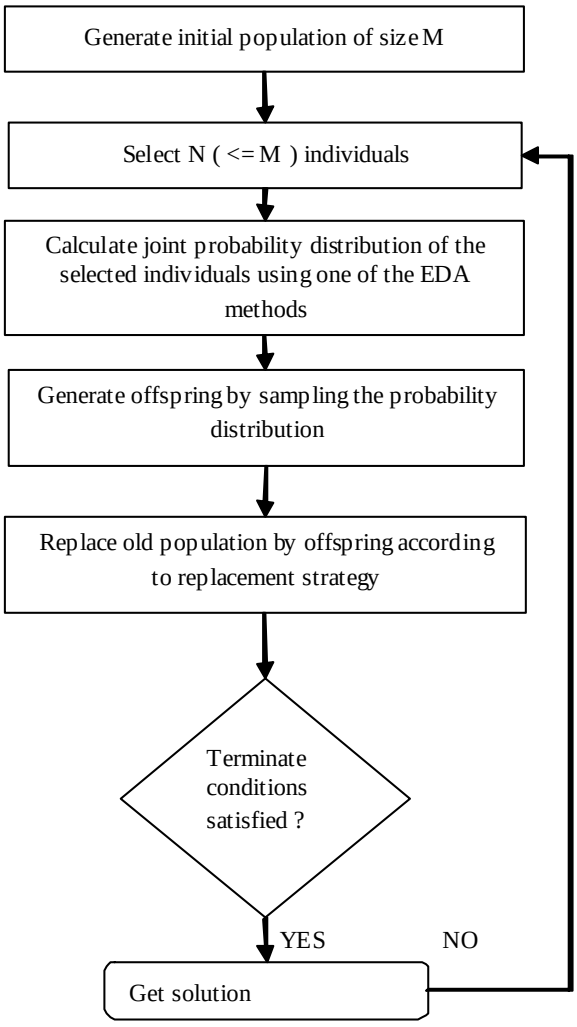

Fig. 1: EDA flowchart

Bi- variate dependencies: To solve the problem of pair wise interaction among variables, population based Mutual Information Maximization for Input Clustering (MIMIC)[4] Algorithm, Combining Optimizers with Mutual Information Trees (COMIT)[5], Bi- variant Marginal Distribution Algorithm (BMDA)[6] were introduced. Where there is at most two-order dependency among variables.

Multiple dependencies: The factorization of the joint probability is calculated as a product of marginal distribution of variable size. These marginal distributions of variable size are related to the variables that are contained in the same group and to the probability distribution associated with them (variables are strongly related).

In this study, the multiple dependencies are used because all the variables (protein angles) are strongly related.

The MVDA has several types like Factorized Distribution Algorithm (FDA)[7], Extended Compact Genetic Algorithm, Bayesian Optimization Algorithm (BOA)[8], Estimation of Bayesian Networks Algorithm (EBNA)[9] and Elitism-based Compact Genetic Algorithm (ECGA)[10].

Elitism-based Compact Genetic Algorithm (ECGA): There is two elitism-based compact Genetic Algorithms (cGAs)-persistent elitist compact genetic algorithm (pe-cGA), and non-persistent elitist compact Genetic Algorithm (ne-cGA)[11]. The aim is to design efficient compact-type GAs by treating them as Estimation of Distribution Algorithms (EDAs) for solving difficult optimization problems without compromising on memory and computation costs. The idea is to deal with issues connected with lack of memory-inherent disadvantage of cGAs-by allowing a selection pressure that is high enough to offset the disruptive effect of uniform crossover. The point is to properly reconcile the cGA with elitism. The pe-cGA finds a near optimal solution (i.e., a winner) that is maintained as long as other solutions (i.e., competitors) generated from probability vectors are no better. It attempts to adaptively alter the selection pressure according to the degree of problem difficulty by employing only the pair-wise tournament selection strategy.

Parameters.
n: population size, *l:* chromosome length,
Echrom: elite chromosome, *Nchrom:* new chromosome.

Step 1.
Initialize probability vector
for i:=1 to l do p[i] := 0.5;

Step 2.
Generate one chromosome from the probability vector if the first generation then

Echrom := generate (p);
/*initialize the elite chromosome*/

Nchrom := generate (p);
/*generate a new chromosome*/

Step 3.
Let them compete and let the winner inherit persistently
Winner, loser := compete (Echrom, Nchrom);
Echrom := winner;
/*update the elite chromosome*/

Step 4.
Update the probability vector
for i:= 1 to l do
if winner[i] ? loser[i] then
if winner[i] == 1 then p[i] := p[i] + 1/n;
else p[i] := p[i] – 1/n;

Step 5.
Check if the probability vector has converged.
Go to **Step 2,** if it is not satisfied.

Step 6.
The probability vector represents the final solution.

Fig. 2: the pe-cGA pseudo code

The ne-cGA further improves the performance of the pe-cGA by avoiding strong elitism that may lead to premature convergence. It may seem that the ne-cGA gives better results, and this is true for some problems. But in this work, we found out (from experimental results) that the pe-cGA is better and more suitable for the protein folding problem.

The pseudo code of the pe-cGA is as in Fig. 2.

But from experimental results the pe-cGA alone did not give good results, so we needed to make enhancement over it to get better results.

Enhancement over pe-cGA: In this study, pe-cGA is proposed in solving protein folding problem with the addition of two modifications: mutation, and keeping the best solution so far.

```
if fitness (Echrom) > fitness (Nchrom)
    {
        chrom := mutate(Echrom);
        Evaluate(chrom);
        if fitness (chrom) > fitness (Echrom)
            Echrom := chrom;
    }
    else
    {
        chrom := mutate(Nchrom);
        Evaluate(chrom);
        if fitness (chrom) > fitness (Nchrom)
            Nchrom := chrom;
    }
```

Fig. 3: Mutation

The first modification is the addition of mutation. Mutation will be performed by adding a secondary tournament to each cycle that compares the performance of the current champion string with a mutated version of itself. If the mutated version wins, it replaces the old champion as the elite string for the next tournament. Note that our implementation of mutation presumes elitism. This allows periodic sampling of individuals around the champion independent of the current state of the genome probability vector. We can more formally describe this operation by modifying the standard cGA to contain elitism and by adding a new step as described in the pseudo code of Fig. 3.

We also found that considering the probability vector as the final solution is not the optimal solution, so the second and final modification is to consider the final solution as the best individual (in our case, the one with minimum energy) in all generations.

The complete algorithm became as in Fig. 4.

The algorithm was implemented in C/C++ using Microsoft Visual Studio. Empirical Conformational Energy Program for Peptides (ECEPP) package was used to evaluate energy of proteins.

RESULTS

The target protein in this study is Met-enkephalin[12]. Met-enkephalin is the protein that consists of five amino acids.

As we search for the structure that give minimum energy, the fitness of individuals is calculated in terms of energy. In our case the best individual is the individual with the minimum energy. We used an energy evaluator called Empirical Conformational Energy Program for Peptides (ECEPP)[13] to evaluate the individual energy.

```
Parameters.
  n: population size, l: chromosome length,
  Echrom: elite chromosome, Nchrom: new chromosome.

Step 1.
 Initialize probability vector
  for i:=1 to l do p[i] := 0.5;

Step 2.
 Generate one chromosome from the probability vector if the
first generation then

 Echrom := generate (p);
/*initialize the elite chromosome*/

 Nchrom := generate (p);
/*generate a new chromosome*/

Step 3.
 Let them compete and let the winner inherit persistently
Winner, loser := compete (Echrom, Nchrom);
 Echrom := winner;
 /*update the elite chromosome*/

Step 4.
Update the probability vector
  for i:= 1 to l do
    if winner[i] ? loser[i] then
      if winner[i] == 1 then p[i] := p[i] + 1/n;
            else p[i] := p[i] – 1/n;
```

Fig. 4: The pe-cGA with enhancement

A comparative study is made with two other algorithms (described below) that solve the same problem on Met-enkephalin protein using "ECEPP. The comparison is based on the best fitness (minimum protein energy) that each algorithm has reached with respect to the overhead (number of energy evaluations) needed to reach this result.

In Distributed Genetic Algorithm(DGA)[14], the total population is divided into sub populations. Each sub population is often called "island". In each sub population, normal genetic operations are performed for several generations. After a certain number of generations, some of the individuals are chosen and are moved to the other island. This operation is called "migration". Because the population size in each island is small, the early convergence may happen in each island. However, the migration operation prevents the early convergence and
maintains the diversity of the solutions during the search. Breeder Genetic Algorithm (BGA)[15], at each generation, the T % best individuals within the current population of N elements are selected. T % is called truncation rate and its typical values are within the range 10 to 50%. The selected individuals are randomly recombined and their offspring are mutated, so as to generate a new population of N-1 elements.

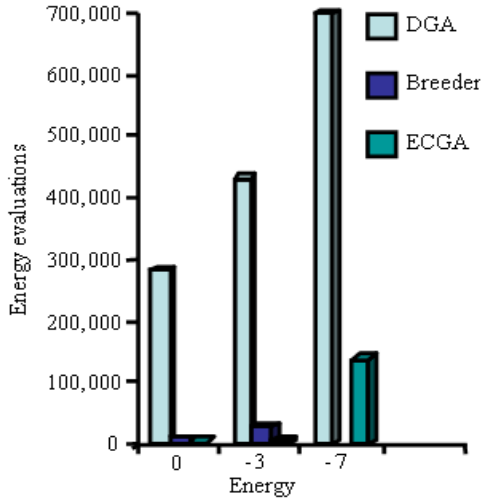

Fig. 5: Comparing results

Table 1: Comparing results

Energy/Algorithm	(0)	(-3)	(-7.378)
DGA	280,000	430,000	700,000
Breeder	10,150	28,320	-
ECGA	3300	3800	140,000

The best individual of the old population is then added to the new population, and the cycle of life continues. By doing so, the best individuals are treated as super-individuals and mated together, hoping that this can lead to a fitter population.

Table 1 shows the performance of DGA and BGA against our enhanced ECGA. The number of evaluations needed is recorded for each algorithm at minimum energy. Figure 5, emphases the difference in performance between the two algorithms and our proposed one. Our proposed algorithm reaches requires little overhead than other two algorithms.

Also we can observe that the breeder algorithm did not reach -7.378 fitness. The DGA reached this fitness but we do not know exactly with how many evaluations, but we can note that our algorithm has reached this fitness with less than half the number of evaluations that the DGA needed to reach fitness 0.

From these results, it is shown that Elitism-based Compact Genetic Algorithm is very effective in solving protein folding problem.

DISCUSSION

From the formulation of the protein folding problem, the angles that describe the protein structure in 3D are strongly inter-related and dependent. This is strongly tackled in Estimation of Distribution Algorithms (EDAs) and consequently the enhanced algorithm that model the interactions between chromosomes in terms of a probability distribution

vector. Thus, the enhanced algorithm moves progressively towards the optimal interacting angles 3D structure, by generating individuals conforming with higher fitness probability distribution individuals.

Moreover, it appears that the enhanced algorithm correctly balances exploration and exploitation needed for this problem. Distributed genetic algorithms (DGA) relies more on exploration rather than exploitation. Breeder genetic algorithm (BGA) did not reach the energy reached by enhanced algorithm in the first place. However, the overhead incurred in DGA and BGA is more than that for the enhanced algorithm.

CONCLUSION

In this study, the molecular structure of Met-enkephalin protein is predicted. The structure is always assumed to be the global minimum free energy state of the system. For this optimization problem, an enhancement of Elitism-based Compact Genetic Algorithm (ECGA) is made to minimize the protein energy. Results show that the enhanced ECGA have little overhead in terms of number of evaluations needed.

REFERENCES

1. Hue, S.C. and K.A. Dill, 1993. The protein folding problem. Phys. Today, 46: 24-32. DOI: 10.1063/1.881371.
2. Larrañaga, P. and J.A. Lozano, 2001. Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation. 2nd Edn., Springer Verlag, London, UK., ISBN: 978-0-7923-7466-4, pp: 416.
3. Heinz, M. and T. Mahnig, 2001. Evolutionary Algorithms: From Recombination to Search Distributions. In: Theoretical Aspects of Evolutionary Computation, Kallel, L., B. Naudts and A. Rogers (Eds.). Springer Verlag, London, UK., ISBN: 3-540-67396-2, pp: 135-173.
4. De Bonet, J.S., C.L. Isbell and P. Viola, 1997. MIMIC: Finding Optima by Estimating Probability Densities. In: Advances in Neural Information Processing Systems, Jordan, M., M. Mozer and M. Perrone (Eds.). MIT Press, Cambridge, MA., pp: 424-430.
5. Larrañaga, P., R. Etxeberria, J. Lozano and J. Pena, 2000. Combinatorial optimization by learning and simulation of Bayesian networks. Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence, June 30-July 3, Morgan Kaufmann, pp: 343-352. http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.30.8671.
6. Pelikan, M. And H. MÄuhlenbein, 1999. The Bivariate Marginal Distribution Algorithm. In: Advances in Soft Computing, Engineering Design and Manufacturing, Roy, R., T. Furuhashi and P. Chawdhry (Eds.). Springer-Verlag, pp: 521-535.
7. MÄuhlenbein, H. And T. Mahnig, 1999. The factorized distribution algorithm for additively decomposed functions. Proceedings of the 1999 Congress on Evolutionary Computation, July 6-9, Washington, DC., USA., pp: 752-759. DOI: 10.1109/CEC.1999.782008
8. Pelikan, M., D.E. Goldberg and Cantu-Paz, 2000. Linkage problem, distribution estimation and bayesian networks. Evolut. Comput., 8: 311-340. DOI: 10.1162/106365600750078808.
9. Etxeberria, R. and P. Larrañaga, 1999. Global optimization using bayesian networks. Proceedings of the 2nd Symposium on Artificial Intelligence, March 22-26, Havana, Cuba, pp: 332- 339. http://citeseerx.ist.psu.edu/showciting;jsessionid=350B89EF45EA562E596A0BFC2AE1F45D?cid=1091024.
10. Chang, W.A., K.P. Kim and R.S. Ramakrishna, 2003. A memory-efficient elitist genetic algorithm. Proceedings of the 5th International Conference on Parallel Processing and Applied Mathematics, September 7-10, Springer Verlag, pp: 552-559. http://books.google.com/books?id=VA-YU9xcDf4C&printsec=frontcover&dq=Parallel+Processing+and+Applied+Mathematics#PPA552,M1
11. Chang, W.A. and R.S. Ramakrishna, 2003. Elitism-based compact genetic algorithms. IEEE Trans. Evolut. Comput., 7: 367-385. DOI: 10.1109/TEVC.2003.814633.
12. Yuko, O., T. Kikuchi and H. Kawai, 1992. Prediction of low-energy structures of met-enkephalin by monte carlo simulated annealing. Chem. Lett., 21: 1275-1278. http://www.jstage.jst.go.jp/login?mid=cl&sourceurl=/article/cl/21/7/1275/_pdf&lang=en
13. Momany, F.A., R.F. McGuire, A.W. Burgess and H.A. Scheraga, 1975. Energy parameters in polypeptides. VII. geometric parameters, partial atomic charges, nonbonded interactions, hydrogen bond interactions and intrinsic torsional potentials for the naturally occurring amino acids. J. Phys. Chem., 79: 2361-2381. DOI: 10.1021/j100589a006.
14. Tanese, R., 1989. Distributed genetic algorithms. In Proceeding of the 3rd International Conference on Genetic Algorithms, June 1-6, Morgan Kaufman Publishers, San Francisco, CA., USA., pp: 434-439. http://portal.acm.org/citation.cfm?id=657245&dl=ACM&coll=GUIDE#.
15. MÄuhlenbein, H. and D. Schlierkamp-Voosen, 1993. The science of breeding and its application to the Breeder Genetic Algorithm (BGA). Evolut. Comput., 1: 335-360. DOI: 10.1162/evco.1993.1.4.335.