Krassimir Markov, Vitalii Velychko, Oleksy Voloshin
(editors)

Information Models of Knowledge

I T H E A®
KIEV – SOFIA
2010

Krassimir Markov, Vitalii Velychko, Oleksy Voloshin (ed.)

Information Models of Knowledge

ITHEA®

Kiev, Ukraine – Sofia, Bulgaria, 2010

ISBN 978-954-16-0048-1

First edition

Recommended for publication by The Scientific Concil of the Institute of Information Theories and Applications FOI ITHEA

ITHEA IBS ISC: 19.

This book maintains articles on actual problems of research and application of information technologies, especially the new approaches, models, algorithms and methods fot information modeling of knowledge in: Intelligence metasynthesis and knowledge processing in intelligent systems; Formalisms and methods of knowledge representation; Connectionism and neural nets; System analysis and sintesis; Modelling of the complex artificial systems; Image Processing and Computer Vision; Computer virtual reality; Virtual laboratories for computer-aided design; Decision support systems; Information models of knowledge of and for education; Open social info-educational platforms; Web-based educational information systems; Semantic Web Technologies; Mathematical foundations for information modeling of knowledge; Discrete mathematics; Mathematical methods for research of complex systems.

It is represented that book articles will be interesting for experts in the field of information technologies as well as for practical users.

General Sponsor: Consortium FOI Bulgaria (www.foibg.com).

Printed in Ukraine

ISBN 978-954-16-0048-1

C\o Jusautor, Sofia, 2010

PROTEIN STRUCTURE PREDICTION ON A THREE-DIMENSIONAL TRIANGULAR LATTICE

Leonid Hulianytskyi, Vitalina Rudyk

***Abstract**: The protein tertiary structure prediction problem is examined, which is one of up-to-date problems of computational biology. The results received earlier for two-dimensional case are extended for 3D and analyzed. Algorithms of local and stochastic search, ant colony optimization are proposed. The questions of software realization and their comparative research are discussed.*

***Keywords**: combinatorial optimization, protein structure prediction, HP model, local search, ant colony optimization.*

***ACM Classification Keywords**: G.1.6. Mathematics of Computing, Numerical Analysis, Optimization.*

***Conference topic**: Decision Making.*

Introduction

Defining features of the protein molecule remains a central problem in computational biology, molecular biology, biochemistry and physics. Three-dimensional protein structure analysis is a key for understanding and manipulating its biochemical and cellular functions. Knowing the protein shape is underlying for pharmacology and medicine, as the majority of drugs' influence is caused by their conjunction to the protein for stabilizing the natural structure or destroying the part of it responsible for harmful disorders. Thus, for developing drugs it is necessary to know the exact three-dimensional structure of molecules.

Protein is a macromolecule that is composed of amino acid sequence, which number varies from 20 to 40000 (mostly about 100-300). There are 20 different amino acids. The research of proteins and their functions is usually based on "sequence-structure-function" dogma. It means that the functionality of the protein is uniquely determined by its 3D structure, while the last is defined by its amino acid sequence. High cost of experiments, amount of required time and diversity of protein forms lead to the development of mathematical methods of determining the molecule shape.

To describe the forces that affect the protein folding complex and large-scale differential equation systems are used but it is impossible to solve them in practice. Therefore, the use of simplified models that produce approximate folding is widespread. Differential equation systems are simplified on basis of these approximations, and the more exact approximation was found, the easier system becomes.

Traditionally, two different approaches are used for solving this problem. The first one, called "de novo", is based on the principles of physics, namely the thermodynamic hypothesis, according to which the natural structure of the protein corresponds to the global minimum of its free energy. Another, "template", is founded on the idea of evolution principles, assuming that similar amino acid sequences correspond to similar structures, and uses data about known folds, statistically analyzing them [Белецкий, Васильев, Гупал, 2007].

The shortcoming of "de novo"-method is the fact that the very mechanism of protein folding is not completely clear. There is a problem known as "Levintal paradox" that lies in the fact that the average protein (200 amino acids long) would be searching its secondary structure of the 3200 possible (in the secondary structure each amino acid refers to one of three forms – α-helix, β-strand or irregular structure), each corresponding to a certain level of free energy. The protein can "feel" the stability of structure only after shaping it. The transition from one conformation to another is not faster than 10-13 seconds, so to scan all possible structures protein would spend about 1080 years (to compare, the lifetime of the Universe is 1010 years) [Финкельштейн, Птицин, 2005]. Moreover, as a rule, template methods are computationally simpler and give more accurate results. However, they are not universal - if given amino acid sequence is not like any of the known, predictable precision of the evaluated fold will be small.

The simplest of "de novo"-models is the HP model offered by Dill [Dill, Bromberg, Yue, Fiebig, Yee, Thomas, Chan, 1995]. According to it 20 amino acids are divided into two groups - hydrophobic (nonpolar) and hydrophilic (polar) residues, basing on their characteristics, so the input sequence is denoted as a word from the alphabet {H, P}, where the symbol P indicates a polar, and H - hydrophobic residues. The idea is that while folding hydrophobic residues move inside the molecule to prevent the contact with water, and the hydrophilic remain outside. According to the hypothesis, the protein shapes a structure that minimizes the contact area of hydrophobic residues with water or polar residues. Formalizing it, every amino acid residue is located in a certain lattice node (neighboring residues in the sequence - in the neighboring nodes), contacting residues are defined as those that are not neighbors in sequence, but are located in adjacent lattice sites, and the problem is to maximize the number of contacts between hydrophobic residues. Despite the simplicity of the HP model, it properly represents real protein folding processes.

More complex models take into account the weighted influence of each contacting hydrophobic residue on the objective function [Decatur, 1996], use space discretization, based on sampling angles between amino acids, avoiding lattice binding [Awadh, Bahamish, Abdullah, Abdul Salam, 2006], base on protein secondary structure, operating with relative positions of α-helices, β-strands and irregular structures [Paluszewski, Winter, 2008].

Various methods of combinatorial optimization are applied for solving the protein tertiary structure prediction problem in the HP model, such as simulation annealing algorithm, evolutionary algorithms, ant colonies optimization algorithms and others. Though the problem remains acute: open questions are improving the model (e.g. choosing more suitable lattice) and optimizing the time and accuracy of computing.

HP-model

In the current research for prediction of the protein tertiary structure Dill's HP-model was chosen, as the most common one. Let's describe it in detail.

Each of the 20 amino acids applies to one of the two classes - hydrophobic (H) or polar (P), according to their physical properties, so amino acid sequence can be denoted as a sequence $\mathrm{S} = \xi_1\xi_2...\xi_n$, $\xi_i \in \{H,P\}$, $i = \overline{1,n}$ of length n. Each residue is placed in a node of some discrete lattice (such correspondence is called fold) in such a way that adjacent elements in a sequence correspond to neighboring nodes of the lattice (connectivity property), defining self-avoiding path on it. Connected self-avoiding folding is considered to be feasible. According to Dill contact appears between those hydrophobic residues that are not adjacent in the sequence, but are located in neighboring nodes of the lattice. The energy of folding is calculated as a number of contacts in it with a negative sign

More formally, if each element of the amino acid sequence assigned to some node, the energy is calculated as follows:

$$E(S) = -\sum_{1 \le i \le j-2 \le n-2} I(L_i, L_j)\, h(\xi_i, \xi_j) \qquad (1)$$

where

$$I(L_i, L_j) = \begin{cases} 1, \text{ if nodes } L_i \text{ and } L_j \text{ are neighbors,} \\ 0 - \text{else,} \end{cases}$$

$$h(\xi_i, \xi_j) = \begin{cases} 1, \text{ if both } \xi_i \text{ and } \xi_j \text{ are hydrophobic,} \\ 0 - \text{else.} \end{cases}$$

The protein tertiary structure prediction is reduced to finding conformation with the lowest energy. In such interpretation the problem is proved to be NP-hard [Berger, Leighton, 1998; Crescenzi, Goldman, Papadimitriou, Piccolboni, Yannakakis, 1998].

Initially for 2D case Dill used square lattice. But one of its significant drawbacks is that the two residues can be adjacent in the lattice only when the number of elements in the sequence between them is even (it is known as parity problem). Thus, the sequence $(HP)^n$ on a square lattice will have no connections. That fact contradicts the natural representation. Moreover, such a strict limitation complicates the analysis of approximate algorithms, identifying artificially high lower energy bound, resulting in the fact that the algorithm accuracy estimation won't have any significant value in real problems. Therefore in current research the triangular lattice was chosen for two-dimensional case and its axonometric variant in case of three dimensions (Fig. 1) where there is no parity problem, as for any two positions in the sequence there exists a conformation with the corresponding residues assigned to neighboring nodes of the lattice.

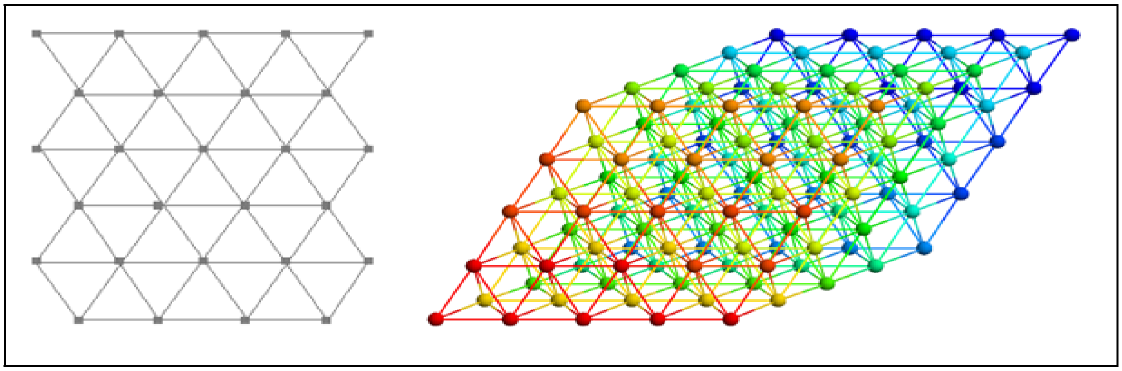

Fig. 1. Two-dimensional and axonometric triangular lattices.

Problem definition on three-dimensional lattice

Let $\xi_1\xi_2...\xi_n$, $\xi_i \in \{H,P\}$ indicate the sequence of amino acids of length n. To formulate the problem as the COP first of all it is necessary to define the decision space, in other words a fold should be denoted as some mathematical object [Гуляницкий, Рудык, 2010].

The simplest way to do it is to encode a folding as a sequence $c_1c_2...c_n$ of integer coordinates of each amino acid residue so that a point $c_i = (x_i, y_i, z_i)$, $x_i, y_i, z_i \in Z$ assigns to a lattice node with Cartesian coordinates $x_ie_1 + y_ie_2 + z_ie_3$, where $e_1 = (\frac{\sqrt{3}}{2}, \frac{1}{2}, 0)$, $e_2 = (0,1,0)$, $e_3 = (\frac{\sqrt{3}}{6}, \frac{1}{2}, \frac{\sqrt{33}}{6})$.

In such notation nodes (x_1, y_1, z_1) and (x_2, y_2, z_2) are neighbors if and only if $(x_1 - x_2, y_1 - y_2, z_1 - z_2) \in \mathrm{A}$,

$$\mathrm{A} = \{ (0,1,0),\ (1,0,0),\ (1,-1,0),\ (-1,0,0),\ (-1,1,0),$$
$$(0,-1,0),\ (0,0,1),\ (0,-1,1),\ (-1,0,1),\ (0,1,-1),\ (1,0,-1),\ (0,0,-1)\}.$$

This representation is called coordinate encoding and it is comfortable for checking if a fold is a self-crossing one, but in general case changing a single sequence element leads to losing a connectivity property. That complicates the process of algorithm development. Another drawback is that the same fold corresponds to some different assignments turning one into another by a turn or parallel transport. To exclude the equal variants from examination the additional constraints that are difficult to hold are imposed.

For avoiding these drawbacks another encoding, called absolute, is proposed. A fold is denoted as a sequence $a_1a_2...a_{n-1}$, $a_i \in \mathrm{A}$, $a_i = c_{i+1} - c_i$, it fixes the position of each amino acid residue (starting from second) with respect to the previous one. Every such code specifies a connected path in the lattice, and for checking self-avoidance constraint it is necessary to convert a sequence to coordinate encoding. Partially the problem with ambiguity is solved - parallel transport of fold does not change its code, but turns in various directions lead to existence of 12 different codes of the same conformation.

The other way to present a fold is to describe it by relative encoding. It represents a position of every residue depending on a part of foregoing fold, in such a way it fixes angles in the conformation. So in case of two-dimensional lattice six elements of absolute encoding represent six possible directions: “west”, “northwest”, “northeast”, “east”, “southeast” and “southwest", while there are only five in relative encoding and they mean "back-left”, “front-left”, “front”, “forward-right”, “back-right” [Гуляницкий, Рудык, 2009] (Fig. 2).

This representation has following advantages:

- each convolution corresponds to its relative code one-to-one;
- the absence of "back" partially direction solves a problem with self-crossing in a fold.

A feature is the fact that the modification of single sequence element of relative code leads to a turn of following part of the fold, while in absolute encoding this action parallel transfers the same part. Thus comparing with absolute encoding a small change of code causes more significant change of fold energy value, but there is higher probability for self-crossing to appear.

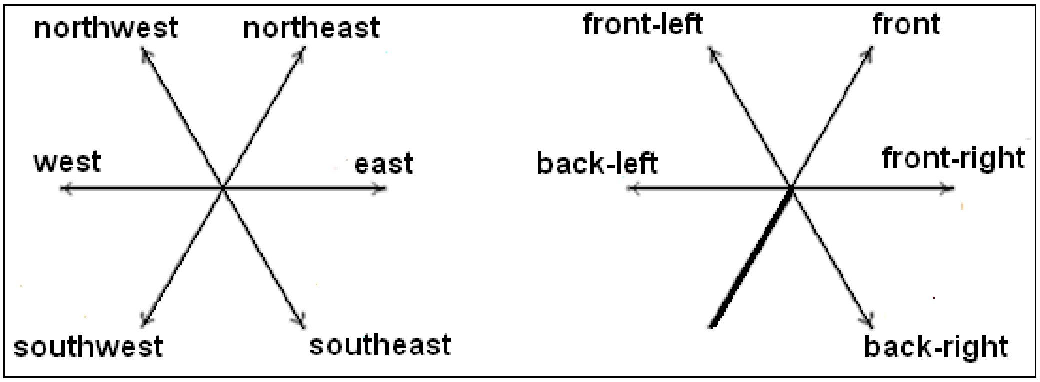

Fig. 2. The elements of absolute and relative code in two-dimensional triangular lattice.

The disadvantage is the complexity of conversion a relative code to coordinate to check self-avoidance: in two-dimensional lattice relative code element depends on the current and previous absolute code elements. (So if the last absolute direction was "east", the next relative direction "front-right" will match the absolute "southeast"). In axonometric case it depends on previous two. To convert the absolute code to relative and vice versa a conversion table is built. Its first two columns are filled with possible absolute directions, the third applies to a desired relative one, and the fourth gives a resulting absolute.

Using the relative encoding a fold is formally represented in the following way:

$$r_1 \in \tilde{R}, r_i \in R, i = \overline{2, n-2}, \;\; R = \mathrm{A} \setminus \{(0,-1,0)\},$$

$$\tilde{R} = \{(0,1,0),\ (1,0,0),\ (1,\text{-}1,0),\ (1,0,\text{-}1)\}.$$

After comparing three encodings two of them (absolute and relative) were chosen for further analysis. Thus COP is defined: the encoding determines a decision space, a set of feasible solutions is the set codes that represent a self-avoiding fold, and the objective function is the energy of fold determined by formula (1).

Deterministic local search

Local search algorithms are a group of algorithms which search optimal values of objective function by iteratively moving from one acceptable solution to another neighboring one, and stop when certain conditions are met (e.g. time or iteration number constraints). The input of the algorithm is some initial value which serves as search starting point. Usually these algorithms are fast and significantly improve initial solution, but in case of multicriterion objective functions they rarely find global optima. These features allow them to be used as a part of more complex algorithms.

Let's examine the deterministic local search algorithm for protein energy minimization in HP model. The distance $\rho(s,v),$ between folds $s = (s_1, s_2, ..., s_k), \;\; v = (v_1, v_2, ..., v_k) \in D$ is determined as the number of differing characters in their encodings:

$$\rho(s,v)=\sum_{i=1}^{k}\chi(s_i,v_i),\quad \chi(x,y)=\begin{cases}1, x=y;\\ 0, x\neq y.\end{cases}$$

Let $O_\delta(s)$ denote a fold s neighborhood of size δ, so $O_\delta(s)=\{v\mid\rho(s,v)\leq\delta\}$. Using such notation the deterministic local search scheme is given in Fig. 3.

```
procedure LocalSearch ( $s_0$ )
  while $O_1(s_0)$ is not totally examined do
    $s$ := some fold from $O_1(s_0)$;
    if $s\in D$ and $E(s)<E(s_0)$ then $s_0 := s$;
  end while;
  return $s_0$ ;
end procedure.
```

Fig. 3. Deterministic local search algorithm scheme.

Thus the algorithm guarantees the improvement of current fold in neighborhood in case it exists.

With the help of deterministic local search the difference between absolute and relative encoding can be demonstrated. As it was mentioned above, a change of single character in absolute encoding leads to parallel shift of the succeeding molecule part, while in relative it results in rotation. Fig. 4 shows the folds produced by the algorithm which was applied to a fold that had a line shape (using relative encoding – on top, and using absolute – on bottom).

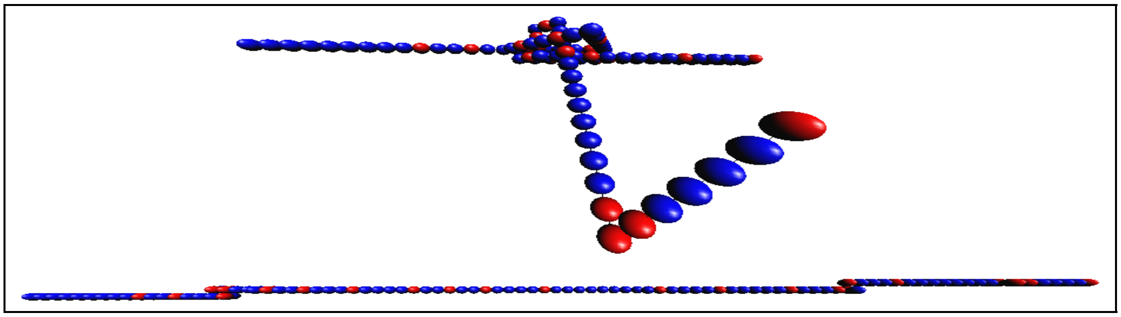

Fig. 4. Deterministic local search results.

The problem is that the condition of self-avoidance is rather strict, so the neighborhood $O_1(s)$ can contain few to none acceptable solutions. Taking this into account it is proposed to replace neighborhood $O_1(s)$ in the procedure LocalSearch by a neighborhood of bigger size. It nonlinearly increases the time of every iteration: the maximum number of elements in $O_1(s)$ is $k*d$, where k denotes the length of encoding and d means the number of characters in it (d =12 for absolute encoding and d =11 for relative one) while $O_2(s)$ may contain $k(k-1)*d^2$ folds. Therefore it is reasonable to reduce $O_2(s)$ to $\tilde{O}_2(s)$ according to the rule

$$v\in\tilde{O}_2(s)\ \Leftrightarrow\ \exists p\in N, 1\leq p\leq k-1:\ v_t=s_t\ \forall t\in\{1,\ldots,k\}\setminus\{p,p+1\},$$

and to use it instead of $O_1(s)$ (maximum number of elements in $\tilde{O}_2(s)$ is $(k-1)*d^2$), although it also contributes greatly to the time. Therefore, to improve performance it is proposed to use stochastic local search methods.

Simulated Annealing Algorithm

Simulated annealing algorithm is one of the widely used local optimization algorithms [Hoos H, Stützle, 2005]. Its idea is quite similar to the one that deterministic local search has, but due to the stochastic component the algorithm may recover from local minima, returning more accurate value. The basic idea is that the move to the best value is executed always, while moves to the "worse" (in terms of objective function values) are accepted with a certain probability that depends on the objective function value and iteration number.

The algorithm is based on simulation of physical process taking place during the crystallization of substances from liquid to solid state, particularly during annealing. It is believed that atoms have taken their stable places in crystal lattice, but transitions of individual atoms from one node to another are still possible. The process proceeds under conditions of gradually lowering temperature. The atom transition from one place to another occurs with some probability that is decreasing with decrease of temperature. Stable crystal lattice corresponds to the minimum of atoms energy, so the atom tends to move to a state with lower energy.

The scheme of the algorithm is shown in Fig. 5.

The temperature varies according to the rule

$$T_t = \frac{H}{\log(\frac{t}{P})},$$

where $H, P \in R$ are the algorithm parameters.

The equilibrium condition is defined as follows. Let ν be some natural number and $\varepsilon > 0$ is real. ν successive iterations of the algorithm are called “run”. If on the current temperature level q runs were made, it is presumed that the equilibrium condition is met if

$$|E_{q+1} - E_i| \leq \varepsilon$$

for some $i \in \{1,...q\}$, where E_i is the average energy value on i-th run.

```
procedure SA($s_0$ )
  $s_{rec} := s_0$; $T :=$ some initial value;
  while $O_1(s_0)$ is not totally examined do
    while equilibrium condition is not met do
          $s :=$ some fold from $O_1(s_0)$;
          if $s \in D$ then
               $\Delta := E(s) - E(s_0)$;
               $p := \min\{1, \frac{\Delta}{T}\}$;
               $\xi :=$ random[0,1];
               if $p \geq \xi$ then
                    $s_0 := s$;
                    if $E(s) < E(s_{rec})$ then $s_{rec} := s$;
               end if;
          end if;
    end while;
    $T :=$ next value;
  end while;
  return $s_{rec}$;
end procedure.
```

Fig. 5. Simulation annealing algorithm scheme

G- algorithm

G-algorithm is an efficient algorithm for solving combinatorial optimization problems [Гуляницкий, 2004]. Unlike the simulation annealing algorithm, the probability of transition to the "worse" value on each step does not depend on the number of iteration, but the bound value that defines dropout conditions for "worse" folds does (Fig. 6).

The function $\varphi(x,y)$ is defined as follows:

$$\varphi(x,y)=1-\frac{E(y)-E(x)}{\gamma\cdot(E(x)-E_{\min})},$$

where $E_{\min}$ denotes the lower bound of protein energy and $\gamma>0$ is a parameter. For evaluating $E_{\min}$ the following property is used: each hydrophobic amino acid residue except the first and the last ones in a triangular lattice can form up to10 contacts, while the first and the last ones can have 11. Therefore, if n_H denotes the number of hydrophobic residues in amino acid sequence, $E_{\min}$ can be calculated by the formula:

$$E_{\min}=-\frac{10n_H+2}{2}=-(5n_H+1).$$

```
procedure G-algorithm ( $s_0$ )
  $s_{rec} := s_0$ ; $\mu_0 := 0$; $h := 0$; $t := 0$;
  while $O_1(s_0)$ is not totally examined do
    while equilibrium condition is not met do
          $s :=$ some fold from $O_1(s_0)$ ;
          if $s \in D$ then
               $\tilde{\varphi} := \varphi(s_0, s)$;
               $p := \min\{1, \max\{0, \tilde{\varphi}\}\}$;
               $\xi := \mu_t + random[0,1]\cdot(1-\mu_t)$ ;
               if $p \geq \xi$ then
                    $s_0 := s$; $h := h+1$ ;
                    if $E(s) < E(s_{rec})$ then $s_{rec} := s$ ;
               end if;
          end if;
    end while;
    $\mu_{t+1} := G(\mu_t)$; $t := t+1$;
  end while;
  return $s_{rec}$ ;
end procedure.
```

Fig. 6. G-algorithm scheme.

Function $G(x)$ was chosen as follows:

$$G(x)=(\eta_h(x)+H)^h,$$

where $0<H<1,\ h\in\{1,2,3\}$ are the algorithm parameters,

$$\eta_k(x)=\begin{cases}0, & x=0,\\ x^{1/h}, & x>0.\end{cases}$$

Equilibrium condition is the same as one used for the simulated annealing algorithm.

Simulated annealing algorithm and G-algorithm in general give more significant improvement of the initial solution than deterministic local search, and the calculation time increases slightly, what makes them suitable for further use.

Ant Colony Optimization Algorithm

Ant colonies optimization algorithm belongs to up-to-date combinatorial optimization stochastic methods and is widely spread [Dorigo M., Stützle, 2004]. The essence of the approach is based on analyzing and using ants' behavior model when they seek shorter paths from colony to food.

The main idea of the algorithm takes as a principle formic colony behavior, such as marking the shorter routes with bigger number of pheromones. In every iteration fixed number of ants simultaneously search their way to the food, using probability based methods of choosing directions that depend on the length of the relevant section of the route and the number of pheromones in it. While traveling some path an ant marks it with the number of pheromones, proportional to its length. To gradually remove nonoptimal plots pheromones evaporation procedure is used. Currently, these methods are quite competitive compared to other metaheuristics and to date they even produce the best results for some problems.

The general algorithm scheme is shown in Fig. 7, where $h, p \in N$ are the algorithm parameters.

The dimension of pheromone matrix for the examined problem is $d \times (n-2)$ (as was mentioned before, d means the number of characters in the encoding, d = 12 for absolute and d = 11 for relative one). In the first step the matrix can be initialized randomly or with certain given value $m_0 \in R^+$. At each iteration the probability follow i-th direction after $(j+1)$-th residue in the structure is proportional to $[i, j]$ -th matrix element to the power α, $\alpha \in [0,1]$ and position estimation $e(pos)$ to the power $1-\alpha$, that is calculated by the following rule:

$$e(pos) = \sum_{p \in O(pos)} e_{\xi_{j+2} f(p)},$$

```
procedure ACO()
  $s_{rec} := null$
  InitPheromoneMatrix (M);
  while stopping condition is not met do
    for $i = 1,...,h$ do
        $s_i$ := GetSomeFeasibleSolution(M);
        if $i \le p$ then $s_i$ := LocalSearch( $s_i$ );
        if $E(s_i) < E(s_{rec})$ then $s_{rec} := s_i$ ;
    end for;
    for $i = 1,...,h$ do   RefreshPheromoneMatrix(M, $s_i$ );
  end while;
  return $s_{rec}$ ;
end procedure.
```

Fig. 7. Ant colony optimization algorithm scheme.

where pos is a lattice node, which the next residue may be positioned in, $O(pos)$ is a set of its neighboring nodes, $f(\cdot) \in \{H, P, 0\}$ is a function that returns H if the node is occupied by hydrophobic residue, P in case of polar and 0 if the node is free. Numbers $e_{HH}, e_{HP}, e_{H0}, e_{PH}, e_{PP}, e_{P0} \ge 0$ are the algorithm parameters and should satisfy following conditions: $e_{HP} \le e_{H0} \le e_{HH}$, $e_{PH} \le e_{P0} \le e_{PP}$.

It is proposed to refresh the pheromone matrix using one of two ways. The first of them is to increase each element proportionally to the absolute value of generated fold according to the formula

$$m_{ij}(t+1)=(1-\rho)m_{ij}(t)+\sum_{l=1}^{h}\chi_{i,j+1}(s_l)(-E(s_l))^q,$$

where $m_{ij}(t)$ is $[i,j]$-th matrix element on the step t, $q\in R^+$ is the algorithm parameter, $\rho\in[0,1)$ defines the rate of pheromones evaporation, $\chi_{i,j}(s)\in\{0,1\}$ is a characteristic function that equals 1 if and only if the encoding of fold s has its j-standing position occupied by symbol i.

The second way is notable for considering separately those minimum parts of molecules, which have contacts inside:

$$m_{ij}(t+1)=(1-\rho)m_{ij}(t)+\sum_{l=1}^{h}\chi_{i,j+1}(s_l)(Z_{j+1}(s_l))^q,$$

where $Z_j(s)$ is a number of contacts in fold s between hydrophobic amino acid residues that have an order not greater than j, and ones with order greater than j

A local search procedure can be selected from three algorithms described above.

Computational Experiment

The efficiency of introduced algorithms was analyzed with the help of specially designed software tools that consist of 3 programs:

• algorithm visualizer is a program that allows to analyze step-by-step the chosen algorithm with the help of charts that demonstrate dependency of energy from iteration number, to customize its input parameters, to observe the energy of output protein folding and it's 3d shape;

• data generator is a software designed for automatic maintenance of database that stores the output of protein structure prediction algorithms;.

• data analyzing module is program designed for visual analysis of algorithms and their comparison one with another or the same ones with different values of parameters.

The computational experiment was conducted using the database of real proteins SwissProt [SwissProt]. For each local search algorithm 20 problems were solved using absolute and relative encoding and fixed input parameter values. Each of them started from three different randomly generated folds and a fold that has a line shape. For each stochastic algorithm for a fixed set of "amino acid sequence-initial folding" 5 runs were conducted. Thus, for each deterministic local search algorithm 80 launches were made, and for simulated annealing and G- algorithm – 400 launches. Using ACO algorithms with fixed set of parameters 5 problems were solved, 3 times each. To compare the effectiveness of the algorithm with different parameter values the additional experiments were carried out.

Folds that were used as the initial approximations for algorithms and the starting seeds of pseudorandom numbers generator are saved in the database to make computational experiment reproducible.

The outcomes of the computational experiment are the following:

Local search algorithms that use absolute encoding in general improve the initial folding better than those, which use relative, except the case it has a shape of line. It can be explained by the fact that varying the single element of relative encoding rotates succeeding part of fold, so it noticeably changes, but at the same time some contacts are saved. The fold that has a line shape is flexible for such actions, but compact folds are not, as their neighborhood in relative encoding in general contains much less feasible solutions.

The increasing of neighborhood radius in deterministic local search algorithms leads to a significant improvement of initial solution, but it nonlinearly increases expended time too. The same is correct for comparing stochastic local search methods with determined local search using single radius and comparing determined local search using extended radius with stochastic local search algorithms. Methods that use a local search procedure for the improvement of somehow generated folds (e.g. genetic algorithms, ACO) should balance using different local search methods to achieve more accurate results during some reasonable time.

On the average solutions generated by G-algorithm are more accurate than those, which were received by simulated annealing with the same time expended.

The efficiency of simulated annealing, G-algorithm and ant colony optimization algorithm considerably relies on parameter values, what confirms the need to develop automatic methods of choosing appropriate parameter values.

The described above features of relative encoding develop into evident advantage in both accuracy and time compared to absolute encoding when using them in ant colony optimization algorithm.

Using position estimation and partial refreshing of pheromone matrix in ant colony optimization algorithms both improve its quality reasonably increasing expended time, whereas the improving of generated folds on every iteration with the help of local search procedure does not result in perfection of derived solution as it was expected. It can be explained by two facts. First of all, using local search for some folds on every step leads to enormous expenditure of time, what considerably decreases the algorithm speed. And the second reason is that significant improvement of generated folds results in premature convergence. From the other side, without local search ant colony optimization algorithm remains uncompetitive, so the open question is varying the parameters and adjusting the algorithm for the use of local search to be effective.

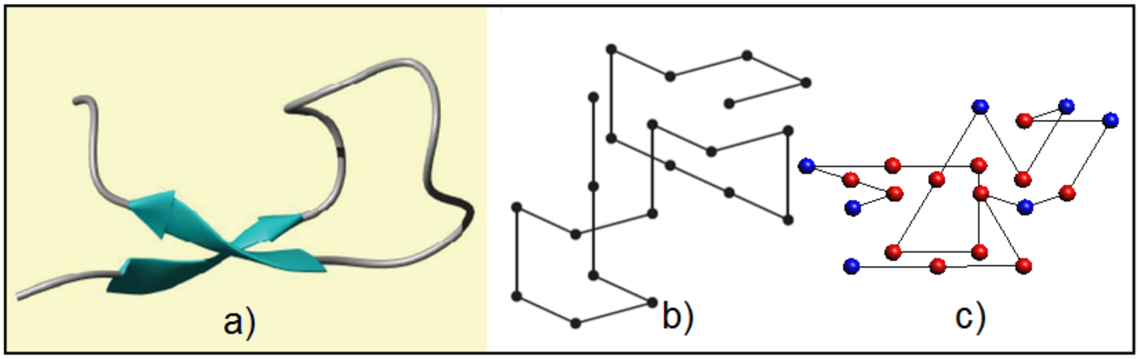

Fig. 8. Hepsidin.

To compare received results with the real ones, the problem of predicting the tertiary structure of Hepsidin molecule (the corresponding amino acid string is HPPHPPPPPHPHPHPHHPPPH) was solved. The results are shown in Fig. 8. The image a) corresponds to Hepsidin real known structure, b) was predicted by ACO method in cubic lattice [Fidanova, Lirkov, 2008], and c) was generated by the proposed ACO method with relative encoding in three-dimensional triangular lattice. The visual comparison on the one hand confirms the adequacy of the HP model and the feasibility of selecting a triangular lattice, on the other it shows the necessity of new sophisticated models' development which more accurately take into account the traits of folding process, and the development of comparison methods to estimate the similarity between the discrete fold and the real one.

Conclusion

The combination of different fold encodings with well-known combinatorial optimization methods results in creating competitive algorithms for solving protein tertiary structure prediction problem. The detailed analysis forms a base for their former use as a part of complex metaheuristic algorithms.

Open questions are the theoretical research of ant colony optimization algorithms convergence and combining them with local search methods, the development of hybrid algorithms for solving the problem, researching new models that describe folding processes in more precise way.

Благодарности

The paper is published with financial support by the project ITHEA XXI of the Institute of Information Theories and Applications FOI ITHEA Bulgaria www.ithea.org, and the Association of Developers and Users of Intelligent Systems ADUIS Ukraine www.aduis.com.ua.

Bibliography

[Белецкий, Васильев, Гупал, 2007] Белецкий Б.А., Васильев С.В., Гупал А.М. Предсказание вторичной структуры белков на основе байесовских процедур распознавания // Проблемы управления и информатики. – 2007. – №2. – С.59-64.

[Гуляницкий, 2004] Гуляницкий Л.Ф. Решение задач комбинаторной оптимизации алгоритмами ускоренного вероятностного моделирования // Компьютерная математика. – 2004. – №1. – С. 64-72.

[Гуляницкий, Рудык, 2009] Гуляницкий Л.Ф., Рудык В.А. Разработка и исследование алгоритмов решения задачи прогнозирования третичной структуры протеина // XVth International Conference «Knowledge-Dialogue-Solution» KDS 2009 – С. 104-112.

[Гуляницкий, Рудык, 2010] Гуляницкий Л.Ф., Рудык В.А. Моделирование свертывания протеина в пространстве // Компьютерная математика. – 2010. – №1. – С. 128-138.

[Финкельштейн, Птицин, 2005] Финкельштейн А.В., Птицин О.Б. Физика белка: Курс лекций с цветными и стереоскопическими иллюстрациями и задачами. – 3-е изд., испр. и доп. – М.: КДУ, 2005. – 456 с.

[Awadh, Bahamish, Abdullah, Abdul Salam, 2006] Awadh H. Bahamish A., Abdullah R., Abdul Salam R. Protein conformational search using honey bee colony optimization // Proceedings of the 2nd IMT-GT Regional Conference on Mathematics, Statistics and Applications. – 2006.

[Berger, Leighton, 1998] Berger B., Leighton T. Protein folding in the hydrophobic-hydrophilic (HP) model is NP-complete // Journal of Computational Biology. – 1998. – № 5(1). – P. 27-40.

[Crescenzi, Goldman, Papadimitriou, Piccolboni, Yannakakis, 1998] Crescenzi P., Goldman D., Papadimitriou C., Piccolboni A., Yannakakis M. On the complexity of protein folding // Journal of Computational Biology. – 1998. – № 5(3). – P. 423-465.

[Decatur, 1996] Decatur S.E. Protein folding in the generalized hydrophobic-polar model on the triangular lattice. MIT LCS Technical Memo: MIT-LCS-TM-559. – 1996.

[Dill, Bromberg, Yue, Fiebig, Yee, Thomas, Chan, 1995] Dill K., Bromberg S., Yue K., Fiebig K.M., Yee D., Thomas P., Chan H. Principles of protein folding – a perspective from simple exact models // Protein Science. – 1995. – № 4. – P. 561-602.

[Dorigo M., Stützle, 2004] Dorigo M., Stützle T. Ant Colony Optimization. – Cambridge: MIT Press, MA, 2004. – 348 p.

[Fidanova, Lirkov, 2008] Fidanova S., Lirkov I. Ant colony system approach for protein folding // Proceedings of the International Multiconference on Computer Science and Information Technology. – 2008. – P. 887-891.

[Hoos H, Stützle, 2005] Hoos H., Stützle T. Stochastic Local Search: Foundations and Applications. – San Francisco: Morgan Kaufmann Publ., 2005. – 658 p.

[Paluszewski, Winter, 2008] Paluszewski M., Winter P. EBBA: Efficient Branch and Bound Algorithm for Protein Decoy Generation // DIKU, Technical Report. – 2008.

[SwissProt] SWISS_PROT Protein Sequence Data Bank: http://www.expasy.ch/sprot/

Authors' Information

***Leonid Hulianytskyi** – Dr.Sc.(Technology), Leading Research Scientist of V.M.Glushkov Institute of Cybernetics of National Academy of Sciences of Ukraine; Prof. of NTUU "KPI"; e-mail: lh_dar@hotmail.com*

Major Fields of Scientific Research: Combinatorial optimization, Decisions Making, Mathematical modeling and practical applications

***Vitalina Rudyk** – Master of Science in Applied Mathematics at Taras Shevchenko National University of Kyiv, Ukraine; e-mail: vitalina.rudyk@gmail.com*

Major Fields of Scientific Research: Combinatorial optimization and practical applications, Computational biology, Protein folding