Incorporating Knowledge of Secondary Structures in a L-System-Based Encoding for Protein Folding

Gabriela Ochoa[1], Gabi Escuela[1], and Natalio Krasnogor[2]

[1] Department of Computer Science, Universidad Simon Bolivar,
Po. Box 89000, Caracas 1080-A, Venezuela
gabro@ldc.usb.ve, gabiescuela@netuno.net.ve
[2] School of Computer Science and I.T., University of Nottingham,
NG81BB, Nottingham, UK
Natalio.Krasnogor@nottingham.ac.uk

Abstract. An encoding scheme for protein folding on lattice models, inspired by parametric L-systems, was proposed. The encoding incorporates problem domain knowledge in the form of predesigned production rules that capture commonly known secondary structures: α-helices and β-sheets. The ability of this encoding to capture protein native conformations was tested using an evolutionary algorithm as the inference procedure for discovering L-systems. Results confirmed the suitability of the proposed representation. It appears that the occurrence of motifs and sub-structures is an important component in protein folding, and these sub-structures may be captured by a grammar-based encoding. This line of research suggests novel and compact encoding schemes for protein folding that may have practical implications in solving meaningful problems in biotechnology such as structure prediction and protein folding.

1 Introduction

Proteins are complex organic compounds made up of amino acids joined by peptide bonds[1]; they are essential to the structure and function of all living beings; and are amongst the most studied molecules in biochemistry. Proteins fold naturally into unique 3-dimensional structures, known as their native state or *tertiary structure*. The biological role of a protein will depend on this 3D conformation which in turn is determined by its amino acid sequence (also known as *primary structure*). Biochemists also distinguish *secondary structures* which are highly patterned sub-structures – mainly α-helices and β-sheets – that are locally defined, so there can be many secondary motifs present in a single protein (see Figure 1).

[1] An amino acid is any molecule that contains both amino and carboxylic acid functional groups. A peptide bond is a chemical bond formed between two molecules when the carboxyl group of one molecule reacts with the amino group of the other molecule, releasing a molecule of water.

E. Talbi et al. (Eds.): EA 2005, LNCS 3871, pp. 247–258, 2006.

Fig. 1. A representation of the 3D structure of myoglobin, showing shaded α-helices

Genome projects are producing vast amounts of amino acid sequences, but understanding the biological role of these proteins will require knowledge of their structure. The problem of predicting the 3D conformation of a protein from its linear sequence, is known as the protein structure prediction problem (PSP). Although, biochemists use empirical techniques (e.g. magnetic resonance, and X-ray crystallography) on protein crystals in order to infer their conformations, these methods are costly and time consuming. Computational structure prediction methods will provide valuable information for the large amount of sequences whose structures will not be determined experimentally. Two classes of computational methods for the PSP are distinguished [1]. The first (e.g. threading and comparative modelling) rely on detectable similarities between the modelled sequence and known structures. The second class of methods, *de novo* or *ab initio* methods, predict the structure from sequence alone, without relying on similarity at the fold level between the modelled sequence and any of the known structures. Several heuristic search methods (e.g. monte-carlo methods, simulated annealing, and evolutionary algorithms) have been applied for de novo structure prediction [20, 4, 16, 11]. However, PSP is still an open problem and large instances are difficult to solve. A possible cause hindering the scaling of these techniques, are the current direct encodings used (see section 2.1). In the context of EAs applied to design, it has been argued that a *generative* or rule-based scheme, that specifies how to construct the phenotype, as opposed to a direct encoding of the phenotype; can achieve greater scalability through self-similar and hierarchical structures [7, 2, 8]. Moreover, a generative encoding would be a more compact representation of a solution. A first approach to a generative encoding for the PSP was presented in [6], where non parametric L-systems were evolved to capture protein tertiary structures. This approach although promising, met only partial success since the search process was slow and required many executions of the algorithm to obtain a successful L-system. Here we improved those results with two extensions: first, we consider parametric L-systems, and secondly, we incorporate knowledge about secondary structures in the form of predefined rules.

Previous work on evolving L-systems both for capturing blood vessels on the eye [9], and the growth process of trees [3], have had to rely on specific knowledge of the problem domain in order to enhance the algorithms' performance. This knowledge was, in both cases, incorporated in the form of predesigned

fixed rules. On the other hand, previous studies on protein folding simulations with evolutionary algorithms, have used knowledge about secondary structure in order to improve the algorithms' behavior on large instances [13, 16, 10, 15]. Thus, the evidence gathered from previous research suggests that incorporating domain knowledge in the form of predesigned rules that capture secondary substructures, would enhance performance. In this paper we test this hypothesis and compare results with the non parametric L-systems (without knowledge about secondary structures) suggested in [6].

The next section will describe the model of folding used for the experiments in this paper, the HP model. This model although simple still captures the essential properties of protein folding. The currently used encodings for heuristic approaches to PSP in the HP model (which are all direct encodings), are also described. Thereafter, section 3 describes the formalism of L-systems. The proposed encoding, and evolutionary algorithm used are described in section 4. Section 5 presents and discusses our results, and finally section 6 offers our conclusions and hints for future work.

2 The HP Model

A major driving force in determining the tertiary structure of proteins is the hydrophobic effect. The idea behind this effect is that, energetically, protein folding is driven by two factors: *hydrophobic*(or "oily) groups "prefer" to "get away" from water, and *hydrophilic* (or polar) groups "prefer" to "dissolve" in water. Thus, the polypeptide chain folds such that the nonpolar amino acids are "hidden" within the structure and the polar residues are exposed on the outer surface. The HP model [5] captures this idea: only two types of monomers are distinguished: hydrophobic (H), and polar or hydrophilic (P). The set of valid protein structure conformations is the space of all self-avoiding paths (on a selected lattice, e.g., square 2D, triangular, cubic, diamond, etc.), with each amino acid located on a lattice bead. Hydrophobic units that are adjacent in the lattice but non-adjacent in the sequence (also called non-local H-H contacts) add a constant negative factor (generally =-1) and all other interactions are ignored. The native state is thought to be the global energy minimum.

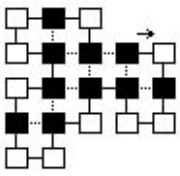

Fig. 2. Native structure in the square 2D lattice for the primary sequence $HPHPPHHPHPPHPHHPPHPH$. White boxes corresponds to H, and black to P amino acids. The arrow indicates the starting point, and the dotted lines the non-local H-H contacts.

2.1 Problem Encoding

In the HP model, the structures can be represented by Cartesian coordinates, internal coordinates or distance geometry. We consider here internal coordinates, which can be absolute or relative. Under the absolute encoding, the structures are represented by a list of absolute moves. In a 2D square lattice, for example, a structure is encoded as a string in the alphabet $\{\mathbf{U}p, \mathbf{D}own, \mathbf{L}eft, \mathbf{R}ight\}$. When using relative coordinates, each move is interpreted in terms of the previous one, like in LOGO turtle graphics; a structure is encoded as a string in the alphabet $\{\mathbf{F}orward, Turn\mathbf{L}eft, Turn\mathbf{R}ight\}$. Figure 2, shows the optimal folding of an example protein, the structure is coded either as $RDDLULDLDLUURULURRD$ (absolute encoding) or $RFRRLLRLRRFRLLRRFR$ (relative encoding). The number of non-local $H-H$ contacts is nine. That is, the folding energy is -9.

3 L-Systems

L-systems are a mathematical formalism proposed by the biologist Aristid Lindenmayer in 1968 as an axiomatic theory of biological development. More recently, L-systems have found several applications in computer graphics [19, 18]. Two principal areas include generation of fractals and realistic modelling of plants. Central to L-systems, is the notion of rewriting, where the basic idea is to define complex objects by successively replacing parts of a simple object using a set of rewriting rules or productions. The rewriting can be carried out recursively.

The essential difference between traditional formal language grammars and L-systems lies in the method of applying productions. In formal languages productions are applied sequentially, whereas in L-systems they are applied in parallel, replacing simultaneously all letters in a given word. This difference reflects the biological motivation of L-systems. Productions are intended to capture cell divisions in multicellular organisms, where many division may occur at the same time.

A formal definition of L-systems is as follows [18]: Let V denote an alphabet, V^* the set of all words over V, and V^+ the set of all nonempty words over V. A $L{-}system$ is an ordered triplet $G = \langle V, \omega, P \rangle$, where V is the *alphabet*, $\omega \in V^+$ is a nonempty word called the *axiom* and $P \subset V \times V^*$ is a finite set of *productions*. If a pair (a, χ) is a production, we write $a \rightarrow \chi$. The letter a and the word χ are called the *predecessor* and *sucessor* of this production respectively. It is assumed that for any letter $a \in V$, there is at least one word $\chi \in V^*$ such that $a \rightarrow \chi$. If no production is explicitly specified for a given predecessor $a \in V$, we assume that the *identity production* $a \rightarrow a$ belongs to the set of productions P.

The derivation process of an L-system can be formally stated as follows: Let $\mu = a_1 \dots a_m$ be an arbitrary word over V. We will say that the word $\nu = \chi_1 \dots \chi_n \in V^*$ is *directly derived* from (or *generated* by) μ, and write $\mu \Rightarrow \nu$, if and only if $a_i \rightarrow \chi_i$ for all $i = 1, \dots, m$. A word ν is generated by G in a derivation of *length* n if there exists a *developmental sequence* of words $\mu_0, \mu_1, \dots, \mu_n$ such that $\mu_0 = \omega, \mu_n = \nu$ and $\mu_0 \Rightarrow \mu_1 \Rightarrow \dots \mu_n$.

L-systems can be classified into context-free and context sensitive, according to whether production rules refer only to an individual symbol, or to a particular symbol only if it has certain neighborhood. L-systems can be also be deterministic or non-deterministic, according to whether there is exactly one production for each symbol, or there are several, and each is chosen with a certain probability during each iteration. Finally, L-systems can be parametric if there are numerical parameters associated with the symbols or productions.

4 Method

4.1 The Proposed Encoding: PFL-System

The encoding proposed is a simplified parametric, context-free L-system. The alphabet will depend on the lattice and coordinate system used. For the experiments reported here, we selected the square 2D lattice with relative coordinates. Thus, the terminal symbols are $\{F, L, R\}$. Two non-terminal symbols: A and H are included, they represent the predecessors of two predefined rules that capture secondary structures. Thus, the l-system's alphabet is $V = \{F, R, L, A, H\}$ (see Table 1). The axiom ω is a nonempty word in V^+, each symbol in the axiom has a parameter associated that determines the number of times it is repeated. The maximum for these repetition values are displayed in Table 1. These values were selected empirically for the set of (relatively short) instances studied in this paper, they are likely to depend on the instances length and complexity. The two prefixed rules are $A = RRLL$ and $H = LLRR$, and represent a single coil of a right-oriented and a left-oriented α-helix respectively (Figure 3). The secondary structure known as β-sheet is represented in the 2D Square HP model as a strings of Fs, so this substructure is also easily captured by the proposed encoding (symbol F with a parameter $n > 1$) (Figure 3). We termed our encoding PFL-system, where P stands for parametric, and F for fixed rules.

Table 1. L-system's symbols and their interpretation

Command	Description	Max. n	Symbol
forward(n)	move forward n times	4	F
right(n)	move right n times	2	R
left(n)	move left n times	2	L
right helix(n)	right helix n times	2	A
left helix(n)	left helix n times	2	H

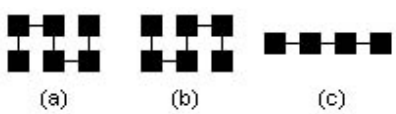

Fig. 3. Secondary structures in the 2D HP model: (a) right-oriented α-helix, $A = RRLL$; (b)left-oriented α-helix $H = LLRR$, (c) β-sheet , $F^n, n\,2$

4.2 Evolutionary Algorithm

In order to test whether the proposed encoding can capture a target folding in the 2D HP model, we used an EA as the inference procedure for exploring the L-system's space. Given a target structure in direct encoding (internal relative coordinates) the EA will evolve a generative encoding (L-system) that, once derived, would match closely the original target. The EA implemented was generational with linear ranking selection and elitism. As the variation operators, a recombination and three mutation operators were implemented. A mate selection strategy [17] (dissasortative mating) was also implemented as a mechanism for increasing the population genetic diversity. Dissasortative mating was implemented as follows: when selecting two individuals for a crossover, the first parent was selected as usual. To chose the second parent, a set of s (scan size) individuals were selected using the GA fitness-based selection method. Thereafter, the similarity between each of these s phenotypes and the first parent was computed, the phenotype with less similarity was chosen. For the experiments reported here, Hamming distance was used as the similarity measure, and the scan size s was set to 5. Two stopping criteria were considered: (i) if an individual arises with the maximum fitness, that is, its L-system grammar exactly represents the target folding; or (ii) a preset maximum number of generations is reached. The initial population, genetic operators, and fitness evaluation are described below.

Initialization. L-systems has two predefined production rules: A and H, and the axiom is a variable length word $\omega \in V^+$. A new individual is created by generating a random axiom of 5 to l symbols; where l is slightly larger (about 5%) than the string length of the target folding. In producing the axioms, the probability of generating a terminal symbol $\{F, L, R\}$ is 0.95 whilst that of generating a non-terminal symbol $\{A, H\}$ is 0.05. These values were empirically selected and more exhaustive studies should be performed, since the algorithm behavior was found to be sensitive to these probabilities. Moreover, the most effective settings are likely to depend on the particular instance under study.

Mutation. Three mutation operators were implemented: (i) addition, (ii) deletion, and (iii) modification of a single symbol in the axiom of an individual. The modification operator may alter either a symbol or its associated parameter. When a mutation is to be performed, 60% of times it will be a modification, 30% an addition, and 10% deletion.

Recombination. Recombination takes two individuals, $p1$ and $p2$ as parents and creates two offspring $o1$ and $o2$. Recall that individual's axioms are of variable length; a single cross point is randomly selected considering the length of the shorter axiom (lets consider it to be $p1$). $o1$ is of the same length as $p1$ and inherits from it the left sub-sequence (before the cross point); and the right sub-sequence from $p2$. $o2$ is of the same length as $p2$, and inherits from it the sub-string before the cross point, then it inherits from $p1$ all the symbols after the cross point, finally any remaining symbols to complete the length of $o2$ are taken from $p2$. Thus the proposed crossover has reminiscences with both 1-point

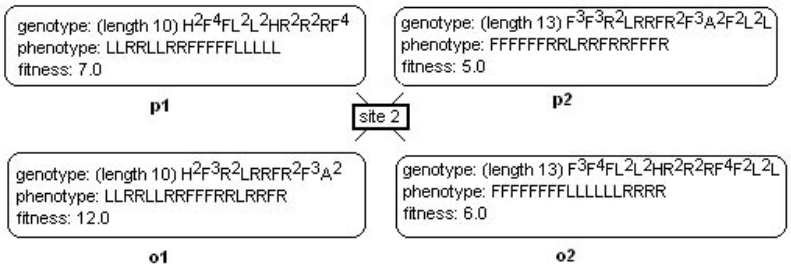

Fig. 4. Example of a crossover operators between two individuals of different lengths

and 2-point crossover. It will be a 1-point crossover if the two parents have the same length. An example of this operator is detailed in Figure 4.

Derivation and Fitness Function. For computing an individual's fitness, its L-system is derived. Phenotypes are directly derived from the axiom, that is, a single derivation step suffices for producing the phenotype from the genotype since the production rules are fixed, and contain only terminal symbols. The derived string will be truncated as soon as the length of the target folding is reached. This means that the rightmost part of the axiom may be discarded. The fitness value will be the number of matches between the produced phenotype and the target folding, that is a generalized Hamming distance. So, the minimum fitness is 0 and the maximum is the length of the desired folding.

5 Experiments and Results

Two sets of experiments were carried out. The first set compared the performance of the newly proposed encoding against the D0L-system implemented in [6]. The same group of proteins instances (see Table 3), and similar EA parameter settings (see Table 2) were employed for the sake of comparison. These four instances are available at http://www.cs.nott.ac.uk/˜nxk/hppdb.html; and their foldings were obtained using MAFRA (Memetic Algorithm FRAmework) [14]. Notice that these foldings are not necessarily optimal, but are close to the optimal solution.

The number of successes (runs that produced the target folding exactly) out of 50 runs, is shown for each encoding (Table 4). Also a summary of the secondary

Table 2. Parameter values used for the experiments

Parameter	Value
Max. Number of Generations	2000
Population Size	50
Mutation Rate	0.05
Recombination Rate	1.0
Mating Strategy	Disassortative (5)

Table 3. Benchmark protein instances for the 2D HP model. L stands for the folding length, which is also the maximum attainable fitness of our EA approach.

Name	Protein Sequence	Target Folding	L
Ins18a	HPHPPHHPHPPHPHHPPHPH	RFRRLLRLRRFRLLRRFR	18
Ins18b	HHHPPHPHPHPPHPHPHPPH	RRFRFRLFRRFLRLRFRR	18
Ins22	HHPPHPPHPPHPPHPPHPPHPPHH	RLLFLFFRRFLLFRRLRFFRRF	22
Ins23	PPHPPHHPPPPHHPPPPHHPPPPHH	FFRRFFFLLFFFFRRFFFFLLFF	23

Table 4. Comparing No. of successful runs (runs that produced the target folding exactly) using both D0L-systems as proposed in [6], and the parametric L-system with fixed rules (PFL-system) proposed here

Instance	Secondary Structures	D0L-system	PFL-system
Ins18a	A, H	5/50	14/50
Ins18b	None	3/50	1/50
Ins22	None	1/50	1/50
Ins23	F^3, F^4, F^4	1/50	49/50

structures found by the algorithm is included. Notice that for the instances where secondary structures were present (Ins18a and Ins23), the new encoding (PFL-system) produced a significant higher rate of success. Whereas for the other two instances where there were not α-helices or β-sheets, the performance was comparable with that of the previously proposed encoding.

In order to have a dynamic view of the two encodings' performance, the best fitness (averaged over 50 runs) was plotted for each generation on a selected instance (Ins18a) (see Figure 5, Left). The best performance over the whole run is clearly produced by the parametric L-system with fixed rules (PFL-system).

The encoding proposed in [6], was unable to capture the folding of instances longer than twenty or so amino acids. For example, for an instance of length 34

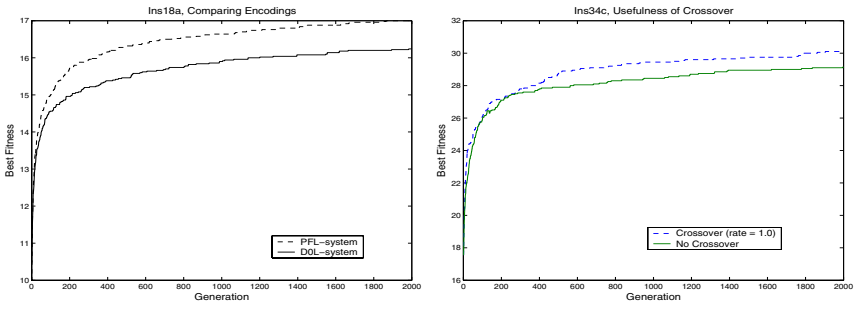

Fig. 5. Best-performance-trace curves. **Left**: Ins18a with two encodings; D0L-systems as proposed in [6], and the parametric L-system with fixed rules (PFL-system) proposed here. **Right**: Ins34c with and without recombination. The curves show the average of 50 (Ins18a) and 20 (Ins34c) runs.

Table 5. Benchmark protein instances of length 34. L stands for the folding length, which is also the maximum attainable fitness of our EA approach.

Name	Protein Sequence	Target Folding	L
Ins34a	PPPHHPPHHPPPPPPHHHHHHHPPHHPPPPHHPPHPP	FFLFRRLLRRFFFRRFLLFRLLFRFFLFLLRFRR	34
Ins34b	HPHHHHPHHPPHPHHHHPHPPHHPPHHPPHHHHHHH	RRFRLLRRLLFLFRRFLFLLRRLLRRLLFLFRRF	34
Ins34c	HHHHHHHPHPHHPHPPHHPPHHPPHPHHPHPPHPH	LLFRLRRFRLLRFRRLLRRLLRRFRLLRFRRLLR	34

Table 6. Results for the benchmark instances of length 34 with and without recombination. The frequencies of obtained maximum fitness values are shown. The maximum possible fitness is 34 (exact match).

Name	Crossover	Frequencies of Obtained Fitness Values	Average	Secondary Structures
Ins34a	On	27:1, 28:3, 29:6, 30:5, 31:4, 32:0, 33:1	29.6	
Ins34a	Off	28:4, 29:5, 30:7, 31:4	29.55	A, F^3
Ins34b	On	29:4, 30:5,31:9,32:2	30.45	
Ins34b	Off	28:2, 29:4, 30:6, 31:4, 32:3, 33:1	30.25	H, A, A
Ins34c	On	28:2, 29:4, 30:6, 31:7, 32:0, 33:1,	30.1	
Ins34c	Off	27:1, 28:5, 29:9, 30:1 ,31:3, 32:1	29.15	H^2, A^2

(Ins34a), the best fitness statistics obtained after 20 runs were: average = 24.05, best run = 27.0, worst run = 19.0. In order to asses whether the newly proposed encoding had better scaling properties; a second group of experiments explored three instances of length 34 (see Table 5). Ins34a was obtained from the same source than the shorter instances described above, whereas Ins34b and c, and their foldings were taken from [21]. The parameter settings for these experiments are the same as before (see Table 2), but the number of replicas were 20 instead of 50 Furthermore, runs with and without crossover were carried out in order to asses the usefulness of this operator in this context.

Results suggest that the PFL-system encoding has better scaling properties than the previous D0L-system. Although the perfect match (34) was not found in any run, most runs arrive very close to the solution. Table 6, shows the frequencies of obtained maximum fitness for each instance with and without crossover. Crossover seems to be helpful to the evolutionary search although

Table 7. Best obtained individual, represented in PFL-system encoding, for each of the benchmark instances studied

Instance	Best Solution (PFL-system Encoding)	Encoding Length
Ins18a	$RFARLR^2FRHFR$	11
Ins18b	$R^2FRFRLFR^2FLRLRFR^2$	15
Ins22	$RL^2FLF^2R^2FL^2FR^2LRF^2R^2F$	15
Ins23	$F^2R^2F^3L^2F^4R^2F^4L^2F^2$	9
Ins34a	$F^2LFAR^2F^3R^2FL^2FRL^2FRF^2LFL^2RFR^2$	21
Ins34b	$R^2FRHL^2FLFR^2FLFL^2A^2FLFR^2F$	19
Ins34c	$L^2F^2R^2RFRL^2RFR^2H^2FRLLRFA^2$	18

the differences are not substantial. Some instances seems to benefit more from crossover than others.In order to have a dynamic view of the algorithm behavior with and without recombination, Figure 5 (Right) shows best-performance curves over the whole run for Inst34c. Clearly the recombinant GA outperform the GA with mutation only.

As a summary of results, the best solutions obtained with the PFL-system, for all the instances studied, are shown in Table 7. Notice that the inclusion of parameters helps in having a more compact representation of a folding, also the occurrence of secondary structures is captured and easily identified with this representation.

6 Discussion

An encoding scheme for protein folding in the HP model, inspired by parametric L-systems, was proposed. The encoding also incorporates problem domain knowledge in the form of predesigned production rules that capture the most commonly known secondary structures: α-helices and β-sheets. The ability of this encoding to capture protein native conformations was tested using an EA as the inference procedure for discovering L-systems. Given a target folding, the EA explores the space of possible L-systems (genotypes) until identifying one whose derivation (phenotype) closely matches the target folding.

This newly proposed encoding was found to improve our first attempt of using L-systems as a generative representation for protein folding [6], where problem domain knowledge was not incorporated. The suitability of the new encoding, however, seems to heavily depend on the particular instance under study. Instances with high frequencies of α-helices and β-sheets, would have a clear advantage. Longer proteins and 3D lattices should be addressed. Furthermore, two somehow opposite but complementary extensions could be suggested. First, incorporating other known secondary structures such as β-turns, β-hairpins, etc., as prefixed rules. Secondly, enabling the EA to discover their own production rules, that could be in principle stored and thereafter used in further runs with new instances. We have evidence on a related bioinformatic problem [12] that enabling the evolutionary algorithm to systematically and vigorously discover new "building blocks" (as the ones we described in this paper) can substantially improve the algorithm performance.

Finally, we believe that this proposed line of research opens up the possibilities for novel and compact encoding schemes of protein structures, that have potential implications in solving meaningful biotechnology problems such as structure prediction and protein folding.

Acknowledgements

Natalio Krasnogor acknowledges EPSRC (GR/T07534/01, EP/D021847/1) and BBSRC (BB/C511764/1) for funding his research on protein structure prediction, comparison and self-assembly

References

[1] David Baker and Andrej Sali, *Protein structure prediction and structural genomics*, Science **294** (2001), 93–96.

[2] Peter J. Bentley, *Exploring component-based representations - the secret of creativity by evolution?*, Fourth International Conference on Adaptive Computing in Design and Manufacture (ACDM 2000) (I. C. Parmee, ed.), 2000, pp. 161–172.

[3] Luis DaCosta and Jacques-Andre Landry, *Generating grammatical plant models with genetic algorithms*, Proceedings of the 7th International Conference on Adaptive and Natural ComputiNG Algorithms (ICANNGA, LNCS, Springer Verlag, 2005.

[4] T. Dandekar and P. Argos, *Folding the main chain of small proteins with the genetic algorithm*, J. Mol. Biol **236** (1994), 844–861.

[5] Ken A. Dill, *Theory for the folding and stability of globular proteins*, Biochemistry **24** (1985), 1501.

[6] Gabi Escuela, Gabriela Ochoa, and Natalio Krasnogor, *Evolving L-systems to capture protein structure native conformations*, Proceedings of the 8th European Conference on Genetic Programming, Lecture Notes in Computer Science, vol. 3447, Springer, 2005, pp. 74–84.

[7] L. J. Fogel, P. J. Angeline, and T. Bäck (eds.), *Shape representations and evolution schemes*, MIT Press, 1996.

[8] Gregory S. Hornby and Jordan B. Pollack, *The advantages of generative grammatical encodings for physical design*, Proceedings of the 2001 Congress on Evolutionary Computation CEC2001, IEEE Press, 2001, pp. 600–607.

[9] G. Kókai, Z. Tóth, and R. Ványi, *Modelling blood vessels of the eye with parametric L-systems using evolutionary algorithms*, Proceedings of the Joint European Conference on Artificial Intellingence in Medicine and Medical Decision Making (AIMDM-99) (Berlin), LNAI, vol. 1620, Springer, 1999, pp. 433–442.

[10] Natalio Kranogor, *Studies on the theory and design space of memetic algorithms*, Ph.D. thesis, University of the West of England, Bristol, UK, 2002.

[11] N. Krasnogor, B. P. Blackburne, E. K. Burke, and J. D. Hirst, *Multimeme algorithms for protein structure prediction*, Lecture Notes in Computer Science **2439** (2002), 769–779.

[12] Natalio Krasnogor and Stephen Gustafson, *The local searcher as a supplier of building blocks in self-generating*, Workshop Proceedings of the 2003 Genetic and Evolutionary Computation Conference, GECCO 2003, 2003.

[13] Natalio Krasnogor, D. Pelta, P.E. Martinez-Lopez, P. Mocciola, and E. de la Canal, *Enhanced evolutionary search of foldings using parsed proteins*, Proceedings of the Argentinian Operational Research Simposium (S.I.O. 97), 1997.

[14] Natalio Krasnogor and Jim Smith, *MAFRA: A java memetic algorithms framework*, Data Mining with Evolutionary Algorithms (Alex A. Freitas, William Hart, Natalio Krasnogor, and Jim Smith, eds.), 2000, pp. 125–131.

[15] Neal Lesh, Michael Mitzenmacher, and Sue Whitesides, *A complete and effective move set for simplified protein folding*, Proceedings 7h Annual International Conference on Research in Computational Molecular Biology (RECMB), 2003.

[16] F. Liang and W. Wong, *Evolutionary monte carlo for protein folding simulations*, Journal of Chemical Physics **115** (2001), no. 7, 3374–3380.

[17] Gabriela Ochoa, C. Mädler-Kron, R. Rodriguez, and K. Jaffe, *Assortative mating in genetic algorithms for dynamic problems*, Applications of Evolutionary Computing, EvoWorkshops2005, LNCS, vol. 3449, Springer Verlag, 2005, pp. 605–610.

[18] P. Prusinkiewicz and A. Lindenmayer, *The algorithmic beauty of plants*, Springer, New York, 1990.
[19] Alvy R. Smith, *Plants, fractals, and formal languages*, Computer Graphics **18** (1984), no. 3, 1–10.
[20] I. Unger and J. Moult, *Genetic algorithms for protein folding simulations*, Journal of Molecular Biology **1** (1993), no. 231, 75–81.
[21] Berrin Yanikoglu and Burak Erman, *Minimum energy configurations of the 2-dimensional hp-model of proteins by self-organizing networks*, Journal of Computational Biology **9** (2002), no. 4, 613–620.