

Evolved Cellular Automata for Protein Secondary Structure Prediction Imitate the Determinants for Folding Observed in Nature

Paras Chopra^{a,*} and Andreas Bender^b

^a*Delhi College of Engineering, Bawana Road, New Delhi-110042, India*

E-mail: paras.chopra@bt.dce.edu; URL: <http://www.paraschopra.com>

^b*Novartis Institutes for BioMedical Research, Lead Discovery Informatics, 250 Massachusetts Avenue, Cambridge/MA, USA*

E-mail: mail@andreasbender.de; URL: <http://www.andreasbender.de/>

Edited by E. Wingender; received 12 September 2006; revised 7 November and 10 December 2006; accepted 10 December 2006; published 11 December 2006

ABSTRACT: We demonstrate the first application of cellular automata to the secondary structure predictions of proteins. Cellular automata use localized interactions to simulate global phenomena, which resembles the protein folding problem where individual residues interact locally to define the global protein conformation. The protein's amino acid sequence was input into the cellular automaton and rules for updating states were evolved using a genetic algorithm. An optimized accuracy (Q3) for the RS126 and CB513 dataset of 58.21% and 56.51%, respectively, could be obtained. Thus, the current work demonstrates the applicability of a rather simple algorithm on a problem as complex as protein secondary structure prediction.

KEYWORDS: Protein secondary structure prediction, protein folding, evolved cellular automata, genetic algorithms, secondary structure, protein fold, protein structure

INTRODUCTION

Protein structure prediction is one of the most important problems in molecular biology. This problem demands an immediate and effective solution due to the fact that the protein sequences are being determined at a much faster rate than their respective 3D structures. One step towards cracking this problem is to predict the secondary structure, which in turn greatly assists the prediction of the final 3D structure [1].

Tremendous work has already been done in this area of secondary structure prediction. This paper presents another attempt in this direction, though with the emphasis on the *simplicity* (and, in turn, interpretability) of the method employed. Here, we employ a cellular automaton to predict the secondary structure of a sequence and use a Genetic Algorithm to optimize the parameters (rules) involved in the method.

*Corresponding author.

Rule no. 110

neighborhood of central cell	000	001	010	011	100	101	110	111
new state for central cell	0	1	1	1	0	1	1	0

Example Simulation

Iteration	Lattice
$t=0$	10100110010
$t=1$	11101110111
$t=2$	00111011010

Fig. 1. The cellular automaton shown here is based on the rule no. 110 given by Wolfram [3]. At each time-step (iteration), rules are applied to each of the cells and their new states are calculated according to the specified rules. The extremes of the lattice are connected, thus making the surface continuous.

Cellular Automata (CA) are an ideally decentralized and massively parallel computation system. It involves an input where each unit, or “cell”, can have n different states. Next, a neighborhood is defined such that each cell can only interact with the cells in its vicinity. How the cell interacts with its neighbors depend on the rules defined for the particular automaton. This interaction happens for each cell in the automaton at each time instant.

In most implementations, these interactions are calculated at discrete time steps or iterations. The state of a cell in next iteration depends upon the rules involving the current state of the cell and the cells belonging to the defined neighborhood. These rules decide how the cellular automata will behave. A well known example of one of such rules is popular Game of Life [2] and an example simulation is given in Fig. 1.

Figuring out the rule table is the most important pre-requisite for designing a successful cellular automaton which carries out a defined calculation. The number of possible rules is generally enormous; therefore testing out all rules is infeasible. Rules crafted by humans also tend to be imperfect, particularly when the task carried out by cellular automata is complex. To search through this exhaustive list of possible rules, Genetic Algorithms (GA) [4] seem to be quite suitable, since they are able to narrow down the hypothesis space to be searched while still in many cases being able to identify global function optima.

While protein folding is probably more determined by short-range interactions than by long-range ones, the particular resulting secondary structure is nonetheless a property of the whole sequence, thus requiring global coordination within the cellular automata. While it is not immediately obvious, due to their local nature, that CAs are able to perform those tasks, applied to several problems their applicability to this kind of computations has recently been discussed comprehensively [4]. An example of the problem requiring global computation is density classification. The density classification task involves finding out if the majority of cells in the CA are in “1” state. It is non-trivial for CA to solve this problem in fixed number of sets because it relies on local interactions only. Thus, a rule set for CA applied to density classification is not obvious and must be found iteratively, such as by employing evolutionary algorithms. GA was applied to this problem and the resultant rule set successfully solved the problem [4]. This

provides an encouraging example for the suitability of CA to the problems requiring global coordination – such as protein secondary structure prediction.

MATERIAL AND METHODS

Cellular Automata have been used to carry out variety of tasks in bioinformatics. They range from predicting protein subcellular location [5] to modeling biochemical pathways [6]. For a general review on numerous cellular automata approaches in biological modeling, see [7]. However, it appears that there are no published accounts of the application of cellular automata to the problem of protein secondary structure prediction.

Protein Folding depends on both local interactions between adjacent residues and global interactions between distant residues. A cellular automaton can capture both of these interactions making it an ideal candidate for studying protein folding. As the computation involved is decentralized and massively parallel [4], each residue can independently affect the conformation of the whole protein chain.

In our implementation, which is programmed in language Python, the input is the sequence of residues of the protein chain and the output is the corresponding sequence of secondary conformations adopted by each residue. The eight possible secondary conformations specified by DSSP [8] have been mapped to three secondary conformations as:

H, I, G \rightarrow H (Helix), E, B \rightarrow B (Beta Sheet), and Rest \rightarrow C (Coil)

In the cellular automaton used, each residue of the chain can be thought of as a cell and the propensities of the residue to be in a conformation (H, B or C) represent the state of the cell. The propensities are real numbers which indicate the expected secondary structure of a residue. The values of these propensities lie between 0 and 1. A large value of propensity for a conformation means that in a protein, the residue is most likely to be found in that particular conformation. Therefore, for a particular residue, the conformation with maximum value decides its predicted secondary structure. The neighborhood defined in the CA of a residue includes the central residue, five residues to its left and five residues to its right.

Let $H_{t,i}$, $B_{t,i}$, $C_{t,i}$ represent propensities of i^{th} residue of the input chain after t iterations of cellular automaton. The initial values of propensities $H_{0,i}$, $B_{0,i}$ and $C_{0,i}$ are assigned the value of respective residue's normalized Chou-Fasman parameters [9]. The parameters are normalized such that their value lies between 0 and 1. In other words, for a particular residue, $H_{0,i} = P(H) / (P(H) + P(B) + P(C))$, where $P(H)$, $P(B)$ and $P(C)$ are original Chou-Fasman parameters. Start parameters for $B_{0,i}$ and $C_{0,i}$ are determined in an analogous manner.

Apart from standard amino acid residues, the symbol “–” is used as a special kind of residue representing the amino terminus and the carboxy terminus of the protein chain. The values of $H_{t,i}$, $B_{t,i}$, $C_{t,i}$ for this residue are calculated beforehand on the basis of propensities of first and last residues in the chains of a dataset to take a particular secondary conformation (H, B or C). At both ends of the chain, multiple “–” are added for aiding calculations in the update formula given below.

Once the initial propensities $H_{0,i}$, $B_{0,i}$ and $C_{0,i}$ have been assigned to the residues, the iterations of the cellular automaton start. The new propensities are calculated according to the update formulas, which are as follows:

$$H_{f+1,k} = \left(\sum_{j=-5}^5 H_{f,k-j} * P_j \right) / \left(\sum_{j=-5}^5 P_j \right)$$

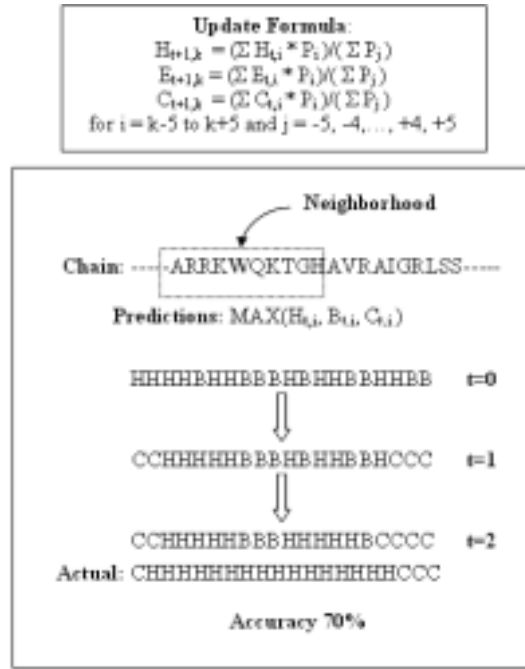


Fig. 2. Essential parts of a cellular automaton for protein structure prediction. Each cell of the lattice represents a residue. The state of the cell is given by the propensities for the conformations H, B and C. The new state in the next iteration of CA is calculated according to the update formula. The conformation (out of H, B and C) with the maximum value is interpreted as the native conformation of the residue. At $t = 0$, the propensities H, B and C are assigned the value of normalized Chou-Fasman parameters. At both ends of the chain, multiple “-” residues are added for aiding calculations required in the update formula.

$$B_{f+1,k} = \left(\sum_{j=-5}^5 B_{f,k-j} * P_j \right) / \left(\sum_{j=-5}^5 P_j \right)$$

$$C_{f+1,k} = \left(\sum_{j=-5}^5 C_{f,k-j} * P_j \right) / \left(\sum_{j=-5}^5 P_j \right)$$

Here, P_j is the weight representing the influence of the j th residue (in the neighborhood) on the conformation of the central residue. Thus, the weights attached to the neighboring residues are represented by $P_{-5}, P_{-4}, P_{-3}, P_{-2}, P_{-1}, P_0, P_{+1}, P_{+2}, P_{+3}, P_{+4}$, and P_{+5} respectively.

With each iteration, the update formula is applied to all residues (cells) of the protein chain. This way, cellular automata updates the propensities attached with the residues based on the influence of local residues. Over time, these local interactions give rise to global interactions between distant residues. Figure 2 gives a schematic overview of the overall method for prediction of the secondary structure for a given polypeptide.

RESULTS AND DISCUSSION

The output of the cellular automaton depends on three factors. Firstly, it depends on the width of the neighborhood taken into account (sometimes referred to as the “window”). Secondly, the values of the

Table 1

Evolved (*Evol*) and symmetric decay (*SD*) weights for different positions (j) in the neighborhood of central (0^{th}) residue

j	-5	-4	-3	-2	-1	0	+1	+2	+3	+4	+5
<i>Evol</i>	0.02	0.09	0.06	0.33	0.90	0.96	0.90	0.41	0.00	0.05	0.13
	0.00	0.00	0.125	0.25	0.50	1.00	0.50	0.25	0.125	0.00	0.00

weights assigned to each position of the neighborhood are important. Thirdly, the number of iterations for which the CA has been allowed to run plays a crucial role.

In the experiments carried out on this method, the length of the neighborhood was fixed (and defined by 5 residues either side from the central residue). However, the weights and the number of iterations were optimized in the current work in order to achieve maximum average Q3 accuracy. The Q3 accuracy for a prediction is defined as the ratio of number of residues for which the predicted secondary structure is correct, relative to the total number of residues present in the chain.

The dataset chosen for validation of the algorithm was the RS126 (126 Protein Rost and Sander) dataset which was defined previously [10]. This dataset contains 126 non-redundant proteins (no pairs of proteins in the set have more than 25% similarity over a length of 80 residues). Given the small similarity of the dataset sequences, this represents a situation that is rather close to real-world settings. Additionally, the CB513 dataset [11] was used to validate the Q3 of the CA.

The weights P_j were assigned using two methods. Firstly, they were assigned based on the assumption that the further the residue away, the smaller its influence on the secondary conformation of the central residue. These weights are called symmetric decay weights. Secondly, the values of weights were found using a Genetic Algorithm such that these evolved weights give maximum Q3 after the same 2 iterations. Once these weights had been found, the Q3 at different number of iterations of CA was determined. The values of symmetric decay and evolved weights are listed in Table 1.

Using the symmetric decay weights shown in the table and then optimizing the number of iterations, the maximum Q3 of 58.03% was obtained after 2 iterations (Table 2). When evolved weights are employed, the optimum Q3 was also found after 2 iterations at a value of 58.21%. As can be seen in the table, before and after 2 iterations Q3 results decreased monotonously from its optimum. With too few (0 and 1) iterations the neighboring residues are not able to influence the secondary structure of central residue to an sufficient extent. At more than 2 iterations, the influence exerted by more distant neighbors increases to such a degree that the local propensity of a central residue to adopt a particular secondary structure is over-ridden by the influence from more distal residues.

The accuracy of the predictions by the cellular automaton was also determined for the CB513 (513 protein Cuff and Barton) dataset. In this case, the maximum Q3 of 56.52% was obtained using symmetric decay weights, again after 2 completed iterations of training. Table 3 summarizes the Q3 values obtained on both datasets, RS126 and CB513. It should be noted that training of cellular automata weights was performed purely on the RS126 dataset, while the CB513 dataset was used as an external validation set.

Compared to one of the more traditional protein secondary structure prediction algorithms, our algorithm shows a better Q3 performance. For example, the Q3 of the GOR IV algorithm on the RS126 dataset is 53.3% [11], while the respective value for our algorithm is 58% (Table 3). Since the GOR IV algorithm is an improvement over the GOR algorithm, which in turn fares better than original Chou-Fasman algorithm [11], it is a valid assumption that the cellular automata presented in the current work show better performance than the traditional Chou-Fasman algorithm.

Since the CB513 dataset is more comprehensive than the RS126 dataset, the Q3 for the CB513 set probably gives a more reliable performance measure. The Q3 value of 56.5% is quite a bit lower as

Table 2
Q3 of evolved and symmetric decay weight parameters for the RS126 dataset, dependent on the number of iterations

Iterations	Q3 using evolved weights (in %)	Q3 using symmetric decay weights (in %)
0	46.14	46.14
1	56.87	56.43
2	58.21	58.03
3	57.23	57.32
4	56.16	56.76
5	55.27	56.20

The numbers in bold indicate the maximum value of accuracy obtained for different number of iterations using evolved and symmetric decay weights.

Table 3
Q3 of evolved and symmetric decay weights for the RS126 and CB513 datasets after the optimum number of 2 iterations

	Q3 on the RS126 dataset (in %)	Q3 on the CB513 dataset (in %)
Evolved weights	58.21	56.43
Symmetric Decay weights	58.03	56.51

compared to Q3 of around 68% by Baldi *et al.*'s [12] BRNN approach, but also without the use of any additional, evolutionary information. One of the possible reasons for this is that our method models a simple linear relationship between neighboring residues, while a neural network can model complex non-linear functions. Hence, the method could be improved if the update formula is substituted by for example a non-linear function evolved using genetic programming.

The other reason for performance differences could be the use of Chou-Fasman parameters [13]. These parameters are independent of the context in which they are used. Hence, a residue will have the same parameters irrespective of its neighboring residues, the overall composition of the protein chain and the position at which it is found in the chain. A possible improvement in the method could be made if the preliminary assignment of propensities is done on the basis of additional information from the whole protein chain. For example, the algorithm PSIPRED [14] outputs the individual probabilities for the residues to adopt a particular secondary conformation (H, B or C). These probabilities could be assigned as the initial propensities, and then the cellular automaton may improve upon the results of the PSIPRED algorithm.

The simple function of MAX(H,B,C), which interprets the secondary conformation of a residue based on its propensities, could also be replaced by a more efficient function. Functions for such purpose can be evolved using Genetic Programming [15].

CONCLUSION

Cellular automata make use of local interactions to simulate global and decentralized phenomena. In this work we demonstrate their applicability to the problem of protein folding, which is related in nature because of residue interactions which are initially local, but still governed by global energetic

considerations. Our method, despite its simplicity, shows a Q3 performance of 58.21% on the RS126 dataset and a value of 56.51% on the more comprehensive CB513 dataset. Those values are achieved without employing any auxiliary information. Compared to the GOR IV algorithm, which in turn represents an improvement over the original Chou-Fasman algorithm, superior performance is observed (the GOR IV algorithm shows a performance of 53.3% for the Q3 value). Thus, we demonstrate a simple algorithm for the prediction of protein secondary structure which at the same time shows similarities to the phenomenon attempted to model in its underlying structure.

ACKNOWLEDGEMENTS

The authors would like to thank the P53 Knowledgebase team at the Bioinformatics Institute, Singapore, and the two anonymous reviewers for their valuable comments. Andreas Bender thanks the Education Office of the Novartis Institutes for BioMedical Research, Inc. for a postdoctoral fellowship.

REFERENCES

- [1] Rost, B. (2001). Review, Protein Secondary Structure Prediction Continues to Rise. *J. Struct. Biol.* **134**, 204-218.
- [2] Berlekamp, E. R., Conway, J. H. and Guy, R. (1982). *Winning Ways for Your Mathematical Plays*, Vol. 2, Academic Press, New York, NY.
- [3] Wolfram, S. (2002). *A New Kind of Science*. Wolfram Media, Inc., Champaign, IL.
- [4] Mitchell, M., Crutchfield, J. P. and Das, R. (1996). Evolving cellular automata with genetic algorithms: a review of recent work. *In: Proceedings of the First International Conference on Evolutionary Computation and Its Applications (EvCA '96)*. Russian Academy of Sciences, Moscow, Russia.
- [5] Xiao, X., Shao, S., Ding, Y., Huang, Z. and Chou, K.-C. (2006). Using cellular automata images and pseudo amino acid composition to predict protein subcellular location. *Amino Acids* **30**, 49-54.
- [6] Kier, L. B., Bonchev, D. and Buck G. A. (2005). Modeling Biochemical Networks, A Cellular-Automata Approach. *Chem. Biodivers.* **2**, 233-243.
- [7] Ermentrout, G. B. and Edelstein-Keshet, L. (1993). Cellular automata approaches to biological modeling. *J. Theor. Biol.* **160**, 97-133.
- [8] Kabsch, W. and Sander, C. (1983). Dictionary of protein secondary structure, pattern recognition of hydrogen-bonded and geometrical features. *Biopolymers* **22**, 2577-637.
- [9] Chou, P. Y. and Fasman, G. D. (1974). Prediction of protein conformation. *Biochemistry* **13**, 222-245.
- [10] Rost, B. and Sander, C. (1993). Prediction of protein secondary structure at better than 70% accuracy. *J. Mol. Biol.* **232**, 584-599.
- [11] Cuff, J. A. and Barton, G. J. (1999). Evaluation and improvement of multiple sequence methods for protein secondary structure prediction. *Proteins* **34**, 508-519.
- [12] Baldi, P., Brunak, S., Frasconi, P., Soda, G. and Pollastri, G. (1999). Exploiting the past and the future in protein secondary structure prediction. *Bioinformatics* **15**, 937-946.
- [13] Kyngas, J. and Valjakka, J. (1998). Unreliability of the Chou-Fasman parameters in predicting protein secondary structure. *Protein Eng.* **11**, 345-348.
- [14] Jones, D. T. (1999). Protein secondary structure prediction based on position-specific scoring matrices. *J. Mol. Biol.* **292**, 195-202.
- [15] Aggarwal, V. and MacCallum, R. M. (2004). Evolved Matrix Operations for Post-Processing Protein Secondary Structure Predictions. *In: Proceedings of the 7th European Conference on Genetic Programming, EuroGP 2004, LNCS 3003*, pp. 220-229.