Non-Isomorphic Coding in Lattice Model and its Impact for Protein Folding Prediction Using Genetic Algorithm

Md Tamjidul Hoque, Madhu Chetty and Laurence S. Dooley

***Abstract*— Traditional encodings for Hydrophobic (H) – Hydrophilic (P) model or HP lattice models is isomorphic, which adds unwanted variations for the same solution, thereby slowing convergence. In this paper a novel non-isomorphic encoding scheme is presented for HP lattice model, which constrains the search space. In addition, similarity comparisons are made easier and more consistent and it will be shown that non-deterministic search approach such as Genetic Algorithm (GA) converges faster when non-isomorphic encoding is employed.**

I. Introduction

*P*ROTEIN *Folding Prediction* (PFP) using *Hydrophobic* (H) and *Hydrophilic* (P or Polar) or HP lattice model was introduced by Dill [1]. It uses a simplified version of amino acid sequence having only two types of monomers, namely 'H' and 'P', and the chain is placed as a *self-avoiding walk* (SAW) on this lattice path. Search using this model looks for the valid conformation (i.e. *SAW*) which has the maximum number of *topological neighboring* (TN) (Fig. 1) of H-H contacts [2], where the *H*s are neither covalent bonded [3] nor sequential in the amino acid chain sequence. The challenge with this search however, is that an inordinate number of the possible *SAW* conformations exist, even for relatively short length amino acid sequences [4]. If there are $(n+1)$ amino acids in a sequence then the number of *SAW* conformations is approximately:

$$C_n = A\mu^n n^{\gamma-1} \quad (1)$$

The connective constant or effective coordinate number [5], μ varies from lattice to lattice and has a estimated value 4.68401 for HP like simple lattice model and $A = 1.205$. The universal exponent γ=43/32 for the 2D HP model and $\gamma \approx 7/6$ for the 3D case. For instance, for only n=26, C_n = 549 493 796 867 100 942 in the 3D model [6].

Md. T. Hoque is with the Gippsland School of Information Technology, Monash University, Australia (corresponding author to provide phone: +61 3 5122 6778; fax: +61 3 9902 6842; e-mail: Tamjidul.Hoque@infotech.monash.edu.au).

M. Chetty is with the Gippsland School of Information Technology, Monash University, Australia (e-mail: madhu.chetty@ infotech.monash.edu.au).

L. S. Dooley is with the Gippsland School of Information Technology, Monash University, Australia (e-mail: Laurence.Dooley@ infotech.monash.edu.au).

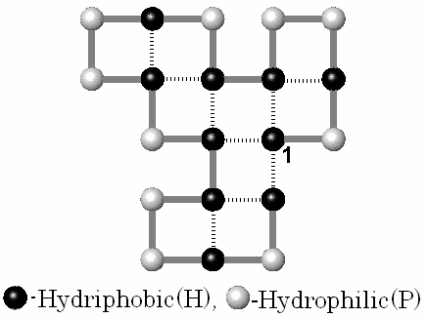

Fig. 1. Conformation in 2D HP Model shown by solid line. Dotted line indicates TN. Fitness = -(TN Count) = -9.

From equation (1), it can be easily seen that for any amino acid sequence of practical length, the conformations of *SAW* in the lattice are enormous, so therefore the exhaustive search or exact enumeration [4]-[9] becomes critical and is clearly infeasible for a straightforward implementation. Also, the *PFP* in HP model has been proven to be NP-complete [10]-[11] so no deterministic search is possible. Evolutionary Algorithm such as Genetic Algorithm is very impressive [12]-[15] in finding the optimum solution as a nondeterministic search approach. Several other outstanding nondeterministic approaches such as a number of versions of Monte Carlo (MC) [16]-[17], Simulated Annealing (SA), and Tabu Search with GA (GTB) [18], Ant Colony Optimization [19] are available. But, all these approaches also affected more or less with the increasing length of the sequence.

For investigating the PFP problem, several encoding schemes [19]-[24] are used for the presentation of the problem, though these schemes are not *non-isomorphic*, i.e., they can result in different encoded presentation of the same conformation, so *isomorphic* encodings are *surjective*. This provides the motivation to develop easily implementable non-isomorphic encoding algorithm which can result in benefits, such as (1) representing 1:1 coding versus conformation, (2) exact comparison between two solutions, (3) reducing the search space (4) reduce the race of selecting a number of equally competent optimal solutions and hence increase the convergence rate. Further, equally competent solutions cause the GA to behave as a random rather than stochastic search and this can be avoided using non-isomorphic encoding.

The organization of paper is as follows. HP lattice model is described in section II. Encoding schemes are described in

1-4244-0623-4/06/$20.00 ©2006 IEEE.

section III including algorithms. Section IV describes the theory behind the impact of non-isomorphic encoding for the Genetic Algorithm (GA). Section V describes the experiments and results. Finally, section VI draws the conclusions.

II. The HP Lattice Model

The HP model is based on the fact that the *hydrophobic* forces dominate protein folding. In this model, amino acids are represented as a reduced set of *H* (Hydrophobic or Non-Polar) and *P* (Hydrophilic or Polar) monomers only. For the purpose of *PFP*, protein conformations of the sequence are placed as a *self-avoiding walk* (SAW) on a 2D square or 3D cube lattice. The energy of a given conformation is defined as a number of *topological neighboring* (TN) contacts between those Hs, which are not sequential with respect to the sequence. The *PFP* is formally defined as:

Given amino-acid sequence, $s = s_1, s_2, s_3, \cdots, s_N$, a conformation *c* needs to be obtained [19] where,

$c^* \in C(s)$

and energy $E^* = E(C) = \min\{E(c) \mid c \in C\}$.

Here, *N* is the total number of amino acids in the sequence and $C(s)$ is the set of all valid (i.e. *SAW*) conformations of *s*. If the number of TNs in a conformation *c* is *q* then the value of $E(c)$ is defined as $E(c) = -q$ and the *fitness function* is defined as $F = -q$. The optimum conformation will have maximum possible value of |F|. In a 2D HP model (Fig. 1) a non-terminal and a terminal residue, both having 4 neighbour can have maximum of 2 *TN*s and 3 *TN*s respectively. In case of 3D model, maximum possible number of neighbours is 6 and the maximum possible TNs are 4 and 5 respectively for a non-terminal and terminal residue of the sequence.

III. Encoding Strategies

The implementation of GA using HP model for *PFP* is pioneered by Unger and Moult [13]-[14]. For the encoding of the sequence (i.e. the chromosome or genotype), rather than using binary string they preferred to use conformations themselves; which can be handled directly using genetic operators. As shown in Fig. 2, encoding of the conformations ((a) and (b)) can be as of Fig. 2 (c) and Fig. 2(d) respectively. One point mutation has been applied in the form of pivot rotation [13]. Single point crossover is done by direct cut and exchange of sub-parts between two conformations which may involve rotation before joining to have *SAW*. Following Unger's approach, later approaches mainly [20]-[21], introduced encodings. There are two common types of encodings namely (a) absolute encoding and (b) relative encoding; which are used to encode the conformation in the lattice model.

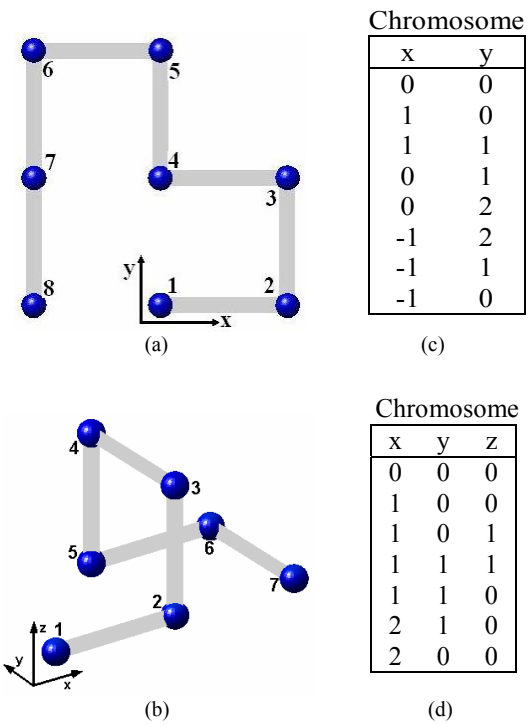

Fig. 2. (a) 2D Conformation (phenotype) (b) 3D Conformation. Corresponding chromosome (genotype) presentation is shown in (c) for 2D and in (d) for 3D, respectively.

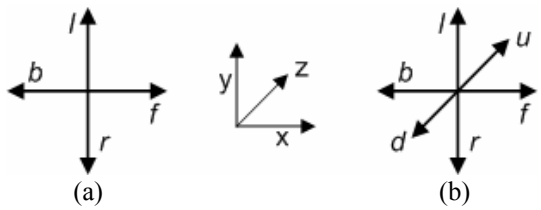

Fig. 3. Absolute moves in (a) 2D and in (b) 3D.

A. Absolute Encoding

It is intuitive to see that the absolute encoding [19]-[23] can be used instead of direct coordinate presentation. With respect to the lattice, the moves presented (shown in small case letters) for absolute encoding, are *f* (forward), *l* (left), *r* (right), *b* (back), *u* (up) and *d* (down) (Fig. 3). A conformation *c* in 2D of *n* residues can be of $c \in \{f, l, r, b\}^{n-1}$ and in 3D can be of $c \in \{f, l, r, b, u, d\}^{n-1}$. The chromosome presented in Fig.2 (a) and Fig.2 (b), can be presented using absolute encoding as *flblbrr* and *fuldfr* respectively. Besides the ease of presentation, we are also interested to investigate the conformational isomorphism of this encoding strategy. In 2D, eight different encoding chromosomes are possible by positioning the same conformation in different direction. In the conformation of Fig. 2(a), it can be assumed the line connecting '1' to '2' is aligned towards +x direction. Now the conformation repositioned in three more directions by aligning that '1' to

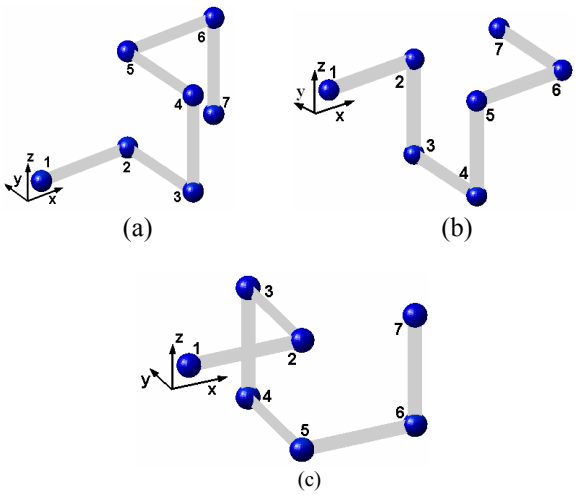

Fig. 4. Rotating Fig. 2(b) by fixing the direction of connecting line of residue '1' and '2', three different instances (a, b, c) are possible.

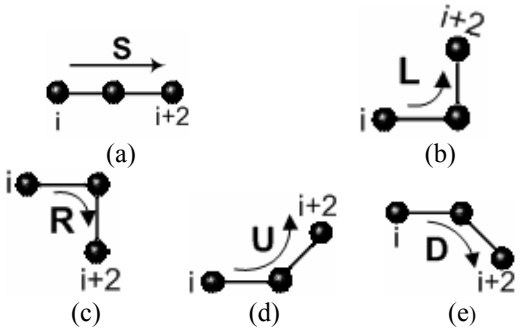

Fig. 5. (a) to (e) represent relative moves scheme: Straight (*S*), Left (*L*), Right (*R*), Up (*U*) and Down (*D*) respectively.

	F	L	R
f	*f*	*l*	*r*
l	*l*	*b*	*f*
r	*r*	*f*	*b*
b	*b*	*r*	*l*

(a)

	F	L	R	U	D
f	*f*	*l*	*r*	*u*	*d*
l	*l*	*u*	*d*	*f*	*b*
r	*r*	*d*	*u*	*b*	*f*
u	*u*	*f*	*b*	*l*	*r*
d	*d*	*b*	*f*	*r*	*l*
b	*b*	*r*	*l*	*d*	*u*

(b)

Fig. 6. The matrices show absolute versus relative coding conversion for (a) square and (b) cube lattice. Left most columns (non-italic small case) showing the previous absolute move while the rest of the columns are showing next choice (*italic*, small case) of moves and the resulting relative move corresponding to that absolute move is indicated by the top most row element (capital letters).

'2' connecting line aligning towards +*y*, -*x* and –*y*. Further, these four positions will have four different mirror images, i.e. in total there are eight possible chromosomes and the chromosomes will be *flblbrr*, *lbrbrff*, *brfrfll*, *rflflbb* and the corresponding chromosomes achieved by mirror images are *blflfrr*, *lfrfrbb*, *frbrbll* and *rblblff* respectively. The conformation in Fig. 2 (b) can have 24 different isomorphic encoded chromosomes in total. By keeping the connecting line of residue '1' to '2' in the +*x* direction, it can be rotated to four different positions (the three other instances are shown in Fig. 4, (a) to (c)). Similarly, four different positions are possible for each while aligning the '1' to '2' connecting line towards the other directions namely +*y*, -*x*, -*y*, +*z* and –*z*. Clearly, a unique conformation can have several different absolute encoded forms. Therefore, absolute encoding is isomorphic.

B. Relative Encoding

Relative encoding was attempted in [20], with a view to improved presentation over absolute encoding. Using relative encoding, pivot mutation can be presented with single locus or character alteration of a chromosome. This property benefits the presentation ease merely. In relative encoding, the move direction are defined relative to the direction of the previous move (Fig. 5), rather than relative to the axes defined by the lattice. Therefore, these moves are lattice automorphic [22]. The initial move is always expressed by *F* represents *Forwards*. A conformation *c* of *n* residues in 2D and 3D can be of $c \in \{F, L, R\}^{n-2}$ and $c \in \{F, L, R, U, D\}^{n-2}$, respectively.

The absolute versus relative conversion matrix [23] (Fig. 6) is useful for the conversion from absolute to relative encoding. The conformations shown in Fig. 2(a) and in Fig. 2(b) for example, can be expressed using relative encoding as *FLLRLLF* and *FUURRR*, respectively. Now, again we like to investigate whether the relative encoding is isomorphic or not. If we rotate Fig. 2(a) and align the residue '1' to '2' connecting line towards +*y*, -*x* and –*y*, the resulting relative encodings for the positioning remain the same as *FLLRLLF*, which can be verified by converting the aforementioned corresponding absolute encoded form to relative encoded form. But, the mirror image of this conformation results *FRRLRRF* which is different. In 2D therefore, relative coding can have two different encoded results of the same conformation. Similarly in 3D, relative encoding can have four different encoded results for the same conformation positioned differently. For instance, conformations in Fig.2 (a), Fig. 4 (a), Fig. 4 (b) and Fig. 4(c) would have relative encoding results *FUURRR*, *FRRUUD*, *FDURLL* and *FLRUDU*, so it is clear that relative encoding is isomorphic as well. Infact, it is obvious that the moves *L*, *R*, *U* and *D* are isomorphic to themselves and also the decoding in this case is non-trivial, with a base transformation for every relative move being necessary [22]. However, it is worth mentioning that, relative encoding reduces search space over absolute encoding. It has also been shown in [21] through experiment results that relative encoding had improved prediction accuracy over absolute encoding. But the reasons for its superiority had never been explained.

Algorithm-1: Non-Isomorphic Encoding

Input: Sequence *S*, presented using (*x*, *y*, *z*) coordinate
Output: *ES* /* Encoded Sequence */

BEGIN
1. $\overline{V}_1 = \overline{P}_2 - \overline{P}_1$; $\overline{V}_2 = \overline{P}_1 - \overline{P}_2$; ES = 1; T_{3D}=T_{2D} = False
2.FOR i = 3 to N /* N = Number of Residues */
3. $\overline{V}_{tmp} = \overline{P}_i - \overline{P}_{i-1}$;
4. IF T_{3D} = False /* T_{3D} : Occurrence of 3D turn */
5. IF T_{2D} = False /* T_{2D} : Occurrence of 2D turn */
6. IF $\overline{V}_{tmp} = \overline{V}_1$ then ES = ES + '1';
7. ELSE T_{2D} = True; ES = ES + '3';
8. $\overline{V}_3 = \overline{V}_{tmp}$; $\overline{V}_4 = \overline{P}_{i-1} - \overline{P}_i$;
9. END IF
10. ELSE $k \leftarrow$ match $\overline{V}_k = \overline{V}_{tmp}$: FOR k = 1 to 4
11. IF match found then ES = ES + str (k);
12. ELSE T_{3D} = True; ES =ES + '5';
13. $\overline{V}_5 = \overline{V}_{tmp}$; $\overline{V}_6 = \overline{P}_{i-1} - \overline{P}_i$;
14. END IF
15. END IF
16. ELSE $k \leftarrow$ match $\overline{V}_k = \overline{V}_{tmp}$: FOR k = 1 to 6
17. ES = ES + str (k);
18. END IF
19. NEXT *i*
END.

C. Non-Isomorphic Encoding

In this paper, a new encoding scheme is proposed which is non-isomorphic. In this encoding, numbers are used instead of letters for indicating move direction to avoid confusion with the actual meaning of any assigned direction. The proposed encoding is very simple and intuitive, which can be applied to remove the usage of isomorphism property of the above mentioned absolute and relative encodings. The general theme is to assign fixed order of directions of a growing chain, based on the first occurrences of the move to any dimension. For all possible positioning of the same conformation, the coding would result in a unique encoded chromosome.

In this encoding, the direction of first points towards the second is marked '1' and the reverse is marked '2', which defines the complete move in single dimension. Direction of first occurrence of any 90 degree (in any co-ordinate based direction) turn, i.e. $move \in \{L, R, U, D\}$ is coded '3' and the reverse is '4', which completes the moves for second dimension. Then, the first occurrence of the move perpendicular to the plane formed by '1' and '3' moves is marked as '5' and the reverse is '6', which finally defines the moves on third dimension. Algorithm-1 describes the encoding procedure in details.

In Algorithm-1, coordinates of residue or point are presented using vector, such as $\overline{P}_i$ is the i^{th} point. Further unit vector are presented, such as $\overline{V}_i$ forms the i^{th} unit vector. As a run through example for non-isomorphic encoding by Algorithm-1, consider the conformation of Fig. 2(b) and the chromosome represented at Fig. 2(d). $\overline{P}_1 = (0\,0\,0)$ and $\overline{P}_2 = (1\,0\,0)$, therefore from line#1, $\overline{V}_1 = (1\,0\,0)$ and $\overline{V}_2 = (-1\,0\,0)$. Now, $\overline{V}_{tmp} = (1\,0\,1) - (1\,0\,0) = (0\,0\,1)$ and then line# 7 is executed and resulted in ES='13' and second dimensional vectors are achieved and then defined as $\overline{V}_3 = (0\,0\,1)$ and $\overline{V}_4 = (0\,0-1)$. Now, $\overline{V}_{tmp} = (1\,1\,1) - (1\,0\,1) = (0\,1\,0)$. Line#12 is executed, the value of the ongoing result is, ES='135' and third dimensional vectors are achieved and defined as $\overline{V}_5 = (0\,1\,0)$ and $\overline{V}_6 = (0-1\,0)$. Now, at this stage, all the vectors for 3D is defined fully, so, in the next iterations Line#3, Line#16 and Line#17 will be executed. Eventually, after 3 more iterations, the value of ES becomes *135416*. For sake of completeness, the decoding algorithm for non-isomorphic encoding is also provided in the Appendix.

Encoded result of the conformations of Fig. 4(a), 4(b) and 4(c) generate the same encoded string *135416*. This can be further verified to be true for any possible positions of the conformation of Fig. 2(b). As an illustration, consider a 'U' pattern or motif in any possible dimensional and rotating position. The encoded output using this proposed scheme always results the same, *132* (i.e. non-isomorphic). Whereas it can be easily verified that traditional encoding schemes will be isomorphic in encoding. For a sequence of N amino acids the proposed encoding will be of length $(N-1)$. It can also be of length $(N-2)$, omitting the default specification of the initial move, as '1' is always at the first position.

IV. Significance of the Non-Isomorphic Encoding

In this section, the significance of the proposed non-isomorphic encoding scheme is examined.

A. 1:1 Mapping

Non-isomorphic encoding provides a 1:1 mapping between conformation and the encoded form, i.e. it performs *bijective* mappings between genotype and phenotype. Certainly, it would ease the detection of distinct conformations having optimum fitness irrespective of the algorithm used. Moreover, with subsequent iterations of a search algorithm, there will be less chance of having detection of the optimum value in the isomorphic encoding presentation rather than non-isomorphic one having the same fitness and the reasons are given in section IV.D.

B. Reducing the Search Space

Certainly the non-isomorphic encoding is eliminating the number of possible mirror conformations and providing a unique presentation of an individual. Compared to absolute encoding in square lattice for *SAW*, non-isomorphic encoding results in 8 times less encoded search space. For cube lattice in 3D, non-isomorphic encoding results in a 24 times reduced encoded search space. This is because, a 3D conformation can be rotated by fixing the connecting line of residues '1' to '2' to four different places leading to 4 different absolute encoding. This can be further varied by placing 3D coordinate with total of 6 possible directions. As the absolute encoding is relative to the axis defined by the lattice, it would produce 24 different encoding for the same conformation. It should be noted that this can be summarized by just one non-isomorphic encoding.

Compared to relative encoding, the non-isomorphic encoding would produce 2 times less for square lattice and 4 times less encoded results for cube lattice.

C. Constrained Move Sequences by Non-Isomorphic Encoding

It is reported in [20]-[22] that the relative encoding is 1-step self-avoiding since there is no *B* (back) move compared to absolute encoding. But, presentation using non-isomorphic encoding imposes more constraints. For example the first encoded value must be '1'. Any of '2', '4', '5' or '6' can not occur unless a '3' can occur. A '6' can not occur unless a '5' can occur. Fig. 7 shows the state diagram depicts the constraints in general, but beyond that, there are other inherent constraints. For example, no reverse move is allowed immediately after the corresponding move, say, no '2' after '1' or no '4' after '3' can occur, so on. That is, for pairs (1, 2), (3, 4) and (5, 6), any member of a pair will never be adjacent to the member of the same pair. Moreover, the encoded sequence must reject conformation that maps to *non-SAW* conformation.

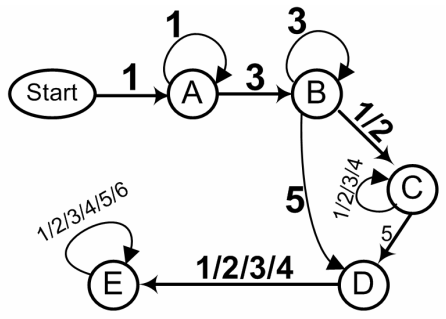

Fig. 7. State diagram of the non-isomorphic encoding. The terminating condition is the final length of the encoding. The encoding can only start with a '1' and remain at initial state ('A') with consecutive '1's. Only the alternative input is '3' which changes the state from 'A' to 'B'. It can remain at 'B' for consecutive '3's. Next with input '1' or '2' the state becomes 'C' or, with input '5' the state becomes 'D' from 'B' or from 'C' as well, so on.

D. Impact on Search Using Genetic Algorithm (GA)

Now the question is, how significant the reduction of the search space is? For manageable search space, 4 (in 2D) to 24 (in 3D) times reduction is clearly significant for helping faster convergence. Even though a search space is larger, its connectivity would make it easier to navigate. In this section, we analyze the effect of isomorphic encoding for a nondeterministic search approach, namely, Genetic Algorithm (GA), for the intractable search space.

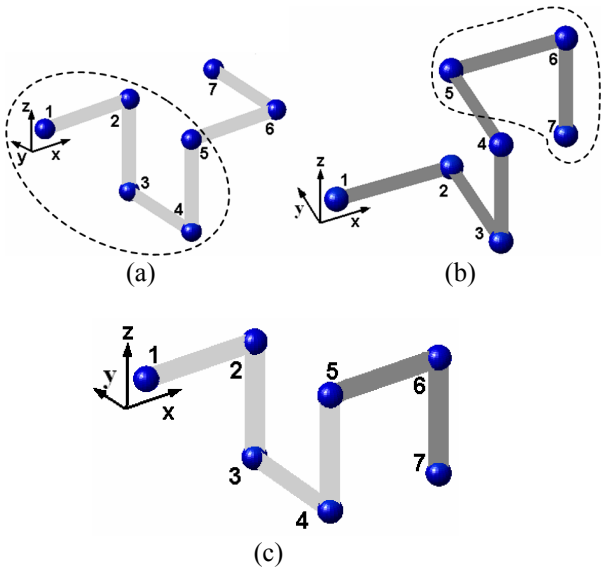

Fig. 8. Conformation (a) and (b) are basically the same but can result different encoded chromosome from isomorphic encoding. The crossover between (a) and (b) can result conformation (c). The inherited part from the parents in (c) has been indicated by closed area, marked using dotted line both in (a) and (b). Clearly, (c) can be achieved from (a) or (b) by (pivot rotation) mutation at residue '5'. The subparts joined and formed the offspring, do not carry any new sub-conformation with respect to any of the parents, hence it is not stochastic.

GA computation is based on schemata theorem, [12] [25]-[30] which shows mathematically that the GA optimizes the effort of testing and producing new individuals if their representation permits development of building blocks (called schemata). GA is driven by an implicit parallelism and generates significantly more successful descendants than random search. Based on effective fitness [30], it has been shown that schemata of higher than average effective fitness receive an exponential increasing number of trials over time. In such a case, where schema reconstruction is favored over schema destruction, large schemata tend to be favored. Therefore, similarity among population would grow quickly and would further lead to competition among equally competent solution having higher but equal fitness [25] [31]-[33]. Therefore, GA would face multiple times additional competition of selecting equally fitter solution at least, which would slow down convergence rate due to the

TABLE I
LIST OF SEQUENCE TESTED

ID	Sequence	Len
273d.1	(PH)2PH3P2(HP)2P10H2P	27
273d.2	PH2P10(H2P2)2HP2HPH	27
273d.5	H4P4HPH2P3H2P10	27
273d.4	H3P2H4P3(HP)2PH2P2HP3H2	27
273d.6	HP6HPH3P2H2P3HP4HPH	27
273d.7	HP2HPH2P3HP5HPH2(PH)3H	27
SL.4	PHPH2P2HPH3P2H2PH2P3H5P2HPH2(PH)2P4HP2(HP)2	48
SL.6	H3P3H2PH(PH2)3PHP7HPHP2HP3HP2H6PH	48
SL.9	(PH)2P4(HP)2HP2HPH6P2H3PHP2HPH2P2HPH3P4H	48

The first six of length (Len) 27 are taken from [14]. Last three are from [34].

TABLE II
PERFORMANCE COMPARISONS

Fitness	ID: 273d.1 (Avg. generation)		ID: 273d.2 (Avg. generation)		ID: 273d.5 (Avg. generation)	
	Iso.	Non-Iso	Iso.	Non-Iso	Iso.	Non-Iso
-5	2.142	1.7	2	1.857	1.75	2.14
-6	4.375	4.8	5.3	3.44	5.2	4.66
-7	10.333	10.5	9.4	8.5	49	20.62
-8	33.2	31.66	14.33	15	237.6	59.33
-9	265.57	145.33	43.33	59.28	-	-
-10	-	-	353.42	240.14	-	-

Convergence rate comparison between Iso. (Isomorphic) versus Non-Iso. runs. Average generation is taken from 10 iterations.

TABLE III
PERFORMANCE COMPARISONS

Fitness	ID: 273d.4 (Avg. generation)		ID: 273d.6 (Avg. generation)		ID: 273d.7 (Avg. generation)	
	Iso.	Non-Iso	Iso.	Non-Iso	Iso.	Non-Iso
-7	-	-	6.25	5.3	-	-
-8	-	-	8.4	8.55	-	-
-9	-	-	15.545	14	7.571	8.125
-10	7.75	7.8	57.125	62.3	14.8	12.33
-11	13.75	11.88	547	138	39.88	22.55
-12	29.4	26.37	810	320	54.9	87.88
-13	98.44	83.14	-	-	362.7	232.83
-14	163.11	202.88	-	-	-	-
-15	512.66	283.25	-	-	-	-

Average generation is taken from 10 iterations.

TABLE IV
PERFORMANCE COMPARISONS

Fitness	SL.4 (Avg. generation)		SL.6 (Avg. generation)		SL.9 (Avg. generation)	
	Iso.	Non-Iso	Iso.	Non-Iso	Iso.	Non-Iso
-21	-	-	57.44	66.37	-	-
-22	120.28	84.5	111.22	85.12	79.11	35.5
-23	274	100.3	161	110.5	105.37	48.6
-24	355.5	132.5	225.1	155.7	213.44	68.66
-25	401.14	249	397.14	256.5	302.87	98
-26	646	315.5	641.25	410	344.85	199.33
-27	-	-	-	-	370.8	309.6

Average generation is taken from 20 iterations.

usage of the isomorphic coding. Certainly, the non-isomorphic presentation would reduce the race by diminishing the isomorphic variations of the same conformation. Therefore, the convergence rate using non-isomorphic encoding would perform faster and more accurate.

Another, very important issue needs to be pointed out. GA is stochastic search [25]-[30] and it is well established [12] [13] that stochastic search perform better than random search. In GA, mutation introduces randomness and crossover introduces stochastic relation. As the mutation rate is usually kept low and the crossover rate is kept high [25] [33], GA therefore behaves as a stochastic search rather than a random search. Conclusively, if the mutation rate is set at a high rate, GA would behave as a random search. Now, with the nature of schemata theorem and the usage of isomorphic encoding, it is highly likely that the equally competent same conformation differentiated by isomorphic encoding variation exist (due to same fitness) in increasing manner, as the search converges. In such case, the crossover between same conformations would increase eventually. The crossover between identical conformations having different (isomorphic) encoding would produce a result, which is equivalent to a mutation operation (Fig. 8). For example, with respect to the relative encoding, the operation shown in Fig.8 can be explained as,

Crossover (*FDURLL, FRRUUD*) → *FDURLD*

But, *Mutation* (*FDURL**L***, L→ D) → *FDURL**D***

Or, *Mutation* (*FRRUU**D***, D→ R) → *FRRUU**R***

where, *FDURLD* and *FRRUUR* correspond to same conformation, which can be expressed as *135413* using non-isomorphic encoding. Thus, even if the crossover rate is explicitly set very high and the mutation rate is set very low, the crossover would act as mutation at an increasing rate and therefore implicitly the mutation rate increases decreasing the actual and effective crossover rate. Clearly, at this stage GA become a random search rather than stochastic. The comparative experimental results in the next section highlight this fact.

V. Experimental results

We have implemented Unger and Moult's version of GA without applying cooling and it is termed as the *isomorphic* (*Iso*.) run, while conversely our proposed algorithm is termed as non-isomorphic (*Non-Iso*), both running concurrently for experimental purpose. For this simulation, relatively short sequences (see Table I) are preferred where the length permits easy convergence using GA. Since, the target is to compare the convergence rate, simple GA used for prediction which is similar to that of [13]-[14] with coordinate based presentation, single point crossover and single point mutation (pivot rotation), but without any special treatment such as cooling. The population size was 200 for all the instances. The crossover rate was set to 0.8, mutation rate at 0.1. The elite rate was set to 0.05, *Roulette Wheel* selection was applied. On the other hand, simulation run for the proposed algorithm using non-isomorphic encoding, the non-isomorphic sub-parts reduce the race being survived altogether and help faster convergence. Table II and Table III shows the simulation results of sequences of length 27 and Table IV shows the simulation results of sequences of length 48. As expected, while the population is close to optimal or near optimal, the convergence rate is measured (from last rows of each comparison) to be 46%, 32%, 75%, 44.74%, 60%, 35%, 51.16%, 36% and 16% faster using non-isomorphic approaches respectively in order, for the sequences listed in Table I, with the average improvement being 43.98%. Longer sequences have not been considered in the studies, since they would have their own complexities and these can unevenly impact on the convergence rate, to soothe that a very high number of simulations run would be required to proof the same.

VI. Conclusion

In this paper, it is shown that the traditional encoding schemes are isomorphic. The proposed novel encoding algorithm is non-isomorphic and proves to be superior over traditional encoding for a number of reasons. From experimental results, it has been proven that the non-isomorphic encoding helps non-deterministic search converge faster. Twins are identifiable and mating is easily restricted between identical chromosomes to avoid ineffective crossover. This novel encoding algorithm simplifies the conformation comparisons and reduces the search space, so any added complexity due to isomorphic encoding is compensated by the usage of the proposed novel non-isomorphic encoding.

Appendix

For the sake of completeness the decoding Algorithm corresponding to the non-isomorphic encoding has been described in Algorithm-2. Unit vectors ($\overline{V}_i$ s) are assigned values arbitrarily. Any consistent set of values to the unit vectors can be assigned provided each vector is distinct and the pairs ($\overline{V}_1, \overline{V}_2$), ($\overline{V}_3, \overline{V}_4$) and ($\overline{V}_5, \overline{V}_6$) have opposite direction to each other. Initial two positions ($\overline{P}_1, \overline{P}_2$) can be changed consistently.

Algorithm-2: Non-Isomorphic Decoding

Input: *ES* /* Encoded Sequence */
Output: Sequence *S*, presented using (*x*, *y*, *z*) coordinate.

BEGIN
  Unit Vector:
    $\overline{V}_1$ =(1 0 0); $\overline{V}_2$ = (-1 0 0); $\overline{V}_3$ = (0 1 0);
    $\overline{V}_4$ =(0 -1 0); $\overline{V}_5$ = (0 0 1); $\overline{V}_6$ = (0 0 -1);
  $\overline{P}_1$ =(0 0 0); $\overline{P}_2$ =(1 0 0);
  FOR *i* = 2 to Length (ES)
    CASE character (ES (*i*)) of
      Case '1' : $\overline{P}_{i+1} = \overline{P}_i + \overline{V}_1$
      Case '2' : $\overline{P}_{i+1} = \overline{P}_i + \overline{V}_2$
      Case '3' : $\overline{P}_{i+1} = \overline{P}_i + \overline{V}_3$
      Case '4' : $\overline{P}_{i+1} = \overline{P}_i + \overline{V}_4$
      Case '5' : $\overline{P}_{i+1} = \overline{P}_i + \overline{V}_5$
      Case '6' : $\overline{P}_{i+1} = \overline{P}_i + \overline{V}_6$
    END CASE
  Next *i*
END.

References

[1] K. A. Dill, "Theory for the Folding and Stability of Globular Proteins", Biochemistry, 1985, Vol. 24, No. 6, pp.1501-1509.
[2] G.B. Fogel and D. W. Corne (Editors), "Evolutionary Computation in Bioinformatics", Elsevier Science, 2004, USA.
[3] K.Yue and K.A. Dill, "Forces of Tertiary Structural Organization in Globular Proteins", Proc Natl Acad Sci USA 1995, Vol-92, pp.146-150.
[4] M. Chen and K.Y. Lin, "Universal amplitude ratios for three-dimensional self-avoiding walks", Journal of Physics A: Mathematical and General, 2002, Vol-35, pp. 1501-1508
[5] R. Schiemann, M. Bachmann and W. Janke, "Exact Enumeration of Three – Dimensional Lattice Proteins", Elsevier Science, 2005.
[6] D. MacDonald, S. Joseph, D. L. Hunter, L. L. Moseley, N. Jan and A. J Guttmann, "Self-avoiding walks on the simple cubic lattice", J. Phys. A: Math. Gen., 2000, Vol. 33, pp: 5973-5983.
[7] A.J. Guttmann, "Self-avoiding walks in constrained and random geometries": Series studies. In "Statistics of Linear Polymers in Disordered Media" ed. B. K. Chakrabarti, Elsevier, 2005, pp:59-101.
[8] R. Backofen and S. Will, "A Constraint-Based Approach to Fast and Exact Structure Prediction in Three-Dimensional Protein Models", 2005, Kluwer Academic Publishers.
[9] A. Irbäck and C. Troein, "Enumerating Designing Sequences in the HP Model", Journal of Biological Physics, 2002, Vol. 28, pp: 1-15.

[10] P. Crescenzi and et al.,"On the complexity of protein folding (extended abstract)",ACM, Proceedings of the second annual international conference on Computational molecular biology, 1998, pp:597-603.
[11] B. Berger, and T. Leighton, "Protein Folding in the Hydrophobic-Hydrophilic (HP) Model is NP-Complete", Journal of Computational Biology, 1998, Spring; Vol. 5, No. 1, pp.27-40.
[12] R. Unger, and J. Moult, "On the Applicability of Genetic Algorithms to Protein Folding", IEEE, 1993, pp. 715-725.
[13] R. Unger and J. Moult, "Genetic Algorithms for Protein Folding Simulations", Journal of Molecular Biology, 1993, Vol-231, pp. 75-81.
[14] R. Unger and J. Moult, "Genetic Algorithm for 3D Protein Folding Simulations", 5th International Conference on Genetic Algorithms, 1993, pp. 581-588.
[15] M. T. Hoque, M. Chetty and L. S. Dooley, "A New Guided Genetic Algorithm for 2D Hydrophobic-Hydrophilic Model to Predict Protein Folding", 2005 IEEE Congress on Evolutionary Computation (CEC), pp. 259-266, Edinburgh.
[16] U. Bastolla, and et al.,(1998) "Testing a new Monte Carlo Algorithm for Protein Folding", National Center for Biotechnology Information, 1998, Vol. 32, No. 1, pp.52-66.
[17] F. Liang, and W.H. Wong, "Evolutionary Monte Carlo for protein folding simulations", J. Chem. Phys., 2001, Vol. 115, No. 7.
[18] T. Jiang, and et al., "Protein folding simulation of the hydrophobic-hydrophilic model by computing tabu search with genetic algorithms", ISMB, 2003, Brisbane Australia.
[19] A. Shmygelska and H.H. Hoos, "An ant colony optimization algorithm for the 2D and 3D hydrophobic polar protein folding problem", BMC Bioinformatics 2005, Vol. 6, Issue 30.
[20] A.L. Patton, W.F. Punch III and E.D. Goodman, "A Standard GA approach to Native Protein Conformation Prediction", 1995, 6th International Conference on Genetic Algorithms, pp:574-581, ISBN:1-55860-370-0.
[21] N. Krasnogor et al., "Protein Structure Prediction With Evolutionary Algorithms", 1999, Genetic and Evolutionary Computation Conference (GECCO-99).
[22] R. Backofen and S. Will, "Algorithm Approach to Quantifying the Hydrophobic Force Contribution in Protein Folding", 2000, Pacific Symposium on Biocomputing, Vol 5, pp: 92 -103.
[23] E. Bornberg-Bauer,"Chain Growth Algorithms for HP-Type Lattice Proteins", RECOMB, 1997, Santa Fe, NM, USA.
[24] T. N. Bui and G. Sundarraj, "An Efficient Genetic Algorithm for Predicting Protein Tertiary structures in the 2D HP Model", GECCO'05, Copyright 2005 ACM.
[25] Z. Michalewicz, "Genetic Algorithms + Data Structures = Evolution Programs", New York: Springer-Verlag, 1992.
[26] S. Schulze-Kremer, "Genetic Algorithms and Protein Folding", [Online] http://www.techfak.uni-bielefeld.de/bcd/Curric/ProtEn/proten.html
[27] D. Whitley, "An Overview of Evolutionary Algorithms", Journal of Information and Software Technology, 2001, Vol-43, pp: 817-831.
[28] M. D. Vose, "The Simple Genetic Algorithm", 1999, The MIT Press, Cambridge, Massachusetts London, England.
[29] J. H. Holland, "Adaptation in Natural And Artificial Systems", Sixth printing 2001, edition 1992, The MIT Press, Cambridge, Massachusetts London, England.
[30] C. Stephens and H. Waelbroeck, "Schemata Evolution and Building Blocks", 1999 by MIT, Evolutionary Computation, Vol. 7, Issue 2, pp:109-124.
[31] S. Ronald, "Duplicate Genetypes in a Genetic Algorithm", IEEE 1998.
[32] M. T. Hoque, M. Chetty and L. S. Dooley, "Critical Analysis of the Schemata Theorem: The Impact of Twins and the Effect in the Prediction of Protein Folding using Lattice Model", 2005, Tech. Report TR-2005/8, GSIT, MONASH University.
[33] R. L. Haupt and S. E. Haupt, "Practical Genetic Algorithms, 2nd Edition, 2004, ISBN 0-471-45565-2.
[34] W. Hart, and S. Istrail, "HP Benchmarks", http://www.cs.sandia.gov/tech_reports/compbio/tortilla-hp-benchmarks.html