Significance of Hybrid Evolutionary Computation for *Ab Initio* Protein Folding Prediction

Md Tamjidul Hoque, Madhu Chetty, Laurence S. Dooley

Gippsland School of Information Technology, Monash University, Churchill,
VIC-3842, Australia.
{Tamjidul.Hoque, Madhu.Chetty, Laurence.Dooley}@ infotech.monash.edu.au

Summary: *Protein folding prediction* (PFP), especially the *ab initio* approach, is one of the most challenging problems facing the bioinformatics research community due to it being extremely complex to solve and computationally very intensive. *Hybrid evolutionary computing* techniques have assumed considerable importance in attempting to overcome these challenges and so this chapter explores some of these *PFP* issues. By using the well-known *Hydrophobic-Hydrophilic* (HP) model, the performance of a number of contemporary non-deterministic search techniques are examined. Particular emphasis is given to the new *Hybrid Genetic Algorithm* (HGA) approach, which is shown to provide a number of performance benefits for *PFP* applications.

1 Introduction

The technological advancements taking place in various sectors are contributing in a major way to address complex problems by sharing and exchanging advanced knowledge from various disciplines. *Protein folding prediction* (PFP) represents one such difficult, yet important challenge that has strong cross-disciplinary involvement such as *molecular biology*, *biophysics*, *computational biology* and *computer science.* If the mysteries of protein folding are to be unravelled, it will not only assist in combating many diseases, but it will also mean that various crucial medical, agricultural and biotechnological bottlenecks that currently exist, can be either fully or partially alleviated.

PFP has so far proven to be an intractable problem to solve within the various disciplines involved. Proteins typically exhibit some pattern in their folding, which is not random, thus the mystery can be explained to some extent, through logical inferences. From this perspective, the field of *computational biology* appears promising as it providing the support necessary to facilitate *PFP* solution, which is a *combinatorial hard optimization* problem. *PFP* research provides the opportunity, therefore to establish the significance of a particular methodological approach like *hybrid evolutionary computation*, which has been applied to solve many real-world problems due to its consistently superior performance [35, 39, 49, 62, 71, 74].

2 Background: The Protein Folding

Proteins are the most important molecules in living organisms both quantitatively as well as functionally. More than half of the dry weight of a cell is made up of proteins, which exist in various shapes and sizes. Proteins are responsible for transporting small molecules such as the hemoglobin that transports oxygen in the bloodstream, catalyzing biological functions and providing structure to collagen and skin, control sense, regulating hormones, process emotion and many other functions [54]. The really exciting information concerning proteins is not about the molecules carrying out vital tasks, but their various *shapes* which enable them to perform the tasks. Furthermore, proteins are sequences of amino acids bound into a linear chain that adopt a specific folded three-dimensional (3-D) shape, which implies to carry out their task, proteins must fold into a 3-D structure [26] from the amino acid sequence.

This is why the understanding as to how protein sequences actually determine their structures has often been referred to as the second half of genetics [19]. Prof. Pande of Stanford University [37] mentioned that "*... just about anything that needs to get done in biology is done by a protein, when you have a machine on this tiny scale, how is it built? When you are dealing with something on an atomic scale, you do not have atomic-sized hammers and screwdrivers. What biology has done is create machines that can assemble themselves. The process of self-assembly they go through is called folding*". In order to explore more deeply in this matter, we will now examine the constituents of a protein.

2.1 Inner Structure of Proteins

The sequence of amino acids in any protein defines its *primary* structure [9]. There are only 20 different amino acids and their various sequential combinations lead to the formation of different proteins. Any two amino acids will have a number of common parts, such as a central carbon (C_α) which is bonded with a hydrogen (-H), an amino group ($-NH_2$), and a carboxyl group (-COOH). They differ between themselves only by the variation of their side chains, represented in general by the symbol "R" (Fig. 1(a)). C_α is also always bonded with carbon (called C_β) of the side chain with the exception of one of the amino acid called *Glycine*, having one hydrogen atom as the side chain instead. Fig. 1(b) shows the ionized form in the aqueous solution, where the amino group are protonated to make ammonium ions and the carboxylic acids are ionized to their conjugate bases (carboxylate ions), which helps the two amino acid to be concatenated. Any two amino acids that are concatenated, release water and form *peptide bonds* as shown in Fig. 2.

From the remarkable investigation (for which he won the Nobel Prize for Chemistry in 1972), Anfinsen [5] concluded that the information determining the *tertiary* structure of a protein resides in the chemistry of its amino acid sequence, a finding that is now known as *Anfinsen's thermodynamic hypothesis*. A protein can be denatured (i.e. have forced folding deformity) by either adding certain chemicals or by applying heat. It has been experimentally verified that after removing the denaturing chemical or heat, proteins spontaneously refold to their native forms. Refolding experiments indicate that the unique native conformation does not depend on the initial state of the chain and is sufficiently stable to be independent of a variety of external factors. The *global minimum claim* (i.e. the aforementioned thermodynamic hypothesis) is supported by the fact that proteins are not experimentally observed to be in different conformations. Each protein appears to have a single native conformation so in almost all cases it is assumed [48] that when predicting a polypeptide (such as protein) structure, the native conformation corresponds to the *global minimum free energy state* of the system.

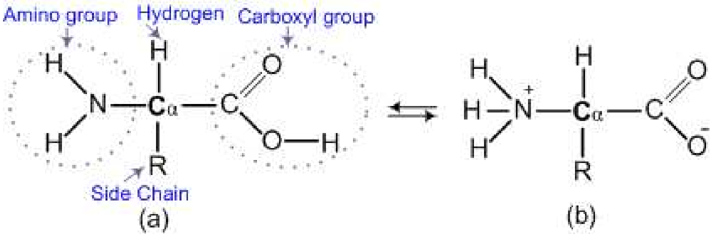


Fig. 1. (a) An amino acid showing bond structure in general. (b) Amino acid at pH 7.0.

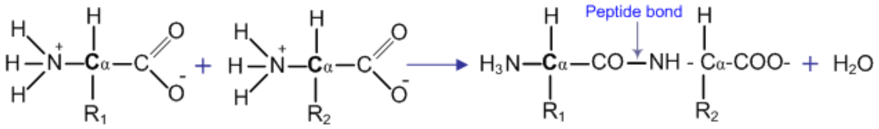


Fig. 2. Two amino acids connected by forming peptide bond and releasing water.

2.2 The Search Problem

Peptide bond helps to form the amide plane (Fig. 3) between two amino acids and to join them together. Connected this way a number of amino acids form a sequence, called the *polypeptide chain*, with this linear sequence of residues being known as its *primary* structure. Proteins actually fold in three dimensions presenting *secondary*, *tertiary* and *quaternary* structures [59]. The *secondary* structure of a protein is formed through interactions between backbone atoms only and results in local structures such as α-helix, β-sheet and so on. *Tertiary* structures are the result of *secondary* structure packing on a more global level and a further level of packing is basically a group of different proteins packed together in what is known as *quaternary* structures. Further details on all these various structures is not presented here, but for the interested reader, additional information can be found in [27].

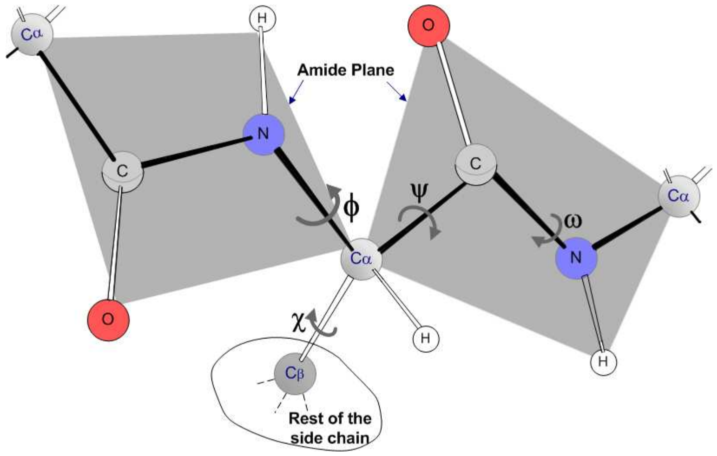


Fig. 3. Constituents of the amide plane and corresponding degrees of freedom. N-C_α, C_α-C and N-C bonds are respectively represented by torsion angle ϕ, ψ and ω. The side chain torsion angle is shown by χ.

The main chain has two degrees of freedom around the dihedral angles, ϕ and ψ (Fig. 3), while the side chains have an additional degree of freedom around their torsion angles χ as the example in Fig. 4 illustrates. Assuming there are two such χ on an average per amino acid and each angle approximately has degree of freedom of $\approx 2\pi$, then the total number of possible conformations can be expressed as:

$$C_{Tot} \approx (X_1 * X_2 * X_3 * X_4)^m \tag{1}$$

Here, m is the number of amino acids in the sequence and X_1, X_2, X_3 and X_4 indicate the permissible degrees of freedom of angles ϕ, ψ, χ_1 and χ_2 respectively. It is clear from the relationship in (1) that the total number of possible conformations is exorbitant. Due to sterical (i.e. spatial arrangement of atoms in a molecule) disallowances, some reduction in this number is feasible using what is commonly referred to as the *Ramachandran plot* [9], though the number remains inordinate. Even if a small degree of freedom is assigned, with say each amino acid having only 3 degrees of freedom (ϕ, ψ, χ) and if we further assign each free angle just 3 different arbitrary values, then for a 100 residues long protein, the total number of conformations is 3^{300}, of which only one will be the native state. In terms of computational time overheads, assuming 100 conformations per second can be sampled then this results in requiring a totally unrealistic $\approx 4.34^{133}$ years in order to explore all possible conformations.

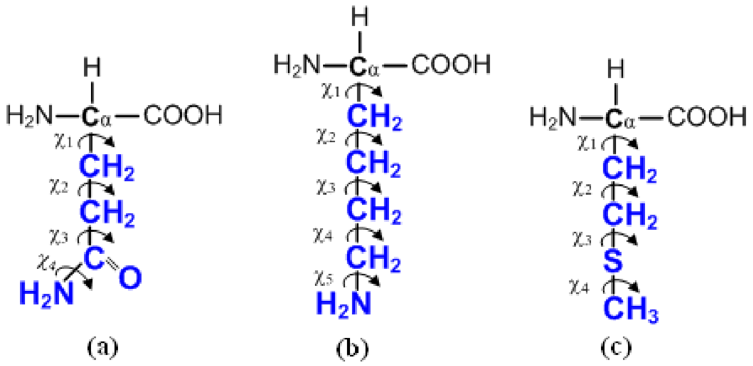


Fig. 4. Possible torsion angles (χ_i), for side chains (shown in bold lettering) for three amino acids, namely (a) Glutamine (b) Lysine (c) Methionine.

In nature, a protein folds remarkably quickly, requiring between a tenth of a millisecond and one second in general, whereas any algorithm [70, 71] on any modern computer is unable to yet exactly simulate this task, which take just a fraction of a second to perform it in nature. As Pande [37] perceptively noted "*...to simulate the very smallest, fastest protein folding right now on a fast workstation would probably take about 30 years.*" It is therefore currently not known how exactly an amino acid chain folds into its *tertiary* structure in the short time scale that occurs in the cell. Cyrus Levinthal postulated, in what is popularly known as the *Levinthal Paradox*, that proteins fold into their specific 3-D conformations in a time-span far shorter than it would be possible for the molecule to actually search the entire conformational space for the lowest energy state. It has been argued that the paradox can be settled if one views each atom as independently computing in its neighborhood, i.e., the atoms compute in parallel whereas the theoretical calculation assumes a sequential search [16]. As proteins cannot while folding sample all possible conformations, so folding is therefore not a random process and a folding pathway must exist. Additional to the astronomical number of possible conformations, there are certain forces [42] like the *hydrophobic* force, *hydrogen* bonding, *electrostatic* force, Van der Waals interactions, *disulphate bridge* and *solvation* to name just a few, that ultimately determine the final 3D conformation. Thus, the native conformation prediction process from the amino acid sequence essentially involves both the structural details of the constituents as well as the aforementioned forces regarded as energy functions or fitness function in a model. The complicated form of the energy function does not readily suggest any obvious efficient search strategy, with most searches becoming trapped in one of the numerous local free energy minima characteristic of the energy landscape. *PFP* therefore represents one of the most challenging contemporary *combinatorial hard optimization* problems.

2.3 Importance of the Protein Folding

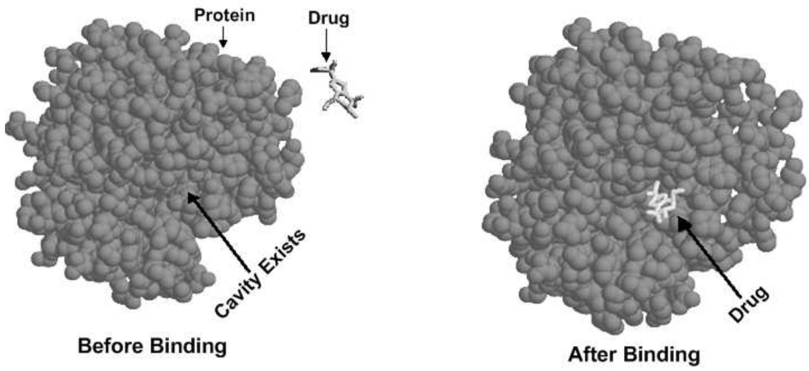


Fig. 5. Ligand binding mechanism used in drug designing; structural information is required to form the protein vs. drug binding.

Protein mis-folding [56] has been identified as the cause of about 20 diseases [37], including for example mad cow, cystic fibrosis, Alzheimer's and Parkinson's. The diseases are thought to occur in many ways, for example just one amino acid mistake in the hemoglobin that carries oxygen through the blood, leads to sickle cell anemia. Since the function of a protein is determined to a large extent by its structure, *protein folding prediction* appears to be the key to progress [2, 71] in many medical applications such as drug design (Fig. 5). A vital element to drug design [53] is that proteins function by docking, [18] where docking receptors (proteins) contain pockets to bind to the ligand (drug). Same pathway searching methodologies of *PFP* are useful for finding the position and orientation of the two molecules being energetically minimized, so that *PFP* can be very helpful in designing and synthesizing drugs as well as in applications related to areas like agriculture and biotechnology.

2.4 Available Prediction Technologies

Due to the huge number of possible conformations generated from the amino acid sequences, systematic exhaustive searching is impracticable [55]. To analyse the protein structure in practice, *X-ray crystallography* (XC) and *nuclear magnetic resonance spectroscopy* (NMR) are used [55, 77]. In the former, the protein must be crystallized before applying *X-rays* for structure determination, though crystallization may take days, or even months or years. Still many proteins, especially those that are attached to the cell membrane cannot be crystallized. For the process of *XC* for prediction, it is not only needs to apply *X-rays* onto the crystal, but it also requires expertise including elaborate calculations and translation of the received deflection pattern in various position of the crystal. Conversely, the principle behind *NMR* is that some atomic nuclei such as hydrogen are intrinsically magnetic and upon application of magnetic field they can adopt different energies. *NMR* avoids the need of crystallized protein and so is faster than *XC*, but for longer proteins, *NMR* results become less precise. Both, *XC* and *NMR* are however labor and time [26] intensive and are inadequate for the increased demand mandated by *PFP*. This has led a raft of computational approaches being proposed in an attempt to solve the *PFP* problem.

3 Computational Approaches

The application of the computational techniques to *PFP*, assists considerably to reduce the labor and time complexity burden for a variety of reasons. The prediction process using computational methods can be broadly divided into three classes [65], namely *homology modeling* , *threading* and *ab initio* or, *de novo* folding. The basis of *homology modeling* is that the proteins with similar sequences tend to fold into similar structures. The key challenge here is to perform the best alignment with the template, with the full conformation being built afterwards by the best placement of the side chain. The goal of protein structure prediction by *threading* [40] is to align a protein sequence correctly to a structural model. *Threading* requires choosing both the compatible structural model from a library of models and the alignment from the space of possible sequence-structure alignments. The alignment helps side chain packing and other substructure from the library to help build the primary mapping between sequences versus structure in the model and finally the full-atom model is formed. Both *homology modeling* and protein *threading* have the intrinsic disadvantage that a solved solution for a related structure must exist. In contrast, *ab initio* prediction seeks conformations based solely on the sequence information and optimized energy or fitness function that measures the goodness of a conformation. This approach is based on *Anfinsen's* aforementioned *thermodynamic hypothesis*, such that the final or native conformation of the corresponding sequence is thermodynamically stable and is located at the *global free energy minimum* [78]. In considering these three categories, protein *threading* can be viewed as an intermediate technique between homology modeling and *ab initio* prediction.

While *ab initio* prediction is computationally intensive, the potential it affords in terms of accuracy and usability are high in a *PFP* context. It enables the adding or removing of functions in existing proteins to change their structure and is able to synthesize new proteins to obtain desired functions (i.e. inverse prediction), with no need to have a template or dataset available from proteins that have been explored previously for the *ab initio* approach. Moreover, dataset or template does not guarantee the prediction of either a non-relevant or an entirely new structure.

3.1 Molecular Dynamics

In principle, computation based on *molecular dynamics*(MD) [20, 61, 66] is the ideal option and most realistic way to obtain the minimal energy conformation from the collaborative motion and energy of the molecules in a protein sequence. Its basis is *Newton's second law* of motion, expressed as:

$$F = ma = m\frac{d^2x}{dt^2} = -\frac{dV}{dx} \tag{2}$$

where, dV = change of Velocity i.e. the potential energy, dx = change of position, F = Force, m = mass, a = acceleration and t = time. The motion of atoms in a molecule and their potential energy (E_{Tot}) is the measure for determining the condition of any state. The potential energy can be divided into bonded and non-bonded and can be expressed by the following set of equations:

$$E_{Tot} = E_{bonded} + E_{non-bonded} \tag{3}$$

$$E_{bonbed} = E_{bond-stretch} + E_{angle-bond} + E_{bond-rotate} \tag{4}$$

$$E_{non-bonded} = E_{Van-der-Waals} + E_{electrostatic} \tag{5}$$

Equation (4) measures the most significant potential energies of the bonded atoms in the form, where a bond can either be stretched like spring or, angular bending occurs and the rotational energy of the two bonded atoms toward the connecting axis. Equation (5) measures the major energy interaction amongst those atoms that are not bonded, such as Van der Waals and *electrostatic* forces. Based on the side chain ("R") property, one of the most dominant forces is *hydrophobic* (also known as *water-hating*) at the composite level with respect to a solvent. This works on some amino acids helping to form the protein core. Conversely, *hydrophilic* (also known as *water-loving*) force works on some other amino acids makes them more attractive to a solvent. This leads to the important phenomenon known as *hydrophobicity.* Further, *hydrogen bonding*, *disulfide bridge* and so on, try to influence the native conformation in their favour.

MD simulates the movements of each atom in the protein and in the surrounding water molecules as a function of time. The system is given an initial thermal energy and atoms are allowed to move according to the rules of classical mechanics. The energy of a conformation (using an empirical energy function) is differentiated to obtain the force, acceleration and velocity of each atom, which is clearly very computationally expensive and requires the fastest possible super-computer, with IBM's *blue gene* [1, 2] project being one approach [3, 24, 64]. With enormous peta-flop computing capability, (10^{15} floating point operation per second), simulation of 100μs of actual protein folding time is estimated to be taking about three years. In order to make the movement realistic, atoms can move only for a very short period of time (typically (10^{-15} seconds) before the energy of the system must be re-calculated. Folding time of approximately 10^{-4} seconds require 10^{11} *MD* time steps, so clearly the computational power is still many orders of magnitude below with respect to what is required to model the real folding process. However, the very short time period for which the current simulations can be run does not allow direct confirmation for their ability to converge [71] to the native conformation from a significantly different starting state, so to achieve the very ambitious goal, *blue gene* cannot go it alone; it is essential to collaborate with the broader research communities [2, 51] to achieve this advancement.

3.2 Model Based Approaches

A near real but complicated approach such as *MD* is infeasible because the computation time is *asymptotic* in nature, so there is still a very long way to go to unravel the complex folding mechanism. Philosophically, this mandates a more bottom-up strategy, which attempts to model the prediction using simplified low-resolution paradigms, before extending it to increasingly high-resolution models to achieve evermore realistic prediction. A robust theoretical framework can be raised in the manner of building blocks. To endeavor this mysterious probe, various levels of models with various resolution are used [19], which is presented chronologically in Fig. 6.

The most simplified model is the lattice model. The lattice can be of several regular structures such as square or cubic, triangular or face-centered-cube (FCC) [6], with dimension 2 to 3. There are 14 regular 3D lattice commonly available, called *Bravais Lattices* in general. With the next level of complexity, the off-lattice (also known as beaded string) [2] model, adds more degree of freedom by relaxing the lattice restrictions. Both the model are used for approximation of the protein's *backbone* conformation. An amino acid is approximated as a residue or a node in both the models. Since the side-chains are encountered and treated as *united atom* with the core residue (non "R" part) in the above mentioned simplified models, these are also referred as a *united atom* model. These models are extremely useful since initial exhaustive investigation is feasible to some extent using these simplified models and computational time remains reasonable [31, 32]. Hence, these models are useful as being effective test-bed for the application and the advancement of the computational intelligence techniques. At the next level of complexity, side chain is considered individually and fully apart from being united with the core residue, introducing additional degree of freedom due to the side chain torsion angles . The presence of the solvent is also sometimes considered. Next, the *all-atom* model considers all the atoms including solvent atoms, all the forces and facts are encountered and the whole approach goes from low resolution to high resolution. Finally, the finest possible model is the quantum mechanical (QM) which quantifies the protein from extreme fine to the quantum level. The *QM* model may be perceived as impossible without the underpinning chronological development of the effective and efficient strategies and theories having been exercised well and derived from the simpler models. Simplified models are thus of immense importance and are applied to aid the understanding of the folding mechanism [17], allow efficient sampling of conformational space and play a key role in advancing the rigorous theoretical basis and methodologies. When designed properly, the model can give a well-defined global energy minimum that can be calculated analytically. Therefore, in this context we shall confine the focus of subsequent sections to these simplified models. The details of simplified models and the folding prediction approaches using them, are discussed next.

The HP Model

There are broadly two types of lattice model simulation [67] - the Gō model and HP model . Due to its effectiveness, popularity and wide usage in almost all developing computing methodologies, the HP model is selected for the comparative study of computation methodologies. The basis of the HP model, which was originally introduced by Dill [17], is *hydrophobicity*, which is one of the properties that strongly affects folding, based upon which the amino acid residues are split into two groups. *Hydrophobic* (H) or *non-polar*

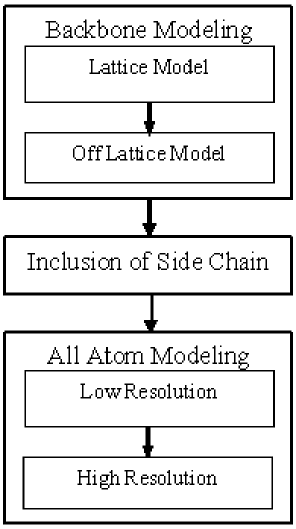


Fig. 6. Models in sequential order of complexity for protein folding prediction.

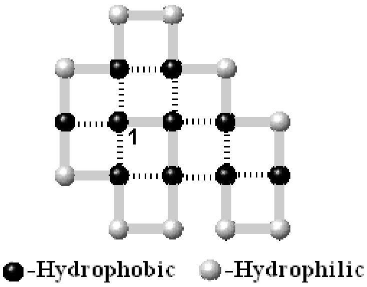


Fig. 7. Conformation in 2D HP model, sequential connection of 20 residues is shown by solid line. Dotted line indicates *TN*. Therefore, fitness = - (*TN* Count) = -10.

residues repel from water [2], and form inside the protein core, while the *hydrophilic* or *polar* (P) residues are attracted towards water and hence tend to remain outside the protein core. The native conformation for a string of amino acids is the one that possesses the lowest energy and is achieved when the numbers of *hydrophobic-hydrophobic* (H-H) pairs, known as *topological neighbours* (TN) (example given in Fig. 7), is maximized. By definition, *TN* is formed by those adjacent *H* pairs that have unit lattice distance from the view point of topological position but which are not sequential with respect to each other in the sequence. The fitness function (F) is then simply defined as:

$$F = \text{-1} \times \text{total number of } TN.$$

The search using this model looks for the valid conformation (i.e. having self-avoiding walk (SAW)) that has the maximum number of *TN*.

Search Procedures using the HP Model

If there are $(n+1)$ amino acids in a sequence then the number of SAW conformations is approximated [60] as:

$$C_n = A\mu^n n^{\gamma-1} \tag{6}$$

Here, μ is the connective constant or effective coordinate number which varies from lattice to lattice, but has a estimated value 4.68401 for HP-like simple lattice model and $A = 1.205$. The universal exponent $\gamma = 43/32$ for a 2D HP model and $\gamma \approx 7/6$ in the case of a 3D model. Enumerating techniques or exhaustive search technique with feasible range are found to be applied approximately upto, $n = 30$. Beyond this limit the search becomes extremely time consuming and hence infeasible. For example, for $n = 40$,

$C_n \approx$ 2 860 274 487 506 831 970 500 921 533 in 2D

and

$C_n \approx$ 1 488 365 480 518 912 276 726 477 968 in 3D,

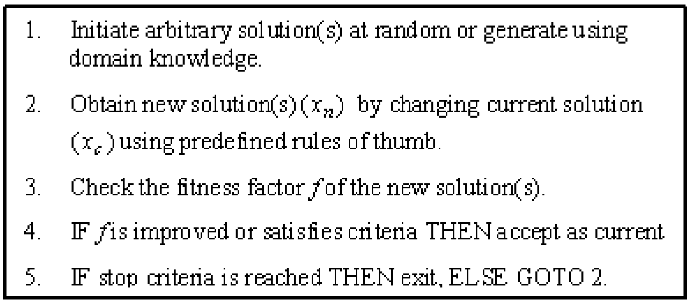
These are extremely large numbers. Searching for the optimal conformation from such an inordinate number is infeasible. Moreover, the number of possible conformations for longer amino acid sequences increases asymptotically. *PFP* strategies include *Artificial Neural Networks* (ANN) [21], *Support Vector Machines* (SVM) [46] and *Bayesian Networks* (BN) [58], while *Hidden Markov Models* (HMMs) [4] which are based on Bayesian learning, have also been used to convert multiple sequence alignment into *position-specific scoring matrices* (PSSM) which are subsequently applied to predict protein structures. The main drawback of *HMM* is that they have to be trained on large sequence sets and they are also unable to identify long distance correlations efficiently between the residues of a sequence which render them unsuitable for *ab initio PFP* applications. *BN* in contrast, perform better than *HMM* in classifying proteins of known structural super-family on amino acid sequences. *HMM* limitations can be overcome somewhat by using *ANNs* in a hybrid architecture [21], although *ANNs* are generally ineffectual for *ab initio PFP* problem because of their inherent dependency on the training set and the reality that information relating to a particular motif may not assist in unravelling the protein folding in different motifs. Regarding deterministic approaches to the *PFP* problem, *approximation* algorithms [29, 50, 57] provide a good theoretical insight, though they are not particularly useful in identifying minimum energy conformations [43], and while *Linear Programming* (LP) [11, 47, 52] methods have been used for protein threading, they have not been applied in *ab initio* applications, with the recent LP focus [12] being confined to approximating the upper bound of the fitness value based on sequence patterns only. This has meant that non-deterministic search approaches have dominated attempts to solve the *PFP* problem. Moreover, the prediction has been proven to be *NP-complete* [8, 14] in these models. Clearly, neither a polynomial time algorithm nor an exhaustive search [6, 13, 28, 60] is feasible for practical amino acid sequence lengths, which are typically 100 or more, so non-deterministic search techniques have become very important.

There are many non-deterministic search approaches for solving the *PFP* problem [49], including *Hill Climbing* (HC), *Simulated Annealing*(SA), *Monte Carlo* (MC) and evolutionary algorithms such as Genetic Algorithms (GA). Statistical approaches to *PFP* include *Contact Interaction* (CI) [69] and *Core-directed chain Growth* (CG) [10], though of which are characterized by lower accuracy as the sequence length increases and also by being non-reversible in their search.

Generally because of their simplicity and effectiveness, GA [35, 36, 41, 71, 72] have been widely applied to the *PFP* problem, while a number of MC versions including, evolutionary MC (EMC) [7, 44], the *Tabu Search* with GA (GTB) [39], and *Ant Colony Optimization* (ACO) [63] are also noteworthy, with GA outperforming MC in [72, 73] for instance. A comparative performance analysis of these various techniques is presented next.

Underlying Principle of the Non-Deterministic Search Approaches

Here, we go though the fundamentals of different non-deterministic approaches such as HC, SA and GA to provide a comparison between them. Fig. 8 provides the generic framework for all non-deterministic search approaches. HC for example, starts with a random bit string and then obtains a set of neigh-



1. Initiate arbitrary solution(s) at random or generate using domain knowledge.
2. Obtain new solution(s) (x_n) by changing current solution (x_c) using predefined rules of thumb.
3. Check the fitness factor *f* of the new solution(s).
4. IF *f* is improved or satisfies criteria THEN accept as current
5. IF stop criteria is reached THEN exit, ELSE GOTO 2.

Fig. 8. General principles of non-deterministic search approach.

bouring solutions by single bit flipping of the current solution. Among these new solutions (including the current one) the best is retained as the current solution, with the process being repeated until the stop criteria is met. SA uses the same framework, but differs in its acceptance criteria. When the new solution is not better than the current, it can still accept it based upon some randomly defined criteria (8), so that for example, *step* 4 for SA could be expressed as:

$$x_c \leftarrow x_n, \text{ if } f(x_n) > f(x_c) \tag{7}$$

Otherwise,

$$x_c \leftarrow x_n, \text{ if random}[0,1) < \exp\Big(\frac{f(x_n) - f(x_c)}{T}\Big) \tag{8}$$

Here, f is the fitness function. T is a (symbolic temperature) variable having an initial value and T is gradually decreased at each iteration, often regarded as *cooling* . SA explores more of the solution space compared to HC, with the randomness introduced for selection in (7) and (8) which being regarded as a Monte Carlo (MC) method, with the terms MC and SA sometimes being used interchangeably in the literature [69, 71, 72, 73].

GA differs mainly in *step* 2 of Fig. 8, as they obtain new solutions by mixing them with current solutions using the well-known *crossover* operation (see Fig. 9(a)) and then randomly inverting particular bits in the process called *mutation* (Fig. 9(b)), which normally has a very small occurrence probability. The *crossover* operation enables the GA to perform inherently parallel searches, which is its most distinguishing and powerful feature, thereby making the search stochastic rather than random. The GA optimizes the effort in testing and generating new individuals if their representation permits development of building blocks (*schemata*), a concept formalized in the *Schemata Theorem* [23, 25, 33, 62, 71, 74, 75]. A more detailed explaination of GA and its functionality is provided in the next section.

Insight of Genetic Algorithm

In a GA, an individual is represented by a list of data and instructions (called locus or gene), with the list representing the solution known as a chromosome. The GA commences with either a randomly generated population or uses domain specific knowledge, with traditionally, the solutions being represented as binary strings, though different encoding strategies are possible such as permutation, value and tree encoding [49, 76]. In each generation, the fitness of the entire population is stochastically evaluated by selecting multiple individuals from the current population based on their fitness before crossover is performed to form a new population, which becomes the current population in the next iteration. The i^{th} chromosome C_i is selected based on the fitness f_i with the probability $(f_i \ / \ \bar{f})$, where $\bar{f}$ is the average fitness of the population. Parents then produce off-springs by crossover at a rate p_c for the population of size n, thus forming the next generation. Mutation is applied on the population of generated off-spring at a rate p_m and the selection probability of any off-spring or chromosome is again $(f_i \ / \ \bar{f})$. A small percentage, typically between 5% and 10% of elite chromosomes - those having high fitness factors, are copied to the next generation to retain potential solutions. The remaining chromosomes (if they exist), which are unaffected by *crossover*, *mutation* or *elitism* operations are then moved to the next generation. If an alphabet of cardinality $|A|$ is used for chromosome presentation then the cardinality of schema would be $(|A|+1)$. For example, if two chromosomes [001101] and [101011] consist of an alphabet set $\{0,\ 1\}$ then the schema $[*01**1]$ is represented using alphabet set $\{0,\ 1,\ *\}$, where $*$ is a *don't-care* which is normally applied to cover the unrestricted locus of the schema. The length of the schema $\delta(H)$ is the distance between the position of the first and last non *don't-care* characters, which actually indicates the number of possible crossover positions, so for example $\delta(*01**1) = 4$. For a chromosome length l, there are $\{(|A|+1)^l - 1\}$ possible schema , excluding the one that is comprising of only *don't-cares*, so a population of n chromosomes evaluates up to $[n\{(|A|+1)^l - 1\}]$ schemata, thus making the GA capable of *implicit parallelism* . The order of schema $o(H)$ equals the number of *non-don't-care* characters, so for example $o(*01**11) = 4$, and this governs the impact of mutation upon the schema. The number of occurrences of schema H in a population of size at time t (which is equal to the number of generations) is given by $m(H,t)$, from which the *Schemata Theorem* can be formally written as:

$$m(H,t+1) = m(H,t)\Big(\frac{f(H)}{\bar{f}} \times (1 - p_c\frac{\delta(H)}{l-1}) \times (1-p_m)^{o(H)}\Big) \qquad (9)$$

Thus in GA implementations, the requirement for perfect energy functions [29] are reduced, with the *crossover* operation aiding the construction of global solutions from the cooperative combination of many local substructures . Furthermore, a particular substructure that may be irrelevant for one solution has a reasonable chance of being useful for another solution. In these circumstances, the GA is driven by an implicit parallelism and generates significantly more successful descendants than using a random search. In certain cases, a number of best solutions or chromosomes are copied into the next generation in an *elitism* process that guarantees fitter parents do not disappear due to inferior offspring. While GA performance can be very effective [26, 35, 41, 68, 71, 72] it still does not ensure that the final generation contains an optimal solution. In fact, a GA can frequently become stuck in local minima, a phenomenon that becomes more prevalent as the sequence length increases. While the impact of the stuck condition may not be critical in many application domains, it assumes particular significance in the *PFP* problem, where sequences are normally long and the folding problem intractable.

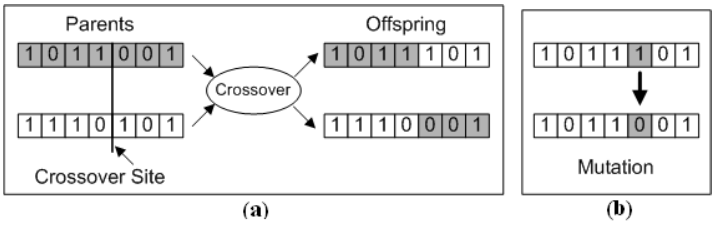


Fig. 9. Example showing (a) 1-point *crossover*, (b) *mutation* by 1 bit flipping.

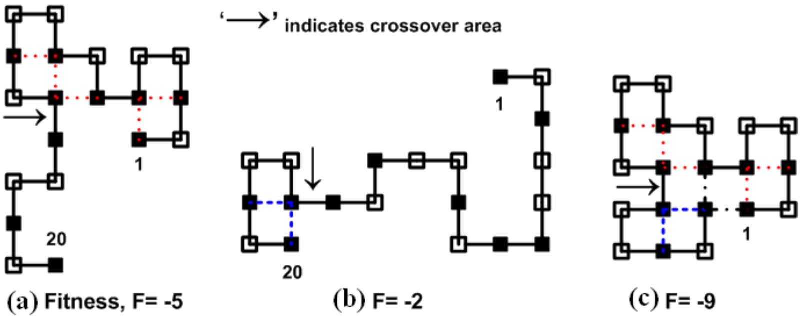


Fig. 10. Example of *crossover* operation. Dotted lines indicate *TN*. Conformations are randomly cut and pasted with the cut point chosen randomly in between residue 14 and 15. The first 14 residues of (a) are joined with the last 6 residues of (b) to form (c), where fitness, F = -9. '■' indicates *hydrophobic* residue and '□' indicates *hydrophilic*.

Hybridization of Genetic Algorithm

Non-deterministic search approaches are still evolving, with GA consistently outperforming all other existing search techniques [49, 72, 73]. In principle, any well performing local search operator can be employed within a GA to generate new solutions, with provision for domain knowledge to also be integrated. This *hybrid* GA (HGA) [15, 30, 48, 82] approach thus combines the power of a GA with the effectiveness of the local optimizer, so having efficiently obtained a potential optimum region, the local optimizer then hones in towards the optimum solution, thereby leading to superior performance.

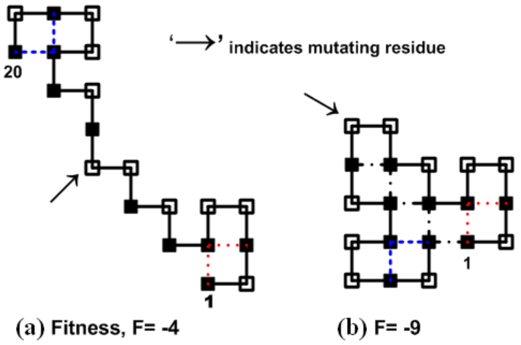


Fig. 11. Example of *mutation* operation. Residue number 11 is chosen randomly as the pivot for move. 180^o rotation alters (a) with F = -4 to (b) F = -9.

To solve the *PFP* problem using the HP model, Unger et al [72] incorporated the selection criteria in equation (7) and (8) of the SA within the GA to outperform all the MC variants, though this *HGA* technique required more computational power to predict folding in longer sequences. Hoque et al. [35, 36] further enhanced the performance of the GA by adapting domain knowledge into *PFP* applications. Some of these strategies are now reviewed in greater detail.

For presenting the solution or chromosome of GA population, Unger and Moult [72] used conformation itself shown in Fig. 10 and Fig. 11 with operations, instead of encoding such as binary encoding. By the nature of the solution of this *PFP* problem, while searching for the optimum conformation, the phenotype of the chromosome, i.e. the conformation corresponding to the solution becomes compact. Therefore, *crossover* and *mutation* both becomes victims of collision increasing more often in producing invalid (i.e. non-SAW) conformation. Therefore, it becomes increasingly harder to get optimal from sub-optimum compared to getting a sub-optimal from random or initial conformation. As a consequence of increasing collisions for relatively long sequences, the prediction fails to get the optimum solution at a very early stage. In the context of *schemata theorem* (9), the *crossover* effectively becomes $p_c \approx 0$, so as the *mutation* $p_m \approx 0$, hence equation (9) becomes:

$$m(H, t+1) = m(H, t)\Big(\frac{f(H)}{\bar{f}}\Big) \tag{10}$$

This indicates that without meaningful *crossover* and *mutation* effects taking place, there will be no variation in the chromosome population and the entire search process becomes stagnant.

An operator that can move the intended part of the compact conformation without disturbing the stable portion unnecessarily is certainly promising, with one such operator being the *pull move* proposed by Lesh [43]. Hoque et al. [35, 36] subsequently introduced two additional move operators, namely *diagonal move* and *tilt move* together with the *pull move*. Less destructive moves are given first preference during implementation as shown in Fig. 12.

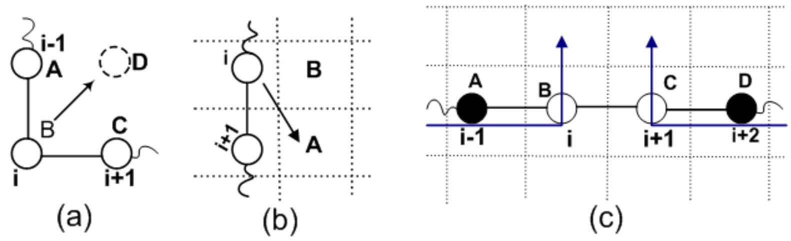


Fig. 12. (a) *Diagonal move* - node at B can be shifted to position D (provided D is unoccupied), which does not alter the position of any other residue. (b) *Pull move* operation. As a precondition place A and B need to be freed or B is permitted to be preoccupied by $(i-1)^{th}$ node. (c) *Tilt move*, with the arrows indicating the moves.

If the covalent bonded two neighbouring nodes are diagonally positioned with respect to each other, then the *diagonal move* shifts the said node obliquely, provided new position is not already occupied. *Pull moves* are *diagonal moves*, where at least two residues are moved. In Fig. 12(b), prior to *pull move*, if $(i-1)^{th}$ residue is already at position B, then the pull move would be a diagonal move. Pulling can occur in either direction towards the *first* or *last* residue. In the *tilt move*, any two connected residue by straight line move together to two free locations (and intermediate residue if any need free location as well), unit lattice distance apart, with the connecting line of those residues being parallel to previous positions. The pull for this move progresses to both ends by dragging all the residues. *Diagonal move* is less destructive in the sense that it only moves one residue. *Pull move* operates on at least two or more residues and stops as soon as a valid conformation is achieved. Although *tilt move* moves all the residues, it is very effective in a congested situation where *pull move* or *diagonal move* does not fulfil the pre-conditions. Lesh's experiments show the search using *pull move* is able to explore optimum [43] conformation even for longer sequences, but it also consumes very high computational resources if applied arbitrarily, which may not be encouraging for regular implementation. Incorporating domain knowledge is therefore an attractive option in attempting to improve the usage of these various residual moves.

With this in mind, Hoque et al, [35] introduced various strategies to embed domain knowledge to guide the GA via the *guided* GA (GGA). The *H*s in an optimum conformation in *HP* model, form a core due to hydrophobic forces, while the Ps exhibit an affinity with the solvent and so tend to migrate to the outer surface, so the final conformation can be conceptualized as shown in Fig. 13.

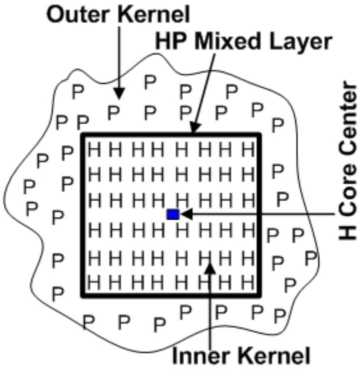


Fig. 13. Metaphorical HP folding kernels. The inner kernel called the *H-Core* is compacted with Hs, while the outer kernel is formed of Ps. In between, a thin layer of *H-P* bonds called the *HP-mixed layer* is formed.

The *HP-mixed layer* crucially maintains the shape of the inner core and has been proved that the optimum shape of the inner core is a square (in 2D) or cube (in 3D) when the HP lattice model is used. While the search procedure proceeds towards an optimum conformation, the *H-Core* forms spontaneously [22, 80, 81], which does not necessarily place all the Hs in best position in order to achieve the optimum conformation, which is reflected at the *H-Core* boundary by the shape of the *HP mixed layer*. This means some TNs get out of the core, with those misplaced Hs becoming immediately bonded with Ps. This observation provided the motivation for Hoque et al. [35, 36] to explore some new strategies to overcome the problem.

A finite set of sub-sequences of the *HP-mixed layer*, corresponding to the most probable sub-conformations is constructed as shown in Fig. 14 and Fig. 15. Two broad categories of sub-sequences are defined; gS_H and gS_P, where $g \in \aleph$ ($\aleph$ is the natural number).These two categories completely cover the *HP mixed layer* including outer kernel. Let S_H and S_P represent segments of H and P respectively. A segment refers to a contiguous string of length g, so $3S_H$ for example means *-PHHHP-*, i.e. $g = 3$ with the two boundary residues being of the opposite type of the run. g is divided into even (g_e) and odd (g_o) numbers. For $g_o > 1$, the category g_oS_P is split into $g_oS_{P\phi}$ and g_oS_{Px}, where $x\epsilon\{1, 2, 3\}$ which implies the run of P is bounded by an additional H at the left ($x = 1$), right ($x = 2$) or both ($x = 3$) sides, while ϕ indicates no additional H, so $3S_{P3}$ means a sub-sequence *-HHPPPHH-*. Collectively, these will be called as *H-Core Boundary Builder Segments* (HBBS) and are mapped

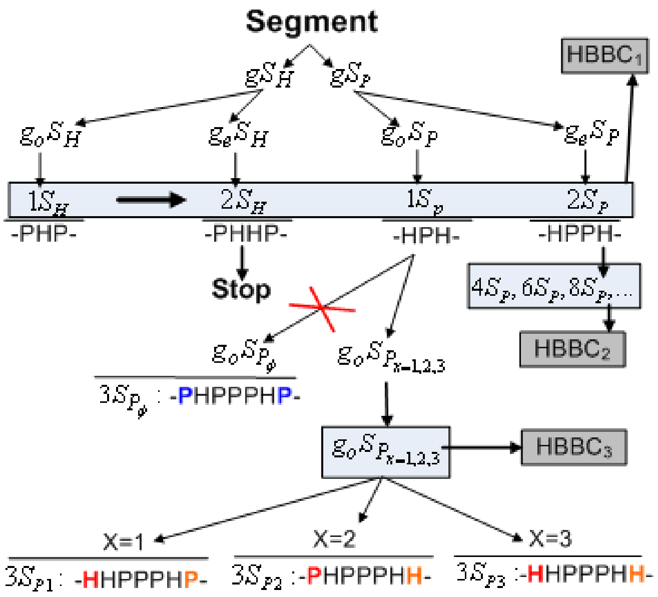


Fig. 14. Highly probable sub-sequences for the *HP mixed layer*.

to potential sub-conformations which are referred to as *H-Core Boundary Builder sub-Conformation* (HBBC). Sub-conformations that are very likely are chosen with properties such as, H is put towards HCC and P is away, or the two Hs contributing TN are encouraged with position towards HCC as well. According to their similarity and importance, the sub-conformations are grouped as $HBBC_1$, $HBBC_2$ and $HBBC_3$ as indicated in Fig. 14, where the expansion of $2S_H$ is stopped; otherwise it would involve the H of the inner *H-Core*. No particular sub-conformation is defined for g_oSP_ϕ since it can be taken care of by the sub-sequence $1S_H$.

The fundamental basis of a sub-conformation is to place the H nearer to the *H-Core* and P as far away from the *H-Core* as possible preserving a TN within the sub-conformation if applicable. The objective is thus to ensure that before becoming trapped in local minima, convergence is guided towards a likely optimum conformation using the *H-Core* formation principles. The protein conformation search can be viewed as a concatenation of favorable schemata or sub-structures. A schema in this case is presented as a string of $\{0, 1, 2, *\}$, where 0, 1, 2 may indicate one of three directions *Left*, *Right* and *Forward* (Fig. 16) of the current H with respect to the previous two residues, and '$*$' is a *don't care*, which signifies no particular goal may be assigned to P as the parsing of the schema through the fitness function does not directly reward P bonding. Fitness F is indifferent to where P is positioned and is assumed to be automatically taken care of [72]. However, as the generation converges, the effectiveness of *crossover* and *mutation* (pivot rotation [71])

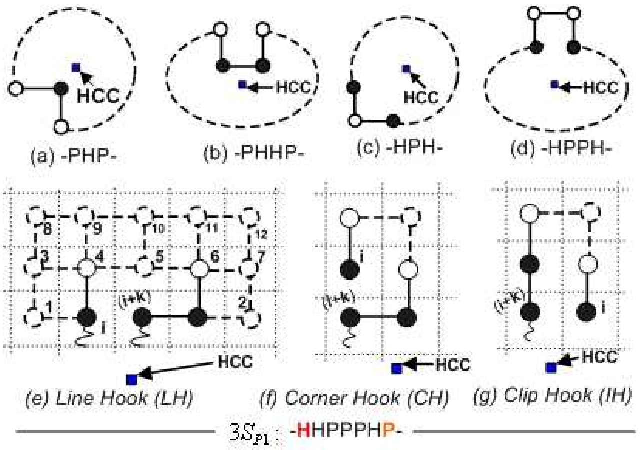


Fig. 15. Highly probable sub-conformations of the corresponding sub-sequences. For (a), (b), (c), (d) respectively correspond to $1S_H$, $2S_H$, $1S_P$, $2S_P$ and (e), (f), (g) are the three possible variations of $3S_{P1}$. The *H-Core Center* (HCC) is the mean value of the coordinate of all the Hs'.

is weakened in *PFP* as the increasingly compact folded structure means the failure of the crossover operation augments the number of self collisions [44]. Furthermore, without a complex sequence of *mutations*, there will often be invalid conformations due to collisions within the compact conformation, so during the search, there are fewer options and less potential in the population to replace the near-optimal with the optimal solution. The *move* operators and their associated domain knowledge used to implement the *HBBC* mappings assist at this stage. With the *H-Core* formation focusing on those Ps that are covalent bonded with Hs, a sub-conformation (HBBC) is temporarily enforced to replace *don't care* $(*)$ with one from $\{0, 1, 2\}$ - whichever is most likely for positioning P. Those Ps covalently bonded with Hs need to be placed in such a way that they either remain (approximately) on the opposite side of the H with respect to the developing *HCC* or outside the *H-Core*. Using this approach there will be a greater likelihood that a part of the proper cavity formed by *HP mixed layer* survives and eventually forms the optimal conformation with maximal $|F|$.

The mapping however, is hard to implement directly as the fitness function F changes, so a *probabilistic constrained fitness* (PCF) function is proposed that rewards the desired mapping of a sub-conformation in the *HP-mixed layer*, if it exists, otherwise penalizes the mapping according to Table 1. Since the corresponding sub-conformations are highly probable, *PCF* as its name suggests therefore applies multi-objective fitness constraints to F. Clearly F and *PCF* cannot be directly combined by summing, so a strategy has been developed to obtain the total fitness (TF) as:

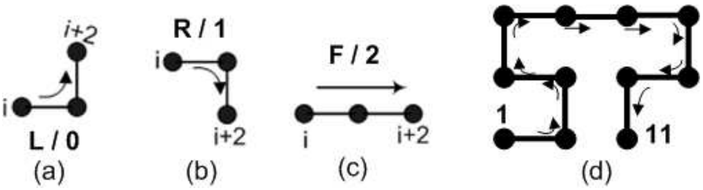


Fig. 16. (a) to (c) indicate directions *Left*, *Right* and *Forward* respectively. (d) Conformation example representing *LLRRFFRRL*, with the corresponding encoded sequence *001122110*.

Table 1. *PCF* value calculation.

HBBC	Reward	Penalty
$HBBC_1$	-1	1
$HBBC_2$	$\left(-\frac{2}{g_e}\right)$	$\left(\frac{2}{g_e}\right)$
$HBBC_3$	$\left(-\frac{2}{(g_o-1)}\right)$	$\left(\frac{2}{(g_o-1)}\right)$

$$TF = \alpha(t) * F + \beta(t) * PCF \tag{11}$$

where t is the number of generations, α and β are time-varying positive weightings whose values are chosen by considering two alternate phases for each generation pass, namely a positive and a negative phase. In the former, α varies with $\alpha > \beta$ while in the latter β varies with $\alpha < \beta$. A sub-conformation is enforced whenever $\alpha < \beta$, so PCF dominates over F to force the change. To vary the two weights α and β alternatively, the following oscillatory (*swing*) function is applied:

$$\delta(t) = A(1 + \cos \omega_m t) \cos \omega_0 t \tag{12}$$

where $\omega_m << \omega_0$. The assignment of α and β is are in (13) to (15):

$$\text{Phase 1} : \alpha(t) = \delta(t),\ \beta(t) = 1,\ \text{if } \delta(t) > 0 \tag{13}$$

Otherwise,

$$\text{Phase 2} : \alpha(t) = 1,\ \beta(t) = -\delta(t),\ \text{if } \delta(t) < 0 \tag{14}$$

Otherwise,

$$\text{Transient Phase} : \alpha(t) = 1,\ \beta(t) = 1 \tag{15}$$

There are assignments of *1*s in equations (13) through to (15). This arrangement is to preserve the already achieved partial conformation by using the less dominant fitness in the early phase. The oscillatory nature of the function switches the dominating role of F and PCF in a non-monotonous manner which is not destructive to the achieved stability, but at the same time the variations cover the best combinatorial dominance, which is hard to predict. To institutively understand why this works in the context of *schemata theorem*, consider, the positive phase (i.e. F is dominating over PCF), a favorable schema had fitness f_t (at time t), with the highly probable sub-conformation enforcement, those *TN*s that resist or contradict the enforcement, in the worst case, are broken and get fitness f_{t+k} where $|f_{t+k}| < |f_t|$ and k is any positive constant. After a number of generations, when $\alpha < \beta$ situation turns into $\alpha > \beta$ and F predominates over *PCF*, say the fitness of the schema becomes f_{t+k+r}, where r is another positive constant. It is very likely that $|f_{t+k+r}| > |f_{t+k}|$ and if the enforcement is adopted then it is expected that $|f_{t+k+r}| > |f_t|$, otherwise, the schema is destroyed with exponential decay. In this way, all likely sub-conformations are selected randomly and eventually this will lead toward a proper cavity being formed which has a maximal $|F|$. If conversely a sub-conformation is reinforced during the negative phase, it will break contradictory all *TN*s which we have tried to keep to a minimum in the strategy to help reform the conformation. If the sub-conformation is inappropriate (which is unlikely) it will disappear in the positive phase with the reinforcement of *TN* formations, otherwise, it will help escaping from becoming stuck in a local minima. In practice, even in a positive phase, sub-conformations are reinforced if convergence is slow, to escape local minima, with Fig. 17 illustrating the effect of this arrangement as the search progresses.

Table 2. Comparison of the performance of non-deterministic search approaches.

Length/Sequence	GGA	GTB	EMC	GA	MC	CI
20/ (HP)2PH(HP)2(PH)2HP(PH)2	-9	-9	-9	-9	-9	-9
24/ H2P2HP2HP2(HPP)4H2	-9	-9	-9	-9	-9	-9
25/ P2HP2H2P4H2P4H2P4H2	-8	-8	-8	-8	-8	-8
36/P3(H2P2)2P2H7P2H2P4H2P2HP2	-14	-14	-14	-12	-13	-14
48/(P2H)2(HP2)2P4H10P6(H2P2)2HP2H5	-23	-23	-23	-22	-20	-23
50/H2(PH)3PH4PH(P3H)2P4H(P3H)2PH4P(HP)3H2	-21	-21	-21	-21	-21	-21
60/P2H3PH8P3H10PHP3H12P4H6PH2PHP	-36	-35	-35	-34	-33	-35
64/H12(PH)2((P2H2)2P2H)3PHPH12	-42	-39	-39	-37	-35	-40

The overall performance of the new *hybrid GGA* approach is very impressive, outperforming the other nondeterministic search approaches based on the series of well established benchmark sequences given in Table 2.

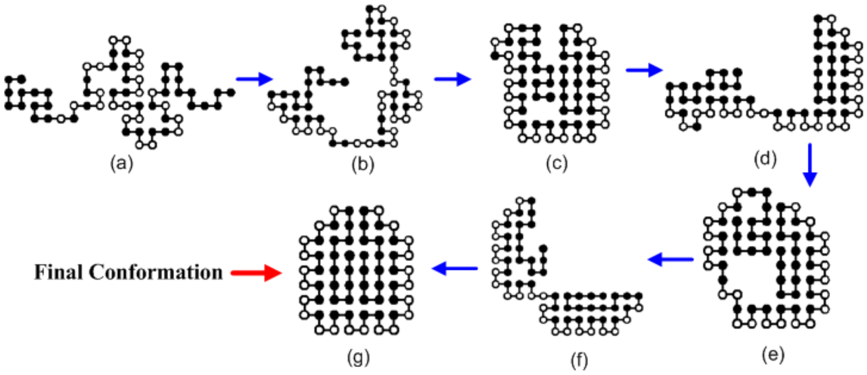


Fig. 17. (a) to (g) shows instances at generation 1, 14, 336, 426, 1434, 5396, 5646 respectively, where in (d) and (f), *PCF* is dominating over *F*. For the *swing* function; A=30, $\omega_m = 0.005$, $\omega_0 = 0.5$.

GA computation can be speeded up in a number of ways. A simple policy is to minimize the computational load of the frequently computed fitness function after each *crossover* and *mutation*. The fitness of the offspring created after *crossover* can be computed faster by partially sharing the already computed fitness of the parents. Similar optimization can be applied to *mutation* which is demonstrated in details by Hoque et al [34].

Other Non deterministic Approaches

To further speed up computation, the core can be separately formed by considering only Hs. A chain comprising only Hs will form the core very quickly using a GA or any other core formation approach, like the *core-directed chain growth* (CG) [10]. Speeding up however does not make much difference to predictability, as the real conundrum is that the sequence has other components together with Hs (i.e. the Ps). It was claimed that CG forms the optimal core (which is a rectangle and cuboid for a 2D square and 3D cube lattice respectively) by firstly counting the number of Hs in the sequence and then use this as a kind of guideline. If this is so, then a library containing optimal core of various size can then be employed to provide even greater speedup. But, this is not happening because, for embedding rest of the parts, exhaustive enumeration is applied, which claimed to guarantee complete search of the all possibilities. This indeed may be feasible for short sequences, but equation (5) clearly reveals the infeasibility for a typical sequence length in general. In CG approaches, the actual power lies within the heuristic fitness function and further within *look-ahead* procedure, though these benefits become blurred with increasing sequence length. The strategy is also likely to fail even for short sequences, when the core needs to have a twist for better fitness as shown in

Fig. 18. Specialized procedures based on some assumptions can show speedup, but thereby also becomes vulnerable in missing the optimum conformation.

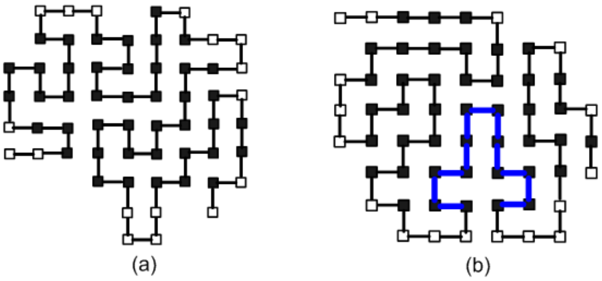


Fig. 18. Sequence length 60 (Table 1), (a) having fitness -35 can be detected by CG, but (b) fitness -36 having twist (indicated by thick (blue) line) is not detected by CG.

An alternative statistical approach is *Contact Interaction* (CI) , which is regarded as improved MC (similar to (7) and (8)) by concept of cooperativity introduced in [69, 70] deriving from non-local interaction. The criteria of accepting the new conformations generated during simulations, are not based on the energy of the entire molecule, but *cooling* factors associated with each residue define regions of the model protein with higher and lower mobility (Fig. 19). CI randomly moves the residue based on MC but with additional constraints upon the *TN* formation, loop will have low mobility and embedded loop will reduce the mobility further, which complies with the cooperatively concept.

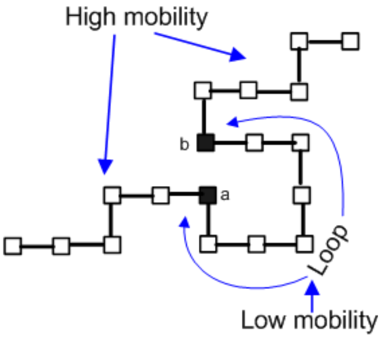


Fig. 19. *Hydrophobic* residues ‘a’ and ‘b’ are forming *TN*. Due to *TN* (indicated as loop), it is considered as having low mobility.

The drawback of CI is that it involves random rather than stochastic moves. Residues forming *TN*s are regarded *sticky* as they have low mobility, while they can provide fast convergence, they possess no technique for back tracking from the wrong solution. This is especially prevalent in long sequences, where increasing the level of embedding can worsen the required back tracking and hence the prediction. The performance of CI has also been compared with other methods in Table 2.

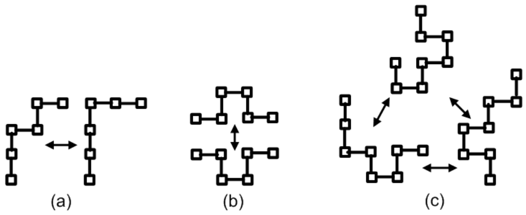


Fig. 20. (a) A three-bead flip (b) Crankshaft moves (c) Rigid rotations; are incorporated within *mutation* operation.

A variation of MC named as new MC algorithm [7], applied as a variant of the *pruned-enriched Rosenbluth method* (PERM)[83] that is basically a chain growth algorithm on the *Rosenbluth-Rosenbluth* (RR) [84]. The residues are placed to an unoccupied position based on some probability distribution, which finally leads to weighted sample. Further, pruning conformation with low weight and enriching high-weighted conformations are done stochastically, which is basically the population based cut and paste (i.e. the *crossover*) operation with a view to achieve higher fitness. This approach is basically combining the effective part of a number of existing systems and thus improves a bit for some cases and not reasoning why it should perform better and further not relating any domain knowledge as well. Therefore, it is not reliable and also does not get the putative ground problem for longer sequences of the problem set.

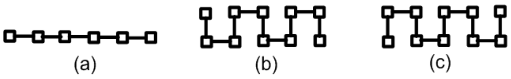


Fig. 21. The constrained secondary structure (a) extended sheet (b) helix with direction 1 (c) helix with direction 2.

Another new MC algorithm namely the evolutionary Monte Carlo (EMC) [44] algorithm developed by incorporating optimization of genetic algorithm , showed improved performance further. *EMC* works by simulating a population of Markov chains , where a different temperature is attached to each chain. The population is updated by *crossover*, *mutation* and *exchange* operation that preserve the *Boltzmann* distributation of the population. It is reported to incorporate the extensive search ability of GA using *crossover* operation and also fast mixing ability of simulated tempering by simulating along a temperature ladder. It incorporated the three different move with the *mutation* such as a *three-bead flip*, *crankshaft move* and *rigid rotations* (Fig. 20). The overall approach is basically the modified version of Unger's GA [72], which is using all the properties of a GA such as *population*, *crossover*, *mutation* and *roulette wheel* selection using *Boltzmann* weight in this case. Again it performs a bit improved respect to Unger's GA but does not get the putative ground energy state for the longer sequences. To improve its prediction, constraints are assigned using *secondary* structure in protein folding such as shown in Fig. 21. But the incorporation of such *secondary* structure has potential risk which can easily miss the putative ground energy state and that is clearly shown in [43].

It is reasonable to surmise that the GA produces so many samples by *crossover* stochastically and by *mutation* randomly (usually set at a low rate) without tracking of chromosome that might be reproduced. From equation (9), it is concluded that favorable schemata are highly likely and survive exponentially; therefore similarity will grow having very high change of producing same chromosome repeatedly. So, memorizing the existing chromosome, the repetition can be subsided. Therefore, GA hybridization with *tabu* search (GTB) could be a potential candidate for the *PFP* problem. *Tabu* search is a local search technique, which enhances the performance of a local search method by using memory structures. Jiang et al. [39] applied the GA with *tabu* search for the 2D HP *PFP* sequence. This procedure enlists dissimilar solution rejecting duplicates or closely similar chromosomes. It performed well to some extent, but again, according to equation (6), the number of possible samples is an inordinate number, therefore, memory requirement tends to be infinite for longer sequence and performance decreases with increasing length. Lesh [43] also incorporated *tabu* search with the *pull move* and indicated that it was a resource intensive problem. Hence, incorporation of *tabu* search within GA, i.e the *GTB* is not promising. Finally, the *HGA* designed and developed by Hoque, et al. [35, 36] removes these problems effectively and efficiently.

Therefore, it can be argued that any approach that is unable to withstand the scaling of the sequence is not promising in the context of trying to solve the *ab initio PFP* problem. On the other hand, *crossover* as the main operator in GA, does not suffer from scaling problems, which makes it capable of locating the optimum region effectively before a local optimizer is employed to complete the prediction process efficiently. For high performance hybridization therefore, local optimizers need to be designed and developed liberated of any possibility of scaling effects.

4 Conclusions

This chapter has analysed the performance of contemporary *hybrid* evolutionary computing techniques and in particular, the *hybrid genetic algorithm* (HGA) in regard to securing an effective solution to the challenging *ab initio protein folding prediction* (PFP) problem. This approach has been proven to be sufficiently robust to withstand the scaling of *PFP* sequences and also to locate optimum solution regions, which subsequently allow for the incorporation of a local optimizer to converge to improved solutions. Integrating additional domain knowledge exhibited considerable promise in the *HGA*, with coarse-grained approach providing a strong theoretical framework for these comparatively simple *PFP*-based models.

References

1. Adiga N, et al (2002) An Overview of the BlueGene/L Supercomputer, Supercomputing, ACM/IEEE, pp. 1-22, 0-7695-1524-X/02.
2. Allen, et al (2001) Blue Gene: A vision for protein science using a petaflop supercomputer, IBM System Journal, 40(2).
3. Almasi G, et al (2005) Early Experience with Scientific Applications on the Blue Gene/L Supercomputer, LNCS, Parallel Processing: 11th International Euro-Par Conference, Lisbon, Portugal, 560-570.
4. Anekonda T S (2002) Artificial Neural Networks and Hidden Markov Models for Predicting the Protein Structures: The Secondary Structure Prediction in Caspases, Computational Molecular Biology.
5. Anfinsen C (2006) Biography, http://nobelprize.org/chemistry/laureates /1972/anfinsen-bio.html, March.
6. Backofen R, Will S (2005) A Constraint-Based Approach to Fast and Exact Structure Prediction in Three-Dimensional Protein Models, Kluwer Academic Publishers.
7. Bastolla U, et al (1998) Testing a new Monte Carlo Algorithm for Protein Folding, National Center for Biotechnology Information, 32(1): 52-66.
8. Berger B, Leighton, T (1998) Protein Folding in the Hydrophobic-Hydrophilic (HP) Model is NP-Complete, Journal of Computational Biology, 5 (1): 27-40.
9. Berg M M, Tymoczko J L, Stryer L (2002) Biochemistry, 5th edition, Edit Freeman W H and Company.
10. Bornberg-Bauer (1997) Chain Growth Algorithms for HP-Type Lattice Proteins, RECOMB, Santa Fe, NM, USA.
11. Brown D, et. al. (2005) Bioinformatics Group, School of Computer Science, University of Waterloo Canada, http://monod.uwaterloo.ca/, April .
12. Carr R, Hart W E, Newman A (2004) Bounding A Proteins Free Energy In Lattice Models Via Linear Programming, RECOMB.
13. Chen M, Lin K Y (2002) Universal amplitude ratios for three-dimensional self-avoiding walks, Journal of Physics, 35: 1501-1508
14. Crescenzi P, et al. (1998) On the complexity of protein folding (extended abstract), ACM, Proceedings of the second annual international conference on Computational molecular biology, 597-603.

15. Davis L, (1991) Handbook of Genetic Algorithm, VNR, New York.
16. Levinthal C (1969) How to fold graciously. In Mssbauer Spectroscopy in Biological Systems, Proceedings of a Meeting Held at Allerton House, Monticello, Illinois, edited by DeBrunner, J. T. P. and Munck, E., University of Illinois Press, 2224.
17. Dill K A (1985) Theory for the Folding and Stability of Globular Proteins, Biochemistry, 24(6): 1501-1509.
18. Docking (2005) www.cmpharm.ucsf.edu/ and www.scripps.edu/mb/olson/doc/autodock/, February.
19. Duan Y, Kollman P A (2001) Computational protein folding: From lattice to all-atom, IBM Systems Journal, 40(2), 2001.
20. Ercolessi, F (1997) A molecular dynamics primer, Spring College in Computational Physics, ICTP, Trieste.
21. Executive Summary (2005) Feasibility of an Artificial Neural Network Approach to Solving the Protein Folding Problem, http://www.ecf.utoronto.ca/ writing/esc300/pdf/draft5.pdf, January.
22. Flebig K M, Dill K A (1993) Protein Core Assembly Processes, J. Chem. Phys., 98 (4): 3475-3487.
23. Fogel D B, (2000) EVOLUTIONARY COMPUTATION Towards a new philosophy of Machine Intelligence, Second edition, IEEE Press.
24. Germain R S, et al (2005) Blue Matter on Blue Gene/L: Massively Parallel Computation for Bio-molecular Simulation, ACM.
25. Goldberg D E (1989) Genetic Algorithm Search, Optimization, and Machine Learning, Addison-Wesley Publishing Company.
26. Greenwood G W, Shin J (2003) On the Evolutionary Search for Solutions to the Protein Folding problem, chapter 6 in Evolutionary Computation in Bioinformatics, Editors Fogel G B, Corne D W, Elsevier Science (USA), ISBN: 1-55860-797-8.
27. Guex N and Peitsch M C (2006): http://swissmodel.expasy.org/course/course-index.htm, March.
28. Guttmann A J (2005) Self-avoiding walks in constrained and random geometries: Series studies. In Statistics of Linear Polymers in Disordered Media ed. Chakrabarti B K, Elsevier, 59-101.
29. Hart E W, Istrail S (1995) Fast Protein Folding in the Hydrophobic-hydrophilic Model Within Three-eights of Optimal, ACM.
30. Haupt R L, Haupt S E (2004) Practical Genetic Algorithms, 2nd Edition, ISBN 0-471-45565-2.
31. Head-Gordon T, Wooley J C (2001) Computational challenges in structural and functional genomics, IBM Systems Journal, 40(2).
32. Head-Gordon T, Brown S (2003) Minimalist models for protein folding and design, Current Opinion in Structural Biology, 12: 160-167.
33. Holland J H (1992) Adaptation in Natural And Artificial Systems, The MIT Press, Cambridge, Massachusetts London, England.
34. Hoque M T, Chetty M, Dooley L S (2004) Partially Computed Fitness Function Based Genetic Algorithm for Hydrophobic-Hydrophilic Model. HIS: 291-296, ISBN 0-7695-2291-2.
35. Hoque M T, Chetty M, Dooley L S (2005) A New Guided Genetic Algorithm for 2D Hydrophobic-Hydrophilic Model to Predict Protein Folding, IEEE Congress on Evolutionary Computation (CEC), 259-266, Edinburgh.

36. Hoque M T, Chetty M, Dooley L S (2006) A Guided Genetic Algorithm for Protein Folding Prediction Using 3D Hydrophobic-Hydrophilic Model, IEEE WCCI, 8103-8110.
37. Howard-Spink S (2006) The power of proteins,
www.research.ibm.com/thinkresearch/pages/2001/20011105_protein.shtml, February.
38. Irbck A, Troein C (2002) Enumerating Designing Sequences in the HP Model, Journal of Biological Physics, 28: 1-15.
39. Jiang T, et al. (2003) Protein folding simulation of the hydrophobic-hydrophilic model by computing tabu search with genetic algorithms, Journal of Chemical Physics, 119(8).
40. Jones D T, Miller R T, Thornton J M (1995) Successful protein fold recognition by optimal sequence threading validated by rigorous blind testing. Proteins, 23:387-397.
41. Knig R, Dandekar T (1999) Refined Genetic Algorithm Simulation to Model Proteins, Journal of Molecular Modeling.
42. Kuwajima K, Arai M (1999) Old and New Views of Protein Folding, ELESEVIER.
43. Lesh N, Mitzenmacher M, Whitesides S (2003) A Complete and Effective Move Set for Simplified Protein Folding, RECOMB, Berlin.
44. Liang F, Wong W H (2001) Evolutionary Monte Carlo for protein folding simulations, J. Chem. Phys., 115 (7): 3374-3380.
45. MacDonald D, Joseph S, Hunter, D L, Moseley, L L, Jan N and Guttmann A J (2000) Self-avoiding walks on the simple cubic lattice, J. Phys. A: Math. Gen., 33: 5973-5983.
46. Markowetz F, Edler L, Vingron M (2003) Support Vector Machines for Protein Fold Class Prediction, Biometrical Journal, 45(3): 377389.
47. Meller J and Elber R (2001) Linear programming Optimization and a Double Statistical Filter for Protein Threading Protocols, PROTEINS: Structure, Function, and Genetics, 45: 241-261.
48. Merkle L D, Gaulke R L, Lamont G B (1996) Hybrid Genetic Algorithm for Polypeptide Energy Minimization, ACM.
49. Michalewicz Z (1992) Genetic Algorithms + Data Structures = Evolution Programs, New York: Springer-Verlag.
50. Newman A (2002) A new algorithm for protein folding in the HP model, Proceedings of the thirteenth annual ACM-SIAM symposium on Discrete Algorithms.
51. Pande V S, et al (2003) Atomistic Protein Folding Simulation on the Submillisecond Time Scale Using Worldwide Distributed Computing, Biopolymers, 68: 91-109.
52. Panik M J (1996) Linear Programming: Mathematics, Theory and Algorithm, ISBN 0-7923-3782-4.
53. Petit-Zeman S (2006) Treating protein folding diseases,
www.nature.com/horizon/proteinfolding/background/treating.html, March.
54. Pietzsch J (2006) The importance of protein folding,
www.nature.com/horizon/proteinfolding/background/importance.html, March.
55. Pietzsch J (2006) Protein folding technology,
www.nature.com/horizon/proteinfolding/background/technology.html, March.

56. Pietzsch J (2006) Protein folding diseases, www.nature.com/horizon/proteinfolding/background/disease.html, March.
57. Rune B L, Christian N S, Pedersen (2005) Protein Folding in the 2D HP model, http://www.brics.dk/RS/99/16/BRICS-RS-99-16.pdf, BRICS, January.
58. Raval A, Ghahramani Z, Wild, D L (2002) A Bayesian network model for protein fold and remote homologue recognition, Bioinformatics, 18(6):788-801.
59. Setubal J, Meidanis J (1997) : Introduction to Computational Molecular Biology, ISBN 0-534-95262-3, An International Thomson Publishing Company.
60. Schiemann R, Bachmann M, Janke W (2005) Exact Enumeration of Three Dimensional Lattice Proteins, Computer Physics Communications 166: 816 Elsevier Science.
61. Schlick T (2002) Molecular Modeling and Simulation, Springer.
62. Schulze-Kremer S (2006) Genetic Algorithms and Protein Folding, http://www.techfak.uni-bielefeld.de/bcd/Curric/ProtEn/proten.html, March
63. Shmygelska A, Hoos, H H (2005) An ant colony optimization algorithm for the 2D and 3D hydrophobic polar protein folding problem, BMC Bioinformatics, 6(30).
64. Siew N, Fischer D (2001) Convergent evolution of protein structure prediction and computer chess tournaments: CASP, Kasparov, and CAFASP, IBM Systems Journal, 40 (2).
65. Skolnick J, Kolinski A (2001) Computational Studies of Protein Folding, Bioengineering and Biophysics, IEEE.
66. Stote R, et al (2006) Theory of Molecular Dynamics Simulations http://www.ch.embnet.org/MD_tutorial/, March.
67. Thirumalai D, Klimov D K, Dima R I (2001) Insights into specific problems in protein folding using simple concepts, Computational Methods for Protein Folding: Advances in Chemical Physics, vol. 120. Edited by Friesner A. ISBNs:0-471-22442-1.
68. Takahashi, O, Kita, H, Kobayashi, S (1999) Protein Folding by A Hierarchical Genetic Algorithm, 4th Int. Symp. AROB.
69. Toma L, Toma S (1996) Contact interactions methods: A new Algorithm for Protein Folding Simulations, Protein Science, 5 (1): 147-153.
70. Toma, L, Toma S (1999) Folding simulation of protein models on the structure-based cubo-octahedral lattice with the Contact interactions algorithm, Protein Science, 8(1): 196-202.
71. Unger R, Moult, J (1993) On the Applicability of Genetic Algorithms to Protein Folding. Proceeding of the Twenty-Sixth Hawaii International Conference on System Sciences, 1: 715-725.
72. Unger R, Moult, J (1993) Genetic Algorithms for Protein Folding Simulations, Journal of Molecular Biology, 231:75-81.
73. Unger R, Moult J (1993) Genetic Algorithm for 3D Protein Folding Simulations, 5th International Conference on Genetic Algorithms, 581-588.
74. Vose M D (1999) The Simple Genetic Algorithm, The MIT Press, Cambridge, Massachusetts London, England.
75. Whitley D (2001) An Overview of Evolutionary Algorithms, Journal of Information and Software Technology, 43: 817-831.
76. Wikipedia (2006) Genetic Algorithm, http://en.wikipedia.org/wiki/Genetic_algorithm, March.
77. Wikipedia, (2006) Nuclear magnetic resonance, http://en.wikipedia.org/wiki/Nuclear_magnetic_resonance, March.

78. Xia Y, Huang E S, Levitt M, Samudrala R (2000) Ab Initio Construction of Protein Tertiary Structures using a Hierarchical Approach, JMB.
79. Yao, X, (1999) EVOLUTIONARY COMPUTATION Theory and Application, World Scientific.
80. Yue K, Dill K A (1995) Forces of Tertiary Structural Organization in Globular Proteins, Proc Natl Acad Sci USA, 92: 146-150.
81. Yue K, Dill K A (1993) Sequence-Structure relationships in proteins and copolymers, Physical Review E, 48(3): 2267-2278.
82. Zhang X (1994) A Hybrid Algorithm for Determining Protein Structure, IEEE Expert, 9(4): 66 74.
83. Grassbegrer P (1997) Pruned-enriched Rosenbluth method: Simulation of θ polymers of chain length up to 1,000,000. Phy. Rev. E, in press.
84. Rosenbluth M N, Rpsenbluth A W (1955) Monte Carlo calculation of the average extension of molecular chains. J. Chem. Phys. 23:256.

Index