Genetic algorithm-based improved sampling for protein structure prediction

Md Tamjidul Hoque* and Sumaiya Iqbal

Computer Science,
University of New Orleans,
New Orleans, Louisiana, USA
Email: thoque@uno.edu
Email: siqbal1@uno.edu
*Corresponding author

Abstract: The quest for efficient sampling algorithms continues to be a demanding research topic due to their wide spread applications. Here, we present an extension of genetic algorithm (GA) to incorporate improved sampling capacity. We develop a fast-navigating genetic algorithm (FNGA) using associated-memory (AM)-based crossover operation which gives more trials with best chromosomes subpart and helps to navigate faster. To mitigate the increased similarity within population, the twin removal genetic algorithm or TRGA is applied. The optimally diverge chromosomes generated by TRGA can introduce potential subpart to enhance the performance of FNGA further. Thus, we combine FNGA and TRGA and named the combination, kite genetic algorithm (KGA). The proposed FNGA and KGA are empirically tested with benchmark functions and the results are found promising. We further employ KGA in the conformational search for the fragment-free protein tertiary structure prediction. The results of *ab initio* protein structure modelling show that the sampling performance of KGA is competitive.

Keywords: genetic algorithms; fast-navigation; twin removal; associated-memory; protein structure prediction; hard optimisation; *ab initio* prediction; crossover; mutation; chromosome correlation factor.

Reference to this paper should be made as follows: Hoque, M.T. and Iqbal, S. (2017) 'Genetic algorithm-based improved sampling for protein structure prediction', *Int. J. Bio-Inspired Computation*, Vol. 9, No. 3, pp.129–141.

Biographical notes: Md Tamjidul Hoque received his PhD in Information Technology from the Monash University, Australia in 2008. He received his MSc and BSc in Computer Science and Engineering from the Bangladesh University of Engineering and Technology (BUET) in 2002 and 1998, respectively. He is currently an Assistant Professor with the Computer Science Department, University of New Orleans (UNO), New Orleans, LA, USA. He is also the Director of the Bioinformatics and Machine Learning Lab at the UNO.

Sumaiya Iqbal is a Graduate Assistant at the University of New Orleans (UNO) and a member of the Bioinformatics and Machine Learning Lab at the UNO. She is an Assistant Professor, Computer Science and Engineering Department, Bangladesh University of Engineering and Technology.

1 Introduction

Given the amino acid sequence of a protein, the task of protein structure prediction (PSP) is to determine its three-dimensional native structure. Anfinsen's (1973) thermodynamic hypothesis informs that the protein structure can be predicted using the information encoded within the amino acid sequence of that protein. PSP matters because the structure of the protein determines its function and proteins systematise the cellular functions in an organism. Once we know sequence to function relationship, we can determine what to do at the molecular level for our health and wellbeing. Protein can fold into astronomical number of possible structures from its amino acid sequence considering admissible degree of freedom of the constituents. Thus, the search of native conformation within the PSP task is a hard optimisation problem. However, the Levinthal (1968) paradox shows the dual property of protein that it folds in a spontaneous manner in nature. Therefore, it is possible to guide the PSP task by well-defined computational approaches.

There are mainly three computational approaches for predicting structure of protein: comparative modelling, threading or fold recognition, and *ab initio* prediction. The comparative modelling (also known as homology modelling) requires one or more experimental tertiary structures of homologous proteins to be present. Protein fold recognition is a useful alternative to understand structures of

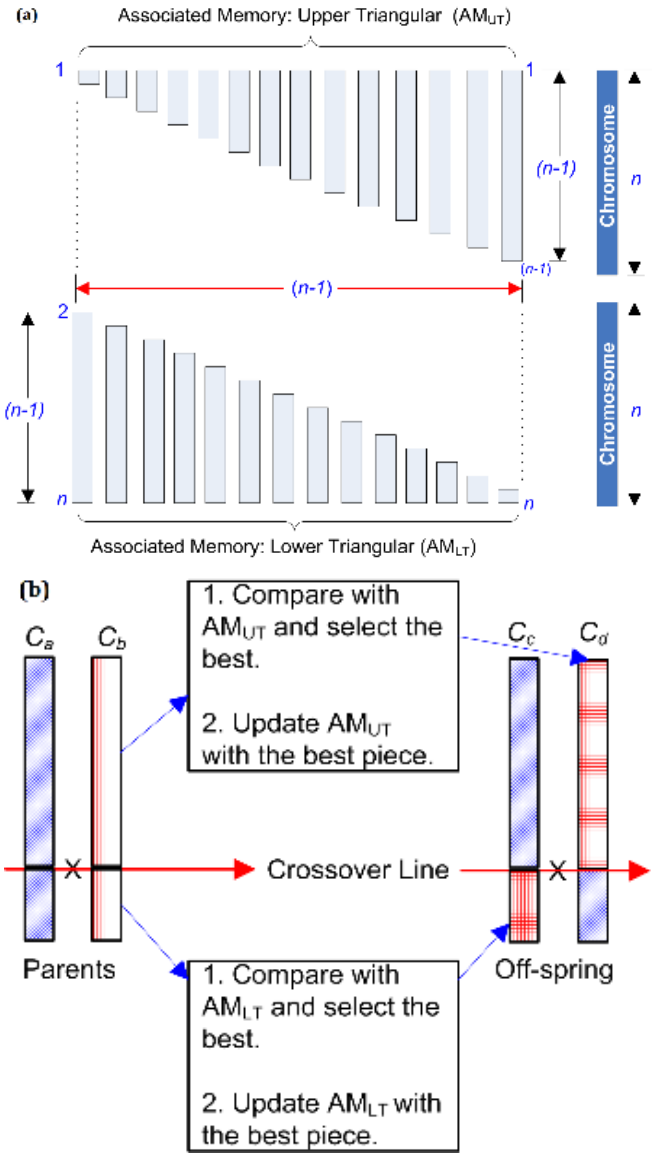
proteins that do not have their homologous protein's structures; however still requires proteins of known structure having similar folds. A wide range of machine learning algorithm-based predictors have been developed in the last two decades to predict protein folds (Sharma et al., 2013; Paliwal et al., 2014; Lyons et al., 2014, 2015) and protein structural classes (Dehzangi et al., 2013a, 2013b, Saini et al., 2014; Islam et al., 2015) using features such as secondary structure profile, PSSM profile, bigram and trigram probabilities, HMM profile, etc.

However, the template free *ab initio* approach for PSP problem is the most exciting one, it is yet very challenging as it does not use previously solved structures. The *ab initio* prediction of protein structure can provide us the insight of how the three-dimensional structure of protein is attained as it transforms sequence into structure from scratch. To solve PSP, we need to have an efficient conformational sampling algorithm along with an accurate energy function (Cooper et al., 2010; Das and Baker, 2008; Iqbal et al., 2015b). For a simplified model-based PSP problem, we have seen that even if we have a well-defined fitness or energy function to recognise the final goal, there is no efficient conformational sampling algorithm that can conveniently get the known final answer starting from a random conformation (Hoque et al., 2005). In this paper, we are especially motivated to provide GA-based improved sampling algorithm for PSP problem.

Genetic algorithm (GA), first proposed by Holland (1992, 2001), is an adaptive heuristic search and optimisation algorithm premised on the Darwin's principle of natural selection and genetics. A population of chromosomes (i.e., the solutions), proportionate selection procedure and the two operators: crossover and mutation are the constituents of a simple genetic algorithm (SGA). GA's crossover operation is regarded as its heart. Crossover has also been utilised within other bio-inspired optimisation algorithms as well to enhance their performances (Iqbal et al., 2015a; Milan, 2013). We have designed an associated-memory (AM)-based crossover within SGA to prioritise the crossover-participated better chromosome fragments. This crossover encourages the selection of fitter fragments stored in the memory. We named this alternate SGA a *fast-navigating genetic algorithm* (FNGA), as it converges faster.

Now, contrary to the diversity issues, our proposed FNGA approach would increase the similarity within population adversely. Therefore, after FNGA's role per generation, we immediately applied twin removal-based genetic algorithm (TRGA), by which similar chromosomes are replaced by new random chromosomes. TRGA approach can introduce potential subpart or sub-solution to enhance the performance of FNGA. And, the improved seed from FNGA can allow TRGA to enhance performance simultaneously when these two approaches are combined in the aforementioned way. We named the combination of FNGA and TRGA, as kite genetic algorithm or, KGA in short, as it resembles the characteristics of the bird kite in hunting down fast as well as searching thoroughly. KGA has been empirically found effective for a wide range of benchmark test functions. Moreover, KGA is proved to be a generic sampling algorithm while compared with state-of-the-art algorithms such as saw-tooth genetic algorithm (STGA) (Koumousis and Katsaras, 2006) and GA (named YGA) for *ab initio* PSP solution (Faraggi et al., 2009).

Figure 1 (a) Structural organisation of the two different sets of AM, where *n* indicates the length of the chromosome* (b) AM-crossover: a single point crossover operation involving AM (see online version for colours)



Note: *The structural organisation of the memory is based on single point crosser operation.

2 Fast-navigating genetic algorithm

The concept of the crossover is virtuous: parents after mating can produce better offspring. The idea for designing FNGA comes from the fact that there is no hard criteria to precisely select better parents for crossover as the selection procedure works probabilistically. Thus, while it is

beneficial to converge faster rather than exploring further, there is no mechanism within SGA to provide more chances to utilise the best available parts of chromosomes. We see the opportunity to allow the available best part, lengthen from minimum to maximum possible fragments, actively rather than relying on selection procedure. That is, exploring more with the best available subpart of the chromosome would help navigate faster as well as accurate.

To apply the ideas, we introduce two AM sets: one upper triangular and one lower triangular in shape (see Figure 1), based on single point crossover operation. Crossover operation is assumed to be applied at a higher rate (Koumousis and Katsaras, 2006), primarily for the intensification of a potential search area. We modify the crossover operation with the help of AM to navigate faster. We termed the modified crossover as 'AM-crossover' (see Figure 2). The number of the individual memory in each AM is equal to (length-1) of the chromosome. The AM stores the crossover-position-based participating best chromosome's subpart for the passing generations. In each crossover, we check that whether the subpart from AM generates better result than that of 2nd parent, in combination with 1st parent's subpart. The increased exploration of the best subpart stored in AM provides better trial for navigating faster as well as accurate solution without giving much chance to lose the better seed(s).

Figure 2 Algorithm for AM-crossover procedure

Procedure: **AM-Crossover** (C_a, C_b, i);
RETURN off-spring: (C_c, C_d)
Input: Parent chromosomes: (C_a, C_b); Crossover point: i
Output: Offspring chromosomes = (C_c, C_d)
// $C_k = k^{th}$ chromosome in the population,
// 'n'= (fixed) length of a chromosome, indicates the number of loci,
// 'i' = immediate lower-indexed-locus of the crossover position, where $1< i < n$, and 'j' = 'i'+1,

BEGIN
IF *fitness* ($C_{a\,[1\text{ to }i]} + C_{b\,[j\text{ to }n]}$) > *fitness* ($C_{a\,[1\text{ to }i]} + \text{AM}_{\text{LT}}(i)$) THEN
$\text{AM}_{\text{LT}}(i) = C_{b\,[j\text{ to }n]}$; $C_c = C_{a\,[1\text{ to }i]} + C_{b\,[j\text{ to }n]}$;
ELSE
$C_c = C_{a\,[1\text{ to }i]} + \text{AM}_{\text{LT}}(i)$;
END IF

IF *fitness* ($C_{b\,[1\text{ to }i]} + C_{a\,[j\text{ to }n]}$) > *fitness* ($\text{AM}_{\text{UT}}(i) + C_{a\,[j\text{ to }n]}$) THEN
$\text{AM}_{\text{UT}}(i) = C_{b\,[1\text{ to }i]}$; $C_d = C_{b\,[1\text{ to }i]} + C_{a\,[j\text{ to }n]}$;
ELSE
$C_d = \text{AM}_{\text{UT}}(i) + C_{a\,[j\text{ to }n]}$;
END IF
END

3 Twin removal genetic algorithm

It has been conclusively shown that a twin removal-based genetic algorithm (TRGA) affords considerably robust performance, especially for the hard optimisation problem such as *ab initio* PSP problem using lattice models as well as real all-atom model (Higgs et al., 2012; Hoque et al., 2011; Rashid et al., 2015, 2016). GA crossover and mutation operations by incorporating twin removal can avoid becoming ineffectual by replacing closely similar chromosomes with optimal number of random conformations (Hoque et al., 2007). Here, we investigate the twin removal approach to formulate a superior algorithm further. The twin removal algorithm is illustrated in Figure 3. The application of twin removal strategy is controlled by *CCF* or, *chromosome correlation factor* (quantification is indicated by r here) which introduces the level of similarities while comparing chromosomes. An optimal value of r ensures the robust performance of GA. The CCF (r), defines the degree of similarity between chromosomes from 0% (when, r = 0) to 100% (when, r = 1). Hence, a value of r = 75% implies that the similarity is 75% between two chromosomes. It has already been studied that TRGA performs the best when r is kept equal to 80% (Higgs et al., 2012; Hoque et al., 2011). Therefore, we also use the value of r = 0.8 in this study.

Figure 3 Twin removal algorithm (TRGA)

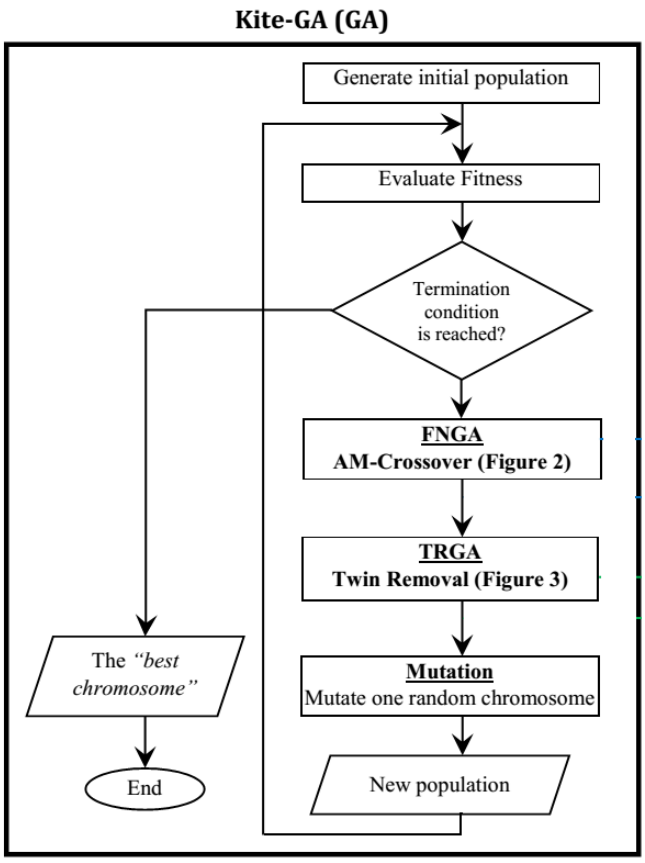
Procedure: **Twin Removal** (Pop_z, n);
RETURN population
Input: Population size = Pop_z,
Chromosome (C) length = n.
Output: Population with admissible level of twin similarity.
// C_i and C_j, where $i \neq j$.
// *RetSimilarity* (i, j) returns similarity of C_i and C_j in %.

BEGIN
FOR i = 1 to $(Pop_z - 1)$ DO
IF C_i.MarkDeleted = False THEN
FOR $j = i+1$ to Pop_z DO
IF *RetSimilarity*(i, j) $\geq$ r% THEN
IF $|\text{Fitness}(C_i)| < |\text{Fitness}(C_j)|$ THEN
Swap (C_i , C_j)
END IF
$C(j)$.MarkDeleted = True
END IF
END FOR
END IF
END FOR
END.

4 Kite genetic algorithm

The exploitation power of FNGA and exploration capacity of TRGA instigates us to combine them to form a novel GA which can perform effectively for a range of problems. We named the combination 'kite-GA' or 'KGA' in short as its ultimate form resembles the characteristic of bird kite in hunting down fast as well as searching thoroughly. The mechanisms of KGA are intended to enable effective search in a small as well as in a large neighbourhood of the search landscape.

Figure 4 Complete overview of KGA algorithm



Notes: The three GA operators, AM-crossover, twin removal and mutation are highlighted in bold. The building blocks of FNGA and TRGA are further expanded within blue and green boxes, respectively.

However, the ultimate form of KGA is determined empirically comparing the performance of two different combinations, termed:

1 Switching-mode (Koumousis and Katsaras, 2006): FNGA is regularly executed in each generation within crossover and after every five generations TRGA is executed and FNGA is not executed at the same go.

2 Mixing-mode (Yao et al., 1999): FNGA and TRGA are executed in the same generation.

However, FNGA executed first within crossover to provide more trial to potential subpart but increases the similarities within the population and TRGA removes and maintains optimal similarities after performing all the operations such as crossover and mutation. From preliminary experiments on benchmark functions, we found that the performances of the two modes were close (Hoque, 2015). However, the mixing-mode was relatively superior over the switching-mode as in this mode every passing generations have balanced exploration and exploitation. Therefore, we integrated the mixing-mode within our final KGA algorithm. Finally, KGA applies three operators sequentially in every generation to generate the new population which are AM-based single point crossover, single point mutation and twin removal. The full flowchart of KGA is shown in Figure 4.

5 Simulation studies

In this section, we analyse the performances of proposed GA variations in two levels:

1 we use widely adopted benchmark test functions to verify the proposed GA's strength of searching for the global optima within the function's complex landscapes

2 we stress the algorithm's sampling capacity to a limit by applying them in finding the optimal protein structure from astronomical conformational search space.

We perform the later experiment for both discrete or lattice model (Park and Levitt, 1995) and real (Rohl et al., 2004) PSP problem.

5.1 Sampling performance on benchmark functions

We collected 14 different benchmark functions from the base functions used in the latest CEC competition (Liang et al., 2013). These comprehensive set includes test functions with several challenging characteristics, like separability, modality, and dimensionality. Seven functions of the set have two variables, yet difficult to optimise due to their complex landscapes and the rest are scaled to include higher number of dimensions. In this study, we set the number of variables equal to 30 for the scalable functions. The test functions are listed in Table 1. It shows the function's name, formula, global minima and search bounds of the landscape. We used sequence of binary bits ('0' or, '1') to encode values of variables within the chromosomes of GA population. Each binary string corresponding to a variable has three parts: sign, decimal and fractional. The bit length of each variable differs for different functions due to different coverage of ranges within the search spaces shown in Table 1.

The GA parameter values for all the experiments in this study are: population size, Pop_z = 200; rate of crossover, p_c = 80%; rate of mutation, p_m = 5%; elite rate, p_e = 5%; maximum number of generations = 2,000. Moreover, roulette wheel approach is applied for the probabilistic selection of parents for the crossover operation. The display of results in Tables 2 and 3 include the best, average, standard deviation (S.D.) of the fitness values found from 30 independent iterations. We also report the average number of generations (avg. gen.) required to converge in all 30 iterations.

Table 2 displays the results for two-dimensional problems. The Easom, Leon, Rosenbrock and Zettl functions are unimodal, however, complex as their variables are interrelated (inseparable). Moreover, the first three of these functions have challenging landscapes with global minima inside a narrow space of the full landscape. Performances of KGA for Easom and Zettl functions are

effective. On the other hand, TRGA achieved minimum average finesses for Leon and Rosenbrock functions, however, KGA is competitive in both of the cases. The carrom table, egg holder and Schaffer's F2 functions have additional complexity of having highly multimodal landscapes. For all the three functions, KGA outperformed the other variations. For carrom table and egg holder functions, only KGA could discover the global minima in every iterations

Table 1 Description of benchmark test function

No.	*Function name*	*Function definition*	*Global minima*	*Bounds (point of minima)*	*Bit length of a variable*	*Properties*
		Number of variables, N = 2				
1	Easom	$-\cos(x_1)\cos(x_2)\exp(-(x_1-\pi)^2-(x_2-\pi)^2)$	–1	[–100, 100] (π, π)	27 (1 + 7 + 19)	Inseparable Unimodal
2	Carrom table	$-\frac{1}{30}\exp\left(2\left\lvert 1-\frac{\sqrt{x_1^2+x_2^2}}{\pi}\right\rvert\right)\cos^2(x_1)\cos^2(x_2)$	–24.1568	[10, 10] (± 9.6461, ± 9.6461)	40 (1 + 4 + 35)	Inseparable Multimodal
3	Egg holder	$-(x_2+47)\sin\sqrt{\left\lvert x_2+\frac{x_1}{2}+47\right\rvert}$ $-x_1\sin\sqrt{\lvert x_1-(x_2+47)\rvert}$	–959.6407	[–512, 512] (512, 404.2319)	36 (1 + 10 + 25)	Inseparable Multimodal
4	Leon	$100(x_2-x_1^2)^2+(1-x_1)^2$	0	[–1.2, 1.2] (1, 1)	18 (1 + 1 + 18)	Inseparable Unimodal
5	Rosenbrock	$100(x_2-x_1^2)^2+(x_1-1)^2$	0	[–2.049, 2.048] (1, 1)	23 (1 + 2 + 20)	Inseparable Unimodal
6	Schaffer's F2	$0.5+\frac{\sin^2(x_1^2-x_2^2)-0.5}{\left[1+0.001(x_1^2+x_2^2)\right]^2}$	0	[–100, 100] (0, 0)	27 (1 + 7 + 19)	Inseparable Multimodal
7	Zettl	$\frac{1}{4}x_1+(x_1^2-2x_1+x_2^2)^2$	–0.00379	[–5, 5] (–0.0299, 0)	29 (1 + 3 + 25)	Inseparable Unimodal
		Number of variables, N = 30				
8	Sphere	$\sum_{i=1}^{N}x_i^2$	0	[–100, 100] (0, …, 0)	21 (1 + 7 + 13)	Separable Unimodal
9	Cigar	$x_1^2+10^6\sum_{i=1}^{N}x_i^2$	0	[–100, 100] (0, …, 0)	20 (1 + 7 + 12)	Separable Unimodal
10	Ellipsoid	$\sum_{i=1}^{N}\sum_{j=1}^{i}x_j^2$	0	[–100, 100] (0, …, 0)	20 (1 + 7 + 12)	Separable Unimodal
11	Griewank	$\sum_{i=1}^{N}\frac{x_{iN}^2}{4000}-\prod_{i=1}^{N}\cos\left(\frac{x_i}{\sqrt{i}}\right)+1$	0	[–600, 600] (0, …, 0)	27 (1 + 10 + 16)	Inseparable Multimodal
12	Levy	$\sin^2(\pi y_1)+\sum_{i=1}^{N-1}\left[(y_i-1)^2\left(1+10(\sin^2 y_{i+1})\right)\right]$ $+(y_n-1)^2\left(1+\sin^2(2\pi y_n)\right),\ y_i=1+\frac{1}{4}(x_i+1)$	0	[–50, 50] (–1, …, –1)	19 (1 + 6 + 12)	Inseparable Multimodal
13	Schaffer's F6	$\sum_{i=1}^{N-1}\left(\frac{\sin^2\left(\sqrt{x_1^2+x_2^2}\right)-0.5}{\left[1+0.001(x_1^2+x_2^2)\right]^2}+0.5\right)$	0	[–100, 100] (0, …, 0)	23 (1 + 7 + 15)	Inseparable Multimodal
14	Zakharov	$\sum_{i=1}^{N}x_i^2\left(\frac{1}{2}\sum_{i=1}^{N}ix_i\right)^2+\left(\frac{1}{2}\sum_{i=1}^{N}ix_i\right)^4$	0	[–5, 10] (0, …, 0)	17 (1 + 4 + 12)	Inseparable Unimodal

Table 2 Comparison among GAs based on benchmark functions (number of variables = 2)

Functions	*Performance measure*	*SGA*	*FNGA*	*TRGA*	*KGA*
f_1 (Easom)	Best	*–1*	*–1*	*–1*	*–1*
	Average	–0.767	–0.755	–0.998	*–0.999*
	S.D.	0.3285	0.3634	0.0076	*0.0032*
	Avg. gen.	*655.567*	751.9	1,299.1	1,178.5
f_2 (Carrom table)	Best	*–24.1568*	*–24.1568*	*–24.1568*	*–24.1568*
	Average	–24.1544	–24.1541	–14.1567	*–24.1568*
	S.D.	0.0041	0.0043	2.31e-04	*1.44e-04*
	Avg. gen.	1,027.7	*995.367*	1,414.3	1,356.5
f_3 (Egg holder)	Best	*–959.6407*	*–959.6407*	–959.5797	*–959.6407*
	Average	–959.9735	–959.6155	–959.4355	*–959.6407*
	S.D.	23.4004	22.9353	0.7254	*1.115E-13*
	Avg. gen.	956.5667	*718.3*	1320.7	796.2667
f_4 (Leon)	Best	3.823e-08	8.307e-08	*2.328e-10*	1.886e-08
	Average	0.0188	0.0171	*1.35e-05*	1.67e-05
	S.D.	0.0402	0.0394	*1.99e-05*	3.09e-05
	Avg. gen.	966.7	972.267	901.667	*836.033*
f_5 (Rosenbrock)	Best	3.647e-10	6.14e-05	3.64e-10	*1.055e-10*
	Average	0.0703	0.0684	*4.692e-05*	5.935e-05
	S.D.	0.0866	0.0749	*5.913e-05*	8.259e-05
	Avg. gen.	891.2333	696.8333	981.5333	*676.166*
f_6 (Schaffer's F2)	Best	0	0	0	*0*
	Average	0.0066	0.005	1.00e-04	*5.09e-05*
	S.D.	0.0152	0.0132	1.53e-04	*7.06e-05*
	Avg. gen.	853	938.7333	*721.8333*	864.5667
f_7 (Zettl)	Best	*–0.0037*	*–0.00379*	*–0.00379*	*–0.00379*
	Average	0.00534	0.00302	*–0.003787*	*–0.003787*
	S.D.	0.0209	0.0186	*3.51e-06*	3.91e-06
	Avg. gen.	476.2667	425.0667	154.7333	*61.8*

Note: Best results are highlighted in italic.

Table 3 focuses on the results of functions with 30 variables. The sphere, cigar and rotated hyper-ellipsoid (in short called as Ellipsoid in this study) are highly separable and unimodal. We observe that all the algorithms could reach the global minima, however, KGA and FNGA resulted faster convergence. The Griewank, Levy, extended Schaffer's F6 and Zakharov functions are the most challenging functions being inseparable and multimodal.

KGA performed best for Zakharov function in terms of fitness values and for Griewank and Levy functions in terms of convergence speed. However, generalised conclusion using a small set of benchmark problems may not be appropriate, as no single search algorithm is best on average for all problems as explained in no free lunch theorem (Wolpert and Macready, 1997).

5.1.1 Convergence test

To further investigate the convergence process while searching for global minima, we plot the average fitness found from the 30 iterations per generation in Figure 5. It exhibits separate plots for each of the 14 test functions, where the left column contains the plots for functions with two variables and the right column includes those for the functions with 30 variables. We observe that KGA gave superior performance than SGA and FNGA with large differences for f_1 to f_4 and f_6 to f_7. For these function, KGA and TRGA performed comparatively. For two-dimensional functions f_5 and 30-dimensional functions, f_8 to f_{12}, performances of KGA were competitive, however better. For Schaffer's F6 function (f_{13}), performance of SGA was effective than others both in terms of fitness value and search progress. KGA gave better fitness value for Zakharov function (f_{14}), however the plots shows that the convergence progress of TRGA was better.

5.2 Sampling performance for PSP

Here, we investigate the sampling capacity of the proposed KGA in locating the conformation of protein within the

complex search space. The primary structure of protein defines the function of protein when folded into tertiary structure. However, due to large degree of freedom, the primary protein sequence can fold into an astronomical number of structures. Lattice models of proteins are extremely useful for the discretisation of the real conformation space by sacrificing the atomic detail (Hart and Newman, 2001). In the form of tertiary structure, proteins have the minimum free energy conformation. Therefore, we first applied KGA to find the minimum energy conformation for widely used 2D *hydrophobic-polar* (HP) model (Hoque et al., 2011). Later, we exercised the technique in case of real PSP.

5.2.1 Sampling discrete protein structure space

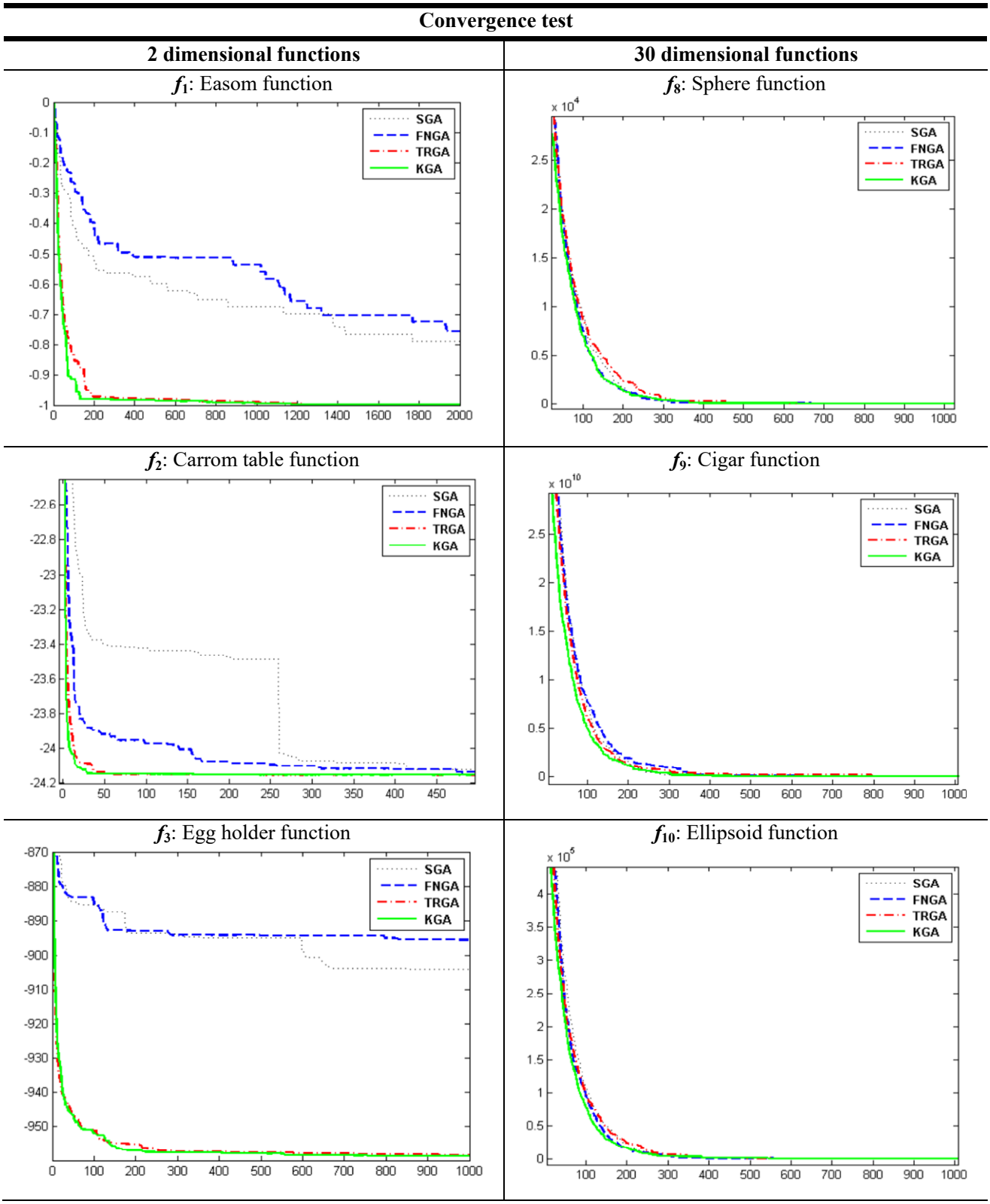
Here, we compared the performances of FNGA, TRGA, KGA and STGA in discrete protein structure sampling. STGA utilises a variable population size between two successive generations following a periodic scheme in the form of a saw-tooth function along with population re-initialisation with randomly generated new chromosomes. In comparison, TRGA replaces portion of the population by randomly generated chromosomes. For the STGA, we used the best recommended parameters in Koumousis and Katsaras (2006). The benchmark HP sequences (see Table 4) were used and the results are given in Table 5. TRGA performed better compared to both FNGA and STGA and showed its robustness while solving this hard optimisation problem. Further, FNGA also outperformed STGA in this case as well. KGA outperformed all the other variations for these hard optimisation problems.

Table 3 Comparison among GAs based on benchmark functions (number of variables = 30)

Functions	*Performance measure*	*SGA*	*FNGA*	*TRGA*	*KGA*
f_8 (Sphere)	Best	*0*	*0*	*0*	*0*
	Average	*0*	*0*	*0*	*0*
	S.D.	*0*	*0*	*0*	*0*
	Avg. gen.	622.667	621.133	640.300	*612.133*
f_9 (Cigar)	Best	*0*	*0*	*0*	0
	Average	*0*	*0*	*0*	0
	S.D.	*0*	*0*	*0*	0
	Avg. gen.	585.5	*580.226*	580.2667	644.1
f_{10} (Ellipsoid)	Best	*0*	*0*	*0*	*0*
	Average	*0*	*0*	*0*	*0*
	S.D.	*0*	*0*	*0*	*0*
	Avg. gen.	619.7	591.4	609.2333	*567.7333*
f_{11} (Griewank)	Best	0.0425	*0*	0.0204	0.0526
	Average	*0.3582*	0.4053	0.4148	0.4047
	S.D.	0.2912	0.4081	*0.2397*	0.3098
	Avg. gen.	1,034.5	982.3667	1,045.7	*918.0667*
f_{12} (Levy)	Best	0.4947	0.4949	*3.95e-07*	0.9576
	Average	2.2665	*1.8086*	2.0548	2.5639
	S.D.	1.0159	0.9203	1.0451	1.0739
	Avg. gen.	1,221.267	1,149.367	1,145.933	*1,142.8*
f_{13} (Schaffer's F6)	Best	0.3733	*0.3733*	0.4297	0.4518
	Average	*0.4447*	0.4527	0.4598	0.4786
	S.D.	0.0315	0.0293	0.0183	*0.014*
	Avg. gen.	1,756.7	1,783.8	*1,377.9*	1,442
f_{14} (Zakharov)	Best	207.1913	206.9732	152.2861	*167.4725*
	Average	345.3214	333.733	234.8338	*233.0159*
	S.D.	78.1125	80.7571	*39.9526*	44.8688
	Avg. gen.	1,985.3	1,960.1	1,980.7	*1,938.8*

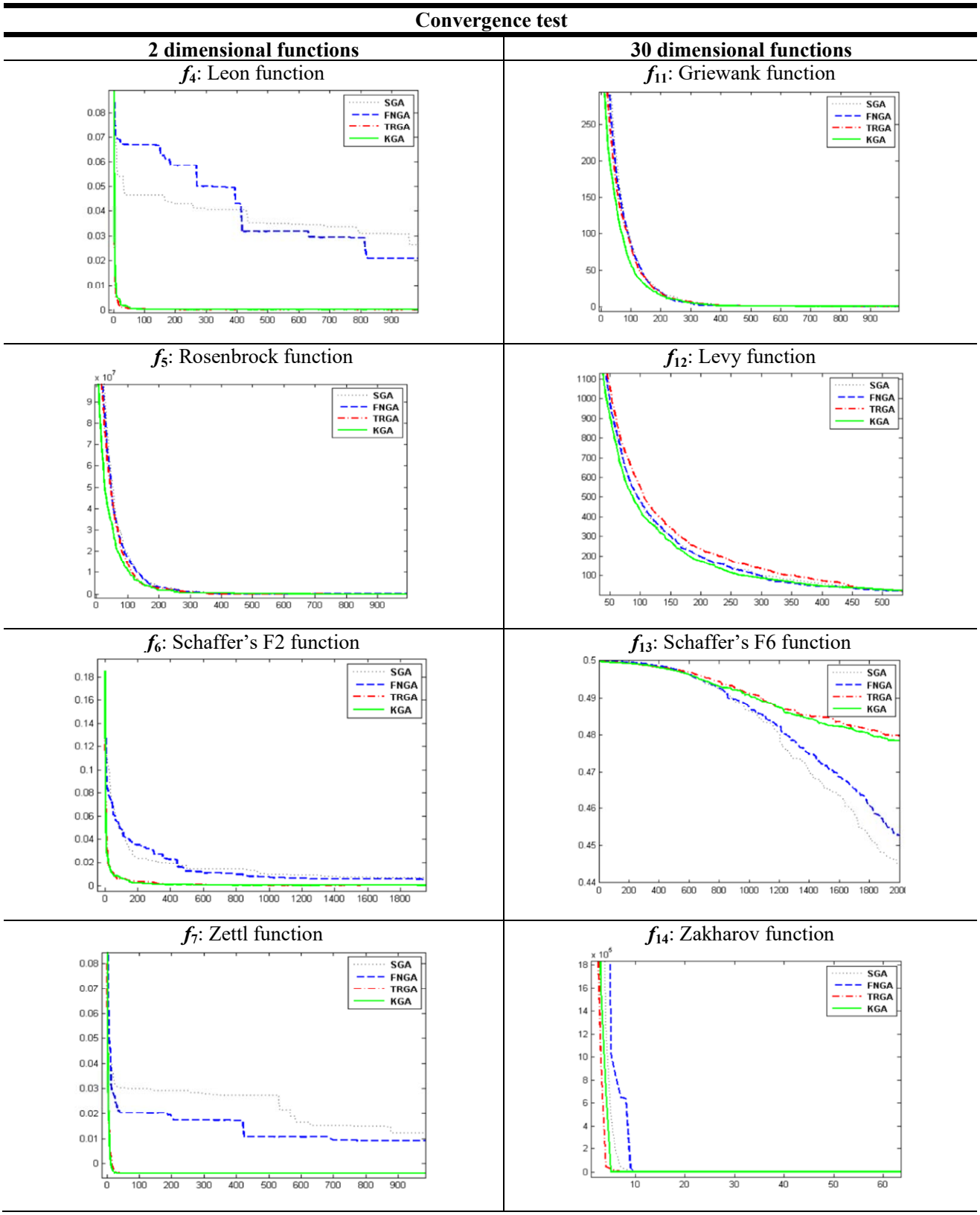
Note: Best results are highlighted in italic.

Figure 5 Convergence progresses by SGA (*black dotted line*), FNGA (*blue dashed line*), TRGA (*red dash-dot line*) and KGA (*green solid line*) in consecutive generations for the 14 benchmark test functions (see online version for colours)



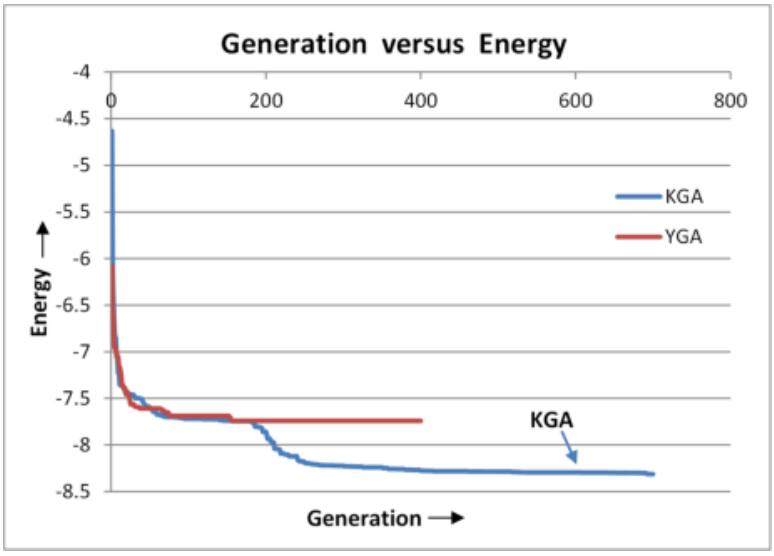
Notes: The left column shows the convergence processes for seven functions with two variables (dimensions), whereas the right column shows those for functions with 30 variables (dimensions). In each plot, the x-axis and y-axis show the number of generations and the average fitness values of 30 iterations, respectively.

Figure 5 Convergence progresses by SGA (*black dotted line*), FNGA (*blue dashed line*), TRGA (*red dash-dot line*) and KGA (*green solid line*) in consecutive generations for the 14 benchmark test functions (continued) (see online version for colours)



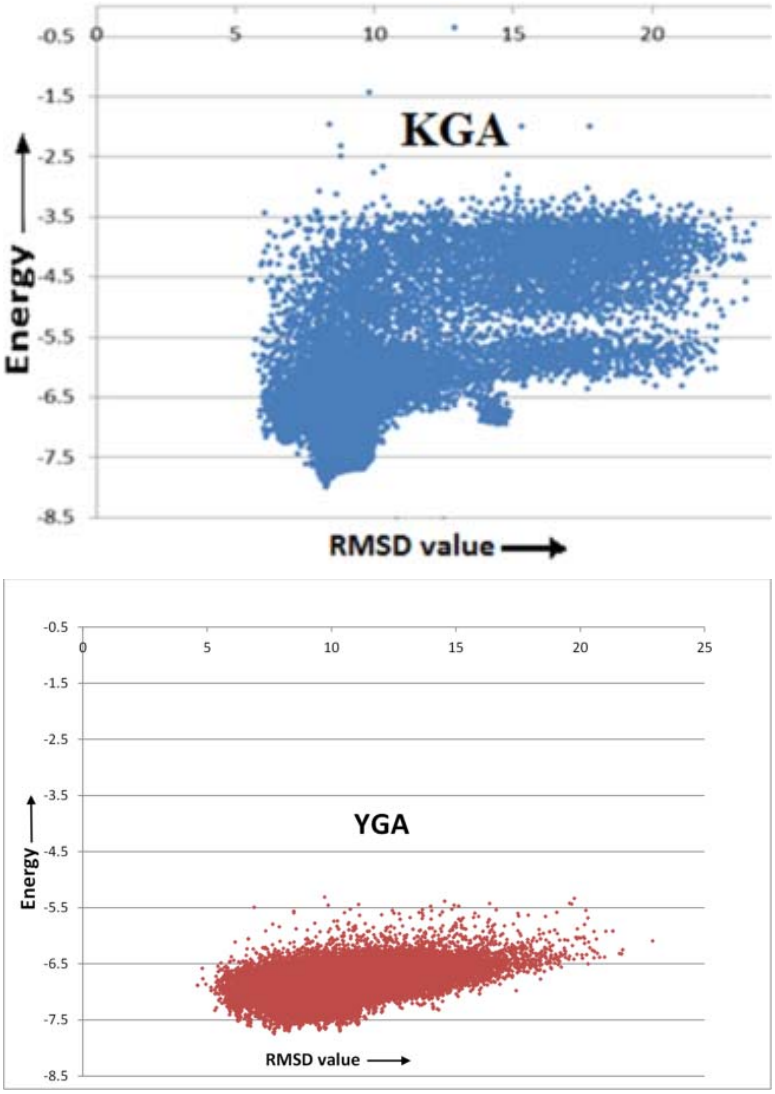
Notes: The left column shows the convergence processes for seven functions with two variables (dimensions), whereas the right column shows those for functions with 30 variables (dimensions). In each plot, the x-axis and y-axis show the number of generations and the average fitness values of 30 iterations, respectively.

Figure 6 KGA versus YGA, in getting lower energy minimum (see online version for colours)



Note: PDB ID: 1b72, 49 residues long.

Figure 7 KGA versus YGA comparison in the energy versus RMSD space for greater coverage (see online version for colours)



Note: Ran for 400 generations, sequence ID 1b72.

Table 4 Benchmark protein sequences for 2D HP model

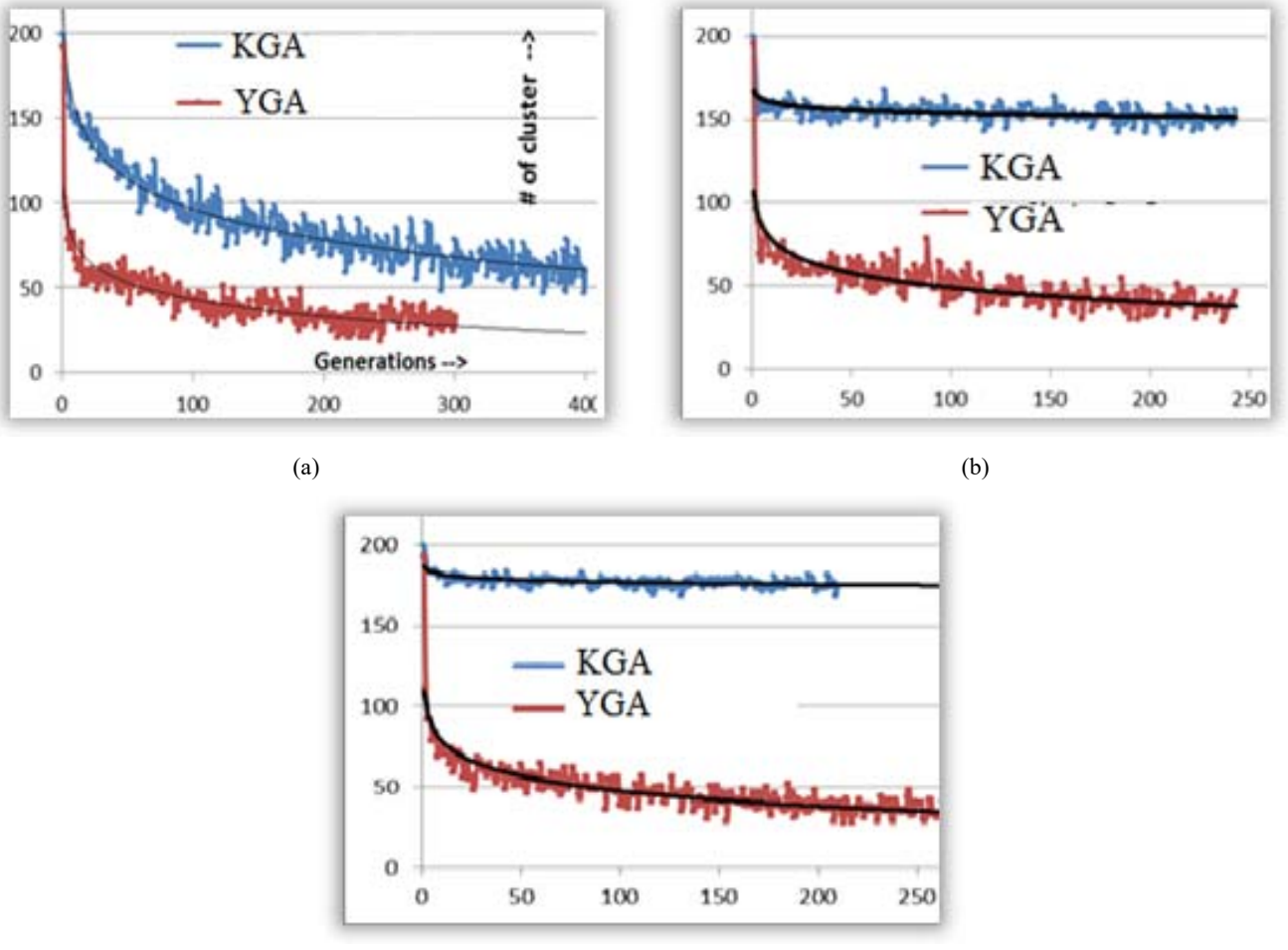
Length	*Sequences*	*Ref.*
50	H2(PH)3PH4PH(P3H)2P4H(P3H)2PH4P(HP)3H2	Unger and Moult (1993)
60	P2H3PH8P3H10PHP3H12P4H6PH2PHP	Unger and Moult (1993)
64	H12(PH)2(P2H2)2P2HP2H2PPHP2H2P2(H2P2)2(HP)2H12	Unger and Moult (1993)
85	4H4P12H6P12H3P12H3P12H3P1H2P2H2P2H2P1H1P1H	Lesh et al. (2003)
100	3P2H2P4H2P3H1P2H1P2H1P4H8P6H2P6H9P1H1P2H1P11H2P3H1P2H1P1H2P1H1P3H6P3H	Lesh et al. (2003)

Table 5 Comparisons of STGA, FNGA, TRGA and KGA based on the benchmark sequences (see Table 4)

Len.	*STGA*		*FNGA*		*TRGA*		*KGA*	
	Fitness (Avg.)	*Fitness (S.D.)*	*Fitness (Avg.)*	*Fitness (S.D.)*	*Fitness (Avg.)*	*Fitness (S.D.)*	*Fitness (Avg.)*	*Fitness (S.D.)*
50	–12.78	1.481	–18.4	2.3664319	*–21*	0	*–21*	0
60	–25.6	2.221	–30	1.6996732	–33.8	1.154701	*–34.7*	0.823273
64	–22.4	1.578	–29.9	1.5238839	–37	1.1301	*–37.5*	0.707107
85	–32.5	2.273	–43.1	2.1832697	–46.8	1.264911	*–49.3*	1.159502
100	–27.6	3.062	–37.9	3.5730473	–44.8	1.135292	*–45.5*	0.849837

Notes: Average (Avg.) and standard deviations (S.D.) values are obtained from ten iterations and the maximum generation was 6,000. Best results are highlighted in italic.

Figure 8 Comparison of KGA versus YGA for sampling diversity PDB ID: length (a) 1b72:49, (b) 2reb:60 and (c) 1af7:72 (see online version for colours)



(a)

(b)

(c)

5.2.2 Sampling real protein structure space

Here, we compare the sampling performance of KGA by putting it in the real PSP scenario. A state-of-the-art, fragment free *ab initio* structure prediction algorithm based on GA had been developed in Faraggi et al. (2009). We call this GA, YGA, in this paper and compare KGA with it. To have a fair comparison between YGA and KGA, we only replaced YGA of the real *ab initio* program of Faraggi et al. (2009) with our KGA keeping other components same. There are 16 benchmark sequences discussed in Faraggi et al. (2009). We ran several of them to characterise the sampling performance of KGA in real scenario of PSP.

First, we check the performance by comparing the achievability of low energy conformation. We found that YGA's improvement gets flat (Figure 6) in obtaining lower energy after around 200 generations. KGA exceed YGA in obtaining lower energy conformation and it did not get flat. Therefore, we let KGA run till 1,000 generations (shown up to 700 generations in Figure 6) to highlight the performance fluctuation between KGA and YGA. The same characteristic is found for other runs as well.

Second, to confirm that KGA is not switching heavily among few sets of diverse conformations, we also compared the total coverage in the RMSD versus energy space. For same number of generations KGA is found to sample relatively larger area as compared in Figure 7.

Third, we wanted to see the effectiveness of KGA in producing new samples in every consecutive generations. For this, as the generation is passing, we created conformational groups or clusters of protein structures (chromosomes of the population) that are at least 2.5 Å root-mean-square-deviation (RMSD) apart. We plotted generation versus the number of such clusters in Figure 8. KGA generated more diverse sample in consecutive generations. It is also interesting to note that as the length of the sequence increases the diversity also increases in KGA which is a good sampling characteristic, whereas YGA remains monotonic and does not vary noticeably based on the length.

6 Conclusions

This paper proposes two variations of classical GA for enhanced sampling performance. At first, FNGA with a new crossover technique is presented. Later, we combined twin removal-based genetic algorithm (TRGA) that maintains optimal diversity, with FNGA to design the final sampling algorithm, KGA. The new KGA can extract more information from a finite number of generations as well as can achieve the robustness.

The performances of SGA, FNGA, TRGA and KGA are empirically compared on a range of continuous benchmark test functions. Moreover, the proposed sampling algorithms are employed in search of minimum energy conformation of PSP problem both in discrete and real scopes. For discrete PSP problem, KGA outperformed the STGA. Moreover, we compared the sampling properties of KGA with YGA, a state-of-the-art real *ab initio* PSP program. KGA is found to have promising sampling characteristics. Therefore, a useful future research direction out of this work would be to apply KGA in other discrete and hard combinatorial optimisation problems.

Supplementary material

The KGA code can be found here: http://cs.uno.edu/~tamjid/Software/FN_KGA/FN_KGA.zip.

Acknowledgements

MTH would like to thank Professor Yaoqi Zhou and Dr. Yuedong Yang for providing the code of the *ab initio* PSP software including YGA to compare with KGA directly and for helpful discussion. MTH and SI gratefully acknowledge the Louisiana Board of Regents through the Board of Regents Support Fund, LEQSF (2013–16)-RD-A-19 and LEQSF (2016-19)-RD-B-07.

References

Anfinsen, C.B. (1973) 'The principles that govern the folding of protein chains', *Science*, Vol. 181, No. 4096, pp.223–230.

Cooper, S., Khatib, F., Treuille, A., Barbero, J., Lee, J., Beenen, M., Leaver-Fay, A., Baker, D., Popović, Z. and Players, F. (2010) 'Predicting protein structures with a multiplayer online game', *Nature*, Vol. 466, No. 7307, pp.756–760.

Das, R. and Baker, D. (2008) 'Macromolecular modeling with Rosetta', *Biochemistry, Annual Reviews*, Vol. 77, pp.363–382.

Dehzangi, A., Paliwal, K., Lyons, J., Sharma, A. and Sattar, A. (2013a) 'Exploring potential discriminatory information embedded in PSSM to enhance protein structural class prediction accuracy', *Pattern Recognition in Bioinformatics*, Springer, Vol. 7986 of the series Lecture Notes in Computer Science, pp.208–219.

Dehzangi, A., Paliwal, K., Sharma, A., Dehzangi, O. and Sattar, A. (2013b) 'A combination of feature extraction methods with an ensemble of different classifiers for protein structural class prediction problem', *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, Vol. 10, No. 3, pp.564–575.

Faraggi, E., Yang, Y., Zhang, S. and Zhou, Y. (2009) 'Predicting continuous local structure and the effect of its substitution for secondary structure in fragment-free protein structure prediction', *Structure*, Vol. 17, No. 11, pp.1515–1527.

Hart, W.E. and Newman, A. (2001) *Protein Structure Prediction with Lattice Models*, CRC Press [online] http://dimacs.rutgers.edu/~alantha/papers2/alantha-bill-bc.pdf.

Higgs, T., Stantic, B., Hoque, M.T. and Sattar, A. (2012) 'Refining genetic algorithm twin removal for high-resolution protein structure prediction', *2012 IEEE Congress on Evolutionary Computation (CEC)*, IEEE, pp.1–8.

Holland, J.H. (1992) 'Genetic algorithms', *Scientific American Journal*, pp.66–72.

Holland, J.H. (2001) *Adaptation in Natural and Artificial Systems*, The MIT Press, Cambridge, Massachusetts London, England.

Hoque, M.T. (2015) *Genetic Algorithms based Improved Sampling*, Tech. Report TR-2015/4.

Hoque, M.T., Chetty, M. and Dooley, L.S. (2005) 'A new guided genetic algorithm for 2D hydrophobic-hydrophilic model to predict protein folding', *IEEE Congress on Evolutionary Computation (CEC)*.

Hoque, M.T., Chetty, M. and Dooley, L.S. (2007) 'Generalized schemata theorem incorporating twin removal for protein structure prediction', *Pattern Recognition in Bioinformatics*, Springer, Singapore.

Hoque, M.T., Chetty, M., Lewis, A. and Sattar, A. (2011) 'Twin removal in genetic algorithms for protein structure prediction using low-resolution model', *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, Vol. 8, No. 1, pp.234–245.

Iqbal, S., Kaykobad, M. and Rahman, M.S. (2015a) 'Solving the multi-objective vehicle routing problem with soft time windows with the help of bees', *Swarm and Evolutionary Computation*, Vol. 24, pp.50–64.

Iqbal, S., Mishra, A. and Hoque, M.T. (2015b) 'Improved prediction of accessible surface area results in efficient energy function application', *Journal of Theoretical Biology*, Vol. 380, pp.380–391.

Islam, M.N., Iqbal, S., Katebi, A.R. and Hoque, M.T. (2015) 'A balanced secondary structure predictor', *Journal of Theoretical Biology*.

Koumousis, V.K. and Katsaras, C.P. (2006) 'A saw-tooth genetic algorithm combining the effects of variable population size and reinitialization to enhance performance', *IEEE Transactions on Evolutionary Computation*, Vol. 10, No. 1, pp.19–28.

Lesh, N., Mitzenmacher, M. and Whitesides, S. (2003) 'A complete and effective move set for simplified protein folding', *RECOMB*, Berlin, Germany.

Levinthal, C. (1968) 'Are there pathways for protein folding', *J. Chim. Phys.*, Vol. 65, No. 1, pp.44–45.

Liang, J., Qu, B. and Suganthan, P. (2013) *Problem Definitions and Evaluation Criteria for the CEC 2014 Special Session and Competition on Single Objective Real-Parameter Numerical Optimization*, Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China and Technical Report, Nanyang Technological University, Singapore.

Lyons, J., Biswas, N., Sharma, A., Dehzangi, A. and Paliwal, K.K. (2014) 'Protein fold recognition by alignment of amino acid residues using kernelized dynamic time warping', *Journal of Theoretical Biology*, Vol. 354, pp.137–145.

Lyons, J., Dehzangi, A., Heffernan, R., Yang, Y., Zhou, Y., Sharma, A. and Paliwal, K. (2015) 'Advancing the accuracy of protein fold recognition by utilizing profiles from hidden Markov models', *IEEE Transaction on NanoBioscience*, Vol. 14, No. 7, pp.761–772.

Milan, T. (2013) 'Artificial bee colony (ABC) algorithm with crossover and mutation', *Appl. Soft Comput*., pp.687–697.

Paliwal, K.K., Sharma, A., Lyons, J. and Dehzangi, A. (2014) 'A tri-gram based feature extraction technique using linear probabilities of position specific scoring matrix for protein fold recognition', *IEEE Transactions on NanoBioscience*, Vol. 13, No. 1, pp.44–50.

Park, B.H. and Levitt, M. (1995) 'The complexity and accuracy of discrete state models of protein structure', *Journal of Molecular Biology*, Vol. 249, No. 2, pp.493–507.

Rashid, M.A., Iqbal, S., Khatib, F., Hoque, M.T. and Sattar, A. (2016) 'Guided macro-mutation in a graded energy based genetic algorithm for protein structure prediction', *Computational Biology and Chemistry*, Elsevier.

Rashid, M.A., Khatib, F., Hoque, M.T. and Sattar, A. (2015) 'An enhanced genetic algorithm for ab initio protein structure prediction', *IEEE Transactions on Evolutionary Computation* No. 4.

Rohl, C.A., Strauss, C.E., Misura, K.M. and Baker, D. (2004) 'Protein structure prediction using Rosetta', *Methods in Enzymology*, Vol. 383, pp.66–93.

Saini, H., Raicar, G., Sharma, A., Lal, S., Dehzangi, A., Rajeshkannan, A., Lyons, J., Biswas, N. and Paliwal, K.K. (2014) 'Protein structural class prediction via k-separated bigrams using position specific scoring matrix', *J. Adv. Comput. Intell. Intell. Informatics*, Vol. 8, No. 4, pp.474–479.

Sharma, A., Lyons, J., Dehzangi, A. and Paliwal, K.K. (2013) 'A feature extraction technique using bi-gram probabilities of position specific scoring matrix for protein fold recognition', *Journal of Theoretical Biology*, Vol. 320, No. 1, pp.41–46.

Unger, R. and Moult, J. (1993) 'Genetic algorithms for protein folding simulations', *Journal of Molecular Biology*, Vol. 231, No. 1, pp.75–81.

Wolpert, D.H. and Macready, W.G. (1997) 'No free lunch theorems for optimization', *IEEE Transactions on Evolutionary Computation*, Vol. 1, No. 1, pp.67–82.

Yao, X., Liu, Y. and Lin, G. (1999) 'Evolutionary programming made faster', *IEEE Transaction on Evolutionary Computation*, Vol. 3, No. 2, pp.82–102..