EasyChair Preprint

№ 7237

2D HP Protein Folding Using Quantum Genetic Algorithm

Moein Atari and Nayereh Majd

December 17, 2021

2D HP protein folding using quantum genetic algorithm

Moein Atari
Department of engineering science, college of engineering
Tehran University
Tehran, Iran
Atari.moein@ut.ac.ir

Nayereh Majd
Department of engineering science, college of engineering
Tehran University
Tehran, Iran
naymajd@ut.ac.ir

***Abstract*— The protein structure prediction (PSP) is one of the major challenges in modern biology. Using new technology which have powerful tools to computation like quantum computation can decrease time of process and memory useless to find optimized solution of PSP. In this paper we introduce new approach to find optimal solution of 2D HP protein folding using quantum genetic algorithm which combination of advantages of quantum computation and genetic evolutionary process. Our approach can find solution of problem faster and more memoryless that traditional genetic algorithm by maintaining accuracy.**

Keywords—Protein structure prediction, Genetic algorithm, Quantum computation, Quantum genetic algorithm.

I. INTRODUCTION

One of the most important macromolecules in all living organisms are proteins [1-2]. Protein structure prediction (PSP) is a significant and very challenging interdisciplinary problem includes biochemistry, biophysics, structural biology, molecular biology and computational biology. In PSP relations between sequences and protein folding is the key to combating many fatal diseases such as Prion disease, Alzheimer's disease, Parkinson's disease and cancer [3-7] and the development of several applications in rational drug design, and biotechnology, so PSP is an important multi-disciplinary research problem.

Most of the methods to find solution of PSP follow the thermodynamics hypothesis, i.e., protein adopts the conformation under physiological conditions with the lowest Gibbs free energy [8]. It means that in this problem we are looking for minimize problem, and it can be divided into two sub-problems: (i) to declare an appropriate energy function that can find global minimum and is able to recognize feasible from infeasible folds; (ii) to introduce an efficient and strange algorithm which capable of dealing with a large search space.

There are several methods for protein structures like X-ray crystallography [9] and MRI (magnetic resonance imaging) and electron microscopy which are time-consuming, cost-intensive and failure-prone and are not sufficient to fill the gap between the number of known protein sequences and the number of solved structures.

Recently, many paper introduced evolutionary algorithms, such as genetic algorithm (GA) [10-13], for solving the PSP problem [14-20]. Recently, GA have been used to find solution of optimization problem in reasonable time. GA was first proposed by Holland inspired by Darwin's principle of survival of the fittest. The first step in GA is defined a representation that describes the possible solutions for a problem. In GA, the possible solution of an optimization problem is encoded in chromosome and each chromosome includes some small parts which call gen. A group of finite chromosomes make population. Every chromosome is evaluated by fitness function, and with fitness function we can compare chromosomes and recognize feasible from infeasible ones. If termination condition occurred, GA shows best founded solution, otherwise it reproduces next population of chromosome by crossover and mutation.

Quantum technology was introduced as a powerful tool for computation [21-22], it has received growing attention in recent years and researchers started to combined quantum computation with genetic algorithm [23-25] and first attempt was made by Narayanan and Moore to propose Quantum Genetic Algorithm (QGA). QGA makes computation of genetic evolutionary process faster by exploiting the power of quantum computation. Also, QGA has less population size, more powerful in global search, higher convergence rate, and less execution time. QGA can be used to find solution of various kind of problems such as combinatorial and functional optimization problems, image processing, identification and engineering optimization problems.

Protein includes sequential-chains of amino acids that connected together by single peptide bonds. The fold of these connected chains can be shown in two-dimensional (2D) structures. In this paper, we present a new approach using quantum genetic algorithm for solving PSP problem in 2D HP lattice model. The proposed QGA for solving PSP problem has some advantages like speeds up computation of genetic evolutionary process, has more powerful in global

search and has less population size. The rest of this paper is organized as followed:

1. Introduce problem, genetic algorithm and describe quantum computation briefly
2. Describe proposed quantum genetic algorithm for solving PSP which includes steps of quantum genetic algorithm, encoding chromosome, reproduces new population and compare steps of GA and steps of QGA
3. Describe our proposed method in detail and simulate it.
4. Simulation
5. The conclusion

II. PRELIMINARIES

In this section, we first describe 2D HP protein folding, and problem detail, discus genetic algorithm and its steps, and finally we introduce quantum computation briefly.

A. *The 2D HP protein folding problem*

The highly simplified computer models that use for protein folding in computer is lattice protein [4]. We can simulate a few microseconds of protein behavior in complete atomic detail with current technology because proteins are large molecules, so they include hundreds or thousands of atoms, in other word, it is impossible to model real protein folding in computer simulation. Lattice protein, are simplified in two steps: (i) every atom in the amino acids is modeled as "beads", (ii) and all these "beads" are restricted to a rigid (usually cubic) lattice. It speeds up simulation process to find protein folding with minimal energy.

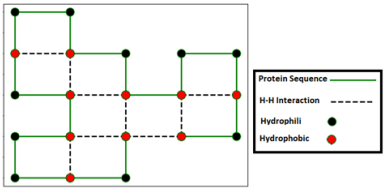


Fig. 1. Representation of folding for HPHPPHHPHPPHPHHPPHPH sequence with free energy -9 .

In this paper, we use the HP model, includes just two bead types: H (hydrophobic or non-polar) and P (hydrophilic or polar). Figure 1 shows an example of 2D lattice model. Red squares denote the hydrophobic amino acid, the black squares denote the hydrophilic and the green dotted line denote H-H interaction, and it means free energy. For calculating the total energy (E) of a conformation based on the HP model, we sum H-H interaction of non-consecutive hydrophobic amino acids.

$$E = \sum_{i<j-1} c_{ij} \times e_{ij} \qquad (1)$$

where, if amino acids i and j are non-consecutive neighbors on the lattice $c_{ij} = 1$, otherwise $c_{ij} = 0$; and $e_{ij} = 1$ if the $i_{th}$and j_{th} amino acids are hydrophobic, otherwise $e_{ij} = 0$. In this problem, we look to minimize equation 1 or maximize the number of H-H interaction of non-consecutive hydrophobic amino acids.

B. *Genetic algorithm for PSP*

The Genetic algorithms (GAs) have recently been used to solve optimization problems very commonly since they can get nearly optimal solutions in reasonable time. The possible solution of an optimization problem is encoded in a chromosome, which consists of an array of genes. The individual chromosome is evaluated by a fitness function. The fitness function is a function that receives the candidate solution as input and displays the fitness of the proposed solution of the problem as output. A genetic population consists of a finite number of chromosomes. The chromosomes of the new population are generated by the application of genetic operations such as crossover, mutation, and reproduction on the present population. Flow diagram of genetic algorithm show on figure 2.

A representation that describes the possible solutions for a problem must first be defined when applying genetic algorithms to solve a problem. The feasibility of using genetic algorithm is problem dependent. Its success strongly depends on whether the right encoding scheme can be adopted or not. A successful encoding scheme must have the following features:

1. It can present all variables' information of any solutions
2. It can keep legality of encoding in genetic operations such as crossover and mutation;
3. The encoding and decoding can be carried out easily;
4. The mapping between encoding space and solution space is a one-to-one mapping, i.e., one coding specification identifies only one solution, and vice versa.

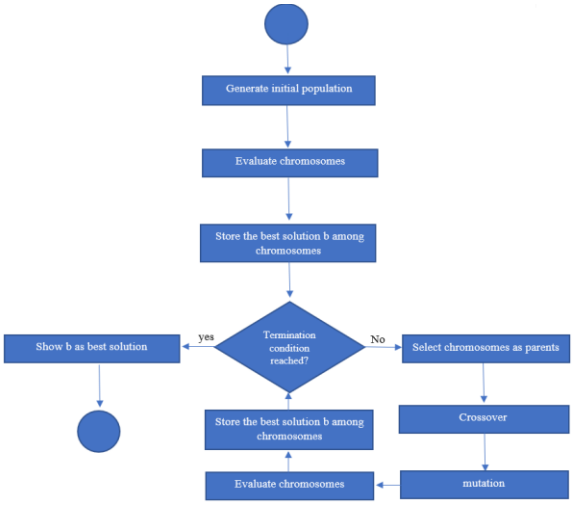


Fig. 2. Flow diagram of genetic algorithm

1) Encoding chromosomes

As mentioned, the first step of GA is encoding chromosomes. In this step is to define a way to represent solution of a problem in chromosomes, so encoding of chromosomes in GA depends on problem. There are several methods to encoding chromosomes in PSP [19-20]. Based on ref [19-20], If the input of problem is amino acid sequence and its length equal to N, then each chromosome in the population has $N-1$ genes, and each gene can be assignment over the symbols = $\{U, R, D, L\}$, and that denotes a feasible conformation in the 2D square lattice. The symbols U, L, R and D are used to denote the fold directions up, left, right and down in the encoding scheme, respectively. Figure 3 shows an example of GA encoding chromosome in 2D square lattice for PSP.

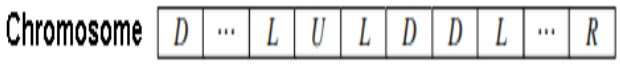


Fig. 3. A method to GA encoding chromosme in 2D square lattice for PSP

2) Termination condition

Termination condition divided into three categories:

1. Termination after the expiration of the specified number of repetitions.
2. Termination after the expiration of the given time.
3. Termination by reaching an acceptable level of the solution.

If termination condition occurs, the best solution is obtained, stored in b and displayed as an output otherwise it enters the loop of the operation to make the termination condition occur.

3) Reproduction population

First, we must select some chromosome as parents to generate new population. There are some methods to select chromosomes as parents. We can mention fitness selection and roulette-wheel as most popular parent selection method in GA.

- Fitness selection: in this method, the chromosome that has the best fit function are copied into the new population, and the rest of the chromosomes are made using conventional methods. This method increases the performance of the GA because it prevents the removal of the best answers found.
- Roulette-wheel selection: this method is one of the most famous and widely used methods of selecting parents in the genetic algorithm. The probability of selection of a sector in a roulette wheel is proportional to the magnitude of the central angle of the sector. Similarly, in Genetic Algorithm, the whole population are partitioned on the wheel and each sector represents an individual. The proportion of chromosome's fitness to the total fitness values of whole population decides the probability of selection of that individual in the next generation.

The crossover is one of the genetic operations to generate the next population, usually exchanges some genes between two chromosomes with constant probability. Single-point crossover and double-point crossover are the most useful and popular method for crossover, and we can use these for PSP problem.

- Single-point crossover: in this method, a location of parent chromosomes is randomly selected and genes are swapped between parents based on it to make children chromosomes.
- Double-point crossover: in this method, two locations of parent chromosomes are randomly selected and genes are swapped between parents based on those to make children chromosomes.

Mutation is a genetic operator that is used to maintain the genetic diversity of a population of chromosomes between generations. For PSP problem, we can use four kinds of mutation as mention in figure 4.

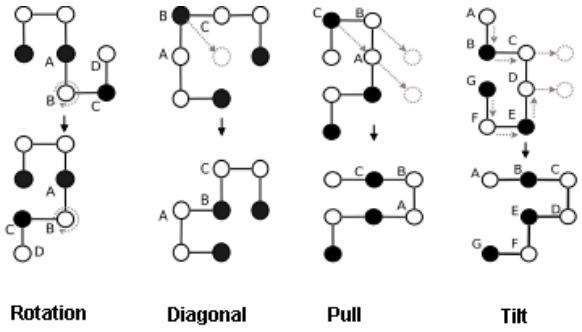


Fig. 4. Mutation methods for PSP problem

C. Quantum genetic algortihm

As mentioned above, quantum genetic algorithm (QGA), is combination of quantum computation and genetic algorithm. QGA can be used for kinds of problems that GA can be used, but QGA speeds up computation of genetic evolutionary process, has less population size and more powerful in global search. In quantum computation, qubit is the basic unit of information and is not deterministic; i.e., it does not have fixed value. The basic state may be in the $|0\rangle$ basis state or the $|1\rangle$ basis state, or in any superposition of the two, so the number of quantum state built of qubits are more than the classical one. The state of a qubit can be represented as

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle \qquad (2)$$

with a normalization constraint

$$|\alpha|^2 + |\beta|^2 = 1 \qquad (3)$$

where α and β are complex numbers and specify the probability amplitudes of the corresponding states. The probability that shows qubit in $|0\rangle$ is $|\alpha|^2$ and the probability that shows qubit in $|1\rangle$ is $|\beta|^2$.

In QGA, each chromosome includes array of qubit, so, each gene is a qubit. Equation 4, shows the encoding of chromosome which has n genes.

$$[qubit_1|qubit_2|\dots|qubit_n] = \begin{bmatrix}\alpha_1 & \alpha_2 & \dots & \alpha_n \\ \beta_1 & \beta_2 & & \beta_n\end{bmatrix} \quad (4)$$

where $qubit_i = \alpha_i|0\rangle + \beta_i|1\rangle$, $|\alpha_i|^2 + |\beta_i|^2 = 1$, $i = 1,2,\dots,n$.

The steps of QGA are combination of quantum computation and genetic algorithm, so, there are classic computation and quantum computation in QGA. In QGA we have several generations, and we index each generation by t, also we have two kinds of population, classic population and quantum population. i.e., P(t) means classic population at generation t, Q(t) means quantum population at generation t. Steps of QGA are showed in figure 5. In classic population, genes of chromosomes are classic but in quantum population, genes of chromosomes are qubit. Steps of figure 5 which have star (*), it means those steps have quantum computation, other steps without star, have classical computation like GA.

Fig. 5. Flow diagram of quantum genetic algorithm.

In initial quantum population step of QGA, every gene of each chromosome have equal probability to be in the state of $|0\rangle$ and$|1\rangle$. This means that at the end of the initialization, each qubit is in the state

$$|qubit\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle \qquad (5)$$

because this representation has an advantage that it is able to represent any superposition of states, so we have all search space of the problem in each chromosome. For example, a four-qubit system with four pairs of amplitudes such as

$$\begin{bmatrix}\frac{1}{\sqrt{2}} & 1.0 & 0.0 & \frac{1}{2} \\ 1 & 0.0 & 1.0 & \frac{\sqrt{3}}{2} \\ \frac{1}{\sqrt{2}} & & & \end{bmatrix} \qquad (6)$$

the state of the system can be represented as

$$\frac{1}{2\sqrt{2}}|0010\rangle + \frac{\sqrt{3}}{2\sqrt{2}}|0011\rangle + \frac{1}{2\sqrt{2}}|1010\rangle + \frac{\sqrt{3}}{2\sqrt{2}}|1011\rangle \qquad (7)$$

The above result means that the probabilities to represent the state $|0010\rangle$, $|0011\rangle$, $|1010\rangle$ and $|1011\rangle$ are $\frac{1}{8}$, $\frac{3}{8}$, $\frac{1}{8}$ and $\frac{3}{8}$ respectively.

III. PRPOPOSED APPROACH

As mention above, we proposed new approach to find optimized solution of PSP problem using QGA, so in this section we describe steps of QGA for finding optimized solution of PSP problem. It is important to mention, we describe quantum steps of QGA which includes encoding chromosomes, initial quantum population, observing quantum population to make classic population and generate next quantum population. The rest of steps which are classical computation can be used like existing GA approaches like fitness function and termination conditions.

A. Encoding chromosomes in QGA

The first step of QGA is encoding chromosome. As mentioned, in one of encoding method for PSP, each chromosome has $N-1$ genes, and each gene can be assignment over the symbols = $\{U, R, D, L\}$, but in QGA the basic information unit is qubit that may be in the $|0\rangle$ basis state or the $|1\rangle$ basis state, or in any superposition of the two, so we have to present new approach to encode four movement symbols using qubit. We use 2 qubits to encode four symbols in QGA according to table 1, so we have $2(N-1)$ gene in each chromosome in QGA because we use two qubits for each movement, and we have $N-1$ movements. Figure 6 shows an example of encoding methods in QGA.

Movement	Decimal	Binary
Up (U)	0	00
Down (D)	1	01
Right (R)	2	10
Left (L)	3	11

Table. 1. Encoding of gene

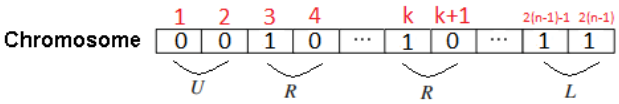


Fig. 6. Classic chromosome that obtain from observing Quantum chromosme and each two bits represent a movement

B. *Initail quantum population*

All genes in QGA are qubits, so we can store all search space of the problem in each quantum chromosome. As mention above, a qubit can be in any superposition of the $|0\rangle$ basis state and the $|1\rangle$ basis state, so if we initial each gene of QGA by setting all probability state amplitudes of each gene to be equal to one another, we can store all search space of problem in each quantum chromosome. Figure 7 shows an example.

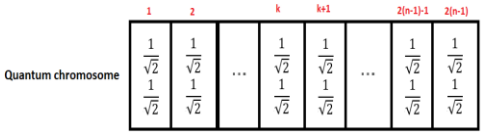


Fig. 7. An initialed quantum chromosome

C. *Observing quantum population*

In this step of QGA, we create a classical population by observing each gene of the chromosome in the quantum population. In this new algorithm, our mentioned pseudocode is defined over n qubits (Algorithm 1), i.e., in this paper $N = 2(n-1)$ and n is amino acid sequence length as described above.

```
(1) for i in 1,...,N do
(2)   r ← random number in range [0, 1]
(3)   if r < [a_i]^2 then
(4)     p ← 0
(5)   else
(6)     p ← 1
(7)   end if
(8) end for
```

Algorithm. 1. Qubit state observation

D. *Reproduction next quantum generation*

The last step of QGA is reproduction of next quantum generation. This step is equal to selection of parent, crossover and mutation which used in GA to generate next population. A lookup table will usually be used in QGAs to generate the next quantum population. In this approach, rotation gate will be used to update qubits of chromosome by comparing each of observed current generation quantum chromosome to best founded classic chromosome based on the lookup table. Rotation and sign of rotation gate will be defined by ~~a~~ the lookup table. But there is another method to update qubits in QGA. To describe this method, we assume that we have a qubit like figure 8 that is ith gene of one of quantum chromosome. The vertical lines show the probability amplitudes α and β, and ith gene of the best chromosome which is a classic chromosome, is b which can be valued by 1 or 0. We assume that b is valued by one, so we will multiply amplitude of α by the factor μ, which is a range between 0 and 1, and increase amplitude of β according to Equation 3. The described method eliminates the necessity to use a lookup table, which exists in the traditional QGA. The pseudocode for this method shows in algorithm 2.

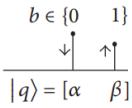


Fig. 8. A simple qubit

```
(1) for i in 1,...,N do
(2)   bestamp ← ith gene of the best
(3)   sum ← 0
(4)   for amp in {0, 1} do
(5)     if amp ≠ bestamp then
(6)       q'[amp] = μ · q
(7)       sum ← sum + q'[amp]^2
(8)     end if
(9)   end for
(10)  q'[bestamp] ← √(1 − sum)
(11) end for
```

Algorithm. 2. Generate next quantum computation

E. *Add Quantum Disaster Operation.*

The algorithm may fall into local optimal solution, while the algorithm has performed several generations and the best quantum chromosome is in a stable state. We need to take the quantum disaster operation to get out of the local optimal solution. The method is to apply a large disturbance to some quantum chromosomes in the population and regenerate some other new random quantum chromosomes. Chromosomes disaster process pseudocode program is described as in Algorithm 3.

```
Begin
   If (disaster-condition)
   Begin
   If (The chromosome is not the best chromosome)
   Initialize the chromosome;
   End
End
```

Algorithm. 3. Quantum disaster operation

IV. SIMULATION

QGA based on quantum system implement using python programming language. The simulations have been performed on Intel Core i5, 6.0 GB RAM and Windows 10.

In this section of paper, we simulate proposed approach, show result of simulation and compare QGA approach to find solution of PSP with genetic algorithms [14][19]. In table 2, there are 6 selected HP instances which are standard benchmarks used to test the ability of the algorithms to find optimized solution of the problem. The optimal or best-known free energy of these instances shows in table 2.

Sequence	Length	Protein sequence	Energy
1	20	$(HP)_2PH(HP)_2(PH)_2HP(PH)_2$	-9
2	24	$H_2P_2(HP_2)_6H_2$	-9
3	25	$P_2HP_2(H_2P_4)_3H_2$	-8
4	36	$P(P_2H_2)_2P_5H_5(H_2P_2)_2P_2H(HP_2)_2$	-14
5	48	$P_2H(P_2H_2)_2P_5H_{10}P_6(H_2P_2)_2HP_2H_5$	-23

Table. 2. The 2D HP benchmarks.

Before to start describing results of our proposed approach, we should mention, the power of QGA to find optimized solution of problem depends on population size and parameter μ . Parameter μ plays a key role in algorithm. The balance between global and local search to find solution of problem depends on Parameter μ. Population size is also another key of QGA. Finding not optimized solution for the problem is one of the main results of having less population and otherwise the time consumed for the running of algorithm directly related to the size of population. Therefore, we simulated affection of population size and parameter μ to find optimal solution of sequence number one of table 2. We simulated parameter μ between 0.5 to 0.98 and increased it by step 0.002 with population size 20,40 and 60. The iteration of simulation equaled to 100, and we considered average output of 10 times of parameter μ. Figure 9 shows the result.

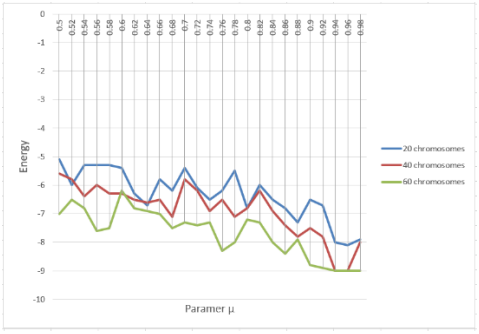


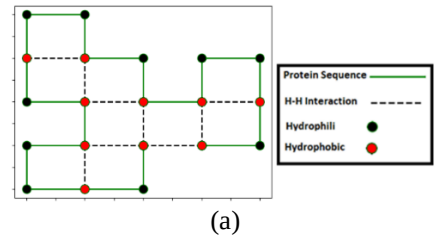
Fig. 9. Affection of parameter μ and population size

As result of Figure 9 on sequence number 1 of table 2, we can say less population size cannot find global optimized solution of problem, and it may fall into local optimal solution. Large population size can find global optimized solution, but it consumes much more time to complete computation. So, we recommend to set population size between 40 and 50. As you can see in figure 9, parameter μ between 0.94 and 0.96 can find better optimal solution.

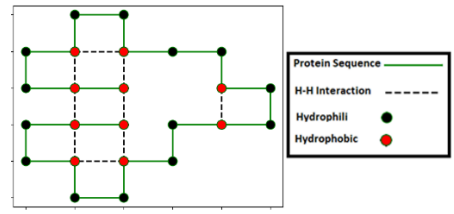
We simulated sequences of table 2 with population size equals to 50 and parameter μ equals to 0.96. Table 3 shows our results.

Sequence		GA	QGA	
1	Time(sec)	-	19.069473505020142	Figure 10 (a)
	comformation scaned	30,492	1,117	
	Optimal	-9	-9	
	Solution	-	DLLURULUURDRDRURDDL	
2	Time(sec)	-	39.961580753326416	Figure 10 (b)
	comformation scaned	30,491	2,192	
	Optimal	-9	-9	
	Solution	-	LLURURDRRDRDLLDLDLULURR	
3	Time(sec)	-	174.38035345077515	Figure 10 (c)
	comformation scaned	20.400	9,206	
	Optimal	-8	-8	
	Solution	-	UURULUUULDDLULDDRDLDRRUU	
4	Time(sec)	-	668.992345180511486	Figure 10 (d)
	comformation scaned	301,339	32,313	
	Optimal	-14	-14	
	Solution	-	ULDLULDLLLDRDRURDRURRDLDR DLLULDLULD	
5	Time(sec)	-	2,568.93060549316410624	Figure 10 (e)
	comformation scaned	126.547	114,081	
	Optimal	-23	-23	
	Solution	-	DLDLULDLULLLURURDRURRULLL LULURRDRURDRURDRDLDDLL	

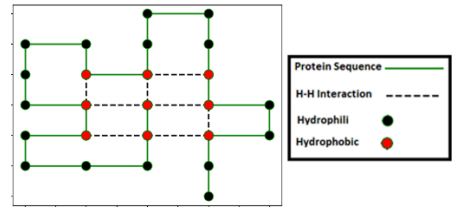
Table. 3. Details of QGA results for 6 selected HP instances



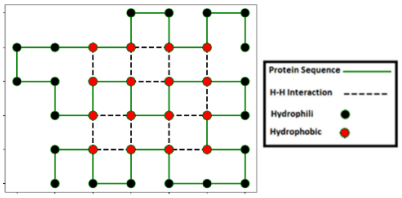
(a)



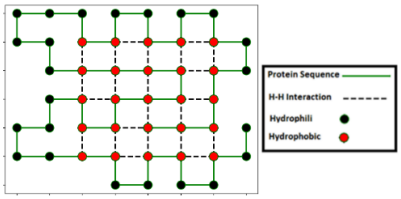
(b)



(c)



(d)



(e)

Fig. 10. QGA results for 6 selected HP instances

V. CONCLUSION

This paper proposed new approach to find optimized solution of 2D HP protein folding using quantum genetic algorithm. QGA combination quantum computation and genetic evolutionary process, so, the introduced QGA can find solution of PSP faster and more memory less than GA method. There are classical and quantum steps in QGA. In this paper we just introduced quantum steps of QGA which includes encoding chromosomes, initial chromosomes, observing quantum chromosomes and update quantum population. We use GA methods for classical steps in GA, like termination conditions and fitness function. In future work, we intend to find solution of the prediction of 3D structures for protein folding using QGA and compare result by GA approach.

REFERENCES

[1] H. J. Morowitz, Energy flow in biology. Academic Press, 1968.

[2] Stouthamer, "A theoretical study on the amount of ATP required for synthesis of microbial cell material," Antonie van Leeuwenhoek, vol. 39, no. 1, pp. 545–565, 1973

[3] Adam Smith, "Protein misfolding," Nature Reviews Drug Discovery, vol. 426, no. 6968, pp. 78–102, December 2003.

[4] Dill, K. A. (1985). Theory for the folding and stability of globular proteins. Biochemistry, 24(6), 1501–1509

[5] M. Dobson, "Protein folding and misfolding," Nature, vol. 426, no. 6968, pp. 884–890, 2003.

[6] F. Chiti and C. M. Dobson, "Protein Misfolding, Functional Amyloid, and Human Disease," Annu Rev Biochem, vol. 75(1), pp. 333–366, 2006.

[7] Jucker Mathias and Walker Lary C., "Self-propagation of pathogenic protein aggregates in neurodegenerative diseases," Nature, vol. 501, no. 7465, pp. 45–51, 2013.

[8] C.B. Anfinsen, Principles that govern the folding of proteins, Science 181 (1973) 187.

[9] A. Yonath, "X-ray crystallography at the heart of life science," Current Opinion in Structural Biology, vol. 21, no. 5, pp. 622–626, 2011

[10] Davis, L. Handbook of Genetic Algorithms; Van Nostrand Reinhold: New York, NY, USA, 1991.

[11] Holland, J.H. Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence; MIT Press: Cambridge, MA, USA, 1992.

[12] Filipi˘c, B.; Juri˘ci´c, D. An interactive genetic algorithm for controller parameter optimization. In Artificial Neural Nets and Genetic Algorithms; Springer: Berlin/Heidelberg, Germany, 1993; pp. 458–462

[13] Grefenstette, J.J. Optimization of control parameters for genetic algorithms. IEEE Trans. Syst. Man Cybern. 1986, 16, 122–128

[14] R. Unger, J. Moult, Genetic algorithms for protein folding simulations, J. Mol. Biol. 231 (1) (1993) 75–81, doi:10.1006/jmbi.1993.1258.

[15] A.L. Patton, W.F. Punch III, E.D. Goodman, A standard GA approach to native protein conformation prediction. In: L.J. Eshelman (Ed.), Proceedings of the 6th International Conference on GeneticAlgorithms,MorganKaufmann Publishers, San Francisco, 1995, pp. 574–581.

[16] M.M. Khimasia, P.V. Coveney, Protein structure prediction as a hard optimization problem: the genetic algorithm approach, Mol. Simul. 19 (1997) 205–226.

[17] Bui, T. N., & Sundarraj, G. (2005). An efficient genetic algorithm for predicting protein tertiary structures in the 2D HP model. In Proceedings of the 2005 conference on Genetic and evolutionary computation (GECCO'05) (pp. 385– 392).

[18] Cordon, O., Herrera, F., Hoffmann, F., & Magdalena, L. (2001). Genetic fuzzy systems evolutionary tuning and learning of fuzzy knowledge bases. Advances in fuzzy systems-applications and theory (Vol. 19). NJ: World Scientific Publishing.

[19] Cheng-Jian Lin, Ming-Hua Hsieh,An efficient hybrid Taguchi-genetic algorithm for protein folding simulation,Expert Systems with Applications,Volume 36, Issue 10,2009,Pages 12446-12453,ISSN 0957-4174,https://doi.org/10.1016/j.eswa.2009.04.074.

[20] Wang S., Wu L., Huo Y., Wu X., Wang H., Zhang Y. (2016) Predict Two-Dimensional Protein Folding Based on Hydrophobic-Polar Lattice Model and Chaotic Clonal Genetic Algorithm. In: Yin H. et al. (eds) Intelligent Data Engineering and Automated Learning – IDEAL 2016. IDEAL 2016. Lecture Notes in Computer Science, vol 9937. Springer, Cham. https://doi.org/10.1007/978-3-319-46257-8_2

[21] Shor, P.W. Algorithms for quantum computation: Discrete logarithms and factoring. In Proceedings of the 35th IEEE Annual Symposium on Foundations of Computer Science, Santa Fe, NM, USA, 20–22 November 1994; pp. 124–134.

[22] Grover, L.K. A fast quantum mechanical algorithm for database search. In Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing, Philadelphia, PA, USA, 22–24 May 1996; pp. 212–219.

[23] Miao, H.; Wang, H.; Deng, Z. Quantum genetic algorithm and its application in power system reactive power optimization. In Proceedings of the 2009 IEEE International Conference on Computational Intelligence and Security, Beijing, China, 11–14 December 2009; pp. 107–111.

[24] Laboudi, Z.; Chikhi, S. Comparison of genetic algorithm and quantum genetic algorithm. Int. Arab J. Inf. Technol. 2012, 9, 243–249.

[25] Malossini, A.; Blanzieri, E.; Calarco, T. Quantum genetic optimization. IEEE Trans. Evol. Comput.