I.J. Image, Graphics and Signal Processing, 2011, 3, 41-48
Published Online April 2011 in MECS (http://www.mecs-press.org/)

DNA 3D Self-assembly Algorithmic Model to Solve Maximum Clique Problem

Jingjing Ma
Department of life science, Shaanxi Normal University, Xian, Shaanxi, China
Email: casy-20@126.com

Li Jia and Yafei Dong
Department of life science, Shaanxi Normal University, Xian, Shaanxi, China
Email: dongdongfeia@126.com
Department of life science, Department of computer science, Shaanxi Normal University, Xian, Shaanxi, China
Email: dongyf@snnu.edu.cn

***Abstract*—Self-assembly reveals the essence of DNA computing, DNA self-assembly is thought to be the best way to make DNA computing transform into computer chip. This paper introduce a method of DNA 3D self-assembly algorithm to solve the Maximum Clique Problem. Firstly, we introduce a non-deterministic algorithm. Then, according to the algorithm we design the types of DNA tiles which the computation needs. Lastly, we demonstrate the self-assembly process and the experimental methods which could get the final result. The computation time is linear, and the number of the distinctive tile types is constant.**

***Index Terms*—DNA self-assembly, DNA computing, Maximum Clique Problem，3D**

I. INTRODUCTION

Self-assembly process is ubiquitous in nature. Systems form on all scales via self-assembly: atoms self-assemble to form molecules, molecules to form complexes, and stars and planets to form galaxies.

In biological systems, cells are self-assembled by all sorts of biological molecules, meanwhile biological organisms use biochemical approaches to control every molecular and chemical activity, such as the storage and reproduction of genetic information, the control of developmental processes, even the sophisticated computations performed by the nervous system.

Molecular computation is a sort of biochemical algorithm, which finds out the connection between the natural biochemical process and the human process which the electronic microprocessors control electro-mechanical devices. Molecular computation is a new computing method comparing the traditional one, it has two complementary perspectives: using the astounding parallelism of chemistry to solve mathematical problems, such as combinatorial search problems; and using the biochemical algorithms to direct and control molecular processes, such as complex fabrication tasks.

DNA is the genetic material of the most biological organisms, and it is also the carrier of genetic message in itself. Molecular computation uses the massive information storage capacity of DNA and the huge parallelism of biochemical reaction to solve many difficult problems which the traditional computer cannot solve.

DNA computing essentially could be divided into three categories: intramolecular, intermolecular and super molecular DNA computing. Takahashi's [1] studies are concerning intramolecular DNA computing, which use only a single DNA molecule. Intramolecular DNA computing is like Adleman's experiments, which are hybridisation of different DNA molecules. Super molecular DNA computing is a process of DNA self-assembly which meanwhile is an algorithm, which is firstly stated by Winfree [2].

Self-assembly reveals the essence of DNA computing, and in a certain extent it avoids the error accumulation caused by the frequent operation in the experimental process, its production is also easy to be analyzed. DNA self-assembly is thought to be the best way to make DNA computing transform into computer chip.

II. THE PRINCIPLE OF DNA SELF-ASSEMBLY ALGORITHM

Adleman's original paper [3] is the beginning of DNA computing, from then on a lot of researches have been made following his method, which use the programmability of DNA hybridization reactions to direct computation according to desired rules. But this linear self-assembly needs more experimental operation and it can only compute some simple questions.

In order to find an algorithm with universal computational ability, Winfree figured out a two-dimensional algorithm of DNA self-assembly, according to Wang's Tiling theory [4] in geometry.

The famous tiling problem is discovered provably unsolvable by Hao Wang [5].To prove that Wang developed a way to create a set of tiles (Fig.1) that fit together uniquely to reproduce the space-time history of any chosen Turing machine.

Wang's method shows that tiling is theoretically as powerful as general-purpose computers. In fact, the tiles which Wang used were all essentially square, distinguished only by markings on their sides that are required to match when tiles are juxtaposed. Thus the way how the tiles fit together is the key to simulate Turing machine.

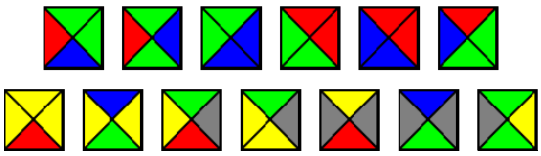


Figure 1.Wang tiles with different colored edges

Coincidentally, the tiles which Wang used can be simulated by DNA nanostructures, which are the production of DNA nanotechnology and are called DNA tiles. Fig.2 shows the basic DNA tiles, they have sticky ends, which can combine together according to the rules of base complementary pairing when they are matched.

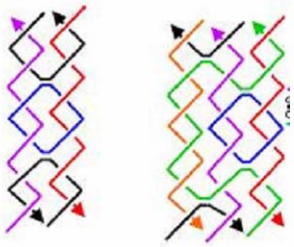


Figure 2.The left is DX molecule; the right is TX molecule.

Winfree figured out a method of 2D self-assembly using DX molecules as molecular Wang tiles and realized a calculation in the growth of crystal. The four arms of the DX molecules can be given sequences corresponding to the labels on the four sides of the Wang tiles. Thus, any chosen Wang tile can be implemented as a DNA tile. Appropriate design of the molecules will encourage assembly into two-dimensional sheets. Because Turing machines and cellular automata can be simulated by this process, so the Turing-universality of tiling is retained.

The 2D self-assembly algorithm offers new capabilities for computation and construction, at the same time it offers a new range of physical phenomena and experimental challenges.

The current state-of-the-art of DNA nanostructure design and implementation with DNA origami, which folds a single long scaffold strand into an arbitrary shape by using small helper strands, have been demonstrated by Rothmund [6]. Similar concepts may be the key to three-dimensional self-assembly and more powerful error-correction techniques.

III. THE MODELS OF DNA SELF-ASSEMBLY ALGORITHM

There are three types of the DNA self-assembly algorithm model: linear model, tile assembly model and 3D self-assembly model.

In 1999 Adleman presented a mathematic model of DNA self-assembly which is called ‘step counting’ model in [7]. Adleman’s model is a model of linear self-assembly, so it supposes that each tile has two gules only in the east and west sides. In 2000 Adleman [8] made a modification to the model and gave a formal definition.

Tile assembly model is well defined by Paul Rothemund and Erik Winfree [9], which is a 2D DNA self-assembly model. The model mainly has four parts:

1. Basic tile types: they are used to construct different kinds of arithmetic operators, which can store the numerical value of the computation. Every edge of a tile has a label to represent different numerical value or symbols;
2. The strength function: to define the strength of the binding domains;
3. The seed configuration: to define the start of a self-assembly body;
4. Temperature τ: to denote the thermodynamic parameter, at “temperature” τ, an aggregate of tiles can grow by addition of a monomer whenever the summed strength of matching edges exceeds τ (mismatched labels neither contribute nor interfere) – these are called stable additions.

According to the Chomsky’s hierarchical system of language, Winfree [10] defined the relation between self-assembly and formal language,

1. Linear Self-Assembly is Equivalent to Regular Languages: Linear Self-Assembly is the self-assembly of duplex DNA while single sticky end bond with the formation of linear DNA complexes.
2. Dendrimer Self-Assembly is Equivalent to Context-Free Languages: Dendrimer Self-Assembly is the self-assembly of hairpin and duplex DNA while single sticky end bond with the formation of linear DNA complexes.
3. Two Dimensional Self-assembly is Universal and it is equivalent to recursively enumerable languages: the Two dimensional self-assembly is the self-assembly of Double Crossover (DX) units while double sticky end bond and the temperature is critical with the formation of 2D DNA complexes.
4. Three Dimensional Self-Assembly Augments Computational Power: Winfree indicated a trivial corollary of the universality of two-dimensional self-assembly, which is that if three dimensional structures are allowed, self-assembly is still universal. His prophesy is proved by the following researches. It is of greater interest if we can exploit all three dimensions to allow for more efficient or more reliable computations.

A. 3D DNA nanostructure for self-assembly

The first work to combine studies of self-assembly with nanotechnology in three dimensions is completed by Kao and Ramachandran [11]. They proposed a general mathematical model for constructing 3D structures from 2D tiles. Their model is a more precise superset of 2D Tile Assembly Model that facilitates building scalable 3D molecules. Under the model, they presented algorithms to build a hollow cube (Fig.3), which is intuitively one of the simplest 3D structures to construct. They built the 3D structures using the folding technique shown in Fig.4 and allow construction of all 2D structures possible with the Tile Assembly Model. They are the first to extend nanostructure fabrication to three dimensions.

Their methods are applicable to more complex 2D and 3D nanostructures. But there are some important biological issues. In particular, design of a strong and rigid DNA tile suitable for computation and design of a 3D building block are two important points to increase the feasibility of 3D self-assembly. Besides the

temperature may be further refined and exploited to improve some complexity results and the number of steps needed in the lab.

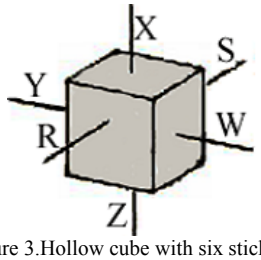


Figure 3.Hollow cube with six sticky ends

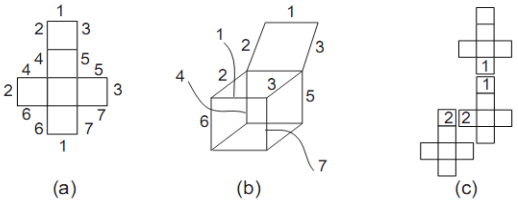


Figure 4.(a) 2D planar shape that will fold into a box. Each section is formed from many smaller 2D DNA tiles. Edges with the same number have complementary sticky ends exposed so they can hybridize. (b) Folding of the shape in (a) into a box. Here, edges 4, 5, 6, and 7 have all hybridized. Hybridization of edges 2 and 3, whose two complements are now in close proximity, will cause edge 1 to hybridize and form the complete box. (c) Multiple copies of the 2D shape in solution. Copies of the shape can interfere and attach infinitely without control as long as edges have matching sticky ends.

B. 3D DNA self-assembly model

Lately Essam Al-Daoud et al [12] presented a new method to perform the vector and integer multiplication by using the self-assembly of 3D DNA nanostructure. Their method simulates the vector multiplication by using a look up table that represents many pairs of the vectors. The multiplication of each corresponding numbers can be performed by adding the carry and the result in each internal 3D tile of the assembled superstructures. The final result can be detected in the first row of the sum layer, where the sum layer uses the sticky ends to accumulate the result from each vertical layer. Their procedure is less energy consumer and can be used at very low cost.

Then Lin Minqi [13] invented a 3D DNA self-assembly model to solve the Graph Vertex Coloring problem. The model can simulate a non-deterministic algorithm and solve the problem in linear time: $\Theta(n)$. The number of distinct tiles used in the model is $\Theta(k^2)$, where k is the size of the color set. For the vertex 3-coloring problem, the model requires only 22 types of distinct tiles. We believe that before long this kind of 3D DNA self-assembly algorithm will probably be simulated and executed in lab.

The 3D self-assembly tile is a structure with 6 sticky ends which can assemble in 3D space and is different from the 2D square tile. Its molecular model is as Fig.5(a). Intuitively, the 3D self-assembly tile is a hexahedron, with each surface represents a direction in the 3D space coordinate system, such as the positive or negative direction of x, y, z axes. The surfaces with sticky ends could connect with each other according to their combining domains. When their adjacent combining domains are matched, and the summed strength of matching domains exceeds τ, the two adjacent surfaces could be connected. A tile's type is decided by its six combining surfaces. Fig.5 (b) demonstrates the structure of the abstract hexahedron.

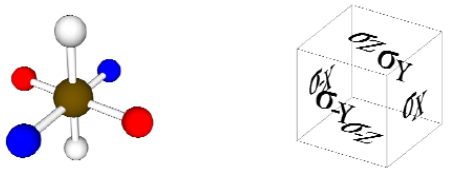


Figure 5.(a) The model of 3D DNA structure. (b) The abstract hexahedron tile.

As an extension of the 2D Tile assembly model, Lin Minqi [13] also gave a formal definition of the 3D DNA self-assembly model.

Definition 1: Let Σ be a finite alphabet of binding domains such that $null \in \Sigma$. A **tile** over a set of binding domains Σ is a 6-tuple $(\sigma_X, \sigma_{-X}, \sigma_Y, \sigma_{-Y}, \sigma_Z, \sigma_{-Z}) \in \Sigma^6$. A special tile $empty=(null,null,null,null,null,null)$ represents the absence of all other tiles. Let T represents the set of tiles including the empty tile.

Definition 2: A **position** is an element of $\mathbb{Z}^3$. The set of directions $D=\{X,X^{-1},Y,Y^{-1},Z,Z^{-1}\}$ is a set of six functions from positions to positions, i.e. $\mathbb{Z}^3$ to $\mathbb{Z}^3$ such that for all positions (x, y, z), $X(x, y, z) = (x+1, y, z)$, $Y(x, y, z) =(x, y+1, z)$, $Z(x, y, z)= (x, y ,z+1)$. The positions (x, y, z) and (x', y', z') are neighbors iff $\exists d \in D$ such that $d(x, y, z) = (x', y',z')$. For a tile t , for $d \in D$, $bd_d(t)$ is the binding domain of tile t on d's side.

Definition 3: A strength function $g:\Sigma\times\Sigma\rightarrow\mathbb{R}$. $\forall\sigma, \sigma' \in \Sigma$, $g(\sigma, \sigma')= g(\sigma', \sigma)$ and $g(null, \sigma) = 0$. It is common to assume that $g(\sigma,\sigma')=0 \quad \sigma\neq\sigma'$. This simplification of the model implies that the abutting binding domains of two tiles have to be matched to bind. Formally,

$$g(\sigma, \sigma')=\begin{cases}\hat{g}(\sigma) & \text{if } \sigma = \sigma' \\ 0 & \text{otherwise}\end{cases} \qquad (1)$$

Definition 4: Let T be a set of tiles containing the empty tile. A **configuration** of T is a function $A: \mathbb{Z}^3\rightarrow T$. $(x, y, z)\in A$ iff $A(x, y, z)\neq$empty. A is finite iff there is only a finite number of distinct positions $(x, y, z) \in A$.

Finally, a **tile system** $\mathbb{S}$ is a 3-tuple (T, g, τ), where T is a finite set of tiles containing empty, g is a strength function and $\tau\in\mathbb{N}$ is the temperature.

Definition 5: If A is a configuration, then within system $\mathbb{S}$, a tile t can attach to A at position (x, y, z) and produce a new configuration A' iff:

- $(x,y, z) \quad A$, and
- $\sum_{d\in D} g(bd_d(t), bd_{d-1}(A(d(x,y,z)))) \geq \tau$, and
- $\forall(u,v,w)\in\mathbb{Z}^3$, $(u,v,w)\neq(x,y,z) \quad A'(u,v,w)=A(u,v,w)$, and
- $A'(x,y,z)=t$.

That is, a tile can attach to a configuration only in empty positions and only if the total strength of the appropriate binding domains on the tiles in neighboring positions meets or exceeds the temperature τ. For example, if g=1 and τ= 2 then a tile t can attach only at

positions with matching binding domains on the tiles in at least two adjacent positions.

Definition 6: Given a tile system $\mathbb{S} = (T, g, \tau)$, a set of tiles Γ, and a **seed configuration** S: $\mathbb{Z}^3 \rightarrow \Gamma$, if the above conditions are meet, one may attach tiles of T to S. Configurations **produced** by repeated attachments of tiles from T are said to be produced by $\mathbb{S}$ on S. If this process terminates, then the configuration achieved when no more attachments are possible is called the **final configuration**. At some times, it may be possible for more than one tile to attach at a given position, or there may be more than one position where a tile can attach. If for all sequences of tile attachments, all possible final configurations are identical, then $\mathbb{S}$ is said to produce a **unique** final configuration on S.

IV. THE MAXIMUM CLIQUE PROBLEM

Given an undirected graph G = (V; E), V is the set of vertexes and E is the set of edges. Two vertexes are said to be adjacent if they are connected by an edge. A clique of a graph is a set of vertexes, any two of which are adjacent. Maximum clique is a clique whose vertexes are not a subset of a larger clique, in other words, it is the largest among all cliques in a graph.

Maximum Clique Problem is an NP-complete problem. There are many algorithms to solve the MCP. But all the algorithms we have known have an exponential complexity. For some large-scale problems, the algorithms need an unrealistic computing time. Guangzhao Cui [14] put forward a method using the tile assembly model to solve the Maximum Clique Problem, which only needs linear time to get the result. This paper using the 3D self-assembly model to solve the Maximum Clique Problem.

V. IMPLEMENTATION OF DNA 3D SELF-ASSEMBLY MODEL ON MAXIMUM CLIQUE PROBLEM

When we use the DNA self-assembly model to solve the practical problems, firstly we should design a DNA self-assembly system and an algorithm according to the specific problem. The DNA tiles based on the DNA bases' complementary pairing rules according to the specific algorithm we have designed assemble together, when the process of self-assembly finish then we can get the result of the specific problem. For some NP-complete problems such as MCP, the massive information storage capacity and the huge parallelism of the biochemistry reaction in the self-assembly process, and the strictness of self-assembly rules can reduce these problems' complexity greatly.

Infra we demonstrate the specific process using the DNA 3D self-assembly model to solve the MCP. We take a specific graph for example. Suppose G = (V，E) is an undirected graph, V is the set of vertexes and E is the set of edges (Fig.6). According to G's adjacent matrix we can get G's adjacent table T_a (TABLE I).

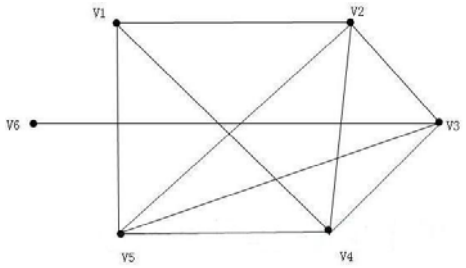


Figure 6. An undirected graph G

TABLE I
GRAPH G'S ADJACENT TABLE T_a

	1	2	3	4	5	6
1	1					
2	1	2				
3	0	1	3			
4	1	1	1	4		
5	1	1	1	1	5	
6	0	0	1	0	0	6

In T_a the first line, the first row and the principal diagonal of the table denote the vertexes of the graph, the cells of the below triangle denote the adjacent condition among the vertexes, if two vertexes are connected by an edge then the value of the cell is "1", otherwise it is "0". For example V_1 and V_2 are connected by an edge, so the value of the cell (1,2) is "1".

We define a function $C(i,j)$ to denote the adjacent relation of two vertexes,

$$C(i,j) = \begin{cases} 1 & if\ \{i,j\} \in E \\ 0 & otherwise \end{cases} \quad (2)$$

The non-deterministic algorithm of the MCP is as below:

maxClique

1) Random select N_x from V;
2) Check N_x if all $C(i,j) == 1$; output all kinds N_x to M;

Else return 1;

3) inprint the max N_x;

A. The design of the 3D DNA tiles

Fig. 7 demonstrates the tile types we have designed: a is the adjacent tiles, they are used for denoting the adjacent relation of vertexes; b is the passing tiles, they are used for passing the information of vertexes; c is the checking tile, it is used for checking and making sure that two certain vertexes are adjacent; d and e are input tiles, e denotes inputting a certain vertex, d denotes inputting an empty tile, that means in a certain inputting position, if an empty tile is added, the vertex which the certain position represents is missing; f is the output tiles; g is the boundary tiles, SS tile represents the success of the self-assembly, namely the completion of the computation.

According to the 3D tile's 6-turple form $(\sigma_X, \sigma_{-X}, \sigma_Y, \sigma_{-Y}, \sigma_Z, \sigma_{-Z}) \in \Sigma^6$ we have defined in the above paragraph, the forms of tiles we have designed are as follows:

1. The adjacent tiles

($\sigma_Z = 0,1; \sigma_X = \sigma_{-X} = \sigma_Y = \sigma_{-Y} = \sigma_{-Z} = null$)；containing two tile types, when σ_Z=1 denotes two vertexes are connected by

an edge; while σ_Z=0 denotes two vertexes are not adjacent. The adjacent tiles can build different seed configuration according to different graphs. Fig.7a denotes the two tile types, the hexahedron with the symbol "0" on the surface of σ_Z represent the cell with value "0" in the table T_a, while the hexahedron with the symbol "1" on the surface of σ_Z represent the cell with value "1" in the table T_a. The other surfaces of the tiles don't have any symbol.

2. The passing tiles

(σ_X=σ_{-X}=*i,0* ； σ_Y=σ_{-Y}=*0,i* ； σ_{-Z}=*0,1* ； σ_Z=*OK*) ； containing four tile types, they are responsible for passing the information of vertex i to all the possible adjacent vertexes. Fig.7b demonstrates two tile types (other two are left out). The surfaces of σ_{-Z} are symbolized by"0 or 1", the σ_X、σ_{-X} and σ_Y、σ_{-Y} are symbolized respectively by"i or 0"； the surface of σ_Z is symbolized by"OK".

3. The checking tile

(σ_X=σ_{-X}=*i*； σ_Y=σ_{-Y}= *j*； σ_{-Z}=*1*； σ_Z=*OK*)，containing one tile type, it is responsible for checking and making sure that two certain vertexes i and j are adjacent. Fig.7c demonstrates the checking tile's form, its σ_{-Z} are symbolized by "1", σ_X、σ_{-X} and σ_Y、σ_{-Y} are symbolized respectively by "i" and "j"; σ_Z is denoted by "OK".

4. The input tiles

Vertex input tile: (σ_X=σ_{-X}=σ_Y=σ_{-Y}=σ_Z=σ_{-Z}=*i*), containing one tile type, it can input the information of a vertex on a certain inputting position of the seed configuration. Fig.7e demonstrates its form, all of its surfaces are symbolized by "i" .

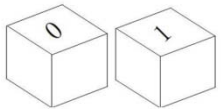
Empty tile: (σ_X=σ_Y =σ_Z= *0*； σ_{-X} =σ_{-Y}=σ_{-Z}=*i*), it can be inputted in any vertex inputting position of the seed configuration, which means the certain vertex is missing. Fig.7d demonstrates its form, the surfaces of σ_{-X}、σ_{-Y}、σ_{-Z} are all symbolized by "i", the surfaces of σ_X、σ_Y、σ_Z are all symbolized by "0".

5. The output tiles

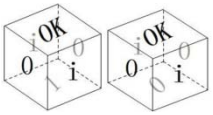
(*σ_X=σ_{-X}=&*； σ_Y= *null*； σ_{-Y} = *i,0*； σ_{-Z}=*#*； σ_Z=*null)*; containing four tile types, they are responsible for outputting the last result of the self-assembly. Fig.7f demonstrates two tile types (other two are left out).Their surfaces of σ_{-Y} are symbolized by "i or 0", the σ_X、σ_{-X} are symbolized by "&", σ_{-Z} is symbolized by "#". In the self-assembly process a set of output tiles are corresponding to a set of input tiles.

6. The boundary tiles

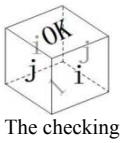
(*σ_Z =i,#;σ_X =σ_{-X} =σ_Y =σ_{-Y} =σ_{-Z} =null)*、(*σ_X =σ_Y =i; σ_{-X} =σ_{-Y}=σ_Z=σ_{-Z}=null)*、(*σ_X=&;σ_{-X}=σ_Y=σ_{-Y}=σ_Z=σ_{-Z}=null)*、(*σ_X=σ_Y= null;σ_{-Z}=#;σ_Z=SS;σ_{-X}=σ_{-Y}=&)*，containing five tile types, they can control the growth direction of the self-assembly. Fig.7g demonstrates their forms, the last tile with "#" on σ_{-Z}, "&" on σ_{-X}、σ_{-Y}, and "SS" on σ_Z represents the completion of the self-assembly process.



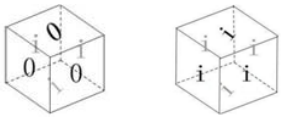
a The adjacent tiles



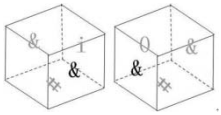
b The passing tiles



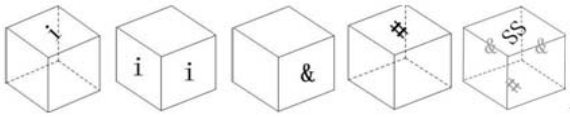
c The checking tile



d The empty tile e The vertex input tile



f The output tiles



g The boundary tiles

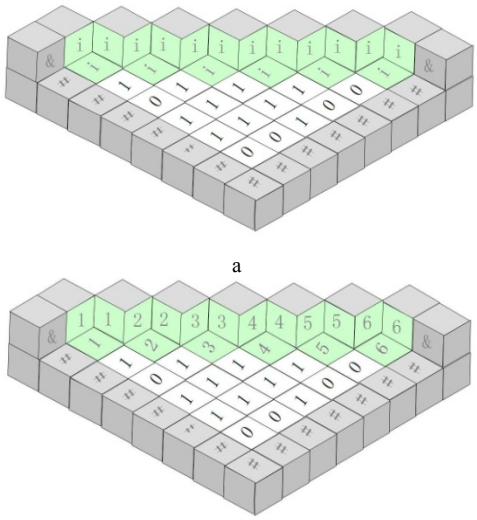
Figure 7. The basic tile types

B. Seed configuration

Fig. 8 demonstrates the seed configuration of the self-assembly, it is constructed by the adjacent tiles and the boundary tiles. It contains G's adjacent information, namely when two vertexes are connected by an edge, its value is "1", otherwise is "0". The green lattices denote the inputting positions, and they must be arranged in turn from left to right according to the vertexes' order (Fig.8b). So in a certain vertex inputting position only the certain vertex can input, others cannot. According to the tiles we have designed (Fig.7d e), every inputting position can connect an empty tile (Fig.7d), which represents a certain vertex is missing, and is denoted by "0".

C. The process of the self-assembly process

When the self-assembly start, the seed configuration can assemble with the input tiles (Fig.7d e) which represent the graph's vertexes via a non-deterministic self-assembly manner. The first step of the self-assembly process is to form the result non-deterministically. Then check up whether the result is correct or not via the self-assembly process. Only when the correct result is formed the self-assembly process could complete, or we cannot get a successful self-assembly system.



a

b

Figure 8. The seed configuration

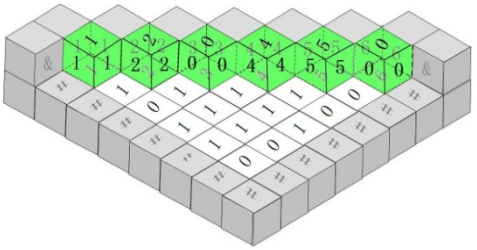
a) The successful self-assembly process

Fig.9a demonstrates the first step, the input tiles demonstrated in Fig.7d e are assembled to the seed configuration non-deterministically. Because the inputting positions in the seed configuration are arranged in turn according to the vertexes' order, the input tiles ($\sigma_X=\sigma_{-X}=\sigma_Y=\sigma_{-Y}=\sigma_Z=\sigma_{-Z}=i$) can only be assembled to the certain position, for example vertex V_1 ($\sigma_X=\sigma_{-X}=\sigma_Y=\sigma_{-Y}=\sigma_Z=\sigma_{-Z}=1$) can only be assembled to the most left inputting position in the seed configuration. Besides every inputting position in the seed configuration can assemble an empty tile ($\sigma_X=\sigma_Y=\sigma_Z=0$; $\sigma_{-X}=\sigma_{-Y}=\sigma_{-Z}=i$), representing the position misses a vertex, for example in the vertex V_3's position a empty tile($\sigma_X=\sigma_Y=\sigma_Z=0$; $\sigma_{-X}=\sigma_{-Y}=\sigma_{-Z}=3$) can be assembled, which means the vertex V_3 is not involved in, we use "0" denotes the missing vertex V_3. Just like this kind of non-deterministically self-assembly process, a result is formed, namely the vertexes set $\{V_1, V_2, V_4, V_5\}$.

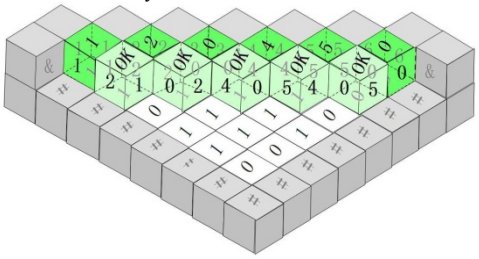
b demonstrates the second step of the self-assembly, it is to check up whether the result is correct or not. This process depends on the checking tiles ($\sigma_X=\sigma_{-X}=i$; $\sigma_Y=\sigma_{-Y}=j$; $\sigma_{-Z}=1$; $\sigma_Z=OK$) and the passing tiles ($\sigma_X=\sigma_{-X}=i,0$; $\sigma_Y=\sigma_{-Y}=0,i$; $\sigma_{-Z}=0,1$; $\sigma_Z=OK$)。For vertexes V_1, V_2, the check tile ($\sigma_X=\sigma_{-X}=1$; $\sigma_Y=\sigma_{-Y}=2$; $\sigma_{-Z}=1$; $\sigma_Z=OK$) can be successively assembled to the configuration. For vertexes V_2, V_0, the check tile ($\sigma_X=\sigma_{-X}=2$; $\sigma_Y=\sigma_{-Y}=0$; $\sigma_{-Z}=1$; $\sigma_Z=OK$) can be successfully assembled to the configuration. Following this manner, the checking tiles can check up all the vertexes' adjacent condition and successfully assemble together.

c demonstrates the step before the self-assembly is completed, the output tiles ($\sigma_X=\sigma_{-X}=\&$; $\sigma_Y=null$; $\sigma_{-Y}=i,0$; $\sigma_{-Z}=\#$; $\sigma_Z=null$) are assembled to the configuration.

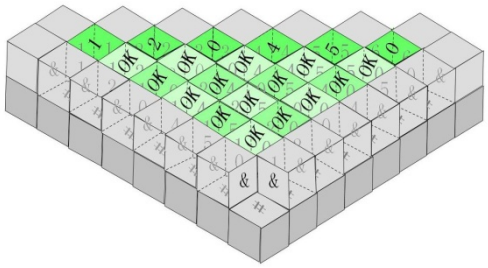
d demonstrates the last step, SS tile ($\sigma_X=\sigma_Y=null;\sigma_{-Z}=\#;\sigma_Z=SS;\sigma_{-X}=\sigma_{-Y}=\&$) is assembled to the configuration, which indicates the self-assembly process has complete successfully.



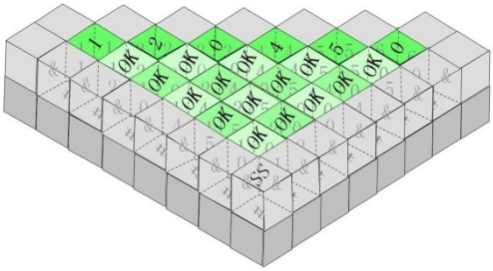
a The first step: the result is formed non-deterministically, $V_1V_2V_4V_5$ are chosen randomly



b The following steps: the checking tile and the passing tiles are assembled in.



c the step before the self-assembly is completed:the output tiles are assembled in.



d The last step:the self-assembly process is finished, the SS tile is assembled in.

Figure 9. The successful self-assembly case

a) The failed self-assembly case

Not all the results formed via non-deterministic self-assembly process are correct, these incorrect results can be detected in a certain step of the self-assembly process. Fig.10 demonstrates a failed self-assembly case. The vertexes set chosen non-deterministically is $\{V_1, V_2, V_3, V_5\}$, the yellow position demonstrates that if existing a tile ($\sigma_X=\sigma_{-X}=1$; $\sigma_Y=\sigma_{-Y}=3$; $\sigma_{-Z}=0$; $\sigma_Z=OK$) then it can be assembled to the configuration, the process can proceed, but we didn't design such a tile, that means the two vertexes are not adjacent, the checking tile cannot be assembled to the configuration, the self-assembly process is stopped. This is a failed case, namely the result is not correct.

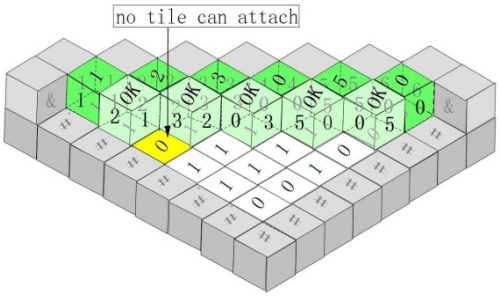


Figure 10. The failed self-assembly case, the yellow shows the checking tiles cannot attach.

D. The detecting and analyzing of the results

The DNA self-assembly process is a specific biochemistry reaction, when the process is completed, the problem's results exist among the self-assembly systems. We must analyze the successful configuration if we want the correct result of the problem.

In the model we have proposed all the successful configurations are G's clique, while the MCP can be obtained by some experimental techniques of biochemistry and molecular biology. For example, we can firstly separate the successful configuration via fluorescence probe technique and amplify the results via PCR technique, then obtain the DNA strand with the largest molecular mass through gel electrophoresis. After that we could also carry on more accurate analysis to ascertain the vertexes information of the Maximum Clique which the DNA strands encode.

VI. DISCUSSION

This paper is using the DNA 3D self-assembly algorithmic model to solve the Maximum Clique Problem. Firstly, we introduce a non-deterministic algorithm. Then, according to the algorithm we design the types of DNA tiles which the computation needs. Lastly, we demonstrate the self-assembly process and the experimental methods which could get the final result. The computation time is linear, and the number of the distinctive tile types is constant.

The model we proposed to solve the Maximum Clique Problem can greatly reduce the computation's complexity, and when the scale of the problem becomes larger, the method is also feasible. Besides the experimental techniques which we use to analyse the results are ripe. As the development of Biological technology, DNA self-assembly algorithm will have more promising applications. Using DNA molecules to made computer chip is under our expectation.

ACKNOWLEDGEMENT

This paper is supported by the key project of Shaanxi Normal University, the Shaanxi province Natural Science Fund 2007F46 and the National Natural Science Fund of China 60970005.The corresponding author is Yafei Dong, his major researching field is DNA molecular computing, e-mail address: dongyf@snnu.edu.cn.

REFERENCES

[1]. Takahashi K, *et al*. "Photo-and Thermoregulation of DNA Nanomachines". *In 11th Int. Mtg on DNA computing,* 147-156, 2005.

[2]. Erik winfree. "On the computational power of DNA annealing and ligation". In Lipton and Baum 1996, pages 199-221

[3]. Adleman LM. "Molecular computation of solutions to combinatorial problems". Science 1994;266:1021–4.

[4]. Wang H. "Proving theorems by pattern recognition I". Bell Syst Tech J 1961;40:1–42.

[5]. Wang, H. "Dominoes and the AEA case of the decision problem". In Mathematica Theory of Automata (ed. J. Fox). Polytechnic Press, New York. 1963.

[6]. Paul Rothemund, "Scaffolded DNA origami: From generalized multicrossovers to polygonal networks". Nanotechnology: Science and Computation 2006 pp3–21.

[7]. Leonard M. Adleman. "Towards a mathematical theory of self-assembly". *Technical Report*, Department of Computer Science, University of Southern California, 2000.

[8]. Leonard M. Adleman，et al. "Linear Self-Assemblies：Equilibria，Entropy and Convergence Rates"，2000.

[9]. Paul Rothemund and ErikWinfree. "The program-size complexity of self-assembled squares". STOC 2000.

[10]. Erik Winfree, Xiaoping Yang, and Nadrian C. Seeman. Universal computation via self-assembly of DNA: Some theory and experiments. In Landweber and Lipton.

[11]. M. Kao and V. Ramachandran "DNA Self-Assembly For Constructing 3D Boxes" *Lecture Notes in Computer Science,* Springer Berlin, Heidelberg, 2001, pp. 429-441.

[12]. Essam Al-Daoud, Belal Zaqaibeh and Feras Al-Hanandeh "3D DNA Nanostructures for Vector Multiplication", *American Journal of Scientific Research* ISSN 1450-223X Issue 1,2009.

[13]. Lin Minqi, Xu Jin, *et al* "3D DNA Self-Assembly Model for Graph Vertex Coloring" *Journal of Computational and Theoretical Nanoscience* Vol.7 No.1246-253,2010

[14]. Guangzhao Cui, *et al. "* Application of DNA Self-assembly on Maximum Clique Problem" *Advances in Soft Computing*,2009,Volume 116/359-368,2009.

Jingjing Ma was born in Yangquan city in Shanxi province of China. She got a medicine bachelor degree when she graduated from Sichuan University in 2006 which located in Chengdu city in Sichuan province of China. Now she is a graduate student in the college of life sciences in Shaanxi Normal University in Xian city in Shaanxi province of China. Her major field of study is DNA computing.

Li Jia was born in Datong city in Shanxi province of China. She got an engineer bachelor degree when she graduated from Shanxi Normal University in Linfen city in Shanxi Province of China. Now she is a graduate student in the college of life sciences in Shaanxi Normal University in Xian city in Shaanxi province of China. Her major field of study is DNA computing.

Yafei Dong was born in Xian city in Shaanxi province. He got a biology and electronics post doctorate degree and an engineer doctor degree when he graduated from the Huazhong

University of Science and Technology in Wuhan city in Hunan province of China.

At present he is an associate professor in the college of life sciences in Shaanxi Normal University in Xian city in Shaanxi province of China. He took charge in or took part in about eight programs on the study of DNA computing. The programs he took charge in are the National Natural Science Fund, number: 60574041, the second-class China National Post Doctorate Degree Premium, the Natural Science Fund of Shaanxi province, number: 2007F46, the Post Doctorate Fund of Huazhong University of Science and Technology. The programs he took part in are the National Natural Science Fund, number: 60573031, an item of the 863 plan of China, number: 2006AA01Z104, two items of the National Natural Science Fund, Number: 60103021, 60174047, an item of National Doctor Fund, number: 20010487018, the National Science Fund of Hubei province, number: 2001ABB034, and an item of the Fund of Huazhong University of Science and Technology. He published many papers and took part in translating the first book about the study of DNA computing (published by the press of Qinghua University, in 2004). His study about the "Intelligent computing in the structure of network and its improvement" won the second-class Natural Science Improvement Reward of the ministry of education, and his paper "Sticker DNA computer model to solve the TSP" won the second-class Natural Science Excellent Paper Reward of the government of Shaanxi Province. His current research interests are DNA molecular computing.

