A New Approach for the Dominating-Set Problem by DNA-Based Supercomputing

Xu Zhou
College of Mathematics and Information Engineering, JiaXing University, JiaXing, P.R.China
zhouxu2006@126.com

GuangXue Yue, ZhiBang Yang and KenLi Li
College of Mathematics and Information Engineering, JiaXing University, JiaXing, P.R. China
School of Computer and Communications,Hunan University, Changsha, C P R.china
{guangxueyue@163.com , yangzhibang2006@126.com , LKL510@263.net }

Abstract—DNA computing has been applied to many different decision or combinatorial problems when being proved of its feasibility in experimental demonstration. In this paper, for the objective to reduce the DNA volume of the dominating set problem which belongs to the NP-complete problem, the pruning strategy is introduced into the DNA supercomputing and a new DNA algorithm is advanced. The new algorithm consists of a dominating set searcher, a dominating set generator, a parallel searcher and a minimum dominating set searcher. In a computer simulation, the new algorithm is testified to be highly space-efficient and error-tolerant compared to conventional bruteforce searching.

Keywords—DNA-based supercomputing, dominating set problem, pruning strategy, NP-complete problem

I. INTRODUCTION

The successful solution of the NP complete Hamiltonian directed path problem with seven-vertex by a DNA algorithm opened the field of biomolecular computing [1]. DNA computing has been employed to many different decision or combinatorial optimization problems for the experimental demonstration of its feasibility and it has led to an important breakthrough in computing [2-6]. DNA computing makes use of biomolecules as its information storage materials and biological laboratory experiments as its information processing operators [1-6].

The power of parallel, high density computation by molecules in solution allows DNA computers to solve hard computational problems such as NP-complete problems in polynomial increasing time, while a conventional Turing machine requires exponentially increasing time. However, most of the current DNA computing strategies are based on enumerating all candidate solutions [7-15]. These algorithms require that the size of the initial data pool increases exponentially with the number of variables in the calculation, so that the capacity of the DNA computer is limited. And what is more, Fu presented the enumeration algorithms made the length may also too long to make the algorithm to be length-efficient [16].

In order to break the barrier of simply enumerate method, Bach et al proposed a n1.89 volume, O(n^3m^2) time molecular algorithm for the 3-coloring problem and a 1.51 volume, O(n^3m^2) time molecular algorithm for the independent set problem, where n and m are, subsequently, the number of vertices and the number of edges in the problems resolved. Fu presented a polynomial-time algorithm with a 1.497 volume for the 3-SAT problem, a polynomial time algorithm with a 1.345 volume for the 3-coloring problem and a polynomial-time algorithm with a 1.229 volume for the independent set. Though the size of those volumes is lower, constructing those volumes is more difficult and the time complexity is higher.

The dominating set problem is widely used in network routing, town planning, and other real applications. Now the algorithm for the dominating set problem can not meet the needs of the application. Hence, we use the problem as an example to clarify the power of DNA computing for solving NP-complete problem.

In this paper, we describe a novel algorithm to solve the Dominating-set problem. Since Huiqin’s paradigm proposed in 2004 demonstrated the feasibility of applying DNA computer to tackle such an NP-complete problem. Instead of surveying all possible assignment sequences generated in the very beginning, we use the operations of Adleman-Lipton model and the solution space of sticker, then applying the pruning strategy, a new DNA algorithm for dominating-set problem is proposed.

The paper is organized as follows. Section 2 introduces the Chang et al.’s model in detail. Section 3 introduces the DNA algorithm to solve the dominating- set problem for the sticker solution space. In section 4, the experimental results by simulated DNA computing are given. Conclusions and future research work are drawn in Section 5.

II. DNA MODEL OF COMPUTATION

Our novel model employs only mature DNA biological operations. We use the model that took biological operations in the Adleman–Lipton model [1] and the solution space of...
stickers\[17,18\] in the sticker-based model in our algorithm. This model has several advantages from the sticker-based model and the Adleman–Lipton model in the following:

1). The new model has finished all the basic mathematical functions and the number of tubes, the longest length of DNA library strands, the number of DNA library strands and the number of biological operations are polynomial.

2). The basic biological operations in the Adleman–Lipton model have been performed in a fully automated manner in their lab. The full automation manner is essential not only for the speedup of computation but also for error-free computation.

3). Chang and Guo \[10, 11\] also employed the sticker-based model and the Adleman–Lipton model for dealing with Cook’s theorem, the set-packing and clique problems, the subset-product problem and many other NP complete problems.

A. The Adleman–Lipton model

Supposing that a tube is a multi-set of DNA strands over an alphabet set \{A, G, C, T\}, one can perform the following operations of the Adleman-Lipton model [10,11]:

1) \(\text{Extract}(T, S, (T, S)^*, (T, S)^*)\): To produce two tubes \((T, S)^*\) and \((T, S)^*\). \((T, S)^*\) is composed of the DNA molecules in \(T\) which contain \(S\) as a sub-strand and \((T, S)^*\) is composed of all the DNA molecules in \(T\) which do not contain \(S\).

2) \(\text{Merge}(T_0, T_1, T_2...T_n)\): To pour the \(n\) tubes \(T_1, T_2...T_n\) into tube \(T_0\). After this operation, the tube \(T_1, T_2...T_n\) will be empty.

3) \(\text{Amplify}(T_0, T_1, T_2...T_n)\): To produce \(n\) new tubes \(T_1, T_2...T_n\), which are copies of \(T_0\) and \(T_0\) becomes empty tube.

4) \(\text{Append}(T, S)\): To append \(S\) onto the end of every strand in \(T\).

5) \(\text{Discard}(T)\): To discard all the DNA strands in tube \(T\).

6) \(\text{Read}(T)\): To describe a single molecule contained in tube \(T\).

7) \(\text{Detect}(T)\): To check weather there is at least one DNA strand left in tube \(T\). If \(T\) includes at least one DNA molecule it returns ‘yes,’ and if \(T\) contains no DNA molecule it returns ‘no’.

B. Sticker-based solution space

Our algorithm is ground on the solution space of sticker model, which is a model of molecular computation introduced in [10]. The sticker model employs two basic groups of single strand DNA molecular in its representation of a bit string. As shown in Figure 1, the model involves stickers and a memory strand. The memory strand was divided into \(k\) non-overlapping sub-strands which has \(m\) bases is a single-stranded DNA with \(n\) bases. The sticker that has \(m\) bases is complementary to one of the \(k\) sub-strands in the memory strand. During the process of computation, each sub-strand is considered ‘1’ (on) or ‘0’ (off). If a sticker is annealed to its corresponding region on the given memory strand, then particular region is on for that strand. If no sticker is annealed to a region, then that region’s bit is off. A memory complex is the term defined as a memory strand where parts of the sub-strands are annealed by the matching stickers. Therefore the computational information can be carried in a binary format along the memory complex.

Stickers

\[
\begin{align*}
5' & \text{TAGCGTATA} & 3' & 5' & \text{AGCCCTATG} & 3' & 5' & \text{CGGAGACGA} & 3' & 5' & \text{GCTCAATCT} & 3' \\
& (1) & & & (2) & & & (3) & & & (4) \\
5' & \text{CTCAGGCTA} & 3' & 5' & \text{TTCAAAGTG} & 3' & 5' & \text{TAACACATA} & 3' \\
& (5) & & & (6) & & & (7) \\
\end{align*}
\]

Memory strand

ATCGCATAT TCGAGGTAC GCCTCTGCT CGAGTTCAGA GAGTCCGAT AAGTTTCAC ATTGTTGAT

Memory complex

\[
\begin{align*}
\text{AGCCCTATG} & \text{GCTCAATCT CTCAGGCTA TAACACATA} \\
(1) & \text{ATCGCATAT TCGAGGTAC GCCTCTGCT CGAGTTCAGA GAGTCCGAT AAGTTTCAC ATTGTTGAT} \\
& 0 & 1 & 0 & 1 & 1 & 0 & 1 \\
\text{AGCCCTATG} & \text{GCTCAATCT TTCAAGTG} \\
(2) & \text{ATCGCATAT TCGAGGTAC GCCTCTGCT CGAGTTCAGA GAGTCCGAT AAGTTTCAC ATTGTTGAT} \\
& 0 & 1 & 0 & 1 & 0 & 1 & 0 \\
\end{align*}
\]

Figure 1. Illustrations of the sticker model, which are encoded 010110 and 0101010 respectively.
In the sticker model, the input is a test tube called initial date pool and the output is a sequence of test tubes which are called final date pool. The final date pool is read by analyzing all the DNA strands in it.

On the assumption that an undirected graph \( G = (V, E) \), where \( V \) is the set of the vertices and \( E \) is the set of the edges. Assume that \( |V| = m \) which represents the number of the vertices in \( V \) and \( |E| = n \) that represents the number of the edges in \( E \).

Suppose \( S \) is one of the dominating sets of \( G \), if the \( i \)-th bit in an \( n \)-digit binary number is set to “1”, then it represents that the \( i \)-th vertex in \( S \) is also in \( G \) but not in \( V \setminus S \) and if the \( i \)-th bit in an \( n \)-digit binary number is set to “0”, then it represents that the \( i \)-th vertex is not in \( S \) but in \( V \setminus S \). To implement this, all of the possible dominating sets in \( G \) are transformed into \( n \)-bits binary numbers. Supposing that \( n \) one-bit binary numbers \( x_1, x_2, \ldots, x_n \) represent \( n \) vertexes in the set \( S \) and the \( i \)-th (0 \( \leq i \) \( \leq n \)) one-bit binary number refers to \( i \)-th the vertex in \( S \).

For the objective of representing all the possible dominating set for the dominating set problem, in our algorithm, vertices are represented by their binary representations using stickers. For every vertex, we denoted two symbols represented by 15-base stickers to encode the information into DNA strands:

\[
x_i = \begin{cases} 
1 & \text{if the vertex } v_i \text{ is in the dominating set} \\
0 & \text{otherwise} 
\end{cases} \quad (1 \leq i \leq n) \tag{1}
\]

### III. THE NOVEL DNA ALGORITHM FOR SOLVING THE DOMINATING SET PROBLEM

#### A. Dominating Set Problem

Let \( G \) be a graph with vertex-set \( V(G) \) and edge-set \( E(G) \). For any vertex \( v \in V \), the neighborhood of \( v \) is defined by \( N(v) = \{u \in V(G) : uv \in E(G)\} \).

Mathematically, a dominating set (DS) of a graph \( G = (V, E) \) is a subset \( S \subseteq V \) such that each vertex in \( V \setminus S \) is adjacent to at least one vertex in \( S \). \(|S|\) is denoted as the dominating number. The dominating-set problem is to find a minimum size dominating set in \( G \) and has been proved to be a NP-complete problem.

Dominating-set’s mathematical model can be described as follows (See Equation 2, 3).

\[
f = \min \sum_{i=1}^{n} x_i
\]
\[
x_i \sqrt{[A(i, j) \land x_j]} = 1, x_i = 1, j \in \{1, 2, \ldots, n\} \tag{2}
\]

\[
A(i, j) = \begin{cases} 
1 & \text{if the vertex } v_i \text{ is adjacent to } v_j \\
0 & \text{otherwise} 
\end{cases} \tag{3}
\]

Now, we will introduce two important theorems about Dominating-set problem in the following.

**Theorems 1:** The vertex \( v_i \) whose degree are one is not dominating vertex and its adjacent vertex \( v_j \) is dominating vertex.

**Theorems 2:** For a graph \( G \) with \( n \) vertexes, the dominating number

\[
\eta(G) = \frac{n}{2}.
\]

**Theorems 3:** Using notation \( \phi(G) \) to denote the max degree of all the vertexes in graph \( G \) and \( \eta(G) \) to denote the dominating number, we can get the formula

\[
\frac{n}{2} \leq \eta(G) \leq n - \phi(G).
\]

A DS is called a connected dominating set (CDS) if it also induces a connected subgraph. A \( k \)-tuple dominating set (\( k \)-DS) \( S \subseteq V \) of \( G \) is a set of vertices such that each vertex \( u \in V \) is \( k \)-dominated by vertices of \( S \). (Note that in some literatures, \( k \)-DS only requires that each node in \( V \setminus S \) is dominated by at least \( k \) nodes in \( S \)).

#### B. The DNA algorithm for Dominating Set Generator

First of all, it produces the solution space for the problem that will be resolved in the Adleman-Lipton model. Then, the biological operations are employed to remove illegal solutions from the solution space. Hence, the first step of settling the dominating-set problem is to generate a test tube consisting of all the possible dominating sets.

Based on Theorems 1 and the definition of Dominating set problem, a new algorithm for constructing the solution space of the dominating set problem is proposed.

**Procedure** Dominating_Set_Generator(\( T_0, G, n \))

- **<Input>**: Test Tube \( T_0 \) and The graph \( G \) with \( n \) vertexes where \( n \) is the number of the vertex.

- **<Output>**: Test Tube \( T_0 \) which contains the solution space of the Dominating sets

1: For every vertex \( v_i \) whose degree is one
2: Append(\( T_0, x^i_j \)).
3: For the vertex \( v_i \) that is adjacent to \( v_i \)
4: Append(\( T_0, x^i_j \)).
5: EndFor
6: \( V = V - v_j \).
7: EndFor
8: For \( i = 1 \) to \( n \) where \( n \) is the number of the vertex in the set \( V \) whose solution space has not produced
9: Amplify(\( T_0, T_1, T_2 \)).
10: Append(\( T_1, x^i_j \)).
11: Append(\( T_1, x^i_j \)).
12: Merge(\( T_0, T_1, T_2 \)).
13:  For each vertex $v_i$ adjacent to $v$, whose solution space has not produced
14:    $\text{Amplify}(T_0, T_1, T_2)$. 
15:    $\text{Append}(T_1, x_i^b)$.
16:    $\text{Append}(T_i, x_i^j)$.
17:    $\text{Merge}(T_0, T_1, T_2)$.
18:  EndFor
19:  Dominating_Set_Searcher$(T_0, v_i)$.
20:  EndFor

Lemma 1. The solution space of the dominating set can be constructed with sticker in a sticker-based model from the algorithm, Dominating_Set_GENERATOR$(T_0, G, n)$.

Proof: The algorithm, Dominating_Set_GENERATOR$(T_0, G, n)$ is implemented via the Append, Amplify and Merge operations.

Due to Theorems 1, we know that the vertex whose degree is one is not a dominating vertex and its adjacent vertex is a dominating vertex. Line(1)-Line(7) is an outer loop. On the first execution of Line(2) to Line(6), we append the DNA strands representing $x_i = 0$ on all the library strands in tube $T_0$ that is to say the vertex $v_i$ is not in the dominating set $S$. Line(3)-Line(5) is an inner loop. For each vertex adjacent to the vertex $v_i$, Line(5) will be run, so the DNA strands representing will be append on the tail of all the library strands in tube $T_0$. In the Line(7), the vertices $v_i$ and its adjacent vertex $v_i$ will be removed from the vertex set $V$.

Line(8)-Line(20) is also an outer loop, it will generate the full solution space of the dominating set problem resolved. Assume that $T_0$, $T_1$ and $T_2$ are distinct test tubes but only $T_1$ and $T_2$ are empty. The inner loop is implemented via the Amplify, Append and Merge operations. Each time Line(9) is used to amplify tube $T_0$ and to generate two new tubes, $T_1$ and $T_2$, which are copies of $T_0$. Tube $T_0$ becomes empty. Then, Line(10) is applied to append a DNA sequence (sticker), representing the value "0" for $x_i$, onto the end of every strand in tube $T_1$. Line(11) is also employed to append a DNA sequence (sticker), representing the value "0" for $x_i$, onto the end of every strand in tube $T_2$. Next, Line(12) is used to pour tube $T_1$ and $T_2$ into tube $T_0$. This indicates that DNA strands in tube $T_0$ include DNA sequences of $x_i = 1$ and $x_i = 0$. Line(13)-Line(18) is an inner loop which produces the solution space of the vertexes adjacent to the vertex $v_i$. Each time Line(9) is used to amplify tube $T_0$ and to generate two new tubes, $T_1$ and $T_2$, which are copies of $T_0$. Tube $T_0$ becomes empty. Then, Line(10) is applied to append a DNA sequence (sticker), representing the value "0" for $x_i$, onto the end of every strand in tube $T_1$. Line(11) is also employed to append a DNA sequence (sticker), representing the value "0" for $x_i$, onto the end of every strand in tube $T_2$. Next, Line(12) is used to pour tube $T_1$ and $T_2$ into tube $T_0$. This indicates that DNA strands in tube $T_0$ include DNA sequences of $x_i = 1$ and $x_i = 0$. On the running of Line(19), it will execute the algorithm Dominating_Set_Searcher$(T_0, v_i)$ to eliminate the illegal DNA strands referring to the vertex $v_i$ and its adjacent vertex.

After repeating execution of Line(1) through Line(20), it finally produces tube $T_0$ that consists of DNA sequences representing all the possible dominating set of the graph $G$.

From Dominating_Set_GENERATOR$(T_0, G, n)$, it takes $(n-c)$ amplify operations, $(cn^2(2n-c))$ append operations, $(n-c)$ merge operations where $c$ is the number of the vertexes whose degrees are one, $n$ Dominating_Set_Searcher$(T_0, v_i)$ and three test tubes to construct sticker-based solution space.

An $n$-bit binary number corresponds to an array of input. A value sequence for every bit contains 15 bases. Therefore, the length of a DNA strand, encoding a subset, is $15 \times n$ bases consisting of the concatenation of one value sequence for each bit.

C. The Construction of a Dominating Set Searcher

Due to the definition of the Dominating Set problem, a Dominating Set Searcher is designed in the following.

Procedure Dominating_Set_Searcher$(T_0, v_i)$

<Input>: Tube $T_0$ includes solution space of all the possible dominating sets for the vertex $v_i$ and its adjacent vertexes.

<Output>: The test tube $T_0$ of the satisfiable solution space for the vertex $v_i$ and its adjacent vertexes.

1:  $\text{Extract}(T_0, x_i^1 + (T_0, x_i^1), -(T_0, x_i^1))$.
2:  $T_1 := + (T_0, x_i^1)$ and $T_2 := -(T_0, x_i^1)$.
3:  For $j = 1$ to $|N(v_i)|$ where $|N(v_i)|$ is the number of elements in $N(v_i)$

4:  $\text{Extract}(T_2, x_j^1 + (T_2, x_j^1), -(T_2, x_j^1))$.
5:  $T_3 := + (T_2, x_j^1)$ and $T_4 := -(T_2, x_j^1)$.
6:  $\text{Merge}(T_0, T_1, T_3)$.
7:  EndFor
8:  $\text{Discard}(T_4)$.

Lemma 2. Algorithm Dominating_Set_Searcher$(T_0, v_i)$ is used to remove the illegal solution space referring to the vertex $v_i$ and its adjacent vertexes.

Proof: From the definition of dominating-set problem, we can see that each vertex in $V \sim S$ is adjacent to at least one vertex in $S$. Hence, if the vertex $v_i \notin S$, there is at least one vertex that is adjacent to $v_i$ in the dominating-set $S$. The algorithm, Dominating_Set_Searcher$(T_0, v_i)$, is implemented by the extract, Merge and Discard operations.

On the running of Line(1), it applies extract operation to produce two tubes $T_1$ and $T_2$. The first tube $T_1$ contains all of the DNA strands that have $x_i=1$ and the second tube $T_2$ consists of the DNA strands that have $x_i=0$. Now, tube $T_1$ represents those partitions that vertex $v_i$ is in the dominating set and tube represents those partitions that the vertex $v_i$ is not in the dominating set. Line(3)-(7) is an loop for finding out the illegal DNA strands referring to the vertex $v_i$ and its adjacent vertexes. Every time Line(3) is applied to extract tube $T_1$ and to generate two new tubes, $T_1$ and $T_4$. The first tube $T_1$ contains all of the
DNA strands that have $x_i=1$ and the second tube $T_4$ consist of the DNA strands that have $x_i=0$. Then, Line(6) is used to pour tube $T_1$ and $T_2$ into tube $T_0$. This indicates that DNA strands in tube $T_0$ include DNA sequences of $x_i = 1$ and $x_i = 0$, $x_j = 0$. After repeating the execution of Line(3) to Line(7), it finally get tube $T_4$ that consists of the illegal DNA sequences. Finally, on the running of Line(8), we use discard operation to discard all the DNA in tube $T_4$.

From Dominating_Set_Searcher ($T_0, v_i$), it takes $n$ extract operations, $n-1$ merge operations, one discard operation and five test tubes.

**D. The Construction of a Parallel Searcher**

In order to remove the DNA strands which are not the dominating set of the graph $G$, a parallel clique generator is designed.

**Procedure** Parallel_Searcher ($T_0$)

**<Input>:** Tube $T_0$ includes solution space of DNA sequences to encode all of the possible dominating sets

**<Output>:** Test tube $T_0$ showing all the dominating sets

1. **For** $i = 1$ to $n$
2. \[ \text{Extract}(T_0, x^i_1, + (T_0, x^i_2), - (T_0, x^i_1)). \]
3. \[ T_1 := + (T_0, x^i_2) \quad \text{and} \quad T_2 := - (T_0, x^i_1). \]
4. **For** every vertex $v_i$ that is adjacent to $v_i$
5. \[ \text{Extract}(T_2, x^i_2, + (T_2, x^i_1), - (T_2, x^i_2)). \]
6. \[ T_3 := + (T_2, x^i_1) \quad \text{and} \quad T_4 := - (T_2, x^i_2). \]
7. \[ \text{Merge}(T_0, T_1, T_2). \]
8. **EndFor**
9. \[ \text{Discard}(T_3). \]
10. **EndFor**

**Lemma 3:** Algorithm Parallel_Searcher ($T_0$) is applied to remove the illegal DNA strands.

**Proof:** The algorithm, Parallel_Searcher ($T_0$) is implemented by the extract, Merge and Discard operations.

On the first running of Line(1) which is an outer loop, it applies extract operation to produce two tubes $T_1$ and $T_2$. The first tube $T_1$ contains all of the DNA strands that have $x_i=1$ and the second tube $T_2$ consist of the DNA strands that have $x_i=0$. Now, tube $T_1$ represents those partitions that vertex $v_i$ is in the dominating set and tube represents those partitions that the vertex $v_i$ is not in the dominating set.

Line(3)-(7) is an loop for finding out the illegal DNA strands referring to the vertex $v_i$ and its adjacent vertexes. Every time Line(3) is applied to extract tube $T_2$ and to generate two new tubes, $T_3$ and $T_4$. The first tube $T_2$ contains all of the DNA strands that have $x_i=1$ and the second tube $T_4$ consist of the DNA strands that have $x_i=0$. Then, Line(6) is used to pour tube $T_1$ and $T_2$ into tube $T_0$. This indicates that DNA strands in tube $T_0$ include DNA sequences of $x_i = 1$ and $x_i = 0$, $x_j = 0$. After repeating the execution of Line(3) to Line(7), it finally get tube $T_4$ that consists of the illegal DNA sequences.

Finally, on the running of Line(8), we use discard operation to discard all the DNA in tube $T_4$. After repeating of Line(2)-Line(9), all the illegal strands of dominating set problem will be eliminated.

From Parallel_Searcher($T_0$), it takes \[ \sum_{i=1}^{n} |V(v_i)| + n = n^2 \]
extract operations, \[ \sum_{i=1}^{n} |V(v_i)| = n(n-1) \] merge operations, $n$ discard operation and five test tubes.

**E. The Construction of a MiniDominating Set Searcher**

**Procedure** MiniDominating_Set_Searcher($T_0$)

**<Input>:** Tube $T_0$ showing all the dominating sets

**<Output>:** Tubes $T_i$ ($0 \leq i \leq n$) representing the dominating set that contains $i$ vertexes

1. **For** $i = 0$ to $n$
2. \[ k = \min \{i, n/2\} \]
3. **For** $j = k$ down to 0
4. \[ \text{Extract}(T_j, y_{j+1}^1 + (T_j, y_{j+1}^2), - (T_j, y_{j+1}^1)). \]
5. \[ T_j := + (T_j, y_{j+1}^1) \quad \text{and} \quad T_j := - (T_j, y_{j+1}^2). \]
6. \[ \text{Merge}(T_{j+1}, T_{j+1}, T_j). \]
7. **EndFor**
8. **EndFor**
9. \[ \text{If}(\text{Detect}(T_{n+1}) = \text{‘yes’}) \quad \text{then} \]
10. \[ \text{Discard}(T_{n+1}). \]
11. **EndIf**
12. **For** $i = 1$ to $\frac{n}{1+\phi(G)}$
13. \[ \text{If}(\text{Detect}(T_i) = \text{‘yes’}) \quad \text{then} \]
14. \[ \text{Discard}(T_i). \]
15. **EndIf**
16. **EndFor**

**Lemma 4:** The algorithm, MiniDominating_Set_Searcher($T_0$), can be applied to search the solution of the minimum dominating set problem.

**Proof:** The algorithm, MiniDominating_Set_Searcher($T_0$), is implemented via Extract, Detect, Merge and Discard operations.

Line(1) is the outer loop and is mainly used to form the $K+2$ tubes and the DNA strands in tube $T_i$ contains $i$ \textquotesingle ‘1’, for $0 \leq i \leq n/2$.

Line(3) is an inner loop. On the first execution of Line(3), it uses the Extract operation to form two test tubes: $T_1$ and $T_0$. 

© 2010 ACADEMY PUBLISHER
Tube $T_1$ includes all of the strands that have $y_1 = 1$. Tube $T_0$ consists all of the strands that have $y_1 = 0$.

On the execution of Line(6) uses the merge operation to pour two tubes $T_i$ and $T_j$ into tube $T_l$. Tube $T_l$ currently consists of one ‘1’.

Repeat execution of Line(4) and Line(6) until every bit in the elements are considered. Line(9) is a detect operation to check if tube $T_{n+2-i}$ contains DNA strands. The DNA strands that have more than $n/2+1$ ‘1’ in the space solution of elements are in tube $T_{n+2-i}$. If it returns a ‘yes’, discard the tube $T_{n+2-i}$.

Due to Theorems 3, we know that the dominating number

$$\eta(G) \geq \frac{1 + \phi(G)}{n}.$$ 

Therefore the DNA strands in tube $T_i$ ($0 \leq i \leq n$)

$$\leq \frac{1 + \phi(G)}{n}$$

are illegal strands.

From MiniDominating_Set_Searcher($T_0$), it takes $k \times (k+1)/2 = n \times (n+2)/8$ ($k \leq n/2$) extract operations, $k \times (k+1)/2 = n \times (n+2)/8$ merge operations, $1 + \phi(G) + 1$ detect operations, and $n/2 + 1$ test tubes.

F. An improved DNA algorithm for Dominating Set Problem

The following DNA algorithm is applied to solve the Maximum Clique Problem.

Algorithm 5. Dominating_Set ($G, n$)

<Input>: The graph $G$ with $n$ vertices where $n$ is the number of the vertex in $G$

<Output>: The minimum Dominating set of the graph $G$ with $n$ vertices

1: Dominating_Set_Generator($T_0$, $n$).
2: Parallel_Searcher($T_0$).
3: MinDominating_Set_Searcher($T_0$).
4: For $i = 1$ to $n/2$
5: If (Detect($T_i$)) = ‘yes’
6: Read($T_i$).
7: EndIf
8: EndFor

Theorem 5: From those steps in Algorithm 1, the improved DNA based algorithm for dominating set problem can be solved.

Proof: On the execution of Line 1, Dominating_Set_Generator($T_0$, $n$) is mainly used to produce the satisfiable solution space of the Dominating set. The vertexes whose degrees are 0 is sure not to be in the dominating set and its adjacent vertexes are in the dominating set.

In Line(2) the algorithm, Parallel_Searcher($T_0$) is employed to search the solution of the dominating set from the solution space. The algorithm, MinDominating_Set_Searcher($T_0$) mainly used to separate the DNA strands in tube $T_0$ according to the number of the DNA sequence representing the value ‘1’.

Line(4) is loop and is mainly employed to search the solution of the dominating set. On the execution of Line(5), it employs the detect operations to detect whether there has any strands in $T_i$. If it returns ‘yes’, then we can get the solution of the dominating set from tube $T_i$.

G. The performance analysis of the proposed DNA algorithm

The following theorems describe time complexity of Algorithm 1, the number of the tube used in Algorithm 1 and the longest library strand in solution space in Algorithm 1.

Theorem 3.7: The Dominating set problem for any undirected $n$-vertex graph $G$ with $m$ edges can be solved with $O(n^c)$ biological operations, $O(n)$ tubes and the longest library strand, $O(n)$, where $n$ is the number of vertices in $G$ and $c$ is the number of the vertexes whose degrees are one, $n$ Dominating_Set_Searcher($T_0, vi$) and three test tubes to construct sticker-based solution space.

Step 2 is mainly applied to satisfiable solution space for the dominating set problem. It is indicated that it $n(n-1)$ merge operations, $n$ discard operation and five test tubes.

Step 3 is mainly applied to figure out the minimum dominating set and it takes $n \times (n+2)/8$ extract operations, $n \times (n+2)/8$ merge operations, $1 + \phi(G) + 1$ detect operations, and $n/2 + 1$ test tubes.

Step 4 is used to find out the final solution of our problem, it takes $n/2$ detect operations, one read operation and $n/2$ test tubes.

Hence, from the statements mentioned above, it is at once inferred that the time complexity of Algorithm 1 is as follows.

$$((c+1/2)(n-c)+(n-c)+m(n-n+1)+n(n-1)+n) +$$

$$n \times (n+2)/8 + n \times (n+2)/8 \times (1 + \phi(G)) + (1/2) + (n/2+1)$$

$$= \frac{13n^2}{4} + (5 + c + \frac{1}{(1 + \phi(G)) + (1 - 4c)} = O(n^2)$$

Refer to the Algorithm 1, the number of the tube used in Algorithm 1 is $O(n)$. Due to theorem 1 the longest strands in Algorithm 1 is $O(n)$. 
IV. EXPERIMENTAL RESULTS BY SIMULATED DNA COMPUTING

The graph in Figure 2 denotes such a problem. In Figure 2, the graph $G$ contains six vertexes and six edges. For convenience, the vertex $v_i$ is represented by $i(0 \leq i \leq 6)$.

![Figure 2. the graph $G$ of our problem](image)

A. DNA code

In our experiment, we used a unique value sequence, a 15-base DNA sequence, to implement each symbol of $\{ x_0^1, x_1^1, x_0^2, x_1^2, x_0^3, x_1^3, x_0^4, x_1^4, x_0^5, x_1^5, x_0^6, x_1^6 \}$ in our algorithms.

A library is a tube containing library strands, and the probes used for separating the library strands have sequences complementary to the value sequences. In DNA-based computation, there are errors in the separation of the library strands. To make the computation reliable, sequences must be designed to ensure that the following two conditions hold: one is that library strands have little secondary structure which might inhibit intended probe-library hybridization, the other is that the design must exclude sequences that might encourage unintended probe-library hybridization. To help achieve the goals, good sequences were generated to satisfy the following seven constraints defined by Braich et al. [7].

1. Library sequences contain only $A$'s, $T$'s, and $C$'s.

2. All library and probe sequences have no occurrence of 5 or more consecutive identical nucleotides; i.e. no runs of more than 4 $A$'s, 4 $T$'s, 4 $C$'s, or 4 $G$'s occur in any library or probe sequences.

3. Every probe sequence has at least 4 mismatches with all 15 base alignment of any library sequence (except for with its matching value sequence).

4. Every 15 base subsequence of a library sequence has at least 4 mismatches with all 15 base alignment of itself or any other library sequence.

5. No probe sequence has a run of more than 7 matches with any 8 base alignment of any library sequence (except for with its matching value sequence).

6. No library sequence has a run of more than 7 matches with any 8 base alignment of itself or any other library sequence.

7. Every probe sequence has 4, 5, or 6 Gs in its sequence.

DNA sequences generated by the modified Adleman program are shown in Table 1. With the nearest neighbor parameters, the Adleman program was used to calculate the enthalpy, entropy, and free energy for the binding of each probe to its corresponding region on a library strand.

<table>
<thead>
<tr>
<th>bit</th>
<th>5'→3' DNA Sequence</th>
<th>Enthalpy energy ($H$)</th>
<th>Entropy energy ($S$)</th>
<th>Free energy ($G$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_1^0$</td>
<td>AATTCACAACACATT</td>
<td>114.4</td>
<td>299.4</td>
<td>25.0</td>
</tr>
<tr>
<td>$x_1^1$</td>
<td>CCTATCATTCCACTC</td>
<td>112.8</td>
<td>284.9</td>
<td>24.3</td>
</tr>
<tr>
<td>$x_2^0$</td>
<td>AATTCCTATCCCTA</td>
<td>108.5</td>
<td>273.0</td>
<td>27.8</td>
</tr>
<tr>
<td>$x_2^1$</td>
<td>TCTCTCTCTACTA</td>
<td>105.2</td>
<td>270.5</td>
<td>28.2</td>
</tr>
<tr>
<td>$x_3^0$</td>
<td>CTCTCTCCACTACTA</td>
<td>111.1</td>
<td>288.3</td>
<td>27.8</td>
</tr>
<tr>
<td>$x_3^1$</td>
<td>CTTTCTAACCCTCA</td>
<td>103.8</td>
<td>272.6</td>
<td>28.2</td>
</tr>
<tr>
<td>$x_4^0$</td>
<td>AAATCTCATACTAC</td>
<td>109.9</td>
<td>285.5</td>
<td>27.0</td>
</tr>
<tr>
<td>$x_4^1$</td>
<td>AATTACAAATCATCT</td>
<td>104.3</td>
<td>273.0</td>
<td>24.1</td>
</tr>
<tr>
<td>$x_5^0$</td>
<td>TTATCTTACACATCT</td>
<td>112.1</td>
<td>282.8</td>
<td>24.4</td>
</tr>
<tr>
<td>$x_5^1$</td>
<td>TTAATCAAATCCCTA</td>
<td>102.1</td>
<td>266.0</td>
<td>22.6</td>
</tr>
<tr>
<td>$x_6^0$</td>
<td>ATCTACTCTACTAC</td>
<td>105.2</td>
<td>277.1</td>
<td>25.0</td>
</tr>
<tr>
<td>$x_6^1$</td>
<td>CTCTATATAATTACT</td>
<td>104.8</td>
<td>283.7</td>
<td>22.4</td>
</tr>
</tbody>
</table>

B. Solving Process of the improved algorithm for the Dominating set problem

After DNA coding, we can simulate the Algorithm 1. In our problem, the vertex set $V$ is $\{1, 2, 3, 4 \}$ and the edge set $E$ is $\{(1, 3), (2, 3), (3, 4), (4, 5), (4, 6), (5, 6)\}$. The process of our novel algorithm for the dominating set problem is in the following.

(1) Firstly, the vertex $a$ and $b$ whose degrees are one and their adjacent vertex $c$ are found out. On the execution of the
algorithms, Dominating Set Generator (Tn), and from Theorems 1, we know that the vertex 1 and 2 are not in the dominating set and their adjacent vertex 3 is in the dominating set. Hence, we append the DNA strands representing the tail of all the DNA strands in tube T0. Then, the DNA strands in tube T0 are \( \{ x_1^0 x_2^0 x_3^1 \} \).

After removing the vertex 1, 2 and from the vertex set, \( V \) is \( \{ 4, 5, 6 \} \). Generating the solution space of the vertex 4, the DNA strands in tube will be \( \{ x_0^4 x_1^0 x_2^1 x_3^0 x_4^1 x_5^1 x_6^0 \} \). From the graph, we can see that the vertex 5 and 6 is the adjacent vertexes of vertex 4. So we produce the solution space of the vertex 5 and 6. Considering the vertex 5, the DNA strands in tube will be \( \{ x_0^5 x_1^0 x_2^1 x_3^0 x_4^1 x_5^1 x_6^0 \} \). Generating the solution space of vertex 6, the DNA strands in tube T6 will be \( \{ x_0^6 x_1^0 x_2^1 x_3^0 x_4^1 x_5^1 x_6^0 \} \). So far, we get the full solution space of the dominating set problem.

On the execution of Dominating Set Searcher (Tn), the DNA strands will contains 'yes'. We will get five tubes \( T0, T1, T2, T3 \) and \( T4 \) in tube \( T0 \). The strands in tube \( T0 \) and \( T1 \) are \( \{ x_0^0 x_1^0 x_2^1 x_3^0 x_4^1 x_5^1 x_6^0 \} \). The strands in tube \( T2 \) and \( T3 \) are \( \{ x_0^2 x_1^0 x_2^1 x_3^0 x_4^1 x_5^1 x_6^0 \} \). Therefore, the dominating set of our problem are \( \{ 3, 6 \} \) and \( \{ 3, 4 \} \).

V. CONCLUSIONS

As we all know, DNA computing has the advantage of huge parallelism. The volume's exponential explosion problem is the critical factor that constraints the development of the DNA computing. For the objective to decrease the DNA volume of the dominating set problem, the pruning strategy is taken into the DNA-based supercomputing and a new DNA-based algorithm is proposed. Comparing with the enumerate DNA-based algorithm for dominating set problem the DNA library strands reduced considerably.

In the future, molecular computers may be a good choice for massively parallel computations [21-26]. For the objective to reach a free stage in using DNA computers just as using classical digital computers, many technical difficulties such as real time updating a solution when the initial condition of a problem changes, finding out the exact answer quickly and efficiently and the size of the initial data pool increases exponentially with the number of variables in the calculation need to be overcome before this becomes real. We expect our study can make a contribution to clarify that DNA-based computing is a technology that worthwhile us seeking.

REFERENCE


[18] Lu S D. The Experimentation of Molecular Biology. Peking Union Medical College Press, 1999


Zhou Xu was born in 1983, Jiangshu, China. She obtained her master in Hunan University. Her main research interests include Biological information, DNA Computing and Parallel Computing. In these areas, she has published many papers in leading journals or conference proceedings.

Guangxue Yue was born in 1963, Guizhou, China. He obtained his master in Hunan University. Professor, the College of Mathematics & Information Engineering, jiaxing University, China. His main research interests include Biological Information, Distributed Computing & Network, and Hybrid & Embedded Systems.

Yangzhi Bang was born in 1984, Hunan, China. obtained his master in Huazhong University. Doctor, the School of Computer and Communications, Hunan University, China. His main research interests include DNA Computing, and Parallel Computing.

Kenli Li was born in 1971, Hunan, China. He obtained his doctor in Huazhong University. Professor, the School of Computer and Communications, Hunan University, China. His main research interests include Biological Information, DNA Computing, and Parallel Computing.