Fast Parallel Molecular Solution to the Maximum Triangle Packing Problem on Massively Parallel Bio-Computing

Babak Dalvand¹, Saeed Safaei², Mojtaba Nazari¹

¹Department of Mathematics, Khorramabad Branch, Islamic Azad University, Khorramabad, Iran
²Department of Mathematics, University of Arak, Markazi, Iran

Abstract - Adleman showed that deoxyribonucleic acid (DNA) strands could be employed towards calculating solutions to an instance of the Hamiltonian path problem (HPP). Lipton also demonstrated that Adleman’s techniques could be used to solve the Satisfiability problem. In this paper, we use Adleman-Lipton model for developing a DNA algorithm to solve maximum triangle packing problem (MTPP). In spite of the NP-hardness of maximum triangle packing problem (MTPP) our DNA procedures is done in a polynomial time.

Keywords: DNA computing, NP-hard problem, maximum triangle packing problem

1 Introduction

Recently, DNA computing has considerable attention as one of non-silicon based computing. Watson-Crick complementarity and massive parallelism are two important features of DNA. By using these features, one can solve an NP-complete problem, which usually needs exponential time on a silicon-based computer, in a polynomial number of steps with DNA molecules. Adleman [1] solved Hamiltonian path problem of size n in spite of NP-hardness of the problem in \( O(n) \) steps using DNA molecules. That is the first work for DNA computing. The second NP-hard problem that has solved by DNA computing is Satisfiability (SAT), Lipton [11] showed that the Adleman’s manner could be used to determine SAT. Ouyang et al. [13] used the model to solve maximal clique problem. Some other NP-hard problems that have been solved by the model are as follow: binary integer programming [18], exact cover by 3-sets [3], maximum cut [17], set cover [3], Solving traveling salesman problems [12], solving the shortest path problem [16] etc. Moreover, procedures for primitive operations, such as logic or arithmetic operations, have also been proposed so as to apply DNA computing in a wide range of problems [4-6, 8-10, 19]. In this paper, the DNA operations proposed by Adleman (1994) and Lipton (1995) are used for figuring out solutions of maximum triangle packing problem.

For the given graph \( G=(V,E) \) we want to find a triangle packing for \( G \) i.e., a collection \( V_1, V_2, ..., V_k \) of disjoint subsets of \( V \), each containing exactly 3 vertices, such that for each \( V_i = \{u_i, v_i, w_i\}, 1 \leq i \leq k \) all three of the edges \((u_i,v_i),(u_i,w_i)\) and \((v_i,w_i)\) belong to \( E \).

And the size of \( k \) is maximum. In other words we are looking for collection \( C = \{V_1, V_2, ..., V_k\} \) containing the following conditions and \( k \) being at its maximum:

All subsets of \( V_i, 1 \leq i \leq n \) should have exactly 3 members.

All three of the edges \((u_i,v_i),(u_i,w_i)\) and \((v_i,w_i)\) belong to \( E \).

For the graph represented in figure 1, \( V_1 = \{A_1, A_2, A_3\}, V_2 = \{A_4, A_5, A_6\} \) is a solution to the problem.

2 Adleman-Lipton model

Bio-molecular computers work at the molecular level. Since biological and mathematical operations have some similarities, DNA, the genetic material that encodes the living organisms, is stable and predictable in its reactions and can be used to encode information for mathematical problems. DNA algorithms typically solve problems by initially assembling large data sets as input and then eliminating undesirable solutions.

A DNA (deoxyribonucleic acid) is a polymer, which is strung together from monomers called deoxyribonucleotides [14, 15]. Distinct nucleotides are detected only with their bases. Those bases are adenine (A), guanine (G), cytosine (C), and thymine (T). Two strands of DNA can form (under appropriate conditions)
a double strand, if the respective bases are the Watson–Crick complements of each other, i.e., A matches T and C matches G; also 3'- end matches 5'- end. For example, strands 5'–ACCGGATGTCA-3' and 3'-TGGCCTACAGT-5' can form a double strand. We also call them as the complementary strand of each other. The length of a single DNA strand is the number of nucleotides comprising the single strand. Thus, if a single DNA strand includes 20 nucleotides, it is called a 20 mer. The length of a double strand (where each nucleotide is base paired) is counted in the number of base pairs. Thus, if we make a double strand from two single strands of length 20 mer, then the length of the double strand is 20 base pairs, also written as 20 bp for more discussion of the relevant biological background, refer to [2, 14, and 15]. The DNA operations proposed by Adleman and Lipton [1, 2, and 11] are described below.

The Adleman–Lipton model: A (test) tube is a set of molecules of DNA (i.e. a multi-set of finite strings over the alphabet {A, C, G, T}). Given a tube, one can perform the following operations:

1. Merge (T1, T2): for two given test tubes T1, T2 it stores the union \( T_1 \cup T_2 \) in T1 and leaves T2 empty;

2. Copy (T1, T2): for a given test tube T1 it produces a test tube T2 with the same contents as T1;

3. Detect (T): Given a test tube T it outputs “yes” if T contains at least one strand, otherwise, outputs “no”;

4. Separation (T1, X, T2): for a given test tube T1 and a given set of strings X it removes all single strands containing a string in X from T1 , and produces a test tube T2 with the removed strands;

5. Selection (T1, L, T2): for a given test tube T1 and a given integer L it removes all strands with length L from T1, and produces a test tube T2 with the removed strands;

6. Cleavage (T, \( \sigma_0\sigma_1 \)): for a given test tube T and a string of two (specified) symbols \( \sigma_0\sigma_1 \) it cuts each double trend containing \( [\sigma_0\sigma_1] \) in T into two double strands as follows:

\[
\frac{\alpha_0\sigma_0\sigma_1\beta_0}{\alpha_0\sigma_0\sigma_1\beta_1} \Rightarrow \frac{\alpha_0\sigma_0}{\alpha_0\sigma_0} \frac{\sigma_1\beta_0}{\sigma_1\beta_1}
\]

7. Annealing (T): for a given test tube T it produces all feasible double strands in T. The produced double strands are still stored in T after Annealing;

8. Denaturation (T): for a given test tube T it dissociates each double strand in T into two single strands;

9. Discard (T): for a given test tube T it discards the tube T;

10. Append (T, Z): for a given test tube T and a given short DNA singled strand Z it appends Z onto the end of every strand in the tube T;

11. Read (T): for a given tube T, the operation is used to describe a single molecule, which is contained in the tube T. Even if T contains many different molecules each encoding a different set of bases, the operation can give an explicit description of exactly one of them.

Since these eleven manipulations are implemented with a constant number of biological steps for DNA strands, we assume that the complexity of each manipulation is \( O(1) \) steps.

### 3 Solving MTPP in Adleman–Lipton model

Let \( G=(V,E) \) be an undirected graph with the set of vertices being \( V = \{ A_k \mid k=1,\ldots,n \} \) and the set of edges being \( E = \{ e_{ij} \mid \text{for some } 1 \leq i,j \leq n \} \). Let \( |E|=d \). Then \( d \leq n(n-1)/2 \). Note that \( e_{ij} \) is in E if the vertices \( A_i \) and \( A_j \) are connected by an edge. In the following, the symbols 0,1,#,X,Y,A_k,B_k (k=1,2,\ldots,n) denote distinct DNA singled strands with same length, say 10-mer.And \( \| \| \) denotes the length of the DNA singled strand. Obviously the length of the DNA singled strands greatly depends on the size of the problem involved in order to distinguish all above symbols and to avoid hairpin formation. We have n edges in this graph so the maximum length of k is equal to \( V_1,V_2,\ldots,V_k \) which comes to \( k^{n/3} \). Then W is equal to \( k^{n/3} \). For graph G we define W subsets and we define a collection \( C = \{ V_1,V_2,\ldots,V_w \} \). The strand \( B_j,A_i \) in which \( 1 \leq i \leq n,0 \leq j \leq W \) means \( A_i \) vertices is in j-th subset. And
the strand \(B_0A_i\) means \(A_i\) does not exist in any subsets.

Tubes P and Q are defined as follows:

Let

\[ P = \{ j|X,A_i\#B_0,A_kB_{j-k}\}, Y|k=1,2,...,n\} \text{ and} \]

\[ Q = \{ j,X,A_i\#B_0|j=1,2,...,n, j=1,2,...,n\} \]

We design the following algorithm to solve the maximum triangle packing problem and give the corresponding DNA operations as follows:

### 3.1 Produce each possible collection \(C\)

For a graph with \(n\) vertices, each possible \(C\) of vertices is represented by an \(n\)-digit number in base \(W\). For example for graph 1 we can represent \(C = \{A_1, A_2, A_3\}, V_2 = \{A_4, A_6, A_7\}\) as 220111 and show \(C = \{V_1 = \{A_1, A_2, A_3\}, V_2 = \{A_4, A_6, A_7\}\}\) as 2221101, in which number \(j\) in \(i\)-th element shows that the vertices \(A_i\) is in the \(j\)-th subset, and if \(j=0\) it means that this vertices doesn’t exist in any of the subsets.

In this way, we transform all possible collection \(C\) in an \(n\)-vertex graph into an ensemble of all \(n\)-digit in base \(W\) numbers. We call this the data pool.

#### (1-1) Merge (\(P;Q\))

#### (1-2) Annealing (\(P\))

#### (1-3) Denaturation (\(P\))

#### (1-4) Separation (\(P, \{A_i\#\}, T_{\text{tmp}}\))

#### (1-5) Discard (\(P\))

#### (1-6) Separation (\(T_{\text{tmp}}\)#\(B_0\), \(P\))

After above six steps of manipulation, singled strands in tube \(P\) will encode all \(W^n\) collection \(C\) in the form of \(n\)-digit base \(W\) numbers. For example, for the graph in Fig. 1 with \(n=7\) we have, e.g. the singled strand

\[ #B_1 \cdot A_1 \cdot A_2 \cdot A_3 \cdot B_2 \cdot A_4 \cdot B_5 \cdot B_1 \cdot A_6 \cdot B_7 \cdot B_1 \cdot A_3 \# \]

Which denotes the subset \(C = \{V_1 = \{A_1, A_2, A_3\}, V_2 = \{A_4, A_6, A_7\}\}\) corresponding to the number 2220111 in base \(W\). This operation can be finished in \(O(1)\) steps since each manipulation above works in \(O(1)\) steps.

### 3.2 Eliminating the sets not having the first condition

First of all for each collection \(C = \{V_1, V_2, ..., V_k\}\) we calculate the members of each \(V_i\) subset for which \(1 \leq i \leq n\). Any \(V_i\) subset not consisting of exactly 3 members is a unique situation which should be eliminated. Therefore any collection containing a unique subset is a unique situation.

For \(r = 1 \text{ to } r = W\)

For \(d = 0 \text{ to } d = n\)

#### (2-1) Separation (\(P, \{B_r\#A_d\}, T_i\))

#### (2-2) Append (\(T_i, r\))

#### (2-3) Merge (\(P, T_i\))

#### (2-4) Discard (\(T_i\))

For \(r = 1 \text{ to } r = W\)

#### (2-5) Separation (\(P, rrr, T_i\))

#### (2-6) Separation (\(T_i, rrr, T_j\))

#### (2-7) Merge (\(P, T_i\))

#### (2-8) Discard (\(T_i\))

End for

The strand \#\(B_1 \cdot A_1 \cdot A_2 \cdot A_3 \cdot B_2 \cdot A_4 \cdot B_5 \cdot B_1 \cdot A_6 \cdot B_7 \cdot B_1 \cdot A_3 \#\) represents \(C = \{V_1 = \{A_1, A_2, A_3\}, V_2 = \{A_4, A_6, A_7\}\}\) and after execution of this algorithm will become

\[ #B_1 \cdot A_1 \cdot A_2 \cdot A_3 \cdot B_2 \cdot A_4 \cdot B_5 \cdot B_1 \cdot A_6 \cdot B_7 \cdot B_1 \cdot A_3 \#0111222 \cdot \]

The strand \#\(B_1 \cdot A_1 \cdot A_2 \cdot A_3 \cdot B_4 \cdot B_5 \cdot B_1 \cdot A_6 \cdot B_7 \cdot B_1 \cdot A_3 \#\) represents \(C = \{V_1 = \{A_1, A_2, A_3\}, V_2 = \{A_4, A_6, A_7\}\}\) and will become

\[ #B_1 \cdot A_1 \cdot A_2 \cdot A_3 \cdot B_4 \cdot B_5 \cdot B_1 \cdot A_6 \cdot B_7 \cdot B_1 \cdot A_3 \#0011222 \cdot \]

Any strand ending with 111,222,333,...WWW is possible. For example \#\(B_1 \cdot A_1 \cdot A_2 \cdot A_3 \cdot B_4 \cdot B_5 \cdot B_1 \cdot A_6 \cdot B_7 \cdot B_1 \cdot A_3 \#0011222 \cdot \) is impossible because it contains 11. It means that a subset in the collection contains 2 vertices.

Each of the above actions will conclude at \(O(1)\). Therefore the algorithm will terminate at \(O(n^2)\).

### 3.3 Eliminating the sets not having the second condition

For every 2 vertices \(u, v \in V\) and \(u \neq v\) we find all collections containing subsets \(V_i, i = 1,2,...,n\) for which \(u, v \in V_i\). If \((u, v) \notin E\) then we add to the end of the strands representing this collection, the single strand \(Y\). At the end we will eliminate the strands containing \(Y\).
For $r = 1$ to $r = n$
For $d = 1$ to $d = n$
If $r \neq d$ and $(r,d) \notin E$ Then
For $i = 1$ to $i = W$
(3-1) Separation($P_i\{A_i\},T_i$);
(3-2) Separation($T_i\{A_i\},T_{i2}$);
(3-3) Append($T_{i2},Y$);
(3-4) Merge($P,T_i$);
(3-5) Merge($P,T_{i2}$);
End For
End If
End For
End For

In graph $1$, $A_1, A_2 \in V$ and $C = \{V_1 = \{A_1, A_2, A_3\}, V_2 = \{A_3, A_4, A_5\}\}$ is a collection for this graph and the strand representing it is $#B_1A_2A_3B_1A_4B_2A_5B_3A_6B_4B_5A_7#A_1A_2A_3A_4A_5A_6A_7#0111222$. After execution of the above algorithm due to $A_7 \in V_1$ and edge $(A_1, A_7) \notin E$ the strand is converted to $#B_1A_2A_3B_1A_4B_2A_5B_3A_6B_4B_5A_7#A_1A_2A_3A_4A_5A_6A_7#0111222$ and is then eliminated.

Each of the above actions will conclude at $O(1)$. Therefore the algorithm will terminate at $O(n^3)$.

3.4 Finding the collection(s) with most members amongst

Collections $C = \{V_1, V_2, ..., V_k\}$ we would like to find the collection(s) with most members. Therefore first of all we should find out the number of collection members.

For each subset $V_i, i = 1, 2, ..., n$ existing in a collection, we add an X strand to the end of the strand representing this collection. For example we add it to collection $C = \{V_1 = \{A_1, A_2, A_3\}, V_2 = \{A_5, A_6, A_7\}\}$ because the strand representing it contains 1 and 2. We add XX to this strand. Obviously $#B_1A_2A_3B_1A_4B_2A_5B_3A_6B_4B_5A_7#A_1A_2A_3A_4A_5A_6A_7#0111222$ is the longest strand representing the collection with most members. We do this using the following algorithm.

For $r = 1$ to $r = W$
(4-1) Selection($P, 30*n + 20 + 10*n + (W - r)*10, T_i$)
(4-2) If Detect($T$) is yes,
then end for else continue the circulation
End for

Note that, in each strand, the sub-strand X can be repeated W times. We will present an example for $n=7$. The strand $#B_1A_2A_3B_1A_4B_2A_5B_3A_6B_4B_5A_7#A_1A_2A_3A_4A_5A_6A_7#0111222$ is made up of sub-strand $B_1A_2A_3B_1A_4B_2A_5B_3A_6B_4B_5A_7#A_1A_2A_3A_4A_5A_6A_7#0111222$ with length 30*7 and 2 sub strands $#A_2A_3B_1A_4B_2A_5B_3A_6B_4B_5A_7#A_1A_2A_3A_4A_5A_6A_7#0111222$ with length 7*10 and strand XX with length 10*2. Hence the total length of this strand is $30*7 + 10*7 + 20 + 2*10$.

Each of the above actions will conclude at $O(1)$. This algorithm will terminate at $O(n^3)$.

4 Conclusions

As the first work for DNA computing, (Adleman, 1994) presented an idea to demonstrate that deoxyribonucleic acid (DNA) strands can be applied to solving the Hamiltonian path NP-complete problem of size $n$ in $O(n^3)$ steps using DNA molecules. Adleman’s work shows that one can solve an NP-complete problem, which usually needs exponential time on a silicon-based computer, in a polynomial number of steps with DNA molecules. From then on, Lipton (1995) demonstrated that Adleman’s experiment could be used to determine the NP-complete Satisfiability (SAT) problem (the first NP-complete problem). Ouyang et al. (1997) showed that restriction enzymes could be used to solve the NP-complete clique problem. In recent years, lots of papers have occurred for designing DNA procedures and algorithms to solve various NP-complete problems. As Guo et al. (2005) pointed out, it is still important to design DNA procedures and algorithms for solving various NP-complete problems since it is very difficult to use biological operations for replacing mathematical operations.

In this paper, we propose a procedure for maximum triangle packing NP-complete problems in the Adleman–Lipton model. The procedure works in $O(n^3)$ steps for maximum triangle packing problem of a directed graph with $n$ vertices. All our results in this paper are based on a theoretical model. However, the proposed procedures can be implemented practically since every DNA manipulation used in this model has been already realized in lab level.

5 References


