IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 5, Issue 7, July 2016

Molecular solutions for the Maximum K-colourable Sub graph Problem in Adleman-Lipton model

Akbar Moazam

Department of Computer science, Isfahan University, Isfahan, Iran

Abstract: Adleman [1] showed that deoxyribonucleic acid (DNA) strands could be employed towards calculating solutions to an instance of the Hamiltonian path problem (HPP). Lipton [5] 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 k-colourable Sub graph problem. In spite of the NP-hardness of Maximum k-colourable Sub graph problem our DNA procedures is done in a polynomial time.

Keywords: DNA computing, NP-hard problem, Maximum k-colourable Sub graph problem

I. 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 [3]. 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 [5] showed that the Adelman's manner could be used to determine SAT. 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 [3-4, 6-14]. In this paper, the DNA operations proposed by Adleman [1] and Lipton [5] are used for figuring out solutions of Maximum k-colourable Sub graph problem. Given an undirected graph G = (V, E) with an assignment of weights to the edges w: E → N and an integer k ∈ {2, 3, …, |V|}, we try to find maximum $E' \subseteq E$ such that the sub graph $G' = (V', E')$ is k-colourable, i.e., there is a coloring for G' of cardinality at most k. There is another definition for this problem.

Given an undirected graph G = (V, E) with an assignment of weights to the edges w: E → N and an integer k ∈ {2, 3, …, |V|}, partition V into k disjoint sets F = { C_1, C_2,…,C_k } while maximizing

$$\sum_{i=1}^{k-1} \sum_{j=i+1}^{k} \sum_{\substack{v_1 \in C_i \\ v_2 \in C_j}} w(\{v_1, v_2\})$$

Here we assume weights of edges are equal 1.

II. 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. 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. The DNA operations proposed by Adleman and Lipton [1, 2,3,4] are described below. 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}).

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

(2) Copy $T_1 \cup T_2$: for a given test tube T_1 it produces a test tube T_2 with the same contents as T_1;

DOI 10.17148/IJARCCE.2016.5701

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319 5940
International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 5, Issue 7, July 2016

(3) Detect (T): Given a test tube T it outputs ‘‘yes’’ if T contains at least one strand, otherwise, outputs ‘‘no’’;
(4) Separation (T_1, X, T_2): for a given test tube T_1 and a given set of strings X it removes all single strands containing a string in X from T_1, and produces a test tube T_2 with the removed strands;
(5) Selection (T_1, L, T_2): for a given test tube T_1 and a given integer L it removes all strands with length L from T_1, and produces a test tube T_2 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 $\begin{bmatrix}\sigma_0\sigma_1\\ \sigma_0\sigma_1\end{bmatrix}$ in T into two double strands as follows:

$$\begin{bmatrix}\alpha_0\sigma_0\sigma_1\beta_0\\ \alpha_1\sigma_0\sigma_1\beta_1\end{bmatrix}\Rightarrow\begin{bmatrix}\alpha_0\sigma_0\\ \alpha_1\sigma_0\end{bmatrix},\begin{bmatrix}\sigma_1\beta_0\\ \sigma_1\beta_1\end{bmatrix}$$

(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.

III. SOLVING MAXIMUM K-COLOURABLE SUBGRAPH PROBLEM IN ADLEMAN–LIPTON MODEL

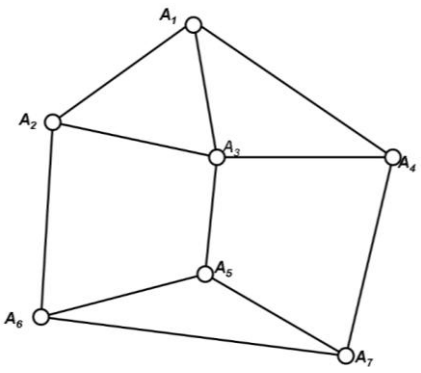

Fig. 1.

Let $G=(V,E)$ be an undirected graph with the set of vertices being $V=\{A_k \mid k=1,2,\ldots,n\}$ and the set of edges being $E=\{e_{ij} \mid for\ some\ 1\le i,\ j\le n\}$. Let |E|=d. Then $d\le n(n-1)/2$. 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. We choose DNA singled strands $y_{i,j}$ to encode the edges connecting the vertices A_i and A_j with length of 10-mer. All these $y_{i,j}$ can be taken the same, say y_1, for our problem. For convenience of argument we still use a dummy symbol $y_{i,j}$ of length 0-mer if the vertices A_i and A_j is not connected by an edge or i=j. For graph G we define W subsets and we define a collection $C=\{V_1,V_2,\ldots,V_w\}$. The strand $B_i jA_i$ in which $1\le i\le n, 0\le j\le W$ means A_i vertices is in j-th subset. And the strand $B_i 0A_i$ means A_i does not exists in any subsets. Tubes P and Q are defined as follows:
Let $P=\{j,X,A_1\#,\#B_n,A_k B_{k-1},Y \mid k=1,2,\ldots,n,\ j=1,2,\ldots,n\}$
and $Q=\{\overline{\#,B_k jA_k} \mid k=1,2,\ldots,n,\ j=1,2,\ldots,n\}$
We design the following algorithm to solve the Maximum k-colourable Sub graph problem and give the corresponding DNA operations as follows:

IV. 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 $C=\{V_1=\{A_1,A_2,A_3\},V_2=\{A_5,A_6,A_7\}\}$ as 2220111 and show $C=\{V_1=\{A_1,A_3,A_4\},V_2=\{A_5,A_6,A_7\}\}$ 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 vertex 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_1#},T_{tmp});
(1-5) Discard (P);
(1-6) Separation (T_{tmp},{#B_n },P);

After above six steps of manipulation, singled strands in tube P will encode all W^n collection C in the form of n-

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319 5940
International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 5, Issue 7, July 2016

digit base W numbers. For example, for the graph in Fig. 1 with n=7 we have, e.g. the singled strand $\#B_7 2A_7 B_6 2A_6 B_5 2A_5 B_4 0A_4 B_3 1A_3 B_2 1A_2 B_1 1A_1 \#$ Which denotes the subset $C=\{V_1=\{A_1, A_2, A_3\}, V_2=\{A_5, A_6, A_7\}\}$ 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.

V. ELIMINATING THE SETS NOT HAVING THE FIRST CONDITION

First of all, for each collection $C=\{V_1, V_2, ..., V_k\}$ all V_i should be disjoint. The definition of P and Q guarantee this. Because each strand can be in one subset each time. It is clear if you see this example $\#B_7 2A_7 B_6 2A_6 B_5 2A_5 B_4 0A_4 B_3 1A_3 B_2 1A_2 B_1 1A_1 \#$. We cannot produce some strands like $\#B_7 2A_7 B_7 0A_7 B_6 2A_6 B_5 2A_5 B_4 0A_4 B_3 1A_3 B_2 1A_2 B_1 1A_1 \#$. vertices A_7 cannot be in two subsets.

In this step we want to select collections that have K subsets. Let's look at $Q=\{\#, \overline{B_k jA_k} | k=1,2,\ldots,n,\ j=1,2,\ldots,n\}$. if we count the different number of j then we can count the number of subsets. Because each j shows one subset. In this example $\#B_7 2A_7 B_6 2A_6 B_5 2A_5 B_4 0A_4 B_3 1A_3 B_2 1A_2 B_1 1A_1 \#$ we have 0,1,2. If each strand contains j we add @ to the end of it. This algorithm does it by following steps. r =1..n

For $d=1$ *to* $d=n$
(2-1) *Separation* $(P, \{B_d rA_d\}, T_1)$
(2-2) *Append* (T_1, r)
(2-3) *Merge* (P, T_1)
(2-4) *Discard* (T_1)
End for

After run this algorithm for strand

$$\#B_7 2A_7 B_6 2A_6 B_5 2A_5 B_4 0A_4 B_3 1A_3 B_2 1A_2 B_1 1A_1 \#$$

we will have

$$\#B_7 2A_7 B_6 2A_6 B_5 2A_5 B_4 0A_4 B_3 1A_3 B_2 1A_2 B_1 1A_1 \#@@$$

because we have two subsets.
Now we should select strands with $\underbrace{@@\ldots@@}_{k}$ .because It means, these strand show collections with k subset. For this purpose, we will remove all strands contain $\underbrace{@@\ldots@@}_{k+1}$ and after that remove all strands contain $\underbrace{@@\ldots@@}_{k-1}$.

(2-1) *Separation* $(P, \{\frac{@@\ldots@@}{k+1}\}, T_1)$
(2-2) *Separation* $(P, \{\frac{@@\ldots@@}{k-1}\}, T_1)$
(2-3) *Discard* (T_1)

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

VI. COUNT THE EDGES OF EACH COLLECTION

In each element in data pool we count the number of edges, that have one vertex in one subset and another one in different subset.
In [4] introduce an algorithm for two subsets now. We want to use that algorithm for K subsets. in the following we illustrate algorithm for K=2 and then expand it to actual size.
For example, for strand

$\#B_7 2A_7 B_6 2A_6 B_5 2A_5 B_4 0A_4 B_3 1A_3 B_2 1A_2 B_1 1A_1 \#$ we have two subsets $\{7,6,5\}, \{3,2,1\}$.
We should add

$$y_{7,3}, y_{7,2}, y_{7,1}, y_{6,3}, y_{6,2}, y_{6,1}, y_{5,1}, y_{5,2}, y_{5,3}.$$

This is the algorithm for K=2.

For $r=1$ *to* $d=n$
For $d=1$ *to* $d=n$
(3-1) *Separation* $(P, \{B_d rA_d\}, T_1)$
For $i=1$ *to* $i=n$
For $j=1$ *to* $j=n$
(3-2) *Separation* $(T_1, \{B_i jA_i\}, T_2)$
(3-3) *Append* $(T_2, y_{r,j})$
(3-4) *Merge* (T_1, T_2)
(3-5) *Discard* (T_2)
End for
End for
(3-6) *Merge* (P, T_1)
End for
End for

Each of the above actions will conclude at $O(1)$. Therefore, the algorithm will terminate at $O(n^4)$.

VII. FIND THE STRANDS WITH MAXIMUM LENGTH

In this step, if we find the longest strand, that strands will show us the solution of problem. For example, for this

$\#B_7 2A_7 B_6 2A_6 B_5 2A_5 B_4 0A_4 B_3 1A_3 B_2 1A_2 B_1 1A_1 \#$
$@@@@@ y_{7,3}, y_{7,2}, y_{7,1}, y_{6,3}, y_{6,2}, y_{6,1}, y_{5,1}, y_{5,2}, y_{5,3}$

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 5, Issue 7, July 2016

Will be

$$\#B_7 2A_7 B_6 2A_6 B_5 2A_5 B_4 0A_4 B_3 1A_3 B_2 1A_2 B_1 1A_1 \#$$
$$@@@@@\ y_{6,2}, y_{5,3}$$

because the length of other edges are equal to zero. Here the length of edge show that maximum solution. This algorithm calculate solution.

For $r = 1$ *to* $r = n$

(3−1) *Selection*$(P,\ 30*n + K*10 + 20 + (n-r)*10, T_1)$

(3-2) *If Detect(T) is yes,*

then end for else continue the circulation

End for

Note that, in each strand, the sub-strand @ can be repeated K times. We will present an example for n=7. The strand $\#B_7 2A_7 B_6 2A_6 B_5 2A_5 B_4 0A_4 B_3 1A_3 B_2 1A_2 B_1 1A_1 \# @@\ y_{6,2}, y_{5,3}$ is made up of sub-strand $B_7 2A_7 B_6 2A_6 B_5 2A_5 B_4 0A_4 B_3 1A_3 B_2 1A_2 B_1 1A_1$ with length 30*7 and 2 sub strands $\#,\#$ with length 20 and $@@$ with length 2*10 and the length of $y_{6,2}, y_{5,3}$ is 2*10 Hence the total length of this strand is $30*7 + 20 + 2*10 + 20$.

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

VIII. CONCLUSION

The version of this template is V2. Most of the formatting as you can see we can finish this algorithm in $O(n^4)$.

ACKNOWLEDGMENT

As the first work for DNA computing, Adleman[1] 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 $O(n)$ 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[5] demonstrated that Adleman's experiment could be used to determine the NP-complete Satisfiability (SAT) problem (the first NP-complete problem). In this paper, we propose a procedure for Maximum k-colourable Sub graph problem NP-complete problems in the Adleman–Lipton model. The procedure works in $O(n^4)$ steps for Maximum k-colourable Sub graph 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.

REFERENCES

[1] Adleman, L. (1994). Molecular computation of solutions to combinatorial problems. Science, 266(5187), 1021-1024.
[2] Dalvand, B., Safaei, S., & Nazari, M. (2009). Fast Parallel Molecular Solution to the Maximum Triangle Packing Problem on Massively Parallel Bio-Computing. FCS, (pp. 169-173).
[3] Dolati, A., Haghighat, M., & Safaee, S. (2008). Solving bin-packing problem in the Adleman–Lipton model. The First Conference and Workshop on Mathematical Chemistry, 1, pp. 70-74.
[4] Dolati, A., Haghighat, M., Safaei, S., & Mozaffar, H. (2008). Solving Minimum Beta-vertex Separator Problems in the Adleman-Lipton Model. FCS, (pp. 97-101).
[5] Lipton, R. (1995). DNA solution of hard computational problems. Science, 268(5210), 542-545.
[6] Safaei, S., Dalvand, B., Derakhshandeh, Z., & Safaei, V. (2009). Molecular Solutions for the Bin-packing and Minimum Makespan Scheduling Problems on DNA-based Supercomputing. PDPTA, (pp. 512-516).
[7] Safaei, S., Dalvand, B., Esmaeili, B., & Safaei, V. (2009). Molecular Solutions for the Minimum Edge Dominating Set Problem on DNA-based Supercomputing. FCS, (pp. 32-36).
[8] Safaei, S., Dalvand, B., Safaei, N., & Safaei, V. (2011). Molecular solutions for the maximum K-Facility dispersion problem on DNA-based supercomputing. FCS, (pp. 513-517).
[9] Safaei, S., Dolati, A., Haghighat, M., & Mozaffar, H. (2008). Linear Solving of Maximum Set Packing Problem by DNA Computing. SISN, 2, pp. 82-86.
[10] Safaei, S., Hakimi, M., & Safaei, V. (2010). Linear algorithm for Solving MAXIMUM DIRECTED CUT in Adleman-Lipton model. The Fourth Conference and Workshop on Mathematical Chemistry.
[11] Safaei, s., Hakimi, M., & Safaei, V. (2012). Molecular Solutions for the Minimum K-Edge Connected Subgraph Problems on DNA-based Supercomputing. The Sixth Conference and Workshop on Mathematical Chemistry.
[12] Safaei, s., Hakimi, M., & Safaei, V. (2013). Solving Minimum Ratio-Cut via DNA computations. The seventh Conference and Workshop on Mathematical Chemistry.
[13] Safaei, S., Mozaffar, H., & Esmaeili, B. (2008). Solving Minimum K-Center Problem in the Adleman? Lipton Model. FCS, (pp. 251-255).
[14] Sohrabi-Haghighat, M., Dolati, A., & Safaee, S. (2015). Solving several kinds of constrained shortest path problems via DNA computations. Applied mathematics in Engineering, Management and Technology, 527-533.

BIOGRAPHY

Dr. Akbar Moazam, PhD Computer Information Systems, Database Administrator, Isfahan University, Isfahan, Iran