Fast Parallel Molecular Solution for Longest and Shortest Paths with Forbidden Pairs

Mahsa Hakimi¹, Akbar Moazzam², Babak Dalvand³, Mehdi S Haghighat¹, Ahmed A. Sallam⁵, Ardashir Dolati⁶

1. INTRODUCTION

With Watson-Crick complementarily and massive parallelism features of DNA, we can solve NP hard problems in linear or polynomial time. It is clear we do not have any polynomial algorithms to solve NP hard problems with silicon base computers; But DNA computing provides powerful features which can solve those problems in polynomial steps. 

Adleman [1] solved Hamiltonian path problem of size n. That was the first algorithm for DNA computing. Lipton [5] solved the second NP hard problem with those operations. Some other NP-hard problems which have been solved [6-21].

In this paper, the DNA operations proposed by Adleman [1] and Lipton [5] are used to solve Longest path with forbidden pairs.

For a given Graph $G = (V, E)$ and a collection $C = \{(a_1, b_1), \ldots, (a_m, b_m)\}$ pairs of vertices from $V$, a minimum solution is simple path in $G$ that contains at most one vertex from each pair in $C$ with longest path.

In Section 2, the Adleman–Lipton model is introduced in detail. Section 3 we will present a DNA algorithm for solving the Longest and shortest path with forbidden pairs problem and the complexity of the proposed algorithm is described. We give conclusions in Section 4.

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 [14].

A DNA (deoxyribonucleic acid) is a polymer, which is strung together from monomers called deoxyribonucleotides [14]. Distinct nucleotides are detected only with their bases [13].

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’-TGCCCTACAGT-5’ can form a double strand. We also call them as the complementary strand of each other [12].

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 [4]. 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 [3]. The DNA operations proposed by Adleman and Lipton [2] 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}). The following operations perform on tubes [2]:

Abstract: Adleman demonstrate that we can deoxyribonucleic acid (DNA) strands to solve an instance of the Hamiltonian path problem (HPP). One year later, Lipton solved another NP hard problem. In this paper, we use this model for developing a new DNA algorithm to solve longest and shortest path with forbidden pairs. Our algorithm works in polynomial time.

Keywords: DNA computing, longest and shortest path with forbidden pairs.
We introduced an algorithm [11] to encode all paths. for instance

\[ #A_1B_1Y_{1,2}A_2B_2Y_{2,4}A_4B_4Y_{4,2}A_2B_2Y_{2,5}A_5B_5Y_{5,7}A_7B_7 # \] denotes 1→2→4→6→7.

III. SOLVING LONGEST AND SHORTEST FORBIDDEN PATH BY ADLEMAN-LIPTON MODEL

Let \( G = (V, E) \) be a directed graph with the set of vertices being \( V = \{A_k | k = 1, 2, \ldots, m\} \) and the set of edges being \( E = \{e_i | i = 1, 2, \ldots, n\} \) [3]. Let |\( E | = d. \) In the following, the symbols \( #, X, Y, A_i, B_j (k = 1, 2, \ldots, m, j = 1, 2, \ldots, m) \) denote distinct DNA single strands with same length, say 10-mer. And \( |\cdot| \) denotes the length of the DNA single strand. Obviously, the length of the DNA single strands greatly depends on the size of the problem involved to distinguish all above symbols [3]. We assume the DNA single strand \( Y_{i,j} \) is used to denote the weights on the edges \( e_{i,j} \) in \( E \), so \( |Y_{i,j}| = w_{i,j} \).

Let \( L = \text{Max}_{e_{i,j} \in E} \{w_{i,j}\} \). Suppose that all weights in the given graph are commensurable, i.e., there exists a number \( y \) such that each weight is an integral multiple of \( y \) (here, take \( y = 10 \)) in the following discussion. In our previous work [11] we defined \( P \) and \( Q \) tubes and introduced an algorithm to produce all paths from a vertex to another one. We use that algorithm to produce all paths.

IV. PRODUCE ALL PATHS WHICH CONTAINS AT MOST ONE VERTEX IN EACH PAIR IN C

We introduced an algorithm [11] to encode all paths. for instance

\[ \#A_1B_1Y_{1,2}A_2B_2Y_{2,4}A_4B_4Y_{4,2}A_2B_2Y_{2,5}A_5B_5Y_{5,7}A_7B_7 # \] denotes 1→2→4→6→7.
In this step, we want to select all paths which have at most one vertex in each pair of C and remove the remaining set. If we have a pair like (5,2) in C, then 1→2→4→6→7 is acceptable answer. but for (4,1) is not acceptable because vertex 1 and 4 are in that path.

In Collection C, we show the i-th pair with Ci1,Ci2 which denotes the first and second element of i-th pair.

\[
\text{For } i = 1 \text{ to } m
\]
\[
\begin{align*}
(1-1) & \text{ Separation } (P, \{A_{i1}B_{i1}\},T_1) \\
(1-2) & \text{ Separation } (T_1, \{A_{i2}B_{i2}\},T_2) \\
(1-3) & \text{ Merge } (P,T_1) \\
(1-4) & \text{ Discard } (T_2)
\end{align*}
\]

End for

In 1-2, we choose all paths which contain the first element of i-th pair, and put them in T1, if any of those paths contains the second element of i-th path, it is invalid path and we need to remove it. Here we have simple loop then our algorithm is O(m) which is depends on the size of set C.

V. FIND THE LONGEST AND SHORTEST ANSWER

Each strand contains n number of \( A_iB_i \) and two # and the length of \( A_iB_i \) are 20. also, the length ||#|| is 10. Each strand has n number of \( |T_{i,j}| = w_{i,j} \) then the maximum length of each strand is 20*n+20+n*L.

For instance, for this strand the maximum length is 20*7+20+7*L
\[
#A_1B_1Y_{1,2}A_2B_2Y_{2,4}A_4B_4Y_{4,2}A_2B_2Y_{2,5}A_5B_5Y_{5,7}A_7B_7#
\]

For \( i = 1 \) to \( n*L/10 \)
\[
\text{if Detect } (T_1) \\
\text{exit()}
\]

End for

For finding the shortest one we use this algorithm.

For \( i = 1 \) to \( n \)
\[
\text{if Detect } (T_1) \\
\text{exit()}
\]

End for

L is a constant, then this algorithm will terminate in O(n). with this algorithm we can find shortest path and longest path in O(n).
Then our algorithm will terminate in O(m+n).

VI. CONCLUSION

In this paper, we proposed new polynomial algorithm for two of NP-Hard Problems.

As you can see this algorithm will finish in \( O(n + m) \)

REFERENCES


