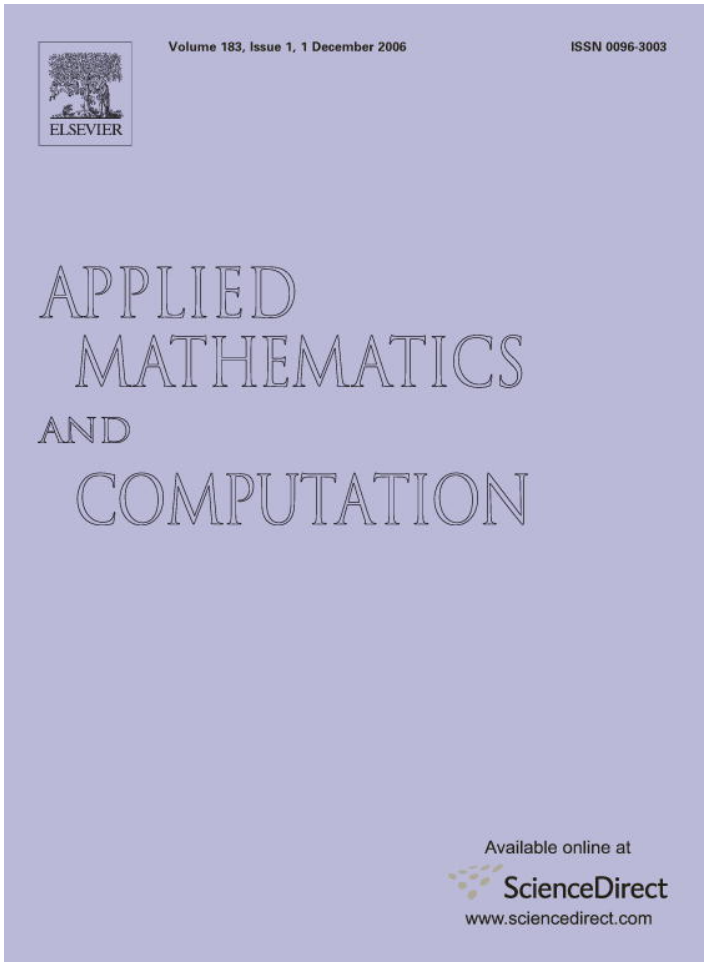

Available online at www.sciencedirect.com
ScienceDirect
Applied Mathematics and Computation 183 (2006) 79–84
APPLIED MATHEMATICS AND COMPUTATION
www.elsevier.com/locate/amc

A DNA procedure for solving the shortest path problem ☆

Zhaocai Wang [a,c], Dongmei Xiao [a,c], Wenxia Li [b,c,*,1], Lin He [c]

[a] Department of Mathematics, Shanghai Jiao Tong University, Shanghai 200030, PR China
[b] Department of Mathematics, East China Normal University, Shanghai 200062, PR China
[c] Bio-X DNA Computer Consortium, Shanghai Jiao Tong University, Shanghai 200030, PR China

Abstract

In this paper, we consider a procedure for solving the shortest path problem in the Adleman–Lipton model. The procedure works in O(n) steps for the shortest path problem of an edge-weighted graph with n vertices.

Keywords: The shortest path problem; NP-complete problem; Adleman–Lipton model; DNA computing

1. Introduction

In recent works for high performance computing, computation with DNA molecules, i.e., DNA computing, has considerable attention as one of non-silicon based computing. Watson-Crick complementarity and massive parallelism are two important features of DNA. Using the 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, e.g., Adleman [1] for Hamiltonian path problem – the first work for DNA computing, Lipton [11] for satisfiability (SAT) problem (the first NP-complete problem), Ouyang et al. [13] for the maximal clique problem, etc. Meanwhile, procedures for primitive operations, such as logic or arithmetic operations, have been also proposed so as to apply DNA computing on a wide range of problems [2–4,6–8,16,17]. However, most of the previous works in DNA computing do not require the consideration of the representation of numerical data in DNA strands. In fact, many practical applications in the real world involve edge-weighted graph problems such as shortest path problem, the travelling-salesman problem, etc. Therefore, representation of numerical data in DNA strands is an important issue toward expanding the capability of DNA computing to solve numerical optimization problems. There have been some previous works to represent the numerical data with DNA. Narayanan et al. [12] presented a conceptual encoding method that represents costs with the lengths of DNA strands. Shin et al. [15] proposed a method for representing the real

☆ Supported by Bio-X DNA Computer Consortium No. 03DZ14025.
* Corresponding author. Address: Department of Mathematics, East China Normal University, Shanghai 200062, PR China.
E-mail address: wxli@math.ecnu.edu.cn (W. Li).
[1] Supported by National Science Foundation of China #10371043 and Shanghai Priority Academic Discipline.

0096-3003/$ - see front matter © 2006 Elsevier Inc. All rights reserved.
doi:10.1016/j.amc.2006.05.061

numbers in fixed-length DNA strands by varying the number of hydrogen bonds. Yamamura et al. [18] proposed a concentration control method which encoded the numerical data by means of the concentrations of DNA strands. Lee et al. [9] introduced a novel encoding method that utilizes a temperature gradient to design the sequences so that the DNA strands for higher-cost values have higher melting temperatures than those for lower-cost values.

In this paper, a DNA procedure is presented for figuring out solutions of the shortest path problem: for an edge-weighted graph $G = (V, E)$ find a path starting and ending at the specified vertices such that the total weights on the path is smallest. For instance, the edge-weighted graph G in Fig. 1 defines such a problem. We assume that the starting and ending vertices are 1 and 7, respectively. It is easy to see that the path $1 \to 2 \to 3 \to 4 \to 6 \to 7$ with total weights 8 is a solution to the shortest path problem for graph G in Fig. 1. We encode the numerical data by means of the lengths of DNA strands, the same way as that in [12]. A DNA procedure is formally presented by means of the DNA operations proposed by Adleman [1] and Lipton [11]. Since the shortest path discussed in the present paper is not required to go through each vertex, e.g., the solution for the graph G in Fig. 1 does not go through the vertex 5, we use the *append* operation in the design of data pool so that the shortest path can be found out by comparing the lengths of the DNA strands.

The rest of this paper is organized as follows. In Section 2, the Adleman–Lipton model is introduced in detail. Section 3 introduces a DNA algorithm for solving the shortest path problem and the complexity of the proposed algorithm is described. We give conclusions in Section 4.

2. The Adleman–Lipton model

Bio-molecular computers work at the molecular level. Because biological and mathematical operations have some similarities, DNA, the genetic material that encodes for living organisms, is stable and predictable in its reactions and can be used to encode information for mathematical systems.

A DNA (deoxyribonucleic acid) is a polymer, which is strung together from monomers called deoxyribonucleotides [14]. Distinct nucleotides are detected only with their bases. Those bases are, respectively, abbreviated as A (adenine), G (guanine), C (cytosine) and T (thymine). Two strands of DNA can form (under appropriate conditions) a double strand, if the respective bases are the Watson-Crick complements of each other – A matches T and C matches G; also 3′ end matches 5′ end, e.g., the singled strands 5′ACCGGATGTCA3′ and 3′TGGCCTACAGT5′ can form a double strand. We also call the strand 3′TGGCCTACAGT5′ as the complementary strand of 5′ACCGGATGTCA3′ and simply denote 3′TGGCCTACAGT5′ by $\overline{\text{ACCGGATGTCA}}$. The length of a single stranded DNA is the number of nucleotides comprising the single strand. Thus, if a single stranded DNA includes 20 nucleotides, it is called a 20 mer. The length of a double stranded DNA (where each nucleotide is base paired) is counted in the number of base pairs. Thus, if we make a double stranded DNA from a single stranded 20 mer, then the length of the double stranded DNA is 20 base pairs, also written as 20 bp.

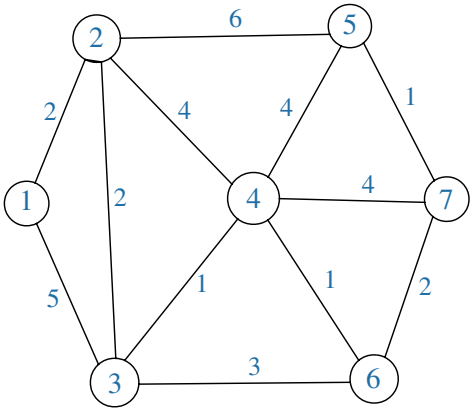

Fig. 1. An edge-weighted graph G with 7 vertices.

The DNA operations proposed by Aldeman [1] and Lipton [11] are described below. These operations will be used for figuring out solutions of the shortest path problem in this paper. 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*(T_1, 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, T_2): for a given test tube T_1 it produces a test tube T_2 with the same contents as T_1;
(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 strand containing $\begin{bmatrix} \sigma_0\sigma_1 \\ \overline{\sigma_0\sigma_1} \end{bmatrix}$ in T into two double strands as follows:

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

(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 [14], we assume that the complexity of each manipulation is O(1) steps.

3. DNA algorithm for the shortest path problem

Let $G = (V, E)$ be an edge-weighted graph with the set of vertices $V = \{v_k \mid k = 1, 2, \ldots, n\}$ and the set of edges $E = \{e_{i,j} \mid \text{for some } 1 \leqslant i, j \leqslant n, i \neq j\}$. Note that both $e_{i,j}$ and $e_{j,i}$ are in E if the vertices v_i and v_j are connected by an edge. Without loss of generality, we assume that v_1 and v_n are the starting and ending vertices, respectively. Let $|E| = s$. Then $s \leqslant \frac{1}{2}n(n-1)$.

In the following, we use the symbols X, #, A_k and B_k $(k = 1, 2, \ldots, n)$ to denote distinct DNA singled strands for which $\|\#\| = \|A_k\| = \|B_k\| = \frac{1}{2}\|X\|$, where $\|\cdot\|$ denotes the length of the DNA singled strand. For simplicity, we take $\|\#\| = 10$ mer. 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 [10]. 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. The DNA singled strands $w_{i,j}$ are used to denote the weights on the edges $e_{i,j} \in E$ with $\|w_{i,j}\| = k_{i,j}w$ if the corresponding weight equals $k_{i,j}y$ where w is a constant, e.g., take $w = 10$ mer in the following discussion. Let

$$P = \{w_{i,j}, \#A_1B_1, A_nB_n\#, A_kB_k \mid k = 2, 3, \ldots, n-1\},$$
$$Q = \{\overline{\#}, \overline{B_i w_{i,j} A_j}, |e_{i,j} \in E\} \quad \text{and} \quad R = \{X\}.$$

We design the following algorithm to solve the shortest path problem and give the corresponding DNA operations as follows:

(1) We choose all possible paths which start at v_1 and end at v_n.
(1-1) *Merge*(P, Q);
(1-2) *Annealing*(P);
(1-3) *Denaturation*(P);
(1-4) *Separation*$(P, \{\#A_1B_1\}, T_{\text{tmp}})$;
(1-5) *Discard*(P);
(1-6) *Separation*$(T_{\text{tmp}}, \{A_nB_n\#\}, P)$.
After the above six steps of manipulations, the singled strands in tube P will encode all paths which start at v_1 and end at v_n. For example, for the graph in Fig. 1 we have singled strands $\#A_1B_1w_{1,2}A_2B_2w_{2,4}A_4B_4w_{4,6}A_6B_6w_{6,7}A_7B_7\#, \#A_1B_1w_{1,2}A_2B_2w_{2,4}A_4B_4w_{4,2}A_2B_2w_{2,5}A_5B_5w_{5,7}A_7B_7\# \in P$, which denote the paths $1 \to 2 \to 4 \to 6 \to 7$ and $1 \to 2 \to 4 \to 2 \to 5 \to 7$, respectively. Note that some cycles may occur in paths of P, e.g., the latter path above contains a cycle $2 \to 4 \to 2$. This operation can be finished in O(1) steps since each manipulation above works in O(1) steps.

(2) Each singled strand in P denotes an admittable path in which the occurrence of the sub-strand A_kB_k with length 20 mer means that the path pass through the vertex v_k. Note that some admittable paths in P may not pass through all vertices of the graph G. For each given admittable path in P we hope that the information of those vertices which do not appear in the path are also contained in the path. This is done by the following manipulations.
For $k = 1$ to $k = n$
(2-1) *Separation*$(P, \{A_kB_k\}, T)$;
(2-2) *Append*(P, X);
(2-3) *Merge*(P, T).
End For
For example, for the graph in Fig. 1 the admittable path $1 \to 2 \to 4 \to 6 \to 7$ lacks vertices 3 and 5. After above manipulations its corresponding DNA singled strand becomes $\#A_1B_1w_{1,2}A_2B_2w_{2,4}A_4B_4w_{4,6}A_6B_6w_{6,7}A_7B_7\#XX$. The two appended single strands XX with total length 40 mer remedy the lack of vertices 3 and 5. In the above operation we use a "For" clause. Thus this operation can be finished in O(n) steps since each single manipulation above works in O(1) steps.

(3) We choose those strands of shortest length in P. Let $k = \max_{e_{i,j} \in E} k_{i,j}$.
For $m = 1$ to $m = kn$
(3-1) *Selection*$(P, 20n + 20 + 10m, T)$;
(3-2) If *Detect*(T) is "yes", then End For and the shortest path is obtained, else continue the circulation.
In the above operation we use a "For" clause. Thus If k is independent of n, then this operation can be finished in O(n) steps since each single manipulation above works in O(1) steps.

(4) Finally the *Read* operation is applied to giving the exact edges in the shortest path problem. For example, for the graph in Fig. 1, the shortest path is $1 \to 2 \to 3 \to 4 \to 6 \to 7$ with total weights 8.
(4-1) *Read*(T).

The following theorem tells that the algorithm proposed above really can give solutions of the shortest path problems in O(n) steps using DNA molecules.

Theorem 3.1. *Let $G = (V, E)$ be a graph with the set of vertices $V = \{v_k | k = 1, 2, \ldots, n\}$ and the set of edges $E = \{e_{i,j} |$ for some $1 \leqslant i, j \leqslant n, i \neq j\}$. Suppose that the maximum weight in the graph is independent of n. The algorithm proposed above can give solutions of the shortest path problems in* O(n) *steps using DNA molecules.*

Proof. After the operations of the second step, each singled strand in the tube P is of form:

$$\#A_{\ell_0}B_{\ell_0}w_{\ell_0,\ell_1}A_{\ell_1}B_{\ell_1}w_{\ell_1,\ell_2}A_{\ell_2}B_{\ell_2} \cdots w_{\ell_{t-1},\ell_t}A_{\ell_t}B_{\ell_t}w_{\ell_t,\ell_{t+1}}A_{\ell_{t+1}}B_{\ell_{t+1}}\#XX \cdots X \quad (1)$$

with $\ell_0 = 1$ and $\ell_{t+1} = n$, which corresponds an admittable path $v_1 \to v_{\ell_1} \to v_{\ell_2} \to \cdots \to v_{\ell_t} \to v_n$. Note that some admittable paths in the tube P may have cycles, i.e., $\ell_i = \ell_j$ for some $0 \leqslant i < j \leqslant t + 1$ in (1). It is clear that the path with the cycles removed still lies in the tube P, i.e., if $\ell_i = \ell_j$ for some $0 \leqslant i < j \leqslant t + 1$ in (1), then the path $v_1 \to v_{\ell_1} \to v_{\ell_2} \to \cdots \to v_{\ell_i} \to v_{\ell_{j+1}} \to v_{\ell_{j+2}} \to \cdots \to v_n$ also belongs to P. Therefore, the shortest paths surely have no cycles. Now let

$$S_1 = \#A_{\ell_0}B_{\ell_0}w_{\ell_0,\ell_1}A_{\ell_1}B_{\ell_1}w_{\ell_1,\ell_2}A_{\ell_2}B_{\ell_2}\cdots w_{\ell_{t-1},\ell_t}A_{\ell_t}B_{\ell_t}w_{\ell_t,\ell_{t+1}}A_{\ell_{t+1}}B_{\ell_{t+1}}\#X\cdots X$$

and

$$S_2 = \#A_{d_0}B_{d_0}w_{d_0,d_1}A_{d_1}B_{d_1}w_{d_1,d_2}A_{d_2}B_{d_2}\cdots w_{d_{s-1},d_s}A_{d_s}B_{d_s}w_{d_s,d_{s+1}}A_{d_{s+1}}B_{d_{s+1}}\#X\cdots X$$

be two singled strands with no cycles in the tube P where $\ell_0 = d_0 = 1$ and $\ell_{t+1} = d_{s+1} = n$. Then we have

$$\|S_1\| = 20n + 20 + \sum_{i=0}^{t}\|w_{\ell_i,\ell_{i+1}}\| \quad \text{and} \quad \|S_2\| = 20n + 20 + \sum_{i=0}^{s}\|w_{d_i,d_{i+1}}\|.$$

Thus the shortest paths, i.e., the paths with smallest total weights, correspond to those paths which corresponding singled strands have smallest length. As we see, the singled strands with smallest length are chosen after the third step above. Therefore, the algorithm proposed above really gives solutions of the shortest path problems.

As the algorithm described above, solutions of the shortest path problem for an edge-weighted graph with n vertices can be implemented in four operations using DNA molecules. The first and fourth operations costs O(1) steps, while the second and third operations costs O(n) steps. Thus solutions of the shortest path problem for an edge-weighted graph with n vertices can be figured out in O(n) steps using DNA molecules. □

4. Conclusions

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 [11] demonstrated that Adleman's experiment could be used to determine the NP-complete satisfiability (SAT) problem (the first NP-complete problem). Ouyang et al. [13] 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. [5] 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 the shortest path NP-complete problem in the Adleman–Lipton model. The procedure works in O(n) steps for the shortest path problem of an edge-weighted graph with n vertices. All our results in this paper are based on a theoretical model. However, the proposed procedure can be implemented practically since every DNA manipulation used in this model has been already realized in lab level.

References

[1] L.M. Adleman, Molecular computation of solution to combinatorial problems, Science 266 (1994) 1021–1024.
[2] P. Frisco, Parallel arithmetic with splicing, Romanian Journal of Information Science and Technology 2 (2002) 113–128.
[3] A. Fujiwara, K. Matsumoto, Wei Chen, Procedures for logic and arithmetic operations with DNA molecules, International Journal of Foundations of Computer Science 15 (2004) 461–474.
[4] F. Guarnieri, M. Fliss, C. Bancroft, Making DNA add, Science 273 (1996) 220–223.
[5] M.Y. Guo, W.L. Chang, M. Ho, J. Lu, J.N. Cao, Is optimal solution of every NP-complete or NP-hard problem determined from its characteristic for DNA-based computing, BioSystems 80 (2005) 71–82.
[6] V. Gupta, S. Parthasarathy, M.J. Zaki, Arithmetic and logic operations with DNA, in: Proceedings of 3rd DIMACS Workshop on DNA Based Computers, 1997, pp. 212–220.
[7] H. Hug, R. Schuler, DNA-based parallel computation of simple arithmetic, in: Proceedings of the 7th International Meeting on DNA Based Computers, 2001, pp. 159–166.
[8] S. Kamio, A. Takehara, A. Fujiwara, Procedures for computing the maximum with DNA strands, in: Humid R. Arabnia, Youngsong Mun (Eds.), Proceedings of the International Conference on DNA Based Computers, 2003.
[9] J.-Y. Lee, S.-Y. Shin, T.-H. Park, B.-T. Zhang, Solving traveling salesman problems with DNA molecules encoding numerical values, BioSystems 78 (2004) 39–47.

[10] D. Li, H. Huang, X. Li, X. Li, Hairpin formation in DNA computation presents limits for large NP-complete problems, BioSystems 72 (2003) 203–207.
[11] R.J. Lipton, DNA solution of HARD computational problems, Science 268 (1995) 542–545.
[12] A. Narayanan, S. Zorbalas, et al., DNA algorithms for computing shortest paths, in: J.R. Koza (Ed.), Proceedings of the Genetic Programming 1998, Morgan Kaufmann, 1998, pp. 718–723.
[13] Q. Ouyang, Peter D. Kaplan, S. Liu, A. Libchaber, DNA solution of the maximal clique problem, Science 278 (1997) 446–449.
[14] G. Păun, G. Rozeberg, A. Salomaa, DNA Computing, Springer-Verlag, 1998.
[15] S.-Y. Shin, B.-T. Zhang, S.-S. Jun, et al., Solving traveling salesman problems using molecular programming, in: P.J. Angeline (Ed.), Proceedings of the Congress on Evolutionary Computation 1999, IEEE Press, 1999, pp. 994–1000.
[16] D.M. Xiao, W.X. Li, Z.Z. Zhang, L. He, Solving maximum cut problem in the Adleman–Lipton model, BioSystems 82 (2005) 203–207.
[17] D.M. Xiao, W.X. Li, J. Yu, X.D. Zhang, Z.Z. Zhang, L. He, Procedures for a dynamical system on $\{0,1\}^n$ with DNA molecules, BioSystems 84 (2006) 207–216.
[18] M. Yamamura, Y. Hiroto, T. Matoba, Solutions of shortest path problems by concentration control, Lecture Notes Computer Science, vol. 2340, 2002, pp. 231–240.