Journal of Mathematical Chemistry Vol. 31, No. 3, April 2002 (© 2002)

DNA computing of bipartite graphs for maximum matching

Shiying Wang [*]

Department of Control Science and Engineering, Huazhong University of Science and Technology, Wuhan 430074 and
Department of Mathematics, Shanxi University, Taiyuan 030006, People's Republic of China

Received 1 March 2002

Let M be a matching in a graph G. It is defined that an M-augmenting path must obtain one element of M. In this paper, it is obtained that a matching M in a graph G is a maximum matching if and only if G contains no M-augmenting path and M is a maximal matching in G. It supplies a theoretical basis to DNA computing. A detailed discussion is given of DNA algorithms for the solutions of the maximal matching problem and maximum matching one in a bipartite graph.

KEY WORDS: maximum matching, maximal matching, augmenting path, DNA computing

1. Introduction

As it is known DNA (deoxyribonucleic acid) is a double helix in which the two coiled strands (chains) are composed each of only different nucleotides. Every nucleotide consists of phosphate, sugar and one of the following bases: adenine (abbreviated A), thymine (T), guanine (G) and cytosine (C). The two chains are held together by hydrogen bonds which exist only between pairs of complementary bases, which are A–T and G–C. It follows that knowing one chain, the other (complementary) can be easily reconstructed. DNA computing must not be confused with biocomputing. Usually, biocomputing means everything that computer scientists can do to help biologists to study genes. For example, algorithms and data structures have been developed to investigate the properties of the sequences of nucleotides in DNA or RNA, and those of amino acids in the primary structure of a protein. In DNA computing, instead, molecular biology is suggested to solve a problem for computer scientists. There are many reasons to investigate DNA computing. As known, the Hamiltonian Path Problem is an NP-complete one. Adleman's experiment [1] showed that DNA can be used to solve the Hamiltonian Path Problem and bio-steps are $O(n)$, where n is the number of the points of the directed graph. Matching theory has a wide range of application. Some practical problems can be converted into matching problems. For example, suppose one has two computers available and p jobs to be processed on these machines. We will assume that any job can be run on either machine. Indeed, we may assume that the computers are

[*] This work is supported by China National Science Foundation.

identical. Let us also suppose that p jobs are partially ordered in the sense that for any two jobs J_i and J_k, $J_i \leqslant J_k$ if J_i must be completed before J_k can be started. If all jobs require an equal amount of time to complete, what is the shortest possible time sufficient to run all p jobs? Let us model this situation using an undirected graph G as follows. Let the points of G be the jobs $J_1, J_2, \ldots, J_p$ and let J_i be adjacent to J_k if and only if they are incomparable in the partial order. So this problem belongs to the class of maximum matching problems. Thus, maximum matching problems in a graph naturally arise in computer science. The paper is organized into four sections. In the second section, we give a theorem of the matching theory. It supplies a theoretical basis to sections 3 and 4. In section 3, a detailed discussion is given of the DNA algorithm for the solution of the maximal matching problem in a bipartite graph. In section 4, on the basis of section 3, a detailed discussion is also given of the DNA algorithm for the solution of the maximum matching problem in a bipartite graph.

2. Matching theory

An undirected graph G consists of a finite non-empty set of elements $V(G)$ called points and a multi-set of unordered pairs of points $E(G)$ called lines. If uv is a line in graph G, line uv is said to join points u and v, to be *incident* with points u and v, and points u and v are said to be *adjacent*. Two lines which share a point are also said to be *adjacent*. A line with distinct ends is called a *link*. A subset M of $E(G)$ is called a matching in G if its elements are links and no two are adjacent in G. A matching M saturates a point v, and v is said to be *M-saturated*, if some line of M is incident with v; otherwise, v is *M-unsaturated*. An *M-alternating path* in G is a path whose line are alternately in $E(G) - M$ and M. An *M-augmenting path* which contains one element of M is an M-alternating path whose origin and terminus are M-unsaturated. A matching M is a maximum matching if G has no matching M' with $|M'| > |M|$. For $M \subseteq E(G)$, set $V(M) = \{v \in V(G)$: there is $x \in V(G)$ such that $vx \in M\}$. A matching M is a maximal matching if $G - V(M)$ has no line e such that $M \cup \{e\}$ is a matching in G. Let $G[S]$ denote the subgraph of G induced by S. $M_1 \oplus M_2$ denotes the symmetric difference of M_1 and M_2. If X is any set in $V(G)$, let $\Gamma(X)$ denote all points in $V(G)$ which are adjacent to at least one point of X. The notations and definitions not defined here can be found in [2,3].

Theorem 2.1. A matching M in a graph G is a maximum matching if and only if G contains no M-augmenting path and M is a maximal matching in G.

Proof. (*only if*) Let M be a maximum matching of G, and suppose, on the contrary, that G obtains an M-augmenting path $v_0v_1 \ldots v_{2m+1}$. Define $M' \subseteq E(G)$ by

$$M' = \bigl(M - \{v_1v_2, v_3v_4, \ldots, v_{2m-1}v_{2m}\}\bigr) \cup \{v_0v_1, v_2v_3, \ldots, v_{2m}v_{2m+1}\}.$$

Then M' is a matching of G, and $|M'| = |M| + 1$. Thus M is not a maximum matching in G, a contradiction. Therefore G contains no M-augmenting path.

M is maximal because M is maximum in G.

(*if*) Let G contain no M-augmenting path and let M be a maximal matching in G. Suppose, on the contrary, that M is not a maximum matching in G. Let M' be a maximum matching in G. Then $|M'| > |M|$. If $M \subset M'$, then there exists a line e belonging to $M' - M$ in $G[V(G) - V(M)]$. Therefore $M \cup \{e\}$ is a matching of G, a contradiction. Thus $M \not\subseteq M'$. Let $H = M \oplus M'$. Then $M \cap H \neq \emptyset$ and $M' \cap H \neq \emptyset$. If there exists a line $e \in M \cap H$ which is not adjacent to any line of $M' \cap H$, then $M' \cup \{e\}$ is a matching of G, contradicting the maximality of M'. Therefore every line of $M \cap H$ is adjacent to one or two lines of $M' \cap H$. If there exists a line $e' \in M' \cap H$ which is not adjacent to any line of $M \cap H$, then e' is a line in $G[V(G) - V(M)]$ and $M \cup \{e\}$ is a matching in G, contradicting the maximality of M. Therefore every line of $M' \cap H$ is adjacent to one or two lines of $M \cap H$. Thus each component of H is either an even cycle with lines alternately in M and M' or else a path with lines alternately in M and M'. By $|M'| > |M|$, there must be a path component P of H, and P must start and end with lines of M'. The origin and terminus of P, being M'-saturated in H, are M-unsaturated in G. Thus P is an M-augmenting path in G, a contradiction. Therefore M is maximum in G. The proof is complete. $\square$

3. DNA computing of bipartite graphs for maximal matching

In this section, G denotes a bipartite graph.

A function $f : A \to B$ is said to be injective (or one-to-one) provided

$$\text{for all } a, a' \in A, \quad a \neq a' \ \Rightarrow\ f(a) \neq f(a').$$

A function f is surjective provided $f(A) = B$. A function f is said to be bijective (or a one-to-one correspondence) if it is both injective and surjective. Let $\{A_i \mid i \in I\}$ be a family of sets indexed by a (nonempty) set I. The Cartesian product of the sets A_i is the set of all functions $f : I \to \bigcup_{i\in I} A_i$ such that $f(i) \in A_i$ for all $i \in I$. It is denoted $\prod_{i\in I} A_i$. If $I = \{1, 2, \ldots, t\}$, the product $\prod_{i\in I} A_i$ is often denoted by $A_1 \times A_2 \times \cdots \times A_t$ and is identified with the set of all ordered t-tuples $(a_1, a_2, \ldots, a_t)$, where $a_i \in A_i$ for $i = 1, 2, \ldots, t$.

Let S_n be the symmetric group. The product of cycles $\alpha\beta$ means β followed by α. For example, (12)(13)=(132). We will be using a one-row representation for permutation. Thus, the permutation whose cycle representation is (12)(345) will be represented by us as 21453.

Let m be a minimum integer such that $4^m \geqslant n$. Set $\langle n\rangle = \{1, 2, \ldots, n\}$, $\langle 4^m\rangle = \{1, 2, \ldots, 4^m\}$ and $A_i = \{1, 2, 3, 4\}$.

Theorem 3.1. There exists an injection from $\langle n\rangle$ to $\prod_{i=1}^m A_i$.

Proof. There exists a bijection f from $\langle 4^m\rangle$ to $\prod_{i=1}^m A_i$. Because of $n \leqslant 4^m$, f is an injection from $\langle n\rangle$ to $\prod_{i=1}^m A_i$. The proof is complete. $\square$

We will use the set $\{A, T, G, C\}$ instead of $\{1, 2, 3, 4\}$.

Corollary 3.2. Let $B_i = \{A, T, G, C\}$. Then there exists an injection from $\langle n\rangle$ to $\prod_{i=1}^{m} B_i$.

By corollary 3.2, let f be an injection from $\langle n\rangle$ to $\prod_{i=1}^{m} B_i$. For any $g = i_1 i_2 \dots i_n \in S_n$, let $f(g) = f(i_1) f(i_2) \dots f(i_n)$ and $f^-(g) = f^-(i_n) \dots f^-(i_2) f^-(i_1)$. For example, $f: 1 \to AT$, $2 \to AG$, $3 \to AC$, $4 \to TG$, $5 \to TC$. For $g = 12345$, $f(g) = ATAGACTGTC3'$ and $f^-(g) = 3'CTGTCAGATA$. The symbol $\overline{f(g)}$ is used to indicate the complementary base of each base of $f(g)$. For example, $f(g) = AGCT3'$ and $\overline{f(g)} = 3'TCGA$. Let $g^* \in S_n$ with $f(g) = \overline{f^-(g^*)}$. Let $f(S_n) = \{f(g)\colon g \in S_n\}$ and $S_c = \{f(g)\colon \overline{f^-(g)} = f(g), g \in S_n\}$. Then $S_c \subseteq f(S_n)$.

Proposition 3.3. $\overline{f^-(g)} = f(g)$ if and only if $f(g)$ can form a hairpin (a completely complementary double strand).

Proof. (*only if*). Suppose $\overline{f^-(g)} = f(g)$. Then nm is even (n and m are defined as above). The ith base of $f(g)$ is complementary to the ith base of $f^-(g)$. The ith base of $f^-(g)$ and the $(mn - i + 1)$th base of $f(g)$ are the same. Therefore the ith base of $f(g)$ is complementary to the $(mn - i + 1)$th base of $f(g)$. Thus $f(g)$ can form a hairpin.

(*if*). Suppose that $f(g)$ can form a hairpin. Then the ith base of $f(g)$ is complementary to the $(mn - i + 1)$th base of $f(g)$. The $(mn - i + 1)$th base of $f(g)$ and the ith base of $f^-(g)$ are the same. Therefore the ith base of $f(g)$ is complementary to the ith base of $f^-(g)$, i.e., $\overline{f^-(g)} = f(g)$. The proof is complete. □

Proposition 3.4. Let $g_i, g_j \in S_n$ with $g_i \neq g_j$. Then $f(g_i) \neq f(g_j)$.

Proof. Let $g_i = i_1 i_2 \dots i_n$ and $g_j = j_1 j_2 \dots j_n$. Since $g_i \neq g_j$, there is an integer b ($1 \leqslant b \leqslant n$) such that $i_b \neq j_b$. $f(i_b) \neq f(j_b)$ because f is injective. Therefore $f(g_i) \neq f(g_j)$. The proof is complete. □

Theorem 3.5. Let G be a bipartite graph with $|V(G)| = k$. Then there exists a positive integer n such that $n! \geqslant 2k + |S_c|$.

Proof. Let $g = i_1 i_2 \dots i_n$ with $f(g) \in S_c$. Then by proposition 3.3, $f(g)$ can form a hairpin. Therefore the ith base of $f(g)$ is complementary to the $(mn - i + 1)$th base of $f(g)$. Thus, $f(i_1) = \overline{f^-(i_n)}, \dots, f(i_{n/2}) = \overline{f^-(i_{n/2+1})}$. Therefore $|S_c| \leqslant n(n-2) \cdots 2$. We replace n with $n+1$. We have the following:

$$\begin{aligned}
&(n+1)! - (n+1)(n-1)(n-3)\cdots 3\\
&\quad = (n+1)\bigl(n! - (n-1)(n-3)\cdots 3\bigr) > (n+1)\bigl(n! - n(n-2)\cdots 4\cdot 2\bigr)\\
&\quad = \bigl(n! - n(n-2)\cdots 2\bigr) + n\bigl(n! - n(n-2)\cdots 2\bigr).
\end{aligned}$$

Therefore there exists a positive integer n such that $n! \geqslant 2k + |S_c|$. The proof is complete. $\square$

By theorem 3.5, there exists a minimum integer n such that $n! \geqslant 2k + |S_c|$. Let $V(G) = \{1, 2, \ldots, k\}$. We give the following.

Label algorithm 3.6.

Step 0. $i := 1$.

Step 1. If $i = k + 1$, then stop.

Step 2. Let $f(g_i) \in f(S_n) - S_c$. Assign $f(g_i)$ to the label of the point i of G. $f(S_n) - S_c := f(S_n) - \{f(g_i), f(g_i^*)\}$ and $i := i + 1$. Return to step 1.

We begin by labeling two test tubes: Points and Lines. Let us abbreviate these labels to PO and LI.

Design 3.7. For every $i \in V(G)$, by label algorithm 3.6 the point i is labeled by $f(g_i)$. Therefore PO $= \{f(g_i)\colon 1 \leqslant i \leqslant k\}$. In LI we place the following molecules that encode any line $(ij) \in E(G)$:

$$\overline{f(g_i)f(g_j)},$$

where $f(g_i)$ is the label of the point i and $f(g_j)$ is the label of the point j.

Theorem 3.8. The following is contained:

(a) all labels of the points of G are different;

(b) the label of each point of G does not form a hairpin;

(c) the label of each arc of G does not form a hairpin.

Proof. (a) By proposition 3.4 and label algorithm 3.6, the all labels of the points of D are different.

(b) By proposition 3.3 and label algorithm 3.6, the label of each point of G does not form a hairpin.

(c) By proposition 3.3, label algorithm 3.6 and design 3.7, the label of each arc of G does not form a hairpin. The proof is complete. $\square$

DNA algorithm 3.9.

Step 1. Add PO and LI. Add a ligase. (Allow time for ligation.)

Step 2. Make a gel separation.

Step 3. Make polymerase chain reaction and sequence for all ssDNAs (single stranded DNA molecules) of length $= n$ and dsDNAs (double stranded DNA molecules).

Comment. (a) We can obtain all points corresponding to all ssDNAs of length $= n$. They are exposed.

(b) We can obtain all lines corresponding to all dsDNAs. They form a maximal matching.

4. DNA computing of bipartite graphs for maximum matching

Let $G = (U, W)$ be a connected bipartite graph with the bipartition U and W. Let M be a maximal matching of $G = (U, W)$ and let M cover $U_1 \subseteq U$ and $W_1 \subseteq W$. By DNA algorithm 3.9, we obtain M, U_1 and W_1. Set $U' = U - U_1$, $W' = W - W_1$ and $U'' = \Gamma(W')$. $U'' \subseteq U$ because G is bipartite. $U' \cap U'' = \emptyset$ because M is a maximal matching. If U' or $W' = \emptyset$, then M is maximum. So suppose that $U' \neq \emptyset \neq W'$. Then $U'' \neq \emptyset$. An augmenting path must start by constructing an alternating path from the exposed points. Because an augmenting path must have one endpoint in U and the other in W, it is no loss of generality to start growing alternating paths only from exposed points of U.

The digraph with respect to $G = (U, W)$ and M, denoted by $D = (U, A)$, is the digraph with point set U and arc set A, where $(u_1, u_2) \in A$ if and only if u_1 is adjacent to the mate of u_2 in G.

Theorem 4.1. Let M be a maximal matching of $G = (U, W)$ and $u_i \in U$, $w_i \in W (1 \leqslant i \leqslant t, t \geqslant 2)$. Then there exists an M-augmenting path $u_1 w_1 u_2 w_2 \ldots u_t w_t$ in G if and only if there exists a directed path $u_1 u_2 \ldots u_t$ with $u_1 \in U'$ and $u_t \in U''$ in $D = (U, A)$.

Proof. (*only if*) Let $u_1 w_1 u_2 w_2 \ldots u_t w_t$ be an M-augmenting path in G. Then $\{w_i u_{i+1}\colon 1 \leqslant i \leqslant t-1\} \subseteq M$, and u_1 and w_t are exposed in G. Therefore $u_1 \in U'$, $w_t \in W'$ and $u_t \in U''$ in G. By the definition of D, $u_1 u_2 \ldots u_t$ is a directed path from $u_1 \in U'$ to $u_t \in U''$ in D.

(*if*) Let $u_1 u_2 \ldots u_t$ be a directed path from $u_1 \in U'$ to $u_t \in U''$ in D. Then by the definition of D, $u_1 w_1 u_2 w_2 \ldots u_t$ is an M-alternating path with $\{w_i u_{i+1}\colon 1 \leqslant i \leqslant t-1\} \subseteq M$ in G. Since $u_t \in U''$, there exists a point $w_t \in W'$ such that w_t is adjacent to u_t. Thus, $u_1 w_1 u_2 w_2 \ldots u_t w_t$ is an M-augmenting path. The proof is complete. $\square$

Example 4.2. In figure 1, the graph (a) is a bipartite graph. Let $M = \{u_1 w_4, u_3 w_2, u_4 w_3, u_5 w_6, u_6 w_5\}$. Then M is a maximal matching. $U' = \{u_2\}$, $W' = \{w_1\}$ and $U'' = \{u_1\}$. The digraph (b) corresponds to the graph (a).

We begin by labeling three test tubes: Initials, Middles and Finals. Let us abbreviate these labels to INI, MID and FIN.

Design 4.3. For every point $i \in U'$, in INI we place the following molecules that encode the point i:

$$
\begin{array}{l}
f(g_i) REGION1 f(g_i) \\
\overline{f(g_i)} REGION1, \\
*
\end{array}
$$

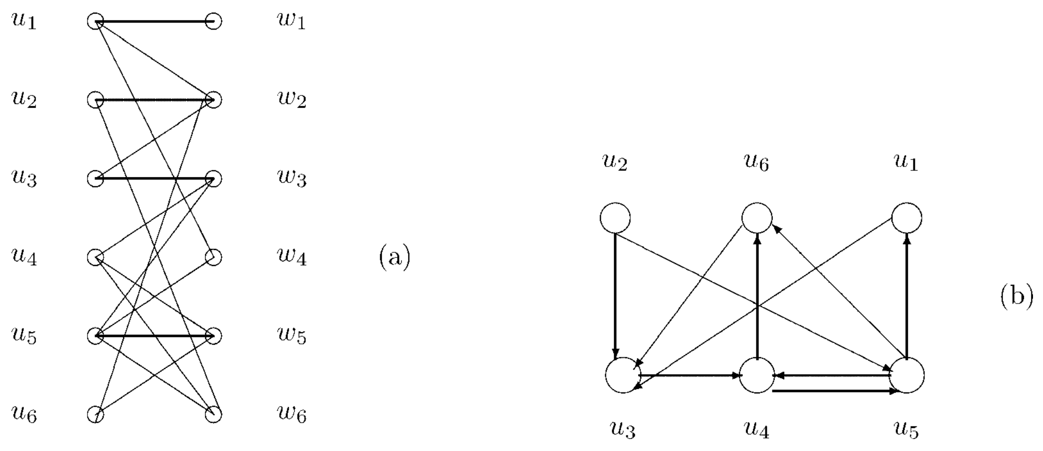

Figure 1.

where $f(g_i)$ is given by label algorithm 3.6 and the *REGION*1 is put by an especial dsDNAs which can be cut by a special restriction enzyme.

For every point $i \in U''$, in FIN we place the following molecules that encode the point i:

$$f(g_i)REGION2f(g_i)$$
$$REGION2\overline{f(g_i)},$$
$$*$$

where $f(g_i)$ is given by label algorithm 3.6 and the *REGION*2 is put by an especial dsDNAs which can be cut by a special restriction enzyme.

For every point $i \in U - U' - U''$, in MID we place the following molecules that encode the point i:

$$f(g_i)f(g_i),$$

where $f(g_i)$ is given by label algorithm 3.6.

For every arc (i, j), in MID we place the following molecules that encode the arc (i, j):

$$\overline{f(g_i)f(g_j)},$$

where $f(g_i)$ and $f(g_j)$ are given by label algorithm 3.6.

By label algorithm 3.6 and design 4.3, we have the following.

Theorem 4.4. The following is contained:

(a) all labels of the points of D are different;

(b) the label of each point of D does not form a hairpin;

(c) the label of each arc of D does not form a hairpin.

As is typical in DNA computing, each of three test tubes INI, FIN and MID will contain on the order of a picomole of each of the chosen molecules, i.e., on the order of a trillion of each.

DNA algorithm 4.5.

Step 1. Attract the molecules of INI and the molecules of FIN to the surface by a biotin. The starred region indicates the region of attachment to the surface.

Step 2. Add MID. Add a ligase. (Allow time for ligation.)

Step 3. Wash away excess MID and ligase.

Step 4. Add restriction enzyme 2. (Allow time for enzyme 2 to act.)

Step 5. Wash away all DNA not attached to the surface and all enzyme.

Step 6. Add ligase. (Allow time for ligation.)

Step 7. Wash away ligase.

Step 8. Add restriction enzyme 1. (Allow time for enzyme 1 digestion.)

Step 9. Wash away all DNA not attached to the surface and all enzyme.

Step 10. Add ligase. (Allow time for ligation.)

Step 11. Wash away ligase.

Step 12. Detach dsDNAs from the surface. Make a gel separation.

Step 13. Make polymerase chain reaction and sequence for all dsDNAs of length $\geqslant 4n$.

Theorem 4.6. If there exists a path from $i_1 \in U'$ to $i_v \in U''$ in D, the restriction enzyme 1 can only cut the dsDNAs of *REGION*1 and the restriction enzyme 2 can only cut the dsDNAs of *REGION*2, then the path from $i_1 \in U'$ to $i_v \in U''$ can be obtained by DNA algorithm 4.5.

Proof. Let $i_1e_1i_2e_2\ldots i_{v-1}e_{v-1}i_v$ be a path with $i_1 \in U'$ and $i_v \in U''$. Applying step 1 of DNA algorithm 4.5, attract the dsDNAs of INI of design 4.3 and the dsDNAs of FIN of design 4.3 to the surface by a biotin. Therefore,

$$\begin{array}{l} f(g_{i_1})\mathit{REGION}1 f(g_{i_1}) \\ \overline{f(g_{i_1})}\mathit{REGION}1 \end{array}$$

and

$$\begin{array}{r} f(g_{i_v})\mathit{REGION}2 f(g_{i_v}) \\ \mathit{REGION}2\overline{f(g_{i_v})} \end{array}$$

are attracted to the surface by a biotin. Applying step 2 of DNA algorithm 4.5, we can obtain the following path (abbreviated P1):

$$\begin{array}{l}\underline{f(g_{i_1})}REGION1\underline{f(g_{i_1})f(g_{i_2})f(g_{i_2})}\ldots\underline{f(g_{i_v})}REGION2\underline{f(g_{i_v})}\\ f(g_{i_1})REGION1f(g_{i_1})f(g_{i_2})f(g_{i_2})\ldots f(g_{i_v})REGION2f(g_{i_v}).\end{array}$$

Applying step 3 of DNA algorithm 4.5, wash away excess MID and ligase. Applying step 4 of DNA algorithm 4.5, we can obtain the following:

$$\begin{array}{l}\underline{f(g_{i_1})}REGION1\underline{f(g_{i_1})f(g_{i_2})f(g_{i_2})}\ldots\underline{f(g_{i_v})}REGION2\underline{f(g_{i_v})}\\ f(g_{i_1})REGION1f(g_{i_1})f(g_{i_2})f(g_{i_2})\ldots f(g_{i_v})REGION2f(g_{i_v}).\end{array}$$

Applying step 5 of DNA algorithm 4.5, cannot wash away the above dsDNAs because they are still attached to the surface. Applying steps 6, 7 of DNA algorithm 4.5, obtain P1 again. Applying step 8 of DNA algorithm 4.5, we can obtain the following:

$$\begin{array}{l}\underline{f(g_{i_1})}REGION1\underline{f(g_{i_1})f(g_{i_2})f(g_{i_2})}\ldots\underline{f(g_{i_v})REGIO}N2f(g_{i_v})\\ f(g_{i_1})REGION1f(g_{i_1})f(g_{i_2})f(g_{i_2})\ldots f(g_{i_v})f(g_{i_v}).\end{array}$$

Applying step 9 of DNA algorithm 4.5, cannot wash away the above dsDNAs because they are still attached to the surface. Applying steps 10, 11 of DNA algorithm 4.5, obtain P1 again. Applying step 12 of DNA algorithm 4.5, detach P1 from the surface. Make a gel separation and obtain all dsDNAs corresponding to the paths from one of U' to one of U''. Clearly, P1 is one of them. Applying step 13 of DNA algorithm 4.5, make polymerase chain reaction and sequence for all dsDNAs of the paths from one of U' to one of U''. Finally, we can obtain P1. The proof is complete. □

Remark. If there is no path from one of U' to one of U'' corresponding to one dsDNAs, then there is no M-augmenting path in $G = (U, W)$. By theorem 2.1 M is maximum. If there is a path P from one of U' to one of U'' corresponding to one dsDNAs, then P corresponds to an M-augmenting path P' in $G = (U, W)$. Let $M' = M \oplus P'$. Then M' is a maximal matching in $G = (U, W)$. We proceed as above. Finally, we can obtain that there is no path from one of U' to one of U'' in $D = (U, A)$ corresponding to one dsDNAs. Thus we can obtain a maximum matching in G.

Acknowledgements

The author would like to thank Professor Yixun Lin and Professor Jin Xu.

References

[1] G. Păun et al., *DNA Computing* (Springer, Berlin, 1998).
[2] L. Lovász and M.D. Plummer, *Matching Theory* (Elsevier Science, New York, 1986).
[3] J.A. Bondy and U.S.R. Murty, *Graph Theory with Applications* (Macmillan Press, 1976).
[4] C.H. Papadimitriou, *Combinatorial Optimization: Algorithms and Complexity* (Prentice-Hall, Englewood Cliffs, NJ, 1982).
[5] G. Păun, *Computing with Bio-Molecules: Theory and Experiments* (Springer, Singapore, 1998).