Finite automata for testing uniqueness of Eulerian trails

Qiang Li
T-Life Research Center, Fudan University, Shanghai, China

Hui-Min Xie
Department of Mathematics, Suzhou University, Suzhou, China

July 17, 2005

Abstract
We investigate the condition under which the Eulerian trail of a digraph is unique, and design a finite automaton to examine it. The algorithm is effective, for if the condition is violated, it will be noticed immediately without the need to trace through the whole trail.

1 Introduction
The problem of finding an Eulerian trail in a traversable directed pseudograph is well solved, and a counting formula is given in [3, 2]. But in some applications, like reconstructing a string from its composition of short substrings, as discussed in various contexts [5, 2, 4, 1], uniqueness rather than the exact number is mostly cared about, so the tedious calculation seems unnecessary. Considering a trail as a symbolic sequence over the set of vertices, Kontorovich showed that the unique Eulerian trails form a regular language [4]. We present a different proof by characterizing its complement, which leads to an effective implementation of a deterministic finite automaton (DFA) that accepts it, and gain an insight into its structure from the aspect of minimal forbidden words.

2 Results
In the following, we will freely switch the concepts from the theories of graph and formal language, and when the latter viewpoint is emphasized, the set of vertices $V$ is noted $\Sigma$.

2.1 The language
Pevzner [6] proved that any two Eulerian trails of a digraph $G$ can be transformed into each other by a series of operations called rotations and transpo-
sitions. Roughly speaking, rotations correspond to the choice of initial vertex if the trail is closed, and a transposition swaps the order of two paths between a pair of vertices in the trail. Not losing generality, we always suppose that the initial vertex is fixed. Thus an Eulerian trail is not unique only if it has a transposition

\[ T : uaxbzaybv \to uaybzaxbv, \]

where \( a, b \in \Sigma \), and \( u, v, x, y, z \in \Sigma^* \). If \( a = b \), it degenerates to the form

\[ T : uaxayav \to uayaxav. \]

On the other hand, only \( x \neq y \) does not assure that the transposition makes a trail different, e.g., let \( x = ba \) and \( u = v = y = z = \epsilon \), then the trail in (1) becomes \( t = ababab \), which is invariant under the operation, and is actually unique. To eliminate this case, we further request that the two \( a \)’s before \( x \) and \( y \) on the left hand side of (1) or (2) are followed by distinct vertices. Then we call the corresponding transposition to be proper.

**Lemma 1.** Every non-identical transposition is equivalent to a proper transposition.

**Proof.** For any transposition \( T(t) \neq t \), we can write it in the form of (1) or (2). If both \( a \)’s are followed by \( a' \), then let \( u' \).

1. If \( a \neq b \), then \( t = u'xbzaybv \), where \( x \neq y \).
   
   (a) If \( x \neq \epsilon \) and \( y \neq \epsilon \), then we can write \( x = a'x' \) and \( y = a'y' \), and let \( z' = za \). Otherwise, \( a' = b \).
   
   (b) If \( x = \epsilon \), then we can write \( y = a'y' \), and let \( x' = za \).
   
   (c) If \( y = \epsilon \), then we can write \( x = a'x' \), and let \( y' = za \).

2. If \( a = b \), then \( t = u'xayav \), where \( x \neq y \).
   
   (a) If \( x \neq \epsilon \) and \( y \neq \epsilon \), then we can write \( x = a'x' \) and \( y = a'y' \). Otherwise, \( a' = a \).
   
   (b) If \( x = \epsilon \), then we can write \( y = a'y' \), and let \( x' = \epsilon \).
   
   (c) If \( y = \epsilon \), then we can write \( x = a'x' \), and let \( y' = \epsilon \).

Therefore, \( t \) has a transposition \( T' : u'a'x'bz'aybv \to u'a'y'bz'a'x'bv \) in case (1a) or \( T' : u'a'x'ay'x'v \to u'a'y'a'x'x'v \) in the other cases. Note \( T'(s) = T(s) \). Substitute \( T' \) for \( T \) and repeat the above process, we will eventually get an equivalent proper transposition.

We conclude that an Eulerian trail \( t \) is unique if and only if it does not have a proper transposition. Let \( L \) be the language composed of unique Eulerian trails and \( L' \) be the language composed of those with proper transpositions, then they are complementary to each other.

2
By the definition of proper transposition, all sequences in $L'$ have a unified form

$$t = uawaybv, \quad (3)$$

where $u, v, w, y \in V^*$, $b$ appears in $aw$, and the vertices next to the two $a$’s are distinct. It results in a right-linear grammar $G$ that generates $L'$:

$$S \rightarrow dS|aA_a, $$

$$A_a \rightarrow cB_{acc}|aC_{aa}, $$

$$B_{acb} \rightarrow dB_{acb}|dB_{acd}|aC_{cb}, $$

$$C_{cb} \rightarrow dB_b \ (d \neq c)|bR \ (b \neq c), $$

$$D_b \rightarrow dB_b|bR, $$

$$R \rightarrow dR|\epsilon, $$

where $a, b, c, d$ run over $\Sigma$. Therefore, $L'$ is a regular language, and $L = \mathcal{T}'$ is also regular.

2.2 The finite automaton

Technically we can construct a finite automaton that accepts $L$ from $G(L')$, but it is more convenient to design it directly, like the following.

Input alphabet

$$\Sigma = V.$$  

States

$$Q = P \times N \times C,$$

where

- $P = \Sigma \cup \{a_0\}$, where $a_0 \notin \Sigma$ denotes the beginning of the sequence, records the last inputed vertex,
- $N = (\Sigma \cup \{\epsilon\})^{m+1}$, where $m = |V|$, records the latest followings of every vertex including $a_0$,
- $C = \{\text{WHITE}, \text{BLACK}\}^m$ is the “color” of every vertex. A vertex is colored black if it is in a circuit $awa$, where the vertex following the tail $a$ differs from that of the head $a$.

Initial state

$$q_0 = (a_0, \epsilon^{m+1}, \text{WHITE}^m).$$

Final states

$$F = \{(p, n, c) \in Q \mid c \neq \text{BLACK}^m\}.$$
Transition function

1: procedure $\delta(q, a)$
2: if $n_p \neq \epsilon$ and $n_p \neq a$ then
3: \[ b \leftarrow p \]
4: repeat
5: \[ c_b \leftarrow \text{BLACK} \]
6: \[ b \leftarrow n_b \]
7: until $b = p$
8: end if
9: if $c_a = \text{BLACK}$ then
10: \[ c \leftarrow \text{BLACK} \]
11: end if
12: $n_p \leftarrow a$
13: $p \leftarrow a$
14: end procedure

Now we prove that the DFA $M = (Q, \Sigma, \delta, q_0, F)$ accepts $L$.

Proof. First we show $L(M) \subset L$ by proving its contrapositive. If $t \notin L$, then it has the form of (3). The design of $M$ assures that $c$ becomes $\text{BLACK}^m$ after $b$ is inputed and remains so, thus $M$ does not accept $t$.

Then we prove $L \subset L(M)$ by induction on the length of the input sequence $t$.

Basis: For $|t| = 0$, $t = \epsilon \in L$. Since $q_0 \in F$, $t \in L(M)$.

Induction: For $|t| > 0$, if $t = sa \in L$, then $s \in L$, and by the inductive hypothesis $s \in L(M)$. We prove $t \in L(M)$ by contradiction. Assume to the contrary that $t \notin L(M)$, then there are two cases:

1. If $c_a = \text{BLACK}$ just after $s$ is inputed, then $s$ must have the form $ubwy$, where $a$ appears in $bw$ and the vertices following the two $b$'s are distinct. Thus $sa \in L'$, which contradicts $t \in L$.

2. If $c_a = \text{WHITE}$ just after $s$ is inputed, then $s$ must have the form $upwp$, where $a$ appears in $pw$ and the vertex following the first $p$ is not $a$. Again $sa \in L'$, which contradicts $t \in L$.

We conclude that $L(M) = L$. \qed

2.3 Minimal forbidden words

Since $L$ is a factorial language, i.e. for any $t \in L$, all factors of $t$ also belong to $L$, it can be determined by its minimal forbidden words (MFW) [7]. A string $r$ is a minimal forbidden word of $L$ if $r \notin L$ while all the factors of $r$ belong to $L$.

We categorize $\text{MFW}(L)$ into sequences in the following two forms, which compose a language $L''$:

\[ r = axbzyb, a \neq b, \quad (4) \]
\[ r = axaya, \quad (5) \]
where

1. \( x \neq \epsilon \) or \( y \neq \epsilon \),
2. \( x, y, z \in L \),
3. \( x, y, z \) do not contain \( a, b \), and each two of \( x, y, z \) do not contain common vertices.

**Theorem 2.** \( L'' = MF\bar{w}(L) \).

**Proof.** By definition all words in \( L'' \) are minimal forbidden words. Then we prove that \( L'' \) is complete, i.e. \( L' \subset \Sigma^*L''\Sigma^* \). For any \( t \in L' \), it must has a form of (3), then \( r = awayb \notin L \) satisfies the condition 1. If it violates the condition 2, e.g. \( x \notin L \), then let \( t = x \). Repeat the above process until the condition 2 holds. Then if \( y \) contains a vertex \( c \) which appears in \( aw \), \( t \) must have a prefix \( away'c \notin L \). Therefore, \( t \) has a word \( r \) in the form (4) or (5) where \( y \) does not contain \( a, b \) or common vertex with \( x, z \). Since reversing every edge's direction in a graph does not change the number of its Eulerian trails, \( L \) is reversal. So we can also request that \( x \) does not contain \( a, b \) or common vertex with \( z \).

We can determine \( L'' \) by recursion on \( |\Sigma| \). For the simplest non-trivial case, say \( \Sigma = \{0,1\} \), \( L'' \) can be represented by a regular expression \( 001^+0 + 01^+00 + 110^+1 + 10^+11 \).

**Acknowledgement**

QL would like to thank Prof. Bai-Lin Hao for asking the probability of an Eulerian trail to be unique in some classes of graphs, and thanks Chan Zhou for indicating the reference [6] for him.

**References**

[1] V. Chauhan and A. Trachtenberg. Reconciliation puzzles [separately hosted strings reconciliation]. In *IEEE Global Telecommunications Conference (GLOBECOM)*, volume 2, pages 600–604, 2004.

[2] Bai-Lin Hao, Hui-Min Xie, and Shu-Yu Zhang. Compositional representation of protein sequences and the number of eulerian loops. arXiv:physics/0103028, 2001.

[3] D. Kandel, Y. Matias, R. Unger, and P. Winkler. Shuffling biological sequences. *Discrete Applied Mathematics*, 71(1-3):171–185, 1996.

[4] L. Kontorovich. Uniquely decodable n-gram embeddings. *Theoretical Computer Science*, 329(1-3):271–284, 2004.
[5] P. A. Pevzner. l-tuple DNA sequencing: computer analysis. *J Biomol Struct Dyn*, 7(1):63–73, 1989.

[6] P. A. Pevzner. *Computational Molecular Biology: An Algorithmic approach*, pages 75–78. MIT, 2000.

[7] Hui-Min Xie. *Grammatical Complexity and One-Dimensional Dynamical Systems*. World Scientific, Singapore, 1996.