ELSEVIER
Available online at www.sciencedirect.com
ScienceDirect
Journal of Computer and System Sciences 74 (2008) 870–874
JOURNAL OF COMPUTER AND SYSTEM SCIENCES
www.elsevier.com/locate/jcss

Finite automata for testing composition-based reconstructibility of sequences

Qiang Li [a,*], Huimin Xie [b]

[a] *T-Life Research Center, Fudan University, Shanghai 200433, China*
[b] *Department of Mathematics, Suzhou University, Suzhou 215006, China*

Received 9 April 2007; received in revised form 17 October 2007

Available online 28 October 2007

Abstract

Symbolic sequences uniquely reconstructible from all their substrings of length k compose a regular factorial language. We thoroughly characterize this language by its minimal forbidden words, and explicitly build up a deterministic finite automaton that accepts it. This provides an efficient on-line algorithm for testing the unique reconstructibility of the sequences.

Keywords: Uniqueness; Sequence reconstruction; Eulerian trail; Factorial language; Deterministic finite automata

1. Introduction

The problem of sequence reconstruction from composition has been raised in various contexts, like high throughput DNA sequencing [1], composition-based prokaryotic phylogenetics [10], and string embedding [6]. It considers whether a symbolic sequence can be uniquely recovered from the multiset of all its constituent "k-tuples." For example, the oligonucleotide sequences `TACTAGACT` and `TAGACTACT` have the same triple composition {`ACT`, `ACT`, `AGA`, `CTA`, `GAC`, `TAC`, `TAG`}, thus neither is uniquely reconstructible. Given a k-tuple composition, there exists a linear-time algorithm to determine whether a conforming sequence exists, and if yes it constructs one [2,8]. The number of sequences with a valid composition is given by the "modified BEST formula" [4,5,10], but the calculation can be tough, because the formula is based on the matrix-tree theorem in graph theory, which involves a determinant whose size is, in cases of interest, comparable with the length of the sequence. Alternatively, the set of uniquely reconstructible sequences can be investigated as a formal language. Recently, Kontorovich [6] proved that this language is regular, and conjectured that a finite automaton that accepts it can be efficiently constructed. The present paper supplies a different proof based on the results of Ukkonen [11] and Pevzner [9], and further characterizes this language by its minimal forbidden words. Finally we explicitly build up the associated deterministic finite automaton (DFA), which provides an efficient on-line algorithm for testing the uniqueness of reconstructions of sequences. We have implemented it in a `C++` program, and the source code is available on request.

* Corresponding author. Fax: +86 21 6565 2305.
E-mail addresses: q.li@fudan.edu.cn (Q. Li), szhmxie@pub.sz.jsinfo.net (H. Xie).

0022-0000/$ – see front matter
doi:10.1016/j.jcss.2007.10.004

2. Conventions and notation

We start by fixing some notation. The empty string will be denoted by ϵ. By convention, we denote by Σ the alphabet in consideration, Σ^* the set of all finite strings over Σ, and $\Sigma^+ = \Sigma^* \setminus \{\epsilon\}$ the nonempty strings. For the sake of convenience, we lay down the rule that low Latin letters a, b, c, d denote characters, and high letters $r, s, \ldots, z$ denote strings. We denote by $|s|$ the length of the string s, and $|s|_a$ the number of the character a's in s. The set of all characters occurring in s, i.e., $\{a \in \Sigma\colon |s|_a \geqslant 1\}$, will be denoted by $\mathrm{alph}(s)$. For a string $s = at$, the character a is said to be the *head* of s, denoted by $\mathrm{head}(s)$. Similarly, b is said to be the *tail* of the string $s = tb$, denoted by $\mathrm{tail}(s)$. For $s = uvw$, v is said to be a *factor* of s, and it is called a *left factor* if $u = \epsilon$. A factor v of s is said to be *proper* if $v \neq s$. For unexplained terms in formal language and automata theory, we refer the reader to the standard textbook [3].

Without losing generality, we only consider the problem of duple composition, i.e., $k = 2$. For $k > 2$, it can be easily reduced to the former case by considering the set of $(k-1)$-tuples, Σ^{k-1}, as the alphabet.

3. The complementary language

Ukkonen [11] conjectured and Pevzner [9] proved that any two sequences with the same composition can be transformed into each other by a series of operations called *rotations* and *transpositions*. A rotation

$$R: aubva \to bvaub$$

applies to a string whose head and tail are the same. This case is simple, and can be eliminated by preceding each string with a special character outside Σ. We will ignore it in the following, such that the frequencies of characters are also conserved, as is usually required in practice. A transposition

$$T: uaxbwaybv \to uaybwaxbv$$

exchanges a pair of nonoverlapping factors of a string (x and y), given that they are flanked by the same character on either side (respectively a and b). In case $a = b$ it has a degenerated form

$$T: uaxayav \to uayaxav.$$

Clearly, these operations do not alter the composition. We can unify them into the form

$$T: uxwyv \to uywxv, \tag{1}$$

where

$$\mathrm{tail}(u) = \mathrm{tail}(w) \quad \text{and} \quad \mathrm{head}(w) = \mathrm{head}(v). \tag{2}$$

We denote by L the language of uniquely reconstructible sequences, and L' its complement, then a string s is in L' only if it has a form $s = uxwyv$ subject to condition (2). However, this condition is not sufficient even in the constraint $x \neq y$. For example, the string 010101 has such a form with $(u, x, w, y, v) = (0, 10, 10, \epsilon, 1)$ while it is in L.

Remark 1. Generally, the string s is invariant under the transposition if and only if $xwy = ywx$, in other words $xwyw = ywxw$. It follows from Proposition 1.3.2 in [7] that the two words xw and yw commute if and only if they are the powers of the same word. Therefore we can write

$$x = (rt)^l r, \qquad w = t(rt)^m, \qquad y = (rt)^n r, \quad l, m, n = 0, 1, \ldots.$$

To rule out transpositions on strings in L, we add to the conditions in (2) that

$$\mathrm{head}(xw) \neq \mathrm{head}(yv), \tag{3}$$

while every string in L' still retains a transposition. This is justified by a slightly stronger proposition:

Proposition 2. *For any transposition $T: s \to s'$ with $s \neq s'$, it can be written in the form* (1)*, such that u is the longest common left factor of s and s'.*

Proof. Suppose s and s' have a common left factor $u' = ua$, then $\mathrm{head}(yw) = a$. We show that the transposition can be written in the form $T' : u'x'w'y'v \to u'y'w'x'v$. There are three cases:

(1) If $x \neq \epsilon$ and $y \neq \epsilon$, then we can write $x = ax'$ and $y = ay'$, and let $w' = wa$.
(2) If $x = \epsilon$, then we can write $w = ax'$ and $y = ay'$, and let $w' = a$.
(3) If $y = \epsilon$, then we can write $x = ax'$ and $w = ay'$, and let $w' = a$.

It turn out that $u'y'w'x'v = s'$.
By repeating this procedure we can extend u until the given condition holds. □

As an immediate corollary,

$$L' = \{s = uxwyv\text{: conditions (2) and (3) hold}\}.$$

Each subset L'_{abc} of L', defined by $\mathrm{tail}(u) = \mathrm{tail}(w) = a$, $\mathrm{head}(w) = \mathrm{head}(v) = b$, $\mathrm{head}(xw) = c$ and $\mathrm{head}(yv) \neq c$, clearly constitutes a regular language. Therefore, as regular languages are closed under union and complementation [3, p. 59], L' and L must also be regular languages. This is the main theorem in [6]. Moreover, we obtain a right-linear grammar for L' composed of productions of the following forms:

$$U \to dU \mid acZ_{acc} \mid aaA_{aa},$$
$$Z_{acc} \to bZ_{acb},$$
$$Z_{acb} \to dZ_{acb} \mid aA_{cb},$$
$$A_{cb} \to dY_b \ (d \neq c) \mid bV \ (b \neq c),$$
$$Y_b \to dY_b \mid bV,$$
$$V \to dV \mid \epsilon,$$

where U is the start symbol, and a, b, c, d run over Σ. It is of little interest to present the routine (but a bit lengthy) proof, instead we note that for given a, b, and c, this grammar generates L'_{abc}.

4. Minimal forbidden words

A language is said to be *factorial* (or *factorizable*) if it contains all factors of its members. Clearly L is factorial. A factorial language can be determined by its *minimal forbidden words* (MFWs, also known as distinct excluded blocks, or DEBs) [13]. A MFW of a language is a string that does not belong to the language while all its proper factors do. They help to understand the structure of the language.

Theorem 3. *A string r is a MFW of L if and only if $r = axwyb$, such that*

(1) $\mathrm{head}(w) = b$ *and* $\mathrm{tail}(w) = a$;
(2) $x \neq \epsilon$ *or* $y \neq \epsilon$;
(3) $x, w, y \in L$;
(4) $\mathrm{alph}(x)$, $\mathrm{alph}(w)$, *and* $\mathrm{alph}(y)$ *are mutually disjoint*;
(5) $|w|_a = 1$ *and* $|w|_b = 1$.

Proof. The sufficiency is trivial. And the necessity of the first three conditions is evident, so we only need to justify the last two conditions.

Suppose y contains a character b' which occurs in xw, then y has a left factor $y'b'$, and we can write $xw = x'w'$, with $\mathrm{head}(w') = b'$. It is the case that r has a left factor $r' = ax'w'y'b' \in L'$, which contradicts that r is a MFW. Therefore, $\mathrm{alph}(xw) \cap \mathrm{alph}(y) = \emptyset$. Clearly, reversing a string does not alter its membership of L, i.e., L is reversal. So similarly we have $\mathrm{alph}(x) \cap \mathrm{alph}(w) = \emptyset$. Hence condition (4) holds.

Suppose $|w|_a > 1$, then we can write $w = zay'a$. Let $x' = xz$, then $axw = ax'ay'a$.

If $x \neq \epsilon$, by $\mathrm{alph}(x) \cap \mathrm{alph}(w) = \emptyset$ we have $\mathrm{head}(x'a) \neq \mathrm{head}(y'a)$. It follows $ax'ay'a \in L'$, which contradicts that r is a MFW. Therefore $|w|_a = 1$. Similarly $|w|_b = 1$ if $y \neq \epsilon$.

If $x = \epsilon$, then $\mathrm{head}(x'a) = \mathrm{head}(w) = b$. It follows from condition (2) that $y \neq \epsilon$, hence $|w|_b = 1$, and $\mathrm{head}(y'a) \neq b$. Again it follows $ax'ay'a \in L'$ and results in a contradiction. Therefore $|w|_a = 1$. Similarly $|w|_b = 1$ if $y = \epsilon$. □

We can enumerate the MFWs of L by recursion on $|\Sigma|$. For the simplest nontrivial case, say $\Sigma = \{0, 1\}$, the MFWs can be represented by a regular expression $001^+0 + 01^+00 + 110^+1 + 10^+11$.

5. The finite automata

Technically we can construct the finite automaton that accepts L from the grammar of L' or the MFWs of L, but it is more convenient to design it directly as follows.

Input alphabet: Without losing generality, we let

$$\Sigma = \{1, 2, \ldots, m\}.$$

States:

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

where

$$P = \Sigma \cup \{0\}, \quad N = \big(\Sigma \cup \{\epsilon\}\big)^{m+1}, \quad \text{and} \quad C = \{\text{WHITE}, \text{BLACK}\}^m.$$

Initial state:

$$q_0 = \big(0, \epsilon^{m+1}, \text{WHITE}^m\big).$$

Final states:

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

Transition function: $\delta : Q \times \Sigma \to Q$ is defined by the following algorithm:

```
δ((p, n, c), a)
1 if n_p ≠ ε and n_p ≠ a
2    then i ← p
3       repeat
4          c_i ← BLACK
5          i ← n_b
6       until i = p
7 if c_a = BLACK
8       then c ← BLACK^m
9 n_p ← a, p ← a
```

Theorem 4. *The DFA $M = (Q, \Sigma, \delta, q_0, F)$ accepts L.*

Before stating the formal proof, we roughly describe the function of each component of the state variable. We use p to register the last read character, and every input string is preceded with a special character $p_0 = 0$. The vector n implements a singly linked data structure, where an element n_a gives the character following the most recent occurrence of a. This implies that a simple linear search in n can always reach p, thus the loop on lines 2–6 of the algorithm never falls infinite. The vector c attributes a "color" to every character, initially all are WHITE. If b occurs in a factor aza of the input string, with two a's followed by distinct characters, then c_b turns BLACK through the loop. Such a factor aza, with the follower different from $\mathrm{head}(za)$, will be called a *bead*. If b occurs after this bead, the string will be in L', thus a character colored BLACK will be forbidden.

Proof. We prove it by induction on the length l of the input, along with an auxiliary proposition: After reading any string $s \in L$, $c_a = \text{BLACK}$ if and only if a occurs in a bead.

The basis is evident. Suppose for any string s of length k the proposition holds. For any t of length $k+1$ we write $t = sb$.

If $s \notin L$, then $t \notin L$, and by the inductive hypothesis $c = \text{BLACK}^m$ before b is read. According to the transition function c will remain BLACK^m, thus t will be rejected by M.

If $s \in L$, then $c \neq \text{BLACK}^m$ before b is read. If the condition on line 1 holds, then t has a bead pzp. The loop body starts a walk from p, and lets $c_d = \text{BLACK}$ for every character d visited. If d in the bead is not reached, then by the rule for assignment of n on the last line of the algorithm, it must be in another bead axa with $a \neq p$, and by the inductive hypothesis $c_d = \text{BLACK}$ already. Therefore, when line 7 of the algorithm is reached, the condition holds if and only if b has occurred in a bead, i.e., c will gets BLACK^m if and only if $t \in L'$. □

6. Discussion

As pointed out in [8], the sequence reconstruction problem in consideration is equivalent to the problem of uniqueness of Eulerian trail in a directed pseudo-graph, since it can be naturally represented by a sequence over the set of vertices $V = \Sigma^{k-1}$. For a graph with an Eulerian trail $t = sb$, the state variable n represents a spanning tree towards b, with edges $\{(a, n_a)\colon\ a \in V,\ a \neq b\}$.

Under reasonable assumptions, the time complexity of the present algorithm is linear for fixed Σ and k. Since the number of $(k-1)$-tuples occurred in the sequence is usually small relative to the total number of possible ones, the state variables can be stored in a dynamic data structure to save space. For example, it can be implemented as a hash table, so that the expected running time is still linear. Furthermore, the algorithm is on-line, and can halt on the first occurrence of a forbidden word. Utilizing it, investigation on real biological sequences [10] and preliminary numerical experiments [12] have revealed some interesting features of the distribution of probability of uniqueness of sequence reconstruction with respect to k.

Acknowledgments

We would like to thank the anonymous referee for helpful comments. Q.L. is grateful to Prof. Bailin Hao for explicitly raising the addressed problem and for stimulation, and thanks Chan Zhou for indicating a reference.

References

[1] M. Chaisson, P. Pevzner, H. Tang, Fragment assembly with short reads, Bioinformatics 20 (13) (2004) 2067–2074.
[2] H. Fleischner, Eulerian Graphs and Related Topics, part 1, vol. 2, North-Holland, Amsterdam, 1990.
[3] J.E. Hopcroft, J.D. Ullman, Introduction to Automata Theory, Languages, and Computation, Addison–Wesley, Reading, MA, 1979.
[4] J.P. Hutchinson, On words with prescribed overlapping subsequences, Util. Math. 7 (1975) 241–250.
[5] D. Kandel, Y. Matias, R. Unger, P. Winkler, Shuffling biological sequences, Discrete Appl. Math. 71 (1–3) (1996) 171–185.
[6] L. Kontorovich, Uniquely decodable n-gram embeddings, Theoret. Comput. Sci. 329 (1–3) (2004) 271–284.
[7] M. Lothaire, Combinatorics on Words, reissued ed., Cambridge University Press, Cambridge, 1997.
[8] P.A. Pevzner, l-Tuple DNA sequencing: Computer analysis, J. Biomol. Struct. Dyn. 7 (1) (1989) 63–73.
[9] P.A. Pevzner, DNA physical mapping and alternating Eulerian cycles in colored graphs, Algorithmica 13 (1) (1995) 77–105.
[10] X. Shi, H. Xie, S. Zhang, B. Hao, Decomposition and reconstruction of protein sequences: The problem of uniqueness and factorizable language, J. Korean Phys. Soc. 50 (1) (2007) 118–123.
[11] E. Ukkonen, Approximate string-matching with q-grams and maximal matches, Theoret. Comput. Sci. 92 (1) (1992) 191–211.
[12] L. Xia, C. Zhou, Phase transition in sequence unique reconstruction, J. Syst. Sci. Complex. 20 (1) (2007) 18–29.
[13] H. Xie, Grammatical Complexity and One-Dimensional Dynamical Systems, World Scientific, Singapore, 1996.