ARTICLE IN PRESS YJCSS:2734

Journal of Computer and System Sciences ••• (••••) •••–•••

Contents lists available at ScienceDirect

Journal of Computer and System Sciences

www.elsevier.com/locate/jcss

Deciding unique decodability of bigram counts via finite automata

Aryeh Kontorovich[a,*], Ari Trachtenberg[b]

[a] Department of Computer Science, Ben-Gurion University, Beer Sheva 84105, Israel
[b] Department of Electrical & Computer Engineering, Boston University, 8 Saint Mary's Street, Boston, MA 02215, United States

ARTICLE INFO

Article history:
Received 28 November 2011
Received in revised form 23 August 2013
Accepted 18 September 2013
Available online xxxx

Keywords:
Uniqueness
Sequence reconstruction
Eulerian graph
Finite-state automata

ABSTRACT

We revisit the problem of deciding by means of a finite automaton whether a string is uniquely decodable from its bigram counts. An efficient algorithm for constructing a polynomial-size Nondeterministic Finite Automaton (NFA) that decides unique decodability is given. This NFA may be simulated efficiently in time and space. Conversely, we show that the minimum deterministic finite automaton for deciding unique decodability has exponentially many states in alphabet size, and compute the correct order of magnitude of the exponent.

1. Introduction

Reconstructing a string from its snippets is a problem of fundamental importance in many areas of computing. In a biological context this problem amounts to sequencing of DNA from short reads [7] and reconstruction of protein sequences from K-peptides [10]. Communications protocols [3,9] recombine snippets from related documents to identify differences between them, and fuzzy extractors [11] use similar techniques for producing keys from noise-prone biometric data. Computational linguistics also makes occasional use of this snippet representation (under the name Wickelfeatures [1]), as a means to learn transformations on varying-length sequences.

In general, there may be a large number of possible string reconstructions from a given collection of overlapping snippets; for example, the snippets {at, an, ka, na, ta} can be combined into katana or kanata. In order to keep the decoding complexity and ambiguity low, it is desirable in practice to choose a snippet length that allows only a few distinct reconstructions — the ideal number being exactly one.

Main results. We consider the problem of *efficiently* determining whether a collection of snippets has a unique reconstruction. More precisely, we construct a nondeterministic finite automaton (NFA) on $O(|\Sigma|^3)$ states that recognizes the language of strings that admit a unique reconstruction from bigram counts over the alphabet Σ. Our NFA has a particularly simple form that provides for an easy and efficient implementation, and runs on a string of length ℓ in time $O(\ell|\Sigma|^3)$ and constant memory.[1] We further show that the minimum equivalent deterministic finite automaton has at least $|\Sigma|!$ states.[2] Together

* Corresponding author. Fax: +972 8 647 7650.
E-mail addresses: karyeh@cs.bgu.ac.il (A. Kontorovich), trachten@bu.edu (A. Trachtenberg).

[1] We have since reduced the time to linear via an entirely different approach that sidesteps automata [4]. See also the algorithm sketched but not formally analyzed in [12].

[2] In a personal communication, Q. Li improved our previous bound of $2^{|\Sigma|-1}$.

0022-0000/$ – see front matter © 2013 Published by Elsevier Inc.
http://dx.doi.org/10.1016/j.jcss.2013.09.003

with the upper bound $2^{O(|\Sigma|\log|\Sigma|)}$ implicit in [12], this pegs the size of the canonical DFA at $\exp(\Theta(|\Sigma|\log|\Sigma|))$; closing this gap is an intriguing open problem.

Related work. It was shown in [8] that the collection of strings having a unique reconstruction from the snippet representation is a regular language. An explicit construction of a deterministic finite-state automaton (DFA) recognizing this language was given in by Li and Xie [12]. Unfortunately, this DFA has

$$2^{|\Sigma|}\big(|\Sigma|+1\big)\big(|\Sigma|+1\big)^{(|\Sigma|+1)} \in 2^{O(|\Sigma|\log|\Sigma|)}$$

states, and thus is not practical except for very small alphabets.[3] As we show in this paper, there is no DFA of subexponential size for recognizing this language; however, we exhibit an equivalent NFA with $O(|\Sigma|^3)$ states.

Outline. We proceed in Section 2 with some preliminary definitions and notation. In Section 3 we present our construction of an NFA recognizing uniquely decodable strings, and we prove its correctness in Section 4. Finally, we present a new lower bound on the size of a DFA accepting uniquely decodable strings in Section 5, and conclude in Section 6 with discussion and an open problem.

2. Preliminaries

We assume a finite alphabet Σ along with a special delimiter character $\$ \notin \Sigma$, and define $\Sigma_\$ = \Sigma \cup \{\$\}$. For $k \geqslant 1$, the k-gram map Φ takes string $x \in \$\Sigma^*\$$ to a vector $\xi \in \mathbb{N}^{\Sigma_\$^k}$, where $\xi_{i_1,\dots,i_k} \in \mathbb{N}$ is the number of times the string $i_1 \dots i_k \in \Sigma^k$ occurred in x as a contiguous subsequence, counting overlaps. In this paper we will focus on the *bigram* case $k = 2$, and leave a detailed study of higher k-grams for future work. As we have seen, the bigram map $\Phi : \$\Sigma^*\$ \to \mathbb{N}^{\Sigma_\$^2}$ is not injective; for example, $\Phi(\$\mathsf{katana}\$) = \Phi(\$\mathsf{kanata}\$)$.

We denote by $L_{\text{UNIQ}} \subseteq \Sigma^*$ the collection of all strings w for which

$$\Phi^{-1}\big(\Phi(\$w\$)\big) = \{\$w\$\}$$

and refer to these strings as *uniquely decodable*, meaning that there is exactly one way to reconstruct them from their bigram snippets. The examples \$katan\$ and \$katana\$ show that $\emptyset \neq L_{\text{UNIQ}} \neq \Sigma^*$ for $|\Sigma| > 1$. The induced *bigram graph* of a string $w \in \Sigma^*$ is a weighted directed graph $G = (V, E)$, with $V = \Sigma_\$$ and $E = \{e(a,b)\colon\ a, b \in \Sigma_\$\}$, where the edge weight $e(a,b) \geqslant 0$ records the number of times a occurs immediately before b in the string $\$w\$$.

We also follow the standard conventions for sets, languages, regular expressions, and automata [2,5,6]. As such, a *factor* of a string (colloquially a *snippet*) is any of its contiguous substrings. The term Σ^* denotes the free monoid over the alphabet Σ, and, for $S \subseteq \Sigma$, the term S^* has the usual regular-expression interpretation; the language defined by a regular expression $\mathbf{R}$ will be denoted $L(\mathbf{R})$. In addition, we will denote the omission of a symbol from the alphabet by $\Sigma_{\bar{x}} := \Sigma \setminus \{x\}$ for $x \in \Sigma$.

Finally, we shall use the standard five-tuple [5] notation $(\Sigma, Q, q_0, \delta, F)$ to specify a given DFA, where Σ is the input alphabet, Q is the set of states, q_0 is the initial state, δ is the transition function, and F are the final states; an analogous notation is used for NFAs. We use the notation $|\cdot|$ both to denote the size of an automaton (measured by the number of states) and the length of a string.

3. Construction and simulation of the NFA

3.1. Obstruction languages and their DFAs

Our starting point is the observation, also made in [12], that L_{UNIQ} is a *factorial* language, meaning that it is closed under taking factors. From here, Li and Xie [12] proceed to characterize L_{UNIQ} in terms of its minimal forbidden words. We shall likewise consider obstructions in the form of simple regular languages. Indeed, our Theorem 2 bears a superficial similarity to [12, Theorem 3]. However, since we deal with forbidden *languages* rather than words, we are able to obtain a much more compact description of L_{UNIQ}.

For $x \in \Sigma$ and $a, b \in \Sigma_{\bar{x}}$, define

$$I_{x,a,b} = L\big(\Sigma^* a \Sigma_{\bar{b}}^* x \Sigma_{\bar{a}}^* b \Sigma^*\big).$$

In particular, note that for every $w \in I_{x,a,b}$, its induced bigram graph has a directed path from a to x avoiding b and a directed path from x to b avoiding a — but this is not a characterization of $I_{x,a,b}$, as the example $w = xbax$ shows. Similarly, for $x \in \Sigma$ and $a, b \in \Sigma_{\bar{x}}$, define

$$J_{x,a,b} = L\big(\Sigma^* a \Sigma_{\bar{x}}^* b \Sigma^*\big).$$

[3] Li and Xie [12] apparently have an efficient algorithm for simulating their exponentially large DFA, but they do not clearly provide all the details.

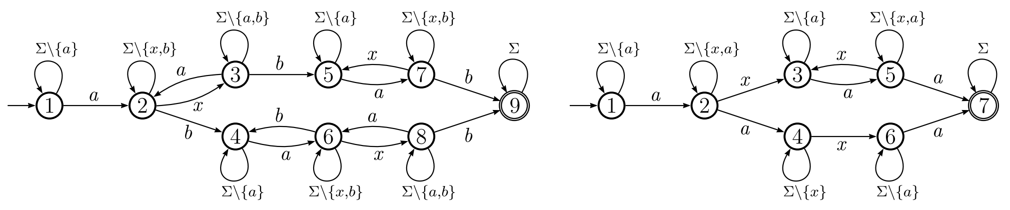

Fig. 1. The canonical DFA for $K_{x,a,b}$, for $a \neq b$ (left) and $a = b$ (right); note that this DFA never has more than 9 states, regardless of alphabet size.

Thus, $J_{x,a,b}$ the collection of all strings $w \in \Sigma^*$ containing a followed by a string not containing x and followed by b. Finally, define an *obstruction language*

$$K_{x,a,b} = I_{x,a,b} \cap J_{x,a,b},$$

whose elements will be called *obstructions*. The language of all obstructions will be denoted

$$L_{\text{OBST}} = \bigcup_{x \in \Sigma} \bigcup_{a,b \in \Sigma_{\bar{x}}} K_{x,a,b}. \tag{1}$$

The DFA recognizing a typical $K_{x,a,b}$ is illustrated in Fig. 1. One can verify that these DFAs indeed recognize $K_{x,a,b}$ straightforwardly for $\Sigma = \{a, b, x\}$, and note that the automata continue to be correct for any $\Sigma' \supseteq \{a, b, x\}$. An important feature of $K_{x,a,b}$ is that 9 states always suffice for its DFA, regardless of Σ (one can also check that the DFAs given in Fig. 1 are canonical by applying the DFA minimization algorithm [5]).

3.2. The NFA as a union of obstructions

For $x \in \Sigma$ and $a, b \in \Sigma_{\bar{x}}$, let $M_{x,a,b} = (\Sigma, Q_{x,a,b}, s_{x,a,b}, F_{x,a,b}, \delta_{x,a,b})$ be the canonical DFA recognizing the obstruction language $K_{x,a,b}$. Observe that there are

$$|\Sigma|\big(|\Sigma| - 1 + \big(|\Sigma| - 1\big)\big(|\Sigma| - 2\big)\big) \in O\big(|\Sigma|^3\big) \tag{2}$$

distinct obstruction languages. Indeed, there are $|\Sigma|$ choices for x. If $a = b$, we have $|\Sigma| - 1$ ways to choose $a \in \Sigma_{\bar{x}}$, and if $a \neq b$, we have $(|\Sigma| - 1)(|\Sigma| - 2)$ ways to choose $(a, b) \in \Sigma_{\bar{x}}^2$.

Define the NFA $M_{\text{OBST}} = (\Sigma, Q, Q_0, F, \Delta)$ as follows:

$$Q = \bigcup_{x \in \Sigma} \bigcup_{a,b \in \Sigma_{\bar{x}}} Q_{x,a,b},$$

$$Q_0 = \bigcup_{x \in \Sigma} \bigcup_{a,b \in \Sigma_{\bar{x}}} \{s_{x,a,b}\},$$

$$F = \bigcup_{x \in \Sigma} \bigcup_{a,b \in \Sigma_{\bar{x}}} F_{x,a,b},$$

$$\Delta = \bigcup_{x \in \Sigma} \bigcup_{a,b \in \Sigma_{\bar{x}}} \delta_{x,a,b}.$$

In words, M_{OBST} is the union NFA comprised of all the DFAs $M_{x,a,b}$; note that its only source of nondeterminism is that it simultaneously starts in each of the start states $s_{x,a,b}$. By design, M_{OBST} is an NFA recognizing the language L_{OBST}.

We collect these observations into a theorem.

Theorem 1. *The NFA M_{OBST}*

(i) *recognizes the language L_{OBST},*
(ii) *has*

$$|\Sigma|\big(7\big(|\Sigma| - 1\big) + 9\big(|\Sigma| - 1\big)\big(|\Sigma| - 2\big)\big) \in O\big(|\Sigma|^3\big)$$

states, and

(iii) *can be simulated on $w \in \Sigma^\ell$ in $O(\ell|\Sigma|^3)$ time and $\Theta(1)$ space.*

Proof. Item (i) follows from the discussion above. The claim in (ii) follows from the calculation in (2) and the construction in Fig. 1, which implies $|M_{x,a,a}| = 7$ and $|M_{x,a,b}| = 9$. To simulate M_{OBST} on a string w with the complexity in (iii), our simulator runs each of the DFAs $M_{x,a,b}$ on w. If any of them accepts, the simulator accepts; if none accepts, it rejects. The DFAs $M_{x,a,b}$ can be constructed in constant time and space, sequentially, by substituting the appropriate values of x, a, b in the transitions of the generic DFAs illustrated in Fig. 1. □

4. Proof of correctness

So far, we have defined two seemingly unrelated objects: L_{UNIQ}, the collection of uniquely decodable strings, and L_{OBST}, the language of obstructions. We shall now prove that the two are complementary.

Theorem 2.

$$L_{\text{UNIQ}} = \Sigma^* \setminus L_{\text{OBST}}.$$

We develop the proof with the aid of several lemmata.

4.1. $L_{\text{OBST}} \subseteq \Sigma^* \setminus L_{\text{UNIQ}}$

The forward direction has the simpler proof, deriving from one lemma.

Lemma 3. *For $x \in \Sigma$ and $a, b \in \Sigma_{\bar{x}}$, we have*

$$K_{x,a,b} \subseteq \Sigma^* \setminus L_{\text{UNIQ}}.$$

Proof. By definition, w contains a factor of the form $u = au'xu''b$, with $u' \in \Sigma_{\bar{b}}^*$, $u'' \in \Sigma_{\bar{a}}^*$, and a factor of the form $v = av'b$, with $v' \in \Sigma_{\bar{x}}^*$. Note that u and v cannot overlap, and so w must be of the form $w' = \alpha u \beta v \gamma$ or $w'' = \alpha v \beta u \gamma$ for some $\alpha, \beta, \gamma \in \Sigma^*$. Since u and v both start with a and end with b, the bigram encodings of w' and w'' will be identical, meaning that their preimage string w is not uniquely decodable. □

4.2. $L_{\text{OBST}} \supseteq \Sigma^* \setminus L_{\text{UNIQ}}$

In the reverse direction, it will be convenient to consider obstructions of a slightly specialized form. Namely, for $x \in \Sigma$ and $a, b \in \Sigma_{\bar{x}}$, define

$$\tilde{I}_{x,a,b} = L\left(\Sigma^* a x \Sigma_{\bar{a}}^* b \Sigma^*\right), \qquad \tilde{K}_{x,a,b} = \tilde{I}_{x,a,b} \cap J_{x,a,b}$$

and

$$\tilde{L}_{\text{OBST}} = \bigcup_{x \in \Sigma} \bigcup_{a,b \in \Sigma_{\bar{x}}} \tilde{K}_{x,a,b}.$$

Since obviously $L_{\text{OBST}} \supseteq \tilde{L}_{\text{OBST}}$, it will suffice to show that $\tilde{L}_{\text{OBST}} \supseteq \Sigma^* \setminus L_{\text{UNIQ}}$.[4]

Our proof draws heavily from the definitions in [8], some of which were reproduced in Section 2. For sake of exposition, we note that the weighted *inflow* and *outflow* of a node v in the bigram graph of a string[5] are given by

$$\text{inflow}(v) = \sum_{u \neq v} e(u, v), \qquad \text{outflow}(v) = \sum_{u \neq v} e(v, u).$$

The *self-flow* of v is simply $\text{self-flow}(v) = e(v, v)$. Finally, for an edge $e(v, w) > 0$, we say that v is a *parent* of w or w is a *child* of v and denote both with $v \to w$.

In addition, the pruning operator $P_x(w)$ deletes all occurrences of the letter $x \in \Sigma$ from the string $w \in \Sigma^*$. A vertex $x \neq \$$ is *removable* in a bigram graph G [8, Definition 4] if:

(a) x has a single child b,
(b) no parent of x has a child b, and
(c) if x is a child of x, then $\text{outflow}(x) = 1$.

[4] The canonical DFA for $\tilde{K}_{x,a,b}$ can have up to 10 states, and thus for the construction of M_{OBST}, the choice of $M_{x,a,b}$ is more economical.
[5] These are distinct from the weighted in-degree and out-degree in graph theory, in that they do not include the weights of self-loops.

The removal of a removable node results in a string with the same number of decodings as w [8]. Where these x correspond to a node with outflow 1 in the bigram graph of w, we call them type-I removable; otherwise, we call them type-II removable.

Our first observation is that pruning a removable node preserves modified obstructions:

Lemma 4. *Suppose that $w \in \Sigma^*$ induces the bigram graph $G(w)$ with a removable node r, and let $w' = P_r(w)$. Then $w \in \tilde{L}_{\mathrm{OBST}}$ if and only if $w' \in \tilde{L}_{\mathrm{OBST}}$.*

Proof. For the forward direction, assume $w \in \tilde{L}_{\mathrm{OBST}}$, meaning that w belongs to some $\tilde{K}_{x,a,b}$. Note that if $r \notin \{x, a, b\}$ then $w' \in \tilde{K}_{x,a,b}$, because deleting r does not change membership in either $\tilde{I}_{x,a,b}$ or $J_{x,a,b}$. Thus, we need only consider what happens when one of $r \in \{a, b, x\}$ is pruned.

We can rule out the case $r = a$ because a has two distinct children and so, by definition, is not removable. For the case $r = b$, we note that b appears at least twice in the string and thus has outflow $\geqslant 2$. For b to be removable, it must have a single child b', making w' an element of $\tilde{K}_{x,a,b'}$.

It remains to consider the case $r = x$. Recall that $w \in \tilde{K}_{x,a,b}$ and thus contains a factor $u = axu'b$, with $u' \in \Sigma^*_{\bar{a}}$. Consider the sub-case where w contains ab as a factor. Now if $u' = \varepsilon$ then x is not removable in G (its parent a points to its child b), so assume that $u' = x'u''$ for $x' \in \Sigma \setminus \{x, a, b\}$ and $u'' \in \Sigma^*_{\bar{a}}$. In this case, x might be removable in G, but then $w' \in \tilde{K}_{x',a,b}$. Alternatively, suppose $w \in \tilde{K}_{x,a,b}$ does not contain ab as a factor. It must, however, contain the factor $v = av'b$ with $v' \in \Sigma^+_{\bar{x}}$. If $u' = \varepsilon$ then w' has the factor ab and also the factor $av'b$, and thus belongs to $\tilde{K}_{y,a,b}$ for some y in v'. Otherwise, w' has the factors $au'b = au'_1u'_2 \ldots u'_kb$ and $av'b = av'_1v'_2 \ldots v'_\ell b$. We cannot have $u'_1 = v'_1$, for then w would have the factors axu'_1 and au'_1, and x would not be removable in G. If u'_1 does not occur in v', then $w' \in \tilde{K}_{u'_1,a,b}$. If u'_1 occurs in v', then $w' \in \tilde{K}_{v'_1,a,u'_1}$.

The direction $w' \in \tilde{L}_{\mathrm{OBST}} \Longrightarrow w \in \tilde{L}_{\mathrm{OBST}}$ is proved analogously. □

Finally, we show that any non-uniquely decodable string must be a modified obstruction:

Lemma 5.

$$\Sigma^* \setminus L_{\mathrm{UNIQ}} \subseteq \bigcup_{x \in \Sigma} \bigcup_{a,b \in \Sigma_{\bar{x}}} \tilde{K}_{x,a,b}.$$

Proof. Pick a $w \in \Sigma^* \setminus L_{\mathrm{UNIQ}}$. Since w is not uniquely decodable, its bigram graph G has more than one valid traversal. Let G' be the graph obtained after pruning the removable nodes from G (in some order) until no removable nodes are remaining. Then G' is a non-trivial graph [8, Theorem 9] and has the same number of decodings (valid traversals) as G [8, Theorems 5, 6]. Furthermore, Lemma 4 above implies that a decoding u of G is an obstruction iff the corresponding pruned decoding u' of G' is an obstruction.

Thus, to prove the theorem, it suffices to show that the corresponding pruned string w' is an obstruction. By construction, G' has no removable nodes, meaning that at least one of the following holds for every node $g \in G'$, $g \neq \$$:

(i) $g \to a$ and $g \to b$ for distinct $a, b \in \Sigma_{\bar{g}}$.
(ii) self-flow$(g) > 0$ and outflow$(g) > 1$.
(iii) $a \to g \to b$ and $a \to b$ for $a, b \in \Sigma_{\bar{g}}$.

If (iii) holds for *any* node g, then every decoding of G' is an obstruction of the type $\tilde{K}_{g,a,b}$.

There are two ways that (ii) can hold for *any* g: (ii′) $g \to g$ and $e(g, x) > 1$ or (ii″) $g \to g$ and $g \to x$, $g \to y$ for $x \neq y$. In case of (ii′), any decoding of G' must contain both a factor gg and also a factor gx and a directed path from x back to g. Thus, any such decoding belongs to $\tilde{K}_{x,g,g}$. Similarly, in case of (ii″), we have a directed path from x or y to g (or both), resulting in the decoding belonging to $\tilde{K}_{x,g,g}$ or $\tilde{K}_{y,g,g}$ respectively.

It remains to examine the case where every node in G' satisfies (i) — i.e., has at least two distinct children. For a string u, we will use the shorthand $\Sigma^*_{\bar{u}} \equiv \Sigma^* \setminus L(\Sigma^* u \Sigma^*)$ and (in a slight abuse of notation) will write $a \in u$ to indicate that the letter a occurs somewhere in the string u. Suppose to the contrary that $w \notin \tilde{L}_{\mathrm{OBST}}$. Let a be the first letter in w' and let $x \neq y$ be a's children, where x occurs in w' for the first time before y. Thus, $w' = axw_1ayw_2$ with $w_1 \in \Sigma^*_{\overline{ay}}$, and we have that $w' \in J_{x,a,y} \cap J_{y,a,x}$. If $y \in w_1$ then $w' \in \tilde{I}_{x,a,y}$, and therefore $w' \in \tilde{K}_{x,a,y}$; to avoid contradiction, we must have $y \notin w_1$. Similarly, if $a \in w_2$ then $w' \in \tilde{I}_{y,a,a} \cap J_{y,a,a} = \tilde{K}_{y,a,a}$, and so to avoid contradiction, we assume $a \notin w_2$. Furthermore, if $x \in w_2$ then $w' \in \tilde{I}_{y,a,x}$ and therefore $w' \in \tilde{K}_{y,a,x}$; thus we conclude that $x \notin w_2$. Since we must have $y \in w_2$, it follows that $w' = axw_1aybw_3ycw_4$, with $b, c \in \Sigma$ and $w_3 \in \Sigma^*_{\overline{yc}}$. By arguments similar to the above, we conclude that $b, y \notin w_4$, $c \notin w_1$, $c \notin w_3$, $c \in w_4$. Clearly, this process cannot continue indefinitely for a finite w'. This shows that any string in whose induced bigram graph every node has two or more children must be in $\tilde{L}_{\mathrm{OBST}}$. □

5. Lower bound for DFAs recognizing L_{UNIQ}

We know from Theorems 1 and 2 (or from earlier work [8]) that $L_{\mathrm{UNIQ}} \subset \Sigma^*$ is a regular language. Let us denote the minimum DFA recognizing L_{UNIQ} by M°_{UNIQ}. In this section we examine the size of M°_{UNIQ}, as measured by the number of states. In [12], Li and Xie constructed a DFA on

$$2^{|\Sigma|}\big(|\Sigma|+1\big)\big(|\Sigma|+1\big)^{(|\Sigma|+1)} \in 2^{O(|\Sigma|\log|\Sigma|)} \tag{3}$$

states recognizing L_{UNIQ}. Their construction is not optimal: for example, when $|\Sigma| = 3$, the left-hand size of (3) is equal to 8192 while the canonical DFA for $L_{\mathrm{UNIQ}} \subset \{a, b, c\}^*$ has 84 states.[6] However, the exponent in (3) is of the correct order, as pointed out to us by Q. Li [13] and explained in the following lemma.

Theorem 6. *A deterministic finite automaton accepting uniquely-decodable strings must have at least $|\Sigma|!$ states.*

Proof. Define $=_{\mathrm{U}}$ to be the usual equivalence relation induced on Σ^* by L_{UNIQ}: $x =_{\mathrm{U}} y$ if and only if there is no $t \in \Sigma^*$ that *distinguishes* xt from yt, meaning that $xt \in L_{\mathrm{UNIQ}}$ from $yt \notin L_{\mathrm{UNIQ}}$ or vice versa. Then the Myhill–Nerode theorem [5] assures us that the number of states in a DFA accepting L_{UNIQ} is at least the number of strings that are pairwise-distinguishable with respect to L_{UNIQ}.

Let $\sigma = \sigma_1 \dots \sigma_n$ and $\tau = \tau_1 \dots \tau_n$ be distinct permutations of Σ. We consider two cases: $\sigma_n = \tau_n$ and $\sigma_n \neq \tau_n$. The second case is simpler: the string $t = \sigma_n\sigma_n$ distinguishes σ from τ since $\sigma t \in L_{\mathrm{UNIQ}}$ while $\tau t \notin L_{\mathrm{UNIQ}}$.

For the first case, consider the largest $1 \leqslant i < n$ such that $\sigma_i \neq \tau_i$; such an i must exist since we assumed σ and τ to be distinct. Thus, σ and τ share a common suffix on indices $i+1$ through n and have distinct prefixes on indices 1 through i. We claim that $t = \sigma_i\sigma_{i+1}$ is a distinguishing string. Indeed, it is easily seen that $\sigma t \in L_{\mathrm{UNIQ}}$. Since our choice of i insures that (1) σ_i occurs somewhere in $\tau_1 \dots \tau_{i-1}$ and (2) $\sigma_{i+1} = \tau_{i+1}$, it follows that $\tau t \notin L_{\mathrm{UNIQ}}$. □

6. Discussion

We have provided a novel, constructive proof that L_{UNIQ} is a regular language, that yields as a by-product an $O(|\Sigma|^3)$-sized NFA recognizing L_{UNIQ} that can be efficiently simulated. We have also shown that the minimum DFA has $\exp(f(|\Sigma|))$ states, where

$$f(n) \in \Theta(n \log n).$$

The exact growth rate of $f(n)$ is an intriguing open problem. An even more important question is the automaton complexity of deciding unique decodability for higher k-grams. In [8] it was proved that the language of strings uniquely decodable from k-gram counts is regular, but no efficient construction was given. How does the complexity of the NFA/DFA for this language grow with k?

Acknowledgments

We thank the referees for carefully reading the manuscript and making useful suggestions. Arnold Filtzer and Omrit Naftali assisted in proving Lemma 5. The first author was supported in part by the Israel Science Foundation (grant No. 1141/12).

References

[1] D.E. Rumelhart, J.L. McClelland, On learning past tenses of English verbs, in: D.E. Rumelhart, J.L. McClelland (Eds.), Parallel Distributed Processing: Vol. 2: Psychological and Biological Models, MIT press, 1986, pp. 216–271.
[2] Michael Sipser, Introduction to the Theory of Computation, 1st edition, International Thomson Publishing, 1996.
[3] Andrei Z. Broder, On the resemblance and containment of documents, in: Compression and Complexity of Sequences (SEQUENCES '97), 1997, pp. 21–29.
[4] Arnold Filtser, Jiaxi Jin, Aryeh Kontorovich, Ari Trachtenberg, Efficient determination of the unique decodability of a string, in: IEEE International Symposium on Information Theory (ISIT), 2013.
[5] Dexter C. Kozen, Automata and Computability, Springer-Verlag, New York, Inc., Secaucus, NJ, USA, 1997.
[6] Harry R. Lewis, Christos H. Papadimitriou, Elements of the Theory of Computation, Prentice Hall PTR, Upper Saddle River, NJ, USA, 1997.
[7] Mark Chaisson, Pavel A. Pevzner, Haixu Tang, Fragment assembly with short reads, Bioinformatics 20 (13) (2004) 2067–2074.
[8] Leonid Kontorovich, Uniquely decodable n-gram embeddings, Theor. Comput. Sci. 329 (1–3) (2004) 271–284.
[9] Sachin Agarwal, Vikas Chauhan, Ari Trachtenberg, Bandwidth efficient string reconciliation using puzzles, IEEE Trans. Parallel Distrib. Syst. 17 (11) (2006) 1217–1225.

[6] This may be verified by determinizing, negating, and then minimizing the NFA M_{OBST} constructed in Section 3 or by minimizing the DFA of Li and Xie [12].

[10] Xiaoli Shi, Huimin Xie, Shuyu Zhang, Bailin Hao, Decomposition and reconstruction of protein sequences: The problem of uniqueness and factorizable language, J. Korean Phys. Soc. 50 (1) (2007) 118–123.

[11] Yevgeniy Dodis, Rafail Ostrovsky, Leonid Reyzin, Adam Smith, Fuzzy extractors: how to generate strong keys from biometrics and other noisy data, SIAM J. Comput. 38 (1) (2008) 97–139.

[12] Qiang Li, Huimin Xie, Finite automata for testing composition-based reconstructibility of sequences, J. Comput. Syst. Sci. 74 (5) (2008) 870–874.

[13] Qiang Li, personal communication, 2012.