Unique decodability of bigram counts by finite automata

Aryeh (Leonid) Kontorovich[a], Ari Trachtenberg[b]

[a] Department of Computer Science, Ben-Gurion University, Beer Sheva, Israel 84105 [corresponding author; fax: +972 8 647 7650]

[b] Department of Electrical & Computer Engineering, Boston University, 8 Saint Mary's Street, Boston, MA 02215

Abstract

We revisit the problem of deciding whether a given string is uniquely decodable from its bigram counts by means of a finite automaton. An efficient algorithm for constructing a polynomial-size nondeterministic finite automaton that decides unique decodability is given. Conversely, we show that the minimum deterministic finite automaton for deciding unique decodability has at least exponentially many states in alphabet size.

Keywords: uniqueness, sequence reconstruction, Eulerian graph, finite-state automata

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 [6] and reconstruction of protein sequences from K-peptides [9]. Communications protocols [3, 8] recombine snippets from related documents to identify differences between them, and fuzzy extractors [10] 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`,

Email addresses: `karyeh@cs.bgu.ac.il` (Aryeh (Leonid) Kontorovich), `trachten@bu.edu` (Ari Trachtenberg)

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 precisely those strings over the alphabet Σ that have a unique reconstruction. 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. We further show that the minimum equivalent deterministic finite automaton has at least $2^{|\Sigma|-1}$ states. This lower bound is still far off from the upper bound $2^{O(|\Sigma|\log|\Sigma|)}$ implicit in [11] and closing this gap is an intriguing open problem.

Related work. It was shown in [7] 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 Lia and Xie [11]. Unfortunately, this DFA has

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

states, and thus is not practical except for very small alphabets. 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 $k \geq 1$, the k-gram map Φ takes string

$x \in \$\Sigma^*\$$ to a vector $\xi \in \mathbb{N}^{\Sigma_\$^k}$, where $\xi_{i_1,\ldots,i_k} \in \mathbb{N}$ is the number of times the string $i_1 \ldots i_k \in \Sigma^k$ occurred in x as a contiguous subsequence, counting overlaps.[1] 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

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

and refer to these strings as *uniquely decodable*, meaning that there is exactly one way to reconstruct them from their bigram snippets. The examples $\$\mathsf{katan}\$$ and $\$\mathsf{katana}\$$ 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 $E = \{e(a, b) : a, b \in \Sigma_\$\}$, where the edge weight $e(a, b) \geq 0$ 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, 4, 5]. 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 [4] 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 [11], that L_{UNIQ} is a *factorial* language, meaning that it is closed under taking factors. From here, Lia and Xie [11] proceed to characterize L_{UNIQ} in terms of its minimal

[1] In this paper we will focus on the *bigram* case when $k = 2$, although the general case $k > 2$ readily follows [7, 11].

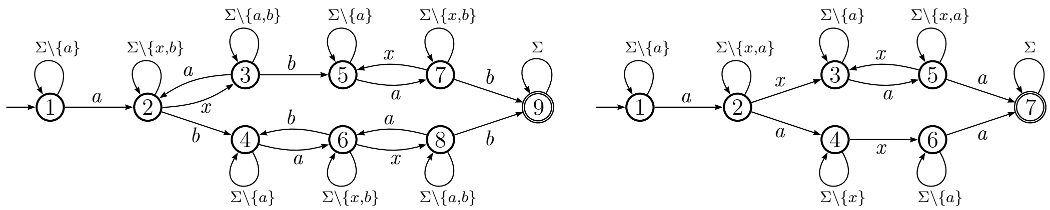

Figure 1: The canonical DFA for $K_{x,a,b}$, for $a \neq b$ (left) and $a = b$ (right); note that this DFA never has more than 9 states, regardless of alphabet size.

forbidden words. Rather than looking at forbidden words, we will consider obstructions in the form of simple regular languages.

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

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

Thus, $I_{x,a,b}$ is the collection of all strings $w \in \Sigma^*$ whose induced bigram graph has an edge from a to x and a directed path from x to b avoiding a. Similarly, for $x \in \Sigma$ and $a, b \in \Sigma_{\bar{x}}$, define

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

Thus, $J_{x,a,b}$ the collection of all strings $w \in \Sigma^*$ whose induced bigram graph has a directed path from a to b avoiding x. 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 Figure 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 Figure 1 are canonical by applying the DFA minimization algorithm [4]).

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

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

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:

$$\begin{aligned} Q &= \bigcup_{x \in \Sigma} \bigcup_{a,b \in \Sigma_{\bar{x}}} Q_{x,a,b} \\ Q_0 &= \bigcup_{x \in \Sigma} \bigcup_{a,b \in \Sigma_{\bar{x}}} \{s_{x,a,b}\} \\ F &= \bigcup_{x \in \Sigma} \bigcup_{a,b \in \Sigma_{\bar{x}}} F_{x,a,b} \\ \Delta &= \bigcup_{x \in \Sigma} \bigcup_{a,b \in \Sigma_{\bar{x}}} \delta_{x,a,b}. \end{aligned}$$

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

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

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 Figure 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 accept, the simulator accepts; if none accept, it reject. 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 Figure 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 $u = axu'b$, with $u' \in \Sigma_{\bar{a}}^*$, 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. $L_{\text{OBST}} \supseteq \Sigma^* \setminus L_{\text{UNIQ}}$

The proof of the reverse direction draws heavily from the definitions in [7], 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[2] are given by

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

[2]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 *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 [7, 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 $\text{outflow}(x) = 1$.

The removal of a removable node results in a string with the same number of decodings as w [7]. 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 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 $w \in L_{\text{OBST}}$ if and only if $w' \in L_{\text{OBST}}$.*

Proof. For the forward direction, assume $w \in L_{\text{OBST}}$, meaning that w belongs to some $K_{x,a,b}$. Note that if $r \notin \{x, a, b\}$ then $w' \in K_{x,a,b}$, because deleting r does not change membership in either $I_{x,a,b}$ 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 ≥ 2. For b to be removable, it must have a single child b', making w' an element of $K_{x,a,b'}$.

It remains to consider the case $r = x$. Recall that $w \in K_{x,a,b}$ 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 $w' \in K_{x',a,b}$. Alternatively, suppose $w \in K_{x,a,b}$ 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 $K_{y,a,b}$ 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 $w' \in K_{u_1',a,b}$. If u_1' occurs in v', then $w' \in K_{v_1',a,u_1'}$.

The direction $w' \in L_{\text{OBST}} \implies w \in L_{\text{OBST}}$ is proved analogously. $\square$

Before stating the next lemma, we introduce another bit of notation. For two nodes a, b (not necessarily distinct) in a given bigram graph, the existence of a directed path from a to b will be denoted by $a \Rightarrow b$. If in addition there is a directed path from a to b avoiding x, we indicate this by $a \stackrel{\bar{x}}{\Rightarrow} b$. These relations may be concatenated with the obvious semantics. Thus, $a \to b \stackrel{\bar{x}}{\Rightarrow} c \Rightarrow d$ implies the existence of a directed path in G that takes the edge $a \to b$, then reaches c having avoided x between b and c, and then reaches d.

Lemma 5. *Suppose the bigram graph G has a node g with distinct children $x, y \in \Sigma_{\bar{g}}$ such that $x \Rightarrow g$ and $y \Rightarrow g$. Then every traversal of G belongs to $K_{x,g,g} \cup K_{y,g,g} \cup K_{x,y,y} \cup K_{y,x,x}$.*

Proof. Our assumptions on G imply $g \to x \Rightarrow g$ and $g \to y \Rightarrow g$. We claim that least one of $x \stackrel{\bar{y}}{\Rightarrow} g$, $y \stackrel{\bar{x}}{\Rightarrow} g$ must hold. Indeed, suppose that every directed path from x to g passes through y — then there is a directed path from y to g avoiding x. Consider the case that $x \stackrel{\bar{y}}{\Rightarrow} g$. In this case, we also have that G also satisfies at least one of (i) $g \to x \stackrel{\bar{y}}{\Rightarrow} g$, (ii) $g \to x \Rightarrow y \Rightarrow x \stackrel{\bar{y}}{\Rightarrow} g$. Case (i) corresponds to traversals belonging to $K_{y,g,g}$ and (ii) corresponds to traversals belonging to $K_{y,x,x}$. A similar analysis of the case $y \stackrel{\bar{x}}{\Rightarrow} g$ proves the claim. $\square$

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

Lemma 6.

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

Proof. Pick a $w \in \Sigma^* \setminus L_{\text{UNIQ}}$. 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 [7, Theorem 9] and has the same number of decodings (valid traversals) as G [7, 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 every decoding of G' 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 $K_{g,a,b}$.

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 $K_{x,g,g}$. Similarly, in case of (ii$''$), we have $x \Rightarrow g$ or $y \Rightarrow g$, resulting in the decoding belonging to $K_{x,g,g}$ or $K_{y,g,g}$ respectively.

It remains to examine the case where every node g satisfies (i). Suppose for now that in addition to $g \to x$ and $g \to y$ for $x \neq y \in \Sigma_{\bar{g}}$ we also have $g \to z$ for some $z \in \Sigma \setminus \{g, x, y\}$. In any decoding of G', at least two of $\{x, y, z\}$ must have a directed path back to g. Lemma 5 then implies that every decoding of G' belongs to

$$\bigcup_{t \neq t' \in \{x,y,z\}} K_{t,g,g} \cup K_{t',g,g} \cup K_{t,t',t'} \cup K_{t',t,t}.$$

Having dispensed with the three-child case and with (ii) and (iii) above, the only remaining scenario is that every $g \neq \$$ in G' has exactly 2 children and self-flow$(g) = 0$. We claim that in this case, there must be a $g \in G'$ with children $x \neq y$ such that $x \Rightarrow g$ and $y \Rightarrow g$. If this were not the case, G' would be uniquely decodable — since at each node g, we would be obligated to first take the unique child that does have a directed path back to g. But this contradicts Lemma 8 in [7], which states that a bigram graph where every node other than \$ has exactly 2 children and no self-flow has multiple decodings. Let g be the requisite node with children $x \neq y$; by Lemma 5 we have that every decoding of G' belongs to $K_{x,g,g} \cup K_{y,g,g} \cup K_{x,y,y} \cup K_{y,x,x}$. $\square$

Theorem 2 follows immediately from Lemmas 3 and 6 — in light of which, the runtime complexity in Theorem 1(iii) can be improved from $O(\ell|\Sigma|^3)$ to $O(\tilde{\ell}|\Sigma|^3)$, where $\tilde{\ell}$ is the length of the shortest prefix $u \notin L_{\text{UNIQ}}$ of w.

5. Lower bound for DFAs recognizing L_{UNIQ}

We know from Theorems 1 and 2 that $L_{\text{UNIQ}} \subset \Sigma^*$ 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 [11], Lia and Xie constructed a DFA on

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

states recognizing L_{UNIQ}. However, 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 $L_{\text{UNIQ}} \subset \{a,b,c\}^*$ has 84 states.[3] The main result of this section is the following lower bound, which is also not tight as it gives a value of 4 states for this alphabet size.

Theorem 7. *For $|\Sigma| \geq 1$,*

$$|M^\circ_{\text{UNIQ}}| \geq 2^{|\Sigma|-1}.$$

Proof. Define $=_{\text{U}}$ to be the usual equivalence relation induced on Σ^* by L_{UNIQ}: $x =_{\text{U}} y$ if and only if there is no $t \in \Sigma^*$ that *distinguishes* x from y, meaning that $xt \in L_{\text{UNIQ}}$ from $yt \notin L_{\text{UNIQ}}$ or vice versa. Then the Myhill-Nerode theorem [4] 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}.

Our proof proceeds by induction on the alphabet size, where we construct a set D_i of 2^i pairwise-distinguishable strings over the alphabet $\Sigma_i = \{\langle j\rangle : 0 \leq j \leq i\}$, $i = 0, 1, 2, \ldots$. For the base case $i = 0$, we take $D_0 = \{0\}$.

Now suppose, as an inductive hypothesis, that we have constructed the set D_i of 2^i distinguished strings over the alphabet Σ_i, for $i \geq 0$. We then define D_{i+1} over the alphabet Σ_{i+1} as the union $D_{i+1} = D_i \cup D'_i$, where D'_i simply appends the letter $\langle i+1\rangle \in \Sigma_{i+1}$ to each string in D_i; more precisely, $D'_i = \{w \cdot \langle i+1\rangle : w \in D_i\}$. Thus, for example, $D_2 = \{0, 01, 02, 012\}$ and $D'_2 = \{03, 013, 023, 0123\}$ combine to form D_3. Note that the letters always appear in $w \in D_i$ in strictly increasing order, and thus $D_i \subset L_{\text{UNIQ}}$ for all $i \geq 0$.

[3] This may be verified by determinizing, negating, and then minimizing the NFA M_{OBST} constructed in Section 3 or by minimizing the DFA of Lia and Xie [11].

What remains to prove is that the members of D_{i+1} as constructed above are all pairwise distinguishable under $=_{\mathrm{U}}$. In proving that $u \neq_{\mathrm{U}} v$ for all distinct $u, v \in D_{i+1}$, we consider three cases: (i) both strings belong to D_i, (ii) both strings belong to D'_i, and (iii) one string belongs to D_i and the other to D'_i. For $u, v \in D_i$, our inductive hypothesis applies to give $u \neq_{\mathrm{U}} v$. Consider $u, v \in D'_i$. Since the sequences u and v are strictly increasing and distinct, there is necessarily a letter x that appears in one and not the other. Then u and v are distinguished by xx. To see this, suppose, without loss of generality, that x appears in u but not in v, and note that last letter of u and v is $\langle i+1 \rangle \neq x$; then $vxx \in L_{\mathrm{UNIQ}}$ and $uxx \notin L_{\mathrm{UNIQ}}$.

Finally, consider the case of $u = u_1 u_2 \ldots u_k \in D_i$ and $v = v_1 v_2 \ldots v_\ell \in D'_i$. We examine two sub-cases. First, suppose the strings $u_1 u_2 \ldots u_{k-1}$ and $v_1 v_2 \ldots v_{\ell-1}$ are distinct. Let x be a letter that appears in one and not the other. Then u and v are distinguished by xx using the argument above. In the other sub-case, we have $u_1 u_2 \ldots u_{k-1} = v_1 v_2 \ldots v_{\ell-1} = w$. Then u and v are distinguished by $t = w u_k w$. Indeed, $ut = w u_k w u_k w \in L_{\mathrm{UNIQ}}$, while $\Phi(\$vt\$)$ can be decoded as $v' = w v_\ell w u_k w$ or as $v'' = w u_k w v_\ell w$.

□

6. Discussion

We have provided a novel, constructive proof that L_{UNIQ} is a regular language, which yields as a by-product a $O(|\Sigma|^3)$-sized NFA recognizing L_{UNIQ} that can be efficiently simulated. We have also shown that the minimum DFA has $2^{f(|\Sigma|)}$ states, where

$$n - 1 \leq f(n) \leq Cn \log n$$

for some universal constant C. The exact growth rate of $f(n)$ is an intriguing open problem.

References

[1] D. E. Rumelhart and J. L. McClelland. On learning past tenses of english verbs. In D. E. Rumelhart and J. L. McClelland, editors, *Parallel Distributed Processing: Vol 2: Psychological and Biological Models*, pages 216–271. MIT press, 1986.

[2] Michael Sipser. *Introduction to the Theory of Computation*. International Thomson Publishing, 1st edition, 1996.

[3] Andrei Z. Broder. On the Resemblance and Containment of Documents, *In Compression and Complexity of Sequences (SEQUENCES '97)*, pp 21-29, 1997.

[4] Dexter C. Kozen. *Automata and Computability.* Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1997.

[5] Harry R. Lewis and Christos H. Papadimitriou. *Elements of the Theory of Computation.* Prentice Hall PTR, Upper Saddle River, NJ, USA, 1997.

[6] Mark Chaisson, Pavel A. Pevzner, and Haixu Tang. Fragment assembly with short reads. *Bioinformatics*, 20(13):2067–2074, 2004.

[7] Leonid Kontorovich. Uniquely decodable n-gram embeddings. *Theor. Comput. Sci.*, 329(1-3):271–284, 2004.

[8] Sachin Agarwal, Vikas Chauhan, and Ari Trachtenberg. Bandwidth efficient string reconciliation using puzzles. *IEEE Trans. Parallel Distrib. Syst.*, 17(11):1217–1225, 2006.

[9] Xiaoli Shi, Huimin Xie, Shuyu Zhang, and Bailin Hao. Decomposition and reconstruction of protein sequences: The problem of uniqueness and factorizable language. *Journal of the Korean Physical Society*, 50(1I):118–123, 2007.

[10] Yevgeniy Dodis, Rafail Ostrovsky, Leonid Reyzin and Adam Smith. Fuzzy Extractors: How to Generate Strong Keys from Biometrics and Other Noisy Data, *SIAM J. Comput.*, 38(1):97-139, 2008.

[11] Qiang Lia and Huimin Xie. Finite automata for testing composition-based reconstructibility of sequences. *Journal of Computer and System Sciences*, 74(5):870–874, 2008.