International Journal of Foundations of Computer Science

Finitely Subsequential Transducers

Cyril Allauzen and Mehryar Mohri
{allauzen,mohri}@research.att.com
AT&T Labs – Research
180 Park Avenue
Florham Park, NJ 07932, USA

ABSTRACT

Finitely subsequential transducers are efficient finite-state transducers with a finite number of final outputs and are used in a variety of applications. Not all transducers admit equivalent finitely subsequential transducers however. We briefly describe an existing generalized determinization algorithm for finitely subsequential transducers and give the first characterization of *finitely subsequentiable transducers*, transducers that admit equivalent finitely subsequential transducers. Our characterization shows the existence of an efficient algorithm for testing finite subsequentiability. We have fully implemented the generalized determinization algorithm and the algorithm for testing finite subsequentiability. We report experimental results showing that these algorithms are practical in large-vocabulary speech recognition applications. The theoretical formulation of our results is the equivalence of the following three properties for finite-state transducers: determinizability in the sense of the generalized algorithm, finite subsequentiability, and the twins property.

1. Introduction

Finite-state transducers are automata in which transitions are labeled with both an input and an output symbol. Transducers have been used successfully to create complex systems in many applications such as text and language processing, speech recognition, image processing, and computer arithmetic [12, 11, 9, 18, 7, 8]. In most of these applications, each component of the system is represented by a finite-state transducer. These are then combined either offline or on-the-fly using a classical algorithm of composition of finite-state transducers [16].

The time efficiency of such systems is substantially increased when *subsequential transducers* [20], i.e. finite-state transducers with deterministic input, are used. Indeed, a subsequential transducer admits at most one path labeled with any given string x. When it exists, that path can be found in linear time $O(|x|)$, independently of the size of the transducer, provided that the appropriate data structure is used for the representation of the transducer.

Subsequential transducers can be generalized to *finitely subsequential transducers* which are deterministic transducers augmented with a finite number of final output strings [13]. This generalization is necessary in many applications such as language processing to account for finite ambiguities [14]. Another advantage of the use of finitely subsequential transducers is that a general minimization algorithm is available for these machines [15] which can help reduce their size.

There exists a general determinization algorithm that takes as input a non-deterministic transducer and outputs a finitely subsequential transducer [13]. That algorithm does not apply to all transducers. In fact, not all transducers admit equivalent finitely subsequential transducers.

We present the first characterization of *finitely subsequentiable transducers*, i.e. transducers that are equivalent to finitely subsequential transducers. Our characterization is based on *the twins property* introduced by [5, 6] and leads to an efficient algorithm for testing finite subsequentiability. More generally, our results show the equivalence of the following three fundamental properties for finite-state transducers: determinizability in the sense of a generalized algorithm, finite subsequentiability, and the twins property.

This can be viewed as a generalization of the following result known in the case of functional transducers: determinizable functional transducers are exactly those that admit equivalent subsequential transducers [6]. Indeed, we generalize these results by relaxing the condition on functionality: determinizable transducers are exactly those that admit equivalent finitely subsequential transducers and exactly those that admit the twins property.

We have fully implemented both the generalized determinization algorithm mentioned above and the algorithm for testing finite subsequentiability. We report experimental results showing that these algorithms are practical in large-vocabulary speech recognition applications where transducers of with several hundred million transitions are used.

We first introduce the notation used in the rest of this paper, then briefly describe a generalized determinization algorithm for finite subsequential transducers introduced by [13], present a fundamental characterization theorem, and describe our experimental results.

2. Preliminaries

Definition 1 *A* finite-state transducer $T = (\Sigma, \Delta, Q, I, F, E, \lambda, \rho)$ *is an 8-tuple where Σ is a finite input alphabet, Δ a finite output alphabet, Q a finite set of states, $I \subseteq Q$ the set of initial states, $F \subseteq Q$ the set of final states, $E \subseteq Q \times \Sigma \times (\Delta \cup \{\epsilon\}) \times Q$ a finite set of transitions, $\lambda : I \to \Delta^*$ the initial output function mapping I to Δ^*, and $\rho : F \to 2^{\Delta^*}$ the final output function mapping each state $q \in F$ to a finite subset of Δ^*.*

Given a transition $e \in E$, we denote by $i[e]$ its input label, $p[e]$ its origin or previous state and $n[e]$ its destination state or next state, $o[e]$ its output label. Given a state $q \in Q$, we denote by $E[q]$ the set of transitions leaving q. We extend the definitions of i, n, p, and E to sets in the following way: $i[\cup_{k \in K} e_k] = \cup_{k \in K} i[e_k]$ and similarly for n, p, and E.

A *path* $\pi = e_1 \cdots e_k$ in T is an element of E^* with consecutive transitions: $n[e_{i-1}] = p[e_i]$, $i = 2, \ldots, k$. We extend n and p to paths by setting: $n[\pi] = n[e_k]$

and $p[\pi] = p[e_1]$. We denote by $P(q, q')$ the set of paths from q to q' and by $P(q, x, q')$ the set of paths from q to q' with input label $x \in \Sigma^*$. These definitions can be extended to subsets $R, R' \subseteq Q$, by: $P(R, x, R') = \bigcup_{q \in R,\, q' \in R'} P(q, x, q')$. The labeling functions i and o can also be extended to paths by defining the label of a path as the concatenation of the labels of its constituent transitions:

$$i[\pi] = i[e_1] \cdots i[e_k], \quad o[\pi] = o[e_1] \cdots o[e_k]$$

The set of output strings associated by a transducer T to an input string $x \in \Sigma^*$ is defined by:

$$[\![T]\!](x) = \bigcup_{\pi \in P(I,x,F)} \lambda(p[\pi])\, o[\pi]\, \rho(n[\pi])$$

$[\![T]\!](x) = \emptyset$ when $P(I, x, F) = \emptyset$. The domain of definition of T is defined as: $Dom(T) = \{x \in \Sigma^* : [\![T]\!](x) \neq \emptyset\}$. Two transducers T and T' are *equivalent* when $[\![T]\!] = [\![T']\!]$. Let p be an integer. A transducer is said to be *p-functional* if it associates at most p strings to each input string, that is if $|[\![T]\!](x)| \leq p$ for any $x \in \Sigma^*$. It is said to be *finitely functional* if it is p-functional for some integer p. See [4, 10] for results about p-functional and finitely functional transducers.

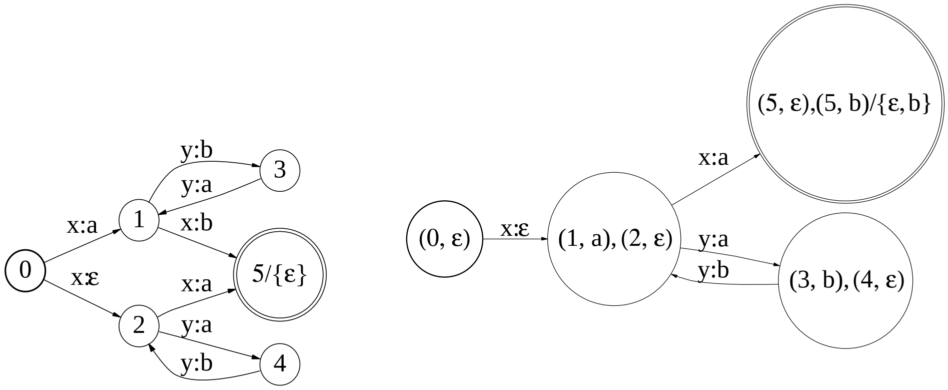
A *successful path* in a transducer T is a path from an initial state to a final state. A state $q \in Q$ is *accessible* if q can be reached from I. It is *coaccessible* if a final state can be reached from q. T is *trim* if all the states of T are both accessible and coaccessible. T is *unambiguous* if for any string $x \in \Sigma^*$ there is at most one successful path labeled with x. An unambiguous transducer is thus p-functional, with $p = \max_{q \in F} |\rho(q)|$.

A transducer T is said to be *p-subsequential* [13] if it has a unique initial state, if no two transitions leaving the same state share the same input label and if there are at most p final output strings at each final state: $|\rho(f)| \leq p$ for all $f \in F$. T is said to be *finitely subsequential* if it is p-subsequential for some integer p. T is said to be *finitely subsequentiable* if there exists a finitely subsequential transducer T' equivalent to T.

Given two strings x and y in Σ^*, we say that y is a *suffix* of x if there exists $z \in \Sigma^*$ such that $x = zy$ and similarly that y is a *prefix* of x if there exists z such that $x = yz$. We denote by $x \wedge y$ the longest common prefix of x and y and denote by $|x|$ the length of a string $x \in \Sigma^*$. We extend Σ by associating to each symbol $a \in \Sigma$ a new symbol denoted by a^{-1} and define Σ^{-1} as: $\Sigma^{-1} = \{a^{-1} : a \in \Sigma\}$. $X = (\Sigma \cup \Sigma^{-1})^*$ is then the set of strings written over the alphabet $(\Sigma \cup \Sigma^{-1})$. If we assume that $aa^{-1} = a^{-1}a = \epsilon$, then X forms a group called the *free group generated by* Σ and is denoted by $\Sigma^{(*)}$. Note that the inverse of a string $x = a_1 \cdots a_n$ is then $x^{-1} = a_n^{-1} \cdots a_1^{-1}$. The formula used in our definitions, theorems and proofs should be interpreted as equations in the free group generated by Σ^*.

3. General determinization algorithm with finitely subsequential outputs

In this section, we give a brief description of a general determinization algorithm introduced by [13] that takes as input a transducer T and outputs a finitely subsequential transducer $T' = (\Sigma, \Delta, Q', \{i'\}, F', E', \lambda', \rho')$. A transducer T for which



(a) (b)

Figure 1: Generalized determinization of finite-state transducers. (a) Non-deterministic transducer. (b) Construction of equivalent 2-subsequential transducer.

the algorithm terminates and thus generates an equivalent finitely subsequential transducer is said to be *determinizable*.

The algorithm is a generalization of the subset construction used in the determinization of finite automata. A state in the output transducer T' is a set of pairs (q, z) where q is a state of the input transducer T and $z \in \Sigma^*$ a remainder output string with the following property: if a state q' in T' containing a pair (q, z) can be reached from the initial state by a path with input x and output y, then q can be reached in T from an initial state by a path with input x and output yz. The following is the pseudocode of the algorithm.

```
Transducer-Determinization(T)
F′ ← Q′ ← E′ ← ∅
S ← i′ ← {(i, λ(i)) : i ∈ I}
while S ≠ ∅
      do p′ ← head(S)
         Dequeue(S)
         for each x ∈ i[E[Q[p′]]]
             do y′ ← ⋀ {z y : (p, z) ∈ p′, (p, x, y, q) ∈ E}
                q′ ← {(q, y′⁻¹ z y) : (p, z) ∈ p′, (p, x, y, q) ∈ E}
                E′ ← E′ ∪ {(p′, x, y′, q′)}
                if (q′ ∉ Q′)
                   then Q′ ← Q′ ∪ {q′}
                        if Q[q′] ∩ F ≠ ∅
                           then F′ ← F′ ∪ {q′}
                                ρ′(q′) ← ⋃ {z ρ(q) : (q, z) ∈ q′, q ∈ F}
                        Enqueue(S, q′)
return T′
```

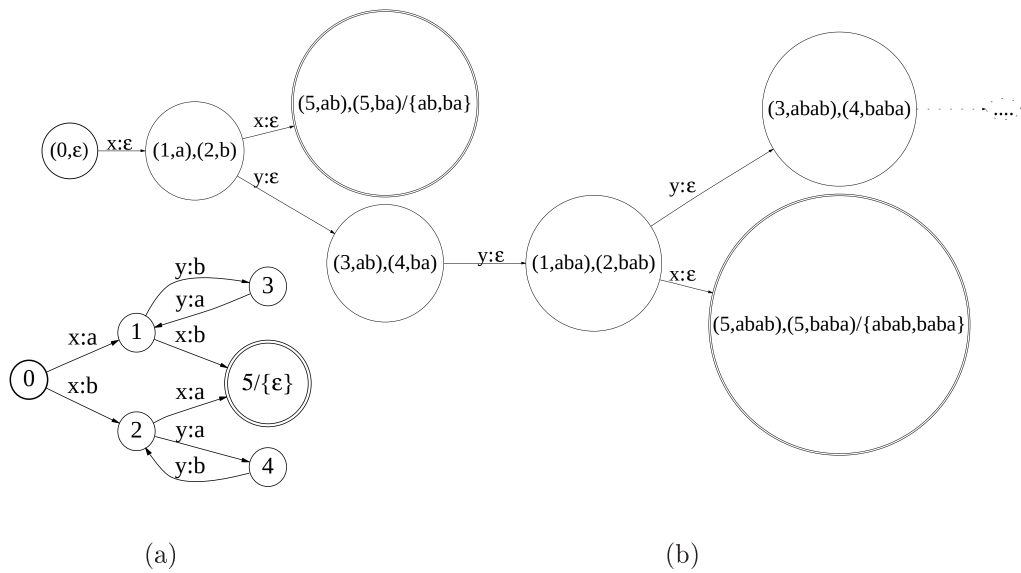


Figure 2: Non-determinizable case. (a) A non-determinizable finite-state transducer; states 1 and 2 are non-twin siblings. (b) Determinization does not halt in this case and creates an infinite number of states.

Line 1 initializes the set of states, final states, and transitions of T' to the empty set. The algorithm uses a queue S containing the set of states of T' to be considered next. S initially contains the unique initial state of T', i', which is the set of pairs of an initial state i of T and the corresponding initial output string $\lambda(i)$ (line 2).

Each time through the loop of lines 3-15, a new subset p' (or equivalently a new state of T') is extracted from S. The algorithm then creates (lines 6-9) a transition with input label $x \in \Sigma$ and output label $y' \in \Sigma^*$ leaving p' if there exists at least one pair $(p, z) \in p'$ such that p admits an outgoing transition with input label x and output label y. y' is then defined as the longest common prefix of all such zy's. The destination state q' of that transition is the subset containing the pairs $(q, y'^{-1}zy)$ such that $(p, z) \in p'$ and (p, x, y, q) is a transition in E. If the destination state q' is new, it is added to Q' (lines 10-11). q' is a final state if it contains at least one pair (q, z), q being a final state. Its final set of output strings is then the union of $z\rho(q)$ over all such pairs (q, z).

There are input transducers that are not determinizable, that is for which the algorithm does not terminate. When it terminates, the output transducer T' is equivalent to T. Thus, it does not terminate with any transducer T that is not finitely subsequentiable.

Figure 1(b) illustrates the application of the algorithm to the transducer of figure 1(a). Figures 2(a)-(b) show an example of non-determinizable transducer.

The worst case complexity of determinization is exponential. However, in many applications such as large-vocabulary speech recognition such a blow-up does not occur and determinization leads to a significant improvement of speed versus accuracy at a reasonable cost in space [18].

4. Characterization

This section presents a characterization of finitely subsequentiable transducers. The characterization is based on the following property.

Definition 2 *Let T be a finite-state transducer. Two states q_1 and q_2 of T are said to be* siblings *if there exist two strings x and y in Σ^* such that both q_1 and q_2 can be reached from I by paths with input label x and there are cycles at q_1 and q_2 both with input label y. Two siblings q_1 and q_2 are said to be* twins *if for any paths $\pi_1 \in P(I, x, q_1)$, $c_1 \in P(q_1, y, q_1)$, $\pi_2 \in P(I, x, q_2)$, $c_2 \in P(q_2, y, q_2)$,*

$$o[\pi_1]^{-1} o[\pi_2] = o[\pi_1 c_1]^{-1} o[\pi_2 c_2] \tag{1}$$

T has the twins property *if any two siblings in T are twins.*

The *twins property* was originally introduced by [5, 6] to give a characterization of functional subsequentiable transducers. Its decidability was proved by the same author, see also [3]. The first polynomial-time algorithm for testing the twins property was given by [21], this algorithm was later improved by [2]. More recently, we gave a more efficient algorithm for testing the twins property based on the general algorithm of composition of finite-state transducers and a new characterization of the twins property in terms of combinatorics of words [1].

The following factorization lemma will be useful in several proofs.

Lemma 1 *Let $T = (\Sigma, \Delta, Q, I, F, E, \lambda, \rho)$ be a finite-state transducer, let π be a path from I to state $p \in Q$ and π' a path from I to state $p' \in Q$ with the same input label $w = i[\pi] = i[\pi']$. Assume that $|w| > |Q|^2 - 1$, then there exist paths π_1, π_2, π_3, π'_1, π'_2, π'_3, such that:*

$$\pi = \pi_1 \pi_2 \pi_3 \quad \pi' = \pi'_1 \pi'_2 \pi'_3 \tag{2}$$

where π_2 and π'_2 are cycles with non-empty input labels and: $i[\pi_k] = i[\pi'_k]$, for $k = 1, 2, 3$.

Proof. Consider the transducer U obtained by composing T and T^{-1}: $U = T \circ T^{-1}$. Since π and π' have the same input label, there exists a path ψ in U with input $o[\pi]$ and output $o[\pi']$. Since $|\psi| = |w| > |Q|^2 - 1$ and U has at most $|Q|^2$ states, ψ goes at least through one non-empty cycle ψ_2: $\psi = \psi_1 \psi_2 \psi_3$. This shows the existence of the common factoring for π and π' since ψ results from matching π and the path obtained from π' by swapping its input and output labels. □

The following lemma will be used to prove that determinization terminates when the twins property holds.

Lemma 2 *Assume that T has the twins property. Let R be defined by:*

$$R = \left\{ o[\pi']^{-1} o[\pi] : i[\pi] = i[\pi'] = w, |w| \leq |Q|^2 \right\}$$

Let q_1 and q_2 be two states of T, π a path from I to q_1, and π' a path from I to q_2 with the same input label: $i[\pi] = i[\pi']$, then $o[\pi']^{-1}o[\pi] \in R$.

Proof. Let w be the common input label of π and π' and assume that $|w| > |Q|^2$. By lemma 1, paths π and π' can be factored in the following way:

$$\pi = \pi_1\pi_2\pi_3 \quad \pi' = \pi'_1\pi'_2\pi'_3$$

where π_2 and π'_2 are cycles with non-empty input labels and: $i[\pi_k] = i[\pi'_k]$, for $k = 1, 2, 3$. Let $\phi = \pi_1\pi_3 \in P(I, q_i)$ and $\phi' = \pi'_1\pi'_3 \in P(I, q_j)$ and $w' = i[\phi] = i[\phi']$. Since T has the twins property, $(o[\pi'_1\pi'_2])^{-1}o[\pi_1\pi_2] = o[\pi'_1]^{-1}o[\pi_1]$. Thus: $o[\pi'_1\pi'_2\pi'_3]^{-1}o[\pi_1\pi_2\pi_3] = o[\pi'_1\pi'_3]^{-1}o[\pi_1\pi_3] = o[\phi']^{-1}o[\phi]$. Since $|i[\pi_k]| > 0$, w' is a string strictly shorter than w. By induction, we can find paths $\phi \in P(I, q_1)$ and $\phi' \in P(I, q_2)$, with $i[\phi] = i[\phi'] = w'$, $|w'| \leq |Q|^2$ and such that $o[\pi']^{-1}o[\pi] = o[\phi']^{-1}o[\phi]$, thus $o[\pi']^{-1}o[\pi] \in R$. This proves the lemma. □

The following two lemmas are used in the proof of our main result.

Lemma 3 *Let $x_1, x_2, y_1, y_2 \in \Sigma^*$. Assume that for some integers $r \geq 0$ and $s > 0$, the following holds:*

$$(x_1 y_1^r)^{-1} x_2 y_2^r = (x_1 y_1^{r+s})^{-1} x_2 y_2^{r+s} \tag{3}$$

then:

$$x_1^{-1} x_2 = (x_1 y_1)^{-1} x_2 y_2 \tag{4}$$

Proof. Let $x_1, x_2, y_1, y_2 \in \Sigma^*$ be strings satisfying the hypothesis of the lemma. Without loss of generality, we can assume that $|x_2| \geq |x_1|$. Equality 3 of the lemma can be rewritten as: $y_1^{-r}x_1^{-1}x_2y_2^r = y_1^{-r-s}x_1^{-1}x_2y_2^{r+s}$, or: $y_1^s(x_1^{-1}x_2) = (x_1^{-1}x_2)y_2^s$. Repeated applications of this identity lead to:

$$y_1^{sn}(x_1^{-1}x_2) = (x_1^{-1}x_2)y_2^{sn} \tag{5}$$

for any $n \geq 1$. This implies that $x_1^{-1}x_2$ is a string and that it is a prefix of y_1^{sn}. Thus, y_1 is a period of $x_1^{-1}x_2$ [19]. There exist an integer p, and two strings u and v such that $y = vu$ and $x_1^{-1}x_2 = y^p v$. Re-injecting this in equation 5 gives $y_2 = uv$ and completes the proof of the lemma. □

The following relationship between the output labels of paths of a finitely subsequentiable transducer T and an equivalent finitely subsequential transducer T' with the same input label will be used in out main characterization theorem.

Lemma 4 *Let $T' = (\Sigma, \Delta, Q', \{i'\}, F', E', \lambda', \rho')$ be a finitely subsequential transducer equivalent to $T = (\Sigma, \Delta, Q, I, F, E, \lambda, \rho)$. Let $q \in F$ be a final state of T and $q' \in F'$ a final state of F' and assume that there exists $x \in \Sigma^*$ such that $P(I, x, q) \neq \emptyset$ in T and $P(i', x, q') \neq \emptyset$ in T'. Then, there exists a finite set $Z \subset \Delta^{(*)}$ such that for any paths $\pi \in P(I, q)$ and $\pi' \in P(i', q')$, if $i[\pi] = i[\pi']$ then $o[\pi]^{-1}o[\pi'] \in Z$.*

Proof. Let π and π' be two paths satisfying the hypotheses of the lemma. Since T and T' are equivalent, we have $o[\pi]\rho(q) \subseteq [\![T]\!](i[\pi]) = [\![T']\!](i[\pi])$. Since T' is finitely subsequential we also have $[\![T']\!](i[\pi]) = o[\pi']\rho'(q')$. Thus:

$$o[\pi]\rho(q) \subseteq o[\pi']\rho'(q') \tag{6}$$

Let $x \in \rho(q)$, there exists y in $\rho'(q')$ such that $o[\pi]x = o[\pi']y$. Define $Z \subset \Delta^{(*)}$ as the finite set $Z = \rho(q)\rho'(q')^{-1}$. Then, $o[\pi]^{-1}o[\pi'] = xy^{-1} \in Z$. This proves the lemma. □

Our main characterization result of this section is given by the following theorem which establishes the equivalence between three properties.

Theorem 1 *Let T be a trim finite-state transducer. Then the following three properties are equivalent:*

1. *T is determinizable.*
2. *T has the twins property.*
3. *T is finitely subsequentiable.*

Proof. $1 \Rightarrow 3$: By definition of the algorithm, the output of determinization is a finitely subsequential transducer.

$3 \Rightarrow 2$: Assume that T is finitely subsequentiable and let T' be a finitely subsequential transducer equivalent to T. Let q_1 and q_2 be two siblings in T and consider four paths π_1, c_1, π_2, c_2 as in the definition of the twins property. Since T is trim, there exist a path π'_1 from q_1 to a final state and a path π'_2 from q_2 to a final state. Figure 3(a) illustrates the definition of these paths. Note that π'_1 or π'_2 may be an empty if q_1, resp. q_2, is a final state.

Since T' is equivalent to T, there must be a path π in T from the initial state to a state q with input label xy^r with $r \geq 0$, a cycle c at q with input label y^s with $s > 0$, and two paths μ_1 and μ_2, potentially empty, from q to a final state, with input labels respectively $i[\pi'_1]$ and $i[\pi'_2]$. Figure 3(b) illustrates the definition of these paths.

By definition of these paths, we have for any $t \geq 0$:

$$i[\pi_1 c_1^{r+st} \pi'_1] = i[\pi c^t \mu_1] \tag{7}$$

The conditions of lemma 4 hold with the final states $n[\pi'_1]$ and $n[\mu_1]$ and the paths $\pi_1 c_1^{r+st} \pi'_1$ and $\pi c^t \mu_1$. Thus, there exists a finite set Z such that for any $t \geq 0$:

$$o[\pi_1 c_1^{r+st} \pi'_1]^{-1} o[\pi c^t \mu_1] \in Z \tag{8}$$

Since Z is finite, there exist at least two distinct integers t_0 and t_1 such that:

$$o[\pi_1 c_1^{r+st_0} \pi'_1]^{-1} o[\pi c^{t_0} \mu_1] = o[\pi_1 c_1^{r+st_1} \pi'_1]^{-1} o[\pi c^{t_1} \mu_1] \tag{9}$$

That is:

$$o[\pi'_1] z o[\mu_1]^{-1} = o[\pi_1 c_1^{r+st_0}]^{-1} o[\pi c^{t_0}] = o[\pi_1 c_1^{r+st_1}]^{-1} o[\pi c^{t_1}] \tag{10}$$

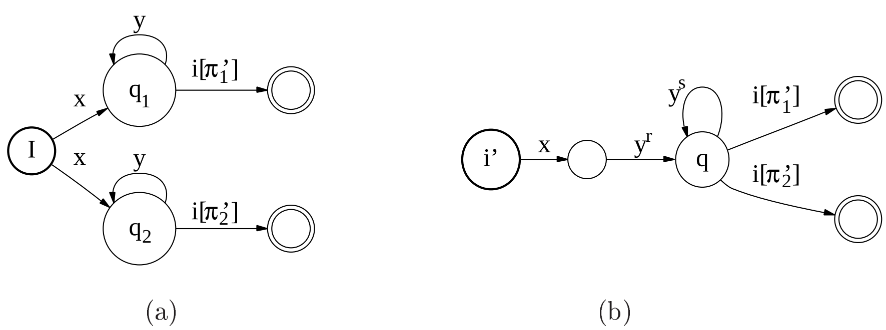


Figure 3: Illustration of the definition of the paths used in the proof of theorem 1. Only the input labels of the paths are indicated. (a) Siblings states q_1 and q_2 in T, paths π_1, c_1, π_2, and c_2 defined as in the definition of the twins property, and paths π'_1 and π'_2 from q_1 and q_2 to final states. (b) State q in T', path π, and paths μ_1 and μ_2 from q to final states in T'.

By lemma 3, this implies:

$$o[\pi_1 c_1^r]^{-1} o[\pi] = o[\pi_1 c_1^{r+s}]^{-1} o[\pi c] \tag{11}$$

We can prove in a similar way that:

$$o[\pi_2 c_2^r]^{-1} o[\pi] = o[\pi_2 c_2^{r+s}]^{-1} o[\pi c] \tag{12}$$

Thus:

$$o[\pi_1 c_1^{r+s}]^{-1} o[\pi_2 c_2^{r+s}] = o[\pi_1 c_1^r]^{-1} o[\pi_2 c_2^r] \tag{13}$$

And by lemma 3:

$$o[\pi_1]^{-1} o[\pi_2] = o[\pi_1 c_1]^{-1} o[\pi_2 c_2] \tag{14}$$

Thus, any two siblings q_1 and q_2 in T are twins, and T has the twins property.

$2 \Rightarrow 1$: Assume that T has the twins property. Let $\{(q_1, z_1), \ldots, (q_n, z_n)\}$ be a subset created during the execution of the determinization algorithm. By construction, states $q_1, \ldots, q_n$ can all be reached from I by paths labeled with the same input string w.[a] Let z be defined by:

$$z = \bigwedge_{p[\pi] \in I,\, i[\pi] = w} o[\pi] \tag{15}$$

By definition of the algorithm, for $i = 1, \ldots, n$, there exists a path π_i from I to q_i with input label w such that:

$$z_i = z^{-1} o[\pi_i] \tag{16}$$

Let $\Pi = \pi_i$ and $\Pi' = \pi_j$ for some $i, j = 1, \ldots, n$. We have:

$$z_j^{-1} z_i = o[\Pi']^{-1} o[\Pi] \tag{17}$$

[a] Note that we may have $q_i = q_j$ for some choices of i and j.

Thus, by lemma 2, $z_j^{-1} z_i \in R$ with:

$$R = \left\{ o[\pi']^{-1} o[\pi] : i[\pi] = i[\pi'] = w, |w| \leq |Q|^2 \right\} \tag{18}$$

Define K as the maximum length of the elements of R: $K = \max_{x \in R} |x|$. Since the remainders in the same subset cannot have a common non-empty prefix, for any remainder string z_i, there exists at least on remainder z_j such that $z_i \wedge z_j = \epsilon$, thus $|z_j| + |z_i| = |z_j^{-1} z_i|$. Since $z_j^{-1} z_i \in R$, we have $|z_j^{-1} z_i| \leq K$, and thus $|z_i| \leq K$. This inequality holds for $i = 1, \ldots, n$, that is any subset remainder belongs to $\Sigma^{\leq K}$. Thus, a subset necessarily belongs to $2^{|Q| \times \Sigma^{\leq K}}$, which is a finite set. This guarantees the termination of the algorithm and thus the determinizability of T. $\square$

We have implemented two algorithms related to these three properties and exploited their equivalence.

5. Experiments and results

We have fully implemented the general determinization algorithm presented in section 3. We used a priority queue implemented with a heap to sort the transitions leaving each subset and another priority queue to sort the final output strings. Since the computation of the transitions leaving a subset only depends on the states and remainder strings of that subset and on the input transducer, one can limit the computation of the result to just the part that is needed. Thus, we gave an on-the-fly implementation of the algorithm which was incorporated in the FSM library [17].

Our experiments in large-vocabulary speech recognition showed the algorithm to be quite efficient. It took about 5s using a Pentium III 700MHz with 2048 Kb of cache and 4Gb of RAM to construct a finitely subsequential transducer equivalent to a transducer T with 440,000 transitions representing the mapping from phonemic sequences to word sequences obtained by composition of two transducers.

We also implemented an efficient algorithm for testing the twins property [1]. With our implementation, the finite subsequentiality of the transducer T already described could be tested in just 60s using the same machine.

6. Conclusion

A new characterization of finitely subsequentiable transducers was given. The twins property was shown to be a necessary and sufficient condition for the finite subsequentiability of a finite-state transducer without requiring it to be finitely functional and a necessary and sufficient condition for the general determinizability of transducers. We reported experimental results demonstrating the practicality of our algorithms for testing finite subsequentiability and for determinizing transducers in large-vocabulary speech recognition applications.

Acknowledgements

We thank Jean-Eric Pin for suggesting the term *finitely subsequential*.

References

1. C. Allauzen and M. Mohri. Efficient Algorithms for Testing the Twins Property. *Journal of Automata, Languages and Combinatorics*, 8(2), 2003.
2. M.-P. Béal, O. Carton, C. Prieur, and J. Sakarovitch. Squaring transducers: An efficient procedure for deciding functionality and sequentiality. *Theoretical Computer Science*, 292:45–63, 2003.
3. J. Berstel. *Transductions and Context-Free Languages.* Teubner Studienbucher: Stuttgart, 1979.
4. T. Chan and O. H. Ibarra. On the finite-valuedness problem for sequential machines. *Theoretical Computer Science*, 23:95–101, 1983.
5. C. Choffrut. Une caractérisation des fonctions séquentielles et des fonctions sous-séquentielles en tant que relations rationnelles. *Theoretical Computer Science*, 5:325–338, 1977.
6. C. Choffrut. *Contributions à l'étude de quelques familles remarquables de fonctions rationnelles.* PhD thesis, (thèse de doctorat d'Etat), Université Paris 7, LITP: Paris, France, 1978.
7. K. Culik II and J. Kari. Digital Images and Formal Languages. In G. Rozenberg and A. Salomaa, editors, *Handbook of Formal Languages*, volume 3, pages 599–616. Springer, 1997.
8. C. Frougny. Conversion between Two Multiplicatively Dependent Linear Numeration Systems. In *Proceedings of Theoretical Informatics, 5th Latin American Symposium (LATIN 2002)*, pages 64–75, Cancun, Mexico, 2002.
9. M. Gross and D. Perrin, editors. *Electronic Dictionnaries and Automata in Computational Linguistics*, volume 377 of *Lecture Notes in Computer Science*. Springer Verlag, 1989.
10. E. M. Gurari and O. H. Ibarra. A note on finite-valued and finitely ambiguous transducers. *Mathematical Systems Theory*, 16(1):61–66, 1983.
11. R. M. Kaplan and M. Kay. Regular Models of Phonological Rule Systems. *Computational Linguistics*, 20(3):331–378, 1994.
12. L. Karttunen. The Replace Operator. In *33rd Annual Meeting of the Association for Computational Linguistics*, pages 16–23. Association for Computational Linguistics, 1995. Distributed by Morgan Kaufmann Publishers, San Francisco, California.
13. M. Mohri. On some Applications of Finite-State Automata Theory to Natural Language Processing. *Journal of Natural Language Engineering*, 2:1–20, 1996.
14. M. Mohri. Finite-State Transducers in Language and Speech Processing. *Computational Linguistics*, 23(2), 1997.
15. M. Mohri. Minimization Algorithms for Sequential Transducers. *Theoretical Computer Science*, 234:177–201, 2000.
16. M. Mohri, F. C. N. Pereira, and M. Riley. Weighted Automata in Text and Speech Processing. In *Proceedings of the 12th biennial European Conference on Artificial Intelligence (ECAI-96), Workshop on Extended finite state models of language*, Budapest, Hungary, 1996. ECAI.
17. M. Mohri, F. C. N. Pereira, and M. Riley. General-Purpose Finite-State Machine Software Tools. *http://www.research.att.com/sw/tools/fsm*, AT&T Labs – Research, 1997.
18. M. Mohri, F. C. N. Pereira, and M. Riley. Weighted Finite-State Transducers in Speech Recognition. *Computer Speech and Language*, 16(1):69–88, 2002.

19. D. Perrin. Words. In M. Lothaire, editor, *Combinatorics on words*, Cambridge Mathematical Library. Cambridge University Press, 1997.

20. M. P. Schützenberger. Sur une variante des fonctions séquentielles. *Theoretical Computer Science*, 4(1):47–57, 1977.

21. A. Weber and R. Klemm. Economy of Description for Single-Valued Transducers. *Information and Computation*, 118(2):327–340, 1995.