

\textbf{p-Subsequentiable Transducers}

Cyril Allauzen and Mehryar Mohri
\{allauzen,mohri\}@research.att.com

AT&T Labs – Research
180 Park Avenue
Florham Park, NJ 07932, USA

\textbf{Abstract.} \textit{p-subsequential} transducers are efficient finite-state transducers with \( p \) final outputs used in a variety of applications. Not all transducers admit equivalent \( p \)-subsequential transducers however. We briefly describe an existing generalized determinization algorithm for \( p \)-subsequential transducers and give the first characterization of \textit{p-subsequentiable transducers}, transducers that admit equivalent \( p \)-subsequential transducers. Our characterization shows the existence of an efficient algorithm for testing \( p \)-subsubsequentiability. We have fully implemented the generalized determinization algorithm and the algorithm for testing \( p \)-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, \( p \)-subsequentiability, and the twins property.

\section{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 and image processing [9, 8, 7, 12, 6].

The time efficiency of such systems is substantially increased when \textit{subsequential transducers} [15], i.e. finite-state transducers with deterministic input, are used. Subsequential machines can be generalized to \textit{p-subsequential transducers} which are transducers with deterministic input with \( p \), \( p \geq 1 \), final output strings [10]. This generalization is necessary in many applications such as language processing to account for finite ambiguities [11].

Not all transducers admit equivalent \( p \)-subsequential transducers however. We present the first characterization of \textit{p-subsequentiable transducers}, i.e. transducers that admit equivalent \( p \)-subsequential transducers.
Our characterization is based on the twins property and leads to an efficient algorithm for testing $p$-subsequentiality. 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, $p$-subsequentiality, and the twins property.

This can also be viewed as a generalization of the results known in the case of functional transducers: determinizable functional transducers are exactly those that admit equivalent subsequential transducers [5]. We generalize these results by relaxing the condition on functionality: determinizable transducers are exactly those that admit equivalent $p$-subsequential transducers and exactly those that admit the twins property.

We have fully implemented the generalized determination algorithm mentioned above and the algorithm for testing $p$-subsequentiality. We report experimental results showing that these algorithms are practical in large-vocabulary speech recognition applications.

We first introduce the notation used in the rest of this paper, then briefly describe a generalized determination algorithm for $p$-subsequential transducers introduced by [10], 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 $\Sigma$ is a finite input alphabet, $\Delta$ 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 $\Delta^*$, and $\rho : F \to 2^\Delta^*$ the final output function mapping each state $q \in F$ to a finite subset of $\Delta^*$.

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: $i[\bigcup_{k \in K} e_k] = \bigcup_{k \in K} i[e_k]$ 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:

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

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

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

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

A successful path in a transducer \( T \) is a path from an initial state to a final state. A state \( a \in Q \) 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 [10] for some integer \( p \) 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 \( p \)-subsequential if there exists a \( p \)-subsequential transducer \( T' \) equivalent to \( T \).

Given two strings \( x \) and \( y \) in \( \Sigma^* \), 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 \( \Sigma \) by associating to each symbol \( a \in \Sigma \) a new symbol denoted by \( a^{-1} \) and define \( \Sigma^{-1} \) 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 \( \Sigma \) and is denoted by \( \Sigma^{(s)} \). 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 \( \Sigma^{(s)} \).
Fig. 1. Generalized determinization of finite-state transducers. (a) Non-deterministic transducer. (b) Construction of equivalent 2-subsequential transducer.

3 General determination algorithm with $p$-subsequential outputs

In this section, we give a brief description of a general determination algorithm introduced by [10] that takes as input a transducer $T$ and outputs a $p$-subsequential transducer $T' = (\Sigma, \Delta, Q', \{i'\}, F', E', \lambda', \rho')$. A transducer $T$ for which the algorithm terminates and thus generates an equivalent $p$-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 pseudocode of the algorithm is given below. 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).
**Fig. 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.

Transducer-Determinization($T$)
1. $F' ← Q' ← E' ← \emptyset$
2. $S ← i' ← \{(i, \lambda(i)) : i ∈ I\}$
3. while $S ≠ \emptyset$
4. do $p' ← head(S)$
5. DEQUEUE($S$)
6. for each $x ∈ i[E[Q']]$
7. do $y' ← \bigwedge\{z y : (p, z) ∈ p', (p, x, y, q) ∈ E\}$
8. $q' ← \{(q, y') : (p, z) ∈ p', (p, x, y, q) ∈ E\}$
9. $E' ← E' ∪ \{(p', x, y, q')\}$
10. if $(q' ∉ Q')$
11. then $Q' ← Q' ∪ \{q'\}$
12. if $Q[q'] ∩ F ≠ \emptyset$
13. then $F' ← F' ∪ \{q'\}$
14. $p'(q') ← \bigcup\{z ρ(q) : (q, z) ∈ q', q ∈ F\}$
15. ENQUEUE($S, q'$)
16. return $T'$

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 $x ∈ Σ$ and output label $y' ∈ Σ^*$ leaving $p'$ if there exists at least one pair $(p, z) ∈ p'$ 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 $(p, z) ∈ p'$ 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 $p$-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 [12].

4 Characterization

This section presents a characterization of $p$-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 $\Sigma^*$ 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 $\pi_1 \in P(I, x, q_1)$, $c_1 \in P(q_1, y, q_1)$, $\pi_2 \in P(I, x, q_2)$, $c_2 \in P(q_2, y, q_2)$,

$$o[\pi_1]^{-1}o[\pi_2] = o[\pi_1c_1]^{-1}o[\pi_2c_2]$$  \hspace{1cm} (1)

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

The twins property was originally introduced by [4, 5] to give a characterization of functional subsequentiable transducers. The decidability of the twins property was also first proved by the same author (see also [3]).

The first polynomial-time algorithm for testing the twins property was given by [16], 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 $\pi$ be a path from $I$ to state $p \in Q$ and $\pi'$ a path from $I$ to $p'$ with the same input label $w = i[\pi] = i[\pi']$. Assume that $|w| > |Q|^2 - 1$, then there exist paths $\pi_1$, $\pi_2$, $\pi_3$, $\pi'_1$, $\pi'_2$, $\pi'_3$, such that:

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

where $\pi_2$ and $\pi'_2$ 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 $\pi$ and $\pi'$ have the same input label, there exists a path $\psi$ in $U$ with input $o[\pi]$ and output $o[\pi']$. Since $|\psi| = |w| > |Q|^2 - 1$ and $U$ has at most $|Q|^2$ states, $\psi$ goes at least through one non-empty cycle $\psi_2$: $\psi = \psi_1 \psi_2 \psi_3$. This shows the existence of the common factoring for $\pi$ and $\pi'$ since $\psi$ results from matching $\pi$ and the path obtained from $\pi'$ 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:

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

Let $q_1$ and $q_2$ be two states of $T$, $\pi$ a path from $I$ to $q_1$, and $\pi'$ 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 $\pi$ and $\pi'$ and assume that $|w| > |Q|^2$. By lemma 1, paths $\pi$ and $\pi'$ can be factored in the following way:

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

where $\pi_2$ and $\pi'_2$ are cycles with non-empty input labels and: $i[\pi_k] = i[\pi'_k]$, for $k = 1, 2, 3$. Let $\phi = \pi_1 \pi_3 \in P(I, q_1)$ and $\phi' = \pi'_1 \pi'_3 \in P(I, q_2)$ and $w' = i[\phi] = i[\phi']$. Since $T$ has the twins property, $(o[\pi_1 \pi_2])^{-1} o[\pi_1 \pi_2] = o[\pi_1 \pi_2]^{-1} o[\pi_1]$. 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: \( y_1^r x_1^{-1} x_2 y_2^r = y_1^{r-s} x_1^{-1} x_2 y_2^{r+s} \), or: \( y_1^n (x_1^{-1} x_2) y_2^n \). Repeated applications of this identity lead to:

\[
y_1^n (x_1^{-1} x_2) y_2^n = (x_1^{-1} x_2)^n \tag{5}
\]

for any \( n \geq 1 \). This implies that \( x_1^{-1} x_2 \) is a string and that it is a prefix of \( y_2^n \). Thus, \( y_1 \) is a period of \( x_1^{-1} x_2 \) [14]. 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. \( \square \)

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

Proof. Let \( \pi \) and \( \pi' \) 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 \( p \)-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 \( Z \subset \Delta^{(s)} \) as the finite set \( Z = \rho(q) \rho'(q')^{-1} \). Then, \( o[\pi]^{-1} o[\pi'] = xy^{-1} \in Z \). This proves the lemma. \( \square \)

Our main characterization result of this section is given by the following theorem which establishes the equivalence between three properties.
Fig. 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 $\pi_1$, $c_1$, $\pi_2$, and $c_2$ defined as in the definition of the twins property, and paths $\pi'_1$ and $\pi'_2$ from $q_1$ and $q_2$ to final states. (b) State $q'$ in $T'$, path $\pi'$, and paths $\mu_1$ and $\mu_2$ from $q$ to final states in $T'$.

**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 $p$-subsequentiable.

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

$3 \Rightarrow 2$: Assume that $T$ is $p$-subsequentiable and let $T'$ be a $p$-subsequential transducer equivalent to $T$. Let $q_1$ and $q_2$ be two siblings in $T$ and consider four paths $\pi_1$, $c_1$, $\pi_2$, $c_2$ as in the definition of the twins property. Since $T$ is trim, there exist a path $\pi'_1$ from $q_1$ to a final state and a path $\pi'_2$ from $q_2$ to a final state. Figure 3 (a) illustrates the definition of these paths. Note that $\pi'_1$ or $\pi'_2$ 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 $\pi$ 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 $\mu_1$ and $\mu_2$, 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 $t \geq 0$:

$$i[\pi_1c_1^{r+st}\pi'_1] = i[\pi c_1^d \mu_1]$$

(7)

The conditions of lemma 4 hold with the final states $n[\pi'_1]$ and $n[\mu_1]$ and the paths $\pi_1c_1^{r+st}\pi'_1$ and $\pi c_1^d \mu_1$. Thus, there exists a finite set $Z$ such that for any $t \geq 0$:

$$o[\pi_1c_1^{r+st}\pi'_1]^{-1}o[\pi c_1^d \mu_1] \in Z$$

(8)
Since $Z$ is finite, there exist at least two distinct integers $t_0$ and $t_1$ such that:

$$o[\pi_1 c_1^{r + s t_0} \pi_1']^{-1} o[\pi c^{t_0} \mu_1] = o[\pi_1 c_1^{r + s t_1} \pi_1']^{-1} o[\pi c^{t_1} \mu_1]$$  (9)

That is:

$$o[\pi_1'] z o[\mu_1]^{-1} = o[\pi_1 c_1^{r + s t_0}]^{-1} o[\pi c^{t_0}] = o[\pi_1 c_1^{r + s t_1}]^{-1} o[\pi c^{t_1}]$$  (10)

By lemma 3, this implies:

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

We can prove in a similar way that:

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

Thus:

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

And by lemma 3:

$$o[\pi_1]^{-1} o[\pi_2] = o[\pi_1 c_1]^{-1} o[\pi_2 c_2]$$  (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 $w$.\(^1\)

Let $z$ be defined by:

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

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

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

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

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

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

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

\(^1\) Note that we may have $q_i = q_j$ for some choices of $i$ and $j$.\n
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 [13].

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 $p$-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 $p$-subsequentiality of the transducer $T$ already described could be tested in just 60s using the same machine.

6 Conclusion

A new characterization of $p$-subsequentiable transducers was given. The twins property was shown to be a necessary and sufficient condition for the $p$-subsequentiality of a finite-state transducer without requiring it to be $p$-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 $p$-subsequentiability and for determinizing transducers in large-vocabulary speech recognition applications.
References