

A Uniformization Theorem for Nested Word to Word Transductions

Dmitry Chistikov and Rupak Majumdar

Max Planck Institute for Software Systems (MPI-SWS)
Kaiserslautern and Saarbrücken, Germany
{dch, rupak}@mpi-sws.org

Technical Report MPI-SWS-2013-002
March 12, 2013

Abstract. We study the class of relations implemented by nested word to word (visibly pushdown) transducers. We show that any such relation can be uniformized by a functional relation from the same class, implemented by an unambiguous transducer. We give an exponential upper bound on the state complexity of the uniformization, generalizing the known Schützenberger construction for the disambiguation of a non-deterministic automaton from standard finite-state machines to nested word (visibly pushdown) automata.

Keywords: uniformization, transduction, nested word, visibly pushdown language.

1 Introduction

A central result in the theory of rational languages is the *unambiguity theorem* [1, 2], which states that every rational function can be implemented by an unambiguous transducer, that is, a transducer that has at most one successful run on any input. Schützenberger [3] gave an elegant and direct proof of the unambiguity theorem, by showing, for any rational function, a matrix representation which can be made unambiguous. It was subsequently shown [2] that the construction of Schützenberger can be used as the foundation for several results in the theory of rational functions, most notably the rational uniformization theorem and the rational cross-section theorem (see [4, 5, 2]).

In more detail, the construction of Schützenberger starts with a (possibly non-deterministic) finite-state transducer \mathcal{T} , and constructs a new finite-state transducer with the property that any input string has at most one successful run. The construction performs a cross product of the subset automaton with the original automaton, and shows that certain edges in the cross product can be removed to obtain unambiguity, while preserving the language. As a simple consequence, the input-output relation that relates two words (u, w) if the transducer \mathcal{T} can output w on input u , can be *uniformized*: for every u in the domain

of \mathcal{T} , we can pick a unique w that is related to it (the rational *uniformization theorem*).

In this paper, we present an extension of the construction by Schützenberger to transducers from *nested words* to words. A nested word consists of a word over an alphabet, together with a *matching relation* that relates “call” positions in the word with corresponding “return” positions. Nested word automata [6] were introduced as a subclass of pushdown automata which are expressive enough for many application domains but nevertheless retain many closure and decidability properties of regular languages. Nested word automata distinguish between “call,” “return,” and “internal” positions of the word, and informally, push a symbol on the runtime stack on a call letter, pop a symbol on a return, and do not touch the stack on an internal letter.

These automata were extended to nested word *transducers* by Raskin and Servais [7] (under the name “visibly pushdown transducers”). The definition of nested word transducers in [7] allows transitions with ε -marking, but ensures that no transition can read and write symbols of different types (call, internal and return symbols, in terms of visibly pushdown automata). We study a model of nested-word to word transducers considered in [8, 9], in which the output word carries no structural information (e.g., call, return, or internal). That is, our transducers can be seen as transforming nested words into (normal) words.

Our main construction generalizes Schützenberger’s construction to give an unambiguous transducer that is at most a single exponential larger than the input transducer. This improves a doubly exponential construction of [10] based on a notion of look-ahead. Our construction relies on a *summarization* of a nested word automaton, which captures the available properly-nested computation paths between two states, as well as the standard determinization construction for nested word automata from [6]. We then show that the product of these automata can be pruned in steps analogous to Schützenberger’s construction for finite-state automata to get an unambiguous automaton.

As a consequence of our construction, we obtain a uniformization theorem for nested-word to word transducers: any relation defined by a nested-word to word transducer can be uniformized by a functional (i. e., single-valued) relation implemented by an unambiguous transducer. A variant of our construction, not described in detail in this extended abstract, also applies to so-called weakly hierarchical nested word automata from [6, 11], which are restricted to record only the current state on the stack.

For several other classes of algebraic (i. e., pushdown) transductions, uniformization theorems were obtained in [12]. However, the natural question of uniformizing nested word transductions was still open.

2 Nested Words and Transducers

A *nested word* of length k over an alphabet Σ is a pair $u = (x, \nu)$, where $x \in \Sigma^k$ and ν is a *matching relation* of length k , that is, a subset $\nu \subseteq \{-\infty, 1, \dots, k\} \times \{1, \dots, k, +\infty\}$ such that, first, if $\nu(i, j)$ holds, then $i < j$; second, for $1 \leq i \leq k$

each of the sets $\{j \mid \nu(i, j)\}$ and $\{j \mid \nu(j, i)\}$ contains at most one element; third, whenever $\nu(i, j)$ and $\nu(i', j')$, it cannot be the case that $i < i' \leq j < j'$. We also assume that $\nu(-\infty, +\infty)$ never holds.

If $\nu(i, j)$, then the position i in the word u is called a *call*, and the position j a *return*. All positions from $\{1, \dots, k\}$ that are neither calls nor returns are *internal*. A call (a return) is *matched* if ν matches it to an element of $\{1, \dots, k\}$ and *unmatched* otherwise.

We shall call a nested word *closed* if it has no unmatched calls, and *well-matched* if it has no unmatched calls and no unmatched returns. We denote the set of all nested words over Σ by Σ^{*n} , the set of all closed nested words by Σ^{*c} , and the set of all well-matched words by Σ^{*w} . Observe that $\Sigma^{*w} \subsetneq \Sigma^{*c} \subsetneq \Sigma^{*n}$.

The family of all non-empty (word) languages over the alphabet Δ is denoted by $\mathcal{L}(\Delta^*)$.

Define a (*nested-word to word*) *transducer* over the input alphabet Σ and output alphabet Δ as a structure $\mathcal{T} = (Q, P, \delta, Q^i, Q^f, P^i)$, where:

- Q is a finite non-empty set of (linear) states,
- P is a finite set of hierarchical states,
- $\delta = (\delta^{\text{call}}, \delta^{\text{int}}, \delta^{\text{ret}})$, where
 - $\delta^{\text{int}} \subseteq Q \times \Sigma \times Q \times \mathcal{L}(\Delta^*)$ is a set of internal transitions,
 - $\delta^{\text{call}} \subseteq Q \times \Sigma \times Q \times P \times \mathcal{L}(\Delta^*)$ is a set of call transitions,
 - $\delta^{\text{ret}} \subseteq P \times Q \times \Sigma \times Q \times \mathcal{L}(\Delta^*)$ is a set of return transitions,
- $Q^i \subseteq Q$ and $Q^f \subseteq Q$ are sets of initial and final linear states, and
- $P^i \subseteq P$ is a set of initial hierarchical states.

A *path* through a transducer \mathcal{T} driven by an input word $u = (a_1 \dots a_k, \nu) \in \Sigma^{*n}$ is a sequence of alternating linear states and transitions of \mathcal{T} , where i th transition leads from the current linear state to the next one, carries the letter $a_i \in \Sigma$ and has type chosen according to the matching relation ν ; furthermore, for every pair of matched call and return, hierarchical states encountered in the corresponding transitions are required to be the same (we say that they are *sent* and *received* along the hierarchical edges). Note that we impose no requirement on unmatched calls—as shown in [6], introducing a separate set of final hierarchical states does not change the expressibility of the model (our automata are then called *linearly accepting*).

The path is *successful* if it starts in a state from Q^i , ends in a state from Q^f , and all states received along the hierarchical edges in unmatched returns belong to P^i . A (*successful*) *computation* consists of a (successful) path and a sequence of words w_i taken from languages $\ell_i \in \mathcal{L}(\Delta^*)$ in the transitions of the path in the same order. The concatenation of these words gives a word $w \in \Delta^*$, which is said to be *output* by the transducer.

We say that the transducer \mathcal{T} *implements* the transduction $T \subseteq \Sigma^{*n} \times \Delta^*$ that contains each pair (u, w) if and only if there exists a successful computation of \mathcal{T} driven by the input u and having the output w . When we are only interested in the behaviour of \mathcal{T} on closed words, we say that \mathcal{T} *weakly implements* $T^c = T \cap (\Sigma^{*c} \times \Delta^*)$.

3 Automata and Auxiliary Constructions

A *nested word automaton* (or simply an *automaton* below) \mathcal{A} is defined similarly to a transducer, with the only difference that it has no output, that is, all $\mathcal{L}(\Delta^*)$ factors are dropped. Words $u \in \Sigma^{*n}$ carried by successful paths are said to be *accepted* by \mathcal{A} , and the automaton itself is then said to *recognize* the language $L \subseteq \Sigma^{*n}$ of all such words. Two automata are *equivalent* if they recognize the same language.

We call \mathcal{A} (*weakly*) *unambiguous* if it has at most one successful path for each (closed) word $u \in \Sigma^{*n}$ ($u \in \Sigma^{*c}$). As usual, \mathcal{A} is *deterministic* if, first, each of the sets Q^i and P^i contains at most one element and, second, for every $q \in Q$, $a \in \Sigma$, and $p \in P$, each of the three sets δ^{int} , δ^{call} , and δ^{ret} contains at most one tuple of the form (q, a, q') , (q, a, q', p) , and (p, q, a, q') , respectively. Every deterministic automaton is unambiguous, but not vice versa. Also recall that a linear state q of an automaton \mathcal{A} is called *accessible* if there exists a path in \mathcal{A} starting in some linear state $q_0 \in Q^i$ and ending in q , in which all states received along the hierarchical edges in unmatched returns belong to P^i .

Determinization. Every automaton is known to be equivalent to a deterministic one, by a variant of the well-known subset construction as defined in [6] (note that the construction described in the print version of this paper is flawed, so we refer the reader to the electronic document available on the Web). Given an automaton $\mathcal{A} = (Q, P, \delta, Q^i, Q^f, P^i)$, construct the automaton with the following components:

- the set of linear states is $2^{Q \times Q}$ (in this context pairs $(q, q') \in Q \times Q$ are called *summaries* and understood as available properly-nested path fragments between pairs of states);
- the set of hierarchical states is $\{p'_0\} \cup (2^{Q \times Q} \times \Sigma)$;
- for every input letter a and every state $S \subseteq Q \times Q$, transitions lead from S to states S' defined as follows:
 - for an internal transition, S' contains all summaries (q, q'') such that there exists a summary $(q, q') \in S$ and an internal transition $(q', a, q'') \in \delta^{\text{int}}$;
 - for a call transition, S' contains all summaries (q'', q'') , for which there exists a summary $(q, q') \in S$ and a call transition $(q', a, q'', p) \in \delta^{\text{call}}$; along the hierarchical edge the pair (S, a) is sent;
 - for a return transition upon the receipt of a hierarchical state $H = (S^0, b)$, where $S^0 \subseteq Q \times Q$, the state S' contains all summaries (q_0, q'') such that there exist summaries $(q_0, q_1) \in S^0$ and $(q, q') \in S$, a call transition $(q_1, b, q, p) \in \delta^{\text{call}}$ and a return transition $(p, q', a, q'') \in \delta^{\text{ret}}$ with matching $p \in P$;
 - for a return transition upon the receipt of a hierarchical state p'_0 , the state S' contains all summaries (q, q'') such that there exists a summary $(q, q') \in S$ and a return transition $(p_0, q', a, q'') \in \delta^{\text{ret}}$ with some $p_0 \in P^i$.
- the only initial linear state is $\{(q_0, q_0) \mid q_0 \in Q^i\}$ (here we deviate from [6], where the set $Q^i \times Q^i$ is used), and an arbitrary linear state S is final whenever it contains some pair $(q, q') \in Q \times Q^f$.

- the only initial hierarchical state is p'_0 .

The accessible part of this automaton is called the *determinization* of \mathcal{A} and denoted \mathcal{A}_{det} . It is deterministic and can be proved equivalent to \mathcal{A} . Also note that \mathcal{A}_{det} *separates unmatched returns*, in the sense that none of its call transitions send an initial hierarchical state along the hierarchical edge.

Summarization. We shall also need another auxiliary automaton, closely related to that of \mathcal{A}_{det} . This automaton will keep track of summaries instead of single states, similarly to \mathcal{A}_{det} , but still rely on nondeterminism rather than subset construction to mimic the behaviour of \mathcal{A} . In short, for any transition in \mathcal{A}_{det} , if it leads from a state containing a summary (q, q') to some state S' , this automaton will have transitions from (q, q') to all summaries that can use (q, q') as a witness for their inclusion in S' . More formally, this summary-tracking automaton has the following components:

- the set of linear states is $Q \times Q$;
- the set of hierarchical states is $\{p'_0\} \cup ((Q \times Q) \times \Sigma)$;
- for every input letter a and every state $(q, q') \in Q \times Q$, transitions from (q, q') are defined as follows:
 - every internal transition $(q', a, q'') \in \delta^{\text{int}}$ of \mathcal{A} is translated here into an internal transition leading to (q, q'') ;
 - every call transition $(q', a, q'', p) \in \delta^{\text{call}}$ is translated into a call transition leading to (q'', q'') , with $((q, q'), a)$ sent along the hierarchical edge;
 - every pair of call and return transitions $(q_1, b, q, p) \in \delta^{\text{call}}$ and $(p, q', a, q'') \in \delta^{\text{ret}}$ with matching $p \in P$ is translated, for all $q_0 \in Q$, into return transitions to (q_0, q'') depending on the state $((q_0, q_1), b)$ received along the hierarchical edge;
 - every return transition $(p_0, q', a, q'') \in \delta^{\text{ret}}$ with $p_0 \in P^i$ is also translated into a return transition to (q, q'') with p'_0 received along the hierarchical edge;
- initial linear states are (q_0, q_0) , for all $q_0 \in Q^i$, and the set of final linear states is $Q \times Q^f$;
- the only initial hierarchical state is p'_0 .

The accessible part of this automaton is called the *summarization* of \mathcal{A} and denoted \mathcal{A}_{sum} .

Product. Let us define the product of two nested word automata $\mathcal{A}_1 = (Q_1, P_1, \delta_1, Q_1^i, Q_1^f, P_1^i)$ and $\mathcal{A}_2 = (Q_2, P_2, \delta_2, Q_2^i, Q_2^f, P_2^i)$ in the standard way:

- sets of linear and hierarchical states are $Q_1 \times Q_2$ and $P_1 \times P_2$, respectively;
- whenever δ_1 and δ_2 contain transitions from q_1 to q'_1 and from q_2 to q'_2 , respectively, having the same type and carrying the same letter $a \in \Sigma$, this results in a transition from (q_1, q_2) to (q'_1, q'_2) of the same type carrying $a \in \Sigma$; the state sent or received along the hierarchical edge, if needed, is the ordered pair of the hierarchical states carried by these transitions;
- sets of initial and final linear states are $Q_1^i \times Q_2^i$ and $Q_1^f \times Q_2^f$;
- the set of initial hierarchical states is $P_1^i \times P_2^i$.

The accessible part of this automaton is called the *product* of \mathcal{A}_1 and \mathcal{A}_2 and denoted $\mathcal{A}_1 \times \mathcal{A}_2$.

Separation. Finally, we shall use the following transformation, which we call the *separation* of a nested word automaton. The purpose of this transformation is to distinguish matched and unmatched call transitions, in the sense that sets of transitions that can be (successfully) invoked on matched and unmatched calls do not intersect. Assume that an automaton $\mathcal{A} = (Q, P, \delta, Q^i, Q^f, P^i)$ is known to separate unmatched returns in the sense defined above.

Construct the automaton \mathcal{A}_{sep} as follows:

- every linear state $q \in Q$ is *annotated* with symbols from $\{\text{main}, \text{sub}\}$, i. e., split into two, q_{main} and q_{sub} ;
- every hierarchical state $p \in P$ is annotated with symbols from $\{\text{z}, \text{n}, \text{uc}, \text{ur}\}$;
- transitions are replaced in the following way:
 - $(q, a, q') \in \delta^{\text{int}}$ with $(q_{\text{main}}, a, q'_{\text{main}})$ and $(q_{\text{sub}}, a, q'_{\text{sub}})$;
 - $(q, a, q', p) \in \delta^{\text{call}}$ with $(q_{\text{main}}, a, q'_{\text{main}}, p_{\text{uc}})$, $(q_{\text{main}}, a, q'_{\text{sub}}, p_{\text{z}})$, and $(q_{\text{sub}}, a, q'_{\text{sub}}, p_{\text{n}})$;
 - $(p, q, a, q') \in \delta^{\text{ret}}$, $p \notin P^i$, with $(p_{\text{n}}, q_{\text{sub}}, a, q'_{\text{sub}})$ and $(p_{\text{z}}, q_{\text{sub}}, a, q'_{\text{main}})$;
 - $(p, q, a, q') \in \delta^{\text{ret}}$, $p \in P^i$, with $(p_{\text{ur}}, q_{\text{main}}, a, q'_{\text{main}})$;
- sets of initial and final states are $\{q_{\text{main}} \mid q \in Q^i\}$ and $\{q_{\text{main}} \mid q \in Q^f\}$, respectively;
- the set of initial hierarchical states is $\{p_{\text{ur}} \mid p \in P^i\}$.

Claim. Sets of successful paths through \mathcal{A}_{sep} and \mathcal{A} are set in bijection by the mapping τ stripping annotations from states of \mathcal{A}_{sep} . Furthermore, for every $u \in \Sigma^{*n}$ accepted by \mathcal{A} and every successful path through \mathcal{A}_{sep} driven by u , all unmatched calls, unmatched returns, and matched calls and returns carry states from disjoint sets P_{uc} , P_{ur} , and $P_{\text{z}} \cup P_{\text{n}}$ on the hierarchical edges, respectively.

4 The Schützenberger Construction for Nested Word Automata

Suppose that \mathcal{B} and \mathcal{A} are nested word automata, and there exists a mapping μ taking states and transitions of \mathcal{B} to states and transitions of \mathcal{A} so that, first, for all linear states s' of \mathcal{B} , transitions arriving in (departing from) s' are mapped to transitions arriving in (departing from) $\mu(s')$; second, initial and final states are mapped to initial and final states, respectively; so that the image of a successful path is always a successful path. Then the automaton \mathcal{B} is said to be an *immersion* in \mathcal{A} , or *immersed* in \mathcal{A} by μ .

We define a special operation eliminating some transitions of an automaton, in the spirit of Schützenberger [3, 2]. Suppose that \mathcal{B} is immersed in \mathcal{A} by μ , and δ' is some subset of \mathcal{B} 's transitions. For every state s' of \mathcal{B} , consider its image $\mu(s')$ in \mathcal{A} and take an arbitrary transition t arriving at $\mu(s')$. Denote the set of all transitions arriving at s' by $\text{IN}(s')$. If the set $\mu^{-1}(t) \cap \text{IN}(s') \cap \delta'$ has cardinality 2 or greater, i. e., if the inverse image $\mu^{-1}(t)$ contains more than one transition from δ' arriving at s' , then transform the automaton \mathcal{B} by eliminating all of these transitions but one, arbitrarily chosen. Repeat this procedure for all states s' of \mathcal{B} and all transitions t arriving at their images $\mu(s')$. We shall say

that the resulting automaton is obtained from \mathcal{B} by *weeding* transitions δ' with respect to the mapping μ .

Now everything is ready for describing our generalization of the Schützenberger construction to nested word automata. Suppose that we are given an automaton \mathcal{A} . First construct the automaton $\mathcal{A}_{\text{sum}} \times \mathcal{A}_{\text{det}}$, where \mathcal{A}_{sum} and \mathcal{A}_{det} are the summarization and determinization of \mathcal{A} . Let the mapping π take linear states and transitions of $\mathcal{A}_{\text{sum}} \times \mathcal{A}_{\text{det}}$ to their projections in \mathcal{A}_{det} . Weed the sets of internal and return transitions of $\mathcal{A}_{\text{sum}} \times \mathcal{A}_{\text{det}}$ with respect to π (note that all call transitions are left intact), and then for every final state S of \mathcal{A}_{det} , make all but one state in $\pi^{-1}(S)$ non-final (this remaining final state should have the form $((q, q'), S)$, where (q, q') is final in \mathcal{A}_{sum}). Denote the obtained automaton by \mathcal{S}_1 .

Theorem 1. *The automaton \mathcal{S}_1 is weakly unambiguous, equivalent to and immersed in \mathcal{A} .*

The proof of this theorem is given in Section 6. The automaton \mathcal{S}_1 can be used to obtain a uniformization of any relation weakly implemented by a transducer whose underlying automaton is \mathcal{A} . (Recall that this relation is defined as a subset of $\Sigma^{*c} \times \Delta^*$.) If we cannot disregard non-closed words, then an auxiliary step is needed.

Take the separation $(\mathcal{S}_1)_{\text{sep}}$ and consider the set δ^{uc} of all call transitions sending a state of the form p_{uc} along the hierarchical edge. Define the mapping π_0 on the linear states of $(\mathcal{S}_1)_{\text{sep}}$ by the rule $\pi_0(((q, q'), S)_{\text{main}}) = \pi_0(((q, q'), S)_{\text{sub}}) = S$ and extend it to transitions in the straightforward fashion. Now weed the set of transitions δ^{uc} with respect to the mapping π_0 and denote the obtained automaton by \mathcal{S}_2 .

Theorem 2. *The automaton \mathcal{S}_2 is unambiguous, equivalent to and immersed in \mathcal{A} .*

The proof of this theorem is also given in Section 6.

5 Uniformization of Transductions

Return to the problem of uniformizing an arbitrary nested-word to word transduction T . Recall that a relation $U \subseteq A \times B$ is a *uniformization* of a relation $T \subseteq A \times B$ if U is a subset of T , single-valued as a transduction, and has the same domain, that is, if $U \subseteq T$ and for every $u \in A$, the existence of a $w \in B$ such that $(u, w) \in T$ implies that there is exactly one $w' \in B$ such that $(u, w') \in U$.

In this section, we show how to use the Schützenberger construction to obtain a uniformization of an arbitrary nested-word to word transduction T . Take any transducer \mathcal{T} implementing T , and denote by \mathcal{A} its underlying automaton, that is, one obtained by removing the output labels from transitions. Transform \mathcal{A} into \mathcal{S}_1 or \mathcal{S}_2 as described in Section 4 (denote the result by \mathcal{S}). By the “immersion” part of Theorems 1 and 2, \mathcal{S} is immersed in \mathcal{A} by some mapping ρ , so that each

transition t' in \mathcal{S} is projected onto a single transition $\rho(t')$ in \mathcal{A} . Take the output label $\ell \subseteq \Delta^*$ of $\rho(t')$ in \mathcal{T} and specify any single word $w \in \ell$ as the output of t' . The automaton \mathcal{S} is thus transformed into a transducer \mathcal{U} , which is claimed to satisfy our needs.

Theorem 3. *Let the transducer \mathcal{T} (weakly) implement a transduction T . Then the transducer \mathcal{U} (weakly) implements a transduction U , which is a uniformization of T .*

Proof. We use the “equivalence” and “(weak) unambiguity” parts of Theorems 1 and 2. First observe that any nested word u belongs to the domain of T (or, respectively, U) if and only if it is accepted by \mathcal{A} (or, respectively, \mathcal{S}). It then follows from the “equivalence” part that T and U have the same domain within Σ^{*n} . Second, if (u, w') and (u, w'') with $w' \neq w''$ are in U , then \mathcal{U} has at least two successful paths driven by u . This is impossible for any (closed) word u by the “(weak) unambiguity” part of Theorems 1 and 2. \square

We say that a relation $U \subseteq \Sigma^{*c} \times \Delta^*$ is a *weak uniformization* of a transduction $T \subseteq \Sigma^{*n} \times \Delta^*$ if U is a uniformization of the transduction $T^c = T \cap (\Sigma^{*c} \times \Delta^*)$.

Corollary. *Any nested-word to word transduction T implemented by a transducer with n linear states has a weak uniformization implemented by a transducer with at most $n^2 2^{n^2-1}$ linear and $1 + |\Sigma| n^2 2^{n^2-1}$ hierarchical states, and a uniformization implemented by a transducer with at most $n^2 2^{n^2}$ linear and $1 + 3|\Sigma| n^2 2^{n^2-1}$ hierarchical states.*

6 Proofs of Theorems 1 and 2

This section is developed as a series of lemmas. Our first lemma gives an upper bound on the set of all accessible states of the automaton $\mathcal{A}_{\text{sum}} \times \mathcal{A}_{\text{det}}$.

Lemma 1. *If a state $((q, q'), S)$ is accessible in $\mathcal{A}_{\text{sum}} \times \mathcal{A}_{\text{det}}$, then (q, q') belongs to S and S is accessible in \mathcal{A}_{det} .*

Proof. It is clear that no state of the form $((q, q'), S)$ can be accessible in $\mathcal{A}_{\text{sum}} \times \mathcal{A}_{\text{det}}$ unless the state S is accessible in \mathcal{A}_{det} . The fact that $(q, q') \in S$ is then proved by induction on the length of the path leading from the initial state of \mathcal{A}_{det} to S . Indeed, $\{(q_0, q_0) \mid q_0 \in Q^i\}$ is the only initial state of \mathcal{A}_{det} and the states (q_0, q_0) , $q_0 \in Q^i$, are initial in \mathcal{A}_{sum} . This forms the induction base. Further, suppose that $((q_1, q_2), S)$ is accessible and a transition leads from this state to some state $((q, q'), S')$. Then $(q_1, q_2) \in S$ and, by the definition of $\mathcal{A}_{\text{sum}} \times \mathcal{A}_{\text{det}}$, there are transitions from S to S' and from (q_1, q_2) to (q, q') . It remains to use the fact that whenever a transition leads in \mathcal{A}_{sum} from (q_1, q_2) to (q, q') , any corresponding transition in \mathcal{A}_{det} from a state containing (q_1, q_2) can only lead to a state containing (q, q') . \square

Lemma 1 shows that every accessible state of $\mathcal{A}_{\text{sum}} \times \mathcal{A}_{\text{det}}$ can be regarded as a set of summaries S with a distinguished element $(q, q') \in S$. From now on we shall only use the states of $\mathcal{A}_{\text{sum}} \times \mathcal{A}_{\text{det}}$ satisfying the conclusion of this lemma. For our second lemma, recall that by π we denote the mapping taking every state of $\mathcal{A}_{\text{sum}} \times \mathcal{A}_{\text{det}}$ to its projection in \mathcal{A}_{det} .

Lemma 2 (in-surjectivity). *For every accessible state $((q, q'), S)$ of $\mathcal{A}_{\text{sum}} \times \mathcal{A}_{\text{det}}$ and every transition t arriving at its projection S in \mathcal{A}_{det} , there exists at least one transition in $\mathcal{A}_{\text{sum}} \times \mathcal{A}_{\text{det}}$ whose projection is t .*

Proof. First apply Lemma 1 to note that $(q, q') \in S$. Let the transition t arriving at S depart from a state S^0 . We claim that there exists a summary $(q_1, q_2) \in S^0$ and a transition from (q_1, q_2) to (q, q') in \mathcal{A}_{sum} such that the inverse image $\pi^{-1}(t)$ in $\mathcal{A}_{\text{sum}} \times \mathcal{A}_{\text{det}}$ contains a transition from $((q_1, q_2), S^0)$ to $((q, q'), S)$.

Indeed, since $(q, q') \in S$, it follows that there exists a summary $(q_1, q_2) \in S^0$ that guaranteed the inclusion of (q, q') into S by the definition of \mathcal{A}_{det} . The definition of \mathcal{A}_{sum} then ensures the existence of a transition of the same type (internal, call, or return with the same input letter and the same state sent or received, if at all, along the hierarchical edge) from this (q_1, q_2) to (q, q') . The rest follows by the definition of $\mathcal{A}_{\text{sum}} \times \mathcal{A}_{\text{det}}$. \square

Lemma 3 (partial in-bijectivity). *For every state $((q, q'), S)$ of \mathcal{S}_1 and every internal or return transition t arriving at its projection S in \mathcal{A}_{det} , there exists a unique transition in \mathcal{S}_1 whose projection is t .*

Proof. Follows from Lemma 2, the definitions of \mathcal{S}_1 and the weeding operation. \square

It follows from Lemma 1 and the definitions of \mathcal{A}_{sum} and \mathcal{A}_{det} that the product $\mathcal{A}_{\text{sum}} \times \mathcal{A}_{\text{det}}$ and, therefore, the automaton \mathcal{S}_1 always send states of the form $((q, q'), b, S, b)$, where $(q, q') \in S$, along the hierarchical edges. We shall henceforth always abbreviate this and write $((q, q'), S, b)$ instead.

Lemma 4. *The automaton \mathcal{S}_1 is weakly unambiguous.*

Proof. Consider any closed word $u \in \Sigma^{*c}$ accepted by \mathcal{S}_1 and assume for the sake of contradiction that there are two different successful paths driven by u . Now observe that the projection by means of π of a successful path in \mathcal{S}_1 is a successful path in \mathcal{A}_{det} . Since the automaton \mathcal{A}_{det} is deterministic and, therefore, unambiguous, it follows that both successful paths in \mathcal{S}_1 are projected onto a single path in \mathcal{A}_{det} .

Consider the last position in this pair of paths, on which these paths disagree, that is, $((q_1, q'_1), S) \neq ((q_2, q'_2), S)$ (note that the projection onto \mathcal{A}_{det} should be the same, hence the common component S). The successors of these two states are the same in both paths, and the following segments up to the end of the paths also coincide (both paths end in the same final state of \mathcal{S}_1 , since for each S at most one state from $\pi^{-1}(S)$ is final). Suppose that our chosen pair of paths leads from this pair of states to some state $((q, q'), S')$.

Now consider the transition t leading from S to S' in the (single) induced path in \mathcal{A}_{det} . This transition has at least two elements in its inverse image $\pi^{-1}(t)$, one of them departing from $((q_1, q'_1), S)$ and another from $((q_2, q'_2), S)$. By the partial in-bijectivity given by Lemma 3, this transition t can only be a (matched) call transition. We shall show, however, that this conclusion leads to a contradiction.

Indeed, consider the remaining segments of the paths, which coincide according to our initial assumptions. It follows that these segments take the same return transition at the return position matching the call in question (recall that our word is closed, so all calls are matched). This transition receives some state $((q_0, q'_0), S^0, a)$ along the hierarchical edge, and, according to the definitions of \mathcal{S}_1 , $\mathcal{A}_{\text{sum}} \times \mathcal{A}_{\text{det}}$ and \mathcal{A}_{sum} , it follows that the call in question can only originate at the state $((q_0, q'_0), S^0)$. This contradicts the availability of the choice between $((q_1, q'_1), S)$ and $((q_2, q'_2), S)$ and establishes that one of the paths must be invalid. \square

Our next goal is to show that every successful path in \mathcal{A}_{det} has an inverse image in \mathcal{S}_1 that is also a successful path. To prove this fact, we need an auxiliary lemma that describes the behaviour of \mathcal{A}_{sum} on well-matched words.

Lemma 5. *Suppose that the automaton \mathcal{A}_{sum} can be driven by some well-matched word $u \in \Sigma^{*w}$ from a state (q_0, q_1) to a state (q_2, q_3) . Then $q_0 = q_2$.*

Proof. Any well-matched word is a concatenation of internal transitions and well-matched fragments enclosed in matched pairs of symbols. By the definition of \mathcal{A}_{sum} , internal transitions do not change the first component of a summary, and every matched return always resets it to the state just before the matching call. \square

Lemma 6. *The automaton \mathcal{S}_1 is equivalent to \mathcal{A} .*

Proof. Since the projection of a successful path through \mathcal{S}_1 is a successful path through \mathcal{A}_{det} , it is sufficient to demonstrate that \mathcal{S}_1 always accepts whenever \mathcal{A}_{det} accepts. Consider a successful path c through \mathcal{A}_{det} and construct a path through \mathcal{S}_1 as follows. First choose the (unique) final state in the inverse image of the last state of c . Second, reconstruct a path through \mathcal{S}_1 by the following procedure. On each step, given a state of \mathcal{S}_1 and a transition t arriving at its projection in \mathcal{A}_{det} , choose a transition from the inverse image $\pi^{-1}(t)$ to obtain the previous state in the path through \mathcal{S}_1 . Lemmas 2 and 3 reveal that this procedure will indeed yield a path in \mathcal{S}_1 . Since all summaries from the initial state of \mathcal{A}_{det} are initial states of \mathcal{A}_{sum} , this path will begin in an initial state of \mathcal{S}_1 . However, we also need to show that one of these paths will indeed be a correct path through \mathcal{S}_1 , that is, states sent and received along the hierarchical edges will match.

Consider a fragment of the input word of the form $\langle bua \rangle$, where $u \in \Sigma^{*w}$. Suppose that the automaton \mathcal{A}_{det} is driven by a) from a state S to a state S' , and the reconstructed path in \mathcal{S}_1 distinguishes summaries $(q, q') \in S$ and $(q_0, q'') \in S'$, with \mathcal{A}_{sum} driven by a) from the former to the latter. Also suppose that

at this point a transition receiving a state $((q_0, q_1), S^0, b)$ along the hierarchical edge is chosen in \mathcal{S}_1 . Now assume that the matching call drives \mathcal{A}_{det} from S^0 to some state S'' . We need to show that when the reconstruction of the path in \mathcal{S}_1 reaches this call, the state S'' will have been mapped to the state $((q, q), S'')$ and a transition to this state from $((q_0, q_1), S^0)$ will be available.

The second of these claims is relatively straightforward. Indeed, by the definitions of \mathcal{A}_{det} and \mathcal{A}_{sum} , a transition from $((q, q'), S)$ to $((q_0, q''), S')$ driven by a upon the receipt of $((q_0, q_1), S^0, b)$ along the hierarchical edge is witnessed by a pair of call and return transitions of the form $(q_1, b, q, p) \in \delta^{\text{call}}$, $(p, q', a, q'') \in \delta^{\text{ret}}$ with matching $p \in P$. Since a call transition from S^0 to S'' is available in \mathcal{A}_{det} with an input letter b , it follows that a call transition with the same input letter leads from $((q_0, q_1), S^0)$ to $((q, q), S'')$ in \mathcal{S}_1 .

Now turn to the first of the claims. Recall that we are now dealing with a fragment of the input word of the form $\langle bua \rangle$ with $u \in \Sigma^{*w}$. Observe that our entire argument can be interpreted as a proof that the reconstructed path segment induces a correct path segment in \mathcal{A}_{sum} . We now use induction to prove this fact. The base case corresponds to words containing internal symbols only, and does not require any analysis. The inductive step corresponds to words of the form $\langle bua \rangle$ with $u \in \Sigma^{*w}$, as specified earlier. Here, since just before the input symbol a the automaton \mathcal{A}_{sum} is set to the state (q, q') and the word u is well-matched, one concludes with the help of the inductive hypothesis and Lemma 5 that after reading the symbol b the automaton \mathcal{A}_{sum} must have been in some state of the form (q, \bar{q}) , where $\bar{q} \in Q$. Since this state (q, \bar{q}) is taken from S'' , and the state S'' is the destination of some call transition in \mathcal{A}_{det} , it follows from the definition of \mathcal{A}_{det} that $\bar{q} = q$. This concludes the proof of Lemma 6. \square

For the following lemma we need the mapping π_0 defined earlier in Section 4. This mapping strips annotations from states of \mathcal{S}_2 and then removes \mathcal{A}_{sum} -components of states, thus sending states of \mathcal{S}_2 to states of \mathcal{A}_{det} .

Lemma 7. *The automaton \mathcal{S}_2 is unambiguous.*

Proof. It follows from Lemma 4 and the properties of the separation construction that \mathcal{S}_2 is at least weakly unambiguous. Now take any non-closed word $u \in \Sigma^{*n}$ and consider any two paths through \mathcal{S}_2 driven by u . By the properties of the separation construction, these two paths have different counterparts in \mathcal{S}_1 . However, their projections onto \mathcal{A}_{det} by means of π_0 are the same, since \mathcal{A}_{det} is deterministic and, therefore, unambiguous.

Consider the last position, on which these two paths disagree. Transitions in these paths corresponding to this position are different, but projected to the same transition in \mathcal{A}_{det} . As shown in the proof of Lemma 4, this projection cannot be an internal or a return transition, nor a matched call. It can only be an unmatched call, but in this case two different unmatched call transitions are sent by π_0 to a single transition of \mathcal{A}_{det} . This contradicts our definition of \mathcal{S}_2 , since one of these transitions must have been weeded out. \square

Lemma 8. *The automaton \mathcal{S}_2 is equivalent to \mathcal{A} .*

Proof. Recall that by Lemma 6 the automaton \mathcal{S}_1 is equivalent to \mathcal{A} , and the automaton \mathcal{S}_2 is obtained by eliminating some transitions from $(\mathcal{S}_1)_{\text{sep}}$. It is then sufficient to demonstrate that \mathcal{S}_2 accepts whenever \mathcal{A} accepts or, equivalently, whenever \mathcal{A}_{det} accepts.

Consider any successful path through \mathcal{A}_{det} and construct a corresponding path through \mathcal{S}_2 , similarly to the proof of Lemma 6. Note that here annotations of states are chosen unambiguously, according to the matching structure of the input word. For matched call, internal, and return transitions, the existence of an appropriate transition in \mathcal{S}_2 follows from the previous arguments. Among unmatched calls, some of the transitions in \mathcal{S}_2 may have been weeded out, but at least one option on each step is still available. \square

Lemma 9. *The automata \mathcal{S}_1 and \mathcal{S}_2 are immersed in \mathcal{A} .*

Proof. Stripping annotations from the states of \mathcal{S}_2 turns them into states of \mathcal{S}_1 . Removing \mathcal{A}_{det} -components maps them to the set of states of \mathcal{A}_{sum} . Transitions are mapped to transitions of \mathcal{A}_{sum} accordingly. It remains to observe that \mathcal{A}_{sum} is itself an immersion in \mathcal{A} . \square

Theorem 1 follows from Lemmas 4, 6 and 9. Theorem 2 follows from Lemmas 7, 8 and 9.

References

1. Kobayashi, K.: Classification of formal languages by functional binary transducers. *Information and Control* 15, 95–109 (1969)
2. Sakarovitch, J.: A construction on finite automata that has remained hidden. *Theoretical Computer Science* 204, 205–231 (1998)
3. Schützenberger, M.P.: Sur les relations rationnelles entre monoïdes libres. *Theoretical Computer Science* 3, 243–259 (1976)
4. Eilenberg, S.: *Automata, Languages, and Machines*. Vol. A. Academic Press (1974)
5. Sakarovitch, J.: *Elements of Automata Theory*. Cambridge University Press (2009)
6. Alur, R., Madhusudan, P.: Adding nesting structure to words. *Journal of the ACM* 56(3), 16 (2009); revised version available online at <http://robotics.upenn.edu/~alur/Jacm09.pdf>
7. Raskin, J.-F., Servais, F.: Visibly pushdown transducers. In: L. Aceto et al. (Eds.) *ICALP 2008, Part II*. LNCS, vol. 5126, pp. 386–397. Springer, Heidelberg (2008)
8. Staworko, S., Laurence, G., Lemay, A., Niehren, J.: Equivalence of deterministic nested word to word transducers. In: Kutylowski, M., Gębala, M., Charatonik, W. (Eds.) *FCT 2009*. LNCS, vol. 5699, pp. 310–322. Springer, Heidelberg (2009)
9. Filiot, E., Raskin, J.-F., Reynier, P.-A., Servais, F., Talbot, J.-M.: Properties of visibly pushdown transducers. In: Hliněný, P., Kučera, A. (Eds.) *MFCS 2010*. LNCS, vol. 6281, pp. 355–367. Springer, Heidelberg (2010)
10. Filiot, E., Servais, F.: Visibly pushdown transducers with look-ahead. In: M. Bieliková et al. (Eds.) *SOFSEM 2012*. LNCS, vol. 7147, pp. 251–263. Springer, Heidelberg (2012)
11. Alur, R., Madhusudan, P.: Adding nesting structure to words. In: Ibarra, O.H., Dang, Z. (Eds.) *DLT 2006*. LNCS, vol. 4036, pp. 1–13. Springer, Heidelberg (2006)
12. Konstantinidis, S., Santean, N., Yu, S.: Representation and uniformization of algebraic transductions. *Acta Informatica* 43, 395–417 (2007)