Hyper-Optimization for Deterministic Tree Automata

Andreas Maletti*

Institute for Natural Language Processing, Universität Stuttgart
Pfaffenwaldring 5b, 70569 Stuttgart, Germany
andreas.maletti@ims.uni-stuttgart.de

Abstract. A recent minimization technique, called hyper-minimization, permits reductions of language representations beyond the limits imposed by classical semantics-preserving minimization. Naturally, the semantics is not preserved by hyper-minimization; rather the reduced representation, which is called hyper-minimal, can accept a language that has a finite symmetric difference to the language of the original representation. It was demonstrated that hyper-minimization for (bottom-up) deterministic tree automata (DTAs), which represent the recognizable tree languages, can be achieved in time $\mathcal{O}(m \cdot \log n)$, where m is the size of the DTA and n is the number of its states. In this contribution, this result is complemented by two results on the quantity of the errors. It is shown that optimal hyper-minimization for DTAs (i.e., computing a hyper-minimal DTA that commits the least number of errors of all hyper-minimal DTAs) can be achieved in time $\mathcal{O}(m \cdot n)$. In the same time bound also the number of errors of any hyper-minimal DTA can be computed.

1 Introduction

In many application areas, large finite-state models are approximated automatically from data. Classical examples in the area of natural language processing include the estimation of tree automata [6,7] for parsing [18] and weighted finite-state automata [19] for speech recognition [16]. To keep the size of those models under control, minimization is used whenever possible and efficient. Unfortunately, computing an equivalent minimal nondeterministic (unweighted) finite-state automaton [21] is PSPACE-complete [8] and thus inefficient; this remains true even if the input automaton is deterministic. However, given a deterministic finite-state automaton (DFA) the computation of an equivalent minimal DFA is very efficient [12]. Consequently, we restrict our focus to deterministic finite-state devices. Exactly, the same situation exhibits itself for tree automata [17,3], which are the finite-state models used in this contribution. We note that (bottom-up) deterministic tree automata are as expressive as (nondeterministic) tree automata (albeit the deterministic device might require exponentially more states as in the string case), which recognize exactly the regular tree languages.

* The author was financially supported by the German Research Foundation (DFG) grant MA 4959 / 1-1.

S. Konstantinidis (Ed.): CIAA 2013, LNCS 7982, pp. 244–255, 2013.

In several applications it is beneficial to reduce the size even further at the expense of errors. In hyper-minimization [2] we simply allow any finite number of errors; i.e., the obtained representation might recognize a language that has a finite symmetric difference to the language recognized by the original representation. While this error profile is rather simplistic, it allows a convenient theoretical treatment [2], efficient minimization algorithms [1,4,11,14], and sometimes finitely many errors are even absolutely inconsequential [20]. Moreover, more refined error profiles often yield NP-hard minimization problems [5] and thus inefficient minimization procedures. Recently, an efficient hyper-minimization algorithm [13] for (bottom-up) deterministic tree automata (DTAs) was developed. It runs in time $\mathcal{O}(m \cdot \log n)$, where m is the size of the input DTA and n is the number of its states. Thus, it is asymptotically as efficient as the fastest classical minimization algorithms [9] for DTAs.

The existing hyper-minimization algorithm for DTAs is purely qualitative in the sense that it guarantees that the resulting hyper-minimal DTA (a DTA M is hyper-minimal if there exists no DTA with fewer states[1] that recognizes a tree language with a finite difference to the tree language recognized by M) commits only finitely many errors, but provides no (non-trivial) bound on this number of errors. Since there are (in general) many (non-isomorphic) hyper-minimal DTAs for a given tree language, returning simply any hyper-minimal DTA is shortsighted. In this contribution, we perform a more quantitative analysis in the spirit of [15]. We develop a hyper-minimization algorithm that returns a hyper-minimal DTA (i.e., it has as many states as the DTA returned by the existing algorithm of [13]) that commits the least number of errors among all hyper-minimal DTAs. To this end, we first characterize all hyper-minimal DTAs for a given tree language. For DFAs the structural differences between hyper-minimal DFAs for the same language were characterized in [2, Thms. 3.8 and 3.9]. Despite the additional complications encountered in DTAs hyper-minimization, we faithfully generalize the results for DFAs to DTAs. Thus, any two hyper-minimal DTAs for a given tree language permit a bijection between their states such that the distinction into preamble (i.e., those states that can only be reached by finitely many trees) and non-preamble (or kernel) states is preserved. Moreover, the DTAs behave equivalently on their preambles except for their acceptance decisions and isomorphically on their kernels. Finally, the strange condition on the initial state in [2, Thms. 3.8 and 3.9] disappears completely for DTAs.

With the help of this characterization we can now easily compare different hyper-minimal DTAs provided that we can compute the number of errors that they commit. Thus, we derive a method to compute the number of errors caused by each relevant decision (finality decision for preamble states and transition targets for transitions from the preamble into the kernel). For DFAs the same approach was used in [15], but our method is slightly more complicated because we have to avoid counting errors several times (because an error tree can contain multiple positions at which a switch from preamble to kernel states happens

[1] Since we consider only deterministic devices, we can as well use the number of transitions as a size measure.

when processing it by the DTA). We solve this problem by attributing the error tree to the left-most such transition. It turns out that this change can easily be incorporated, so that our approach closely resembles the approach of [15]. Overall, we obtain an algorithm that, given a DTA M and a hyper-minimal DTA N that recognizes a finitely different tree language, can compute the number of errors committed by N in time $\mathcal{O}(m \cdot n)$, where m is the size of M and n is the number of states of M. In addition, we can also compute an optimal hyper-minimal DTA N' in time $\mathcal{O}(m \cdot n)$, which is a hyper-minimal DTA that commits the least number of errors among all hyper-minimal DTAs that recognize a finitely different tree language. Of course, we can also compute the exact number of errors committed by this optimal DTA.

2 Preliminaries

The set $\mathbb{N}$ consists of all nonnegative integers and $[k] = \{i \in \mathbb{N} \mid 1 \leq i \leq k\}$ for all $k \in \mathbb{N}$. The symmetric difference $S \ominus T$ of sets S and T is $(S - T) \cup (T - S)$. A binary relation $\cong \subseteq S \times S$ is an equivalence relation if it is reflexive, symmetric, and transitive. Given such an equivalence relation $\cong$, the equivalence class $[s]$ of $s \in S$ is $\{s' \in S \mid s \cong s'\}$. If S is finite, then we write $|S|$ for its cardinality.

An alphabet Σ is a finite set, and a ranked alphabet (Σ, rk) consists of an alphabet Σ and a mapping $\mathrm{rk} \colon \Sigma \to \mathbb{N}$ that assigns a rank to each symbol of Σ. The set of all symbols of rank $k \in \mathbb{N}$ is $\Sigma_k = \mathrm{rk}^{-1}(k)$. We typically denote (Σ, rk) by just Σ, and we let $\Sigma(T) = \{\sigma(t_1, \ldots, t_k) \mid \sigma \in \Sigma_k, t_1, \ldots, t_k \in T\}$ for every set T. The set $T_\Sigma(Q)$ of Σ-trees with states Q is the smallest set T such that $Q \cup \Sigma(T) \subseteq T$. We write T_Σ for $T_\Sigma(\emptyset)$. The mapping height $\mathrm{ht}(t) \colon T_\Sigma(Q) \to \mathbb{N}$ is defined by $\mathrm{ht}(q) = 0$ for all $q \in Q$ and $\mathrm{ht}(\sigma(t_1, \ldots, t_k)) = 1 + \max\{\mathrm{ht}(t_i) \mid i \in [k]\}$ for all $\sigma \in \Sigma_k$ and $t_1, \ldots, t_k \in T_\Sigma(Q)$. The subset $C_\Sigma(Q) \subseteq T_{\Sigma \cup \{\Box\}}(Q)$ of contexts contains all trees in which the special nullary symbol $\Box$ occurs exactly once. Again, we write C_Σ for $C_\Sigma(\emptyset)$. For all $c \in C_\Sigma(Q)$ and $t \in T_{\Sigma \cup \{\Box\}}(Q)$, we write $c[t]$ for the tree obtained from c by replacing $\Box$ by t. The tree t is a subtree of $c[t]$ for all contexts $c \in C_\Sigma(Q)$.

A (total bottom-up) deterministic (finite-state) tree automaton (DTA) [6,7] is a tuple $M = (Q, \Sigma, \delta, F)$ where Q is the finite, nonempty set of states, Σ is the ranked alphabet of input symbols, $\delta \colon \Sigma(Q) \to Q$ is the transition mapping, and $F \subseteq Q$ is the set of final states. The transition mapping δ extends to $\delta \colon T_\Sigma(Q) \to Q$ by $\delta(q) = q$ for all $q \in Q$ and

$$\delta\big(\sigma(t_1, \ldots, t_k)\big) = \delta\big(\sigma(\delta(t_1), \ldots, \delta(t_k))\big)$$

for all $\sigma \in \Sigma_k$ and $t_1, \ldots, t_k \in T_\Sigma(Q)$. We let $L(M)^q_{q'} = \{c \in C_\Sigma \mid \delta(c[q']) = q\}$ for all $q, q' \in Q$. Moreover, $L(M)_{q'} = \bigcup_{q \in F} L(M)^q_{q'}$ contains all (stateless) contexts that take q' into a final state, and $L(M)^q = \delta^{-1}(q) \cap T_\Sigma$ contains all (stateless) trees that are recognized in the state q. A state $q \in Q$ is a kernel (resp., preamble) state [2] if $L(M)^q$ is infinite (resp., finite). The sets Ker_M and Pre_M contain all kernel and preamble states, respectively. The DTA M recognizes the

tree language $L(M) = \bigcup_{q \in F} L(M)^q$, and all DTAs that recognize the same tree language are equivalent. A DTA is minimal if there exists no equivalent DTA with strictly fewer states. We can compute a minimal DTA that is equivalent to M using an adaptation [9] of HOPCROFT's algorithm [12], which runs in time $\mathcal{O}(|M| \cdot \log n)$ where $|M| = \sum_{k \in \mathbb{N}} k \cdot |\Sigma_k| \cdot n^k$ is the size of M and $n = |Q|$.

3 Structural Characterization of Hyper-Minimal DTAs

In this section we develop a characterization that points out the differences between different *hyper-minimal* DTAs for the same tree language. It will tell us which alternatives to consider when we search for an optimal hyper-minimal DTA, which commits the least number of errors. However, before we start, we recall the basic notions (e.g., *almost equivalence* and *hyper-minimality*).

Throughout the paper, let $M = (Q, \Sigma, \delta, F)$ and $N = (P, \Sigma, \mu, G)$ be minimal DTAs. Since we ultimately want to compare two DTAs, we introduce all basic notions for M and N. However, we often use them in the particular case that $M = N$. Two states $q \in Q$ and $p \in P$ are *almost equivalent* [13, Def. 1], written $q \sim p$ or $p \sim q$, if $E_{q,p} = L(M)_q \ominus L(N)_p$ is finite. We also say that q and p *disagree* on each element of $E_{q,p}$. If $E_{q,p} = \emptyset$, then q and p are *equivalent*, which is written $q \equiv p$ or $p \equiv q$. It is well-known [3, Sect. 1.5] that minimal DTAs do not have different, but equivalent states. Correspondingly, the DTAs M and N are *almost equivalent*, also written $M \sim N$, if $E = L(M) \ominus L(N)$ is finite.

Lemma 1 (see [13, Lm. 4]). *If $M \sim N$, then $\delta(t) \sim \mu(t)$ and $\delta(t') \equiv \mu(t')$ if $\mathrm{ht}(t') > |Q \times P|$ for all $t, t' \in T_\Sigma$.*[2]

Proof. The property $\delta(t) \sim \mu(t)$ is proven in [13, Lm. 4]. For the other property, we consider the product DTA $M \times N = (Q \times P, \Sigma, \delta \times \mu, F \times G)$, where

$$(\delta \times \mu)\big(\sigma(\langle q_1, p_1\rangle, \ldots, \langle q_k, p_k\rangle)\big) = \big\langle \delta(\sigma(q_1, \ldots, q_k)), \mu(\sigma(p_1, \ldots, p_k))\big\rangle$$

for all $\sigma \in \Sigma_k$ and $\langle q_1, p_1\rangle, \ldots, \langle q_k, p_k\rangle \in Q \times P$. Clearly, $(\delta \times \mu)(t) = \langle \delta(t), \mu(t)\rangle$ for all $t \in T_\Sigma$. If $\mathrm{ht}(t) > |Q \times P|$, then $(\delta \times \mu)(t)$ is a kernel state of $M \times N$ because the tree t can be pumped [6,7]. For the sake of a contradiction, suppose that $\delta(t) \not\equiv \mu(t)$; i.e., there exists $c \in E_{\delta(t),\mu(t)}$. Since $\langle \delta(t), \mu(t)\rangle$ is a kernel state of $M \times N$, there exist infinitely many $u \in T_\Sigma$ such that $\langle \delta(u), \mu(u)\rangle = \langle \delta(t), \mu(t)\rangle$. However, for each such tree u we have $c[u] \in E$, which contradicts $M \sim N$. □

The previous lemma shows that almost equivalent DTAs are in almost equivalent states after processing the same (stateless) tree. If the tree is tall, then they are even in equivalent states. Before we proceed with the comparison of almost equivalent DTAs, we recall another notion and a related result. A DTA is *hyper-minimal* if all almost equivalent DTAs have at least as many states.

Theorem 2 ([13, Thm. 7]). *A minimal DTA is hyper-minimal if and only if all pairs of different, but almost equivalent states consist only of kernel states.*

[2] If $M = N$, then $\mathrm{ht}(t') > |Q|$ is actually sufficient.

Now we can investigate how almost equivalent hyper-minimal DTAs differ. We extend each mapping $h\colon Q \to P$ to a mapping $h\colon T_\Sigma(Q) \to T_\Sigma(P)$ by $h(\sigma(t_1,\dots,t_k)) = \sigma(h(t_1),\dots,h(t_k))$ for every $\sigma \in \Sigma_k$ and $t_1,\dots,t_k \in T_\Sigma(Q)$. Such a mapping $h\colon Q \to P$ is a *transition homomorphism*[3] if $h(\delta(s)) = \mu(h(s))$ for every $s \in \Sigma(Q)$. Moreover, h is a DTA *homomorphism* if additionally $h(q) \in G$ if and only if $q \in F$. As usual, a bijective homomorphism is called *isomorphism*. Next, we show that two almost equivalent hyper-minimal DTAs have DTA-isomorphic kernels and transition-isomorphic preambles.

Theorem 3. *If $M \sim N$ and both M and N are hyper-minimal, then there exists a bijection $h\colon Q \to P$ such that*

1. *h is bijective on $\mathrm{Ker}_M \times \mathrm{Ker}_N$,*
2. *$h(q) \in G$ if and only if $q \in F$ for all $q \in \mathrm{Ker}_M$, and*
3. *$h(\delta(s)) = \mu(h(s))$ for every $s \in \Sigma(Q) - \{s \in \Sigma(\mathrm{Pre}_M) \mid \delta(s) \in \mathrm{Ker}_M\}$.*

Proof. Clearly, $|Q| = |P|$ since M and N are both hyper-minimal. For every $q \in Q$ select $t_q \in L(M)^q$ such that $\mathrm{ht}(t_q) > |Q \times P|$ whenever $q \in \mathrm{Ker}_M$.[4] We let $h\colon Q \to P$ be such that $h(q) = \mu(t_q)$ for every $q \in Q$, which immediately proves that $h(q) \in \mathrm{Ker}_N$ for all $q \in \mathrm{Ker}_M$ because $\mathrm{ht}(t_q) > |Q \times P|$. Moreover, for each $q \in \mathrm{Ker}_M$ the facts $M \sim N$ and $\mathrm{ht}(t_q) > |Q \times P|$ imply $q = \delta(t_q) \equiv \mu(t_q) = h(q)$ by Lemma 1. Thus, h is injective on $\mathrm{Ker}_M \times \mathrm{Ker}_N$ because $h(q_1) \equiv q_1 \not\equiv q_2 \equiv h(q_2)$ for all different $q_1, q_2 \in \mathrm{Ker}_M$.[5] Finally, for every $p \in \mathrm{Ker}_N$, select $u_p \in L(N)^p$ such that $\mathrm{ht}(u_p) > |Q \times P|$. Clearly, $\delta(u_p) \in \mathrm{Ker}_M$ and by Lemma 1 we obtain $p = \mu(u_p) \equiv \delta(u_p) \equiv h(\delta(u_p))$. Since N is minimal, we can conclude that $p = h(\delta(u_p))$, which shows that h is surjective on $\mathrm{Ker}_M \times \mathrm{Ker}_N$, thereby proving the first item.

Recall from the previous paragraph that $q \equiv h(q)$ for every $q \in \mathrm{Ker}_M$. Thus, $h(q) \in G$ if and only if $q \in F$, which proves the second item. For the third objective, let $s = \sigma(q_1,\dots,q_k) \in \Sigma(Q)$. Then

$$\begin{aligned}\delta\big(s\big) &= \delta\big(\sigma(t_{q_1},\dots,t_{q_k})\big) \overset{\dagger}{\sim} \mu\big(\sigma(t_{q_1},\dots,t_{q_k})\big) = \mu\big(\sigma(\mu(t_{q_1}),\dots,\mu(t_{q_k}))\big)\\ &= \mu\big(\sigma(h(q_1),\dots,h(q_k))\big) = \mu\big(h(s)\big)\ ,\end{aligned}$$

where the step marked † is due to Lemma 1. In the following, assume that $s \notin \Sigma(\mathrm{Pre}_M)$, which yields that there exists $i \in [k]$ such that $q_i \in \mathrm{Ker}_M$. Consequently, $\mathrm{ht}(\sigma(t_{q_1},\dots,t_{q_k})) > |Q \times P|$ by the selection of t_{q_i}, which can be used in Lemma 1 to show that the step marked † is actually equivalence ($\equiv$). We obtain that $\delta(s) \equiv \mu(h(s))$ and $\delta(s) \in \mathrm{Ker}_M$. Thus, $h(\delta(s)) \equiv \delta(s) \equiv \mu(h(s))$ by the argument in the previous paragraph. Since N is minimal, we can conclude that $h(\delta(s)) = \mu(h(s))$ as desired.

Before we prove the missing case, in which $s \in \Sigma(\mathrm{Pre}_M)$ with $\delta(s) \in \mathrm{Pre}_M$, we prove that h is bijective on $\mathrm{Pre}_M \times \mathrm{Pre}_N$, which automatically also proves

[3] or a homomorphism between the Σ-algebras [6,7] associated with M and N

[4] Such trees exist because each state is reachable (by hyper-minimality) and $L(M)^q$ is infinite for each kernel state q.

[5] We have $q_1 \not\equiv q_2$ because M is minimal.

that $h\colon Q \to P$ is a bijection. Since h is bijective on $\mathrm{Ker}_M \times \mathrm{Ker}_N$ by the proven first item, which yields $|\mathrm{Ker}_M| = |\mathrm{Ker}_N|$, and additionally $|Q| = |P|$, we obtain that $|\mathrm{Pre}_M| = |\mathrm{Pre}_N|$. Suppose that $h(q) \in \mathrm{Ker}_N$ for some $q \in \mathrm{Pre}_M$. Then $q \sim h(q) = \mu(u_{h(q)}) \sim \delta(u_{h(q)})$ with $\mathrm{ht}(u_{h(q)}) > |Q \times P|$ because $h(q) \in \mathrm{Ker}_N$, where the almost equivalences are due to Lemma 1. Clearly, $\delta(u_{h(q)})$ is a kernel state of M, which yields that $q \neq \delta(u_{h(q)})$. Together with $q \sim \delta(u_{h(q)})$ and $q \in \mathrm{Pre}_M$, these facts contradict the hyper-minimality of M by Theorem 2. It remains to prove that h is injective on $\mathrm{Pre}_M \times \mathrm{Pre}_N$, which due to $|\mathrm{Pre}_M| = |\mathrm{Pre}_N|$ also proves that h is surjective. For the sake of a contradiction, let $q_1, q_2 \in \mathrm{Pre}_M$ be such that $q_1 \neq q_2$ but $h(q_1) = h(q_2)$. Using Lemma 1 we obtain $q_1 \sim h(q_1) = h(q_2) \sim q_2$, which together with $q_1 \neq q_2$ contradicts the hyper-minimality of M by Theorem 2. Consequently, h is bijective on $\mathrm{Pre}_M \times \mathrm{Pre}_N$.

Now we return to the final missing objective, which requires us to show that $h(\delta(s)) = \mu(h(s))$ if $\delta(s) \in \mathrm{Pre}_M$. Recall that $\delta(s) \sim \mu(h(s))$ for all $s \in \Sigma(Q)$. Moreover, if $\delta(s) \in \mathrm{Pre}(M)$, then $h(\delta(s)) \in \mathrm{Pre}_N$ by the results of the previous paragraph and additionally $h(\delta(s)) \sim \delta(s) \sim \mu(h(s))$ by Lemma 1. Consequently, we have a preamble state $h(\delta(s))$ of N that is almost equivalent to $\mu(h(s))$. Since N is hyper-minimal, we have $h(\delta(s)) = \mu(h(s))$ by Theorem 2. □

The previous theorem states that two almost equivalent hyper-minimal DTAs are indeed very similar. They have a bijection between their states that preserves the distinction between preamble and kernel states. Moreover, via this bijection the two DTAs behave equally (besides acceptance) on the preamble states and isomorphically on the kernel states. Thus, two such DTAs can only differ in two aspects, which mirror the corresponding aspects for deterministic finite-state string automata [2]:

1. the finality (i.e., whether the state is final or not) of preamble states, and
2. transitions from exclusively preamble states to a kernel state.

4 Computing the Number of Errors

Now we can compute the number of errors made by a particular hyper-minimal DTA N that is almost equivalent to the reference DTA M. In addition, we show how to obtain a hyper-minimal DTA that commits the least number of errors among all almost equivalent hyper-minimal DTAs. More precisely, let N be hyper-minimal and almost equivalent to M, which itself is not necessarily hyper-minimal. Recall that $E = L(M) \ominus L(N)$ is the set of error trees. We partition E into $(E_p)_{p \in P}$, where $E_p = L(N)^p \cap E$ for every $p \in P$. In other words, we associate each error tree $t \in E$ with the state $\mu(t)$. In the following development, we distinguish errors associated to preamble and kernel states. Theorem 3 shows that the preamble-kernel error distinction is stable among all almost equivalent hyper-minimal DTAs.[6] Finally, [13, Sect. 4] shows how to obtain one hyper-minimal DTA N' that is almost equivalent to M. Roughly speaking, we identify

[6] An error associated to a preamble state of N can only be associated to a preamble state of another almost equivalent hyper-minimal DTA N'. Naturally, the error can be avoided in N', but the same error cannot be associated to a kernel state of N'.

the almost equivalence $\sim$ on M and then merge each preamble state that is almost equivalent to another state into this state. For every two different states $q, q' \in Q$, the DTA $\mathrm{merge}(M, q \to q')$ is $(Q - \{q\}, \Sigma, \delta', F - \{q\})$ where $\delta'(s) = q'$ if $\delta(s) = q$ and $\delta'(s) = \delta(s)$ otherwise for every $s \in \Sigma(Q - \{q\})$. We start with the errors E_p associated to a preamble state $p \in \mathrm{Pre}_N$. Since the preambles of N and N' are transition-isomorphic by Theorem 3, we can essentially compute with N' and only need to remember that the preamble states of N and N' can differ in finality.

Lemma 4 (see [13]). *Let $N' = \mathrm{merge}(M, q \to q')$ for some $q \sim q'$ with $[q] \subseteq \mathrm{Pre}_M$. Then $L(N')^{q'} = L(M)^q \cup L(M)^{q'}$. If N'' is the* DTA *returned by [13] and $B \in \{[q] \mid q \in Q\}$ is such that $B \subseteq \mathrm{Pre}_M$, Then $L(N'')^{q_B} = \bigcup_{q \in B} L(M)^q$.*

By Theorem 3 there exists a mapping $h\colon P' \to P$ such that N' and N are transition-isomorphic on their preambles via h. Together with Lemma 4 we thus have

$$L(N)^{h(q_B)} = L(N')^{q_B} = \bigcup_{q \in B} L(M)^q$$

for every $B \in \{[q] \mid q \in Q\}$ with $B \subseteq \mathrm{Pre}_M$.[7] Next, we demonstrate how to compute $a_q = |L(M)^q|$ for each state $q \in \mathrm{Pre}_M$.

Proposition 5. *For every $q \in \mathrm{Pre}_M$*

$$a_q = \sum_{\sigma(q_1, \dots, q_k) \in \delta^{-1}(q) \cap \Sigma(Q)} \Big(\prod_{i=1}^{k} a_{q_i} \Big) \ .$$

It is clear that the equations in Proposition 5 yield a recursive algorithm that runs in time $\mathcal{O}(|M|)$, if we do not recompute already computed values. With the help of Lemma 4 and Proposition 5, we can now compute the number of errors made due to the finality of preamble states. For every $q_B \in \mathrm{Pre}_{N'}$, we know that its block $B \subseteq \mathrm{Pre}_M$ consists of exclusively preamble states. Consequently, Lemma 4 can be applied to compute the number of errors associated to q_B.

Theorem 6. *For every $p \in \mathrm{Pre}_N$,*

$$|E_p| = \begin{cases} \sum_{q \in [\delta(u_p)] - F} a_q & \text{if } p \in G \\ \sum_{q \in [\delta(u_p)] \cap F} a_q & \text{otherwise,} \end{cases}$$

where $u_p \in L(N)^p$ is arbitrary.

Proof. The result follows from Theorem 3 and Lemma 4. □

Since $L(N)^p$ and $L(N)^{p'}$ are disjoint if $p \neq p'$, the total number of errors associated to preamble states is $\sum_{p \in \mathrm{Pre}_N} |E_p|$. To obtain the minimal number of errors, we select the finality of p such that E_p is minimal (see Algorithm 1).[8]

[7] The union is actually disjoint as $(L(M)^q)_{q \in Q}$ is a partition of T_Σ.

[8] Clearly, the DTA remains hyper-minimal and almost equivalent as only a finite number of errors is introduced by making p final or non-final.

Finally, we need to compute the number $\sum_{p \in \mathrm{Ker}_N} |E_p|$ of errors associated to kernel states. Recall that N' is the hyper-minimal DTA returned by [13]. Theorem 3 shows that the preambles of N and N' are transition-isomorphic and the kernels are DTA-isomorphic, but the transitions from exclusively preamble to kernel states are not covered in this characterization. As in the string case, we thus try to attribute errors to these preamble-to-kernel transitions because we know what happens before (transition-isomorphic on preamble) and what happens afterwards (DTA-isomorphic on the kernel). However, in the tree case this is complicated by the fact that such transitions can be taken several times in a single error tree as the next example demonstrates.

Example 7. Consider the DTA $M = (Q, \Sigma, \{q_\alpha\}, \delta)$ such that $Q = \{q_\alpha, q_\beta, q_\sigma\}$, $\Sigma = \{\alpha^{(0)}, \beta^{(0)}, \sigma^{(2)}\}$, and

$$\delta(\alpha) = q_\alpha \qquad \delta(\beta) = q_\beta \qquad \delta(\sigma(q_\beta, q_\beta)) = q_\sigma \qquad \delta(\sigma(q, q')) = q_\alpha$$

for all $(q, q') \in Q^2 - \{(q_\beta, q_\beta)\}$. Obviously, $L(M) = T_\Sigma - \{\beta, \sigma(\beta, \beta)\}$. An almost equivalent hyper-minimal DTA is $N = (\{\top\}, \Sigma, \{\top\}, \mu)$, where μ is such that $\mu^{-1}(\top) = \Sigma(\{\top\})$. Since $L(N) = T_\Sigma$, we have that $E = \{\beta, \sigma(\beta, \beta)\}$. However, when N processes the error tree $\sigma(\beta, \beta)$, then it will take two transitions (both times $\delta(\beta) = \top$) that switch from exclusively preamble states (no states in this case as α is nullary) to the kernel state $\top$.

We solve this problem by selecting the left-most occurrence of such a transition and disregarding all other occurrences to avoid counting duplicates. To this end, we first need to introduce positions. Let Δ be a ranked alphabet and $t \in T_\Delta(Q)$. The set $\mathrm{pos}(t) \subseteq \mathbb{N}^*$ of *positions in* t is defined by $\mathrm{pos}(q) = \{\varepsilon\}$ for every $q \in Q$ and $\mathrm{pos}(\sigma(t_1, \ldots, t_k)) = \{\varepsilon\} \cup \{iw \mid i \in [k], w \in \mathrm{pos}(t_i)\}$ for all $\sigma \in \Delta_k$ and $t_1, \ldots, t_k \in T_\Delta(Q)$. For every $w \in \mathrm{pos}(t)$, we write $t|_w$ for the subtree of t that is rooted in position w. A position $w_1 \in \mathrm{pos}(t)$ is *to the left of* another position $w_2 \in \mathrm{pos}(t)$, written $w_1 \sqsubset w_2$, if $w_1 \preceq w_2$ and $w_1 \not\leq w_2$, where $\preceq$ and $\leq$ are the lexicographic and prefix order on $\mathbb{N}^*$, respectively. In a context $c \in C_\Sigma(Q)$ the unique position of $\Box$ is denoted by $\mathrm{pos}_\Box(c)$. Now we can define the set LC of *left-most contexts*, which have no subtree to the left of (the occurrence of) $\Box$ that is recognized (by M) in a state that is almost equivalent to a kernel state, as follows:

$$\mathrm{LC} = \{c \in C_\Sigma \mid \forall w \in \mathrm{pos}(c) \colon w \sqsubset \mathrm{pos}_\Box(c) \text{ implies } [\delta(c|_w)] \subseteq \mathrm{Pre}_M\} \ .$$

With the help of the set LC we can now make the error attribution more formal. We already know that each remaining error tree has a special transition that switches from exclusively preamble states to a kernel state. Moreover, we will now prove that every such error tree decomposes uniquely into a context of LC, which encodes the part of the tree that is processed after a special transition, and a tree that uses a special transition at the root. Due to the definition of LC, we know that the decomposition selects exactly the left-most occurrence of a special transition.

Lemma 8. *Every error tree $t \in E_p$ with $p \in \operatorname{Ker}_N$ decomposes uniquely via $t = c[u]$ into a left-most context $c \in \mathrm{LC}$ and $u \in T_\Sigma$ such that $\mu(u) \in \operatorname{Ker}_N$, but $\mu(u|_w) \in \operatorname{Pre}_N$ for all $w \in \operatorname{pos}(u)$ with $w \neq \varepsilon$.*

Proof. Since $\mu(t) \in \operatorname{Ker}_N$, there must exist positions $w \in \operatorname{pos}(t)$ such that $\mu(t|_w) \in \operatorname{Ker}_N$ but $\mu(t|_{wv}) \in \operatorname{Pre}_N$ for all $wv \in \operatorname{pos}(t)$ with $v \neq \varepsilon$. Let w be the left-most such position (i.e., a minimal such position with respect to $\sqsubset$). It remains to show that $t[\Box]_w \in \mathrm{LC}$, where $t[\Box]_w$ denotes the context obtained from t by replacing the subtree rooted at w by $\Box$. By the selection of w, all positions $v \sqsubset w$ are such that $\mu(t|_v) \in \operatorname{Pre}_N$. Thus, $[\delta(t|_v)] \subseteq \operatorname{Pre}_M$ by Theorem 3 and the earlier discussion, which proves that $t[\Box]_w \in \mathrm{LC}$. Thus, we obtain the suitable decomposition $t = c[t|_w]$ with $c = t[\Box]_w$. The uniqueness is also easy to show as each other suitable position w' obeys $w \sqsubset w'$ and $\mu(t|_w) \in \operatorname{Ker}_N$, which by Lemma 1 yields that $\delta(t|_w) \sim q$ for some $q \in \operatorname{Ker}_M$. Consequently, $t[\Box]_{w'} \notin \mathrm{LC}$ for all other suitable positions $w' \neq w$. □

The decomposition $t = c[u]$ already hints at the next steps. We can compute $\delta(u)$ and $\mu(u)$, for which we know that $\delta(u) \sim \mu(u)$ by Lemma 1. The error is then made between those two states, so $c \in E_{\delta(u),\mu(u)} = L(M)_{\delta(u)} \ominus L(N)_{\mu(u)}$ is an error context of LC. To make the computation even simpler, we observe that $\mu(u) \in \operatorname{Ker}_N$, which with the help of Theorem 3 yields that there exists a state $q \in Q$ such that $q \equiv \mu(u)$. Consequently, it is sufficient to compute $E_{\delta(u),q}$ for all $\delta(u) \sim q$. In fact, for all $q \sim q'$ we know that $E_{q,q'}$ is finite, but we need the exact cardinality $d(q, q')$ of the subset $E_{q,q'} \cap \mathrm{LC}$. More exactly, for all $q \sim q'$, let $d(q, q') = |E_{q,q'} \cap \mathrm{LC}|$ and

$$C_M = C_\Sigma(Q) \cap \Sigma(Q \cup \{\Box\}) \qquad \text{and} \qquad \overline{C}_M = C_M \cap C_\Sigma(\operatorname{Pre}_M) \ ,$$

of which the elements are called *transition* and *preamble transition contexts*, respectively. To compute d, we adjust the straightforward counting procedure [15].

Lemma 9. *For all $q \sim q'$ we have $d(q, q) = 0$ and*

$$d\big(q, q'\big) = \Big(\sum_{\substack{c \in \overline{C}_M \\ c = \sigma(q_1, \ldots, q_i, \Box, q_{i+1}, \ldots, q_k) \\ [q_1], \ldots, [q_i] \subseteq \operatorname{Pre}_M}} a_{q_1} \cdot \ldots \cdot a_{q_k} \cdot d\big(\delta(c[q]), \delta(c[q'])\big) \Big) + \begin{cases} 1 & \text{if } q \in F \text{ xor } q' \in F \\ 0 & \text{otherwise.} \end{cases}$$

Proof. The first equation is trivial and the second equation straightforwardly formalizes $|E_{q,q'}|$, but only counts the error contexts of LC. More precisely, the final summand checks whether $\Box \in \mathrm{LC}$ is in the difference $E_{q,q'}$. Every other difference context $c'' = c'[\bar{c}] \in E_{q,q'}$ consists of (i) a context $\bar{c}$ obtained from a transition context $c = \sigma(q_1, \ldots, q_i, \Box, q_{i+1}, \ldots, q_k)$ of $\overline{C}_M$ by replacing the states $q_1, \ldots, q_k \in Q$ by $t_1 \in L(M)^{q_1}, \ldots, t_k \in L(M)^{q_k}$, respectively, which yields the factors $a_{q_1}, \ldots, a_{q_k}$, and (ii) an error context c' for the states $\delta(c[q])$ and $\delta(c[q'])$, which yields the factor $d(\delta(c[q]), \delta(c[q']))$. We can immediately restrict ourselves to preamble transition contexts because $\delta(c[q]) = \delta(c[q'])$ by [13, Prop. 18],

which yields that $d(\delta(c[q]), \delta(c[q'])) = 0$, for all $c \in C_M - \overline{C}_M$. Moreover, if the states $q_1, \ldots, q_i$ to the left of $\Box$ are not such that $[q_1], \ldots, [q_i] \subseteq \mathrm{Pre}_M$, then the context c'' is not in LC and thus discarded. $\square$

Since we now have a recursive procedure to compute d, let us quickly analyse its time complexity. The analysis is based on the idea that entries in d are never recomputed once they have been computed once.

Corollary 10 (of Prop. 5 and Lm. 9). *For all $q \sim q'$ we can compute $d(q, q')$ in time $\mathcal{O}(m \cdot n)$ where $m = |M|$ and $n = |Q|$.*

Proof. We can trivially compute all a_q with $q \in \mathrm{Pre}_M$ in time $\mathcal{O}(m)$ as already mentioned, and we can compute each entry in d in time $\mathcal{O}(\frac{m}{n})$ without the time needed to compute the recursive calls because there are $\sum_{k \geq 1} k \cdot |\Sigma_k| \cdot |Q|^{k-1}$ transition contexts.[9] Since there are at most n^2 entries in d, we obtain the stated time-bound. $\square$

Thus, we can now identify and count the errors caused in kernel states of M. To this end, we look at all the transitions that switch from exclusively preamble to kernel states and compute the number of errors induced by this transition. Let $s = \sigma(p_1, \ldots, p_k) \in \Sigma(\mathrm{Pre}_N)$ and $\mu(s) \in \mathrm{Ker}_N$ be such a transition. The set E_s of errors caused by this transition s is

$$E_s = E \cap \{c[\sigma(t_1, \ldots, t_k)] \mid c \in L(N)_{\mu(s)} \cap \mathrm{LC}, \forall i \in [k] \colon t_i \in L(N)^{p_i}\} \ ,$$

which contains all errors that use the transition s as the left-most special transition.

Lemma 11. *For every $s = \sigma(p_1, \ldots, p_k) \in \Sigma(\mathrm{Pre}_N)$ with $\mu(s) \in \mathrm{Ker}_N$*

$$|E_s| = e_{s,q} = \sum_{q_1 \in [\delta(u_{p_1})], \ldots, q_k \in [\delta(u_{p_k})]} a_{q_1} \cdot \ldots \cdot a_{q_k} \cdot d\big(\delta(\sigma(q_1, \ldots, q_k)),\, q\big) \ ,$$

where $u_p \in L(N)^p$ for every $p \in \mathrm{Pre}_N$ and $q \in \mathrm{Ker}_M$ is such that $q \equiv \mu(s)$.

Proof. Let $N' = (P', \Sigma, \mu', G')$ be the DTA returned by [13], and for every $i \in [k]$ let $p'_i = \mu'(u_{p_i})$. Then $L(N)^{p_i} = L(N')^{p'_i} = \bigcup_{q_i \in [\delta(u_{p_i})]} L(M)^{q_i}$ by Theorem 3 and Lemma 4. Moreover, $L(N)_{\mu(s)} = L(M)_q$ by assumption. Together with these statements, the equation is a straightforward implementation of the definition of E_s. $\square$

By Lemma 8 the sets E_s for suitable $s \in \Sigma(\mathrm{Pre}_N)$ are pairwise disjoint, so the errors just add up. In addition, any state $p \in P$ such that $p \sim \mu(s)$ is a valid transition target, so to optimize the errors, we can simply select the transition target $p \in P$ with $p \sim \mu(s)$ that minimizes the number of caused errors. In summary, this yields our main theorem (and Algorithm 1).

[9] This actually needs another trick. Given a transition $\sigma(q_1, \ldots, q_k) \in \Sigma(Q)$ we obtain k transition contexts $c_1, \ldots, c_k$ by replacing in turn each state $q_1, \ldots, q_k$ by $\Box$. To avoid a multiplication effort of $\mathcal{O}(k)$ we once compute $a_{q_1} \cdot \ldots \cdot a_{q_k}$ and then compute the value for the context c_i by dividing this product by the value a_{q_i}.

Algorithm 1 Optimal choice of preamble-to-kernel transitions.

Require: a minimal DTA M, its almost equivalence $\sim \subseteq Q \times Q$, and an almost equivalent hyper-minimal DTA N
Return: an almost equivalent hyper-minimal DTA N minimizing $|L(M) \ominus L(N)|$

select $u_p \in L(N)^p$ for all $p \in \operatorname{Pre}_N$
$G \leftarrow \{p \in P \mid \sum_{q \in [\delta(u_p)]-F} a_q < \sum_{q \in [\delta(u_p)] \cap F} a_q\}$
for all $s \in \Sigma(\operatorname{Pre}_N)$ with $\mu(s) \in \operatorname{Ker}_N$ **do**
  select $p \in P$ such that $p \equiv \arg\min_q \big(e'_{s,q} \mid q \in \operatorname{Ker}_M, q \sim \mu(s)\big)$
  $\mu(s) \leftarrow p$ // reroute transition
return N

Theorem 12. *Let $m = |M|$ and $n = |Q|$. For every hyper-minimal* DTA *N that is almost equivalent to M we can determine $|L(M) \ominus L(N)|$ in time $\mathcal{O}(m \cdot n)$. Moreover, we can compute a hyper-minimal* DTA *N' that minimizes the number $|L(M) \ominus L(N')|$ of errors in time $\mathcal{O}(m \cdot n)$.*

Future Work

Recently, [10] showed results in the string case for other regular languages of allowed differences. These should translate trivially to tree automata. The difference in the number of errors between the optimal dta and the worst dta can be exponential, so the optimization can avoid a large number of errors. A practical evaluation for dta remains future work, but a simple experiment was already conducted in [15, Sect. 6] for the string case. A reviewer suggested to consider the sum of the error tree sizes instead of the simple count of error trees, but the optimization of that criterion seems closely related to bin packing already in the acyclic case (i.e., the case where the automaton has no kernel states), but the details should still be worked out. In addition, the reviewer suggested to consider those languages $L \subseteq T_\Sigma$, for which the minimal dta is hyper-minimal and optimal and in addition no other dta of strictly smaller size recognizes a tree language that is almost equivalent to L. Clearly, such tree languages exist (e.g., T_Σ), but the author is unaware of the particular properties of those languages.

References

1. Andrew Badr. Hyper-minimization in $O(n^2)$. *Int. J. Found. Comput. Sci.*, 20(4):735–746, 2009.
2. Andrew Badr, Viliam Geffert, and Ian Shipman. Hyper-minimizing minimized deterministic finite state automata. *RAIRO Theor. Inf. Appl.*, 43(1):69–94, 2009.
3. Hubert Comon, Max Dauchet, Rémi Gilleron, Christof Löding, Florent Jacquemard, Denis Lugiez, Sophie Tison, and Marc Tommasi. Tree automata: Techniques and applications. `http://tata.gforge.inria.fr/`, 2007. release October, 12.

4. Paweł Gawrychowski and Artur Jeż. Hyper-minimisation made efficient. In *Proc. 34th Int. Symp. Mathematical Foundations of Computer Science*, volume 5734 of LNCS, pages 356–368. Springer, 2009.
5. Paweł Gawrychowski, Artur Jeż, and Andreas Maletti. On minimising automata with errors. In *Proc. 36th Int. Conf. Mathematical Foundations of Computer Science*, volume 6907 of LNCS, pages 327–338. Springer, 2011.
6. Ferenc Gécseg and Magnus Steinby. *Tree Automata*. Akadémiai Kiadó, Budapest, 1984.
7. Ferenc Gécseg and Magnus Steinby. Tree languages. In Grzegorz Rozenberg and Arto Salomaa, editors, *Handbook of Formal Languages*, volume 3, chapter 1, pages 1–68. Springer, 1997.
8. Gregor Gramlich and Georg Schnitger. Minimizing nfa's and regular expressions. *J. Comput. System Sci.*, 73(6):908–923, 2007.
9. Johanna Högberg, Andreas Maletti, and Jonathan May. Backward and forward bisimulation minimization of tree automata. *Theoret. Comput. Sci.*, 410(37):3539–3552, 2009.
10. Markus Holzer and Sebastian Jakobi. From equivalence to almost-equivalence, and beyond — minimizing automata with errors. In *Proc. 16th Int. Conf. Developments in Language Theory*, volume 7410 of LNCS, pages 190–201. Springer, 2012.
11. Markus Holzer and Andreas Maletti. An $n \log n$ algorithm for hyper-minimizing a (minimized) deterministic automaton. *Theoret. Comput. Sci.*, 411(38–39):3404–3413, 2010.
12. John E. Hopcroft. An $n \log n$ algorithm for minimizing states in a finite automaton. In Zvi Kohavi and Azaria Paz, editors, *Theory of Machines and Computations*, pages 189–196. Academic Press, 1971.
13. Artur Jeż and Andreas Maletti. Hyper-minimization for deterministic tree automata. In *Proc. 17th Int. Conf. Implementation and Application of Automata*, volume 7381 of LNCS, pages 217–228. Springer, 2012.
14. Andreas Maletti and Daniel Quernheim. Hyper-minimisation of deterministic weighted finite automata over semifields. In *Proc. 13th Int. Conf. Automata and Formal Languages*, pages 285–299. Nyíregyháza College, 2011.
15. Andreas Maletti and Daniel Quernheim. Optimal hyper-minimization. *Int. J. Found. Comput. Sci.*, 22(8):1877–1891, 2011.
16. Mehryar Mohri. Finite-state transducers in language and speech processing. *Comput. Linguist.*, 23(2):269–311, 1997.
17. Maurice Nivat and Andreas Podelski. *Tree Automata and Languages*. Studies in Computer Science and Artificial Intelligence. North-Holland, 1992.
18. Slav Petrov, Leon Barrett, Romain Thibaux, and Dan Klein. Learning accurate, compact, and interpretable tree annotation. In *Proc. 44th Ann. Meeting of the ACL*, pages 433–440. Association for Computational Linguistics, 2006.
19. Jacques Sakarovitch. Rational and recognisable power series. In Manfred Droste, Werner Kuich, and Heiko Vogler, editors, *Handbook of Weighted Automata*, EATCS Monographs on Theoretical Computer Science, chapter IV, pages 105–174. Springer, 2009.
20. Sven Schewe. Beyond hyper-minimisation — minimising DBAs and DPAs is NP-complete. In *Proc. 30th Int. Conf. Foundations of Software Technology and Theoretical Computer Science*, volume 8 of *LIPIcs*, pages 400–411. Schloss Dagstuhl, 2010.
21. Sheng Yu. Regular languages. In Grzegorz Rozenberg and Arto Salomaa, editors, *Handbook of Formal Languages*, volume 1, chapter 2, pages 41–110. Springer, 1997.