On the Translation of Automata to Linear Temporal Logic

Udi Boker, Karoliina Lehtinen, Salomon Sickert

On the Translation of Automata to Linear Temporal Logic*

Udi Boker[1] Karoliina Lehtinen[2] Salomon Sickert[3]**

[1] Reichman University, Herzliya, Israel
[2] CNRS, Aix-Marseille University, LIS, Marseille, France
[3] The Hebrew University, Jerusalem, Israel

Abstract While the complexity of translating future linear temporal logic (LTL) into automata on infinite words is well-understood, the size increase involved in turning automata back to LTL is not. In particular, there is no known elementary bound on the complexity of translating deterministic ω-regular automata to LTL.
Our first contribution consists of tight bounds for LTL over a unary alphabet: alternating, nondeterministic and deterministic automata can be exactly exponentially, quadratically and linearly more succinct, respectively, than any equivalent LTL formula. Our main contribution consists of a translation of general counter-free deterministic ω-regular automata into LTL formulas of double exponential temporal-nesting depth and triple exponential length, using an intermediate Krohn-Rhodes cascade decomposition of the automaton. To our knowledge, this is the first elementary bound on this translation. Furthermore, our translation preserves the acceptance condition of the automaton in the sense that it turns a looping, weak, Büchi, coBüchi or Muller automaton into a formula that belongs to the matching class of the syntactic future hierarchy. In particular, it can be used to translate an LTL formula recognising a safety language to a formula belonging to the safety fragment of LTL (over both finite and infinite words).

Keywords: Linear temporal logic · Automata · Cascade decomposition

1 Introduction

Linear Temporal Logic with only future temporal operators (from here on LTL) and ω-regular automata, whether deterministic, nondeterministic or alternating, are both well-established formalisms to describe properties of infinite-word languages. LTL is popular in formal verification and synthesis due to its simple syntax and semantics. Yet, while properties might be convenient to define in LTL, most verification and synthesis algorithms eventually compile LTL formulas into ω-regular automata. The expressiveness of both these key formalisms, as

* This is the full version of a chapter with the same title that appears in the FoSSaCS 2022 conference proceedings.

** Salomon Sickert is supported by the Deutsche Forschungsgemeinschaft (DFG) under project number 436811179.

well as translations from LTL to automata of various types, are well understood. Here, we consider the converse translations, which, in comparison, have received less attention: up till now, no elementary upper bound on the size blow-up of going from automata to LTL was known.

Regarding expressive power, deterministic Muller automata, nondeterministic Büchi automata, and weak alternating automata recognise all ω-regular languages [20,39]. LTL-definable languages (surveyed in [12]) are a strict subset thereof, also defined by first-order logic, star-free regular expressions, aperiodic monoids, counter-free automata, and very weak alternating automata. As for succinctness, nondeterministic and alternating automata can be exponentially and double-exponentially more succinct than deterministic automata, respectively. Determinisation in particular has precise bounds [31,34,23,35,11,3].

The succinctness of various representations of LTL-definable languages is less clear: effective translations between the different models are far from straightforward, and their complexity is sometimes uncertain. In particular, to the best of our knowledge, up to now there has been no elementary bound even on the translation of deterministic counter-free automata, arguably the simplest automata model for this class of languages, into LTL formulas. (Considering LTL with both future and past temporal operators, there is a double-exponential upper bound on the length of the formula [25][4].) The complexity of obtaining a deterministic counter-free automaton from a nondeterministic one is also, to the best of our knowledge, open.

We study the complexity of translating automata to LTL (equivalently, to very weak alternating automata), considering formula length, size, and nesting depth of temporal operators.

We begin (Section 3), as a warm-up, with the unary alphabet case on finite words. We show that the size-blow up involved in translating deterministic, non-deterministic and alternating automata to LTL, when possible, is linear, quadratic and exponential, respectively, and these bounds are tight. In contrast, going from LTL to alternating, nondeterministic and deterministic automata is linear, exponential and double-exponential, respectively [32,40,18].

The case of non-unary alphabets is much more difficult. We provide a translation of counter-free deterministic ω-regular automata (with any acceptance condition) into LTL formulas with double exponential depth and triple exponential length. Our translation uses an intermediate Krohn-Rhodes *reset cascade decomposition* (wreath product) of deterministic automata, which is a deterministic automaton built from simple components.

Our main technical contribution consists of a translation of a reset cascade into an LTL formula of depth linear and length singly exponential in the number of cascade configurations. Combining this with Eilenberg's Holonomy translation of a semigroup into a cascade [13, Corollary II.7.2] and Pnueli and Maler's adaptation of it to automata [25, Theorem 3] (see Remark 1), we obtain a translation of counter-free deterministic ω-regular automata into LTL formulas of double exponential depth and triple exponential length. Our construction preserves the

[4] See Remark 1 on whether the upper bound in [25] is single or double exponential.

acceptance condition of the automaton in the sense that it turns a Büchi-looping, coBüchi-looping, weak, Büchi or coBüchi automaton into a formula that belongs to the matching class of the syntactic future hierarchy (see Definition 1 and [7]).

Related work

Finite words. While LTL is usually interpreted over infinite words, it also admits finite-word semantics that coincide with the finite word version of the other equivalent formalisms. The equivalence between FO and star-free languages on finite words is due to McNaughton and Papert [30]. Cohen, Perrin and Pin [9] used the Krohn-Rhodes decomposition to characterise the expressive power of LTL with only **X** and **F** (eventually), but do not provide bounds on the size trade-off between the different models. Wilke [41] gives a double-exponential translation from counter-free DFA to LTL. More recently, Bojańczyk provided an algebraically flavoured adaptation of Wilke's proof [2, Section 2.2.2].

Infinite words. With substantial effort over several decades, the above techniques have been extended to infinite words using intricate tools with opaque complexities. Ladner [21] and Thomas [37,38] for example extended the equivalence of star-free regular expressions and FO to infinite words, while the ω-extension of the equivalence with aperiodic languages is due to Perrin [33]. The correspondence with LTL is due to Kamp [17] and Gabbay, Pnueli, Shelah and Stavi [15]. Diekert and Gastin's survey [12] provides an algebraic translation into LTL via ω-monoids while Cohen-Chesnot gives a direct algebraic proof of the equivalence of star-free ω-regular expressions and LTL [10]. Wilke takes an automata-theoretic approach, using backward deterministic automata [42,43]. However, none of the above address the complexity of the transformations. Zuck's dissertation [45] gives a translation of star-free regular expressions into LTL, with at least non-elementary complexity. Subsequently, Chang, Mana and Pneuli [7] use Zuck's results to show that the levels of their hierarchy of future temporal properties coincide with syntactic fragments of LTL. Sickert and Esparza [36] gave an exponential translation of any LTL formula into level Δ_2 of this hierarchy.

2 Preliminaries

Languages. An alphabet Σ, of size $|\Sigma|$, is a finite set of letters. Σ^*, Σ^+, and Σ^ω denote the sets of finite, nonempty finite, and infinite words over Σ, respectively. A language of finite or infinite words is a subset of Σ^* or Σ^ω, respectively. We write $[i..j]$ and $[i..j)$, with integers $i \leq j$, for the sets $\{i, i+1, \ldots, j\}$ and $\{i, i+1, \ldots, j-1\}$, respectively. For a word $w = \sigma_0 \cdot \sigma_1 \cdots$, we write $|w|$ for its length (∞ if w is infinite), $w[i]$ for σ_i, $w_{[i..j]}$ and $w_{[i..j)}$ for its corresponding infixes ($w_{[i..i)}$ is the empty word), and $w_{[i..]}$ for its (finite or infinite) suffix $\sigma_i \cdot \sigma_{i+1} \cdots$.

Linear Temporal Logic (LTL). Let AP be a finite set of atomic propositions. LTL formulas are constructed from the constant **true**, atomic propositions $a \in AP$, the connectives $\neg$ (negation) and $\wedge$ (and), and the temporal operators $\mathbf{U}$ (until) and $\mathbf{X}$ (next). Their semantics are given by a satisfiability relation $\models$ between finite or infinite words $w \in (2^{AP})^+ \cup (2^{AP})^\omega$, and a formula φ inductively as follows:

$$
\begin{array}{lll}
w \models \mathbf{true} & & w \models a \quad \textit{iff} \quad a \in w[0] \\
w \models \neg\varphi \quad \textit{iff} \quad w \not\models \varphi & & w \models \varphi \wedge \psi \quad \textit{iff} \quad w \models \varphi \textit{ and } w \models \psi \\
w \models \mathbf{X}\varphi \quad \textit{iff} \quad |w| > 1 \textit{ and } w_{[1..]} \models \varphi & & \\
w \models \varphi \mathbf{U} \psi \quad \textit{iff} \quad \exists i \in [0..|w|).\ w_{[i..]} \models \psi \textit{ and } \forall j \in [0..i).\ w_{[j..]} \models \varphi & &
\end{array}
$$

We also use the common shortcuts $\mathbf{false} \coloneqq \neg\mathbf{true}$, $\varphi \vee \psi \coloneqq \neg((\neg\varphi) \wedge (\neg\psi))$, $\mathbf{F}\varphi \coloneqq \mathbf{true}\mathbf{U}\varphi$, $\mathbf{G}\varphi \coloneqq \neg\mathbf{F}\neg\varphi$, and $\psi_1 \mathbf{R} \psi_2 \coloneqq \neg(\neg\psi_1)\mathbf{U}(\neg\psi_2)$. The language of finite words of φ is $L^{<\omega}(\varphi) \coloneqq \{w \in (2^{AP})^+ \mid w \models \varphi\}$, and the language of infinite words is $L(\varphi) \coloneqq \{w \in (2^{AP})^\omega \mid w \models \varphi\}$. Note that we omit the "$< \omega$" superscript if it is clear from the context which set is used. The *length* $|\varphi|$ of φ is the number of nodes in its syntax tree, the *size* of φ is the number of nodes in a DAG representing this syntax tree, and its *temporal nesting depth*, denoted by $\mathsf{depth}(\varphi)$, is defined by: $\mathsf{depth}(\mathbf{true}) = 0$; $\mathsf{depth}(a) = 0$ for an atomic proposition $a \in AP$; $\mathsf{depth}(\neg\psi) = \mathsf{depth}(\psi)$; $\mathsf{depth}(\psi_1 \wedge \psi_2) = \max(\mathsf{depth}(\psi_1), \mathsf{depth}(\psi_2))$; $\mathsf{depth}(\mathbf{X}\psi) = \mathsf{depth}(\psi) + 1$; and $\mathsf{depth}(\psi_1 \mathbf{U} \psi_2) = \max(\mathsf{depth}(\psi_1), \mathsf{depth}(\psi_2)) + 1$. Chang, Manna, and Pnueli define in [7] a syntactic hierarchy for LTL formulas (over infinite words):

Definition 1 (LTL Syntactic future hierarchy [7] [5]).

- $\Sigma_0 = \Pi_0 = \Delta_0$ *is the least set containing all atomic propositions and their negations, and is closed under the application of conjunction and disjunction.*
- Σ_{i+1} *is the least set containing* Π_i *and negated formulas of* Π_{i+1} *closed under the application of conjunction, disjunction, and the* $\mathbf{X}$ *and* $\mathbf{U}$ *operators.*
- Π_{i+1} *is the least set containing* Σ_i *and negated formulas of* Σ_{i+1} *closed under the application of conjunction, disjunction, and the* $\mathbf{X}$ *and* $\mathbf{R}$ *operators.*
- Δ_{i+1} *is the least set containing* Σ_{i+1} *and* Π_{i+1} *that is closed under the application of conjunction, disjunction, and negation.*

Σ_1 is referred to as *syntactic co-safety* formulas, Π_1 as *syntactic safety* formulas.

Automata. A *deterministic semiautomaton* is a tuple $\mathcal{D} = (\Sigma, Q, \delta)$, where Σ is an alphabet; Q is a finite nonempty set of states; and $\delta\colon Q \times \Sigma \to Q$ is a transition function and we extend it to finite words in the usual way. A *path* of $\mathcal{D}$ on a word $w = \sigma_0 \cdot \sigma_1 \cdots$ is a sequence of states $q_0, q_1, \ldots$, such that for every $i < |w|$, we have $\delta(q_i, \sigma_i) = q_{i+1}$.

It is a *reset* semiautomaton if for every letter $\sigma \in \Sigma$, either i) for every state $q \in Q$ we have $\delta(q, \sigma) = q$, or ii) there exists a state $q' \in Q$, such that for every state $q \in Q$ we have $\delta(q, \sigma) = q'$.

[5] This extends [5,36] with negation, which can be removed via negation normal form.

It is *counter free* if for every state $q \in Q$, finite word $u \in \Sigma^+$, and number $n \in \mathbb{N} \setminus \{0\}$, there is a self loop of q on u^n iff there is a self loop of q on u.

A *deterministic automaton* is a tuple $\mathcal{D} = (\Sigma, Q, \iota, \delta, \alpha)$, where (Σ, Q, δ) is a deterministic semiautomaton, $\iota \in Q$ is an initial state; and α is some acceptance condition, as detailed below. A run of $\mathcal{D}$ on a word w is a path of $\mathcal{D}$ on w that starts in ι. It is a reset or counter-free automaton if its semiautomaton is.

The *acceptance condition* of an automaton on finite words is a set $F \subseteq Q$; a run is accepting if it ends in a state $q \in F$. The *acceptance condition* of an *ω-regular automaton*, on infinite words, is defined with respect to the set $inf(r)$ of states visited infinitely often along a run r. We define below several acceptance conditions that we use in the sequel; for other conditions, see, for example, [3].

The *Muller* condition is a set $\alpha = \{M_1, \ldots, M_k\}$ of sets $M_i \subseteq Q$ of states, and a run r is accepting if there exists a set M_i, such that $M_i = inf(r)$. The *Rabin* condition is a set $\alpha = \{(G_1, B_1), \ldots, (G_k, B_k)\}$ of pairs of sets of states, and r is accepting if there exists a pair (G_i, B_i), such that $G_i \cap inf(r) \neq \emptyset$ and $B_i \cap inf(r) = \emptyset$. The *Büchi* (resp. *coBüchi*) condition is a set $\alpha \subseteq Q$ of states, and r is accepting if $\alpha \cap inf(r) \neq \emptyset$ (resp. $\alpha \cap inf(r) = \emptyset$). A *weak* automaton is a Büchi automaton, in which every strongly connected component (SCC) contains only states in α or only states out of α. A *looping* automaton is a Büchi or coBüchi automaton, where all states are in α, except for a single sink state.

Deterministic automata of the above types correspond to the hierarchy of temporal properties [27]: Looping-Büchi, looping-coBüchi, weak, Büchi, coBüchi, and Rabin/Muller deterministic automata define respectively safety, guarantee (co-safety), obligation, recurrence, persistence, and reactivity languages. If the language is also LTL-definable, then there exists an equivalent LTL formula in Π_1, Σ_1, Δ_1, Π_2, Σ_2, and Δ_2, respectively [7]. Every deterministic ω-regular automaton is equivalent to deterministic Muller and Rabin automata, where the Muller (but not always Rabin) one can be defined on the same semiautomaton.

Nondeterministic and *alternating* automata (to which we only refer in Section 3, on finite words over a unary alphabet) extend deterministic automata by having a transition function $\delta\colon Q \times \Sigma \to 2^Q$ and $\delta\colon Q \times \Sigma \to$ (positive Boolean formulas over Q), respectively. (See, for example, [6] for formal definitions.)

3 Unary Alphabet

Kupferman, Ta-Shma and Vardi [19] compared the succinctness of different automata models when *counting*, that is, recognising the singleton language $\{a^k\}$ for some k over the singleton alphabet $\{a\}$. For the succinctness gap between automata and LTL, we study the task of recognising arbitrary languages over the unary alphabet, which can be seen as sets of integers, rather than a single integer.

For a unary alphabet, since there is only one infinite word, only languages on finite words are interesting. We thus consider LTL formulas over (no) atomic propositions $AP = \emptyset$, and automata on finite unary words over the corresponding alphabet $\Sigma = 2^{AP} = \{\emptyset\}$, where we use the shorthand $a = \emptyset$. The size of a deterministic automaton is the number of its states, of a nondeterministic

automaton the number of its transitions, and of an alternating automaton the number of subformulas in its transition function.

We show that the size blow-up involved in translating deterministic, nondeterministic, and alternating automata to LTL, when possible, is linear, quadratic, and exponential, respectively.

In our analysis, we shall use the following folklore theorem, which extends Wolper's Theorem [44]. The proof is given in Appendix A.1.

Proposition 1 (Extended Wolper's theorem, Folklore). *Consider an LTL formula φ with $\mathsf{depth}(\varphi) = n$ over the atomic propositions AP, and let $\Sigma = 2^{AP}$. Then for every words $u \in \Sigma^*$, $v \in \Sigma^+$ and $t \in \Sigma^\omega$, and numbers $i, j > n$, φ has the same truth value on the words $(uv^i t)$ and $(uv^j t)$.*

We use this to establish that unary LTL describes only finite and co-finite properties, and that there is a tight relation between the depth of LTL formulas and the length of words above which they are all in or all out of the language.

Proposition 2. *Given an LTL formula φ with $\mathsf{depth}(\varphi) = n$ on finite words over the unary alphabet $\{a\}$, $a^i \in L(\varphi)$ for all $i > n$ or $a^i \notin L(\varphi)$ for all $i > n$.*

Proposition 3. *Consider a language $L \subseteq \{a\}^+$ that agrees on all words of length over n, that is, has the same truth value on all such words. Then there is an LTL formula of size in $O(n)$ with language L.*

We now establish the trade-off between LTL and alternating automata (AFA) over unary alphabets. AFA are closed under (linear) complementation, so we use a pumping argument to bound the length after which all words have the same truth value, giving an upper bound on the LTL formula.

Lemma 1. *Every alternating automaton with n states that recognises an LTL-expressible language $L \subseteq \{a\}^+$ is equivalent to an LTL formula of size in $O(2^n)$.*

We show next that this upper bound is tight. Consider the language $\{a^{2^{n-1}}\}$, which, according to Proposition 2, is only recognised by LTL formulas of size at least 2^{n-1}. It is recognised by a weak alternating automaton with $2n$ states and size in $O(n)$, using an automaton based on Leiss's construction [22]. Intuitively, the alternating automaton represents an n-bit up-counter with two states for each bit, one for 1 and one for 0 (see Fig. 1), where the universal transitions enforce that nondeterministic transitions correctly update the counter.

Lemma 2 (Adaptation of [22, proof of Theorem 1]). *For every $n \in \mathbb{N} \setminus \{0\}$, there is a weak alternating automaton with $2n$ states and transition function of size in $O(n)$ recognising the language $\{a^{2^{n-1}}\}$.*

We continue to nondeterministic automata (NFAs), for which the arguments are more involved as they do not allow for linear complementation.

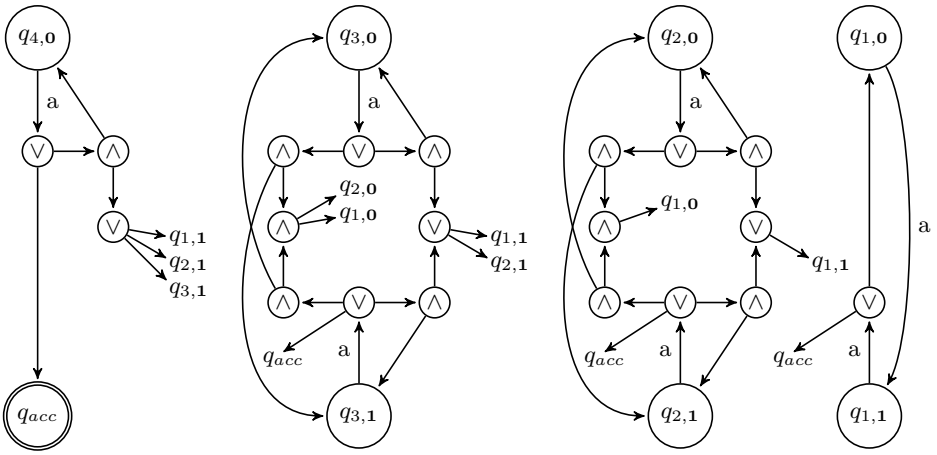

Figure 1. A weak alternating automaton of size in $O(n)$ recognising $\{a^{2^{n-1}}\}$; here with $n = 4$, where the initial configuration is $q_{1,0} \wedge q_{2,0} \wedge q_{3,0} \wedge q_{4,0}$.

Lemma 3. *Every nondeterministic automaton with n states recognising an LTL-expressible language $L \subseteq \{a\}^+$ is equivalent to an LTL formula of size in $O(n^2)$.*

Proof sketch. For finite L, by a pumping argument, $\mathcal{A}$ only accepts words up to length n, and by Proposition 3 we are done. We now consider a co-finite L.

We use 2-way deterministic automata, which are deterministic automata that process words of the form $\vdash w \dashv$, where $\vdash$ and $\dashv$ are start- and end-of-word markers respectively, and where transitions specify whether to read the letter to the right or to the left of the current position. They accept by reaching an end state, and reject by reaching a rejecting state or by failing to terminate [16], and every unary NFA $\mathcal{A}$ can be turned into a 2-way DFA $\mathcal{D}$ of size $O(n^2)$ [8].

We construct from an NFA $\mathcal{A}$ a 2-way DFA $\mathcal{D}$, and then a 2-way DFA $\mathcal{D}'$ of the same size that recognises $a^* \setminus \{a^k\}$, where a^k is the longest word not in L. We use the fact that a 2-way DFA of size m can be complemented into one of size $4m$ [16] to complement $\mathcal{D}'$ into $\mathcal{D}''$ that recognises $\{a^k\}$ and must therefore be of size at least $k + 2$ [1], so k, and by Proposition 2, an LTL formula for L, is in $O(n^2)$. □

We now show that this upper bound is tight. The previous lower bound ideas do not work with nondeterminism, since we need n states to recognise $\{a^n\}$ [19]. Yet, we need not count *exactly* to n for achieving a lower bound. We can use a variant of a language used in [4, pages 10–11]: For every positive integer k, define the set of positive integers $S_k = \{m > 0 \mid \exists i, j \in \mathbb{N}.\ m = ik + j(k + 1)\}$, and the language $V_k = \{a^m \mid m \in S_k\} \subseteq \{a\}^*$.

Proposition 4 (Folklore, [4, Theorem 3]). *For every $k \in \mathbb{N}$ the number $k^2 - k - 1$ is the maximal number not in S_k.*

Proposition 5 ([4, proof of Theorem 4]). *For every $n \in \mathbb{N}$, there is an NFA of size in $O(n)$ recognising a co-finite language $L \subseteq \{a\}^*$, such that a^{k^2-k-1} is not in L, while for every $t \geq k^2 - k$, we have that $a^t \in L$.*

Theorem 1. *The size blow-up involved in translating deterministic, nondeterministic, and alternating automata on finite unary words to LTL, when possible, is $\Theta(n)$, $\Theta(n^2)$, and $\Theta(2^n)$, respectively.*

4 General Alphabet

In this section we consider the more challenging task of turning counter-free ω-regular automata over arbitrary alphabets into LTL. We use the fact that these automata can be turned into reset cascade automata (Krohn-Rhodes-Holonomy decomposition), which we describe in Section 4.1. Our technical contribution is then the translation of reset cascade automata into LTL.

In brief, we build, in Section 4.2, a *parameterised LTL formula* that is satisfied by a word w iff the run of the cascade on w, starting in the parameter configuration S, reaches a parameter configuration T, such that the remaining suffix of w satisfies a parameter LTL formula τ. We then use this formula, in Section 4.4, to describe the automaton's acceptance condition.

When encoding the behavior of a cascade by an LTL formula, we need to overcome two major challenges: First, the cascade is a formalism that looks at the *past*, namely at the word read so far, to determine the next configuration, while an LTL formula obtains its value only from the future. Second, the cascade has an internal state, while an LTL formula does not. Our reachability formulas are therefore quite involved, built inductively over the number of levels in the cascade, and implicitly allowing to track the internal configuration of the cascade.

In Section 4.3 we analyse the length and depth of the resulting formulas.

4.1 Cascaded Automata

Cascades. A cascaded semiautomaton (analogous to the algebraic wreath product) over an alphabet Σ is a semiautomaton that can be described as a sequence of simple semiautomata, such that the alphabet of each of them is Σ together with the current state of each of the preceding semiautomata in the sequence. It is a reset cascade if it is a sequence of reset semiautomata. Formally, a *cascaded semiautomaton*, or just *cascade*, over alphabet Σ with n levels is a tuple $\mathcal{A} = \langle \Sigma, \mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_n \rangle$, such that $\mathcal{A}_i = (\Sigma_i, Q_i, \delta_i)$ is a semiautomaton for each level i, where $\Sigma_i = \Sigma \times Q_1 \times \cdots \times Q_{i-1}$. (So $\Sigma_1 = \Sigma$, $\Sigma_2 = \Sigma \times Q_1$, etc.). It is a *reset cascade* if all $\mathcal{A}_i$'s are reset semiautomata.

An i-configuration S of $\mathcal{A}$ is a tuple $\langle q_1, q_2, \ldots, q_i \rangle \in Q_1 \times \cdots \times Q_i$. If $q_{i+1} \in Q_{i+1}$ is a state of level $i+1$, we write $\langle S, q_{i+1} \rangle$ for the $(i+1)$-configuration $\langle q_1, \ldots, q_i, q_{i+1} \rangle$. Note that the 0-configuration is the empty tuple $\langle\rangle$. Further, we derive the transition relation for configurations by point-wise application of the respective δ_i's. We define $\delta_{\leq i}(\langle q_1, q_2, \ldots q_i \rangle, \sigma)$ as $\langle \delta_1(q_1, \langle\sigma\rangle), \delta_2(q_2, \langle \sigma, q_1 \rangle), \ldots \rangle$.

Note that we will omit the "$\leq i$"-subscript if it is clear from context, and by just writing "configuration", we mean an n-configuration.

Notice that $\mathcal{A}$ describes a standard semiautomaton $\mathcal{D}_\mathcal{A}$ over Σ, whose states are the configurations of $\mathcal{A}$ of level n, and its transition function is $\delta_{\leq n}$. If there are up to j states in each level of $\mathcal{A}$, there are up to j^n states in $\mathcal{D}_\mathcal{A}$. Observe that when $\mathcal{A}$ is a reset cascade, it can be translated to an equivalent reset cascade with up to $n \log j$ levels, and 2 states in each level [13, Ex. I.10.2].

For a state $q \in Q_i$ of level i of a reset cascade, we denote by $\mathsf{Enter}(q)$, $\mathsf{Stay}(q)$, and $\mathsf{Leave}(q) \subseteq \Sigma \times Q_1 \times \cdots \times Q_{i-1}$ the sets of (combined) letters that enter q, stay in it, and leave it, respectively. These are sets of pairs $\langle \sigma, S \rangle$, where S is an $(i-1)$-configuration and $\sigma \in \Sigma$. Notice that $\mathsf{Enter}(q) \subseteq \mathsf{Stay}(q)$, and that $\mathsf{Leave}(q)$ is the complement of $\mathsf{Stay}(q)$ (w.r.t. the relevant (combined) letters).

A semiautomaton (Σ, Q, δ) is *homomorphic* to a cascade $\langle \Sigma, \mathcal{A}_1, \ldots, \mathcal{A}_n \rangle$ if there exists a partial surjective function $\varphi \colon Q_1 \times \cdots \times Q_n \to Q$, such that for every $\sigma \in \Sigma$ and $S \in Q_1 \times \cdots \times Q_n$, we have $\delta(\varphi(S), \sigma) = \varphi(\delta_{\leq n}(S, \sigma))$.

Proposition 6 (Part of the Krohn-Rhodes-Holonomy Decomposition [13, Corollary II.7.2], [25, Theorem 3]). *Every counter-free deterministic semiautomaton $\mathcal{D}$ with n states is homomorphic to a reset cascade $\mathcal{A}$ with up to 2^n levels and 2^n states in each level.*

Remark 1. The Krohn-Rhodes and Holonomy decomposition theorems consider also more general cascades and give results with respect to arbitrary semiautomata. The Holonomy decomposition in [13], as opposed to many other proofs of the Krohn-Rhodes decomposition, guarantees up to 2^n levels with up to 2^n states in each level. Yet, it shows that $\mathcal{A}$ *covers* $\mathcal{D}$, allowing $\mathcal{A}$ to operate over an alphabet different from that of $\mathcal{D}$. In [25,26,24], the algebraic proof of [13] is translated to an automata-theoretic one, providing the stated homomorphism. It is also stated in [25, Theorem 3.1], [26, Corollary 20], and [24, Corollary 2] that the number of configurations in $\mathcal{A}$ is singly exponential in n, but to the best of our understanding they do not provide an explicit proof for it.

Cascades with acceptance conditions. As a cascade $\mathcal{A}$ describes a standard semiautomaton (whose states are the configurations of $\mathcal{A}$), we can add to it an initial configuration and an acceptance condition to make it a standard deterministic automaton. We show below that the homomorphism between an automaton and a cascade can be extended to also transfer the same acceptance condition.

Proposition 7. *Let $\mathcal{D}$ be a deterministic Büchi, coBüchi or Rabin automaton, with a semiautomaton homomorphic to a cascade $\mathcal{A}$. There is respectively a deterministic Büchi, coBüchi or Rabin automaton $\mathcal{D}'$ equivalent to $\mathcal{D}$ with semiautomaton $\mathcal{A}$. For Rabin, $\mathcal{D}$ and $\mathcal{D}'$ have the same number of acceptance pairs.*

Proposition 8. *Consider a deterministic Muller automaton $\mathcal{D}$ with n states, whose semiautomaton is homomorphic to a reset cascade $\mathcal{A}$ with m configurations. Then there is a deterministic Muller automaton $\mathcal{D}'$ equivalent to $\mathcal{D}$, whose semiautomaton is $\mathcal{A}$ and its Muller condition has up to $2^{O(mn)}$ acceptance sets.*

4.2 Encoding Reachability within Reset Cascades by LTL Formulas

For the rest of this section, let us fix a set of atomic propositions AP, an alphabet $\Sigma = 2^{AP}$, and a reset cascade $\mathcal{A} = \langle \Sigma, \mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_n \rangle$.

The main reachability formula. For every level i of $\mathcal{A}$, three configurations S, B and T of level i, and two LTL formulas β and τ, we will define the LTL formula $S \overset{}{\underset{B(\beta)}{\rightsquigarrow}} T\,(\tau)$ with the intended semantics that it holds on a word $w \in \Sigma^\omega$ iff $\mathcal{A}$ goes from the 'starting' configuration S to the 'target' configuration T along some prefix u of w, such that the suffix of w after u satisfies τ and the path along u avoids the 'bad' configuration B with a suffix satisfying β.

Auxiliary reachability formulas. We will formally define the main reachability formula by induction on the level i of the involved configurations, and using four auxiliary formulas, whose intended semantics is described in Table 1. These formulas distinguish between the case that the top-level state is unchanged along the reachability path, denoted with a solid arrow $\longrightarrow$, and the case that it is changed, denoted by a dashed arrow $\dashrightarrow$. They also have dual, weak, versions.

Observe that intuitively $S \underset{B(\beta)}{\rightsquigarrow} T\,(\tau)$ is an extended *Until* operator, while its dual $S \overset{\text{weak}}{\underset{B(\beta)}{\rightsquigarrow}} T\,(\tau) = \neg(S \underset{T(\tau)}{\rightsquigarrow} B\,(\beta))$ is an extended *Weak until* (or *Release*) operator. We build the formulas so that for appropriate choices of β and τ, the (strong) reachability formulas 1, 3, and 5 (as numbered in Table 1) are syntactic co-safety and the weak formulas 2 and 4 are syntactic safety formulas.

Formulas 1 and 2. The main formula is simply defined as the union of two auxiliary formulas, corresponding to whether or not the top-level state changes, and its weak version is defined to be its dual.

$$S \underset{B(\beta)}{\rightsquigarrow} T\,(\tau) \coloneqq \begin{cases} (\neg\beta)\mathbf{U}\tau & \text{if } S = \langle\rangle \\ S \underset{B(\beta)}{\longrightarrow} T\,(\tau) \vee S \underset{B(\beta)}{\dashrightarrow} T\,(\tau) & \text{otherwise.} \end{cases}$$

$$S \overset{\text{weak}}{\underset{B(\beta)}{\rightsquigarrow}} T\,(\tau) \coloneqq \neg\left(S \underset{T(\tau)}{\rightsquigarrow} B\,(\beta)\right)$$

Formula 3. Since the formula should ensure that the top-level state s is unchanged, we first distinguish between four cases, depending on which of the source configuration $\langle S, s\rangle$, bad configuration $\langle B, b\rangle$, and target configuration $\langle T, t\rangle$ are equal. The definitions of the four cases only differ in whether or not each of β and τ are satisfied in the first position of the word.

We define them using an intermediate common formula that is indifferent to the first position, which we mark by "> 0" on top of the arrow. We then define the "> 0" formula by using the main reachability formula with respect to a lower level, namely with respect to the configurations S and T instead of $\langle S, s\rangle$ and $\langle T, t\rangle$, and having corresponding disjunctions and conjunctions on all the combined letters of the top level that belong to $\mathsf{Stay}(s)$ and $\mathsf{Leave}(s)$.

Reachability formula φ	Intended semantics Intuitively: Reading a word w from the configuration S or $\langle S, s\rangle$ Formally: $w \models \varphi \iff$
1. $S \overset{}{\rightsquigarrow}_{\not B(\beta)} T\,(\tau)$	not reaching $B(\beta)$ until reaching $T(\tau)$. $\exists i \geq 0.\ \delta(S, w_{[0..i)}) = T \wedge w_{[i..]} \models \tau$ $\wedge\ (\forall j \in [0..i).\ \delta(S, w_{[0..j)}) \neq B \vee w_{[j..]} \not\models \beta)$
2. $S \overset{\text{weak}}{\rightsquigarrow}_{\not B(\beta)} T\,(\tau)$	reaching $T(\tau)$ releases not reaching $B(\beta)$. $\forall i \geq 0.\ (\delta(S, w_{[0..i)}) = B \wedge w_{[i..]} \models \beta)$ $\rightarrow\ (\exists j \in [0..i).\ \delta(S, w_{[0..j)}) = T \wedge w_{[j..]} \models \tau)$
3. $\langle S, s\rangle \xrightarrow[\not\langle B,b\rangle(\beta)]{} \langle T, t\rangle\,(\tau)$	not reaching $\langle B, b\rangle(\beta)$ until reaching $\langle T, t\rangle(\tau)$, while staying in s. $\exists i \geq 0.\ \delta(\langle S, s\rangle, w_{[0..i)}) = \langle T, t\rangle \wedge w_{[i..]} \models \tau$ $\wedge\ (\forall j \in [0..i).\ \delta(\langle S, s\rangle, w_{[0..j)}) \neq \langle B, b\rangle \vee w_{[j..]} \not\models \beta)$ $\wedge\ (\forall j \in [0..i).\ \langle w[j], \delta(S, w_{[0..j)})\rangle \in \mathsf{Stay}(s))$
4. $\langle S, s\rangle \xrightarrow[\not\langle B,b\rangle(\beta)]{\text{weak}} \langle T, t\rangle\,(\tau)$	reaching $\langle T, t\rangle(\tau)$ releases not (reaching $\langle B, b\rangle(\beta)$ or leaving s). $\forall i \geq 0.\ \big((\delta(\langle S, s\rangle, w_{[0..i)}) = \langle B, b\rangle \wedge w_{[i..]} \models \beta)$ $\vee\ (i > 0 \wedge \langle w[i{-}1], \delta(S, w_{[0..i-1)})\rangle \in \mathsf{Leave}(s))\big)$ $\rightarrow\ (\exists j \in [0..i).\ \delta(\langle S, s\rangle, w_{[0..j)}) = \langle T, t\rangle \wedge w_{[j..]} \models \tau)$
5. $\langle S, s\rangle \xrightarrow[\not\langle B,b\rangle(\beta)]{} \langle T, t\rangle\,(\tau)$	not reaching $\langle B, b\rangle(\beta)$ until reaching $\langle T, t\rangle(\tau)$ and leaving s. $\exists i_1, i_2 \geq 0.\ \delta(\langle S, s\rangle, w_{[0..i_1)}) = \langle T, t\rangle \wedge w_{[i_1..]} \models \tau$ $\wedge\ (\exists j_1 \in [0..i_1).\ \langle w[j_1], \delta(S, w_{[0..j_1)})\rangle \in \mathsf{Enter}(t))$ $\wedge\ \langle w[i_2], \delta(S, w_{[0..i_2)})\rangle \in \mathsf{Leave}(s)$ $\wedge\ (\forall j_2 \in [0..\max(i_1{-}1, i_2)].\ \delta(\langle S, s\rangle, w_{[0..j_2)}) \neq \langle B, b\rangle$ $\vee\ w_{[j_2..]} \not\models \beta)$

Table 1. The intended semantics of reachability formulas. Orange subformulas show the difference between the auxiliary formulas and the first or second (main) formula.

$$\langle S, s\rangle \xrightarrow[\not\langle B,b\rangle(\beta)]{} \langle T, t\rangle\,(\tau) \coloneqq \begin{cases} \langle S, s\rangle \xrightarrow[\not\langle B,b\rangle(\beta)]{>0} \langle T, t\rangle\,(\tau) & \text{if } \langle S, s\rangle \neq \langle B, b\rangle \text{ and } \langle S, s\rangle \neq \langle T, t\rangle \\ \langle S, s\rangle \xrightarrow[\not\langle B,b\rangle(\beta)]{>0} \langle T, t\rangle\,(\tau) \vee \tau & \text{if } \langle S, s\rangle \neq \langle B, b\rangle \text{ and } \langle S, s\rangle = \langle T, t\rangle \\ \langle S, s\rangle \xrightarrow[\not\langle B,b\rangle(\beta)]{>0} \langle T, t\rangle\,(\tau) \wedge \neg\beta & \text{if } \langle S, s\rangle = \langle B, b\rangle \text{ and } \langle S, s\rangle \neq \langle T, t\rangle \\ \left(\langle S, s\rangle \xrightarrow[\not\langle B,b\rangle(\beta)]{>0} \langle T, t\rangle\,(\tau) \wedge \neg\beta\right) \vee \tau & \text{if } \langle S, s\rangle = \langle B, b\rangle \text{ and } \langle S, s\rangle = \langle T, t\rangle \end{cases}$$

$$\text{where } \langle S, s\rangle \xrightarrow[\not\langle B,b\rangle(\beta)]{>0} \langle T, t\rangle\,(\tau) \coloneqq \bigvee_{\substack{\langle \sigma, T'\rangle \in \mathsf{Stay}(s) \\ \text{s.t. } \langle T', s\rangle \xrightarrow{\sigma} \langle T, t\rangle}} \left(S \overset{}{\rightsquigarrow}_{\not S(\mathbf{false})} T'\,(\sigma \wedge \mathbf{X}\tau) \right.$$

$$\wedge \bigwedge_{\langle \eta, L\rangle \in \mathsf{Leave}(s)} S \overset{\not{L(\eta)}}{\rightsquigarrow} T' \, (\sigma \wedge \mathbf{X}\tau) \quad \wedge \bigwedge_{\substack{\langle \rho, B'\rangle \in \mathsf{Stay}(s) \\ \text{s.t. } \langle B', s\rangle \xrightarrow{\rho} \langle B, b\rangle}} S \overset{\not{B'(\rho \wedge \mathbf{X}\beta)}}{\rightsquigarrow} T' \, (\sigma \wedge \mathbf{X}\tau) \Bigg)$$

Formula 4. Its intended semantics is also that the top-level state s is unchanged, but we weaken Formula 3 by not enforcing that the target configuration $\langle T, t\rangle$ is reached and τ is satisfied. Thus as long as the top-level state s stays unchanged and the bad configuration $\langle B, b\rangle$ is not reached while satisfying β, Formula 4 is also satisfied. Note that since both Formula 3 and Formula 4 need to ensure that the top-level state s is unchanged they cannot simply be defined as the dual of each other. However, they share the same construction principle:

$$\langle S, s\rangle \overset{\text{weak}}{\underset{\not{\langle B,b\rangle(\beta)}}{\longrightarrow}} \langle T, t\rangle \, (\tau) \coloneqq \begin{cases} \langle S, s\rangle \overset{\text{weak},>0}{\underset{\not{\langle B,b\rangle(\beta)}}{\longrightarrow}} \langle T, t\rangle \, (\tau) & \text{if } \langle S, s\rangle \neq \langle B, b\rangle \text{ and } \langle S, s\rangle \neq \langle T, t\rangle \\ \langle S, s\rangle \overset{\text{weak},>0}{\underset{\not{\langle B,b\rangle(\beta)}}{\longrightarrow}} \langle T, t\rangle \, (\tau) \vee \tau & \text{if } \langle S, s\rangle \neq \langle B, b\rangle \text{ and } \langle S, s\rangle = \langle T, t\rangle \\ \langle S, s\rangle \overset{\text{weak},>0}{\underset{\not{\langle B,b\rangle(\beta)}}{\longrightarrow}} \langle T, t\rangle \, (\tau) \wedge \neg\beta & \text{if } \langle S, s\rangle = \langle B, b\rangle \text{ and } \langle S, s\rangle \neq \langle T, t\rangle \\ \left(\langle S, s\rangle \overset{\text{weak},>0}{\underset{\not{\langle B,b\rangle(\beta)}}{\longrightarrow}} \langle T, t\rangle \, (\tau) \vee \tau \right) \wedge \neg\beta & \text{if } \langle S, s\rangle = \langle B, b\rangle \text{ and } \langle S, s\rangle = \langle T, t\rangle \end{cases}$$

where

$$\langle S, s\rangle \overset{\text{weak},>0}{\underset{\not{\langle B,b\rangle(\beta)}}{\longrightarrow}} \langle T, t\rangle \, (\tau) \coloneqq$$

$$\bigvee_{\substack{\langle \sigma, T'\rangle \in \mathsf{Stay}(s) \\ \text{s.t. } \langle T', s\rangle \xrightarrow{\sigma} \langle T, t\rangle}} \left(\bigwedge_{\langle \eta, L\rangle \in \mathsf{Leave}(s)} S \overset{\text{weak}}{\underset{\not{L(\eta)}}{\rightsquigarrow}} T' \, (\sigma \wedge \mathbf{X}\tau) \wedge \bigwedge_{\substack{\langle \rho, B'\rangle \in \mathsf{Stay}(s) \\ \text{s.t. } \langle B', s\rangle \xrightarrow{\rho} \langle B, b\rangle}} S \overset{\text{weak}}{\underset{\not{B'(\rho \wedge \mathbf{X}\beta)}}{\rightsquigarrow}} T' \, (\sigma \wedge \mathbf{X}\tau) \right) \tag{1}$$

$$\bigvee \left(\bigwedge_{\langle \eta, L\rangle \in \mathsf{Leave}(s)} S \overset{\text{weak}}{\underset{\not{L(\eta)}}{\rightsquigarrow}} S \, (\mathbf{false}) \quad \wedge \bigwedge_{\substack{\langle \rho, B'\rangle \in \mathsf{Stay}(s) \\ \text{s.t. } \langle B', s\rangle \xrightarrow{\rho} \langle B, b\rangle}} S \overset{\text{weak}}{\underset{\not{B'(\rho \wedge \mathbf{X}\beta)}}{\rightsquigarrow}} S \, (\mathbf{false}) \right) \tag{2}$$

Formula 5. The definition of the last reachability formula is the most challenging, since the top-level state changes ($s \neq t$), which prevents the direct usage of lower level configurations.

Intuitively, before reaching the target configuration $\langle T, t\rangle$, the run must see a combined letter $\langle \sigma, T'\rangle \in \mathsf{Enter}(t)$, after which the top-level state t is preserved and the bad situation $\langle B, b\rangle(\beta)$ is avoided. This is line (1) of the definition.

The run must also not see $\langle B, b\rangle(\beta)$ before reaching T', which is handled in line (2), whose difference from line (1) is the additional constraint on the path from S to T'. (Line (1) is required for the case that $\mathsf{Enter}(b)$ is empty.) We use Formula 4 for that constraint, rather than Formula 3 which could also be used, in order to ensure that Formula 5 can be a syntactic co-safety formula.

Lastly, line (3) ensures that the top-level state is indeed changed.

$$\langle S,s\rangle \xrightarrow[\langle B,b\rangle(\beta)]{} \langle T,t\rangle(\tau) \coloneqq$$

$$\bigvee_{\substack{\langle\sigma,T'\rangle\in\\ \mathsf{Enter}(t)}} \left(S \overset{}{\rightsquigarrow}_{S(\mathbf{false})} T'\left(\sigma \wedge \mathbf{X}\Big(\delta(\langle T',\cdot\rangle,\sigma) \xrightarrow[\langle B,b\rangle(\beta)]{} \langle T,t\rangle(\tau)\Big)\right) \right) \wedge \tag{1}$$

$$\bigwedge_{\substack{\langle\eta,R\rangle\in\\ \mathsf{Enter}(b)}} S \overset{}{\rightsquigarrow}_{R(\eta\wedge\mathbf{X}(\delta(\langle R,\cdot\rangle,\eta)\xrightarrow[\langle T,t\rangle(\tau)]{\text{weak}}\langle B,b\rangle(\beta)))} T'\left(\sigma \wedge \mathbf{X}\Big(\delta(\langle T',\cdot\rangle,\sigma) \xrightarrow[\langle B,b\rangle(\beta)]{} \langle T,t\rangle(\tau)\Big)\right) \tag{2}$$

$$\wedge \bigvee_{\substack{\langle\sigma,L\rangle\in\\ \mathsf{Leave}(s)}} \langle S,s\rangle \xrightarrow[\langle B,b\rangle(\beta)]{} \langle L,s\rangle \left(\sigma \wedge \begin{cases} \neg\beta & \text{if } \langle L,s\rangle = \langle B,b\rangle \\ \mathbf{true} & \text{otherwise.} \end{cases}\right) \tag{3}$$

We prove the correctness of the above definitions with respect to the intended meaning of Table 1 by induction on the level of the involved configurations.

Lemma 4. *The intended semantics of Table 1 hold for all infinite words $w \in \Sigma^\omega = (2^{AP})^\omega$, configurations S, B, T of level $m \leq n$, states s, b, t in level $m+1$ (when $m < n$), and LTL formulas β and τ over AP.*

Using the same induction principle we prove that the reachability formulas stay within certain classes of the syntactic future hierarchy (Definition 1). We use $S \overset{}{\rightsquigarrow}_{B(X)} T(Y) \in Z$ as a shorthand for saying that for every formulas $\beta \in X$ and $\tau \in Y$, the formula $S \overset{}{\rightsquigarrow}_{B(\beta)} T(\tau)$ is in Z.

Lemma 5. *Let S, B, T be configurations of level $m \leq n$, and let s, b, t be states in level $m+1$ (when $m < n$). Then for $i \geq 1$ it holds that:*

- $S \overset{}{\rightsquigarrow}_{B(\Pi_i)} T(\Sigma_i)$, $\langle S,s\rangle \xrightarrow[\langle B,b\rangle(\Pi_i)]{} \langle T,t\rangle(\Sigma_i)$, $\langle S,s\rangle \xrightarrow[\langle B,b\rangle(\Pi_i)]{} \langle T,t\rangle(\Sigma_i) \in \Sigma_i$
- $S \overset{\text{weak}}{\rightsquigarrow}_{B(\Sigma_i)} T(\Pi_i)$, $\langle S,s\rangle \xrightarrow[\langle B,b\rangle(\Sigma_i)]{\text{weak}} \langle T,t\rangle(\Pi_i) \in \Pi_i$

4.3 Depth and Length Analysis

We analyze the length and temporal-nesting depth of the LTL reachability formulas defined in Section 4.2. Notice that both measures are of independent interest, as there might be a non-elementary gap between the depth and length of LTL formulas [14, Theorem 6]. Since we provide upper bounds, the bound on the length of formulas obviously gives also a bound on their size.

We consider a reset cascade $\mathcal{A}$ with n levels, as in Section 4.2, and further assume for the length and depth analysis that it has up to n states in each level. (This assumption holds in the reset cascades that result from the Krohn-Rohdes decomposition as per Proposition 6.)

We define for each of the five reachability formulas a *depth function* $\mathsf{D}_x(i,d)$ and a *length function* $\mathsf{L}_x(i,l)$, where x refers to the number of the reachability

formula, to bound the depth and length of the formulas. These depend on the level i of its input configurations S, B and T, and the maximal depth d and length l of its input formulas β and τ. For the main (first) reachability formula, we also use D and L, standing for D_1 and L_1. For example, the length of the first formula $S \underset{B(\beta)}{\rightsquigarrow} T\,(\tau)$ over configurations S, B and T of level 7 and formulas β and τ of length up to 77 is bounded by the value of $\mathsf{L}_1(7, 77)$.

For simplicity, we consider the LTL representation of an alphabet letter $\sigma \in \Sigma$ to be of length 1, while its actual length is $3 \log_2 |\Sigma|$. This increase is due to the need to encode an alphabet letter $\sigma \in \Sigma = 2^{AP}$ as a conjunction of atomic propositions in AP. The representation length can be multiplied by the total length of the final relevant formula (e.g., a formula equivalent to the entire reset cascade), since it remains constant along all steps of our inductive computation.

We provide in Table 2 upper bounds on the depth and length functions, relative to values of other depth and length functions with respect to configurations of the same or lower-by-one level. The table is constructed by following the syntactic definitions of the reachability formulas, and applying basic simplifications to the resulting expressions. For example, $\mathsf{L}_1(0, l) = 2+2l$ standing for the length of $(\neg\beta)\mathbf{U}\tau$. In Lemma 6 we will use Table 2 to bound the absolute depth and length of the main reachability formula.

Depth Analysis. The temporal nesting depth of the main reachability formula $S \underset{B(\beta)}{\rightsquigarrow} T\,(\tau)$ is intuitively exponential in the number n of levels of the reset cascade (linear in the number of configurations), since it is defined inductively along these levels, and the depth of a level-$(i+1)$ formula is about twice the depth of a level-i formula. The parameters of the reachability formula are both the configurations S, B and T of level i, and the formulas β and τ; yet, the depth of the reachability formula only linearly depends on the depth of β and τ.

Length Analysis. Intuitively, the overall length of the main reachability formula $S \underset{B(\beta)}{\rightsquigarrow} T\,(\tau)$ with respect to configurations of the top level is doubly exponential in the number n of levels of the reset cascade (and thus singly exponential in the number of configurations), since the formula is defined inductively along these levels, and the length $\mathsf{L}(i, l)$ is roughly $\mathsf{L}(i{-}1, l) \cdot \mathsf{L}(i{-}1, l)$. More precisely, $\mathsf{L}(i, l) = l \cdot f(i)$ for some doubly exponential function $f(i)$.

Now, why is $\mathsf{L}(i, l)$ roughly equal to $\mathsf{L}(i{-}1, l) \cdot \mathsf{L}(i{-}1, l)$? The dominant component of the level-i reachability formula is line (2) in the definition of $\langle S, s\rangle \underset{\langle B,b\rangle(\beta)}{\dashrightarrow} \langle T, t\rangle\,(\tau)$. It is a level-$(i{-}1)$ reachability formula whose formula-parameters are themselves auxiliary reachability formulas of level i with formula parameters of length l. The length of an auxiliary reachability formula of level i is roughly as of the main reachability formula of level $i{-}1$, implying that the length of $L_i(l)$ is roughly $L_{i-1}(L_{i-1}(l))$. By the inductive proof that $L_{i-1}(l) = l \cdot f(i{-}1)$, we get that $L_i(l) = L_{i-1}(L_{i-1}(l)) = L_{i-1}(l) \cdot f(i{-}1) = l \cdot f(i{-}1) \cdot f(i{-}1)$.

As for the many disjunctions and conjunctions that appear in the formulas, observe that the number of disjuncts and conjuncts does not depend on the

Reachability formula φ	Bounds on $\mathsf{depth}(\varphi)$ and length $\lvert\varphi\rvert$
1. $S \overset{B(\beta)}{\rightsquigarrow} T\,(\tau)$	$\mathsf{D}_1(i,d) \leq \begin{cases} d+1 & \text{if } i=0 \\ \max(\mathsf{D}_3(i,d)\,,\,\mathsf{D}_5(i,d)) & \text{otherwise.} \end{cases}$ $\mathsf{L}_1(i,l) \leq \begin{cases} 2+2l & \text{if } i=0 \\ 1+\mathsf{L}_3(i,l)+\mathsf{L}_5(i,l) & \text{otherwise.} \end{cases}$
2. $S \overset{B(\beta)}{\overset{\text{weak}}{\rightsquigarrow}} T\,(\tau)$	$\mathsf{D}_2(i,d) = \mathsf{D}_1(i,d)$ $\mathsf{L}_2(i,l) = 1+\mathsf{L}_1(i,l)$
3. $\langle S,s\rangle \xrightarrow{\langle B,b\rangle(\beta)} \langle T,t\rangle\,(\tau)$	$\mathsf{D}_3(i,d) \leq \mathsf{D}_1(i{-}1,d+1)$ $\mathsf{L}_3(i,l) \leq 3{+}2l+\lvert\Sigma\rvert n^{i-1}\big(1{+}\mathsf{L}_1(i{-}1,3{+}l){+}$ $1+\lvert\Sigma\rvert n^{i-1}(\mathsf{L}_1(i{-}1,3{+}l)+1){+}$ $1+\lvert\Sigma\rvert n^{i-1}(\mathsf{L}_1(i{-}1,3{+}l)+1)\big)$ $\leq 3+2l+4\lvert\Sigma\rvert^2 n^{2(i-1)}\mathsf{L}_1(i{-}1,l{+}3)$
4. $\langle S,s\rangle \overset{\text{weak}}{\xrightarrow{\langle B,b\rangle(\beta)}} \langle T,t\rangle\,(\tau)$	$\mathsf{D}_4(i,d) \leq \mathsf{D}_2(i{-}1,d+1) = \mathsf{D}_1(i{-}1,d+1)$ $\mathsf{L}_4(i,l) \leq 3+2l+(1+\lvert\Sigma\rvert n^{i-1})\big(1+\lvert\Sigma\rvert n^{i-1}(1+\mathsf{L}_2(i{-}1,l{+}3))\big)$ $\leq 3+2l+4\lvert\Sigma\rvert^2 n^{2(i-1)}\mathsf{L}_1(i{-}1,l+3)$
5. $\langle S,s\rangle \overset{\langle B,b\rangle(\beta)}{\dashrightarrow} \langle T,t\rangle\,(\tau)$	$\mathsf{D}_5(i,d) \leq \mathsf{D}_1(i{-}1,\max(1+\mathsf{D}_3(i,d)\,,\,1+\mathsf{D}_4(i,d)))$ $\mathsf{L}_5(i,l) \leq \lvert\Sigma\rvert n^{i-1}\cdot\big(\mathsf{L}_1(i-1,3+\mathsf{L}_3(i,l))+2+$ $\lvert\Sigma\rvert n^{i-1}\cdot\big(\mathsf{L}_1(i-1,\max(3+\mathsf{L}_3(i,l),3+\mathsf{L}_4(i,l)))+1\big)\big)$ $+1+\lvert\Sigma\rvert n^{i-1}\cdot(1+\mathsf{L}_3(i,3+l))$

Table 2. The relative depths and lengths of the reachability formulas over configurations of level i, and LTL formulas β and τ of depth at most d and length at most l. For the first two reachability formulas, we consider $i \geq 0$ and for the other formulas $i \geq 1$.

formula-parameters β and τ, but only the level i of the configurations S, B, and T. Hence, they do not dominate the growth rate of the overall formula length.

Lemma 6. *Consider a reset cascade $\mathcal{A}$ with n levels and up to n states in each level, and a formula $\zeta = S \overset{B(\beta)}{\rightsquigarrow} T\,(\tau)$ with configurations S, B and T of $\mathcal{A}$ of level $i \leq n$. Let $d = \max(\mathsf{depth}(\beta), \mathsf{depth}(\tau))$ and let $l = \max(\lvert\beta\rvert, \lvert\tau\rvert)$. Then:*

$$(a)\ \mathsf{depth}(\zeta) \leq d + 3^i \quad \text{and} \quad (b)\ \lvert\zeta\rvert \leq l \cdot (10\lvert\Sigma\rvert^2 n)^{4^i}$$

Lemma 6 is proven by induction on i. The details are given in Appendix A.4.

4.4 Translating Deterministic Counter-Free Automata to LTL

We use the reachability formulas of Section 4.2 to translate a reset cascade $\mathcal{A}$ to an equivalent LTL formula. Our LTL formulation of $\mathcal{A}$'s acceptance condition is based on an LTL formulation of "C is visited finitely/infinitely often along

a run of $\mathcal{A}$ on a word w", for a given configuration C of $\mathcal{A}$. It thus applies to every ω-regular acceptance condition and by Propositions 6 and 8 to every deterministic counter-free ω-regular automaton. We introduce two shorthands to the main reachability formula: the first is satisfied if we reach T from S without any side constraints (which is always satisfied in the case that $S = T$), and the second requires that we reach it along a nonempty prefix.

$$S \rightsquigarrow T \coloneqq S \underbrace{\rightsquigarrow}_{T\,(\mathbf{false})} T\,(\mathbf{true}) \qquad S \overset{>0}{\rightsquigarrow} T \coloneqq \bigvee_{\sigma \in \Sigma} \Big(\sigma \wedge \mathbf{X}(\delta(S, \sigma) \rightsquigarrow T)\Big)$$

With Lemmas 4 and 5 we then obtain (a proof is given in Appendix A.5):

Lemma 7. *Consider a reset cascade $\mathcal{A} = \langle 2^{AP}, \mathcal{A}_1, \ldots, \mathcal{A}_n \rangle$ together with an initial configuration ι and some configuration C. Then for a word $w \in (2^{AP})^\omega$, the run of $\mathcal{A}$ on w starting in ι visits C finitely often iff w satisfies the formula $Fin(C) \coloneqq \neg(\iota \rightsquigarrow C) \vee \iota \rightsquigarrow C(\neg(C \overset{>0}{\rightsquigarrow} C))$. Furthermore, $Fin(C) \in \Sigma_2$.*

We are now in position to give our main result.

Theorem 2. *Every counter-free deterministic ω-regular automaton $\mathcal{D}$ over alphabet 2^{AP} with n states (and any acceptance condition) is equivalent to an LTL formula φ over atomic propositions AP of double-exponential temporal-nesting depth (in $O(2^{2^n})$) and triple-exponential length (in $2^{2^{O(2^n)}}$). If $\mathcal{D}$ is a looping-Büchi, looping-coBüchi, weak, Büchi, coBüchi, or Muller automaton then φ is respectively in the $\Pi_1, \Sigma_1, \Delta_1, \Pi_2, \Sigma_2$, or Δ_2 syntactic fragment of LTL.*

Proof. We first prove the general result, w.r.t. an arbitrary counter-free deterministic automaton $\mathcal{D}$, and then take into account $\mathcal{D}$'s acceptance condition, to establish the last part of the theorem.

Consider a counter-free deterministic ω-regular automaton $\mathcal{D}$ with some acceptance condition and n states. Recall that there is a Muller automaton $\mathcal{D}'$ equivalent to $\mathcal{D}$ over the semiautomaton of $\mathcal{D}$. By Propositions 6 and 8, $\mathcal{D}'$ is equivalent to a deterministic Muller automaton $\mathcal{D}''$ that is described by a reset cascade $\mathcal{A}$ with up to $m = 2^n$ levels and m states in each level (and thus up to m^m configurations), and whose acceptance condition has up to $k \in 2^{O(m^m n)} = 2^{O(m^m)}$ acceptance sets. An LTL formula φ equivalent to $\mathcal{D}$ can be defined by formulating the acceptance condition of $\mathcal{D}'$ along Lemma 7.

Recall that the Muller condition is a k-elements disjunction, where each disjunct M is a conjunction of requirements to visit infinitely often every configuration from some set G and finitely often every configuration not in G. Observe that M can be formulated as a disjunction over all the configurations in $\mathcal{D}''$ (at most m^m), having for each configuration C the LTL formula $Fin(C)$ or $\neg Fin(C)$, as defined in Lemma 7, depending on whether or not $C \in G$. Hence, the overall formula φ is a combination of disjunctions and conjunctions of up to $k \cdot m^m$ subformulas of the form $Fin(C)$ or $\neg Fin(C)$. Therefore, the depth of φ is the same as of $Fin(C)$, while $|\varphi| \in O(km^m |Fin(C)|) \leq 2^{O(m^m)} |Fin(C)|$. For calculating $\mathsf{depth}(Fin(C))$ and $|Fin(C)|$, we use Lemma 6 bottom up over the subformulas of $Fin(C)$.

Depth.

$\mathsf{depth}(\iota \rightsquigarrow C) \leq 3^m$; $\mathsf{depth}(C \overset{>0}{\rightsquigarrow} C) \leq 3^m + 1$

$\mathsf{depth}(\iota \rightsquigarrow C(\neg(C \overset{>0}{\rightsquigarrow} C))) \leq 2 \cdot 3^m + 1$

$\mathsf{depth}(\mathit{Fin}(C)) = \max(3^m, 2 \cdot 3^m + 1) \in O(3^m) = O(2^{2^n})$,
implying $\mathsf{depth}(\varphi) \in O(2^{2^n})$.

Length.

$|\iota \rightsquigarrow C| \leq (10|\Sigma|^2 m)^{4^m}$; $|C \overset{>0}{\rightsquigarrow} C| \leq (4|\Sigma|) \cdot (10|\Sigma|^2 m)^{4^m}$

$|\iota \rightsquigarrow C(\neg(C \overset{>0}{\rightsquigarrow} C))| \leq (4|\Sigma|(10|\Sigma|^2 m)^{4^m} + 1)(10|\Sigma|^2 m)^{4^m} \in (|\Sigma| m)^{2^{O(m)}}$

$|\mathit{Fin}(C)| \in 2 + (10|\Sigma|^2 m)^{4^m} + (|\Sigma| m)^{2^{O(m)}} \in (|\Sigma| m)^{2^{O(m)}}$.

Therefore, $|\varphi| \in 2^{O(m^m)} \cdot (m^m) \cdot ((|\Sigma| m)^{2^{O(m)}}) = |\Sigma|^{2^{O(m)}}$.

Expressing the length of φ with respect to the number n of states in the automaton $\mathcal{D}$, and taking into account the fact that the alphabet Σ has at most n^n different letters (any additional letter must have the same behavior as another letter), we have: $|\varphi| \in |\Sigma|^{2^{O(2^n)}} \leq (2^n)^{2^{O(2^n)}} = 2^{2^{O(2^n)}}$.

We now sketch the second part of the theorem connecting the syntactic hierarchy and the different acceptance conditions of $\mathcal{D}$. We only consider the cases in which $\mathcal{D}$ is either a Muller or a coBüchi automaton. The complete analysis is given in Appendix A.5. If $\mathcal{D}$ is a Muller automaton, then the overall formula φ is in Δ_2, since it is a Boolean combination of $\mathit{Fin}(C)$ formulas, which by Lemma 7 belong to Σ_2. If $\mathcal{D}$ is a coBüchi automaton, then we construct the formula φ directly from the coBüchi condition α: φ is a conjunction of $\mathit{Fin}(C)$ formulas over all configurations C that are mapped to states in α. As $\mathit{Fin}(C)$ belongs to Σ_2, so does φ. □

Observe that by Theorem 2, we get the following result, extending the result of [38, Theorem 3.2] that only considers Rabin automata.

Corollary 1. *Every counter-free deterministic ω-regular automaton (with any acceptance condition) recognises an LTL-definable language.*

Proof. Recall that every deterministic ω-regular automaton is equivalent to a deterministic Muller automaton over the same semiautomaton (see, e.g., [3]). The claim is then a direct consequence of Theorem 2. □

Remark 2. Theorem 2 can be adapted to the finite-word setting. While on infinite words, the ne**X**t operator is self-dual, i.e., $\neg\mathbf{X}\psi$ is equivalent to $\mathbf{X}\neg\psi$, over finite words, this equivalence does not hold on words of length 1. Thus **X** gains a dual *weak next*, defined as $\tilde{\mathbf{X}}\psi \coloneqq \neg\mathbf{X}\neg\psi$. In the finite word case, syntactic cosafety (safety) formulas are constructed from **true**, **false**, a, $\neg a$, $\vee$, $\wedge$, and the temporal operators **U** and **X** (**R** and $\tilde{\mathbf{X}}$). Observe that **X** and $\tilde{\mathbf{X}}$ differ only on words of length 1, and thus the only required change in our translation scheme is to replace some **X**s with $\tilde{\mathbf{X}}$s in the reachability formula 4. For finite words a

translation of a counter-free DFA to an LTL formula with only a double exponential size blow-up is known [41]; however, unlike our translation, it does not guarantee syntactic safety (cosafety) formulas for safety (cosafety) languages.

Lastly, we provide a corollary on looping automata, using Theorem 2 and the following known result.

Proposition 9 (Rephrased Theorem 13 from [28]). *Let $\mathcal{D}$ be a deterministic looping-Büchi automaton with n states that recognises an LTL-definable language. Then there exists an equivalent counter-free deterministic looping-Büchi automaton $\mathcal{D}'$ with at most n states.*

Corollary 2. *Every deterministic looping-Büchi (looping-coBüchi) automaton with n states that recognises an LTL-definable language is equivalent to an LTL formula $\varphi \in \Pi_1$ (Σ_1) of temporal nesting depth in $O(2^{2^n})$ and length in $2^{2^{O(2^n)}}$.*

This is an elementary upper bound for two constructions for which either the upper bound was unknown or non-elementary: the liveness-safety decomposition of LTL [28] and the translation of semantic safety LTL to syntactic safety LTL.

5 Conclusions

We have studied the size trade-offs between LTL and automata. Over a unary alphabet, the situation is straightforward and we provided tight complexity bounds. The general case of infinite words over an arbitrary alphabet is more complex. We gave to our knowledge the first elementary complexity bound on the translation of counter-free deterministic ω-regular automata into LTL formulas.

Every ω-regular automaton recognising an LTL-definable language can be translated to a counter-free deterministic automaton [38, Theorem 3.2]. Yet, we are unaware of a bound on the size blow-up involved in such a translation. Once established, it can be combined with our translation to get a general bound on the translation of automata to LTL. It will also provide a (currently unknown[6]) elementary upper bound on the translation of LTL with both future and past operators to LTL with only future operators (which is the version of LTL that we have considered), as (both version of) LTL can be translated to nondeterministic Büchi automata with a single exponential size blow-up [40, Theorem 2.1].

While going from non-elementary to double-exponential depth and triple-exponential length is an improvement, these upper bounds might not be tight—there is currently no known non-linear lower bound! Closing this gap is a challenging open problem, which might require new lower bound techniques for alternating automata, as LTL formulas are an inherently alternating model.

Acknowledgements. We thank Moshe Vardi and Orna Kupferman for suggesting studying the succinctness gap between semantic and syntactic safe formulas, and Mikołaj Bojańczyk for answering our questions on algebraic automata theory.

[6] In consultation with the author of [29], we have confirmed that while the lower bound provided in that paper holds, the stated upper bound is erroneous.

References

1. Birget, J.C.: Two-way automata and length-preserving homomorphisms. Mathematical Systems Theory **29**(3), 191–226 (1996)
2. Bojańczyk, M.: Languages recognised by finite semigroups, and their generalisations to objects such as trees and graphs, with an emphasis on definability in monadic second-order logic (2020)
3. Boker, U.: Why these automata types? In: Proc. of LPAR. pp. 143–163 (2018)
4. Boker, U., Kupferman, O.: The quest for a tight translation of Büchi to co-Büchi automata. In: Fields of Logic and Computation, pp. 147–164. Springer (2010)
5. Cerná, I., Pelánek, R.: Relating hierarchy of temporal properties to model checking. In: MFCS. Lecture Notes in Computer Science, vol. 2747, pp. 318–327. Springer (2003)
6. Chandra, A.K., Kozen, D.C., Stockmeyer, L.J.: Alternation. J. ACM **28**(1), 114–133 (Jan 1981)
7. Chang, E.Y., Manna, Z., Pnueli, A.: Characterization of temporal property classes. In: Kuich, W. (ed.) Automata, Languages and Programming, 19th International Colloquium, ICALP92, Vienna, Austria, July 13-17, 1992, Proceedings. Lecture Notes in Computer Science, vol. 623, pp. 474–486. Springer (1992)
8. Chrobak, M.: Finite automata and unary languages. Theoretical Computer Science **47**, 149–158 (1986)
9. Cohen, J., Perrin, D., Pin, J.E.: On the expressive power of temporal logic. Journal of computer and System Sciences **46**(3), 271–294 (1993)
10. Cohen-Chesnot, J.: On the expressive power of temporal logic for infinite words. Theoretical Computer Science **83**(2), 301–312 (1991)
11. Colcombet, T., Zdanowski, K.: A tight lower bound for determinization of transition labeled Büchi automata. In: International Colloquium on Automata, Languages, and Programming. pp. 151–162. Springer (2009)
12. Diekert, V., Gastin, P.: First-order definable languages. In: Logic and Automata: History and Perspectives [in Honor of Wolfgang Thomas]. Texts in Logic and Games, vol. 2, pp. 261–306 (2008)
13. Eilenberg, S.: Automata, Languages, and Machines Volume B. Academic Press, Inc., USA (1976)
14. Etessami, K., Vardi, M.Y., Wilke, T.: First-order logic with two variables and unary temporal logic. Inf. Comput. **179**(2), 279–295 (2002)
15. Gabbay, D., Pnueli, A., Shelah, S., Stavi, J.: On the temporal analysis of fairness. In: Proc. of POPL. p. 163–173. New York, NY, USA (1980)
16. Geffert, V., Mereghetti, C., Pighizzini, G.: Complementing two-way finite automata. Information and Computation **205**(8), 1173–1187 (2007)
17. Kamp, J.A.W.: Tense logic and the theory of linear order. University of California, Los Angeles (1968)
18. Kupferman, O., Rosenberg, A.: The blowup in translating LTL to deterministic automata. In: Proc. of Model Checking and Artificial Intelligence. pp. 85–94 (2010)
19. Kupferman, O., Ta-Shma, A., Vardi, M.Y.: Counting with automata. In: Proc. of LICS (1999)
20. Kupferman, O., Vardi, M.Y.: Weak alternating automata are not that weak. ACM Transactions on Computational Logic (TOCL) **2**(3), 408–429 (2001)
21. Ladner, R.E.: Application of model theoretic games to discrete linear orders and finite automata. Information and Control **33**(4), 281–303 (1977)

22. Leiss, E.: Succinct representation of regular languages by boolean automata. Theoretical computer science **13**(3), 323–330 (1981)
23. Löding, C.: Optimal bounds for transformations of ω-automata. In: Rangan, C.P., Raman, V., Ramanujam, R. (eds.) Foundations of Software Technology and Theoretical Computer Science. pp. 97–109. Springer Berlin Heidelberg, Berlin, Heidelberg (1999)
24. Maler, O.: On the Krohn-Rhodes cascaded decomposition theorem. In: Time for Verification, Essays in Memory of Amir Pnueli. Lecture Notes in Computer Science, vol. 6200, pp. 260–278. Springer (2010)
25. Maler, O., Pnueli, A.: Tight bounds on the complexity of cascaded decomposition of automata. In: Proc. of FOCS. pp. 672–682 (1990)
26. Maler, O., Pnueli, A.: On the cascaded decomposition of automata, its complexity and its application to logic. Unpublished. Available at: http://www-verimag.imag.fr/~maler/Papers/decomp.pdf (1994)
27. Manna, Z., Pnueli, A.: A hierarchy of temporal properties. In: PODC. pp. 377–410. ACM (1990)
28. Maretic, G.P., Dashti, M.T., Basin, D.A.: LTL is closed under topological closure. Inf. Process. Lett. **114**(8), 408–413 (2014)
29. Markey, N.: Temporal logic with past is exponentially more succinct. Bull. EATCS **79**, 122–128 (2003)
30. McNaughton, R., Papert, S.A.: Counter-Free Automata (MIT research monograph no. 65). The MIT Press (1971)
31. Michel, M.: Complementation is more difficult with automata on infinite words. CNET, Paris **15** (1988)
32. Muller, D.E., Saoudi, A., Schupp, P.E.: Weak alternating automata give a simple explanation of why most temporal and dynamic logics are decidable in exponential time. In: Proceedings Third Annual Symposium on Logic in Computer Science. pp. 422–423. IEEE Computer Society (1988)
33. Perrin, D.: Recent results on automata and infinite words. In: International Symposium on Mathematical Foundations of Computer Science. pp. 134–148. Springer (1984)
34. Safra, S.: Complexity of automata on infinite objects. Ph.D. thesis, Weizmann Institute, Rehovot, Israel (1989)
35. Schewe, S.: Büchi Complementation Made Tight. In: Albers, S., Marion, J.Y. (eds.) Proc. of 26th International STACS. Leibniz International Proceedings in Informatics (LIPIcs), vol. 3, pp. 661–672 (2009)
36. Sickert, S., Esparza, J.: An efficient normalisation procedure for linear temporal logic and very weak alternating automata. In: LICS. pp. 831–844. ACM (2020)
37. Thomas, W.: Star-free regular sets of ω-sequences. Information and Control **42**(2), 148–156 (1979)
38. Thomas, W.: A combinatorial approach to the theory of ω-automata. Information and Control **48**(3), 261–283 (1981)
39. Thomas, W.: Automata on infinite objects. In: Formal Models and Semantics, pp. 133–191. Elsevier (1990)
40. Vardi, M., Wolper, P.: An automata-theoretic approach to automatic program verification. In: Proc. of LICS. pp. 332–344 (1986)
41. Wilke, T.: Classifying discrete temporal properties. In: Annual symposium on theoretical aspects of computer science. pp. 32–46. Springer (1999)
42. Wilke, T.: Past, present, and infinite future. In: 43rd International Colloquium on Automata, Languages, and Programming (ICALP 2016). Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik (2016)

43. Wilke, T.: Backward deterministic Büchi automata on infinite words. In: 37th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2017). Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik (2018)
44. Wolper, P.: Temporal logic can be more expressive **56**(1–2), 72–99 (1983)
45. Zuck, L.D.: Past Temporal Logic. Ph.D. thesis, The Weizmann Institute of Science, Israel (Aug 1986)

A Omitted Proofs

A.1 Proofs from Section 3

Proposition 1 (Extended Wolper's theorem, Folklore). *Consider an LTL formula φ with $\mathsf{depth}(\varphi) = n$ over the atomic propositions AP, and let $\Sigma = 2^{AP}$. Then for every words $u \in \Sigma^*$, $v \in \Sigma^+$ and $t \in \Sigma^\omega$, and numbers $i, j > n$, φ has the same truth value on the words $(uv^i t)$ and $(uv^j t)$.*

Proof. Consider an LTL formula φ and words $w_i = (uv^i t)$ and $w_j = (uv^j t)$, such that $i, j > \mathsf{depth}(\varphi)$. We prove the claim by induction on the structure of φ.

Base case: φ is an atomic proposition or a Boolean constant. Indeed, $\mathsf{depth}(\varphi) = 0$ and we have that $w_i \models \varphi$ iff $w_j \models \varphi$, because the first letter of these two words, which is the first letter of u if u is not empty and the first letter of v otherwise, is the same.

Induction step: We assume that the claim holds for all strict subformulas of φ. Let ψ_1 and ψ_2 be strict subformulas of φ. We show that the claim holds for:

- $\varphi = \neg\psi_1$: Since $\mathsf{depth}(\varphi) = \mathsf{depth}(\psi_1)$, it follows that $i, j > \mathsf{depth}(\psi_1)$, and therefore by the induction assumption $w_i \models \psi_1$ iff $w_j \models \psi_1$, implying that $w_i \models \varphi$ iff $w_j \models \varphi$.
- $\varphi = \psi_1 \wedge \psi_2$: Since $\mathsf{depth}(\varphi) = \max(\mathsf{depth}(\psi_1), \mathsf{depth}(\psi_2))$, it follows that $i, j > \mathsf{depth}(\psi_1)$ and $i, j > \mathsf{depth}(\psi_2)$. Therefore by the induction assumption $(w_i \models \psi_1$ iff $w_j \models \psi_1)$ and $(w_i \models \psi_2$ iff $w_j \models \psi_2)$. Hence, $w_i \models \psi_1 \wedge \psi_2$ iff $w_j \models \psi_1 \wedge \psi_2$.
- $\varphi = \mathbf{X}\psi_1$: Recall that a word w satisfies φ iff w^1 satisfies ψ_1. Observe that $w_i^1 = u'v^{i-1}t$ and $w_j^1 = u'v^{j-1}t$, where $u' = (uv)^1$. Since $\mathsf{depth}(\psi_1) = \mathsf{depth}(\varphi) - 1$, it follows that $i-1, j-1 > \mathsf{depth}(\psi_1)$. Hence, by the induction assumption $w_i^1 \models \psi_1$ iff $w_j^1 \models \psi_1$, and therefore $w_i \models \varphi$ iff $w_j \models \varphi$.
- $\varphi = \psi_1 \mathbf{U} \psi_2$: We will show that if w_i satisfies φ then so does w_j.
  If w_i satisfies φ then there is a position p of w_i, such that $w_i^p \models \psi_2$ and for every $k < p$, $w_i^k \models \psi_1$. Let o be the position of w_i that appears at the beginning of the v-block that is $\mathsf{depth}(\varphi)$ blocks of v before the t part of w_i, namely $o = |uv^{i-\mathsf{depth}(\varphi)}|$. (See Figure 2.)

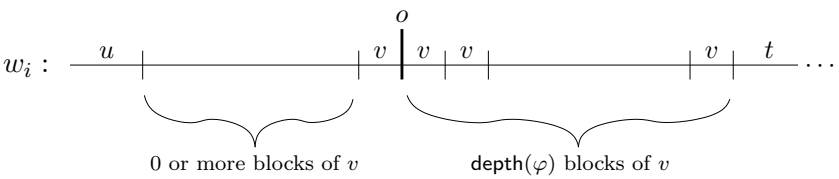

Figure 2. The structure of the word w_i from the proof of Proposition 1.

We split the proof into disjoint cases, depending on the location of p within w_i.

- $p < |u|$: Let u' be the infix of w_j from p to the end of u, namely $u' = w_i[p..|u|-1]$. Then $w_i^p = u'v^it$ and $w_j^p = u'v^jt$. Since $i, j > \mathsf{depth}(\varphi) > \mathsf{depth}(\psi_2)$, by the induction assumption $w_i^p \models \psi_2$ iff $w_j^p \models \psi_2$ and therefore $w_j^p \models \psi_2$. Likewise, since $i, j > \mathsf{depth}(\varphi) > \mathsf{depth}(\psi_1)$, by the induction assumption for every position $m < p$, $w_i^m \models \psi_1$ iff $w_j^m \models \psi_1$ and therefore $w_j^m \models \psi_1$.
- $p \in [|u|..o-1]$: Let p' be the position in w_j that appears in the first v-block after u and that is located within that v-block like p is located within its v-block. That is, $p' = |u| + ((p-|u|) \mod |v|)$. Let u' be the remaining suffix in the v-block of p and p', that is $u' = w_j[p'+1..|u|+|v|]$. Let h be the number of v-blocks that appear after p and before the t part of w_i, that is $h = \mathsf{depth}(\varphi) + \lfloor (o-p)/|v| \rfloor$. Then $w_i^p = u'v^ht$ and $w_j^{p'} = u'v^{j-1}t$. Since $h \geq \mathsf{depth}(\varphi) > \mathsf{depth}(\psi_2)$ and $j-1 \geq \mathsf{depth}(\varphi) > \mathsf{depth}(\psi_2)$, we have by the induction assumption that $w_j^{p'} \models \psi_2$ iff $w_i^p \models \psi_2$, and therefore $w_j^{p'} \models \psi_2$.
  Now, for every position $m < p'$, let u' be the infix of w_j from m to the end of the first v-block of w_j, that is $u' = w_j[m..|u|+|v|]$. Then $w_i^m = u'v^{i-1}t$ and $w_j^m = u'v^{j-1}t$. Since $i-1 > \mathsf{depth}(\psi_1)$ and $j-1 > \mathsf{depth}(\psi_1)$, by the induction assumption $w_i^m \models \psi_1$ iff $w_j^m \models \psi_1$ and therefore $w_j^m \models \psi_1$.
- $p \geq o$: Let p' and o' be the positions in w_j that are at the same distance from the t part of w_j as p and o are from the t part w_i, namely $o' = |uv^{j-\mathsf{depth}(\varphi)}|$ and $p' = o' + (p-o)$. Observe that $w_i^o = w_j^{o'}$ and $w_j^{p'} = w_i^p$, implying that $w_j^{p'} \models \psi_2$.
  Further, for every position $m \in [o-|v|..p]$ of w_i, let m' be the corresponding position in w_j, namely $m' = o' + (m-o)$. Then $w_j^{m'} = w_i^m$, and accordingly $w_j^{m'} \models \psi_1$.
  Now, for every position $m' \in [|u|..o-|v|-1]$, we have by the induction assumption that $w_j^{m'} \models \psi_1$ iff $w_j^{m'+|v|} \models \psi_1$ (as both words have the same prefix until the end of the first v block, followed by at least $\mathsf{depth}(\psi_1)+1$ blocks of v and then t), implying that $w_j^{m'} \models \psi_1$.
  Finally, for every position $m \in [0..|u|-1]$, we have by the induction assumption that $w_i^m \models \psi_1$ iff $w_j^m \models \psi_1$, implying that $w_j^m \models \psi_1$.

□

Proposition 2. *Given an LTL formula φ with $\mathsf{depth}(\varphi) = n$ on finite words over the unary alphabet $\{a\}$, $a^i \in L(\varphi)$ for all $i > n$ or $a^i \notin L(\varphi)$ for all $i > n$.*

Proof. Let $\# = \{p\}$ be the shorthand for a new atomic proposition p. Given a unary LTL formula recognising $L \subseteq \{a\}^+$, it can be turned into a general LTL formula of linear length that recognises $L' = \{v\#^\omega \mid v \in L\}$ over the alphabet $\Sigma' = 2^{\{p\}}$. Then, the statement is a direct consequence of Proposition 1. □

Proposition 3. *Consider a language $L \subseteq \{a\}^+$ that agrees on all words of length over n, that is, has the same truth value on all such words. Then there is an LTL formula of size in $O(n)$ with language L.*

Proof. The very weak deterministic automaton for L consists of $n+2$ states $\{0,\ldots,n+1\}$, with an a-transition from state i to state $i+1$ and a self loop at state $n+1$. The state i is accepting whenever $a^i \in L$. □

Theorem 1. *The size blow-up involved in translating deterministic, nondeterministic, and alternating automata on finite unary words to LTL, when possible, is $\Theta(n)$, $\Theta(n^2)$, and $\Theta(2^n)$, respectively.*

Proof. Deterministic automata: For the upper bound, consider a DFA $\mathcal{A}$ recognising an LTL definable language. By a pumping argument, if $w \in L(\mathcal{A})$ for some w longer than n, then $L(\mathcal{A})$ is infinite. Considering the dual automaton $\mathcal{A}'$ of $\mathcal{A}$, recognising the complement language, by the pumping argument if $w \notin L(\mathcal{A})$ for some w longer than n, then the complement of $L(\mathcal{A})$ is infinite. Hence, by Proposition 2, $L(\mathcal{A})$ agrees on all words of length more than n, and by Proposition 3 there is an LTL formula for it of length in $O(n)$. As for the lower bound, since there is a DFA of size n recognising a^k, it directly follows from Proposition 1.

Nondeterministic automata: Directly follows from Lemmas 1 and 2.

Alternating automata: Directly follows from Lemma 3 and Proposition 5. □

Lemma 1. *Every alternating automaton with n states that recognises an LTL-expressible language $L \subseteq \{a\}^+$ is equivalent to an LTL formula of size in $O(2^n)$.*

Proof. First recall that the run of an alternating automaton is a tree, of which all paths are accepting if and only if the run itself is accepting. These runs can be pumped in the same way as runs of nondeterministic automata, except that the run to be pumped must be of length over 2^n to guarantee that the *set of states* in a cross-section of the run are repeated.

Then, by a pumping argument, if $w \in L(\mathcal{A})$ for some w longer than 2^n, then $L(\mathcal{A})$ is infinite: indeed, let $\mathcal{A}'$ be an NFA equivalent to $\mathcal{A}$ of size at most 2^n; if $\mathcal{A}'$ accepts a word longer than 2^n, then its run sees some state more than once and can be pumped to build infinitely many accepting runs. Dually, since AFA are easy to complement, if $w' \notin L(\mathcal{A})$ for some w' longer than 2^n, then $\Sigma^* \setminus L(\mathcal{A})$ is infinite. The existence of both $w \in L(\mathcal{A})$ and $w' \notin L(\mathcal{A})$ longer than 2^n therefore contradicts Lemma 2. We conclude that $L(A)$ agrees on all words of length over 2^n. Thus, by Proposition 3 there is an LTL formula for $L(\mathcal{A})$ of length in $O(2^n)$. □

Lemma 2 (Adaptation of [22, proof of Theorem 1]). *For every $n \in \mathbb{N}\setminus\{0\}$, there is a weak alternating automaton with $2n$ states and transition function of size in $O(n)$ recognising the language $\{a^{2^{n-1}}\}$.*

Proof. The idea of the construction is that it represents an n-bit up-counter, having two states for each bit, one corresponding to 1, one to 0 (see Fig. 1).

Given a way to resolve the nondeterministic choices, the resulting set of "active" states of the automaton represents a configuration of the counter: The nondeterminism in each bit-state chooses whether to change the bit's value (going left in Fig. 1), in which case the universality ensures that all lower bits are

set to 0, or to preserve the bit's value (going right in Fig. 1), in which case the universality ensures that at least one lower bit is set to 1.

The automaton thus preserves the invariant that a correct update (for example from 011 to 100) ensures that the number of active states is constant, in particular, if all updates are correct, then exactly one state per bit is active at a time; an incorrect update on the other hand increases the number of active states. The set of accepting states corresponds to the bit-configuration for 2^{n-1}, where a transition to a state $q_{i,\mathbf{0}}$, for every $i < n$, can be changed to a move to q_{acc}, provided that it is on the last letter of the input word. (There is no transition out of q_{acc}). The only way to build an accepting run is to correctly update the counter at each step; an incorrect update ensures that both states of some bit are set from thereon, of which one must be rejecting.

As for the automaton size, which is the number of subformulas in the transition function, observe that it is linear in n, since there are $2n$ states and the total number of subformulas in the transition function is as follows: The transition functions of $q_{1,\mathbf{0}}$ and $q_{1,\mathbf{1}}$ have together four subformulas, and the transition function of every other state $q_{(i,\cdot)}$ adds up to 4 subformulas: (i) the topmost disjunction of whether to change the bit's value or not; (ii)&(iii) the universality involved in each of these two options; and (iv) within each of the two universality subformulas – a disjunction or conjunction between $q_{(i-1,\cdot)}$ and $(q_{(i-1,\cdot)}, \ldots, q_{(i-1,\cdot)})$, where the latter subformula (within the parenthesis) is already used in the transition function of $q_{(i-1,\cdot)}$ and these two formulas (one of conjunction and one of disjunction) is used by both $q_{i,\mathbf{0}}$ and $q_{i,\mathbf{1}}$ (except for $q_{n,\mathbf{1}} = q_{acc}$, which has no outgoing transitions). □

Lemma 3. *Every nondeterministic automaton with n states recognising an LTL-expressible language $L \subseteq \{a\}^+$ is equivalent to an LTL formula of size in $O(n^2)$.*

Proof. If $L(\mathcal{A})$ is finite, then by a pumping argument, $\mathcal{A}$ only accepts words up to length n, and by Proposition 3 we are done. We consider co-finite $L(\mathcal{A})$.

We will argue using 2-way deterministic automata, which are deterministic automata that process words of the form $\vdash w \dashv$, where $\vdash$ and $\dashv$ are start- and end-of-word markers respectively, and where transitions specify whether to read the letter to the right or to the left of the current position. They accept by reaching an end state, and reject by reaching a rejecting state or by failing to terminate. For a more formal definition, see [16]. We will use the facts that a unary NFA can be turned into a 2-way DFA of size $O(n^2)$ [8], that a 2-way DFA of size m can be complemented into one of size $4m$ [16], and that the smallest 2-way DFA recognising $\{a^k\}$ is of length $k + 2$ [1].

The NFA $\mathcal{A}$ can be turned into a 2-way DFA $\mathcal{D}$ with $O(n^2)$ states recognising the same language. From $\mathcal{D}$, we can obtain a 2-way DFA $\mathcal{D}'$ that recognises $a^* \setminus \{a^k\}$, where a^k is the longest word not in $L(\mathcal{A})$, as follows: we keep the states and transitions involved in the run of $\mathcal{D}$ on $\vdash a^k \dashv$, which is rejecting and must read $\dashv$ (since, by the co-finiteness of $L(\mathcal{A})$, there are accepted words longer than a^k). Notice that states in $\mathcal{D}'$ may have only some of the transitions they had in $\mathcal{D}$ – only those that are involved in the run on a^k. We then add an a-

transition and a $\dashv$-transition from all states without such transitions, and these all lead to the (accepting) end state.

Notice that $\mathcal{D}'$ is still deterministic and not larger than $\mathcal{D}$. It still rejects a^k, on which it has the same run as $\mathcal{D}$, but accepts all other words, which are either shorter than a^k and therefore accepted via one of the new $\dashv$-transitions, or are longer than a^k and accepted when they read the $k+1^{th}$ letter.

We can then complement $\mathcal{D}'$ into a 2-way DFA $\mathcal{D}''$ of size linear in the size of D', namely in $O(n^2)$, that recognises $\{a^k\}$. However, since $\mathcal{D}''$ must be of size at least $k+2$, we get that $k+2$ is at most in $O(n^2)$, meaning that the longest word rejected by $\mathcal{A}$, which is of length k, is of length in $O(n^2)$.

Then, there is an equivalent very weak automaton of the size of the longest word not in $L(\mathcal{A})$, from Lemma 2. □

A.2 Proofs from Section 4.1

Proposition 7. *Let $\mathcal{D}$ be a deterministic Büchi, coBüchi or Rabin automaton, with a semiautomaton homomorphic to a cascade $\mathcal{A}$. There is respectively a deterministic Büchi, coBüchi or Rabin automaton $\mathcal{D}'$ equivalent to $\mathcal{D}$ with semiautomaton $\mathcal{A}$. For Rabin, $\mathcal{D}$ and $\mathcal{D}'$ have the same number of acceptance pairs.*

Proof. We provide the proof for Rabin automata. The proofs for Büchi and coBüchi automata are special cases of the Rabin case.

Let $\mathcal{D} = (\Sigma, Q, \iota, \delta, \alpha)$, and h be the mapping via which (Σ, Q, δ) is homomorphic to $\mathcal{A}$. We shall define a configuration ι' of $\mathcal{A}$, and a set α' of pairs of sets of configurations of $\mathcal{A}$ that will provide the initial state and acceptance condition of $\mathcal{D}'$, making it equivalent to $\mathcal{D}$.

For the initial state of $\mathcal{D}'$, one can choose any configuration $\iota' \in h^{-1}(\iota)$: A run of $\mathcal{A}$ starting in ι' corresponds via h to a run of $\mathcal{D}$ starting in $h(\iota') = \iota$.

As for the acceptance condition, consider the run r' of $\mathcal{D}'$ on a word w, starting in ι', and let $r = h(r')$ be the corresponding run of $\mathcal{D}$ on w. (With $h(r')$, we mean the sequence of states of $\mathcal{D}$, obtained by mapping with h each configuration of r'.) Then, r' should be accepting iff r is.

Recall that r is accepting iff $inf(r) \cap G \neq \emptyset$ and $inf(r) \cap B = \emptyset$ for some pair $(G, B) \in \alpha$. Hence, r' should be accepting "according to (G, B)" iff $h(inf(r')) \cap G \neq \emptyset$ and $h(inf(r')) \cap B = \emptyset$.

Thus, r' should visit infinitely often some configuration in $G' = h^{-1}(G)$ and finitely often every configuration in $B' = h^{-1}(B)$. Hence, there are k acceptance pairs (G', B') in α', each corresponding to an acceptance pair (G, B) in $\mathcal{D}$. □

Proposition 8. *Consider a deterministic Muller automaton $\mathcal{D}$ with n states, whose semiautomaton is homomorphic to a reset cascade $\mathcal{A}$ with m configurations. Then there is a deterministic Muller automaton $\mathcal{D}'$ equivalent to $\mathcal{D}$, whose semiautomaton is $\mathcal{A}$ and its Muller condition has up to $2^{O(mn)}$ acceptance sets.*

Proof. Let $\mathcal{D} = (\Sigma, Q, \iota, \delta, \alpha)$, and h be the mapping via which (Σ, Q, δ) is homomorphic to $\mathcal{A}$. We shall define a configuration ι' of $\mathcal{A}$, and a set α' of sets

of configurations of $\mathcal{A}$ that will provide the initial state and acceptance condition of $\mathcal{D}'$, making it equivalent to $\mathcal{D}$.

For the initial state of $\mathcal{D}'$, one can choose any configuration $\iota' \in h^{-1}(\iota)$: A run of $\mathcal{A}$ starting in ι' corresponds via h to a run in $\mathcal{D}$ starting in $h(\iota') = \iota$.

As for the acceptance condition, consider the run r' of $\mathcal{D}'$ on w, starting in ι', and let $r = h(r')$ be the run of $\mathcal{D}$ on w (that is defined by mapping each configuration of r' to a state of $\mathcal{D}$ via h). Then, r' should be accepting iff r is.

Recall that r is accepting iff $inf(r) = M$, for some $M = \{q_1, \dots q_l\} \in \alpha$. Hence, r' should be accepting "according to M" iff $h(inf(r')) = M$. Thus, r' should visit finitely often every configuration in $h^{-1}(Q \setminus M)$, and for *every* $i \in [1..l]$, visit *some* configurations in $G_i = h^{-1}(q_i)$ infinitely often.

Since we should consider every choice of configurations in $G_i = h^{-1}(q_i)$, where $|G_i| \leq m$, there are up to 2^m choices for G_i, and therefore up to $(2^m)^n$ choices for M, each providing a Muller set G of configurations to be visited infinitely often.

As there are up to 2^n sets in α, we end up with up to $2^n(2^m)^n \in 2^{O(mn)}$ sets in α'. □

A.3 Proofs from Section 4.2

Lemma 4. *The intended semantics of Table 1 hold for all infinite words $w \in \Sigma^\omega = (2^{AP})^\omega$, configurations S, B, T of level $m \leq n$, states s, b, t in level $m+1$ (when $m < n$), and LTL formulas β and τ over AP.*

Proof. Observe first that there is no circularity in the definitions of the five reachability formulas, even though they are defined by each other: Formula 2 is defined on top of formula 1, which is defined on top of formulas 3 and 5, while formulas 3, 4, and 5 are defined with respect to reachability formulas over configurations of a lower level.

We prove the statement by induction on the level m of the configurations S, B, T in the reachability formulas. We split the proof to five cases corresponding to the five reachability formulas in Table 1. In order to clarify which equivalence of the induction hypothesis is used we denote by (I.H.1) the equivalence in Table 1 for reachability formula 1, (I.H.2) the equivalence for reachability formula 2, and so on.

Reachability formula 1
($m = 0$): There is exactly one configuration of level 0 and it is the empty configuration. Thus $S = T = B = \langle\rangle$. We then derive:

$$
\begin{aligned}
& w \models \langle\rangle \xrightarrow[\langle\rangle(\beta)]{} \langle\rangle(\tau) \\
\iff\ & w \models (\neg\beta)\mathbf{U}\tau \\
\iff\ & \exists i \geq 0.\ w_{[i..]} \models \tau \wedge \forall j \in [0..i).\ w_{[j..]} \not\models \beta \\
\iff\ & \exists i \geq 0.\ \delta(\langle\rangle, w_{[0..i)}) = \langle\rangle \wedge w_{[i..]} \models \tau \\
& \qquad \wedge \forall j \in [0..i).\ \delta(\langle\rangle, w_{[0..j)}) \neq \langle\rangle \vee w_{[j..]} \not\models \beta
\end{aligned}
$$

$(m \to m+1)$: Let $\langle S, s\rangle, \langle T, t\rangle, \langle B, b\rangle \in Q_1 \times \cdots \times Q_{m+1}$ be arbitrary configurations of level $m+1$. Thus we can use the equalities from Table 1 for all configurations of $Q_1 \times \cdots \times Q_m$ as induction hypotheses. We need to show that:

$$
\begin{aligned}
& w \models \langle S, s\rangle \overset{\langle B,b\rangle(\beta)}{\rightsquigarrow} \langle T, t\rangle\,(\tau) \\
\iff\ & w \models \langle S, s\rangle \xrightarrow[\langle B,b\rangle(\beta)]{} \langle T, t\rangle\,(\tau) \vee \langle S, s\rangle \overset{\langle B,b\rangle(\beta)}{\dashrightarrow} \langle T, t\rangle\,(\tau) \\
\iff\ & \exists i \geq 0.\ \delta(\langle S, s\rangle, w_{[0..i)}) = \langle T, t\rangle \wedge w_{[i..]} \models \tau \\
& \qquad \wedge\ (\forall j \in [0..i).\ \delta(\langle S, s\rangle, w_{[0..j)}) \neq \langle B, b\rangle \vee w_{[j..]} \not\models \beta)
\end{aligned}
$$

We split this into the $(\Rightarrow)$- and $(\Leftarrow)$-direction:

$(\Rightarrow)$: We further refine this and first assume that the first disjunct is satisfied by w and defer the other case to a later point in the proof. We then apply (I.H.3) and derive:

$$
\begin{aligned}
& w \models \langle S, s\rangle \xrightarrow[\langle B,b\rangle(\beta)]{} \langle T, t\rangle\,(\tau) \\
\iff\ & \exists i \geq 0.\ \delta(\langle S, s\rangle, w_{[0..i)}) = \langle T, t\rangle \wedge w_{[i..]} \models \tau \\
& \qquad \wedge\ (\forall j \in [0..i).\ \delta(\langle S, s\rangle, w_{[0..j)}) \neq \langle B, b\rangle \vee w_{[j..]} \not\models \beta) \\
& \qquad \wedge\ (\forall j \in [0..i).\ \langle w[j], \delta(S, w_{[0..j)})\rangle \in \mathsf{Stay}(s)) \\
\implies\ & \exists i \geq 0.\ \delta(\langle S, s\rangle, w_{[0..i)}) = \langle T, t\rangle \wedge w_{[i..]} \models \tau \\
& \qquad \wedge\ (\forall j \in [0..i).\ \delta(\langle S, s\rangle, w_{[0..j)}) \neq \langle B, b\rangle \vee w_{[j..]} \not\models \beta)
\end{aligned}
$$

We now assume that the first disjunct is *not* satisfied by w and thus the second disjunct is satisfied by w. We then derive using (I.H.5):

$$
\begin{aligned}
& w \models \langle S, s\rangle \overset{\langle B,b\rangle(\beta)}{\dashrightarrow} \langle T, t\rangle\,(\tau) \\
\iff\ & \exists i_1, i_2 \geq 0.\ \delta(\langle S, s\rangle, w_{[0..i_1)}) = \langle T, t\rangle \wedge w_{[i_1..]} \models \tau \\
& \qquad \wedge\ (\exists j_1 \in [0..i_1).\ \langle w[j_1], \delta(S, w_{[0..j_1)})\rangle \in \mathsf{Enter}(t) \\
& \qquad \wedge\ \langle w[i_2], \delta(S, w_{[0..i_2)})\rangle \in \mathsf{Leave}(s) \\
& \qquad \wedge\ (\forall j \in [0..\max(i_1{-}1, i_2)].\ \delta(\langle S, s\rangle, w_{[0..j)}) \neq \langle B, b\rangle \vee w_{[j..]} \not\models \beta) \\
\implies\ & \exists i \geq 0.\ \delta(\langle S, s\rangle, w_{[0..i)}) = \langle T, t\rangle \wedge w_{[i..]} \models \tau \\
& \qquad \wedge\ (\forall j \in [0..i).\ \delta(\langle S, s\rangle, w_{[0..j)}) \neq \langle B, b\rangle \vee w_{[j..]} \not\models \beta)
\end{aligned}
$$

$(\Leftarrow)$: We assume that w satisfies the right-hand side of the equation and we instantiate i to be the smallest non-negative integer such that:

$$
\begin{aligned}
& \delta(\langle S, s\rangle, w_{[0..i)}) = \langle T, t\rangle \wedge w_{[i..]} \models \tau \\
& \qquad \wedge\ (\forall j \in [0..i).\ \delta(\langle S, s\rangle, w_{[0..j)}) \neq \langle B, b\rangle \vee w_{[j..]} \not\models \beta)
\end{aligned}
$$

Assume that for all proper prefixes of $w_{[0..i)}$ the combined letter stays in s, i.e., $\langle w[j], \delta(S, w_{[0..j)})\rangle \in \mathsf{Stay}(s)$ for all $j \in [0..i)$. We then apply (I.H.3) and obtain that w satisfies $\langle S, s\rangle \xrightarrow[\langle B,b\rangle(\beta)]{} \langle T, t\rangle\,(\tau)$ and so the left-hand side of

the equation is satisfied by w. Thus let $k \in [0..i)$ be the smallest non-negative integer such that:

$$\langle w[k], \delta(S, w_{[0..k)})\rangle \in \mathsf{Leave}(s)$$

Since $k < i$, we immediately obtain one of the missing preconditions for applying (I.H.5):

$$\forall j_2 \in [0..k].\ \delta(\langle S, s\rangle, w_{[0..j_2)}) \neq \langle B, b\rangle \vee w_{[j_2..]} \not\models \beta$$

The second missing precondition is that we need to find a $j_1 \in [0..i)$ such that $\langle w[j_1], \delta(S, w_{[0..j_1)})\rangle \in \mathsf{Enter}(t)$: In the case $s \neq t$ this is straightforward, since after reading $w_{[0..i)}$ we reach $\langle T, t\rangle$, and thus there must be a $j_1 < i$ such that $\langle w[j_1], \delta(S, w_{[0..j_1)})\rangle \in \mathsf{Enter}(t)$. Thus let us assume $s = t$. In general there might be not such an index j_1. However, we have shown that $\langle w[k], \delta(S, w_{[0..k)})\rangle \in \mathsf{Leave}(s)$ for some $k < i$ and by the same reasoning as above there must be j_1 between k and i such that $\langle w[j_1], \delta(S, w_{[0..j_1)})\rangle \in \mathsf{Enter}(t)$.

We now can apply (I.H.5) and conclude this direction of the proof.

Reachability formula 2

We proceed by a straightforward derivation for which we use (I.H.1) in the second step:

$$\begin{aligned}
& w \models S \overset{\text{weak}}{\rightsquigarrow} T\,(\tau) \\
\iff\ & w \not\models S \rightsquigarrow B\,(\beta) \\
\iff\ & \neg\big(\exists i \geq 0.\ \delta(S, w_{[0..i)}) = B \wedge w_{[i..]} \models \beta \\
& \qquad \wedge\ (\forall j \in [0..i).\ \delta(S, w_{[0..j)}) \neq T \vee w_{[j..]} \not\models \tau)\big) \\
\iff\ & \forall i \geq 0.\ (\delta(S, w_{[0..i)}) = B \wedge w_{[i..]} \models \beta) \\
& \qquad \rightarrow\ (\exists j \in [0..i).\ \delta(S, w_{[0..j)}) = T \wedge w_{[j..]} \models \tau)
\end{aligned}$$

Reachability formula 3

We want to prove the following equivalence:

$$\begin{aligned}
& w \models \langle S, s\rangle \xrightarrow[\langle B,b\rangle(\beta)]{} \langle T, t\rangle\,(\tau) \\
\iff\ & \exists i \geq 0.\ \delta(\langle S, s\rangle, w_{[0..i)}) = \langle T, t\rangle \wedge w_{[i..]} \models \tau \\
& \qquad \wedge\ (\forall j_1 \in [0..i).\ \delta(\langle S, s\rangle, w_{[0..j_1)}) \neq \langle B, b\rangle \vee w_{[j_1..]} \not\models \beta) \\
& \qquad \wedge\ (\forall j_2 \in [0..i).\ \langle w[j_2], \delta(S, w_{[0..j_2)})\rangle \in \mathsf{Stay}(s))
\end{aligned}$$

($\Rightarrow$): Assume that w satisfies the left-hand side of the equivalence. Notice that if w does not satisfy $\varphi := \langle S, s\rangle \xrightarrow[\langle B,b\rangle(\beta)]{>0} \langle T, t\rangle\,(\tau)$, then necessarily $\delta(\langle S, s\rangle, w_{[0..0)}) = \langle T, t\rangle$ and $w_{[0..]} \models \tau$ and thus the right-hand side trivially holds for $i = 0$.

Thus we can assume that w satisfies φ and using the same reasoning we know that either $\delta(\langle S, s\rangle, w_{[0..0)}) = \langle B, b\rangle$ or $w_{[0..]} \models \beta$ does not hold, which takes care of the case $j_1 = 0$ in the second line of the right-hand side, assuming $i > 0$.

Since we have $w \models \varphi$, there must be a $\langle \sigma, T'\rangle \in \mathsf{Stay}(s)$ with $\delta(\langle T', s\rangle, \sigma) = \langle T, t\rangle$ such that the matching disjunct ψ of φ is satisfied by w. Note that this

immediately implies that $s = t$. Observe that ψ is a conjunction of formulas with the shape

$$S \underset{\not\triangleleft}{\rightsquigarrow} T' \, (\sigma \wedge \mathbf{X}\tau)$$

and we now apply (I.H.1) to all reachability formulas. Since they all share the same target, we can instantiate them to the same i' (where i' is the smallest non-negative integer satisfying the conditions) such that:

(a) $\delta(S, w_{[0..i')}) = T'$
(b) $w[i'] = \sigma$
(c) $w_{[i'+1..]} \models \tau$
(d) For every $\langle \eta, L \rangle \in \mathsf{Leave}(s)$ and every $j \in [0..i')$ at least one of the following statements holds:
  (i) $\delta(S, w_{[0..j)}) \neq L$
  (ii) $w[j] \neq \eta$
(e) For every $\langle \rho, B' \rangle \in \mathsf{Stay}(s)$ such that $\delta(\langle B', s \rangle, \rho) = \langle B, b \rangle$ and every $j \in [0..i')$ at least one of the following statements holds:
  (i) $\delta(S, w_{[0..j)}) \neq B'$
  (ii) $w[j] \neq \rho$
  (iii) $w_{[j+1..]} \not\models \beta$

We now instantiate i of the right-hand side with $i' + 1$. Remember that we have $\langle \sigma, T' \rangle \in \mathsf{Stay}(s)$ and together with (a-c) we obtain the first conjunct of the right hand side.

We now establish the third conjunct. Note that for every $j_2 \in [0..i')$ we have $\langle w[j_2], \delta(\langle S, s \rangle, w_{[0..j_2)}) \rangle \in \mathsf{Stay}(s)$, since $\mathsf{Stay}(s) = (\Sigma \times Q_1 \times \cdots \times Q_m) \setminus \mathsf{Leave}(s)$ and by (d) we do not encounter an element of $\mathsf{Leave}(s)$ for any j_2. Further, for $j_2 = i'$ we obtain $\langle w[j_2], \delta(\langle S, s \rangle, w_{[0..j_2)}) \rangle \in \mathsf{Stay}(s)$ from (a,b) and the choice of $\langle \sigma, T' \rangle \in \mathsf{Stay}(s)$.

For the second conjunct it remains to show that for every $j_1 \in [0..i)$:

$$\delta(\langle S, s \rangle, w_{[0..j_1)}) \neq \langle B, b \rangle \vee w_{[j_1..]} \not\models \beta$$

Assume that $\delta(\langle S, s \rangle, w_{[0..j_1)}) = \langle B, b \rangle$ (if this is not the case we are immediately done). Due to the already established third conjunct, we have $\langle w[j_1 - 1], \delta(S, w_{[0..j_1-1)}) \rangle = \langle \rho, B' \rangle \in \mathsf{Stay}(s)$ and $\delta(\langle B', s \rangle, \rho) = \langle B, b \rangle$. Thus (e,i) and (e,ii) cannot hold and (e,iii) must hold for $j_1 (= j + 1)$.

($\Leftarrow$): We assume that w satisfies the right-hand side and instantiate i as the smallest $i \geq 0$ such that:

(a) $\delta(\langle S, s \rangle, w_{[0..i)}) = \langle T, t \rangle$
(b) $w_{[i..]} \models \tau$
(c) $\forall j \in [0..i).\ \delta(\langle S, s \rangle, w_{[0..j)}) \neq \langle B, b \rangle \vee w_{[j..]} \not\models \beta$
(d) $\forall j \in [0..i).\ \langle w[j], \delta(S, w_{[0..j)}) \rangle \in \mathsf{Stay}(s)$

If $i = 0$, then due to (a-b) we have immediately $w \models \langle S, s\rangle \xrightarrow[\langle B,b\rangle(\beta)]{} \langle T, t\rangle\,(\tau)$. Thus without loss of generality we can assume $i > 0$ from now on. Further, due to (c) we have that either $\langle S, s\rangle \neq \langle B, b\rangle$ or $w \not\models \beta$. Thus it remains to show that $w \models \langle S, s\rangle \xrightarrow[\langle B,b\rangle(\beta)]{>0} \langle T, t\rangle\,(\tau)$. Define σ as $w[i-1]$ and T' as $\delta(S, w_{[0..i-1)})$. From (a,d) we then follow:

(e) $\langle \sigma, T'\rangle \in \mathsf{Stay}(s)$
(f) $\delta(S, w_{[0..i-1)}) = T'$ and $\delta(\langle T', s\rangle, \sigma) = \langle T, t\rangle$
(g) $w_{[i-1..]} \models \sigma \wedge \mathbf{X}\tau$

We are going to use the induction hypothesis to show that w satisfies the disjunct corresponding to $\langle \sigma, T'\rangle$. The first conjunct is satisfied from (f,g) and (I.H.1). From (d) it follows that for every $j \in [0..i)$ the tuple $\langle w[j], \delta(S, w_{[0..j)})\rangle$ is not in $\mathsf{Leave}(s)$. Thus for every $\langle \eta, L\rangle \in \mathsf{Leave}(s)$, we either have $w_{[j..]} \not\models \eta$ or $\delta(S, w_{[0..j)}) \neq L$. Thus by (I.H.1) and (f,g) we obtain that w satisfies the second conjunct. Analogously, from (c) it follows that for every $j \in [1..i)$ and for every $\langle \rho, B'\rangle \in \mathsf{Stay}(s)$ such that $\delta(\langle B', s\rangle, \rho) = \langle B, b\rangle$, either $\delta(S, w_{[0..j-1)}) \neq B'$ or $w_{[j-1..]} \not\models \rho \wedge \mathbf{X}\beta$. Thus by (I.H.1) and (f,g) we obtain that w satisfies the third and final conjunct.

Reachability formula 4
We want to prove the following equivalence:

$$w \models \langle S, s\rangle \xrightarrow[\langle B,b\rangle(\beta)]{\text{weak}} \langle T, t\rangle\,(\tau)$$

$$\begin{aligned}\iff\ & \forall i \geq 0.\ \big((\delta(\langle S, s\rangle, w_{[0..i)}) = \langle B, b\rangle \wedge w_{[i..]} \models \beta)\\ &\qquad \vee (i > 0 \wedge \langle w[i-1], \delta(S, w_{[0..i-1)})\rangle \in \mathsf{Leave}(s))\big)\\ &\quad \rightarrow\ (\exists j \in [0..i).\ \delta(\langle S, s\rangle, w_{[0..j)}) = \langle T, t\rangle \wedge w_{[j..]} \models \tau)\end{aligned}$$

($\Rightarrow$): Assume that w satisfies the left-hand side of the equivalence. Then we immediately obtain from the definition that either $\delta(\langle S, s\rangle, w_{[0..0)}) \neq \langle B, b\rangle$ or $w_{[0..]} \not\models \beta$. Further, notice that if w does not satisfy $\varphi \coloneqq \langle S, s\rangle \xrightarrow[\langle B,b\rangle(\beta)]{\text{weak},>0} \langle T, t\rangle\,(\tau)$, then necessarily $\delta(\langle S, s\rangle, w_{[0..0)}) = \langle T, t\rangle$ and $w_{[0..]} \models \tau$, and thus the right-hand side trivially holds for all i by instantiating j with 0.

Thus we can assume that w satisfies φ and using the fact that at least one of $\delta(\langle S, s\rangle, w_{[0..0)}) = \langle B, b\rangle$ and $w_{[0..]} \models \beta$ does not hold, we only need to prove the right-hand side for $i > 0$.

Since we have $w \models \varphi$, then either w satisfies the disjunct ψ in line (2) or there exists a $\langle \sigma, T'\rangle \in \mathsf{Stay}(s)$ with $\delta(\langle T', s\rangle, \sigma) = \langle T, t\rangle$ such that the matching disjunct χ in (1) is satisfied by w. We assume that the first case holds, and defer the second case to later.

Note, that when applying the induction hypothesis (I.H.2) to each conjunct of ψ, we can simplify the nested existential quantification to **false**, since $w \not\models$ **false**. Thus we obtain for all $i \geq 0$:

(a) For every $\langle \eta, L \rangle \in \mathsf{Leave}(s)$ at least one of the following statements holds:
  (i) $\delta(S, w_{[0..i)}) \neq L$
  (ii) $w[i] \neq \eta$
(b) For every $\langle \rho, B' \rangle \in \mathsf{Stay}(s)$ such that $\delta(\langle B', s \rangle, \rho) = \langle B, b \rangle$ at least one of the following statements holds:
  (i) $\delta(S, w_{[0..i)}) \neq B'$
  (ii) $w[i] \neq \rho$
  (iii) $w_{[i+1..]} \not\models \beta$

Because w and the sequence of generated configurations does not contain a combined letter that leaves s due to (a), we do have $\langle w[i], \delta(S, w_{[0..i)}) \rangle \notin \mathsf{Leave}(s)$ for all $i \geq 0$. Thus $\langle w[i], \delta(S, w_{[0..i)}) \rangle \in \mathsf{Stay}(s)$ for all $i \geq 0$. Further, due to (b) we either do not see a combined letter at position i that enters the configuration $\langle B, b \rangle$ (and thus $\delta(\langle S, s \rangle, w_{[i+1..]}) = \langle B, b \rangle$) or the infinite suffix of $w_{[i+1..]}$ does not satisfy β. From this, we can conclude that the right-hand side holds. (The case for $i = 0$ was already established in the second paragraph.)

We now assume that there exists a tuple $\langle \sigma, T' \rangle$ such that w satisfies the matching disjunct χ of line (1). Note that the existence of $\langle \sigma, T' \rangle$ immediately implies $s = t$. Moreover, we can assume that w does not satisfy the disjunct ψ from line (2). Thus by applying (I.H.2), we know that there exists an $k \geq 0$ such that either $\delta(S, w_{[0..k)}) = L$ and $w[k] = \eta$ from $\langle \eta, L \rangle \in \mathsf{Leave}(s)$ or $\delta(S, w_{[0..k)}) = B$, $w[k] = \rho$, and $w_{[k+1..]} \models \beta$ for some $\langle \rho, B' \rangle \in \mathsf{Stay}(s)$ and $\delta(\langle B', s \rangle, \rho) = \langle B, b \rangle$. Thus when we apply the (I.H.2) to the corresponding reachability formula of χ we obtain a $k' \in [0..k)$ such that $\delta(S, w_{[0..k')}) = T'$ and $w_{[k'..]} \models \sigma \wedge \mathbf{X}\tau$. Observe that χ is a conjunction of formulas with the shape

$$S \overset{\text{weak}}{\rightsquigarrow} T' \, (\sigma \wedge \mathbf{X}\tau)$$

and we now apply the induction hypothesis to all reachability formulas. Since they all share the same "target", we can instantiate the nested existential quantification to the same j' (where j' is the smallest non-negative integer satisfying the condition) such that:

(a) $\delta(S, w_{[0..j')}) = T'$
(b) $w[j'] = \sigma$
(c) $w_{[j'+1..]} \models \tau$
(d) For every $\langle \eta, L \rangle \in \mathsf{Leave}(s)$ and every $i' \in [0..j']$ at least one of the following statements holds:
  (i) $\delta(S, w_{[0..i')}) \neq L$
  (ii) $w[i'] \neq \eta$
(e) For every $\langle \rho, B' \rangle \in \mathsf{Stay}(s)$ such that $\delta(\langle B', s \rangle, \rho) = \langle B, b \rangle$ and every $i' \in [0..j']$ at least one of the following statements holds:
  (i) $\delta(S, w_{[0..i')}) \neq B'$
  (ii) $w[i'] \neq \rho$
  (iii) $w_{[i'+1..]} \not\models \beta$

In order to show that w satisfy the right-hand side of the equation, let now $i \geq 0$ be an arbitrary integer and we need to prove:

$$\big((\delta(\langle S, s\rangle, w_{[0..i)}) = \langle B, b\rangle \wedge w_{[i..]} \models \beta) \vee (i > 0 \wedge \langle w[i{-}1], \delta(S, w_{[0..i)})\rangle \in \mathsf{Leave}(s))\big) \\ \rightarrow (\exists j \in [0..i).\ \delta(\langle S, s\rangle, w_{[0..j)}) = \langle T, t\rangle \wedge w_{[j..]} \models \tau)$$

The case $i = 0$ was already discussed in the second paragraph. Further, if $i > j' + 1$, then (a-c) show that j can be instantiated with $j' + 1$. Thus assume that $i \in [1..(j' + 1)]$. Further, since j' was chosen to be the smallest integer such that (a-c) holds, we can simplify the expression we need to prove to:

$$(\delta(\langle S, s\rangle, w_{[0..i)}) \neq \langle B, b\rangle \vee w_{[i..]} \not\models \beta) \wedge \langle w[i{-}1], \delta(S, w_{[0..i-1)})\rangle \in \mathsf{Stay}(s))$$

Note that second conjunct is direct consequence of (d). For the first conjunct, we proceed by contradiction and assume that i is the smallest integer such that:

$$\delta(\langle S, s\rangle, w_{[0..i)}) = \langle B, b\rangle \wedge w_{[i..]} \models \beta$$

Due to the already established second conjunct, we have $\langle w[i{-}1], \delta(S, w_{[0..i-1)})\rangle = \langle \rho, B'\rangle \in \mathsf{Stay}(s)$ and $\delta(\langle B', s\rangle, \rho) = \langle B, b\rangle$. But since $i - 1 \in [0..j']$, we have a contradiction to (e).

($\Leftarrow$): We assume that w satisfies the right-hand side. We first consider the case that the nested existential quantification is not true for any $i \geq 0$. Thus for all $i \geq 0$ it holds that:

(a) $\langle w[i], \delta(S, w_{[0..i)})\rangle \in \mathsf{Stay}(s)$
(b) $\delta(\langle S, s\rangle, w_{[0..i)}) \neq \langle B, b\rangle$ or $w_{[i..]} \not\models \beta$

Since w satisfies the right-hand side, it is easy to see that we have either $\langle S, s\rangle \neq \langle B, b\rangle$ or $w \not\models \beta$. Hence it remains to show that $w \models \langle S, s\rangle \xrightarrow[\cancel{\langle B,b\rangle(\beta)}]{\text{weak},>0} \langle T, t\rangle\,(\tau)$, which we do by showing that w satisfies (2). From (a,b) we obtain:

(a') For every $\langle \eta, L\rangle \in \mathsf{Leave}(s)$ we have either $\delta(S, w_{[0..i)}) \neq L$ or $w[i] \neq \eta$.
(b') Either $w_{[i+1..]} \not\models \beta$ or for every $\langle \rho, B'\rangle \in \mathsf{Stay}(s)$ such that $\delta(\langle B', s\rangle, \rho) = \langle B, b\rangle$ we have either $\delta(S, w_{[0..i)}) \neq B'$ or $w[i] \neq \rho$.

We then apply the induction hypothesis (I.H.2) to (a',b') and obtain that w satisfies the disjunct of line (2).

We now consider the second case and assume that there is a j such that:

(a) $\delta(\langle S, s\rangle, w_{[0..j)}) = \langle T, t\rangle$
(b) $w_{[j..]} \models \tau$

Without loss of generality, we can assume j to be the smallest integer with such a property. Further, since w satisfies the right-hand side, we have:

(c) $\forall i \in [0..j).\ \langle w[i], \delta(S, w_{[0..i)})\rangle \in \mathsf{Stay}(s)$
(d) $\forall i \in [0..j].\ \delta(\langle S, s\rangle, w_{[0..i)}) \neq \langle B, b\rangle \vee w_{[i..]} \not\models \beta$

If $j = 0$, then w immediately satisfies the left-hand side. Thus assume $j > 0$. Due to (c), we have that $\langle \sigma, T' \rangle = \langle w_{[j-1]}, \delta(S, w_{[0..j-1)}) \rangle \in \mathsf{Stay}(s)$ and $\delta(\langle \delta(S, w_{[0..j-1)}), s \rangle, w_{[j-1]}) = \langle T, t \rangle$. Thus there exists a matching disjunct for $\langle \sigma, T' \rangle$ in line (1) and we know it is satisfied by w due the (a, b, d) and the induction hypothesis (I.H.2).

Reachability formula 5
We want to prove the following equivalence:

$$w \models \langle S, s \rangle \xrightarrow[\langle B,b \rangle (\beta)]{} \langle T, t \rangle (\tau)$$

$$\begin{aligned} \iff \exists i_1, i_2 \geq 0.\ & \delta(\langle S, s \rangle, w_{[0..i_1)}) = \langle T, t \rangle \wedge w_{[i_1..]} \models \tau \\ & \wedge\ (\exists j_1 \in [0..i_1).\ \langle w[j_1], \delta(S, w_{[0..j_1))} \in \mathsf{Enter}(s) \rangle) \\ & \wedge\ \langle w[i_2], \delta(S, w_{[0..i_2)}) \rangle \in \mathsf{Leave}(s) \\ & \wedge\ (\forall j_2 \in [0.. \max(i_1 - 1, i_2)]. \\ & \qquad \delta(\langle S, s \rangle, w_{[0..j_2)}) \neq \langle B, b \rangle \vee w_{[j_2..]} \not\models \beta) \end{aligned}$$

($\Rightarrow$): We assume that

$$w \models \langle S, s \rangle \xrightarrow[\langle B,b \rangle (\beta)]{} \langle T, t \rangle (\tau)$$

holds and proceed by first constructing a witness for i_2 and then one for i_1.

($\exists i_2$): Since w satisfies line (3), there must be a combined letter $\langle \sigma, L \rangle \in \mathsf{Leave}(s)$ such that w satisfies the corresponding disjunct. We apply to this disjunct the induction hypothesis (I.H.3) and instantiate the existential quantifier to i_2 (which we intend to be the witness for $\exists i_2$) with the following properties:

1. $\delta(\langle S, s \rangle, w_{[0..i_2)}) = \langle L, s \rangle$
2. $w[i_2] = \sigma$ and $\delta(\langle S, s \rangle, w_{[0..i_2)}) \neq \langle B, b \rangle \vee w_{[i_2..]} \not\models \beta$
3. $\forall j_2 \in [0..i_2).\ \delta(\langle S, s \rangle, w_{[0..j_2)}) \neq \langle B, b \rangle \vee w_{[j_2..]} \not\models \beta$
4. $\forall j_2 \in [0..i_2).\ \langle w[j_2], \delta(S, w_{[0..j_2)}) \rangle \in \mathsf{Stay}(s)$.

Observe that due to (1,2) we have $\langle w[i_2], \delta(S, w_{[0..i_2)}) \rangle \in \mathsf{Leave}(s)$ and together with (2,3) one can see that i_2 is a sufficient witness for the $\exists i_2$ in the right-hand side of the equation we want to prove. Note that property (4) is not useless and will be of importance later.

($\exists i_1$): Since w also satisfies lines (1,2), there must be a combined letter $\langle \sigma, T' \rangle \in \mathsf{Enter}(t)$ such that w satisfies the corresponding disjunct. We then apply the induction hypothesis (I.H.1) to all terms of the matching conjunction in lines (1,2). Since all reachability formulas share the same target configuration and formula, we can instantiate all existential quantifiers to the same integer k by picking the smallest instance for each existential quantifier. Let now k be this smallest non-negative integer. We introduce the following two abbreviations $T'' = \delta(\langle T', \cdot \rangle, \sigma)$ and $R'_\eta = \delta(\langle R, \cdot \rangle, \eta)$ for some $\langle \eta, R \rangle \in \mathsf{Enter}(b)$. Note that we can leave out the last state in the definition of T'', since for all states q, p the transition relation maps the the same successor configuration, i.e., $\delta(\langle T', q \rangle, \sigma) = \delta(\langle T', p \rangle, \sigma)$. Analogously, for we omit it from R'_η. We now list all relevant properties of k derived from applying the induction hypothesis:

5. $\delta(S, w_{[0..k)}) = T'$
6. $w[k] = \sigma$
7. $w_{[k+1..]} \models T'' \xrightarrow[\cancel{\langle B,b\rangle(\beta)}]{} \langle T, t\rangle\,(\tau)$
8. For all $\langle \eta, R\rangle \in \mathsf{Enter}(b)$ and all $\ell \in [0..k)$ at least one of the following statements is true:
   (a) $\delta(S, w_{[0..\ell)}) \neq R$
   (b) $w[\ell] \neq \eta$
   (c) $w_{[\ell+1..]} \not\models R'_\eta \xrightarrow[\cancel{\langle T,t\rangle(\tau)}]{\text{weak}} \langle B, b\rangle\,(\beta)$

We continue and apply the induction hypothesis to (7) and then obtain i_1 (which we intend to be the witness for $\exists i_1$) with the following properties (already shifted to indices relative to w). Further, we can assume i_1 to be the smallest integer with these properties.

9. $\delta(T'', w_{[k+1..i_1)}) = \langle T, t\rangle$ and $w_{[i_1..]} \models \tau$
10. $\forall j \in [(k+1)..i_1).\ \delta(T'', w_{[k+1..j)}) \neq \langle B, b\rangle \vee w_{[j..]} \not\models \beta$
11. $\forall j \in [(k+1)..i_1).\ \langle w[j], \delta(\delta(T', \sigma), w_{[k+1..j)})\rangle \in \mathsf{Stay}(t)$

From (5,6,9) we obtain that $\delta(\langle S, s\rangle, w_{[0..i_1)}) = \langle T, t\rangle$ and that $w_{[i_1..]} \models \tau$ taking care of the first line of the right-hand side of the equation. Further, by construction $\langle w[k], \delta(S, w_{[0..k)})\rangle = \langle \sigma, T\rangle \in \mathsf{Enter}(t)$ and since $k < i_1$, we now know that the second line is satisfied by our witness for $\exists i_1$. Next, we inline the definition of T'' in (10) and obtain: $\forall j \in [(k+1)..i_1).\ \delta(\langle S, s\rangle, w_{[0..j)}) \neq \langle B, b\rangle \vee w_{[j..]} \not\models \beta$. Thus in order to complete this direction of the proof it remains to show that:

$$\forall j \in [(i_2+1)..k].\ \delta(\langle S, s\rangle, w_{[0..j)}) \neq \langle B, b\rangle \vee w_{[j..]} \not\models \beta$$

If $[(i_2+1)..k]$ is the empty set, we are done. Thus assume that $i_2 < k$. We proceed with a proof by contradiction and assume that there exists an index $j \in [i_2+1..k]$ such that $\delta(\langle S, s\rangle, w_{[0..j)}) = \langle B, b\rangle$ and $w_{[j..]} \models \beta$. Due to (1) and (4), there must be some $j' \in [i_2..j-1]$ such that the combined letter $\langle \eta, R\rangle = \langle w[j'], \delta(S, w_{[0..j')})\rangle$ is in $\mathsf{Enter}(b)$. Without loss of generality we can further assume that:

12. $\forall j'' \in [j'..j).\ \langle w[j''], \delta(S, w_{[0..j'')})\rangle \in \mathsf{Stay}(b)$

Note that $j' < k$ and since (8a) and (8b) cannot be true we can instantiate (8c) to:

$$w_{[j'+1..]} \not\models R'_\eta \xrightarrow[\cancel{\langle T,t\rangle(\tau)}]{\text{weak}} \langle B, b\rangle\,(\beta)$$

We now apply the induction hypothesis to this and obtain an index $\ell \geq j' + 1$ such that the following two statements hold (already with adjusted indices):

13. At least one of the following statements hold:
    (a) $\delta(\langle S, s\rangle, w_{[0..\ell)}) = \langle T, t\rangle \wedge w_{[\ell..]} \models \tau$
    (b) $\ell > j' + 1 \wedge \langle w[\ell-1], \delta(\langle S, s\rangle, w_{[0..\ell-1)})\rangle \in \mathsf{Leave}(b)$
14. $\forall j'' \in [(j'+1)..\ell).\ \delta(\langle S, s\rangle, w_{[0..j'')}) \neq \langle B, b\rangle \vee w_{[j''..]} \not\models \beta$

Since $j' < j$ and since we assumed that j is the smallest integer satisfying $\delta(\langle S, s\rangle, w_{[0..j)}) = \langle B, b\rangle$ and $w_{[j''..]} \models \beta$, we can conclude with (14) that $\ell \leq j$. However, we have a contradiction between (13a), $\ell \leq j \leq k < i_1$, and i_1 being the smallest index reaching $\langle T, t\rangle$ and satisfying τ. Thus (13b) must hold. However, since $\ell \leq j$ this contradicts (12). Thus there cannot be such a j and we can conclude this direction.

($\Leftarrow$): Assume that w satisfies the left-hand side. Then there exists $i_1, i_2, j_1 \geq 0$, further let i_1, i_2 be the smallest integers and let j_1 be the largest integer such that:

(a) $\delta(\langle S, s\rangle, w_{[0..i_1)}) = \langle T, t\rangle$
(b) $w_{[i_1..]} \models \tau$
(c) $j_1 < i_1$
(d) $\langle w[j_1], \delta(S, w_{[0..j_1)})\rangle \in \mathsf{Enter}(t)$
(e) $\langle w[i_2], \delta(S, w_{[0..i_2)})\rangle \in \mathsf{Leave}(s)$
(f) $\forall j_2 \in [0..\max(i_1 - 1, i_2)].\ \delta(\langle S, s\rangle, w_{[0..j_2)}) \neq \langle B, b\rangle \vee w_{[j_2..]} \not\models \beta$

Since i_2 is the smallest integer with this property, we have $\langle w[k], \delta(S, w_{[0..k)})\rangle \in \mathsf{Stay}(s)$ for all $k \in [0..i_2{-}1]$. Thus we can apply (I.H.3) to this and (e,f) to show that w satisfies line (3) for $\langle w[i_2], \delta(S, w_{[0..i_2)})\rangle = \langle \sigma, L\rangle \in \mathsf{Leave}(s)$. In order to show the disjunct of (1) and (2), we need to identify a suitable $\langle \sigma, T'\rangle \in \mathsf{Enter}(t)$. We argue that $\langle \sigma, T'\rangle = \langle w[j_1], \delta(S, w_{[0..j_1)})\rangle$ is a suitable choice.

We proceed by first showing that w satisfies line (1). Since j_1 is the largest integer with such a property, we immediately get $\langle w[k], \delta(S, w_{[0..k)})\rangle \in \mathsf{Stay}(s)$ for all $k \in [j_1{+}1, i_1{-}1]$. By applying (I.H.3) we get:

$$w_{[j_1+1..]} \models \delta(\langle T', \cdot\rangle, \sigma) \xrightarrow[\cancel{\langle B,b\rangle(\beta)}]{} \langle T, t\rangle\,(\tau)$$

We now apply (I.H.1) and use (d) to establish that the first line is satisfied by w. In order show that line (2) is satisfied it remains to show that for all $k \in [0..j_1]$ and all $\langle \eta, L\rangle \in \mathsf{Enter}(b)$ either $\delta(\langle S, s\rangle, w_{[0..k)}) \neq \langle \eta, L\rangle$ or $w_{[k..]} \not\models \eta \wedge \mathbf{X}\Big(\delta(\langle R, \cdot\rangle, \eta) \xrightarrow[\cancel{\langle T,t\rangle(\tau)}]{\text{weak}} \langle B, b\rangle\,(\beta)\Big)$.

We proceed by contradiction. Assume that there is $k \leq j_1$ such that $\delta(\langle S, s\rangle, w_{[0..k)}) = \langle \eta, L\rangle$, $w[k] = \eta$, and $w_{[k+1..]} \models \delta(\langle R, \cdot\rangle, \eta) \xrightarrow[\cancel{\langle T,t\rangle(\tau)}]{\text{weak}} \langle B, b\rangle\,(\beta))$. By applying (I.H.4) and adjusting the indices we obtain:

$$\begin{aligned}\forall i \geq k+1.\ \big(&(\delta(\langle S, s\rangle, w_{[0..i)}) = \langle T, t\rangle \wedge w_{[i..]} \models \tau)\\ &\vee (i > k+1 \wedge \langle w[i{-}1], \delta(S, w_{[0..i-1)})\rangle \in \mathsf{Leave}(b))\big)\\ &\rightarrow\ (\exists j \in [k+1..i).\ \delta(\langle S, s\rangle, w_{[0..j)}) = \langle B, b\rangle \wedge w_{[j..]} \models \beta)\end{aligned}$$

By instantiating this with (a,b) we get:

$$\exists j \in [k+1..i_1).\ \delta(\langle S, s\rangle, w_{[0..j)}) = \langle B, b\rangle \wedge w_{[j..]} \models \beta$$

However this contradicts (f).
Thus we can apply (I.H.1) to establish that w satisfies (2). □

Lemma 5. *Let S, B, T be configurations of level $m \leq n$, and let s, b, t be states in level $m+1$ (when $m < n$). Then for $i \geq 1$ it holds that:*

- $S \overset{}{\rightsquigarrow_{B(\Pi_i)}} T(\Sigma_i)$, $\langle S, s\rangle \xrightarrow[\langle B,b\rangle(\Pi_i)]{} \langle T, t\rangle(\Sigma_i)$, $\langle S, s\rangle \overset{}{\dashrightarrow_{\langle B,b\rangle(\Pi_i)}} \langle T, t\rangle(\Sigma_i) \in \Sigma_i$
- $S \overset{weak}{\rightsquigarrow_{B(\Sigma_i)}} T(\Pi_i)$, $\langle S, s\rangle \xrightarrow[\langle B,b\rangle(\Sigma_i)]{weak} \langle T, t\rangle(\Pi_i) \in \Pi_i$

Proof. Observe first that there is no circularity in the definitions of the five reachability formulas, even though they are defined by each other: Formula 2 is defined on top of formula 1, which is defined on top of formulas 3 and 5, while formulas 3, 4, and 5 are defined with respect to reachability formulas over configurations of a lower level.

Let $i \geq 1$ be an arbitrary index. We prove the statement by induction on the level m of the configurations S, B, T in the reachability formulas. We split the proof to five cases corresponding to the five reachability formulas.

Reachability formula 1. Let $\beta \in \Pi_i$ and let $\tau \in \Sigma_i$. We proceed by an induction on the the configuration level m.

($m = 0$): There is only one configuration of level 0 and it is the empty configuration. Thus $S = T = B = \langle\rangle$. Applying the definition we obtain:

$$\langle\rangle \overset{}{\rightsquigarrow_{\langle\rangle(\beta)}} \langle\rangle(\tau) = (\neg\beta)\mathbf{U}\tau$$

Note that $\neg\beta \in \Sigma_i$ and thus the whole formula $(\neg\beta)\mathbf{U}\tau$ is also in Σ_i.

($m \to m+1$): Let $\langle S, s\rangle, \langle T, t\rangle, \langle B, b\rangle \in Q_1 \times \cdots \times Q_{m+1}$ be arbitrary configurations of level $m+1$. Thus:

$$\langle S, s\rangle \overset{}{\rightsquigarrow_{\langle B,b\rangle(\beta)}} \langle T, t\rangle(\tau) = \langle S, s\rangle \xrightarrow[\langle B,b\rangle(\beta)]{} \langle T, t\rangle(\tau) \vee \langle S, s\rangle \overset{}{\dashrightarrow_{\langle B,b\rangle(\beta)}} \langle T, t\rangle(\tau)$$

We now can use apply the induction hypothesis and obtain that the formula is in Σ_i.

Reachability formula 2. Let $\beta \in \Sigma_i$ and let $\tau \in \Pi_i$. Then by induction hypothesis $S \overset{}{\rightsquigarrow_{T(\tau)}} B(\beta)$ is in Σ_i and thus $S \overset{weak}{\rightsquigarrow_{B(\beta)}} T(\tau)$ is in Π_i.

Reachability formula 3. Let $\beta \in \Pi_i$ and let $\tau \in \Sigma_i$. Note that under this assumption **false**, η, and $\rho \wedge \mathbf{X}\beta$ belong to Π_i and that $\sigma \wedge \mathbf{X}\tau$ belongs to Σ_i. Thus by applying the induction hypothesis we obtain that $\langle S, s\rangle \xrightarrow[\langle B,b\rangle(\beta)]{>0} \langle T, t\rangle(\tau)$ is in Σ_i. Since $\neg\beta$ also belongs to Σ_i, we obtain that $\langle S, s\rangle \xrightarrow[\langle B,b\rangle(\beta)]{} \langle T, t\rangle(\tau)$ is from Σ_i.

Reachability formula 4. Let $\beta \in \Sigma_i$ and let $\tau \in \Pi_i$. Note that under this assumption η and $\rho \wedge \mathbf{X}\beta$ belong to Σ_i and that **false**, $\neg\beta$, and $\sigma \wedge \mathbf{X}\tau$ belong to Π_i. We then apply again the induction hypothesis and obtain that $\langle S, s\rangle \xrightarrow[\langle B,b\rangle(\beta)]{weak,>0} \langle T, t\rangle(\tau)$ and thus also $\langle S, s\rangle \xrightarrow[\langle B,b\rangle(\beta)]{weak} \langle T, t\rangle(\tau)$ belongs to Π_i.

Reachability formula 5. Let $\beta \in \Pi_i$ and let $\tau \in \Sigma_i$. By an analogous argument to the two preceding cases we obtain that the formulas in line (1) and (3) belong to Σ_i. Further, note that by the induction hypothesis $\eta \wedge \mathbf{X}\Big(\delta(\langle R, \cdot\rangle, \eta) \xrightarrow[\langle T,t\rangle(\tau)]{\text{weak}} \langle B, b\rangle(\beta)\Big)$ belongs to Π_i and thus the overall reachability formula 5 belongs to Σ_i. □

A.4 Proofs from Section 4.3

Lemma 6. *Consider a reset cascade $\mathcal{A}$ with n levels and up to n states in each level, and a formula $\zeta = S \xrightarrow[B(\beta)]{} T(\tau)$ with configurations S, B and T of $\mathcal{A}$ of level $i \leq n$. Let $d = \max(\mathsf{depth}(\beta), \mathsf{depth}(\tau))$ and let $l = \max(|\beta|, |\tau|)$. Then:*

$$(a)\ \mathsf{depth}(\zeta) \leq d + 3^i \quad \text{and} \quad (b)\ |\zeta| \leq l \cdot (10|\Sigma|^2 n)^{4^i}$$

Proof. We first prove the upper bound on the depth of the formula and then move on to the claim about the length of the formula.

Depth Analysis. We prove the claim by induction on the level i. For the base case of $i = 0$, the main reachability formula is just $(\neg\beta)\mathbf{U}\tau$, having $\mathsf{D}(0, d) = d + 1$, which is equal to $d + 3^0$, as required. For the induction step of a general level $i > 0$, we will first establish a bound on $\mathsf{D}(i, d)$ that is relative to $\mathsf{D}(i{-}1, d)$, and then get from it an absolute bound, using the induction hypothesis.

Observe that $\mathsf{D}(i, d)$ is bounded in Table 2 to the maximum between $\mathsf{D}_3(i, d)$ and $\mathsf{D}_5(i, d)$, while the bound on $\mathsf{D}_5(i, d)$ is at least as on $\mathsf{D}_3(i, d)$. Hence:

$$\begin{aligned} \mathsf{D}(i,d) \ \leq\ \mathsf{D}_5(i,d) &\leq \mathsf{D}(i{-}1, \max(1 + \mathsf{D}_3(i,d), 1 + \mathsf{D}_4(i,d))) \\ &\leq \mathsf{D}(i{-}1, 1 + \mathsf{D}(i{-}1, d+1)) \end{aligned}$$

Applying the induction hypothesis on $\mathsf{D}(i{-}1, d)$, we get an absolute bound:

$$\begin{aligned} \mathsf{D}(i,d) &\leq \mathsf{D}(i{-}1, 1 + (d + 1 + 3^{i-1})) \\ &\leq 2 + d + 3^{i-1} + 3^{i-1} \leq d + 3^i \end{aligned}$$

Length Analysis. We prove the claim by induction on the level i. For the base case of $i = 0$, the main reachability formula is just $(\neg\beta)\mathbf{U}\tau$, having $\mathsf{L}(0, l) = 2 + 2l$, which is indeed not larger than $l \cdot (10|\Sigma|^2 n)^{4^i}$. For the induction step of a general level $i > 0$, we will first establish a bound on $\mathsf{L}(i, l)$ relative to $\mathsf{L}(i{-}1, \cdot)$, and then apply the induction hypothesis to get an absolute bound.

Observe that $\mathsf{L}(i, l)$ is bounded in Table 2 with respect to $\mathsf{L}_3(i, \cdot)$ and $\mathsf{L}_5(i, \cdot)$, and $\mathsf{L}_3(i, l)$ is already bounded in Table 2 with respect to $\mathsf{L}(i{-}1, \cdot)$. We simplify the L_5 bound, and substitute the bound on L_3, getting:

$$
\begin{aligned}
\mathsf{L}_5(i,l) \leq\ & |\Sigma|n^{i-1} \cdot \big(\mathsf{L}_1(i-1, 3+\mathsf{L}_3(i,l)) + 2 + \\
& |\Sigma|n^{i-1} \cdot \big(\mathsf{L}_1(i-1, \max(3+\mathsf{L}_3(i,l), 3+\mathsf{L}_4(i,l))) + 1\big) + \\
& 1 + |\Sigma|n^{i-1} \cdot (1 + \mathsf{L}_3(i, 3+l)) \\
\leq\ & |\Sigma|n^{i-1} \cdot \big(\mathsf{L}_1(i-1, 3+\mathsf{L}_3(i,l)) + 2 + \\
& |\Sigma|n^{i-1} \cdot \big(\mathsf{L}_1(i-1, 3+\mathsf{L}_3(i,l)) + 1\big) + \\
& 1 + |\Sigma|n^{i-1} \cdot (1 + \mathsf{L}_3(i, 3+l)) \\
\leq\ & |\Sigma|^2 n^{2(i-1)} \mathsf{L}_1(i-1, 3+\mathsf{L}_3(i,l)) + \\
& |\Sigma|n^{(i-1)} \mathsf{L}_1(i-1, 3+\mathsf{L}_3(i,l)) + \\
& |\Sigma|n^{(i-1)} \mathsf{L}_3(i, 3+l)) + \\
& |\Sigma|^2 n^{2(i-1)} + \\
& 3|\Sigma|n^{(i-1)} + 1 \\
\leq\ & 4|\Sigma|^2 n^{2(i-1)} \mathsf{L}_1(i-1, 3+\mathsf{L}_3(i, l+3)) \\
\leq\ & 4|\Sigma|^2 n^{2(i-1)} \mathsf{L}_1(i-1, 12+2l+4|\Sigma|^2 n^{2(i-1)} \mathsf{L}_1(i-1, l+3))
\end{aligned}
$$

We can now bound $\mathsf{L}(i,l)$ relative to $\mathsf{L}(i-1,\cdot)$, getting:

$$
\begin{aligned}
\mathsf{L}(i,l) \leq\ & 1 + \mathsf{L}_3(i,l) + \mathsf{L}_5(i,l) \\
\leq\ & 4 + 2l + 4|\Sigma|n^{2(i-1)} \mathsf{L}_1(i-1, l+3) + \\
& 4|\Sigma|^2 n^{2(i-1)} \mathsf{L}_1(i-1, 12+2l+4|\Sigma|^2 n^{2(i-1)} \mathsf{L}_1(i-1, l+3)) \\
\leq\ & 5|\Sigma|^2 n^{2(i-1)} \mathsf{L}_1(i-1, 12+2l+4|\Sigma|^2 n^{2(i-1)} \mathsf{L}_1(i-1, l+3)) \\
\leq\ & 5|\Sigma|^2 n^{2(i-1)} \mathsf{L}_1(i-1, 5|\Sigma|^2 n^{2(i-1)} \mathsf{L}_1(i-1, l+3))
\end{aligned}
$$

Applying the induction hypothesis on $\mathsf{L}(i-1,\cdot)$, we get an absolute bound:

$$
\begin{aligned}
\mathsf{L}(i,l) \leq\ & 5|\Sigma|^2 n^{2(i-1)} \mathsf{L}_1(i-1, 5|\Sigma|^2 n^{2(i-1)} \mathsf{L}_1(i-1, l+3)) \\
\leq\ & 5|\Sigma|^2 n^{2(i-1)} (5|\Sigma|^2 n^{2(i-1)} (l+3)(2|\Sigma|n)^{4^{i-1}})(10|\Sigma|^2 n)^{4^{i-1}} \\
\leq\ & 25|\Sigma|^4 n^{4(i-1)} (l+3)(10|\Sigma|^2 n)^{2 \cdot 4^{i-1}} \\
\leq\ & l(10|\Sigma|^2 n)^{4^i}
\end{aligned}
$$

The last inequality clearly holds for $i \geq 2$ as the $(10|\Sigma|^2 n)^{4^i}$ term becomes very large. For $i = 1$, it also holds as the $n^{4(i-1)}$ term disappears and we get $25 \cdot 10^2 (l+3) \leq 10^4 l n^2$, which holds even for $l = n = 1$. □

A.5 Proofs from Section 4.4

Lemma 7. *Consider a reset cascade $\mathcal{A} = \langle 2^{AP}, \mathcal{A}_1, \ldots, \mathcal{A}_n \rangle$ together with an initial configuration ι and some configuration C. Then for a word $w \in (2^{AP})^\omega$,*

the run of $\mathcal{A}$ on w starting in ι visits C finitely often iff w satisfies the formula $Fin(C) := \neg(\iota \rightsquigarrow C) \vee \iota \rightsquigarrow C(\neg(C \overset{>0}{\rightsquigarrow} C))$. *Furthermore,* $Fin(C) \in \Sigma_2$.

Proof. Observe that C is visited finitely often iff either C is not visited at all, which by Lemma 4 is formulated by $\neg(\iota \rightsquigarrow C)$ or there exists a last visit to C, namely a visit to C after which there is no other visit to C, which is formulated by $\iota \rightsquigarrow C(\neg(C \overset{>0}{\rightsquigarrow} C))$. Note that by Lemma 5 we have $\iota \rightsquigarrow C, C \overset{>0}{\rightsquigarrow} C \in \Sigma_1$. Thus $\neg\iota \rightsquigarrow C, \neg C \overset{>0}{\rightsquigarrow} C \in \Pi_1$ and then again by Lemma 5 we get $\iota \rightsquigarrow C(\neg(C \overset{>0}{\rightsquigarrow} C)) \in \Sigma_2$. Thus $Fin(C) \in \Sigma_2$. □

Theorem 2. *Every counter-free deterministic ω-regular automaton $\mathcal{D}$ over alphabet 2^{AP} with n states (and any acceptance condition) is equivalent to an LTL formula φ over atomic propositions AP of double-exponential temporal-nesting depth (in $O(2^{2^n})$) and triple-exponential length (in $2^{2^{O(2^n)}}$). If $\mathcal{D}$ is a looping-Büchi, looping-coBüchi, weak, Büchi, coBüchi, or Muller automaton then φ is respectively in the $\Pi_1, \Sigma_1, \Delta_1, \Pi_2, \Sigma_2$, or Δ_2 syntactic fragment of LTL.*

Proof. We complete the proof of the Theorem and give a complete analysis of all six acceptance conditions:

- $\mathcal{D}$ is a Muller automaton: the overall formula φ is in Δ_2, since it is a Boolean combination of $Fin(C)$ formulas, which by Lemma 7 belong to Σ_2.
- $\mathcal{D}$ is a coBüchi automaton: we construct the formula φ directly from the coBüchi condition α, having a conjunction of $Fin(C)$ formulas, over all configurations C that are mapped to states in α. As $Fin(C)$ belongs to Σ_2, so does φ.
- $\mathcal{D}$ is a Büchi automaton: we can complement it, apply the above argument over the resulting coBüchi automaton, and negate the resulting formula to obtain a formula from Π_2.
- $\mathcal{D}$ is a looping-coBüchi automaton: Let $s \in Q$ be the unique sink state that all accepting runs end up in, and let S be the set of configurations mapped to s. We then define φ as $\bigvee_{C \in S} \iota \rightsquigarrow C$. Note that φ belongs to Σ_1 since every disjunct belongs to Σ_1 due to Lemma 5.
- $\mathcal{D}$ is a looping-Büchi automaton: the dual of the previous argument.
- $\mathcal{D}$ is a weak automaton: Let $G \subseteq Q$ be an accepting SCC of $\mathcal{D}$ and $G' \subseteq Q$ be all states that are reachable from G, but are not in G. Let H and H' be the set of configurations that are mapped to G and G', respectively. Then by Lemma 4, we have that $(\bigvee_{C \in H} \iota \rightsquigarrow C) \wedge (\bigwedge_{C' \in H'} \neg\iota \rightsquigarrow C')$ exactly captures all words that are accepted by eventually ending up in G. The overall formula φ is then the disjunction of these formulas constructed for each accepting SCC of $\mathcal{D}$. The membership in Δ_1 then follows immediately from Lemma 5. □