One Theorem to Rule Them All: A Unified Translation of LTL into ω-Automata*

Javier Esparza
esparza@in.tum.de
Technische Universität München
Germany

Jan Křetínský
jan.kretinsky@in.tum.de
Technische Universität München
Germany

Salomon Sickert
sickert@in.tum.de
Technische Universität München
Germany

Abstract

We present a unified translation of LTL formulas into deterministic Rabin automata, limit-deterministic Büchi automata, and nondeterministic Büchi automata. The translations yield automata of asymptotically optimal size (double or single exponential, respectively). All three translations are derived from one single Master Theorem of purely logical nature. The Master Theorem decomposes the language of a formula into a positive boolean combination of languages that can be translated into ω-automata by elementary means. In particular, Safra's, ranking, and breakpoint constructions used in other translations are not needed.

CCS Concepts • **Theory of computation → Automata over infinite objects**; **Modal and temporal logics**;

Keywords Linear temporal logic, Automata over infinite words, Deterministic automata, Non-deterministic automata

1 Introduction

Linear temporal logic (LTL) [32] is a prominent specification language, used both for model checking and automatic synthesis of systems. In the standard automata-theoretic approach [38] the input formula is first translated into an ω-automaton, and then the product of this automaton with the input system is further analyzed. Since the size of the product is often the bottleneck of all the verification algorithms, it is crucial that the ω-automaton is as small as possible. Consequently, a lot of effort has been spent on translating LTL into small automata, e.g. [4, 10–12, 17, 18, 20, 21, 36].

While non-deterministic Büchi automata (NBA) can be used for model checking non-deterministic systems, other applications such as model checking probabilistic systems or synthesis usually require automata with a certain degree of determinism, such as deterministic parity automata (DPA) or deterministic Rabin automata (DRA) [5], deterministic generalized Rabin automata (DGRA) [8], limit-deterministic (or semi-deterministic) Büchi automata (LDBA) [9, 22, 35, 37], unambiguous Büchi automata [6] etc. The usual constructions that produce such automata are based on Safra's determinization and its variants [31, 33, 34]. However, they are known to be difficult to implement efficiently, and to be practically inefficient in many cases due to their generality. Therefore, a recent line of work shows how DPA [14, 28], DRA and DGRA [13, 15, 26, 27], or LDBA [23, 24, 35] can be produced directly from LTL, without the intermediate step through a non-deterministic automaton. All these works share the principle of describing each state by a collection of formulas, as happens in the classical tableaux construction for translation of LTL into NBA. This makes the approach particularly apt for semantic-based state reductions, e.g., for merging states corresponding to equivalent formulas. These reductions cannot be applied to Safra-based constructions, where this semantic structure gets lost.

In this paper, we provide a unified view of translations of LTL into NBA, LDBA, and DRA enjoying the following properties, absent in former translations:

Asymptotic Optimality. D(G)RA are the most compact among the deterministic automata used in practice, in particular compared to DPA. Previous translations to D(G)RA were either limited to fragments of LTL [3, 26, 27], or only shown to be triply exponential [13, 15]. Here we provide constructions for all mentioned types of automata matching the optimal double exponential bound for DRA and LDBA, and the optimal single exponential bound for NBA.

Symmetry. The first translations [26, 27] used auxiliary automata to monitor each *Future*- and *Globally*-subformula. While this approach worked for fragments of LTL, subsequent constructions for full LTL [13, 15, 35] could not preserve the symmetric treatment. They only used auxiliary automata for **G**-subformulas, at the price of more complex constructions. Our translation re-establishes the symmetry of the first constructions. It treats **F** and **G** equally (actually, and more generally, it treats each operator and its dual equally), which results into simpler auxiliary automata.

Independence of Syntax. Previous translations were quite sensitive to the operators used in the syntax of LTL. In particular, the only greatest-fixed-point operator they allowed was *Globally*. Since formulas also had to be in negation normal form, pre-processing of the input often led to unnecessarily large formulas. While our translations still requires negation normal form, it allows for direct treatment of *Release*, *Weak until*, and other operators.

Unified View. Our translations rely on a novel *Master Theorem*, which decomposes the language of a formula into a positive boolean combination of "simple" languages, in the sense that they are easy to translate into automata. This approach is arguably simpler than previous ones (it is certainly simpler than our previous papers [15, 35]). Besides, it provides a unified treatment of DRA, NBA, and LDBA, differing only in the translations of the "simple" languages. The automaton for the formula is obtained from the automata for the "simple" languages by means of standard operations for closure under union and intersection.

*This work was partially funded and supported by the Czech Science Foundation, grant No. P202/12/G061, and the German Research Foundation (DFG) project "Verified Model Checkers" (317422601).

LICS '18, July 9–12, 2018, Oxford, United Kingdom

ACM ISBN 978-1-4503-5583-4/18/07…$15.00
https://doi.org/10.1145/3209108.3209161

On top of its theoretical advantages, our translation is comparable to previous DRA translations in practice, even without major optimizations. Summarizing, we think this paper finally achieves the goals formulated in [26], where the first translation of this kind—valid only for what we would now call a small fragment of LTL—was presented.

Structure of the Paper. Section 2 contains preliminaries about LTL and ω-automata. Section 3 introduces some definitions and results of [15, 35]. Section 4 shows how to use these notions to translate four simple fragments of LTL into deterministic Büchi and coBüchi automata; these translations are later used as building blocks. Section 5 presents our main result, the Master Theorem. Sections 6, 7, and 8 apply the Master Theorem to derive translations of LTL into DRA, NBA, and LDBA, respectively. Section 9 compares the paper to related work and puts the obtained results into context. The appendix of the accompanying technical report [16] contains the few omitted proofs and further related material.

2 Preliminaries

2.1 ω-Languages and ω-Automata

Let Σ be a finite alphabet. An ω-word w over Σ is an infinite sequence of letters $w[0]w[1]w[2]\ldots$. We denote the finite infix $w[i]w[i+1]\cdots w[j-1]$ by w_{ij}, and the infinite suffix $w[i]w[i+1]\ldots$ by w_i. An ω-language is a set of ω-words.

For the sake of presentation, we introduce ω-automata with accepting conditions defined on states. However, all results can be restated with accepting conditions defined on transitions, more in line with other recent papers and tools [2, 12, 25].

Let Σ be a finite alphabet. A *nondeterministic pre-automaton* over Σ is a tuple $\mathcal{P} = (Q, \Delta, Q_0)$ where Q is a finite set of states, $\Delta\colon Q \times \Sigma \to 2^Q$ is a transition function, and Q_0 is a set of initial states. A transition is a triple (q, a, q') such that $q' \in \Delta(q, a)$. A pre-automaton $\mathcal{P}$ is deterministic if Q_0 is a singleton and $\Delta(q, a)$ is a singleton for every $q \in Q$ and $a \in \Sigma$.

A *run* of $\mathcal{P}$ on an ω-word w is an infinite sequence of states $r = q_0 q_1 q_2 \ldots$ with $q_{i+1} \in \delta(q_i, w[i])$ for all i and we denote by $\mathit{inf}(r)$ the set of states occurring infinitely often in r. An *accepting condition* is an expression over the syntax $\alpha ::= \mathit{inf}(S) \mid \mathit{fin}(S) \mid \alpha_1 \vee \alpha_2 \mid \alpha_1 \wedge \alpha_2$ with $S \subseteq Q$. Accepting conditions are evaluated on runs and the evaluation relation $r \models \alpha$ is defined as follows:

$$\begin{array}{lllll} r & \models & \mathit{inf}(S) & \text{iff} & \mathit{inf}(r) \cap S \neq \emptyset \\ r & \models & \mathit{fin}(S) & \text{iff} & \mathit{inf}(r) \cap S = \emptyset \\ r & \models & \alpha_1 \vee \alpha_2 & \text{iff} & r \models \alpha_1 \text{ or } r \models \alpha_2 \\ r & \models & \alpha_1 \wedge \alpha_2 & \text{iff} & r \models \alpha_1 \text{ and } r \models \alpha_2 \end{array}$$

An accepting condition α is a

- Büchi condition if $\alpha = \mathit{inf}(S)$ for some set S of states.
- coBüchi condition if $\alpha = \mathit{fin}(S)$ for some set S of states.
- Rabin condition if $\alpha = \bigvee_{i=1}^{k} (\mathit{inf}(I_i) \wedge \mathit{fin}(F_i))$ for some $k \geq 1$ and some sets $I_1, F_1, \ldots, I_k, F_k$ of states.

An ω-automaton over Σ is a tuple $\mathcal{A} = (Q, \Delta, Q_0, \alpha)$ where (Q, Δ, Q_0) is a pre-automaton over Σ and α is an accepting condition. A run r of $\mathcal{A}$ is *accepting* if $r \models \alpha$. A word w is accepted by $\mathcal{A}$ if some run of $\mathcal{A}$ on w is accepting. An ω-automaton is a Büchi (coBüchi, Rabin) automaton if its accepting condition is a Büchi (coBüchi, Rabin) condition.

Limit-Deterministic Büchi Automata. Intuitively, a NBA is limit-deterministic if it can be split into a non-deterministic component without accepting states, and a deterministic component. The automaton can only accept by "jumping" from the non-deterministic to the deterministic component, but after the jump it must stay in the deterministic component forever. Formally, a NBA $\mathcal{B} = (Q, \Delta, Q_0, \alpha)$ is *limit-deterministic* (LDBA) if Q can be partitioned into two disjoint sets $Q = Q_{\mathcal{N}} \uplus Q_{\mathcal{D}}$, s.t.

1. $\Delta(q, \nu) \subseteq Q_{\mathcal{D}}$ and $|\Delta(q, \nu)| = 1$ for every $q \in Q_{\mathcal{D}}$, $\nu \in \Sigma$, and
2. $S \subseteq Q_{\mathcal{D}}$ for all $S \in \alpha$.

2.2 Linear Temporal Logic

We work with a syntax for LTL in which formulas are written in negation-normal form, i.e., negations only occur in front of atomic propositions. For every temporal operator we also include in the syntax its dual operator. On top of the next operator **X**, which is self-dual, we introduce temporal operators **F** (eventually), **U** (until), and **W** (weak until), and their duals **G** (always), **R** (release) and **M** (strong release). The syntax may look redundant but as we shall see it is essential to include **W** and **M** and very convenient to include **F** and **G**.

Syntax and semantics of LTL. A formula of LTL in *negation normal form* over a set of atomic propositions (Ap) is given by the syntax:

$$\varphi ::= \mathbf{tt} \mid \mathbf{ff} \mid a \mid \neg a \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \mathbf{X}\varphi \mid \mathbf{F}\varphi \mid \mathbf{G}\varphi \mid \varphi \mathbf{U} \varphi \mid \varphi \mathbf{W} \varphi \mid \varphi \mathbf{M} \varphi \mid \varphi \mathbf{R} \varphi$$

where $a \in Ap$. We denote $sf(\varphi)$ the set of subformulas of φ. A subformula ψ of φ is called *proper* if it is neither a conjunction nor a disjunction, i,e., if the root of its syntax tree is labelled by either a, $\neg a$, or a temporal operator. The satisfaction relation $\models$ between ω-words over the alphabet 2^{Ap} and formulas is inductively defined as follows:

$$\begin{array}{lll} w \models \mathbf{tt} & & \\ w \not\models \mathbf{ff} & & \\ w \models a & \text{iff} & a \in w[0] \\ w \models \neg a & \text{iff} & a \notin w[0] \\ w \models \varphi \wedge \psi & \text{iff} & w \models \varphi \text{ and } w \models \psi \\ w \models \varphi \vee \psi & \text{iff} & w \models \varphi \text{ or } w \models \psi \\ w \models \mathbf{X}\varphi & \text{iff} & w_1 \models \varphi \\ w \models \mathbf{F}\varphi & \text{iff} & \exists k.\, w_k \models \varphi \\ w \models \mathbf{G}\varphi & \text{iff} & \forall k.\, w_k \models \varphi \\ w \models \varphi \mathbf{U} \psi & \text{iff} & \exists k.\, w_k \models \psi \text{ and } \forall j < k.\, w_j \models \varphi \\ w \models \varphi \mathbf{W} \psi & \text{iff} & w \models \mathbf{G}\varphi \text{ or } w \models \varphi \mathbf{U} \psi \\ w \models \varphi \mathbf{M} \psi & \text{iff} & \exists k.\, w_k \models \varphi \text{ and } \forall j \leq k.\, w_j \models \psi \\ w \models \varphi \mathbf{R} \psi & \text{iff} & w \models \mathbf{G}\psi \text{ or } w \models \varphi \mathbf{M} \psi \end{array}$$

Two formulas are equivalent if they are satisfied by the same words. We also introduce the stronger notion of propositional equivalence:

Definition 2.1 (Propositional Equivalence)**.** Given a formula φ, we assign to it a propositional formula φ_P as follows: replace every maximal proper subformula ψ by a propositional variable x_ψ. Two formulas φ, ψ are *propositionally equivalent*, denoted $\varphi \equiv_P \psi$, iff φ_P and ψ_P are equivalent formulas of propositional logic. The set of all formulas propositionally equivalent to φ is denoted by $[\varphi]_P$.

Example 2.2. Let $\varphi = \mathbf{X}b \vee (\mathbf{G}(a \vee \mathbf{X}b) \wedge \mathbf{X}b)$ with $\psi_1 = \mathbf{X}b$ and $\psi_2 = \mathbf{G}(a \vee \mathbf{X}b)$. We have $\varphi_P = x_{\psi_1} \vee (x_{\psi_2} \wedge x_{\psi_1}) \equiv_P x_{\psi_1}$. Thus $\mathbf{X}b$ is propositionally equivalent to φ and $\mathbf{X}b \in [\varphi]_P$. △

Observe that propositional equivalence implies equivalence, but the converse does not hold.

3 The "after" Function

We recall the definition of the"after function" $af(\varphi, w)$, read "φ after w" [13, 15]. The function assigns to a formula φ and a finite word w another formula such that, intuitively, φ holds for ww' iff $af(\varphi, w)$ holds "after reading w", that is, iff $w' \models af(\varphi, w)$.[1]

Definition 3.1. Let φ be a formula and $\nu \in 2^{Ap}$ a single letter. The formula $af(\varphi, \nu)$ is inductively defined as follows:

$$af(a, \nu) = \begin{cases} \mathbf{tt} & \text{if } a \in \nu \\ \mathbf{ff} & \text{if } a \notin \nu \end{cases} \qquad af(\mathbf{tt}, \nu) = \mathbf{tt} \qquad af(\mathbf{ff}, \nu) = \mathbf{ff}$$

$$af(\neg a, \nu) = \begin{cases} \mathbf{ff} & \text{if } a \in \nu \\ \mathbf{tt} & \text{if } a \notin \nu \end{cases}$$

$$af(\varphi \wedge \psi, \nu) = af(\varphi, \nu) \wedge af(\psi, \nu) \qquad af(\varphi \vee \psi, \nu) = af(\varphi, \nu) \vee af(\psi, \nu)$$

$$\begin{aligned} af(\mathbf{X}\varphi, \nu) &= \varphi \\ af(\mathbf{F}\varphi, \nu) &= af(\varphi, \nu) \vee \mathbf{F}\varphi \\ af(\mathbf{G}\varphi, \nu) &= af(\varphi, \nu) \wedge \mathbf{G}\varphi \\ af(\varphi \mathbf{U} \psi, \nu) &= af(\psi, \nu) \vee (af(\varphi, \nu) \wedge \varphi \mathbf{U} \psi) \\ af(\varphi \mathbf{W} \psi, \nu) &= af(\psi, \nu) \vee (af(\varphi, \nu) \wedge \varphi \mathbf{W} \psi) \\ af(\varphi \mathbf{M} \psi, \nu) &= af(\psi, \nu) \wedge (af(\varphi, \nu) \vee \varphi \mathbf{M} \psi) \\ af(\varphi \mathbf{R} \psi, \nu) &= af(\psi, \nu) \wedge (af(\varphi, \nu) \vee \varphi \mathbf{R} \psi) \end{aligned}$$

Furthermore, we generalize the definition to finite words by setting $af(\varphi, \epsilon) = \varphi$ and $af(\varphi, \nu w) = af(af(\varphi, \nu), w)$ for every $\nu \in 2^{Ap}$ and every finite word w. Finally, we define the set of formulas *reachable from* φ as $Reach(\varphi) = \{[\psi]_P \mid \exists w.\ \psi = af(\varphi, w)\}$.

Example 3.2. Let $\varphi = a \vee (b \mathbf{U} c)$. We then have $af(\varphi, \{a\}) \equiv_P \mathbf{tt}$, $af(\varphi, \{b\}) \equiv_P (b \mathbf{U} c)$, $af(\varphi, \{c\}) \equiv_P \mathbf{tt}$, and $af(\varphi, \emptyset) \equiv_P \mathbf{ff}$. △

The following lemma states the main properties of *af*, which are easily proved by induction on the structure of φ. For convenience we include the short proof in the appendix of [16].

Lemma 3.3. *[15]*

1. *For every formula φ, finite word $w \in (2^{Ap})^*$, and infinite word $w' \in (2^{Ap})^\omega$: $ww' \models \varphi$ iff $w' \models af(\varphi, w)$*
2. *For every formula φ and finite word $w \in (2^{Ap})^*$: $af(\varphi, w)$ is a positive boolean combination of proper subformulas of φ.*
3. *For every formula φ: If φ has n proper subformulas, then $Reach(\varphi)$ has at most size 2^{2^n}.*

It is easy to show by induction that $\varphi \equiv_P \psi$ implies $af(\varphi, w) \equiv_P af(\psi, w)$ for every finite word w. We extend *af* to equivalence classes by defining $af([\varphi]_P, w) := [af(\varphi, w)]_P$. Sometimes we abuse language and identify a formula and its equivalence class. For example, we write "the states of the automaton are pairs of formulas" instead of "pairs of equivalence classes of formulas".

4 Constructing DRAs for Fragments of LTL

We show that the function *af* can be used to construct deterministic Büchi and coBüchi automata for some fragments of *LTL*. The constructions are very simple. Later, in Sections 6, 7, and 8 we use these constructions as building blocks for the translation of general LTL formulas. The fragments are:

- The μ-fragment μLTL and the ν-fragment νLTL.
  μLTL is the fragment of LTL restricted to temporal operators $\mathbf{F}, \mathbf{U}, \mathbf{M}$, on top of Boolean connectives ($\wedge, \vee$), literals ($a, \neg a$), and the next operator ($\mathbf{X}$). νLTL is defined analogously, but with the operators $\mathbf{G}, \mathbf{W}, \mathbf{R}$. In the literature μLTL is also called syntactic co-safety and νLTL syntactic safety.
- The fragments $\mathbf{GF}(\mu LTL)$ and $\mathbf{FG}(\nu LTL)$.
  These fragments contain the formulas of the form $\mathbf{GF}\varphi$, where $\varphi \in \mu LTL$, and $\mathbf{FG}\varphi$, where $\varphi \in \nu LTL$.

The reason for the names μLTL and νLTL is that $\mathbf{F}, \mathbf{U}, \mathbf{M}$ are least-fixed-point operators, in the sense that their semantics is naturally formulated by least fixed points, e.g. in the μ-calculus, while the semantics of $\mathbf{G}, \mathbf{W}, \mathbf{R}$ is naturally formulated by greatest fixed points.

The following lemma characterizes the words w satisfying a formula φ of these fragments in terms of the formulas $af(\varphi, w)$.

Lemma 4.1. *[15] Let $\varphi \in \mu LTL$ and let w be a word. We have:*

- $w \models \varphi$ *iff* $\exists i.\ af(\varphi, w_{0i}) \equiv_P \mathbf{tt}$.
- $w \models \mathbf{GF}\varphi$ *iff* $\forall i.\ \exists j.\ af(\mathbf{F}\varphi, w_{ij}) \equiv_P \mathbf{tt}$.

Let $\varphi \in \nu LTL$ and let w be a word. We have:

- $w \models \varphi$ *iff* $\forall i.\ af(\varphi, w_{0i}) \not\equiv_P \mathbf{ff}$.
- $w \models \mathbf{FG}\varphi$ *iff* $\exists i. \forall j.\ af(\mathbf{G}\varphi, w_{ij}) \not\equiv_P \mathbf{ff}$

The following proposition constructs DBAs or DCAs for the fragments. The proof is an immediate consequence of the lemma.

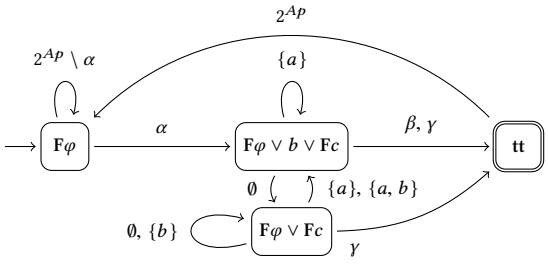
Proposition 4.2. *Let $\varphi \in \mu LTL$.*

- *The following DBA over the alphabet 2^{Ap} recognizes $L(\varphi)$:*
$$\mathcal{A}^\varphi_\mu = (Reach(\varphi), af, \varphi, inf(\mathbf{tt}))$$
- *The following DBA over the alphabet 2^{Ap} recognizes $L(\mathbf{GF}\varphi)$:*
$$\mathcal{A}^\varphi_{\mathbf{GF}\mu} = (Reach(\mathbf{F}\varphi), af_{\mathbf{F}\varphi}, \mathbf{F}\varphi, inf(\mathbf{tt}))$$
$$af_{\mathbf{F}\varphi}(\psi, \nu) = \begin{cases} \mathbf{F}\varphi & \text{if } \psi \equiv_P \mathbf{tt} \\ af(\psi, \nu) & \text{otherwise.} \end{cases}$$

Let $\varphi \in \nu LTL$.

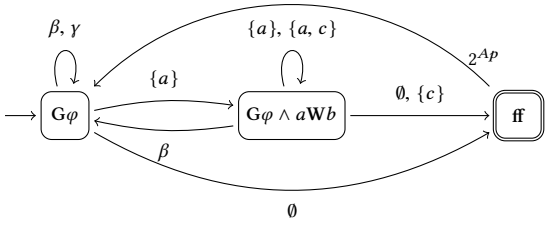
- *The following DCA over the alphabet 2^{Ap} recognizes $L(\varphi)$:*
$$\mathcal{A}^\varphi_\nu = (Reach(\varphi), af, \varphi, fin(\mathbf{ff}))$$
- *The following DCA over the alphabet 2^{Ap} recognizes $L(\mathbf{FG}\varphi)$:*
$$\mathcal{A}^\varphi_{\mathbf{FG}\nu} = (Reach(\mathbf{G}\varphi), af_{\mathbf{G}\varphi}, \mathbf{G}\varphi, fin(\mathbf{ff}))$$
$$af_{\mathbf{G}\varphi}(\psi, \nu) = \begin{cases} \mathbf{G}\varphi & \text{if } \psi \equiv_P \mathbf{ff} \\ af(\psi, \nu) & \text{otherwise.} \end{cases}$$

Example 4.3. Let $\varphi = a \wedge \mathbf{X}(b \vee \mathbf{F}c) \in \mu LTL$. The DBA $\mathcal{A}^\varphi_{\mathbf{GF}\mu}$ recognizing $L(\mathbf{GF}\varphi)$ is depicted below. We use the abbreviations $\alpha := \{\nu \in 2^{Ap} \mid a \in \nu\}$, $\beta := \{\nu \in 2^{Ap} \mid b \in \nu\}$, and $\gamma := \{\nu \in 2^{Ap} \mid c \in \nu\}$. △



[1] There is a conceptual correspondences to the derivatives of [7] and *af* directly connects to the classical "LTL expansion laws" [5]. Furthermore, the yet to be introduced $af^\vee$ relates to [1] in a similar way.

Example 4.4. Let $\varphi = a\mathbf{W}b \vee c \in \nu LTL$. The DCA $\mathcal{A}^{\varphi}_{\mathbf{FG}\nu}$ recognizing $L(\mathbf{FG}\varphi)$ is depicted below. We use the abbreviations of Example 4.3 again.



Now consider the formula $\varphi = \mathbf{FG}(a\mathbf{U}b \vee c)$. It does not belong to any of the fragments due to the deeper alternation of the least- and greatest-fixed-point operators: $\mathbf{F}-\mathbf{G}-\mathbf{U}$. If we construct $\mathcal{A}^{\varphi}_{\mathbf{FG}\nu}$ we obtain a DCA isomorphic to the one above, because $af(\psi_1\mathbf{U}\psi_2, \nu)$ and $af(\psi_1\mathbf{W}\psi_2, \nu)$ are defined in the same way. However, the DCA does not recognize $L(\varphi)$: For example, on the word $\{a\}^\omega$, it loops on the middle state and accepts, even though $\{a\}^\omega \not\models \varphi$. The reason is that $\mathcal{A}^{\varphi}_{\mathbf{FG}\nu}$ checks that the greatest fixed point holds, and cannot enforce satisfaction of the least-fixed-point formula $a\mathbf{U}b$.

If only we were given a promise that $a\mathbf{U}b$ holds infinitely often, then we could conclude that such a run is accepting. We can actually get such promises: for NBA and LDBA via the non-determinism of the automaton, and for DRA via the "non-determinism" of the acceptance condition. In the next section, we investigate how to utilize such promises (Section 5.3) and how to check whether the promises are fulfilled or not (Section 5.4). △

5 The Master Theorem

We present and prove the Master Theorem: A characterization of the words satisfying a given formula from which we can easily extract deterministic, limit-deterministic, and nondeterministic automata of asymptotically optimal size.

We first provide some intuition with the help of an example. Consider the formula $\varphi = \mathbf{FG}((a\mathbf{R}b)\vee(c\mathbf{U}d))$, which does not belong to any of the fragments in the last section, and a word w. Assume we are promised that along w the μ-subformula $c\mathbf{U}d$ holds infinitely often (this is the case e.g. for $w = (\emptyset\{d\})^\omega$). In particular, we then know that d holds infinitely often, and so we can "reduce" $w \models^? \varphi$ to $w \models^? \mathbf{FG}((a\mathbf{R}b)\vee(c\mathbf{W}d))$, which belongs to the fragment $\mathbf{FG}(\nu LTL)$.

Assume now we are promised that $c\mathbf{U}d$ only holds finitely often (for example, because $w = \{d\}^4\{c\}^\omega$). Even more, we are promised that along the suffix w_5 the formula $c\mathbf{U}d$ never holds any more. How can we use this advice? First, $w \models^? \varphi$ reduces to $w_5 \models^? af(\varphi, w_{05})$ by the fundamental property of af, Lemma 3.3(1). Further, a little computation shows that $af(\varphi, w_{05}) \equiv_P \varphi$, and so that $w \models^? \varphi$ reduces to $w_5 \models^? \varphi$. Finally, using that $c\mathbf{U}d$ never holds again, we reduce $w \models^? \varphi$ to $w_5 \models^? \mathbf{FG}(a\mathbf{R}b \vee \mathbf{ff}) \equiv_P \mathbf{FG}(a\mathbf{R}b)$ which belongs to the fragment $\mathbf{FG}(\nu LTL)$.

This example suggests a general strategy for solving $w \models^? \varphi$:

- Guess the set of least-fixed-point subformulas of φ that hold infinitely often, denoted by $\mathcal{GF}_w$, and the set of greatest-fixed-point subformulas that hold almost always, denoted by $\mathcal{FG}_w$.
- Guess a *stabilization point* after which the least-fixed-point subformulas outside $\mathcal{GF}_w$ do not hold any more, and the greatest-fixed-point subformulas of $\mathcal{FG}_w$ hold forever.
- Use these guesses to reduce $w \models^? \varphi$ to problems $w \models^? \psi$ for formulas ψ that belong to the fragments introduced in the last section.
- Check that the guesses are correct.

In the rest of the section we develop this strategy. In Section 5.1 we introduce the terminology needed to formalize stabilization. Section 5.2 shows how to use a guess X for $\mathcal{GF}$ or a guess Y for $\mathcal{FG}$ to reduce $w \models^? \varphi$ to a simpler problem $w \models^? \varphi[X]_\nu$ or $w \models^? \varphi[Y]_\mu$, where $\varphi[X]_\nu$ and $\varphi[Y]_\mu$ are read as "φ with $\mathbf{GF}$-advice X" and "φ with $\mathbf{FG}$-advice Y", respectively. Section 5.3 shows how to use the advice to decide $w \models^? \varphi$. Section 5.4 shows how to check that the advice is correct. The Master Theorem is stated and proved in Section 5.5.

5.1 μ- and ν-stability.

Fix a formula φ. The set of subformulas of φ of the form $\mathbf{F}\psi$, $\psi_1\mathbf{U}\psi_2$, and $\psi_1\mathbf{M}\psi_2$ is denoted by $\mu(\varphi)$. So, loosely speaking, $\mu(\varphi)$ contains the set of subformulas of φ with a least-fixed-point operator at the top of their syntax tree. Given a word w, we are interested in which of these formulas hold infinitely often, and which ones hold at least once, i.e., we are interested in the sets

$$\mathcal{GF}_w = \{\psi \mid \psi \in \mu(\varphi) \wedge w \models \mathbf{GF}\psi\}$$
$$\mathcal{F}_w = \{\psi \mid \psi \in \mu(\varphi) \wedge w \models \mathbf{F}\psi\}$$

Observe that $\mathcal{GF}_w \subseteq \mathcal{F}_w$. We say that w is μ-stable with respect to φ if $\mathcal{GF}_w = \mathcal{F}_w$.

Example 5.1. For $\varphi = \mathbf{G}a \vee b\mathbf{U}c$ we have $\mu(\varphi) = \{b\mathbf{U}c\}$. Let $w = \{a\}^\omega$ and $w' = \{b\}\{c\}\{a\}^\omega$. We have $\mathcal{F}_w = \emptyset = \mathcal{GF}_w$ and $\mathcal{GF}_{w'} = \emptyset \subset \{b\mathbf{U}c\} = \mathcal{F}_{w'}$. So w is μ-stable with respect to φ, but w' is not. △

Dually, the set of subformulas of φ of the form $\mathbf{G}\psi$, $\psi_1\mathbf{W}\psi_2$, and $\psi_1\mathbf{R}\psi_2$ is denoted by $\nu(\varphi)$. This time we are interested in whether these formulas hold everywhere or almost everywhere, i.e., in the sets

$$\mathcal{FG}_w = \{\psi \mid \psi \in \nu(\varphi) \wedge w \models \mathbf{FG}\psi\}$$
$$\mathcal{G}_w = \{\psi \mid \psi \in \nu(\varphi) \wedge w \models \mathbf{G}\psi\}$$

(Observe that the question whether a ν-formula like, say, $\mathbf{G}a$, holds once or infinitely often makes no sense, because it holds once iff it holds infnitely often.) We have $\mathcal{FG}_w \supseteq \mathcal{G}_w$, and we say that w is ν-stable with respect to φ if $\mathcal{FG}_w = \mathcal{G}_w$.

Example 5.2. Let φ, w and w' as in Example 5.1. We have $\nu(\varphi) = \{\mathbf{G}a\}$. The word w is ν-stable, but w' is not, because $\mathcal{FG}_{w'} = \{\mathbf{G}a\} \supset \emptyset = \mathcal{G}_{w'}$. △

So not every word is μ-stable or ν-stable. However, as shown by the following lemma, all but finitely many suffixes of a word are μ- and ν-stable.

Lemma 5.3. *For every word w there exist indices $i, j \geq 0$ such that for every $k \geq 0$ the suffix w_{i+k} is μ-stable and the suffix w_{j+k} is ν-stable.*

Proof. We only prove the μ-stability part; the proof of the other part is similar. Since $\mathcal{GF}_{w_i} \subseteq \mathcal{F}_{w_i}$ for every $i \geq 0$, it suffices to exhibit an index i such that $\mathcal{GF}_{w_{i+k}} \supseteq \mathcal{F}_{w_{i+k}}$ for every $k \geq 0$. If $\mathcal{GF}_w \supseteq \mathcal{F}_w$ then we can choose $i := 0$. So assume $\mathcal{F}_w \setminus \mathcal{GF}_w \neq \emptyset$. By definition, every $\psi \in \mathcal{F}_w \setminus \mathcal{GF}_w$ holds only finitely often along w. So for every $\psi \in \mathcal{F}_w \setminus \mathcal{GF}_w$ there exists an index i_ψ such that

$w_{i_\psi+k} \not\models \psi$ for every $k \geq 0$. Let $i := \max\{i_\psi \mid \psi \in \mathcal{F}_w\}$, which exists because $\mathcal{F}_w$ is a finite set. It follows $\mathcal{GF}_{w_{i+k}} \supseteq \mathcal{F}_{w_{i+k}}$ for every $k \geq 0$, and so every w_{i+k} is μ-stable. □

Example 5.4. Let again $\varphi = \mathbf{G}a \vee b\mathbf{U}c$. The word $w' = \{b\}\{c\}\{a\}^\omega$ is neither μ-stable nor ν-stable, but all suffixes $w'_{(2+k)}$ of w' are both μ-stable and ν-stable. △

5.2 The formulas $\varphi[X]_\nu$ and $\varphi[Y]_\mu$.

We first introduce $\varphi[X]_\nu$. Assume we have to determine if a word w satisfies φ, and we are told that w is μ-stable. Further, we are given the set $X \subseteq \mu(\varphi)$ such that $\mathcal{GF}_w = X = \mathcal{F}_w$. We use this oracle information to reduce the problem $w \models^? \varphi$ to a "simpler" problem $w \models^? \varphi[X]_\nu$, where "simpler" means that $\varphi[X]_\nu$ is a formula of νLTL, for which we already know how to construct automata. In other words, we define a formula $\varphi[X]_\nu \in \nu LTL$ such that $\mathcal{GF}_w = X = \mathcal{F}_w$ implies $w \models \varphi$ iff $w \models \varphi[X]_\nu$. (Observe that $X \subseteq \mu(\varphi)$ but $\varphi[X]_\nu \in \nu LTL$, and so the latter, not the former, is the reason for the ν-subscript in the notation $\varphi[X]_\nu$.)

The definition of $\varphi[X]_\nu$ is purely syntactic, and the intuition behind it is very simple. All the main ideas are illustrated by the following examples, where we assume $\mathcal{GF}_w = X = \mathcal{F}_w$:

- $\varphi = \mathbf{F}a \wedge \mathbf{G}b$ and $X = \{\mathbf{F}a\}$. Then $\mathbf{F}a \in \mathcal{GF}_w$, which implies in particular $w \models \mathbf{F}a$. So we can reduce $w \models^? \mathbf{F}a \wedge \mathbf{G}b$ to $w \models^? \mathbf{G}b$, and so $\varphi[X]_\nu := \mathbf{G}b$.
- $\varphi = \mathbf{F}a \wedge \mathbf{G}b$ and $X = \emptyset$. Then $\mathbf{F}a \notin \mathcal{F}_w$, and so $w \not\models \mathbf{F}a$. So we can reduce $w \models^? \mathbf{F}a \wedge \mathbf{G}b$ to the trivial problem $w \models^? \mathbf{ff}$, and so $\varphi[X]_\nu := \mathbf{ff}$.
- $\varphi = \mathbf{G}(b\mathbf{U}c)$ and $X = \{b\mathbf{U}c\}$. Then $b\mathbf{U}c \in \mathcal{GF}_w$, and so $w \models \mathbf{GF}(b\mathbf{U}c)$. This does not imply $w \models b\mathbf{U}c$, but implies that c will hold in the future. So we can reduce $w \models^? \mathbf{G}(b\mathbf{U}c)$ to $w \models^? \mathbf{G}(b\mathbf{W}c)$, a formula of νLTL, and so $\varphi[X]_\nu := \mathbf{G}(b\mathbf{W}c)$.

Definition 5.5. Let φ be a formula and let $X \subseteq \mu(\varphi)$. The formula $\varphi[X]_\nu$ is inductively defined as follows:

- If $\varphi = \mathbf{tt}, \mathbf{ff}, a, \neg a$, then $\varphi[X]_\nu = \varphi$.
- If $\varphi = op(\psi)$ for $op \in \{\mathbf{X}, \mathbf{G}\}$ then $\varphi[X]_\nu = op(\psi[X]_\nu)$.
- If $\varphi = op(\psi_1, \psi_2)$ for $op \in \{\wedge, \vee, \mathbf{W}, \mathbf{R}\}$ then $\varphi[X]_\nu = op(\psi_1[X]_\nu, \psi_2[X]_\nu)$.
- If $\varphi = \mathbf{F}\psi$ then $\varphi[X]_\nu = \begin{cases} \mathbf{tt} & \text{if } \varphi \in X \\ \mathbf{ff} & \text{otherwise.} \end{cases}$
- If $\varphi = \psi_1\mathbf{U}\psi_2$ then $\varphi[X]_\nu = \begin{cases} (\psi_1[X]_\nu)\mathbf{W}(\psi_2[X]_\nu) & \text{if } \varphi \in X \\ \mathbf{ff} & \text{otherwise.} \end{cases}$
- If $\varphi = \psi_1\mathbf{M}\psi_2$ then $\varphi[X]_\nu = \begin{cases} (\psi_1[X]_\nu)\mathbf{R}(\psi_2[X]_\nu) & \text{if } \varphi \in X \\ \mathbf{ff} & \text{otherwise.} \end{cases}$

We now introduce, in a dual way, a formula $\varphi[Y]_\mu \in \mu LTL$ such that $\mathcal{FG}_w = Y = \mathcal{G}_w$ implies $w \models \varphi$ iff $w \models \varphi[Y]_\mu$.

Definition 5.6. Let φ be a formula and let $Y \subseteq \nu(\varphi)$. The formula $\varphi[Y]_\mu$ is inductively defined as follows:

- If $\varphi = \mathbf{tt}, \mathbf{ff}, a, \neg a$, then $\varphi[Y]_\mu = \varphi$.
- If $\varphi = op(\psi)$ for $op \in \{\mathbf{X}, \mathbf{F}\}$ then $\varphi[Y]_\mu = op(\psi[Y]_\mu)$.
- If $\varphi = op(\psi_1, \psi_2)$ for $op \in \{\wedge, \vee, \mathbf{U}, \mathbf{M}\}$ then $\varphi[Y]_\mu = op(\psi_1[Y]_\mu, \psi_2[Y]_\mu)$.
- If $\varphi = \mathbf{G}\psi$ then $\varphi[Y]_\mu = \begin{cases} \mathbf{tt} & \text{if } \varphi \in Y \\ \mathbf{ff} & \text{otherwise.} \end{cases}$
- If $\varphi = \psi_1\mathbf{W}\psi_2$ then $\varphi[Y]_\mu = \begin{cases} \mathbf{tt} & \text{if } \varphi \in Y \\ (\psi_1[Y]_\mu)\mathbf{U}(\psi_2[Y]_\mu) & \text{otherwise.} \end{cases}$
- If $\varphi = \psi_1\mathbf{R}\psi_2$ then $\varphi[Y]_\mu = \begin{cases} \mathbf{tt} & \text{if } \varphi \in Y \\ (\psi_1[Y]_\mu)\mathbf{M}(\psi_2[Y]_\mu) & \text{otherwise.} \end{cases}$

Example 5.7. Let $\varphi = ((a\mathbf{W}b) \wedge \mathbf{F}c) \vee a\mathbf{U}d$. We have:

$$\begin{array}{lcll} \varphi[\{\mathbf{F}c\}]_\nu & = & ((a\mathbf{W}b) \wedge \mathbf{tt}) \vee \mathbf{ff} & \equiv_P \; a\mathbf{W}b \\ \varphi[\{a\mathbf{U}d\}]_\nu & = & ((a\mathbf{W}b) \wedge \mathbf{ff}) \vee a\mathbf{W}d & \equiv_P \; a\mathbf{W}d \\ \varphi[\emptyset]_\nu & = & ((a\mathbf{W}b) \wedge \mathbf{ff}) \vee \mathbf{ff} & \equiv_P \; \mathbf{ff} \\ \varphi[\{a\mathbf{W}b\}]_\mu & = & (\mathbf{tt} \wedge \mathbf{F}c) \vee a\mathbf{U}d & \equiv_P \; \mathbf{F}c \vee a\mathbf{U}d \\ \varphi[\emptyset]_\mu & = & (a\mathbf{U}b \wedge \mathbf{F}c) \vee a\mathbf{U}d & \end{array}$$

△

5.3 Utilizing $\varphi[X]_\nu$ and $\varphi[Y]_\mu$.

The following lemma states the fundamental properties of $\varphi[X]_\nu$ and $\varphi[Y]_\mu$. As announced above, for a μ-stable word w we can reduce the problem $w \models^? \varphi$ to $w \models^? \varphi[X]_\nu$, and for a ν-stable word to $w \models^? \varphi[Y]_\mu$. However, there is more: If we only know $X \subseteq \mathcal{GF}_w$, then we can still infer $w \models \varphi$ from $w \models \varphi[X]_\nu$, only the implication in the other direction fails.

Lemma 5.8. *Let φ be a formula and let w be a word.*
For every $X \subseteq \mu(\varphi)$:

(a1) *If $\mathcal{F}_w \subseteq X$ and $w \models \varphi$, then $w \models \varphi[X]_\nu$.*
(a2) *If $X \subseteq \mathcal{GF}_w$ and $w \models \varphi[X]_\nu$, then $w \models \varphi$.*

In particular:

(a3) *If $\mathcal{F}_w = X = \mathcal{GF}_w$ then $w \models \varphi$ iff $w \models \varphi[X]_\nu$.*

For every $Y \subseteq \nu(\varphi)$:

(b1) *If $\mathcal{FG}_w \subseteq Y$ and $w \models \varphi$, then $w \models \varphi[Y]_\mu$.*
(b2) *If $Y \subseteq \mathcal{G}_w$ and $w \models \varphi[Y]_\mu$, then $w \models \varphi$.*

In particular:

(b3) *If $\mathcal{FG}_w = Y = \mathcal{G}_w$ then $w \models \varphi$ iff $w \models \varphi[Y]_\mu$.*

Proof. All parts are proved by a straightforward structural induction on φ. We consider only (a1), and only two representative cases of the induction. Representative cases for (a2), (b1), and (b2) can be found in the appendix of [16].

(a1) Assume $\mathcal{F}_w \subseteq X$. Then $\mathcal{F}_{w_i} \subseteq X$ for all $i \geq 0$. We prove the following stronger statement via structural induction on φ:

$$\forall i.\ (\ (w_i \models \varphi) \rightarrow (w_i \models \varphi[X]_\nu)\ )$$

We consider one representative of the "interesting" cases, and one of the "straightforward" cases.
Case $\varphi = \psi_1\mathbf{U}\psi_2$: Let $i \geq 0$ arbitrary and assume $w_i \models \psi_1\mathbf{U}\psi_2$. Then $\psi_1\mathbf{U}\psi_2 \in \mathcal{F}_{w_i}$ and so $\varphi \in X$. We prove $w_i \models (\psi_1\mathbf{U}\psi_2)[X]_\nu$:

$$\begin{array}{lll} & w_i \models \psi_1\mathbf{U}\psi_2 & \\ \Longrightarrow & w_i \models \psi_1\mathbf{W}\psi_2 & \\ \Longrightarrow & \forall j.\ w_{i+j} \models \psi_1 \vee \exists k \leq j.\ w_{i+k} \models \psi_2 & \\ \Longrightarrow & \forall j.\ w_{i+j} \models \psi_1[X]_\nu \vee \exists k \leq j.\ w_{i+k} \models \psi_2[X]_\nu & \text{(I.H.)} \\ \Longrightarrow & w_i \models (\psi_1[X]_\nu)\mathbf{W}(\psi_2[X]_\nu) & \\ \Longrightarrow & w_i \models (\psi_1\mathbf{U}\psi_2)[X]_\nu & (\varphi \in X, \text{Def. 5.5}) \end{array}$$

Case $\varphi = \psi_1 \vee \psi_2$: Let $i \geq 0$ arbitrary and assume $w_i \models \psi_1 \vee \psi_2$:

$$\begin{array}{lll} & w_i \models \psi_1 \vee \psi_2 & \\ \Longrightarrow & (w_i \models \psi_1) \vee (w_i \models \psi_2) & \\ \Longrightarrow & (w_i \models \psi_1[X]_\nu) \vee (w_i \models \psi_2[X]_\nu) & \text{(I.H.)} \\ \Longrightarrow & w_i \models (\psi_1 \vee \psi_2)[X]_\nu & \text{(Def. 5.5)} \quad \square \end{array}$$

Lemma 5.8 suggests to decide $w \models^? \varphi$ by "trying out" all possible sets X. Part (a2) shows that the strategy of checking for every set X if both $X \subseteq \mathcal{GF}_w$ and $w \models \varphi[X]_\nu$ hold is sound.

Example 5.9. Consider $\varphi = \mathbf{GF}a \vee \mathbf{GF}(b \wedge \mathbf{G}c)$. Since $\mu(\varphi) = \{\mathbf{F}a, \mathbf{F}(b \wedge \mathbf{G}c)\}$, there are four possible X's to be tried out: $\emptyset$, $\{\mathbf{F}a\}$, $\{\mathbf{F}(b \wedge \mathbf{G}c)\}$, and $\{\mathbf{F}a, \mathbf{F}(b \wedge \mathbf{G}c)\}$. For $X = \emptyset$ we get $\varphi[X]_\nu = \mathbf{ff}$, indicating that if neither a nor $b \wedge \mathbf{G}c$ hold infinitely often, then φ cannot hold. For the other three possibilities (a holds infinitely often, $b \wedge \mathbf{G}c$ holds infinitely often, or both) there are words satisfying φ, like a^ω, $\{b, c\}^\omega$, and $\{a, b, c\}^\omega$. △

However there are still two questions open. First, is this strategy complete? Part (a3) shows that it is complete for μ-stable words: Indeed, in this case there is a set X such that $\mathcal{GF}_w = X = \mathcal{F}_w$, and for this particular set $w \models \varphi[X]_\nu$ holds. For words that are not μ-stable, we will use the existence of μ-stable suffixes: Instead of checking $w \models \varphi[X]_\nu$, we will check the existence of a suffix w_i such that $w_i \models af(\varphi, w_{0i})[X]_\nu$. This will happen in Section 5.5. The second open question is simply how to check $X \subseteq \mathcal{GF}_w$. We deal with it in Section 5.4.

5.4 Checking $X \subseteq \mathcal{GF}_w$ and $Y \subseteq \mathcal{FG}_w$.

Consider again the formula $\varphi = \mathbf{GF}a \vee \mathbf{GF}(b \wedge \mathbf{G}c)$ of Example 5.9. If $X = \{\mathbf{F}a\}$, then checking whether X is a correct advice (i.e., whether $X \subseteq \mathcal{GF}_w$ holds) is easy, because $\mathbf{GFF}a \in \mathbf{GF}(\mu LTL)$, see Proposition 4.2. In contrast, for $X = \{\mathbf{F}(b \wedge \mathbf{G}c)\}$ this is not so. In this case it would come handy if we had an advice $Y = \{\mathbf{G}c\}$ promising that $\mathbf{G}c$ holds almost always, as is the case for e.g. $\emptyset^5(\{b, c\}\{c\})^\omega$. Indeed, we could easily check correctness of this advice, because $\mathbf{FGG}c \in \mathbf{FG}(\nu LTL)$, and with its help checking $\mathbf{GF}(b \wedge \mathbf{G}c)$ reduces to checking $\mathbf{GF}(b \wedge \mathbf{tt}) = \mathbf{GF}b$, which is also easy.

One of the main ingredients of our approach is that in order to verify a promise $X \subseteq \mathcal{GF}_w$ we can rely on a promise $Y \subseteq \mathcal{FG}_w$ about subformulas of X, and vice versa. There is no circularity in this rely/guarantee reasoning because the subformula order is well founded, and we eventually reach formulas ψ such that $\psi[X]_\nu = \psi$ or $\psi[Y]_\mu = \psi$. This argument is formalized in the next lemma. The first part of the lemma states that mutually assuming correctness of the other promise is correct. The second part states that, loosely speaking, this rely/guarantee method is complete.

Lemma 5.10. *Let φ be a formula and let w be a word.*

(1.) *For every $X \subseteq \mu(\varphi)$ and $Y \subseteq \nu(\varphi)$, if*

$$\forall \psi \in X.\ w \models \mathbf{GF}(\psi[Y]_\mu)$$
$$\forall \psi \in Y.\ w \models \mathbf{FG}(\psi[X]_\nu)$$

then $X \subseteq \mathcal{GF}_w$ and $Y \subseteq \mathcal{FG}_w$.

(2.) *If $X = \mathcal{GF}_w$ and $Y = \mathcal{FG}_w$ then*

$$\forall \psi \in X.\ w \models \mathbf{GF}(\psi[Y]_\mu)$$
$$\forall \psi \in Y.\ w \models \mathbf{FG}(\psi[X]_\nu)$$

Proof. (1.) Let $X \subseteq \mu(\varphi)$ and $Y \subseteq \nu(\varphi)$. Observe that $X \cap Y = \emptyset$. Let $n := |X \cup Y|$. Let $\psi_1, \ldots, \psi_n$ be an enumeration of $X \cup Y$ compatible with the subformula order, i.e., if ψ_i is a subformula of ψ_j, then $i \leq j$. Finally, let $(X_0, Y_0), (X_1, Y_1), \ldots, (X_n, Y_n)$ be the unique sequence of pairs satisfying:

- $(X_0, Y_0) = (\emptyset, \emptyset)$ and $(X_n, Y_n) = (X, Y)$.
- For every $0 < i \leq n$, if $\psi_i \in X$ then $X_i \setminus X_{i-1} = \{\psi_i\}$ and $Y_i = Y_{i-1}$, and if $\psi_i \in Y$, then $X_i = X_{i-1}$ and $Y_i \setminus Y_{i-1} = \{\psi_i\}$.

We prove $X_i \subseteq \mathcal{GF}_w$ and $Y_i \subseteq \mathcal{FG}_w$ for every $0 \leq i \leq n$ by induction on i. For $i = 0$ the result follows immediately from $X_0 = \emptyset = Y_0$. For $i > 0$ we consider two cases:

Case 1: $\psi_i \in Y$, i.e., $X_i = X_{i-1}$ and $Y_i \setminus Y_{i-1} = \{\psi_i\}$.

By induction hypothesis and $X_i = X_{i-1}$ we have $X_i \subseteq \mathcal{GF}_w$ and $Y_{i-1} \subseteq \mathcal{FG}_w$. We prove $\psi_i \in \mathcal{FG}_w$, i.e., $w \models \mathbf{FG}\psi_i$, in three steps.
Claim 1: $\psi_i[X]_\nu = \psi_i[X_i]_\nu$.
By the definition of the $\cdot[\cdot]_\nu$ mapping, $\psi_i[X]_\nu$ is completely determined by the μ-subformulas of ψ_i that belong to X. By the definition of the sequence $(X_0, Y_0), \ldots, (X_n, Y_n)$, a μ-subformula of ψ_i belongs to X iff it belongs to X_i, and we are done.
Claim 2: $X_i \subseteq \mathcal{GF}_{w_k}$ for every $k \geq 0$.
Follows immediately from $X_i \subseteq \mathcal{GF}_w$.
Proof of $w \models \mathbf{FG}\psi_i$. By the assumption of the lemma we have $w \models \mathbf{FG}(\psi_i[X]_\nu)$, and so, by Claim 1, $w \models \mathbf{FG}(\psi_i[X_i]_\nu)$. So there exists an index j such that $w_{j+k} \models \psi_i[X_i]_\nu$ for every $k \geq 0$. By Claim 2 we further have $X_i \subseteq \mathcal{GF}_{w_{j+k}}$ for every $j, k \geq 0$. So we can apply part (a2) of Lemma 5.8 to X_i, w_{j+k}, and ψ_i, which yields $w_{j+k} \models \psi_i$ for every $k \geq 0$. So $w \models \mathbf{FG}\psi_i$.

Case 2: $\psi_i \in X$, i.e., $X_i \setminus X_{i-1} = \{\psi_i\}$ and $Y_i = Y_{i-1}$.
In this case $X_{i-1} \subseteq \mathcal{GF}_w$ and $Y_i \subseteq \mathcal{FG}_w$. We prove $\psi_i \in \mathcal{GF}_w$, i.e., $w \models \mathbf{GF}\psi_i$ in three steps.
Claim 1: $\psi_i[Y]_\mu = \psi_i[Y_i]_\mu$.
The claim is proved as in Case 1.
Claim 2: There is an $j \geq 0$ such that $Y_i \subseteq \mathcal{G}_{w_k}$ for every $k \geq j$.
Follows immediately from $Y_i \subseteq \mathcal{FG}_w$.
Proof of $w \models \mathbf{GF}\psi_i$. By the assumption of the lemma we have $w \models \mathbf{GF}(\psi_i[Y]_\mu)$. Let j be the index of Claim 2. By Claim 1 we have $w \models \mathbf{GF}(\psi_i[Y_i]_\mu)$, and so there exist infinitely many $k \geq j$ such that $w_k \models \psi_i[Y_i]_\mu$. By Claim 2 we further have $Y_i \subseteq \mathcal{G}_{w_k}$. So we can apply part (b2) of Lemma 5.8 to Y_i, w_k, and ψ_i, which yields $w_k \models \psi_i$ for infinitely many $k \geq j$. So $w \models \mathbf{GF}\psi_i$.

(2.) Let $\psi \in \mathcal{GF}_w$. We have $w \models \mathbf{GF}\psi$, and so $w_i \models \psi$ for infinitely many $i \geq 0$. Since $\mathcal{FG}_{w_i} = \mathcal{FG}_w$ for every $i \geq 0$, part (b1) of Lemma 5.8 can be applied to w_i, $\mathcal{FG}_{w_i}$, and ψ. This yields $w_i \models \psi[\mathcal{FG}_w]_\mu$ for infinitely many $i \geq 0$ and thus $w \models \mathbf{GF}(\psi[\mathcal{FG}_w]_\mu)$.

Let $\psi \in \mathcal{FG}_w$. Since $w_i \models \mathbf{FG}\psi$, there is an index j such that $w_{j+k} \models \psi$ for every $k \geq 0$. By Lemma 5.3 the index j can be chosen so that it also satisfies $\mathcal{GF}_w = \mathcal{F}_{w_{j+k}} = \mathcal{GF}_{w_{j+k}}$ for every $k \geq 0$. So part (a1) of Lemma 5.8 can be applied to $\mathcal{F}_{w_{j+k}}$, w_{j+k}, and ψ. This yields $w_{j+k} \models \psi[\mathcal{GF}_w]_\nu$ for every $k \geq 0$ and thus $w \models \mathbf{FG}(\psi[\mathcal{GF}_w]_\nu)$. □

Example 5.11. Let $\varphi = \mathbf{F}(a \wedge \mathbf{G}(b \vee \mathbf{F}c))$, $X = \{\varphi\}$, and $Y = \{\mathbf{G}(b \vee \mathbf{F}c)\}$.

- The condition $\forall \psi \in X.\ w \models \mathbf{GF}(\psi[Y]_\mu)$ becomes
$$w \models \mathbf{GF}\left(\varphi[Y]_\mu\right) = \mathbf{GF}(\mathbf{F}a) \equiv \mathbf{GF}a$$
- The condition $\forall \psi \in Y.\ w \models \mathbf{FG}(\psi[X]_\nu)$ becomes
$$w \models \mathbf{FG}\left(\mathbf{G}(b \vee \mathbf{F}c)[X]_\nu\right) = \mathbf{FG}(\mathbf{G}b) \equiv \mathbf{FG}b$$

By Lemma 5.10 (1) we then have that $w \models \mathbf{GF}a \wedge \mathbf{FG}b$ implies $\varphi \in \mathcal{GF}_w$ and $\mathbf{G}(b \vee \mathbf{F}c)) \in \mathcal{FG}_w$. △

5.5 Putting the pieces together: The Master Theorem.

Putting together Lemma 5.8 and Lemma 5.10, we obtain the main result of the paper, which we will use as "Master Theorem" for the construction of automata.

Theorem 5.12 (Master Theorem)**.** *For every formula φ and for every word w: $w \models \varphi$ iff there exists $X \subseteq \mu(\varphi)$ and $Y \subseteq \nu(\varphi)$ satisfying*

$$(1)\quad \exists i.\ w_i \models af(\varphi, w_{0i})[X]_\nu$$
$$(2)\quad \forall \psi \in X.\ w \models \mathbf{GF}(\psi[Y]_\mu)$$
$$(3)\quad \forall \psi \in Y.\ w \models \mathbf{FG}(\psi[X]_\nu)$$

Observe that $af(\varphi, w_{0i})[X]_\nu$, $\mathbf{GF}(\psi[Y]_\mu)$, and $\mathbf{FG}(\psi[X]_\nu)$ are formulas of νLTL, $\mathbf{GF}(\mu LTL)$, and $\mathbf{FG}(\nu LTL)$, respectively, i.e., they all belong to the fragments of Section 4.

Before proving the theorem, let us interpret it in informal terms. The Master Theorem states that in order to decide $w \models^? \varphi$ we can guess two sets $X \subseteq \mu(\varphi)$ and $Y \subseteq \nu(\varphi)$ and an index i, and then proceed as follows: verify $Y \subseteq \mathcal{FG}_w$ assuming that $X \subseteq \mathcal{GF}_w$ holds (3), verify $X \subseteq \mathcal{GF}_w$ assuming that $Y \subseteq \mathcal{FG}_w$ holds (2), and verify $w_i \models af(\varphi, w_{0i})$ assuming that $X \subseteq \mathcal{GF}_w$ holds (1). The procedure is sound by Lemma 5.8 and Lemma 5.10, and complete because the guess where $X := \mathcal{GF}_w$, $Y := \mathcal{FG}_w$, and i is a stabilization point of w, is guaranteed to succeed.

Example 5.13. Let $\varphi = \mathbf{F}(a \wedge \mathbf{G}(b \vee \mathbf{F}c))$ as in Example 5.11, and let $\varphi' = d\mathbf{U}\varphi$. For $X = \{\varphi, \varphi'\}$, $Y = \{\mathbf{G}(b \vee \mathbf{F}c)\}$, and $i = 0$ the Master Theorem yields that $w \models \varphi'$ is implied by

(1) $w \models (d\mathbf{U}\varphi)[X]_\nu = d\mathbf{W}(\varphi[X]_\nu) = d\mathbf{W}\mathbf{tt} \equiv \mathbf{tt}$,
(2) $w \models \mathbf{GF}(\varphi[Y]_\mu) \wedge \mathbf{GF}(\varphi'[Y]_\mu) = \mathbf{GF}a \wedge \mathbf{GF}(d\mathbf{U}(\mathbf{F}a))$, and
(3) $w \models \mathbf{FG}((\mathbf{G}(b \vee \mathbf{F}c)[X]_\nu) \equiv \mathbf{FG}b$.

For $X = \{\varphi\}$, $Y = \{\mathbf{G}(b \vee \mathbf{F}c)\}$, and $i = 0$, condition (1) is $w \models \mathbf{ff}$, and we do not derive any useful information. △

Proof (of the Master Theorem).
($\Rightarrow$): Assume $w \models \varphi$, and set $X := \mathcal{GF}_w$ and $Y := \mathcal{FG}_w$. Properties (2) and (3) follow from Lemma 5.10. For property (1), let i be an index such that $\mathcal{F}_{w_i} = \mathcal{GF}_{w_i}$; this index exists by Lemma 5.3. By Lemma 3.3 we have $w_i \models af(\varphi, w_{0i})$, and by Lemma 5.8 (a1) $w_i \models af(\varphi, w_{0i})[X]_\nu$.

($\Leftarrow$): Assume that properties (1-3) hold for sets X, Y and an index i. By Lemma 5.10 (1.) we have $X \subseteq \mathcal{GF}_w$, and so $X \subseteq \mathcal{GF}_{w_i}$. By Lemma 5.8 (a2) we obtain $w_i \models af(\varphi, w_{0i})[X]_\nu$, and thus $w_i \models af(\varphi, w_{0i})$. Lemma 3.3 yields $w \models \varphi$. □

Let $L^j_{X,Y}$ be the language of all words that satisfy condition (j) of the Master Theorem for the sets X and Y. The Master Theorem can then be reformulated as:

$$L(\varphi) = \bigcup_{\substack{X \subseteq \mu(\varphi) \\ Y \subseteq \nu(\varphi)}} L^1_{X,Y} \cap L^2_{X,Y} \cap L^3_{X,Y}$$

Therefore, given an automata model effectively closed under union and intersection, in order to construct automata for all of LTL it suffices to exhibit automata recognizing $L^1_{X,Y}, L^2_{X,Y}, L^3_{X,Y}$. In the next section we consider the case of DRAs, and then we proceed to NBAs and LDBAs.

6 Constructing DRAs for LTL Formulas

Let φ be a formula of length n. We use the Master Theorem to construct a DRA for $L(\varphi)$ with $2^{2^{O(n)}}$ states and $O(2^n)$ Rabin pairs. Since our purpose is only to show that we can easily obtain automata of asymptotically optimal size, we give priority to a simpler construction over one with the least number of states. We comment in Section 9 on optimizations that reduce the size by using other acceptance conditions.

We first construct DRAs for $L^1_{X,Y}$, $L^2_{X,Y}$, and $L^3_{X,Y}$ with $2^{2^{O(n)}}$ states and one single Rabin pair. More precisely, for each of these languages we construct either a DBA or a DCA. We then construct a DRA for $L(\varphi)$ by means of intersections and unions.

A DCA for $L^1_{X,Y}$. We define a DCA $C_{\varphi,X}$ that accepts a word w iff $w_i \models af(\varphi, w_{0i})[X]_\nu$ for some suffix w_i of w. In the rest of this part of the section we abbreviate $af(\varphi, w_{0i})$ to φ_i. Recall that $\varphi_i[X]_\nu$ is a formula of νLTL, and so for every $i \geq 0$ there is a DCA with a state $\mathbf{ff}$ such that the automaton rejects iff it reaches this state. Intuitively, if the automaton rejects, then it rejects "after finite time". We prove the following lemma:

Lemma 6.1. *Let $\varphi_i := af(\varphi, w_{0i})$. If $w \models \varphi[X]_\nu$ then $w_i \models \varphi_i[X]_\nu$ for all $i > 0$.*

Proof. Assume $w \models \varphi[X]_\nu$. It suffices to prove $w_1 \models \varphi_1[X]_\nu$, since the general case follows immediately by induction. For $i = 1$ we proceed by structural induction on φ, and consider only some representative cases.
Case $\varphi = a$**.** Since $w \models a[X]_\nu = a$ we have $a \in w[0]$. So $\varphi_1[X]_\nu = \mathbf{tt}[X]_\nu = \mathbf{tt}$, and thus $w_1 \models \varphi_1[X]_\nu$.
Case $\varphi = \psi \mathbf{U} \chi$**.** Since $w \models \varphi[X]_\nu$ we have $\varphi[X]_\nu \neq \mathbf{ff}$, and so $\varphi \in X$. We have:

$$\begin{aligned}
& w \models \varphi[X]_\nu \\
\Longrightarrow\ & w \models (\psi[X]_\nu)\mathbf{W}(\chi[X]_\nu) && \text{(Def. 5.5)} \\
\Longrightarrow\ & w \models (\psi[X]_\nu \wedge \mathbf{X}((\psi[X]_\nu)\mathbf{W}(\chi[X]_\nu))) \vee \chi[X]_\nu \\
\Longrightarrow\ & w \models (\psi[X]_\nu \wedge \mathbf{X}((\psi\mathbf{U}\chi)[X]_\nu)) \vee \chi[X]_\nu && (\varphi \in X) \\
\Longrightarrow\ & w_1 \models (\psi_1[X]_\nu \wedge \varphi[X]_\nu) \vee \chi_1[X]_\nu && \text{(I.H.)} \\
\Longrightarrow\ & w_1 \models ((\psi_1 \wedge (\psi\mathbf{U}\chi)) \vee \chi_1)[X]_\nu && \text{(Def. 5.5)} \\
\Longrightarrow\ & w_1 \models \varphi_1[X]_\nu && \text{(Def. 3.1)}
\end{aligned}$$

□

Loosely speaking, $C_{\varphi,X}$ starts by checking $w \models^? \varphi[X]_\nu$. For this it maintains the formula $(\varphi[X]_\nu)_i$ in its state. If the formula becomes $\mathbf{ff}$ after, say, j steps, then $w \not\models \varphi[X]_\nu$, and $C_{\varphi,X}$ proceeds to check $w \models^? \varphi_j[X]_\nu$. In order to "switch" to this new problem, $C_{\varphi,X}$ needs to know φ_j, and so it maintains φ_j it in its state. In other words, after j steps $C_{\varphi,X}$ is in state $(\varphi_j, af(\varphi_i[X]_\nu, w_{ij}))$, where $i \leq j$ is the number of steps after which $C_{\varphi,X}$ switched to a new problem for the last time. If the second component of the state becomes $\mathbf{ff}$, then the automaton uses the first component to determine which formula to check next. The accepting condition states that the transitions leading to a state of the form $(\psi, \mathbf{ff})$ must occur finitely often, which implies that eventually one of the checks $w \models^? \varphi_j[X]_\nu$ succeeds.

The formal description of $C_{\varphi,X}$ is as follows:

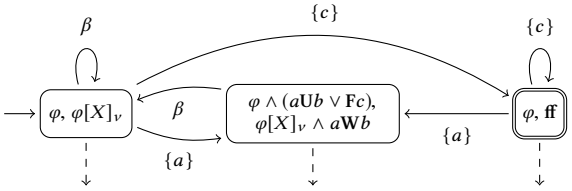
$$C_{\varphi,X} = (Reach(\varphi) \times Reach(\varphi)[X]_\nu, \delta, (\varphi, \varphi[X]_\nu), fin(F))$$

where

- $Reach(\varphi)[X]_\nu = \bigcup_{\psi \in Reach(\varphi)} Reach(\psi[X]_\nu)$
- $\delta((\xi, \zeta), \nu) = \begin{cases} (af(\xi, \nu), af(\xi[X]_\nu, \nu)) & \text{if } \zeta \equiv_P \mathbf{ff} \\ (af(\xi, \nu), af(\zeta, \nu)) & \text{otherwise.} \end{cases}$
- $F = Reach(\varphi) \times \{\mathbf{ff}\}$

Since $Reach(\varphi)$ has at most size 2^{2^n}, the number of states of $C_{\varphi,X}$ is bounded by $\left(2^{2^n}\right)^2 = 2^{O(2^n)}$.

Example 6.2. Let $\varphi = \mathbf{G}(a\mathbf{U}b \vee \mathbf{F}c)$, $X = \{a\mathbf{U}b\}$, and $\varphi[X]_\nu = \mathbf{G}(a\mathbf{W}b)$. Below we show a fragment of $C_{\varphi,X}$, with α, β, γ as in Example 4.3.



For $w = \{c\}\{c\}(\{a\}\{b\})^\omega$ we have $X = \mathcal{GF}_w$; the word is accepted. For $w' = \{c\}^\omega$ we have $X \neq \mathcal{GF}_{w'}$, and the word is rejected. △

A DBA for $L^2_{X,Y}$**.** We define a DBA recognizing $L\left(\bigwedge_{\psi\in X} \mathbf{GF}(\psi[Y]_\mu)\right)$. Observe that $\mathbf{GF}(\psi[Y]_\mu) \in \mathbf{GF}(\mu LTL)$ for every $\psi \in X$, and that $\psi[Y]_\mu$ has at most n subformulas. By Proposition 4.2, $L(\mathbf{GF}(\psi[Y]_\mu)$ is recognized by a DBA with at most $2^{2^{O(n)}}$ states. Recall that the intersection of the languages of k DBAs with $s_1, \ldots, s_k$ states is recognized by a DBA with $k \cdot \prod_{j=1}^k s_j$ states. Since $|X| \leq n$, the intersection of the DBAs for the formulas $\mathbf{GF}(\psi[Y]_\mu)$ yields a DBA with at most $n \cdot \left(2^{2^{O(n)}}\right)^n = 2^{n2^{O(n)}} = 2^{2^{O(n)}}$ states.

A DCA for $L^3_{X,Y}(\varphi)$**.** The DCA for $L\left(\bigwedge_{\psi\in Y} \mathbf{FG}(\psi[X]_\nu\right)$ is obtained dually to the previous case, applying $\mathbf{FG}(\psi[X]_\nu) \in \mathbf{FG}(\nu LTL)$, and Proposition 4.2.

A DRA for $L(\varphi)$**.** By the Master Theorem we have:

$$L(\varphi) = \bigcup_{\substack{X\subseteq\mu(\varphi)\\ Y\subseteq\nu(\varphi)}} L^1_{X,Y} \cap L^2_{X,Y} \cap L^3_{X,Y}$$

We first construct a DRA $A_{X,Y}$ for the intersection of $L^i_{X,Y}$, where $i = 1, 2, 3$. Let $A^i_{X,Y}$ be the DCA or DBA for $L^i_{X,Y}$. The set of states of $A_{X,Y}$ is the cartesian product of the sets of states of the $A^i_{X,Y}$, the transition function is as usual, and the accepting condition is

$$fin\left((S_1 \times Q_2 \times Q_3) \cup (Q_1 \times Q_2 \times S_3)\right) \wedge inf(Q_1 \times S_2 \times Q_3)$$

where Q_i is the set of states of $A^i_{X,Y}$, and $fin(S_1)$, $inf(S_2)$, $fin(S_3)$ are the accepting conditions of $A^1_{X,Y}$, $A^2_{X,Y}$, and $A^3_{X,Y}$.

We construct a DRA A_φ for $L(\varphi)$. Since $X \subseteq \mu(\varphi)$ and $Y \subseteq \nu(\varphi)$, there are at most 2^n pairs of sets X, Y. Let $A_1, \ldots, A_k$ be an enumeration of the DRAs for these pairs, where $k \leq 2^n$, and let Q_i and $\alpha_i = fin(U_i) \wedge inf(V_i)$ be the set of states and accepting condition of A_i, repectively. The set of states of A_φ is $Q_1 \times \cdots \times Q_k$, the transition function is as usual, and the accepting condition is

$$\bigvee_{i=1}^k \begin{array}{l} fin(Q_1 \times \cdots \times Q_{i-1} \times U_i \times Q_{i+1} \times \cdots \times Q_k) \quad \wedge \\ inf(Q_1 \times \cdots \times Q_{i-1} \times V_i \times Q_{i+1} \times \cdots \times Q_k) \end{array}$$

So A_φ has $\left(2^{2^{O(n)}}\right)^{2^n} = 2^{2^{O(n)}\cdot 2^n} = 2^{2^{O(n)}}$ states and at most 2^n Rabin pairs.

7 Constructing NBAs for LTL Formulas

Assume that φ has length n. We use the Master Theorem to construct a NBA for $L(\varphi)$ with $2^{O(n)}$ states.

We first describe how to construct NBAs for the LTL fragments of Section 4. Let us start with some informal intuition. Consider the formula $\varphi = \mathbf{GX}(a \vee b)$. In the DRA for φ we find states for the formulas φ and $af(\varphi, \emptyset)$ and a transition

$$\varphi \xrightarrow{\emptyset} af(\varphi, \emptyset)$$

where $af(\varphi, \emptyset) \equiv_P \varphi \wedge (a \vee b)$. The languages recognized from the states φ and $af(\varphi, \emptyset)$ are precisely $L(\varphi)$ and $L(af(\varphi, \emptyset))$. The basic principle for the construction of the NBAs is to put $af(\varphi, \emptyset)$ in disjunctive normal form (DNF)

$$\varphi \wedge (a \vee b) \equiv_P (\varphi \wedge a) \vee (\varphi \wedge b)$$

and instead of a single transition, have two transitions

$$\varphi \xrightarrow{\emptyset} \varphi \wedge a \quad \text{and} \quad \varphi \xrightarrow{\emptyset} \varphi \wedge b.$$

In other words, the nondeterminism is used to guess which of the two disjuncts of the DNF is going to hold. Formally, we proceed as follows:

Definition 7.1. We define $dnf(\varphi)$ as the set of clauses obtained by putting the propositional formula φ in DNF, i.e., $\varphi \equiv_P \bigvee_{\psi\in dnf(\varphi)} \psi$. Further let

$$Reach^\vee(\varphi) = \bigcup_{w\in(2^{AP})^*} af^\vee(\psi, w)$$

with $af^\vee(\psi, \epsilon) = dnf(\psi)$, $af^\vee(\psi, \nu) = dnf(af(\psi, \nu))$, and $af^\vee(\psi, \nu w) = \bigcup_{\psi'\in af^\vee(\psi,\nu)} af^\vee(\psi', w)$ for every formula ψ, letter ν, and word w.

Notice that $dnf(\mathbf{ff}) = \emptyset$ and $dnf(\mathbf{tt}) = \{\mathbf{tt}\}$. Since the automata defined below have sets of states of the form $Reach^\vee(\varphi)$, they have a state labeled by $\mathbf{tt}$, but no state labeled by $\mathbf{ff}$.

The proof of the next proposition follows immediately from the definitions.

Proposition 7.2. *Let $\varphi \in \mu LTL$.*

- *The following NBA over the alphabet 2^{AP} recognizes $L(\varphi)$:*

$$\mathcal{A}^\varphi_\mu = (\, Reach^\vee(\varphi), af^\vee, dnf(\varphi), inf(\mathbf{tt})\,)$$

- *The following NBA over the alphabet 2^{AP} recognizes $L(\mathbf{GF}\varphi)$:*

$$\mathcal{A}^\varphi_{\mathbf{GF}\mu} = (\, Reach^\vee(\mathbf{F}\varphi), af^\vee_{\mathbf{F}\varphi}, \{\mathbf{F}\varphi\}, inf(\mathbf{tt})\,)$$

$$af^\vee_{\mathbf{F}\varphi}(\psi, \nu) = \begin{cases} \{\mathbf{F}\varphi\} & \text{if } \psi \equiv_P \mathbf{tt} \\ af^\vee(\psi, \nu) & \text{otherwise.} \end{cases}$$

Let $\varphi \in \nu LTL$.

- *The following NBA over the alphabet 2^{AP} recognizes $L(\varphi)$:*

$$\mathcal{A}^\varphi_\nu = (\, Reach^\vee(\varphi), af^\vee, dnf(\varphi), inf(Reach^\vee(\varphi))\,)$$

- *The following NBA over the alphabet 2^{AP} recognizes $L(\mathbf{FG}\varphi)$:*

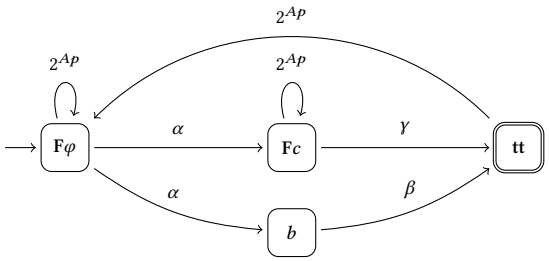
$$\mathcal{A}^\varphi_{\mathbf{FG}\nu} = (\, Reach^\vee(\mathbf{G}\varphi) \cup \{\mathbf{FG}\varphi\}, af^\vee_{\mathbf{G}\varphi}, \{\mathbf{FG}\varphi\}, inf(Reach^\vee(\mathbf{G}\varphi))\,)$$

$$af^\vee_{\mathbf{G}\varphi}(\psi, \nu) = \begin{cases} \{\mathbf{FG}\varphi, \mathbf{G}\varphi\} & \text{if } \psi = \mathbf{FG}\varphi \\ af^\vee(\psi, \nu) & \text{otherwise.} \end{cases}$$

Recall that the elements of $Reach(\varphi)$ are positive boolean combinations of proper subformulas of φ. It follows that the elements of $Reach^\vee(\varphi)$ are *conjunctions* of proper subformulas of φ. Since the number of proper subformulas is bounded by the length of the formula, we immediately obtain:

Proposition 7.3. *If φ has n proper subformulas, then $Reach^\vee(\varphi)$ has at most 2^n elements, and so all the NBAs of Proposition 7.2 have at most $2^{n+1} + 1 = O(2^n)$ states.*

Example 7.4. Let $\varphi = a \wedge \mathbf{X}(b \vee \mathbf{F}c)$, the formula for which a DBA was given in Example 4.3. The NBA $\mathcal{A}^{\varphi}_{\mathbf{GF}\mu}$ is shown below. The figure uses the abbreviations of Example 4.3.



Compared to the DBA of Example 4.3, the NBA has a simpler structure, although in this case the same number of states. △

To define NBAs for arbitrary formulas we apply the Master Theorem. This is routine, and so we only sketch the constructions.

A NBA for $L^1_{X,Y}$**.** We define a NBA $C_{\varphi,X}$ that accepts a word w iff $w_i \models \mathit{af}(\varphi, w_{0i})[X]_\nu$ for some suffix w_i of w. Recall that $\mathit{af}(\varphi, w_{0i})[X]_\nu \in \nu LTL$ for every $i \geq 0$. The automaton consists of two components with sets of states Q_1 and Q_2 given by

$$Q_1 = \{(\psi, 1) \mid \psi \in \mathit{Reach}^{\vee}(\varphi)\} \quad Q_2 = \{(\psi[X]_\nu, 2) \mid \psi \in \mathit{Reach}^{\vee}(\varphi)\}$$

Transitions either stay in the same component, or "jump" from the first component to the second. Transitions that stay in the same component are of the form $(\psi, i) \xrightarrow{\nu} (\psi', i)$ for $\psi' \in \mathit{af}^{\vee}(\psi, \nu)$ and $i = 1, 2$. "Jumps" are transitions of the form $(\psi, 1) \xrightarrow{\epsilon} (\psi[X]_\nu, 2)$. Jumping amounts to nondeterministically guessing the suffix w_i satisfying $\mathit{af}(\varphi, w_{0i})[X]_\nu$. The accepting condition is $\mathit{inf}(Q_2)$. Notice that the state $(\mathbf{ff}, 2)$ does not have any successors.
Since $\mathit{Reach}^{\vee}(\varphi)$ has at most 2^n states, $C_{\varphi,X}$ has $2^{O(n)}$ states.

A NBA for $L^2_{X,Y}$**.** As in the case of DRAs, we define a NBA recognizing $L\left(\bigwedge_{\psi \in X} \mathbf{GF}(\psi[Y]_\mu)\right)$. To obtain an NBA with $2^{O(n)}$ states we use a well-known trick. Given a set $\{\psi_1, \ldots, \psi_k\}$ of formulas, we have

$$\bigwedge_{i=1}^{k} \mathbf{GF}\psi_i \equiv \mathbf{GF}(\psi_1 \wedge \mathbf{F}(\psi_2 \wedge \mathbf{F}(\psi_3 \wedge \ldots \wedge \mathbf{F}(\psi_{k-1} \wedge \mathbf{F}\psi_k)\ldots))$$

The formula obtained after applying the trick belongs to $\mathbf{GF}(\mu LTL)$ and has $O(n)$ μ-subformulas. By Proposition 7.2.2 we can construct a NBA for it with $2^{O(n)}$ states.

A NBA for $L^3_{X,Y}$**.** In this case we apply

$$\bigwedge_{i=1}^{k} \mathbf{FG}\psi_i \equiv \mathbf{FG}\left(\bigwedge_{i=1}^{k} \psi_i\right)$$

and Proposition 7.2.4, yielding an automaton with $2^{O(n)}$ states.

A NBA for $L(\varphi)$**.** We proceed as in the case of DRAs, using the well-known operations for union and intersection of NBAs. The NBA A_φ is the union of at most 2^n NBAs $A_{X,Y}$, each of them with $2^{O(n)}$ states. The difference with the DRA case is that, given NBAs with $n_1, \ldots, n_k$ states accepting languages $L_1, \ldots, L_k$, we can construct a NBA for $\bigcup_{i=1}^{k} L_i$ with $\sum_{i=1}^{k} n_i$ states, instead of $\prod_{i=1}^{k} n_i$ states, as was the case for DRAs. So A_φ has $2^n \cdot 2^{O(n)} = 2^{O(n)}$ states.

8 Constructing LDBAs for LTL Formulas

The translation of LTL into LDBA combines the translations into DRA and NBA. Recall that the states of an LDBA are partitioned into an initial component and a deterministic accepting component containing all accepting states. While in the definition of a LDBA the initial component can be nondeterministic, in our construction we can easily make it deterministic: Every accepting run has exactly one non-deterministic step. This makes the LDBA usable for quantitative (and not only qualitative) probabilistic model checking, as described in [35].

Lemma 6.1 shows that checking property (1) of Theorem 5.12 can be arbitrarily delayed, which allows us to slightly rephrase the Master Theorem as follows:

Theorem 8.1. *(Variant of the Master Theorem) For every formula φ and for every word w: $w \models \varphi$ iff there exists $X \subseteq \mu(\varphi)$, $Y \subseteq \nu(\varphi)$, and $i \geq 0$ satisfying*

$$\begin{array}{ll} (1') & w_i \models \mathit{af}(\varphi, w_{0i})[X]_\nu \\ (2') & \forall \psi \in X.\ w_i \models \mathbf{GF}(\psi[Y]_\mu) \\ (3') & \forall \psi \in Y.\ w_i \models \mathbf{G}(\psi[X]_\nu) \end{array}$$

Proof. Clearly, the existence of an index i satisfying (1'-3') implies that conditions (1-3) hold. For the other direction, assume conditions (1-3) hold. By Lemma 6.1 the index i of condition (1) can be chosen arbitrarily large. Since $w \models \bigwedge_{\psi \in X} \mathbf{FG}(\psi[X]_\nu)$, we can choose i so that it also satisfies $w_i \models \bigwedge_{\psi \in X} \mathbf{G}(\psi[X]_\nu)$. □

The idea of the construction is to use the initial component to keep track of $\mathit{af}(\varphi, w_{0i})$—that is, after reading a finite word w_{0i} the initial component is in state $\mathit{af}(\varphi, w_{0i})$—and use the jump to the accepting component to guess sets X and Y and the stabilization point i. The jump leads to the initial state of the intersection of three DBAs, which are in charge of checking (1'), (2'), and (3').

Recall that $\mathit{af}(\varphi, w_{0i}) \in \mathit{Reach}(\varphi)$ for every word w and every $i \geq 0$. For every $\psi \in \mathit{Reach}(\varphi)$ and for each pair of sets X, Y we construct a DBA $\mathcal{D}_{\psi,X,Y}$ recognizing the intersection of the languages of the formulas:

$$\psi[X]_\nu \qquad \bigwedge_{\psi \in X} \mathbf{GF}(\psi[Y]_\mu) \qquad \bigwedge_{\psi \in Y} \mathbf{G}(\psi[X]_\nu)$$

These formulas belong to νLTL, $\mathbf{GF}(\mu LTL)$, and νLTL, respectively, and so we can obtain DBAs for them following the recipes of Proposition 4.2. As argued before, each of these DBAs have $2^{2^{O(n)}}$ states, and so we can also construct a DBA for their intersection with the same upper bound. Summarizing, we obtain:

Initial component. The component is $(\mathit{Reach}(\varphi), \mathit{af}, \{\varphi\})$ and thus the component has at most 2^{2^n} states. Recall that this component does not have accepting states.

Accepting component. The component is the disjoint union, for every $\psi \in \mathit{Reach}(\varphi)$, $X \subseteq \mu(\varphi)$, and $Y \subseteq \nu(\varphi)$, of the DBA $\mathcal{D}_{\psi,X,Y}$. Since $\mathit{Reach}(\varphi)$ has at most 2^{2^n} formulas and there are at most 2^n pairs (X, Y), the component is the disjoint union of at most $2^{2^n} \cdot 2^n$ automata, each of then with $2^{2^{O(n)}}$ states. Thus in total $2^{2^{O(n)}}$ states.

A LDBA for $L(\varphi)$**.** The LDBA is the disjoint union of the initial and accepting components. The initial component is connected to the accepting component by ϵ-transitions: For every formula $\psi \in \mathit{Reach}(\varphi)$ and for every two sets X, Y, there is an ϵ-transition from state ψ of the initial component to the initial state of $\mathcal{D}_{\psi,X,Y}$.

The LDBA has $2^{2^{O(n)}} + 2^{2^n} = 2^{2^{O(n)}}$ states. Recall that the lower bound for the blowup of a translation of LTL to LDBA is also doubly exponential (see e.g. [35]).

9 Discussion

This paper builds upon our own work [13, 15, 19, 26, 27, 35]. In particular, the notion of *stabilization point* of a word with respect to a formula, and the idea of using oracle information that is subsequently checked are already present there. The translations of LTL to LDBAs of [23, 24] are based on similar ideas, also with resemblance to obligation sets of [29, 30].

The essential novelty of this paper with respect to the previous work is the introduction of the symmetric mappings $\cdot[\cdot]_\mu$ and $\cdot[\cdot]_\nu$. Applying them to an arbitrary formula φ yields a simpler formula, but *not in the sense one might expect*. In particular, $\varphi[Y]_\mu$ may be *stronger* than φ. For example, the information that, say, the formula $a\mathbf{W}b$ does not hold infinitely often makes us check the *stronger* formula $a\mathbf{U}b = (a\mathbf{W}b)[\emptyset]_\mu$. However, exactly this point makes the "μ-ν-alternation" work: The formulas $\varphi[X]_\nu$ and $\varphi[Y]_\mu$ are only simpler in the sense of *easier to translate*. This is the reason why operators **W** and **M** are present in the core syntax and the missing piece since the symmetric solutions [26, 27], limited to fragments based on the simpler operators **F** and **G**.

The Master Theorem can be applied beyond what is described in this paper. In order to translate LTL into universal automata we only need to normalize formulas into conjunctive normal form. Furthermore one can obtain a double exponential translation into deterministic parity automata adapting the approach described in [14]. Another intriguing question is whether our translation into NBA, which is very different from the ones described in the literature is of advantage in some application like runtime verification.

The target automata classes used in practice typically use an acceptance condition defined on transitions, instead of states. Further, they use *generalized* acceptance conditions, be it Büchi or Rabin. All our constructions can be restated effortlessly to yield automata with transition-based acceptance, and if generalized acceptance conditions are allowed then they become simpler and more succinct. The implementation used in our experiments actually uses these two features, which is described in the appendix of [16].

To conclude, in our opinion this paper successfully finishes the journey started in [26]. Via a single theorem it provides an arguably elegant (unified, symmetric, syntax-independent, not overly complex) and efficient (asymptotically optimal and practically relevant) translation of LTL into your favourite ω-automata.

Acknowledgments. The authors want to thank Alexandre Duret-Lutz, Benedikt Seidl, the anonymous reviewers, and the participants of the 2nd Jerusalem Winter School in Computer Science and Engineering on "Formal Verification" for their helpful comments and remarks.

References

[1] Valentin M. Antimirov. 1996. Partial Derivatives of Regular Expressions and Finite Automaton Constructions. *Theor. Comput. Sci.* 155, 2 (1996), 291–319.
[2] Tomáš Babiak, František Blahoudek, Alexandre Duret-Lutz, Joachim Klein, Jan Křetínský, David Müller, David Parker, and Jan Strejček. 2015. The Hanoi Omega-Automata Format. In *CAV*. 479–486.
[3] Tomás Babiak, Frantisek Blahoudek, Mojmír Křetínský, and Jan Strejcek. 2013. Effective Translation of LTL to Deterministic Rabin Automata: Beyond the (F, G)-Fragment. In *ATVA*. 24–39.
[4] Tomás Babiak, Mojmír Křetínský, Vojtech Rehák, and Jan Strejcek. 2012. LTL to Büchi Automata Translation: Fast and More Deterministic. In *TACAS*. 95–109.
[5] Christel Baier and Joost-Pieter Katoen. 2008. *Principles of Model Checking*. MIT Press.
[6] Christel Baier, Stefan Kiefer, Joachim Klein, Sascha Klüppelholz, David Müller, and James Worrell. 2016. Markov Chains and Unambiguous Büchi Automata. In *CAV*. 23–42.
[7] Janusz A. Brzozowski. 1964. Derivatives of Regular Expressions. *J. ACM* 11, 4 (1964), 481–494. https://doi.org/10.1145/321239.321249
[8] Krishnendu Chatterjee, Andreas Gaiser, and Jan Křetínský. 2013. Automata with Generalized Rabin Pairs for Probabilistic Model Checking and LTL Synthesis. In *CAV*. 559–575.
[9] Costas Courcoubetis and Mihalis Yannakakis. 1995. The Complexity of Probabilistic Verification. *J. ACM* 42, 4 (1995), 857–907.
[10] Jean-Michel Couvreur. 1999. On-the-Fly Verification of Linear Temporal Logic. In *FM*. 253–271.
[11] Marco Daniele, Fausto Giunchiglia, and Moshe Y. Vardi. 1999. Improved Automata Generation for Linear Temporal Logic. In *CAV*. 249–260.
[12] Alexandre Duret-Lutz, Alexandre Lewkowicz, Amaury Fauchille, Thibaud Michaud, Etienne Renault, and Laurent Xu. 2016. Spot 2.0 - A Framework for LTL and ω-Automata Manipulation. In *ATVA*. 122–129.
[13] Javier Esparza and Jan Křetínský. 2014. From LTL to Deterministic Automata: A Safraless Compositional Approach. In *CAV*. 192–208.
[14] Javier Esparza, Jan Křetínský, Jean-François Raskin, and Salomon Sickert. 2017. From LTL and Limit-Deterministic Büchi Automata to Deterministic Parity Automata. In *TACAS*. 426–442.
[15] Javier Esparza, Jan Křetínský, and Salomon Sickert. 2016. From LTL to deterministic automata - A safraless compositional approach. *Formal Methods in System Design* 49, 3 (2016), 219–271.
[16] Javier Esparza, Jan Křetínský, and Salomon Sickert. 2018. One Theorem to Rule Them All: A Unified Translation of LTL into ω-Automata. *CoRR* abs/1805.00748 (2018). arXiv:1805.00748 http://arxiv.org/abs/1805.00748
[17] Kousha Etessami and Gerard J. Holzmann. 2000. Optimizing Büchi Automata. In *CONCUR*. 153–167.
[18] Carsten Fritz. 2003. Constructing Büchi Automata from Linear Temporal Logic Using Simulation Relations for Alternating Büchi Automata. In *CIAA*. 35–48.
[19] Andreas Gaiser, Jan Křetínský, and Javier Esparza. 2012. Rabinizer: Small Deterministic Automata for LTL(F,G). In *ATVA*. 72–76.
[20] Paul Gastin and Denis Oddoux. 2001. Fast LTL to Büchi Automata Translation. In *CAV*. 53–65.
[21] Dimitra Giannakopoulou and Flavio Lerda. 2002. From States to Transitions: Improving Translation of LTL Formulae to Büchi Automata. In *FORTE*. 308–326.
[22] Ernst Moritz Hahn, Guangyuan Li, Sven Schewe, Andrea Turrini, and Lijun Zhang. 2015. Lazy Probabilistic Model Checking without Determinisation. In *CONCUR*. 354–367.
[23] Dileep Kini and Mahesh Viswanathan. 2015. Limit Deterministic and Probabilistic Automata for LTL \ GU. In *TACAS*. 628–642.
[24] Dileep Kini and Mahesh Viswanathan. 2017. Optimal Translation of LTL to Limit Deterministic Automata. In *TACAS*. 113–129.
[25] Zuzana Komárková and Jan Křetínský. 2014. Rabinizer 3: Safraless Translation of LTL to Small Deterministic Automata. In *ATVA (LNCS)*, Vol. 8837. 235–241.
[26] Jan Křetínský and Javier Esparza. 2012. Deterministic Automata for the (F,G)-Fragment of LTL. In *CAV*. 7–22.
[27] Jan Křetínský and Ruslán Ledesma-Garza. 2013. Rabinizer 2: Small Deterministic Automata for LTL \ GU. In *ATVA*. 446–450.
[28] Jan Křetínský, Tobias Meggendorfer, Clara Waldmann, and Maximilian Weininger. 2017. Index Appearance Record for Transforming Rabin Automata into Parity Automata. In *TACAS*. 443–460.
[29] Jianwen Li, Geguang Pu, Lijun Zhang, Zheng Wang, Jifeng He, and Kim Guldstrand Larsen. 2013. On the Relationship between LTL Normal Forms and Büchi Automata. In *Theories of Programming and Formal Methods - Essays Dedicated to Jifeng He on the Occasion of His 70th Birthday*. 256–270.
[30] Jianwen Li, Lijun Zhang, Shufang Zhu, Geguang Pu, Moshe Y. Vardi, and Jifeng He. 2018. An explicit transition system construction approach to LTL satisfiability checking. *Formal Asp. Comput.* 30, 2 (2018), 193–217.
[31] Nir Piterman. 2006. From Nondeterministic Buchi and Streett Automata to Deterministic Parity Automata. In *LICS*. 255–264.
[32] Amir Pnueli. 1977. The Temporal Logic of Programs. In *FOCS*. 46–57.
[33] Shmuel Safra. 1988. On the Complexity of omega-Automata. In *FOCS*. 319–327.
[34] Sven Schewe. 2009. Tighter Bounds for the Determinisation of Büchi Automata. In *FoSSaCS*. 167–181.
[35] Salomon Sickert, Javier Esparza, Stefan Jaax, and Jan Křetínský. 2016. Limit-Deterministic Büchi Automata for Linear Temporal Logic. In *CAV*. 312–332.
[36] Fabio Somenzi and Roderick Bloem. 2000. Efficient Büchi Automata from LTL Formulae. In *CAV*. 248–263.
[37] Moshe Y. Vardi. 1985. Automatic Verification of Probabilistic Concurrent Finite-State Programs. In *FOCS*. 327–338.
[38] Moshe Y. Vardi and Pierre Wolper. 1986. An Automata-Theoretic Approach to Automatic Program Verification (Preliminary Report). In *LICS*. 332–344.