Noname manuscript No.
(will be inserted by the editor)

Visibly Linear Temporal Logic

Laura Bozzelli · César Sánchez

Received: date / Accepted: date

Abstract We introduce *Visibly Linear Temporal Logic* (VLTL), a linear-time temporal logic that captures the full class of Visibly Pushdown Languages over infinite words. The novel logic avoids fix points and instead provides natural temporal operators with simple and intuitive semantics. We prove that the complexities of the satisfiability and visibly pushdown model checking problems are the same as for other well known logics, like CaRet and the nested word temporal logic NWTL, which in contrast are strictly more limited in expressive power than VLTL. Moreover, formulas of CaRet and NWTL can be translated inductively and in linear-time into VLTL.

1 Introduction

Input-driven languages introduced by Mehlhorn [24], and later reintroduced and thoroughly investigated by Alur et al. [4,5] under the name of *Visibly Pushdown Languages* (VPL), are a subclass of context-free languages that is similar in tractability and robustness to the less expressive class of regular languages. VPL are closed under all Boolean operations, and problems such as universality and inclusion that are undecidable for context-free languages are EXPTIME-complete for VPL [4,5]. A VPL consists of *nested words*, that is, words over an alphabet (pushdown alphabet) which is partitioned into three

A preliminary version of this paper appears in the Proceedings of the 7th International Joint Conference on Automated Reasoning (IJCAR'14), Vienna, Austria, July 19-22, 2014.

Laura Bozzelli
Technical University of Madrid (UPM), Madrid, Spain
E-mail: laura.bozzelli@fi.upm.es

César Sánchez
IMDEA Software Institute, Madrid, Spain &
Institute for Applied Physics, CSIC, Madrid, Spain
E-mail: cesar.sanchez@imdea.org

disjoint sets of calls, returns and internal symbols. This partition induces a nested hierarchical structure in a given word obtained by associating to each call the corresponding matching return (if any) in a well-nested manner. VPL are accepted by Non-deterministic Visibly Pushdown Automata (NVPA) [4,5], a subclass of pushdown automata where the input symbol controls the kind of operations permissible on the stack. Alternative characterizations of VPL have been given in terms of operational and declarative formalisms. Here, we recall characterizations [6,10] based on alternating automata, like the class of parity alternating visibly pushdown automata and the more tractable class of parity two-way *alternating finite-state jump automata* (AJA) [10], which extend standard alternating finite-state automata (AFA) with non-local moves for navigating the nested structure of words in VPL.

VPL have applications in the formal verification of recursive programs with finite data modeled by pushdown systems [8,5,3]. Runs in these programs can be seen as nested words, where procedure calls and returns define the nesting. VPL turn out to be useful also in the streaming processing of semi-structured data, such as XML documents, where each open-tag is matched with a closing-tag in a well-nested manner (see e.g. [23,1]). The theory of VPL is connected to the theory of regular *tree*-languages since nested words can be encoded by labeled binary trees satisfying some regular constraints, and there are translations from VPL into regular tree languages over tree-encodings of nested words, and vice-versa. However, as shown in [23,1], NVPA are often more natural (and sometimes exponentially more succinct) than tree automata, and preferable in the streaming processing of XML documents.

Linear Temporal logics for VPL-properties. CaRet [3] and its extension NWTL$^+$ [2] are context-free versions of standard linear temporal logic LTL [27]. It is well-known [33] that LTL does not allow to specify all the linear-time ω-regular properties. Similarly, the logics CaRet and NWTL$^+$ can only express a strict subclass of VPL. There are logical frameworks which capture the full class of VPL. For example, [4] introduce MSO$_\mu$, which extends the standard monadic second order logic (MSO) with a binary matching-predicate to handle nested words. One drawback of this approach is that MSO$_\mu$ is not elementarily decidable. Also, [10] introduces a fix-point calculus for which satisfiability and visibly pushdown model checking are EXPTIME-complete. Additionally, fix-point logics are in some sense low-level logics, which are considered to be "unfriendly" as specification languages. In the setting of regular languages, some tractable formalisms allow to avoid fix-point binders and still obtain full expressivity, like ETL [31] and fragments of the industrial-strength logic PSL [17], like dynamic linear time temporal logic [16] and the regular linear temporal logic RLTL [21,28]. RLTL, in particular, fuses regular expressions and LTL modalities. In the linear-time setting, providing both regular expressions and temporal operators as been motivated by the need for human-readable specification languages,[1] as witnessed by the widespread adoption of

[1] From [14,7] : "PSL and SVA balance well [...] expressiveness, usability, and implementability."

ForSpec, PSL and SVA in industry (see e.g. [7]). Our work follows this direction for visibly-pushdown languages. We recently introduced [11,12] an algebraic characterization of VPL over finite nested words in terms of the class of *visibly rational expressions* (VRE). VRE extend regular expressions with two novel operators which capture in a natural way the nested relation between calls and matching returns in nested words. These two operators, when applied to languages $\mathcal{L}$ of *well-matched* words—that is, nested words without pending calls and pending returns—, correspond to classical tree substitution and Kleene closure applied to the tree language encoding of $\mathcal{L}$ (in accordance with the encoding of well-matched words by ordered unranked finite trees [1]). However, as observed in [1] when comparing *well-matched* words with ordered unranked trees, "word operations such as prefixes, suffixes, and concatenation [...] do not have analogous tree operations." This expressive ability is explicit in VRE, which provides both word-like concatenation and tree-like substitution (and their Kleene closures), which allows VRE to describe both the linear structure and the hierarchical structure of nested words.

Our contribution. We investigate a new linear temporal logic for VPL specifications, which merges in a convenient way VRE and LTL modalities. The task of combining language operators (such as concatenation and Kleene closure) and logical modalities is in general not easy. For example, extending regular expressions by allowing unrestricted complementation (logical negation) results in a formalism which is still decidable but with a non-elementary computational complexity [30].

Thus, we propose a generalization of RLTL with past that we call *Visibly Linear Temporal Logic* (VLTL), which is obtained by replacing regular expressions for VRE expressions as building blocks for the temporal modalities. Our natural choice leads to a unifying and convenient logical framework for specifying VPL-properties because:

- VLTL is closed under Boolean combinations including negation and captures the full class of VPL. Moreover, VLTL avoids fix points and only offers temporal operators with simple and intuitive semantics.
- VLTL is elementarily decidable. In particular, satisfiability and visibly pushdown model checking have the same complexity as for the strictly less expressive logics CaRet and NWTL$^+$. Namely, these decision problems are EXPTIME-complete.

Another feature of VLTL is that CaRet and NWTL$^+$ can be inductively translated in linear-time into VLTL, so that algorithms and results for VLTL are readily applicable to CaRet and NWTL$^+$. In particular, the temporal modalities of CaRet and NWTL$^+$ can be viewed as derived operators of VLTL. In this manner, one can introduce additional user-friendly temporal modalities as VLTL expressions, which can then be combined arbitrarily. Thus, VLTL can be also used as a common unifying logical framework for obtaining efficient decision procedures for other simple-to-use logics for VPL (where efficient here means with optimal complexity).

In order to tackle the decision problems for VLTL, we follow an automata theoretic approach, extending recent results for future RLTL [29]. These results introduce a translation of RLTL into parity AFA, which is crucially based on the well-known polynomial-time translation of regular expressions into *nondeterministic* finite-state automata. A direct generalization of this construction to parity alternating visibly pushdown automata would lead to *doubly* exponential time decision procedures. Instead, the approach we present in this paper is based on a compositional polynomial-time translation of VLTL formulas into a subclass of parity two-way alternating AJA with index 2. We call this class *stratified AJA with main states* (SAJA). Essentially, SAJA are the two-way AJA version of one-way hesitant AFA over infinite words introduced in [20], where the ability to combine both forward and backward moves is syntactically restricted in a way that guarantees that every infinite path in a run has a suffix which is fully forward (that is, eventually, every copy of the automaton only moves, at each step, to future input positions). Moreover, we identify a subclass of VRE such that the corresponding fragment of VLTL has the same expressiveness as full VLTL and admits a translation into SAJA whose number of states is *linear* in the size of the specification. Hence, we obtain a translation for this fragment of VLTL into equivalent Büchi NVPA of size $2^{O(|\varphi| \log m)}$, where m is the size of the largest VRE used in the given VLTL formula φ.

Related work. The branching temporal logic PDL [15] combines modal logic and regular expressions, and is extended to recursive programs in [22]. In this extension low-level operational aspects are allowed in the form of path expressions given by NVPA. This logic is incomparable with VLTL. Furthermore, the related satisfiability and visibly pushdown model-checking problems are 2-EXPTIME-complete. A similar logic is studied in [9] which introduces a linear temporal framework for VPL, by allowing PDL-like path regular expressions extended with the binary matching-predicate μ of MSO_μ. The setting is parameterized by a finite set of MSO-definable temporal modalities, which leads to an infinite family of linear temporal logics having the same complexity as VLTL and also subsuming the logics CaRet and NWTL^+. However, it seems clear (even if this issue is not discussed in [9]) that each of these logics does not capture the full class of VPL. Moreover, the complexity analysis in [9], based on the use of two-way alternating tree automata, is not fine-grained and it just allows to obtain a generic polynomial in the exponent of the complexity upper bound.

2 Preliminaries

In this section, we recall two different characterizations of the class of Visibly Pushdown Languages (VPL, for short) [4], namely, Visibly Pushdown Automata [4] (representing the basic formalism for VPL) and Visibly Rational Expressions [11,12].

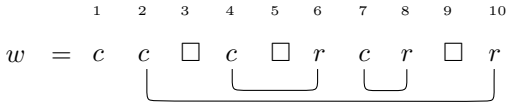
In the rest of the paper, we fix a *pushdown alphabet* $\Sigma = \Sigma_{call} \cup \Sigma_{ret} \cup \Sigma_{int}$, that is, a finite alphabet Σ which is partitioned into a set Σ_{call} of *calls*, a set Σ_{ret} of *returns*, and a set Σ_{int} of *internal actions*.

Let $\mathbb{N}$ be the set of natural numbers. For a finite or infinite word w on Σ, $|w|$ is the length of w (we set $|w| = \omega$ if w is infinite). The set of positions $Pos(w)$ in a non-empty word w is $\{1 \dots |w|\}$ if w is finite and $\mathbb{N} \setminus \{0\}$ if w is infinite. The empty word is denoted by ε and its set $Pos(\varepsilon)$ of positions is empty. For all $i \in Pos(w)$, $w(i)$ is the i^{th} symbol of w, and for all $j \in Pos(w)$ with $j \geq i$, $w[i, j]$ is the non-empty finite word $w(i)w(i+1) \dots w(j)$. A *pointed word* over Σ is a pair (w, i) consisting of a word w over Σ and a position $i \in Pos(w)$.

Matched calls and returns. The set $WM(\Sigma)$ of *well-matched words* is the subset of Σ^* inductively defined as follows:
- $\varepsilon \in WM(\Sigma)$;
- $\Box \cdot w \in WM(\Sigma)$ if $\Box \in \Sigma_{int}$ and $w \in WM(\Sigma)$;
- $c \cdot w \cdot r \cdot w' \in WM(\Sigma)$ if $c \in \Sigma_{call}$, $r \in \Sigma_{ret}$, and $w, w' \in WM(\Sigma)$.

Let i be a call position of a word w. If there is $j > i$ such that j is a return position of w and $w(i+1) \dots w(j-1)$ is a well-matched word, we say that j is the *matching return* of i along w. Note that j is uniquely determined if it exists. The set $MWM(\Sigma)$ of *minimally well-matched words* is the set of well-matched words of the form $c \cdot w \cdot r$ such that c is a call, r is a return, and w is well-matched. Note that r corresponds to the matching return of c. For a language $\mathcal{L} \subseteq \Sigma^*$, we define $MWM(\mathcal{L}) \stackrel{\text{def}}{=} \mathcal{L} \cap MWM(\Sigma)$, that is, the set of words in $\mathcal{L}$ which are minimally well-matched.

Example 1 Let $\Sigma_{call} = \{c\}$, $\Sigma_{ret} = \{r\}$, and $\Sigma_{int} = \{\Box\}$. Consider the word w below. The word w is not well-matched, in particular, the call at position 1 has no matching return in w. Moreover, note that the sub-word $w(2) \dots w(10)$ is minimally well-matched.



2.1 Visibly Pushdown Automata

Non-deterministic Visibly Pushdown Automata (NVPA) [4] are standard Pushdown Automata operating on finite words over a *pushdown alphabet* Σ satisfying the following "visibly" restriction: (i) on reading a call, one symbol is pushed onto the stack, (ii) on reading a return, one symbol is popped from the stack (if the stack is empty, the stack content remains unchanged), and (iii) on reading an internal action, no stack operation is performed. The languages of finite words accepted by NVPA are called *visibly pushdown languages* (VPL). We also consider *Büchi ω-NVPA* [4], which are standard Büchi Pushdown Automata on infinite words over Σ satisfying the above visibly restriction. The

ω-languages accepted by Büchi NVPA are called *ω-visibly pushdown languages* (ω-VPL). We now proceed with the formal definition of the syntax and semantics of NVPA and Büchi ω-NVPA.

Syntax and semantics of NVPA. An NVPA over the pushdown alphabet $\Sigma = \Sigma_{call} \cup \Sigma_{ret} \cup \Sigma_{int}$ is a tuple $\mathcal{P} = \langle Q, Q_0, \Gamma, \Delta, F \rangle$, where Q is a finite set of (control) states, $Q_0 \subseteq Q$ is a set of initial states, Γ is a finite stack alphabet, $\Delta \subseteq (Q \times \Sigma_{call} \times Q \times \Gamma) \cup (Q \times \Sigma_{ret} \times (\Gamma \cup \{\bot\}) \times Q) \cup (Q \times \Sigma_{int} \times Q)$ is a transition relation (where $\bot \notin \Gamma$ is the special *stack bottom symbol*), and $F \subseteq Q$ is a set of accepting states. On reading a call $c \in \Sigma_{call}$, $\mathcal{P}$ chooses a push transition of the form (q, c, q', γ), pushes the symbol $\gamma \neq \bot$ onto the stack, and changes the control from q to q'. On reading a return $r \in \Sigma_{ret}$, $\mathcal{P}$ chooses a pop transition of the form (q, r, γ, q'), where γ is popped from the stack (if $\gamma = \bot$, then γ is read but not popped). Finally, on reading an internal action $\Box \in \Sigma_{int}$, $\mathcal{P}$ can choose only transitions of the form $(q, \Box, q')$ which do not use the stack.

A configuration of $\mathcal{P}$ is a pair (q, β), where $q \in Q$ and $\beta \in \Gamma^* \cdot \{\bot\}$ is a stack content. A run π of $\mathcal{P}$ over a finite word $\sigma_1 \dots \sigma_{n-1} \in \Sigma^*$ is a finite sequence of configurations of the form $\pi = (q_1, \beta_1) \xrightarrow{\sigma_1} (q_2, \beta_2) \dots \xrightarrow{\sigma_{n-1}} (q_n, \beta_n)$ such that $q_1 \in Q_0$, $\beta_1 = \bot$ (initialization requirement), and the following holds for all $1 \leq i \leq n-1$:

Push If σ_i is a call, then for some $\gamma \in \Gamma$, $(q_i, \sigma_i, q_{i+1}, \gamma) \in \Delta$ and $\beta_{i+1} = \gamma \cdot \beta_i$.
Pop If σ_i is a return, then for some $\gamma \in \Gamma \cup \{\bot\}$, $(q_i, \sigma_i, \gamma, q_{i+1}) \in \Delta$, and *either* $\gamma \neq \bot$ and $\beta_i = \gamma \cdot \beta_{i+1}$, *or* $\gamma = \bot$ and $\beta_i = \beta_{i+1} = \bot$.
Internal If σ_i is an internal action, then $(q_i, \sigma_i, q_{i+1}) \in \Delta$ and $\beta_{i+1} = \beta_i$.

The run π is *accepting* whenever the last state is accepting, that is, if $q_n \in F$. The language $\mathcal{L}(\mathcal{P})$ of $\mathcal{P}$ is the set of finite words $w \in \Sigma^*$ such that there is an accepting run of $\mathcal{P}$ on w.

Syntax and semantics of Büchi ω-NVPA. Büchi ω-NVPA are syntactically defined as NVPA. Fix a Büchi ω-NVPA $\mathcal{P} = \langle Q, Q_0, \Gamma, \Delta, F \rangle$ over the pushdown alphabet Σ. A run π over an infinite word $\sigma_1 \sigma_2 \dots \in \Sigma^\omega$ is an infinite sequence $\pi = (q_1, \beta_1) \xrightarrow{\sigma_1} (q_2, \beta_2) \dots$ that is defined using the natural extension of the definition of runs on finite words. The run is *accepting* if for infinitely many i in the sequence, $q_i \in F$. The ω-language $\mathcal{L}_\omega(\mathcal{P})$ of $\mathcal{P}$ is the set of infinite words $w \in \Sigma^\omega$ such that there is an accepting run of $\mathcal{P}$ on w.

2.2 Visibly Rational Expressions (VRE)

We recall the classes of VRE and ω-VRE [11,12]. VRE extend regular expressions (RE) with two non-regular operators: the binary M-substitution operator and the unary S-closure operator.[2] Given $\mathcal{L} \subseteq \Sigma^*$ and a language $\mathcal{L}'$ of finite

[2] The origin of the name M-substitution is *minimally well-matched substitution*, while S-closure stands for *Strict Minimally Well-Matched Closure*, see [11,12].

or infinite words on Σ, we use $\mathcal{L} \cdot \mathcal{L}'$ for the concatenation of $\mathcal{L}$ and $\mathcal{L}'$, $\mathcal{L}^*$ for the Kleene closure of $\mathcal{L}$, and $\mathcal{L}^\omega$ for the ω-Kleene closure of $\mathcal{L}$. Recall that $\mathcal{L}^\omega$ is the set of infinite words w of the form $w_1 \cdot w_2 \cdot \ldots$ such that $w_i \in \mathcal{L}$ for all $i \geq 1$. Note that $\mathcal{L}^\omega$ is empty iff either $\mathcal{L} = \emptyset$ or $\mathcal{L} = \{\varepsilon\}$.

Definition 1 (M-substitution [11,12]) Let $w \in \Sigma^*$, $\Box \in \Sigma_{int}$, and $\mathcal{L} \subseteq \Sigma^*$. The M-substitution of $\Box$ by $\mathcal{L}$ in w, denoted by $w \curvearrowleft_\Box \mathcal{L}$, is the language of finite words over Σ obtained by replacing occurrences of $\Box$ in w by minimally well-matched words in $\mathcal{L}$. Formally, $w \curvearrowleft_\Box \mathcal{L}$ is inductively defined as follows:

- $\varepsilon \curvearrowleft_\Box \mathcal{L} \stackrel{\text{def}}{=} \{\varepsilon\}$;
- $(\Box \cdot w') \curvearrowleft_\Box \mathcal{L} \stackrel{\text{def}}{=} \big(MWM(\mathcal{L}) \cdot (w' \curvearrowleft_\Box \mathcal{L})\big) \cup \big((\{\Box\} \cap \mathcal{L}) \cdot (w' \curvearrowleft_\Box \mathcal{L})\big)$
- $(\sigma \cdot w') \curvearrowleft_\Box \mathcal{L} \stackrel{\text{def}}{=} \{\sigma\} \cdot (w' \curvearrowleft_\Box \mathcal{L})$ for each $\sigma \in \Sigma \setminus \{\Box\}$.

For two languages $\mathcal{L}, \mathcal{L}' \subseteq \Sigma^*$ and $\Box \in \Sigma_{int}$, the *M-substitution of $\Box$ by $\mathcal{L}'$ in* $\mathcal{L}$, written $\mathcal{L} \curvearrowleft_\Box \mathcal{L}'$, is defined as $\mathcal{L} \curvearrowleft_\Box \mathcal{L}' \stackrel{\text{def}}{=} \bigcup_{w \in \mathcal{L}} w \curvearrowleft_\Box \mathcal{L}'$. Note that $\curvearrowleft_\Box$ is associative, and $\{\Box\} \curvearrowleft_\Box \mathcal{L} = MWM(\mathcal{L})$ if $\{\Box\} \cap \mathcal{L} = \emptyset$.

Definition 2 (S-closure [11,12]) Given $\mathcal{L} \subseteq \Sigma^*$ and $\Box \in \Sigma_{int}$, the S-closure of $\mathcal{L}$ through $\Box$, denoted by $\mathcal{L}^{\circlearrowleft_\Box}$, is defined as follows:

$$\mathcal{L}^{\circlearrowleft_\Box} \stackrel{\text{def}}{=} \bigcup_{n \geq 0} MWM(\mathcal{L}) \underbrace{\curvearrowleft_\Box (\mathcal{L} \cup \{\Box\}) \curvearrowleft_\Box \ldots \curvearrowleft_\Box (\mathcal{L} \cup \{\Box\})}_{n \text{ occurrences of } \curvearrowleft_\Box}.$$

Example 2 Let $\Sigma_{call} = \{c_1, c_2\}$, $\Sigma_{ret} = \{r_1, r_2\}$, and $\Sigma_{int} = \{\Box\}$. Let us consider the languages $\mathcal{L} = \{c_1 \,\Box\, r_1,\ c_2 \,\Box\, r_2\}$ and $\mathcal{L}' = \{c_1\, r_1,\ c_2\, r_2\}$. Then,

$$\begin{aligned} \mathcal{L}^{\circlearrowleft_\Box} &= \{c_{i_1}\, c_{i_2} \ldots c_{i_n} \Box\, r_{i_n} \ldots r_{i_2}\, r_{i_1} \mid n \geq 1,\ i_1, \ldots, i_n \in \{1,2\}\} \text{ and} \\ \mathcal{L}^{\circlearrowleft_\Box} \curvearrowleft_\Box \mathcal{L}' &= \{c_{i_1}\, c_{i_2} \ldots c_{i_n}\, r_{i_n} \ldots r_{i_2}\, r_{i_1} \mid n \geq 2,\ i_1, \ldots, i_n \in \{1,2\}\}. \end{aligned}$$

Definition 3 The syntax of VRE α and ω-VRE β over Σ is defined as follows:

$$\begin{aligned} \alpha &:= \emptyset \mid \varepsilon \mid \mathit{int} \mid \mathit{call} \mid \mathit{ret} \mid \sigma \mid \alpha \cup \alpha \mid \alpha \cdot \alpha \mid \alpha^* \mid \alpha \curvearrowleft_\Box \alpha \mid \alpha^{\circlearrowleft_\Box} \\ \beta &:= \alpha^\omega \mid \beta \cup \beta \mid \alpha \cdot \beta \end{aligned}$$

where $\sigma \in \Sigma$. The basic expressions *int*, *call*, *ret* are used to denote in a succinct way the languages Σ_{int}, Σ_{call}, and Σ_{ret} (this is just syntactic sugar). Note that, for any given $a \in \Sigma$—and in particular for $a \in \Sigma_{int}$—, a is a VRE expression.

A VRE α denotes a language of finite words over Σ, written $\mathcal{L}(\alpha)$, defined inductively in the obvious way. Similarly, an ω-VRE β denotes a language of infinite words over Σ, written $\mathcal{L}(\beta)$.

Note that ω-VRE are defined in terms of VRE in the same way as ω-regular expressions are defined in terms of regular expressions. We also consider syntactical fragments of VRE and ω-VRE.

Definition 4 (Well-matched and well-formed VRE) The subclass of *well-matched* VRE η over Σ is defined by the following syntax:

$$\eta := \emptyset \mid \varepsilon \mid \Box \mid c \cdot \xi \cdot r \mid \eta \cup \eta \mid \eta \cdot \eta \mid \eta^* \mid \eta \curvearrowleft_{\Box} \eta \mid \eta^{\circlearrowleft_{\Box}}$$

where $\Box \in \Sigma_{int}$, $c \in \Sigma_{call}$, $r \in \Sigma_{ret}$, and ξ is a standard regular expression over Σ_{int}, defined as follows:

$$\xi := \emptyset \mid \varepsilon \mid \Box \mid \xi \cup \xi \mid \xi \cdot \xi \mid \xi^*$$

A VRE α is *well-formed* if each sub-expression of α of the form $(\alpha_1 \curvearrowleft_{\Box} \alpha_2)$ or $\alpha_1^{\circlearrowleft_{\Box}}$ is well-matched, and an ω-VRE β is *well-formed* if each VRE occurring in β is well-formed.

As usual, the size $|\alpha|$ of a VRE α is the length of the string describing α. Well-formed VRE are introduced because, as we will establish below, they are enough to express the whole class of visibly pushdown languages. Moreover, as we will see in Section 5, they allow a more efficient treatment than the general class of VRE.

Theorem 1 (from [11,12]) (*Well-formed*) *VRE and* (*well-formed*) *ω-VRE capture the classes of VPL and ω-VPL, respectively.*

Proof The results for VRE and ω-VRE were established in [11,12]. By definition of well-formed ω-VRE, in order to conclude the proof of the theorem, we only need to show that well-formed VRE are sufficient to capture the class of VPL. This can be proved by a straightforward adaptation of the translation from NVPA to VRE in [11,12] (details can be found in Appendix A). □

3 Visibly Linear Temporal Logic (VLTL)

In this section, we introduce the Visibly Linear Temporal Logic (VLTL), an extension of Regular Linear Temporal Logic (RLTL) with past (see [21,28]) obtained by replacing regular expressions in the temporal modalities of RLTL with VRE.

Syntax and semantics of VLTL. The syntax of VLTL formulas φ over the pushdown alphabet Σ is as follows:

$$\varphi := \mathtt{true} \mid \varphi \vee \varphi \mid \neg\varphi \mid \alpha;\varphi \mid \varphi;\alpha \mid \varphi|\alpha\rangle\!\rangle\varphi \mid \varphi|\alpha\rangle\varphi \mid \varphi\langle\!\langle\alpha|\varphi \mid \varphi\langle\alpha|\varphi$$

where α is a VRE over Σ, the symbol ; is the sequencing operator, $|\rangle\!\rangle$ and $\langle\!\langle|$ are the *future power operator* and the *past power operator*, and $|\rangle$ and $\langle|$ are the *future weak power operator* and the *past weak power operator*. The power formulas $\varphi_1|\alpha\rangle\!\rangle\varphi_2$, $\varphi_1\langle\!\langle\alpha|\varphi_2$, $\varphi_1|\alpha\rangle\varphi_2$, and $\varphi_1\langle\alpha|\varphi_2$ are built from three elements: φ_2 (the *attempt*), φ_1 (the *obligation*), and α (the *delay*). Informally, for $\varphi_1|\alpha\rangle\!\rangle\varphi_2$ to hold, either the attempt holds, or the obligation is met and the whole formula evaluates successfully after the delay; additionally, the attempt

must be eventually met. The expression $\varphi_1 \langle\!\langle \alpha | \varphi_2$ works similarly except that the formula is repeated before the delay. The weak formulas $\varphi_1 | \alpha \rangle \varphi_2$ and $\varphi_1 \langle \alpha | \varphi_2$ do not require the attempt to be eventually met. A VLTL formula φ is *well-formed* if every VRE occurring in φ is well-formed. For a VLTL formula φ, we denote by $\|\varphi\|$ the integer 1 if either $\varphi = \texttt{true}$ or φ has a Boolean connective at its root; otherwise, $\|\varphi\|$ is the size of the VRE associated with the root operator of φ. The size $|\varphi|$ of a VLTL formula φ is defined as $|\varphi| \stackrel{\text{def}}{=} \sum_{\psi \in SF(\varphi)} \|\psi\|$, where $SF(\varphi)$ is the set of sub-formulas of φ. For clarity in the presentation, we assume that Boolean operators have lower precedence than all other operators, so $\alpha; \varphi \vee (\varphi \wedge \psi; \texttt{true})$ is equivalent to $(\alpha; \varphi) \vee (\varphi \wedge (\psi; \texttt{true}))$.

VLTL formulas φ are interpreted over *infinite pointed words* (w, i) over Σ. The semantics of VLTL is given by the satisfaction relation $(w, i) \models \varphi$, defined inductively as follows:

$$
\begin{array}{lll}
(w,i) \models \texttt{true} & & \\
(w,i) \models \neg\varphi & \Leftrightarrow & (w,i) \not\models \varphi \\
(w,i) \models \varphi_1 \vee \varphi_2 & \Leftrightarrow & (w,i) \models \varphi_1 \text{ or } (w,i) \models \varphi_2 \\
(w,i) \models \alpha; \varphi & \Leftrightarrow & \text{for some } j > i,\ (w,j) \models \varphi \text{ and } w[i,j] \in \mathcal{L}(\alpha) \\
(w,i) \models \varphi; \alpha & \Leftrightarrow & \text{for some } j < i,\ (w,j) \models \varphi \text{ and } w[j,i] \in \mathcal{L}(\alpha) \\
(w,i) \models \varphi_1 | \alpha \rangle\!\rangle \varphi_2 & \Leftrightarrow & \text{for some sequence } i = j_1 < \ldots < j_n,\ (w,j_n) \models \varphi_2 \\
 & & \text{and for all } 1 \leq k < n,\ w[j_k, j_{k+1}] \in \mathcal{L}(\alpha) \text{ and } (w,j_k) \models \varphi_1 \\
(w,i) \models \varphi_1 \langle\!\langle \alpha | \varphi_2 & \Leftrightarrow & \text{for some sequence } j_1 < \ldots < j_n = i,\ (w,j_1) \models \varphi_2 \\
 & & \text{and for all } 1 < k \leq n,\ w[j_{k-1}, j_k] \in \mathcal{L}(\alpha) \text{ and } (w,j_k) \models \varphi_1 \\
(w,i) \models \varphi_1 | \alpha \rangle \varphi_2 & \Leftrightarrow & \textit{either } (w,i) \models \varphi_1 | \alpha \rangle\!\rangle \varphi_2, \\
 & & \textit{or for some infinite sequence } i = j_1 < j_2 < \ldots, \\
 & & w[j_k, j_{k+1}] \in \mathcal{L}(\alpha) \text{ and } (w,j_k) \models \varphi_1 \text{ for all } k \geq 1 \\
(w,i) \models \varphi_1 \langle \alpha | \varphi_2 & \Leftrightarrow & \textit{either } (w,i) \models \varphi_1 \langle\!\langle \alpha | \varphi_2, \\
 & & \textit{or for some sequence } 1 = j_1 < \ldots < j_n = i,\ (w,j_n) \models \varphi_1 \\
 & & \text{and } w[j_k, j_{k+1}] \in \mathcal{L}(\alpha) \text{ and } (w,j_k) \models \varphi_1 \text{ for all } 1 \leq k < n
\end{array}
$$

It is important to note that we have adopted an *overlapping* and *strict* semantics for the temporal modalities of VLTL. The term overlapping means that adjacent position intervals $[i, j]$ and $[j, h]$, resp. $[i, j]$ and $[j, \infty[$, share position j, while the term strict means that we additionally require that the start point and the end point of a position interval are distinct (i.e, $i < j$ and $j < h$). Thus, for example, $w[j_{k-1}, j_k]$ and $w[j_k, j_{k+1}]$ in the semantics of $\varphi_1 | \alpha \rangle\!\rangle \varphi_2$ both share the letter $w(j_k)$ at position j_k. Similarly, in the semantics of the future sequencing operator, the overlapping intervals are $[i, j]$ and $[j, \infty]$, where $i < j$. The choice considered subsumes the other possible choices. In other terms, one can introduce derived operators for expressing the non-overlapping and/or non-strict semantics. We illustrate this by considering the future sequencing operator and the future power operator. The non-overlapping semantics of $\varphi_1 | \alpha \rangle\!\rangle \varphi_2$ and $\alpha; \varphi$ can be expressed as $\varphi_1 | \alpha' \rangle\!\rangle \varphi_2$ and $\alpha'; \varphi$, respectively, where $\alpha' = \alpha \cdot (int \cup ret \cup call)$. It is easy to show that the non-strict semantics and the proposed semantics of $\varphi_1 | \alpha \rangle\!\rangle \varphi_2$ coincide. On the other hand, the non-strict semantics of $\alpha; \varphi$ can be expressed in the proposed

semantics by the formula $\alpha; \varphi \vee (\varphi \wedge \alpha''; \mathtt{true})$, where $\alpha'' = [\alpha]_1 \cdot (int \cup ret \cup call)$, and $[\alpha]_1 = \sigma_1 \cup \ldots \cup \sigma_n$, where $\{\sigma_1, \ldots, \sigma_n\}$ corresponds to $\mathcal{L}(\alpha) \cap \Sigma$. One can compute in linear time (in the size of α and n) the possible empty set of such letters $\sigma_1, \ldots, \sigma_n$. Note that the overlapping semantics is necessary for allowing a direct and linear-time translation of known temporal logics like CaRet [3] and NWTL [2] into VLTL (see Subsection 3.1).

The ω-pointed language $\mathcal{L}_p(\varphi)$ of φ is the set of infinite pointed words (w, i) over Σ satisfying φ, that is $(w, i) \models \varphi$. The ω-language $\mathcal{L}(\varphi)$ of φ is the set of infinite words w over Σ such that $(w, 1) \in \mathcal{L}_p(\varphi)$. Two formulas φ_1 and φ_2 are *globally equivalent* if $\mathcal{L}_p(\varphi_1) = \mathcal{L}_p(\varphi_2)$. The *satisfiability* problem for VLTL is checking for a VLTL formula φ, whether $\mathcal{L}(\varphi) \neq \emptyset$. The *visibly pushdown model checking* problem for VLTL is checking for a VLTL formula φ over Σ and a *pushdown system* $\mathcal{P}$ (defined as a Büchi NVPA $\mathcal{P}$ over the same pushdown alphabet Σ and with all states accepting), whether $\mathcal{L}_\omega(\mathcal{P}) \subseteq \mathcal{L}(\varphi)$.

Note that the VLTL operators generalize both the operators of standard LTL with past (in particular, the next, previous, until, and since modalities) and the operators of ω-visibly rational expressions. For example, the until formula $\varphi_1 \,\mathcal{U}\, \varphi_2$ requires that either φ_2 holds (attempt) or otherwise φ_1 holds (obligation) and the formula is reevaluated after a delay of a single step. Similarly, the ω-visibly rational expression α^ω has no possible escape, a trivially fulfilled obligation, with a delay indicated by α. Thus, ω-VRE can describe sophisticated delays with trivial obligations and attempts, while conventional LTL constructs allow complex obligations and attempts, but trivial one-step delays. Power operators can be seen as a generalization of both types of constructs.

In the rest of this section, we use some VRE of constant size:

$$\begin{aligned}
\alpha_{ONE} &\stackrel{\text{def}}{=} int \cup ret \cup call \\
\alpha_{MWM} &\stackrel{\text{def}}{=} \Box \curvearrowleft_\Box (call \cdot (\alpha_{ONE})^* \cdot ret) \\
\alpha_{WM} &\stackrel{\text{def}}{=} (int^* \cdot (\alpha_{MWM})^*)^* \\
\alpha_{MR} &\stackrel{\text{def}}{=} (call^* \cdot \alpha_{WM})^*
\end{aligned}$$

where $\Box \in \Sigma_{int}$. Note that $\mathcal{L}(\alpha_{ONE}) = \Sigma$, $\mathcal{L}(\alpha_{MWM}) = MWM(\Sigma)$, $\mathcal{L}(\alpha_{WM}) = WM(\Sigma)$, and $\mathcal{L}(\alpha_{MR})$ is the set of finite words where no unmatched return occurs. We use some shortcuts in VLTL:

$$\mathbf{G}\varphi \stackrel{\text{def}}{=} \varphi \,|\alpha_{ONE} \cdot \alpha_{ONE}\rangle\, \neg\mathtt{true} \qquad \ominus\varphi \stackrel{\text{def}}{=} \varphi; (\alpha_{ONE} \cdot \alpha_{ONE})$$

That is, $\mathbf{G}\varphi$ is the LTL always operator and $\ominus\varphi$ is the LTL previous operator.

Expressiveness of VLTL. First, we observe that (well-formed) ω-VRE can be translated in linear-time into language-equivalent (well-formed) VLTL formulas

by the mapping f from ω-VRE to VLTL inductively defined as follows:

$$\begin{aligned} f(\alpha^\omega) &\stackrel{\text{def}}{=} \mathtt{true}|\, \alpha \cdot \alpha_{ONE}\, \rangle \neg \mathtt{true} \\ f(\beta \cup \beta') &\stackrel{\text{def}}{=} f(\beta) \vee f(\beta') \\ f(\alpha \cdot \beta) &\stackrel{\text{def}}{=} (\alpha \cdot \alpha_{ONE}); f(\beta) \end{aligned}$$

Thus, by Theorem 1, (well-formed) VLTL formulas can express every ω-VPL. Note that we do not need past modalities. The converse direction holds as well (see Section 6). Hence, we obtain the following result.

Theorem 2 (*Future Well-formed*) *VLTL formulas capture the class of ω-VPL.*

3.1 Comparison with Known Context-free Extensions of LTL.

We compare now VLTL with some known context-free extensions of LTL: CaRet [3], NWTL [2], and NWTL$^+$ [2]. Recall that NWTL and NWTL$^+$ are expressively complete for the first-order fragment FO$_\mu$ of MSO$_\mu$ [2], while it is an intriguing open question whether the same holds for CaRet [2], the latter being subsumed by NWTL$^+$. In the analysis of recursive programs, an important feature of CaRet and NWTL$^+$ is that they allow to express in a natural way LTL properties over non-regular patterns such as (*) the stack content at a given position, and (**) the local computations of procedures which skip over nested procedure invocations. As we will prove in the following, these logics can be easily translated in linear time into VLTL. Additionally, the logic VLTL can specify more expressive regular properties over the patterns (*) and (**) such as the following requirement for a given $N \geq 1$, "whenever the procedure A is invoked, the depth of the stack content is a multiple of N", which can be expressed by the following VLTL formula φ_N,

$$\mathbf{G}(c_A \longrightarrow (\neg \ominus \mathtt{true}); \alpha_N) \qquad \alpha_N \stackrel{\text{def}}{=} [\underbrace{(\alpha_{WM} \cdot call \cdot \ldots \cdot \alpha_{WM} \cdot call}_{N \text{ times}} \cdot \alpha_{WM})]^*$$

where the call c_A denotes the invocation of procedure A. We claim that no CaRet formula can express requirement φ_N. Indeed, let c be a call action such that $c \neq c_A$, and for each $n \geq 1$, let w_n be the infinite word over Σ given by $c^n \cdot c_A \cdot c^\omega$. Evidently, $(w_n, 1) \in \mathcal{L}(\varphi_N)$ iff $n+1$ is a multiple of N. By the syntax and semantics of CaRet (which is recalled later), one can easily show by structural induction that for each CaRet formula ψ, there is a natural number $k_\psi \geq 1$ (actually, it suffices to take $k_\psi > |\psi|$) such that for all $n, n' \geq k_\psi$, ψ cannot distinguish between w_n and $w_{n'}$ (that is, $(w_n, 1) \in \mathcal{L}(\psi)$ iff $(w_{n'}, 1) \in \mathcal{L}(\psi)$). On the other hand for all $k \geq 1$, there are $n, n' \geq k$ such that $n+1$ is a multiple of N and $n'+1$ is not. Hence, the claim holds.

As another interesting example, we consider a bounded-response context-free requirement which can be easily expressed in FO$_\mu$ and VLTL, but for which we are not aware of any simple equivalent NWTL$^+$ formula: "every

internal request $req \in \Sigma_{int}$ is followed by a response $res \in \Sigma_{int}$ *before* the procedure where *req* occurs terminates". This property can be specified in VLTL as follows:

$$\mathbf{G}(req \longrightarrow (\alpha_{MR} \cdot res); \mathtt{true})$$

We show now that formulas of the logics CaRet, NWTL and NWTL$^+$ can be inductively translated in linear time into VLTL. We begin by recalling the syntax and semantics of CaRet [3], NWTL [2], and NWTL$^+$ [2].

Full CaRet [3]. CaRet extends standard LTL with past by allowing non-regular versions of the standard LTL temporal modalities. The semantics of these modalities is based on two distinct notions of non-local successor for a position i along an infinite word w over the fixed pushdown alphabet Σ (see [3]):

The abstract successor of i along w, $\mathsf{succ}(\mathsf{a}, w, i)$.
- If i is a call with a matching return, then $\mathsf{succ}(\mathsf{a}, w, i)$ is the matching return of i.
- If i is a call with no matching return, then $\mathsf{succ}(\mathsf{a}, w, i) = \perp$ (where the symbol $\perp$ means 'undefined').
- If i is not a call, then $\mathsf{succ}(\mathsf{a}, w, i) = i + 1$ if $i + 1$ is not a return position, and $\mathsf{succ}(\mathsf{a}, w, i) = \perp$ otherwise.

The caller of i along w, $\mathsf{succ}(\mathsf{c}, w, i)$.
- $\mathsf{succ}(\mathsf{c}, w, i)$ is the greatest call position $i_c < i$ such that either $\mathsf{succ}(\mathsf{a}, w, i_c) = \perp$ or $\mathsf{succ}(\mathsf{a}, w, i_c) > i$ if such a call position exists;
- otherwise, $\mathsf{succ}(\mathsf{c}, w, i) = \perp$.

For every pair of positions i and j with $1 \leq i \leq j$, and $dir \in \{\mathsf{a}, \mathsf{c}\}$, a *dir-path of w from i to j*, is a sequence of positions $i = j_1 < j_2 < \ldots < j_n = j$ such that for all $1 \leq k < n$, $j_{k+1} = \mathsf{succ}(dir, w, j_k)$ if $dir = \mathsf{a}$, and $j_k = \mathsf{succ}(dir, w, j_{k+1})$ if $dir = \mathsf{c}$. Note that there is at most one *dir*-path of w from i to j. Intuitively, in the analysis of recursive programs, the abstract paths (i.e., the a-paths) capture the local computation within a procedure removing computation fragments corresponding to nested calls, while a caller path (i.e., a c-path) captures the content of the call-stack of a procedure. The syntax of CaRet over Σ is defined as follows:

$$\varphi := \sigma \mid \varphi \vee \varphi \mid \neg\varphi \mid \bigcirc\varphi \mid \ominus\varphi \mid \bigcirc^{dir}\varphi \mid \ominus^{dir}\varphi \mid \varphi\,\mathcal{U}\varphi \mid \varphi\,\mathcal{S}\varphi \mid \varphi\,\mathcal{U}^{dir}\varphi \mid \varphi\,\mathcal{S}^{dir}\varphi$$

where $\sigma \in \Sigma$, $dir \in \{\mathsf{a}, \mathsf{c}\}$, $\bigcirc$, $\ominus$, $\mathcal{U}$, and $\mathcal{S}$ are the standard 'next', 'previous', 'until', and 'since' LTL modalities, respectively, and $\bigcirc^{\mathsf{a}}$, $\ominus^{\mathsf{a}}$, $\mathcal{U}^{\mathsf{a}}$, and $\mathcal{S}^{\mathsf{a}}$ are their non-regular abstract versions, and $\bigcirc^{\mathsf{c}}$, $\ominus^{\mathsf{c}}$, $\mathcal{U}^{\mathsf{c}}$, and $\mathcal{S}^{\mathsf{c}}$ are their non-regular caller versions. The semantics of the non-regular temporal modalities

is as follows.

$$
\begin{array}{ll}
(w,i) \models \bigcirc^{dir}\varphi & \Leftrightarrow (w,j) \models \varphi \text{ for some } j > i \text{ such that } j = \mathsf{succ}(\mathsf{a},w,i) \\
& \quad \text{if } dir = \mathsf{a}, \text{ and } i = \mathsf{succ}(\mathsf{c},w,j) \text{ if } dir = \mathsf{c} \\
(w,i) \models \ominus^{dir}\varphi & \Leftrightarrow (w,j) \models \varphi \text{ for some } j < i \text{ such that } i = \mathsf{succ}(\mathsf{a},w,j) \\
& \quad \text{if } dir = \mathsf{a}, \text{ and } j = \mathsf{succ}(\mathsf{c},w,i) \text{ if } dir = \mathsf{c} \\
(w,i) \models \varphi_1 \,\mathcal{U}^{dir}\, \varphi_2 & \Leftrightarrow \text{there is a } dir\text{-path } i = j_1 < j_2 \ldots < j_n \text{ such that} \\
& \quad (w,j_n) \models \varphi_2 \text{ and } (w,j_k) \models \varphi_1 \text{ for all } 1 \leq k < n \\
(w,i) \models \varphi_1 \,\mathcal{S}^{dir}\, \varphi_2 & \Leftrightarrow \text{there is a } dir\text{-path } j_1 < j_2 \ldots < j_n = i \text{ such that} \\
& \quad (w,j_1) \models \varphi_2 \text{ and } (w,j_k) \models \varphi_1 \text{ for all } 1 < k \leq n
\end{array}
$$

The logics NWTL and NWTL$^+$ [2]: these logics are based on the notion of *summary path*. Formally, for an infinite word w over Σ and two positions i and j such that $i \leq j$, a *summary path of w from i to j* is a sequence $i = j_1 < j_2 \ldots < j_n = j$ such that for all $1 \leq k < n$: if j_k is a matched-call and $\mathsf{succ}(\mathsf{a},w,j_k) \leq j$, then $j_{k+1} = \mathsf{succ}(\mathsf{a},w,j_k)$; otherwise $j_{k+1} = j_k + 1$. Note that there is exactly one summary path from i to j. The logic NWTL$^+$ extends CaRet with the binary modalities $\mathcal{U}^{\mathsf{c}}$ and $\mathcal{S}^{\mathsf{c}}$, which correspond to the standard until and since modalities of LTL interpreted on summary paths. NWTL is obtained from NWTL$^+$ by disallowing modalities $\mathcal{U}^{dir}, \mathcal{S}^{dir}, \bigcirc^{\mathsf{c}}, \ominus^{\mathsf{c}}$, where $dir \in \{\mathsf{a},\mathsf{c}\}$.

From NWTL$^+$ to VLTL. We assume that Σ_{int} contains at least a symbol $\Box$. Fix an infinite word w over Σ. For a VRE α over Σ and a sequence of positions $\nu = j_1 < \ldots < j_n$, ν *satisfies* α *along* w if $w[j_k, j_{k+1}] \in \mathcal{L}(\alpha)$ for all $1 \leq k < n$. We define the following constant-size VRE:

$$
\begin{array}{l}
\alpha_{\mathsf{a}} \stackrel{\text{def}}{=} \alpha_{MWM} \cup ((int \cup ret) \cdot (int \cup call)) \\
\alpha_{\mathsf{c}} \stackrel{\text{def}}{=} call \cdot \alpha_{WM} \cdot (call \cup \varepsilon) \\
\alpha^1_{\varsigma} \stackrel{\text{def}}{=} \alpha_{MWM} \cup ((int \cup ret) \cdot \alpha_{ONE}) \\
\alpha^2_{\varsigma} \stackrel{\text{def}}{=} \alpha_{MWM} \cup (\alpha_{ONE} \cdot (int \cup call))
\end{array}
$$

Lemma 1 *Let $\nu = j_1 < \ldots < j_n$ be a sequence of positions along w. Then:*
1. *ν is an abstract path of w from j_1 to j_n* iff *ν satisfies α_{a} along w.*
2. *ν is a caller path of w from j_1 to j_n* iff *ν satisfies α_{c} along w.*
3. *ν is the summary path of w from j_1 to j_n* iff *there is $1 \leq h \leq n$ such that $j_1 < \ldots < j_h$ satisfies α^1_{ς} along w and $j_h < \ldots < j_n$ satisfies α^2_{ς} along w.*

Proof Properties 1 and 2 easily follow, while Property 3 is a consequence of the following observation: the path $\nu = j_1 < \ldots < j_n$ is the summary path of w from j_1 to j_n iff there is $1 \leq h \leq n$ such that (i) each call position between j_1 and $j_h - 1$ has a matching return inside $[j_1, j_h]$, and (ii) each return position between $j_h + 1$ and j_n has a matched call inside $[j_h, j_n]$. $\Box$

We can prove the desired result.

Theorem 3 *For a CaRet, NWTL or NWTL$^+$ formula φ, one can construct in linear time a VLTL formula with* constant-size VRE *which is globally equivalent to φ.*

Proof Since NWTL$^+$ subsumes both CaRet and NWTL, it suffices to consider the logic NWTL$^+$. We define a linear-time computable mapping H : NWTL$^+$ $\rightarrow$ VLTL which associates to each NWTL$^+$ formula φ over Σ a globally equivalent VLTL formula $H(\varphi)$. The mapping is defined by structural induction as follows, where $dir \in \{\mathsf{a}, \mathsf{c}\}$. The mapping H is homomorphic with respect to the Boolean connectives and

- $H(\sigma) \stackrel{\text{def}}{=} (\sigma \cdot \alpha_{ONE}); \mathtt{true}$
- $H(\bigcirc \varphi) \stackrel{\text{def}}{=} (\alpha_{ONE} \cdot \alpha_{ONE}); H(\varphi)$
- $H(\ominus \varphi) \stackrel{\text{def}}{=} H(\varphi); (\alpha_{ONE} \cdot \alpha_{ONE})$
- $H(\bigcirc^{dir} \varphi) \stackrel{\text{def}}{=} \alpha_{dir}; H(\varphi)$
- $H(\ominus^{dir} \varphi) \stackrel{\text{def}}{=} H(\varphi); \alpha_{dir}$
- $H(\varphi_1 \; \mathcal{U} \; \varphi_2) \stackrel{\text{def}}{=} H(\varphi_1) \, |\alpha_{ONE} \cdot \alpha_{ONE}\rangle\!\rangle \, H(\varphi_2)$
- $H(\varphi_1 \; \mathcal{S} \; \varphi_2) \stackrel{\text{def}}{=} H(\varphi_1) \, \langle\!\langle \alpha_{ONE} \cdot \alpha_{ONE}| \, H(\varphi_2)$
- $H(\varphi_1 \; \mathcal{U}^{dir} \; \varphi_2) \stackrel{\text{def}}{=} H(\varphi_1) \, |\alpha_{dir}\rangle\!\rangle \, H(\varphi_2)$
- $H(\varphi_1 \; \mathcal{S}^{dir} \; \varphi_2) \stackrel{\text{def}}{=} H(\varphi_1) \, \langle\!\langle \alpha_{dir}| \, H(\varphi_2)$
- $H(\varphi_1 \; \mathcal{U}^{\varsigma} \; \varphi_2) \stackrel{\text{def}}{=} H(\varphi_1) \, |\alpha^1_{\varsigma}\rangle\!\rangle \, \big(H(\varphi_2) \vee (H(\varphi_1) \, |\alpha^2_{\varsigma}\rangle\!\rangle H(\varphi_2))\big)$
- $H(\varphi_1 \; \mathcal{S}^{\varsigma} \; \varphi_2) \stackrel{\text{def}}{=} H(\varphi_1) \, \langle\!\langle \alpha^2_{\varsigma}| \, \big(H(\varphi_2) \vee (H(\varphi_1) \, \langle\!\langle \alpha^1_{\varsigma}| H(\varphi_2))\big)$

Correctness of the construction can be easily proved by induction on the structure of the NWTL$^+$ formula and by using Lemma 1. This concludes the proof of the theorem. □

4 A Subclass of Alternating Jump Automata Over Infinite Words

Alternating Jump Automata (AJA) over finite and infinite words [10] are an alternative automata-theoretic characterization of VPL and ω-VPL. In this section, in order to capture VLTL formulas compositionally and efficiently, we introduce a subclass of two-way parity AJA with index 2 over infinite words. We call this class of automata *two-way stratified AJA with main states* (SAJA). In Section 5, we will show how to translate (well-formed) VRE into a subclass of AJA over finite words. This result, Theorem 6, is then used in Section 6 to handle the temporal operators in the translation of VLTL formulas into SAJA (Theorem 8).

Note that a naive approach based on the use of unrestricted two-way parity AJA would lead to decision procedures for VLTL that are computationally more expensive. More concretely, following [13], two-way parity AJA with n states and index k can be translated into equivalent Büchi ω-NVPA with $2^{O((nk)^2)}$ states and stack symbols. We show that SAJA with n states can be more efficiently translated into equivalent Büchi NVPA with $2^{O(n \log m)}$ states and

stack symbols, where m is the size of the largest non-trivial coBüchi stratum. We say that a coBüchi stratum is non-trivial when it contains both final and non-final states (otherwise it can be treated as a rejecting or as an accepting stratum).

Another technical issue is the efficient handling of logical negation. Like for RLTL, VLTL does not have a positive normal form. Hence, a construction for the negation operator must be given explicitly. Like for standard parity AFA, complementation of parity two-way AJA is easy: one only has to dualize the transition function and to complement the acceptance condition. However, the classical complementation for the parity acceptance condition increases the color assigned to every state by one unit, so that the total number of colors could grow linearly with the size of the formula.

Even though a negation immediately followed by another negation can be handled by an easy modification of the classical parity construction—by decreasing all colors in the second negation, instead of increasing them—, a correct inductive construction must also handle arbitrary operations in between two negations. Many constructions allow to reintroduce colors arbitrarily which precludes the use of the simple mechanism of decreasing colors whenever possible. Instead, we show here how to exploit the internal structure of SAJA sub-automata to provide a general complementation procedure that still uses only three colors.

The rest of this section is organized as follows. First, in Subsection 4.1 we recall the framework of parity two-way AJA. Then, in Subsection 4.2, we introduce the class of SAJA. In Subsection 4.3, we provide a characterization of the accepting runs of SAJA, which will be used in Subsection 4.4 for obtaining an efficient translation of SAJA into equivalent Büchi ω-NVPA.

4.1 Parity two-way AJA

AJA operate on words over a pushdown alphabet and extend standard alternating finite-state automata by also allowing non-local moves: when the current input position is a matched call, a copy of the automaton can move (jump) in a single step to the matched-return position. We also allow ε-moves and local and non-local backward moves. We first give the notion of *Alternating Jump Transition Tables* (AJT), which represent AJA without acceptance conditions. Let $DIR = \{\varepsilon, \rightarrow, \leftarrow, \curvearrowright, \curvearrowleft, \curvearrowright\!\!\bullet, \bullet\!\!\curvearrowleft\}$. Intuitively, the symbols $\rightarrow$ and $\leftarrow$ are used to denote forward and backward *local* moves, and $\curvearrowright$ and $\curvearrowleft$ are for *non-local* moves which lead from a matched call to the matching return, and vice-versa. The two additional symbols $\curvearrowright\!\!\bullet$ and $\bullet\!\!\curvearrowleft$ are used to denote variants of non-local moves. Intuitively, a $\curvearrowright\!\!\bullet$-move is a forward move which leads from a matched call i_c to the position following the matched return of i_c, while a $\bullet\!\!\curvearrowleft$-move is a backward move which leads from a matched return i_r to the position preceding the matching call of i_r. Even though these additional moves can be easily simulated with the other moves, they are crucial to handle in an

efficient way M-substitution and S-closure in the compositional translation of well-formed VRE into AJA over finite words shown in Section 5 below.

For a set X, $\mathcal{B}^+(X)$ denotes the set of positive Boolean formulas over X built from elements in X using $\vee$ and $\wedge$ (we also allow the formulas `true` and `false`). A model of a formula $\theta \in \mathcal{B}^+(X)$ is a subset of X which satisfies θ. A model is minimal if no strict subset of it satisfies θ. The *dual formula* $\widetilde{\theta}$ of θ is obtained from θ by switching $\vee$ and $\wedge$, and switching `true` and `false`.

A *tree* is a directed graph $\langle V, E\rangle$, where the set of vertices V is a prefix closed subset of $\mathbb{N}^*$ (ε is the root of the tree) and the set of edges E satisfies the following: $(v, v') \in E$ iff $v, v' \in V$ and $v' = v \cdot i$ for some $i \in \mathbb{N}$ (note that the set of edges is completely specified by V). For a set S, a *S-labeled tree* is a tuple $\langle V, E, L\rangle$, where $\langle V, E\rangle$ is a tree and $L : V \to S$ is a mapping associating to each vertex of the tree an element of S.

Two-way AJT [10]. A two-way AJT $\mathcal{T}$ over the pushdown alphabet Σ is a tuple $\mathcal{T} = \langle Q, q_0, \delta\rangle$, where Q is a finite set of states, $q_0 \in Q$ is the initial state, and $\delta : Q \times \Sigma \to \mathcal{B}^+(DIR \times Q \times Q)$ is a transition function. Given an atom $(dir, q, q') \in DIR \times Q \times Q$ of $\mathcal{A}$ and a pointed word (w, i) over Σ, the *effect* of the move (dir, q, q') with respect to (w, i) is the pair (j, p) defined as follows:

- $dir = \varepsilon$: $j = i$ and $p = q$.
- $dir = \rightarrow$: $j = i + 1$, and $p = q$ if $i + 1 \in Pos(w)$ and $p = q'$ otherwise.
- $dir = \leftarrow$: $j = i - 1$, and $p = q$ if $i - 1 \in Pos(w)$ and $p = q'$ otherwise.
- $dir = \curvearrowright$: if i is a call with matching return j_r, then $j = j_r$ and $p = q$; otherwise $j = i + 1$ and $p = q'$.
- $dir = \curvearrowleft$: if i is a return with matching call j_c, then $j = j_c$ and $p = q$; otherwise $j = i - 1$ and $p = q'$.
- $dir = \curvearrowright\!\!\bullet$: if i is a call with matching return j_r and $j_r + 1 \in Pos(w)$, then $j = j_r + 1$ and $p = q$; otherwise $j = i + 1$ and $p = q'$.
- $dir = \bullet\!\!\curvearrowleft$: if i is a return with matching call j_c and $j_c - 1 \in Pos(w)$, then $j = j_c - 1$ and $p = q$; otherwise $j = i - 1$ and $p = q'$.

Note that $(j, p) \in (\{0, |w|+1\} \cup Pos(w)) \times Q$ if w is finite, and $(j, p) \in (\{0\} \cup Pos(w)) \times Q$ otherwise. Accordingly, let $\overline{Pos}(w)$ be the set $\{0, |w|+1\} \cup Pos(w)$ if w is finite, and the set $\{0\} \cup Pos(w)$ otherwise. We introduce now the notion of run. Since we will also consider AJA over finite words, runs are defined for both finite and infinite words. Given a finite or infinite pointed word (w, i) on Σ and a state $p \in Q$, a *(i, p)-run of $\mathcal{T}$ over w* is a $\overline{Pos}(w) \times Q$-labeled tree $r = \langle V, E, L\rangle$ such that the root is labeled by (i, p). Intuitively, a vertex having label (j, q) with $j \in Pos(w)$ describes a copy of the automaton which is in state q and reads the j^{th} input position. Vertices having label $(0, q)$ or, in case w is finite, label $(|w| + 1, q)$ are used as markers for representing terminal backward and forward moves, respectively. Additionally, we require that the set of edges E is consistent with the transition function δ. Formally, vertices of r having label $(0, q)$ or, in case w is finite, label $(|w| + 1, q)$ have no successor. Moreover, for every vertex v with label (j, q) such that $j \in Pos(w)$, there is

a *minimal* model $X = \{(dir_1, q_1, q'_1), \ldots, (dir_n, q_n, q'_n)\}$ of $\delta(q, w(j))$ such that the set of successors of v is $\{v_1, \ldots, v_n\}$ and for all $k \in \{1, \ldots, n\}$, $L(v_k)$ is the effect of the move (dir_k, q_k, q'_k) with respect to (w, j).

For an infinite path $\pi = v_1, v_2, \ldots$ of the run r with $L(v_i) = (j_i, q_i)$ for all $i \geq 1$, we denote by $Inf(\pi)$ the set of states that occur infinitely many times along the sequence of states $q_1, q_2, \ldots$ visited by π. The path π is *strictly-forward* if (i) $j_i \leq j_{i+1}$ for all $i \geq 1$ and (ii) for infinitely many i, $j_i < j_{i+1}$. Moreover, the path π is *eventually strictly-forward* if π has a suffix which is strictly-forward.

The run r is *memoryless* if the behavior of the AJT $\mathcal{T}$ along r depends only on the current input position and current state. Formally, for all $v, v' \in V$ such that $L(v) = L(v')$, we have that $\{L(v'') \mid v'' \in E(v)\} = \{L(v'') \mid v'' \in E(v')\}$, where $E(v)$ is the set of successors of v and $E(v')$ is the set of successors of v'. Note that if r is *memoryless*, then r can be alternatively represented by the directed graph $G(r) = \langle V', E', v_0 \rangle$, where $V' \subseteq \overline{Pos}(w) \times Q$ is the set of labels associated with the vertices of r, $v_0 = (i, p)$ is the initial vertex, and $((j, q), (j', q')) \in E'$ iff there is $(v, v') \in E$ such that $L(v) = (j, q)$ and $L(v') = (j', q')$. In the rest of the paper, we use this representation for memoryless runs.

Parity two-way AJA. A parity two-way AJA $\mathcal{A}$ on infinite words over Σ is a tuple $\mathcal{A} = \langle Q, q_0, \delta, F_-, \Omega \rangle$, where $\langle Q, q_0, \delta \rangle$ is a two-way AJT over Σ, $F_- \subseteq Q$ is a *backward acceptance condition*, and $\Omega : Q \to \mathbb{N}$ is a parity acceptance condition assigning to each state an index (or color). A run r of $\mathcal{A}$ (that is a run of the associated AJT) over an infinite word is accepting if the following two conditions are satisfied:

- for each infinite path π of the run, the maximum index $\Omega(q)$ over the states $q \in Inf(\pi)$ is even;
- for each vertex of r with label $(0, q)$, it holds that $q \in F_-$.

The *ω-pointed language* $\mathcal{L}_p(\mathcal{A})$ of $\mathcal{A}$ is the set of infinite pointed words (w, i) over Σ such that there is an accepting (i, q_0)-run of $\mathcal{A}$ on w. The *ω-language* $\mathcal{L}(\mathcal{A})$ of $\mathcal{A}$ is the set of infinite words w over Σ such that $(w, 1) \in \mathcal{L}_p(\mathcal{A})$. The *dual automaton* $\widetilde{\mathcal{A}}$ of $\mathcal{A}$ is the parity two-way AJA obtained from $\mathcal{A}$ by dualizing the transition function, and by complementing the acceptance condition: formally, $\widetilde{\mathcal{A}} = \langle Q, q_0, \widetilde{\delta}, Q \setminus F_-, \widetilde{\Omega} \rangle$, where $\widetilde{\delta}(q, \sigma)$ is the dual formula of $\delta(q, \sigma)$ and $\widetilde{\Omega}(q) = \Omega(q) + 1$ for all $q \in Q$ and $\sigma \in \Sigma$.

Given an infinite pointed word (w, i) over Σ, we can associate in a standard way [26] to $\mathcal{A}$ and (w, i) an infinite-state parity game, where player 0 plays for acceptance, while player 1 plays for rejection. Winning strategies of player 0 correspond to accepting (i, q_0)-runs of $\mathcal{A}$ over w, and the plays conforming to a strategy correspond to the maximal paths of the associated run starting from the initial vertex. Since the existence of a winning strategy in parity games implies the existence of a memoryless one (see e.g. [34]), we can restrict ourselves to consider only memoryless runs of $\mathcal{A}$. Moreover, by [26] (see also [10]) the dual automaton $\widetilde{\mathcal{A}}$ of $\mathcal{A}$ accepts the complement of $\mathcal{L}_p(\mathcal{A})$. Note that the

backward acceptance condition F_- is necessary to handle the dualization of the terminal backward moves. Hence, the following holds.

Proposition 1 ([26,10]) *Given a parity two-way AJA $\mathcal{A} = \langle Q, q_0, \delta, F_-, \Omega\rangle$, $\mathcal{L}_p(\mathcal{A})$ coincides with the set of infinite pointed words (w, i) such that there is a* memoryless *(i, q_0)-run of $\mathcal{A}$ over w. Moreover, the dual automaton $\widetilde{\mathcal{A}}$ of $\mathcal{A}$ accepts the complement of $\mathcal{L}_p(\mathcal{A})$.*

4.2 Two-way stratified AJA with main states (SAJA)

We introduce now the class of SAJA as an extension of one-way hesitant AFA over infinite words introduced in [20]. This extension equips SAJA with two-way and non-regular capabilities in a restricted form. Intuitively, the ability of combining both forward and backward moves is syntactically restricted in such a way to ensure that every infinite path in a run is eventually strictly-forward. Moreover, for efficiency issues, we distinguish between *main states* and *secondary states*. Intuitively, in the translation of VLTL formulas into SAJA, main states are associated with the regular part of the formula, while secondary states are associated with the non-regular part (the M-substitution and S-closure operators in the VRE of the formula). The number of secondary states can be quartic in the number of main states. We now proceed with the formal definition of the syntax and semantics of SAJA.

A two-way AJT $\mathcal{T} = \langle Q, q_0, \delta\rangle$ is an AJT *with main states* if:

- the set of states is partitioned into a set M of *main states* and into a set S of *secondary states* such that $q_0 \in \mathsf{M}$.
- there are no moves from secondary states to main states. Hence, every path starting from a secondary state visits only secondary states.

Definition 5 (SAJA) A SAJA $\mathcal{A}$ is a tuple $\mathcal{A} = \langle Q, q_0, \delta, \mathsf{F}\rangle$ with $Q = \mathsf{M} \cup \mathsf{S}$, where $\langle Q, q_0, \delta\rangle$ is a two-way AJT with main states and F is a *strata family* of the form $\mathsf{F} = \{\langle \rho_1, Q_1, F_1\rangle, \ldots, \langle \rho_k, Q_k, F_k\rangle\}$, where

- $Q_1, \ldots, Q_k$ is a partition of the set of states Q;
- for all $1 \leq i \leq k$, $\rho_i \in \{-, \mathtt{t}, \mathsf{B}, \mathsf{C}\}$; and
- $F_i \subseteq Q_i$, such that $F_i = \emptyset$ whenever $\rho_i = \mathtt{t}$.

A stratum $\langle \rho_i, Q_i, F_i\rangle$ is called a *negative* stratum if $\rho_i = -$, a *transient* stratum if $\rho_i = \mathtt{t}$, a Büchi stratum (with Büchi acceptance condition F_i) if $\rho_i = \mathsf{B}$, and a coBüchi stratum (with coBüchi acceptance condition F_i) if $\rho_i = \mathsf{C}$. Additionally, we require that there is a partial order $\leq$ on the sets $Q_1, \ldots, Q_k$ such that the following holds:

R1. Moves from states in Q_i lead to states in components Q_j such that $Q_j \leq Q_i$. Additionally, if Q_i belongs to a transient stratum, there are no moves from Q_i leading to Q_i.

R2. For all atoms (dir, q, q') or (dir, q', q) occurring in δ, the following holds: (i) If the stratum of q is negative then $dir \in \{\leftarrow, \curvearrowleft, \overset{\bullet}{\curvearrowleft}, \varepsilon\}$ and otherwise $dir \in \{\rightarrow, \curvearrowright, \overset{\bullet}{\curvearrowright}, \varepsilon\}$; (ii) if $dir = \varepsilon$, then there are no ε-moves from q.

R3. For every Büchi or coBüchi stratum $\langle \rho_i, Q_i, F_i \rangle$, $F_i \cap \mathsf{S} = \emptyset$.

The SAJA $\mathcal{A}$ encodes a parity two-way AJA, denoted by $P(\mathcal{A})$, having the same AJT as $\mathcal{A}$ and whose backward acceptance condition F_- and parity acceptance condition Ω are defined as follows:

- F_- is the set of states q such that there is a negative stratum $\langle -, P, F \rangle$ of $\mathcal{A}$ so that $q \in F$.
- For all strata $\mathcal{S} = \langle \rho, P, F \rangle$ of $\mathcal{A}$ and $q \in P$, the index $\Omega(q)$ of q is
  - 0 if either $\mathcal{S}$ is a transient stratum, or $\mathcal{S}$ is a negative stratum, or $\mathcal{S}$ is a coBüchi stratum and $q \in P \setminus F$;
  - 2 if $\mathcal{S}$ is a Büchi stratum and $q \in F$;
  - otherwise, the index of q is 1.

An (accepting) run of $\mathcal{A}$ is an (accepting) run of $P(\mathcal{A})$. The *ω-pointed language* $\mathcal{L}_p(\mathcal{A})$ of $\mathcal{A}$ is $\mathcal{L}_p(P(\mathcal{A}))$ and the *ω-language* $\mathcal{L}(\mathcal{A})$ of $\mathcal{A}$ is $\mathcal{L}(P(\mathcal{A}))$.

In the above definition, R1 is the *stratum order requirement* and it ensures that every infinite path π of a run gets trapped in the component Q_i of some non-transient stratum. R2 is the *eventually syntactical requirement* and it ensures that Q_i belongs to a Büchi or coBüchi stratum and that π is eventually strictly-forward. R2 also ensures that for all runs and vertices having label of the form $(0, q)$, q belong to a negative stratum.

Let π be an infinite path of a run of a SAJA $\mathcal{A}$ and $\langle \rho_i, Q_i, F_i \rangle$ be the Büchi or coBüchi stratum in which π gets trapped. We say that the path π is *accepting* whenever $\mathit{Inf}(\pi) \cap F_i \neq \emptyset$ if $\rho_i = \mathsf{B}$ and $\mathit{Inf}(\pi) \cap F_i = \emptyset$ otherwise (i.e. π satisfies the corresponding Büchi or coBüchi requirement). Note that requirement R3 in the definition of SAJA ensures that whenever π starts at a vertex associated with a secondary state (hence, π visits only secondary states), then π is accepting if the stratum $\langle \rho_i, Q_i, F_i \rangle$ is a coBüchi stratum, and it is rejecting otherwise. By Definition 5, the backward acceptance condition and the parity acceptance condition encoded by a SAJA $\mathcal{A}$ ensure that a run of $\mathcal{A}$ is accepting *iff*: (1) all its infinite paths are accepting and (2) for each vertex with label $(0, q)$ such that q is in the stratum $\mathcal{S} = \langle \rho_i, Q_i, F_i \rangle$ (recall that $\mathcal{S}$ is ensured to be a negative stratum), it holds that $q \in F_i$.

The *SAJA-dual* $\widetilde{\mathcal{A}}$ of a SAJA $\mathcal{A} = \langle Q, q_0, \delta, \mathsf{F} \rangle$ is defined as $\widetilde{\mathcal{A}} = \langle Q, q_0, \widetilde{\delta}, \widetilde{\mathsf{F}} \rangle$, where $\widetilde{\delta}(q, \sigma)$ is the dual formula of $\delta(q, \sigma)$, and $\widetilde{\mathsf{F}}$ is obtained from F by converting a Büchi stratum $\langle \mathsf{B}, Q_i, F_i \rangle$ into the coBüchi stratum $\langle \mathsf{C}, Q_i, F_i \rangle$, a coBüchi stratum $\langle \mathsf{C}, Q_i, F_i \rangle$ into the Büchi stratum $\langle \mathsf{B}, Q_i, F_i \rangle$, and a negative stratum $\langle -, Q_i, F_i \rangle$ into the negative stratum $\langle -, Q_i, Q_i \setminus F_i \rangle$. By construction the dual automaton $\widetilde{P(\mathcal{A})}$ of the parity two-way $P(\mathcal{A})$ encoded by $\mathcal{A}$ and the parity two-way $P(\widetilde{\mathcal{A}})$ encoded by $\widetilde{\mathcal{A}}$ have the same AJT. Moreover, it is easy to check that a run r is accepting for $\widetilde{P(\mathcal{A})}$ iff it is accepting for $P(\widetilde{\mathcal{A}})$. Thus, by Proposition 1, we obtain the following lemma, which is crucial for handling, compositionally and efficiently, negation in VLTL formulas.

Lemma 2 *The SAJA-dual $\widetilde{\mathcal{A}}$ of a SAJA $\mathcal{A}$ accepts the complement of $\mathcal{L}_p(\mathcal{A})$.*

4.3 Characterization of accepting memoryless runs in SAJA

In this section, we first give a characterization of the fulfillment of the acceptance condition for a coBüchi stratum along a memoryless run of a SAJA in terms of the existence of an *odd ranking function*. The latter generalizes the notion of odd ranking function for standard coBüchi alternating finite-state automata [19] which intuitively allows to convert a coBüchi acceptance condition into a Büchi-like acceptance condition. Then, by exploiting the above result and a generalization of the Miyano-Hayashi construction [25], we give a characterization of the accepting memoryless runs of a given SAJA in terms of infinite sequences of finite sets satisfying determined requirements which can be easily checked by a Büchi NVPA. The approach we propose is a refinement of the method used in [13] to convert parity two-way AJA into equivalent Büchi ω-NVPA.

A SAJA is *pure* if it does not use the non-local moves $\curvearrowright$ and $\curvearrowleft$. The following observation is straightforward.

Remark 1 Given a SAJA $\mathcal{A}$, one can construct in linear time a *pure* SAJA $\mathcal{A}'$ such that $\mathcal{L}_p(\mathcal{A}') = \mathcal{L}_p(\mathcal{A})$.

The constructions given in [19, 25] are crucially based on the fact that in the runs of one-way alternating finite-state automata (AFA) over infinite words, for all positions i, every infinite path π starting from i visits all the positions $j \geq i$. The following definition and Remark 2 show that a weak variant of the above property holds for the class of SAJA. As we will see later, this is sufficient to adapt the constructions in [19, 25].

Definition 6 (Dominant positions) Let w be an infinite word over Σ. A position i of w is a *dominant position* whenever for every matched call j_c with matched return j_r, either $i \leq j_c$ or $i > j_r$.

Intuitively, the dominant positions of an infinite word w over Σ are obtained by erasing all the non-initial positions of the *maximal* minimally well-matched sub-words of w. Note that the set of dominant positions of w is infinite. We first make the following observation.

Remark 2 Let ρ be a run of a parity two-way AJA $\mathcal{A}$ over an infinite word w such that $\mathcal{A}$ does not use the non-local moves $\curvearrowright$ and $\curvearrowleft$. Let π be an infinite strictly-forward path of ρ that starts from a vertex with label (i, q), and let j be a dominant position with $j > i$. Then, π visits positions $j - 1$ and j.

We introduce now the notion of ranking function for memoryless runs of SAJA. Recall that for the class of coBüchi AFA, a ranking function [19] for a run is a mapping assigning to each vertex a rank taken from a finite set, with the restriction that the rank cannot increase along a move in the run. This ensures that along every infinite path, the rank always converges to a value. The ranking function is said to be *odd* [19] if it is guaranteed that for each infinite path in the run, the convergence value is always odd. If the values

always converges to an even value, the run is called even. It is known that a run of a coBüchi AFA is accepting iff there exists an odd ranking function. We adapt this result to the class of SAJA and we focus on *odd* ranking functions.

The *size* of a SAJA stratum $\langle \rho, P, F\rangle$ is the number of *main states* in P (we do not take into account the number of secondary states in P). A coBüchi stratum $\langle \mathsf{C}, P, F\rangle$ is *trivial* whenever $F = \emptyset$. Note that an infinite path of a SAJA run which gets trapped in a trivial coBüchi stratum is always accepting.

Definition 7 (Odd ranking functions) Let $\mathcal{A} = \langle Q, q_0, \delta, \mathsf{F}\rangle$ be a SAJA with $Q = \mathsf{M} \cup \mathsf{S}$, $\mathcal{S} = \langle \mathsf{C}, P, F\rangle$ be a non-trivial coBüchi stratum of $\mathcal{A}$ and $n = |P \cap \mathsf{M}|$ be the size of the stratum. For an infinite word w on Σ and a memoryless run $G = \langle V, E, v_0\rangle$ of $\mathcal{A}$ over w, a *ranking function of the stratum* $\mathcal{S}$ *for the run* G is a function $f_{\mathcal{S}} : V \to \{1, \ldots, 2n\}$ that satisfies the following:

1. for all $(j, q) \in V$ such that $q \in F$, $f_{\mathcal{S}}(j, q)$ is even (recall that $F \subseteq P \cap \mathsf{M}$);
2. for all $(j, q), (j', q') \in V$ such that (j', q') is a successor of (j, q) in G and $q, q' \in P \cap \mathsf{M}$, it holds that $f_{\mathcal{S}}(j', q') \leq f_{\mathcal{S}}(j, q)$.

Thus, since the image of $f_{\mathcal{S}}$ is bounded, for every infinite path $\pi = v_0, v_1, \ldots$ of G that gets trapped in the set of main states of the coBüchi stratum $\mathcal{S}$, $f_{\mathcal{S}}$ converges to a value. That is, there is a number l such that $f_{\mathcal{S}}(v_{l'}) = f_{\mathcal{S}}(v_l)$ for all $l' \geq l$. We say that $f_{\mathcal{S}}$ is *odd* if for all such infinite paths π of G, $f_{\mathcal{S}}$ converges to an odd value (or, equivalently, any of such paths π visits infinitely many times vertices v such that $f_{\mathcal{S}}(v)$ is odd).

Note that in the above definition, if $f_{\mathcal{S}}$ is odd, then π is accepting. The following lemma asserts that the existence of an odd ranking function is also a necessary condition for a memoryless run of a pure SAJA to be accepting. The construction in the proof of Lemma 3 is similar to the ranking construction in [19].

Lemma 3 *Let G be a memoryless run of a pure SAJA $\mathcal{A}$ over an infinite word w. Then, G is accepting* iff

1. *for every non-trivial co-Büchi stratum $\mathcal{S} = \langle \mathsf{C}, P, F\rangle$, there is an odd ranking function of $\mathcal{S}$ for the run G;*
2. *every infinite path of G which gets trapped in the component of a Büchi stratum $\mathcal{S} = \langle \mathsf{B}, P, F\rangle$ satisfies the Büchi acceptance condition F;*
3. *for each vertex $(0, q)$ such that q is in the stratum $\mathcal{S} = \langle \rho, P, F\rangle$, it holds that $q \in F$.*

Proof Let $\mathcal{A} = \langle Q, q_0, \delta, \mathsf{F}\rangle$ with $Q = \mathsf{M} \cup \mathsf{S}$.

Recall that in SAJA, moves from secondary states lead to secondary states as well, and for every coBüchi stratum $\mathcal{S} = \langle \mathsf{C}, P, F\rangle$, F does not contain secondary states. Hence, by Definition 7, it follows that Conditions 1–3 in the lemma imply that the memoryless run G is accepting.

For the converse implication, assume that $G = \langle V, E, v_0\rangle$ is an *accepting* memoryless run over an infinite word w. Conditions 2 and 3 in the lemma hold. For Condition 1, let $\mathcal{S} = \langle \mathsf{C}, P, F\rangle$ be a non-trivial coBüchi stratum of

$\mathcal{A}$ and let $n = |P \cap \mathsf{M}|$ be the size of the stratum. We need to show that there is an odd ranking function of $\mathcal{S}$ for the accepting run G. In the following, we construct a ranking function of $\mathcal{S}$ for G and show that it is odd. An F-vertex is a vertex of G associated with a state in F. Let $G' = \langle V', E' \rangle$ be a sub-graph of G and $v \in V'$. The vertex v is *finite in* G' if the set of vertices which are reachable from v in G' is finite. The vertex v is *F-free in* G' if no F-vertex is reachable from v in G'. Moreover, for each $l \geq 1$, we define

$$\mathit{width}(G', l) \stackrel{\text{def}}{=} |\{(l, q) \in V'\}|$$

to be the number of vertices in G' associated with position l. First, we inductively define an infinite sequence $(G_i = \langle V_i, E_i \rangle)_{i \geq 0}$ of sub-graphs of G as follows:

- V_0 is the set of $(P \cap \mathsf{M})$-vertices and $E_0 = E \cap V_0 \times V_0$.
- $V_{2i+1} \stackrel{\text{def}}{=} V_{2i} \setminus \{v \mid v \text{ is finite in } G_{2i}\}$ and $E_{2i+1} = E_{2i} \cap V_{2i+1} \times V_{2i+1}$.
- $V_{2i+2} \stackrel{\text{def}}{=} V_{2i+1} \setminus \{v \mid v \text{ is } F\text{-free in } G_{2i+1}\}$ and $E_{2i+2} = E_{2i+1} \cap V_{2i+2} \times V_{2i+2}$.

Since G_{2i+1} is obtained from G_{2i} by removing all the vertices that can only access finitely many vertices and the number of successors of any vertex is finite, it follows that every maximal path in the graph G_{2i+1} is infinite. We claim that if G_{2i+1} is not empty, then G_{2i+1} contains some F-free vertex. By contradiction, we assume the contrary, which implies that there is an infinite path π of G_{2i+1} which visits F-vertices infinitely many times. Since G_{2i+1} is a sub-graph of the run G, we obtain that π is an infinite path of the run G which gets trapped in the coBüchi stratum $\mathcal{S} = \langle \mathsf{C}, P, F \rangle$ and does not satisfy the coBüchi acceptance condition F. This is a contradiction since G is an accepting run. Hence, the claim follows. Since every maximal path in the graph G_{2i+1} is infinite, the claim implies that if G_{2i+1} is not empty, then there is an infinite path π of G_{2i+1} which visits only F-free vertices. Now, every infinite path of the run G is eventually strictly-forward because $\mathcal{A}$ satisfies the eventually syntactical requirement. Consequently, the infinite path π is eventually strictly forward and, by Remark 2, there is some position l such that for all *dominant* positions h of w with $h \geq l$, π visits some vertex associated with the position h. Since all the vertices of π, which are F-free vertices, are removed in G_{2i+2}, we obtain that for some $l \geq 1$ and all the dominant positions $h \geq l$,

$$\mathit{width}(G_{2i+2}, h) \leq \mathit{width}(G_{2i+1}, h) - 1$$

Since each step only removes vertices, we obtain that for some $l \geq 1$ and all the dominant positions $h \geq l$,

$$\mathit{width}(G_{2i+2}, h) \leq \mathit{width}(G_0, h) - (i + 1)$$

Since G_0 contains only $P \cap \mathsf{M}$-vertices and $n = |P \cap \mathsf{M}|$, we obtain that for some $l \geq 1$ and for all the dominant positions $h \geq l$, G_{2n} does *not* contain vertices associated with the dominant position h. Since the set of dominant

positions $h \geq l$ is infinite and, in particular, every infinite path of G, which is eventually strictly forward, visits some dominant position $h \geq l$, it follows that every vertex of G_{2n} is finite in G_{2n}. Thus, we obtain the following result.

Claim 1. G_{2n+1} is empty.

We define a ranking function $f_{\mathcal{S}} : V \to \{1, \ldots, 2n\}$ of the stratum $\mathcal{S} = \langle \mathsf{C}, P, F \rangle$ for the accepting memoryless run G as follows:

$$f_{\mathcal{S}}(v) = \begin{cases} 2i & \text{if } i \leq n,\ v \in V_{2i}, \text{ and } v \notin V_{2i+1} \\ 2i+1 & \text{if } i < n,\ v \in V_{2i+1}, \text{ and } v \notin V_{2i+2} \\ 1 & \text{otherwise} \end{cases}$$

Note that $f_{\mathcal{S}}$ is well-defined since $V_i \supseteq V_{i+1}$ for all $i \geq 0$. We only need to show the following.

Claim 2. $f_{\mathcal{S}}$ is an odd ranking function of the stratum $\mathcal{S}$ for the run G.

Proof of Claim 2. First, we show that $f_{\mathcal{S}}$ is a ranking function of $\mathcal{S}$ for the run G, in other words, $f_{\mathcal{S}}$ satisfies Properties 1 and 2 of Definition 7. For Property 1, let $(j, q) \in V$ such that $q \in F$. We need to prove that $f_{\mathcal{S}}(j, q)$ is even. Since $F \subseteq \mathsf{M} \cap P$, (j, q) is a vertex of G_0. Moreover, since (j, q) is not F-free, there is no $i \geq 0$ such that $(j, q) \in V_{2i+1}$ and $(j, q) \notin V_{2i+2}$. Thus, by Claim 1, there is $i \leq n$ such that $(j, q) \in V_{2i}$ and $(j, q) \notin V_{2i+1}$. By definition of $f_{\mathcal{S}}$, we obtain that $f_{\mathcal{S}}(j, q) = 2i$ and the result follows. For Property 2 of Definition 7, let $(j, q), (j', q') \in V$ such that (j', q') is a successor of (j, q) in G and $q, q' \in P \cap \mathsf{M}$. We need to show that $f_{\mathcal{S}}(j', q') \leq f_{\mathcal{S}}(j, q)$. Note that (j, q) and (j', q') are vertices of G_0. By Claim 1, there are $0 \leq i, i' \leq 2n$ such that $(j, q) \in V_i$, $(j, q) \notin V_{i+1}$, $(j', q') \in V_{i'}$, and $(j', q') \notin V_{i'+1}$. Moreover, either i is even and (j, q) is finite in G_i or i is odd and (j, q) is F-free in G_i. Since (j', q') is a successor of (j, q) in G_0, it follows that $i' \leq i$. Thus, by definition of $f_{\mathcal{S}}$, we obtain that $f_{\mathcal{S}}(j', q') \leq f_{\mathcal{S}}(j, q)$, and the result holds.

We show now that $f_{\mathcal{S}}$ is odd. Let $\pi = v_0, v_1, \ldots$ be an infinite path of G that gets trapped in the set of main states of the non-trivial coBüchi stratum $\mathcal{S}$. We need to show that $f_{\mathcal{S}}$ converges to an odd value along π. We assume the contrary and derive a contradiction. Since $f_{\mathcal{S}}$ is a ranking function of $\mathcal{S}$ for the run G, there is $k \geq 0$ such that for all $h \geq k$, $f_{\mathcal{S}}(v_h) = f_{\mathcal{S}}(v_k)$ and $f_{\mathcal{S}}(v_k)$ is even. By definition of $f_{\mathcal{S}}$, this means that there is $i \leq n$ such that $v_h \in V_{2i}$ and $v_h \notin V_{2i+1}$ for all $h \geq k$. This entails that v_h is finite in G_{2i} for all $h \geq k$. Hence, v_k is finite in G_{2i} and there is an infinite path from v_k in G_{2i}, which is a contradiction. Thus, the result follows, which concludes the proof of Claim 2.

This concludes the proof of Lemma 3. □

Now, based on Lemma 3 and the classical breakpoint construction [25], we give a characterization of the accepting memoryless $(1, q_0)$-runs of a pure SAJA $\mathcal{A} = \langle Q, q_0, \delta, \mathsf{F} \rangle$. Our characterization is given in terms of infinite sequences of finite sets, called *regions*, satisfying determined requirements which can be checked by Büchi NVPA.

Definition 8 (Regions) Let $\mathcal{A} = \langle Q, q_0, \delta, \mathsf{F}\rangle$ be a pure SAJA. A *rank* of a main state q of a coBüchi stratum is a natural number in $\{1, \ldots, 2n\}$, where n is the size of the stratum of q.

A *region of* $\mathcal{A}$ is a triple (R, O, f), where $R \subseteq Q$ is a set of states, $O \subseteq R$, and f is a mapping assigning to each non-trivial coBüchi main state $q \in R$ a rank of q such that $f(q)$ is even if $q \in F$, where $\langle \mathsf{C}, P, F\rangle$ is the stratum of q. A state q of $\mathcal{A}$ is *accepting* with respect to f if (1) either q is an accepting state of a Büchi stratum, or (2) q belongs to a coBüchi stratum and, additionally, $f(q)$ is odd if $q \in R$ and q is a main state of some non-trivial coBüchi stratum. The *stop* region is the region given by $(R, \emptyset, \emptyset)$ where R is the set of states belonging to the F-components of the negative strata $\langle -, P, F\rangle$ of $\mathcal{A}$.
Let $w \in \Sigma^\omega$. An infinite sequence of regions $\nu = (R_1, O_1, f_1), (R_2, O_2, f_2), \ldots$ is *good with respect to* w if for all $i \geq 1$, there is a mapping g_i assigning to each $q \in R_i$ a minimal model of $\delta(q, w(i))$ such that the following holds, where Acc_i denotes the set of accepting states of $\mathcal{A}$ with respect to f_i, and $(R_0, \emptyset, \emptyset)$ is the stop region:

- *Initialization.* $q_0 \in R_1$ and $O_1 = R_1 \setminus Acc_1$.
- For all $p \in R_i$ and $(dir, q, q') \in g_i(p)$, let (j, p') be the effect of (dir, q, q') with respect to (w, i); then, $p' \in R_j$ (*δ-consistency with respect to g_i*). Moreover, if p and p' are main states in the same non-trivial coBüchi stratum, then $f_j(p') \leq f_i(p)$ (*Ranking requirement with respect to g_i*).
- *Miyano-Hayashi requirement with respect to g_i.* For all $p \in O_i$ and $(dir, q, q') \in g_i(p)$ such that $dir \in \{\rightarrow, \curvearrowright\}$, let (j, p') be the effect of (dir, q, q') w.r.t. (w, i) (note that $j \neq 0$); if $p' \notin Acc_j$, then $p' \in O_j$.

The infinite sequence of regions ν is *accepting* with respect to w precisely when there are infinitely many *dominant* positions $k > 1$ of w such that $O_{k-1} = \emptyset$ and $O_k = R_k \setminus Acc_k$.

Intuitively, the infinite sequence of regions ν represents a graph $G = \langle V \subseteq \mathbb{N} \times Q, E, v_0\rangle$ where $v_0 = (1, q_0)$ and for all input positions $i \geq 1$, R_i is the set of vertices of G associated with position i. The initialization and δ-consistency requirement ensure that G is a memoryless $(1, q_0)$-run of $\mathcal{A}$ over w and for each vertex $(0, q)$ of the run, state q is in the F-component of some negative stratum $\langle -, P, F\rangle$. Additionally, the ranking requirement ensures that for each non-trivial coBüchi stratum $\mathcal{S}$, there is a ranking function $f_\mathcal{S}$ of $\mathcal{S}$ for the run G. By Lemma 3, the run is accepting if $f_\mathcal{S}$ is odd and Condition 2 in Lemma 3 holds. This, in turn, is equivalent to require that every infinite path of G visits infinitely many vertices in Acc, where Acc is the set of G-vertices (i, q) such that $q \in Acc_i$. We show in the proof of the following Theorem 4 that this last condition is satisfied iff there is an infinite sequence of dominant positions $1 = h_1 < h_2 < \ldots$ of w such that for all $i \geq 1$, any finite path of G that starts at position h_i and ends at position $h_{i+1} - 1$ visits a vertex in Acc. Thus, the Miyano-Hayashi and the acceptance requirements on the sets O_i ensure the existence of such an infinite sequence of dominant positions h_j (in particular, $O_{h_j - 1} = \emptyset$ and $O_{h_j} = R_{h_j} \setminus Acc_{h_j}$ for all $j > 1$). Formally, we obtain the following result.

Theorem 4 (Characterization theorem for pure SAJA) *For a pure SAJA $\mathcal{A}$ and an infinite word w over Σ, $w \in \mathcal{L}(\mathcal{A})$ iff there is an infinite sequence of regions of $\mathcal{A}$ which is good and accepting with respect to w.*

Proof Let $\mathcal{A} = \langle Q, q_0, \delta, \mathsf{F}\rangle$ be a pure SAJA with $Q = \mathsf{M} \cup \mathsf{S}$ and w be an infinite word over Σ.

"$\Leftarrow$" Assume that there is an infinite sequence ν of regions of $\mathcal{A}$ of the form $\nu = (R_1, O_1, f_1), (R_2, O_2, f_2), \ldots$ which is good and accepting with respect to w. We need to show that $w \in \mathcal{L}(\mathcal{A})$. For all $i \geq 1$, let Acc_i be the set of accepting states of $\mathcal{A}$ with respect to f_i, and g_i be the mapping assigning to each $q \in R_i$ a minimal model of $\delta(q, w(i))$ such that ν satisfies the δ-consistency requirement, the ranking requirement, and the Miyano-Hayashi requirement w.r.t. g_i. Let Q_0 be the set of states $p' \in Q$ such that $(0, p')$ is the effect of (dir, q, q') w.r.t. (w, i) for some $i \geq 1$, $p \in R_i$, and $(dir, q, q') \in g_i(p)$. Note that the δ-consistency requirement ensures that Q_0 contains only states belonging to the F-components of the negative strata $\langle -, P, F\rangle$ of $\mathcal{A}$. We define a graph $G = \langle V, E, v_0\rangle$ and show that it is an accepting memoryless $(1, q_0)$-run of $\mathcal{A}$ over w. The graph G is defined as follows:

- $v_0 = (1, q_0)$, $V \subseteq \mathbb{N} \times Q$ such that: (i) $(0, q) \in V$ iff $q \in Q_0$ and (ii) for all $i > 0$, $(i, q) \in V$ iff $q \in R_i$;
- there is an edge from (i, q) to (j, q') iff $i > 0$ and for some $(dir, p, p') \in g_i(q)$, (j, q') is the effect of (dir, p, p') w.r.t. (w, i).

Since the sequence of regions ν satisfies the initialization requirement and the δ-consistency requirement with respect to g_i for all $i \geq 1$, G is a memoryless $(1, q_0)$-run of $\mathcal{A}$ over w. It remains to be shown that G is accepting. We assume the contrary and derive a contradiction. Then, since $\mathcal{A}$ is a SAJA and the acceptance condition for the vertices $(0, q)$ is satisfied, there must be an infinite path $\pi = (h_1, q_1), (h_2, q_2), \ldots$ of G such that π is strictly forward and one of the following holds:

- for some Büchi stratum $\langle \mathsf{B}, P, F\rangle$, $q_i \in P \setminus F$ for all $i \geq 1$. Since $q_i \in R_{h_i}$, we obtain that $q_i \in R_{h_i} \setminus Acc_{h_i}$ for all $i \geq 1$.
- for some non-trivial coBüchi stratum $\langle \mathsf{C}, P, F\rangle$, $q_i \in P \cap \mathsf{M}$ for all $i \geq 1$, and for infinitely many $i \geq 1$, $q_i \in F$. [3] Since ν satisfies the ranking requirement w.r.t. g_{h_i}, $f_{h_{i+1}}(q_{i+1}) \leq f_{h_i}(q_i)$ for all $i \geq 1$. It follows that there is $l \geq 1$ such that $q_l \in F$ and for all $i \geq l$, $f_{h_i}(q_i) = f_{h_l}(q_l)$. In particular, $f_{h_l}(q_l)$ is even. Hence, for all $i \geq l$, $q_i \in R_{h_i} \setminus Acc_{h_i}$.

Thus, we obtain that there is an infinite path $\pi = (h_1, q_1), (h_2, q_2), \ldots$ of G which is strictly forward and such that $q_i \in R_{h_i} \setminus Acc_{h_i}$ for all $i \geq 1$. Hence, $R_{h_i} \setminus Acc_{h_i} \neq \emptyset$ for all $i \geq 1$. Since π is strictly forward, by Remark 2, π must visit all the positions $j - 1$ and j of w such that $j > h_1$ and j is a dominant position of w. Thus, since the sequence of regions ν is accepting, the following

[3] Recall that $F \subseteq \mathsf{M}$ and moves from secondary states lead to secondary states as well.

holds: (i) there is $\ell \geq 1$ such that $O_{h_\ell} = R_{h_\ell} \setminus Acc_{h_\ell} \neq \emptyset$ and (ii) for infinitely many $i \geq l$, $O_{h_i} = \emptyset$. On the other hand, since ν satisfies the Miyano-Hayashi requirement w.r.t. the mappings g_j, we deduce that $O_{h_i} \neq \emptyset$ for all $i > \ell$. We have obtained a contradiction. Hence, G is an accepting memoryless $(1, q_0)$-run of $\mathcal{A}$ over w, and we are done.

"$\Rightarrow$" Let $w \in \mathcal{L}(\mathcal{A})$. Hence, by Proposition 1, there is an accepting memoryless $(1, q_0)$-run $G = \langle V, E, v_0 \rangle$ of $\mathcal{A}$ over w. By Lemma 3, for every non-trivial coBüchi stratum $\mathcal{S}$ of $\mathcal{A}$, there is an odd ranking function $f_{\mathcal{S}}$ of $\mathcal{S}$ for the run G. Let Acc be the set of vertices (i, q) of the run G such that (1) either q is an accepting state of a Büchi stratum, or (2) q belongs to a coBüchi stratum $\mathcal{S}$ and, additionally, $f_{\mathcal{S}}(i, q)$ is odd if q is a main state in some non-trivial coBüchi stratum. Since G is accepting and every infinite path of G gets eventually trapped either in a Büchi stratum or a coBüchi stratum, by Definition 7, every infinite path of G visits infinitely many times vertices in Acc. We define an infinite sequence of regions $\nu = (R_1, O_1, f_1), (R_2, O_2, f_2), \ldots$ and show that it is accepting and good with respect to w, which implies the desired result. For all $i \geq 1$, R_i and f_i are defined as follows:

- $R_i \stackrel{\text{def}}{=} \{(i, q) \mid (i, q) \in V \text{ for some } q \in Q\}$;
- for all non-trivial coBüchi strata $\mathcal{S} = \langle \mathsf{C}, P, F \rangle$ and $q \in R_i \cap P \cap \mathsf{M}$, $f_i(q) = f_{\mathcal{S}}(i, q)$.

Let Acc_i be the set of the accepting states of $\mathcal{A}$ with respect to f_i. Note that for all $q \in R_i$, $q \in Acc_i$ iff $(i, q) \in Acc$.

Since G is a run over w, for all $i \geq 1$, there must be a mapping g_i over R_i such that for all $q \in R_i$, $g_i(q)$ is a minimal model of $\delta(q, w(i))$ and the sequence ν satisfies the δ-consistency requirement with respect to g_i. Moreover, since for every non-trivial coBüchi stratum $\mathcal{S}$ of $\mathcal{A}$, $f_{\mathcal{S}}$ is an odd ranking function of $\mathcal{S}$ for the run G, the sequence ν satisfies the ranking requirement with respect to g_i. We need to define the sets O_i and show that the resulting sequence is accepting and satisfies the initialization requirement and the Miyano-Hayashi requirement. For this, we use the following claim.

Claim 3. There is an infinite sequence $1 = h_1 < h_2 < \ldots$ of *dominant* positions of w such that for all $j \geq 1$ and finite paths of G of the form $\pi = (h_j, p), \ldots, (h_{j+1} - 1, q)$, π visits some state in Acc.

First, we show that the result follows from the claim above and then we prove the claim. So, let $1 = h_1 < h_2 < \ldots$ be an infinite sequence of dominant positions of w satisfying the claim above. For every $i \geq 1$, let $j \geq 1$ be the unique integer such that $h_j \leq i < h_{j+1}$. Then, O_i is defined as follows:

- O_i is the set of states q such that there is a finite path of G of the form $\pi = (h_j, p), \ldots, (i, q)$ which does *not* visit vertices in Acc.

Note that $O_i \cap Acc_i = \emptyset$ and $O_i \subseteq R_i$. By construction and the claim above, we have that $O_1 = R_1 \setminus Acc_1$ and for all $j > 1$, $O_{h_j - 1} = \emptyset$ and $O_{h_j} = R_{h_j} \setminus$

Acc_{h_j}. Hence, the infinite sequence of regions $\nu = (R_1, O_1, f_1), (R_2, O_2, f_2), \ldots$ is accepting and satisfies the initialization requirement because $q_0 \in R_1$. For the Miyano-Hayashi requirement with respect to g_i, let $p \in O_i$, $(dir, q, q') \in g_i(p)$ with $dir \in \{\rightarrow, \curvearrowright\}$, and (k, p') be the effect of (dir, q, q') with respect to (w, i) such that $p' \notin Acc_k$, and then $(k, p') \notin Acc$. We need to show that $p' \in O_k$. Let $j \geq 1$ such that $h_j \leq i < h_{j+1}$. Since $O_i \neq \emptyset$ and $O_{h_{j+1}-1} = \emptyset$, we have that $i < h_{j+1} - 1$. Moreover, since h_{j+1} is a dominant position of w, it follows that $k < h_{j+1}$. Hence, since $p \in O_i$ and (k, p') is a successor of (i, p) in G which is not in Acc, we obtain that $p' \in O_k$. Thus, $\nu = (R_1, O_1, f_1), (R_2, O_2, f_2), \ldots$ is an infinite sequence of regions which is good and accepting w.r.t. w. We only need to prove Claim 3.

Proof of Claim 3. Fix $k \geq 1$ such that k is a dominant position of w. Note that the first position of w is dominant. For each $i \geq 1$, let T_i be the set of states $q \in Q$ such that there is a finite path of G of the form $(k, p), \ldots, (i, q)$ which does *not* visit Acc-vertices. Since k is arbitrary, in order to prove Claim 3, it suffices to show that there is a dominant position $m > k$ such that $T_{m-1} = \emptyset$. Let $K = \{(i, q) \in \mathbb{N} \times Q \mid q \in T_i\}$. Note that $K \cap Acc = \emptyset$. First, we prove that the set K is finite. We assume the contrary and derive a contradiction. Let G_K be the sub-graph of G given by the restriction of G to the set of vertices K. Note that by construction, every vertex in G_K is reachable in G_K from a vertex of the form (k, p). Moreover, each vertex of G_K has only finitely many successors. Since G_K is infinite and the set of vertices of the form (k, p) is finite, by König's Lemma, G_K contains an infinite path π. This is a contradiction since π does not visit vertices in Acc and π is also an infinite path of G. Thus, the set $K = \{(i, q) \in \mathbb{N} \times Q \mid q \in T_i\}$ is finite. It follows that there is $j \geq 1$ such that for all $i \geq j$, $T_j = \emptyset$. Since the set of dominant position of w is infinite, we obtain that there is a dominant position $m > k$ such that $T_{m-1} = \emptyset$. Hence, the result follows, which concludes the proof of Claim 3 and Theorem 4 as well. □

4.4 From SAJA to Büchi ω-NVPA

Given a SAJA $\mathcal{A}$ we show how to exploit Remark 1, Definition 8 and Theorem 4 to construct for an equivalent Büchi ω-NVPA whose set of control states and stack symbols range over the set of regions of $\mathcal{A}$. Formally, we show the following result.

Theorem 5 *For a SAJA $\mathcal{A} = \langle \mathsf{M} \cup \mathsf{S}, q_0, \delta, \mathsf{F} \rangle$, one can build in exponential time a Büchi ω-NVPA $\mathcal{P}$ accepting $\mathcal{L}(\mathcal{A})$ with $2^{O(|\mathsf{S}|+|\mathsf{M}|\cdot\log(k))}$ states and stack symbols, where k is the size of the largest non-trivial coBüchi stratum of $\mathcal{A}$ if such a stratum exists, and a fixed constant $c > 1$ otherwise.*

Proof Let $\mathcal{A} = \langle Q, q_0, \delta, \mathsf{F} \rangle$ be a SAJA over Σ, where $Q = \mathsf{M} \cup \mathsf{S}$. By Remark 1, we can assume that $\mathcal{A}$ is pure. We construct a Büchi ω-NVPA $\mathcal{P} = \langle P, P_0, \Gamma, \Delta, F \rangle$ such that that given an input word $w \in \Sigma^\omega$, $\mathcal{P}$ accepts w if and

only if there is an infinite sequence of regions of $\mathcal{A}$ which is good and accepting w.r.t. w. At each step, the automaton keeps track in its control state of the guessed region associated with the current input position i, the guessed region associated with the previous input position (if $i > 1$), and a flag which is 1 iff i is a dominant position. Whenever i is a call position, then $\mathcal{P}$ either guesses that i is pending along w, or i is a matched call position. In the first case, $\mathcal{P}$ pushes on the stack the special symbol `pending`; the guess is correct iff the symbol `pending` in never popped from the stack. In the second case, $\mathcal{P}$ pushes on the stack the current region, the current flag ℓ, and the guessed region $\mathcal{R}_r$ associated with the matched return position. The guess is correct if and only if the symbol on the stack is eventually popped at a position j whose region is $\mathcal{R}_r$ and such that the flag associated with the next position $j+1$ is ℓ. The Büchi acceptance condition of $\mathcal{P}$ ensures that the symbol pushed on the stack is eventually popped. Note that the automaton can check locally, using its transition relation, whether the infinite sequence of regions guessed satisfies the δ-consistency requirement, the ranking requirement, and the Miyano-Hayashi requirement. Finally, the Büchi acceptance condition of $\mathcal{P}$ is also used to check that the guessed sequence of regions is accepting.

In order to simplify the formal definition of $\mathcal{P}$, we introduce additional notation. For a region $\mathcal{R} = (R, O, f)$ and $\sigma \in \Sigma$, a $(\mathcal{R}, \sigma)$-*model* is a mapping assigning to each $q \in R$, a minimal model of $\delta(q, \sigma)$. Now, we give a notion for representing the effects of moves (dir, q, q') of the SAJA $\mathcal{A}$ at the current input symbol σ. We use the term 'left' for representing that the automaton moves to state q, and 'right' for a move to state q'. Given a direction $dir \in \{\varepsilon, \rightarrow, \leftarrow, \curvearrowright, \curvearrowleft\}$, two regions $\mathcal{R} = (R, O, f)$ and $\mathcal{R}_{dir} = (R_{dir}, O_{dir}, f_{dir})$, and a $(\mathcal{R}, \sigma)$-model g for some $\sigma \in \Sigma$, we say that $\mathcal{R}$ *is left (resp., right) dir-consistent w.r.t. g and $\mathcal{R}_{dir}$* if the following holds:

- For all $p \in R$ and $(dir, q, q') \in g(p)$, let $p' = q$ (resp., $p' = q'$). Then, $p' \in R_{dir}$ (*δ-consistency requirement*). Moreover, if p and p' are in the same non-trivial coBüchi stratum, then $f_{dir}(p') \leq f(p)$ (*Ranking requirement*).
- *Miyano-Hayashi requirement.* For all $p \in O$ and $(dir, q, q') \in g(p)$, let $p' = q$ (resp., $p' = q'$). If $dir \in \{\rightarrow, \curvearrowright\}$ and p' is *not* accepting w.r.t. f_{dir}, then $p' \in O_{dir}$.

Additionally, if $dir \in \{\leftarrow, \curvearrowleft\}$, we say that $\mathcal{R}$ *is initially dir-consistent w.r.t. g* if for all $p \in R$ and $(dir, q, q') \in g(p)$, q' is in the F-component of some negative stratum $\langle -, P, F\rangle$ of $\mathcal{A}$.

In the following, an internal transition $(p, \Box, p')$ of a Büchi ω-NVPA is denoted by $p \xrightarrow{\Box} p'$, a push transition (p, c, p', γ) is denoted by $p \xrightarrow{c, push(\gamma)} p'$, and a pop transition (p, r, γ, p') is denoted by $p \xrightarrow{r, pop(\gamma)} p'$. Formally, the Büchi NVPA $\mathcal{P} = \langle P, P_0, \Gamma, \Delta, F\rangle$ is defined as follows:

- $P = \Gamma = \{\texttt{pending}\} \cup (REG \cup \{\emptyset\}) \times REG \times \{0, 1\}$, where REG is the set of regions of $\mathcal{A}$.
- P_0 is the set of states of the form $(\emptyset, (R, O, f), 1)$ such that $q_0 \in R$ and $O = R \setminus Acc$, where Acc is the set of accepting states of $\mathcal{A}$ w.r.t. f.

– F consists of the states of the form $((R_-, \emptyset, f_-), (R, O, f), 1)$ such that $O = R \setminus Acc$, where Acc is the set of accepting states of $\mathcal{A}$ w.r.t. f.

Finally, the transition relation Δ consists of the following transitions.

Internal transitions: $(\mathcal{R}_-, \mathcal{R}, \ell) \xrightarrow{\Box} (\mathcal{R}, \mathcal{R}_+, \ell')$ such that $\ell' = \ell$ and there is a $(\mathcal{R}, \Box)$-model g so that

– $\mathcal{R}$ is left ε-consistent with respect to g and $\mathcal{R}$, and $\mathcal{R}$ is left $\rightarrow$-consistent and right $\curvearrowright$-consistent with respect to g and $\mathcal{R}_+$.
– If $\mathcal{R}_- \neq \emptyset$, then $\mathcal{R}$ is left $\leftarrow$-consistent and right $\curvearrowleft$-consistent with respect to g and $\mathcal{R}_-$. Otherwise, for all $dir \in \{\leftarrow, \curvearrowleft\}$, $\mathcal{R}$ is initially dir-consistent with respect to g.

Push transitions: $(\mathcal{R}_-, \mathcal{R}, \ell) \xrightarrow{c, push(\gamma)} (\mathcal{R}, \mathcal{R}_+, \ell')$ such that there is a $(\mathcal{R}, c)$-model g so that

– $\mathcal{R}$ is left ε-consistent with respect to g and $\mathcal{R}$, and $\mathcal{R}$ is left $\rightarrow$-consistent with respect to g and $\mathcal{R}_+$.
– If $\mathcal{R}_- \neq \emptyset$, then $\mathcal{R}$ is left $\leftarrow$-consistent and right $\curvearrowleft$-consistent with respect to g and $\mathcal{R}_-$. Otherwise, for all $dir \in \{\leftarrow, \curvearrowleft\}$, $\mathcal{R}$ is initially dir-consistent with respect to g.
– If $\gamma = \texttt{pending}$, then $\ell = \ell' = 1$ and $\mathcal{R}$ is right $\curvearrowright$-consistent with respect to g and $\mathcal{R}_+$. Otherwise, $\ell' = 0$, γ is of the form $(\mathcal{R}, \mathcal{R}_r, \ell)$ for some region $\mathcal{R}_r$, and $\mathcal{R}$ is left $\curvearrowright$-consistent with respect to g and $\mathcal{R}_r$.

Pop transitions: $(\mathcal{R}_-, \mathcal{R}, \ell) \xrightarrow{r, pop(\gamma)} (\mathcal{R}, \mathcal{R}_+, \ell')$ such that there is a $(\mathcal{R}, r)$-model g so that

– $\mathcal{R}$ is left ε-consistent with respect to g and $\mathcal{R}$, and $\mathcal{R}$ is left $\rightarrow$-consistent and right $\curvearrowright$-consistent with respect to g and $\mathcal{R}_+$.
– If $\mathcal{R}_- \neq \emptyset$, then $\mathcal{R}$ is left $\leftarrow$-consistent with respect to g and $\mathcal{R}_-$. Otherwise, for all $dir \in \{\leftarrow, \curvearrowleft\}$, $\mathcal{R}$ is initially dir-consistent with respect to g.
– If $\gamma = \bot$, then $\ell = \ell' = 1$ and $\mathcal{R}$ is right $\curvearrowleft$-consistent with respect to g and $\mathcal{R}_-$ if $\mathcal{R}_- \neq \emptyset$.
– If $\gamma \neq \bot$, then $\ell = 0$, γ is of the form $(\mathcal{R}_c, \mathcal{R}, \ell')$ for some region $\mathcal{R}_c$, and $\mathcal{R}$ is left $\curvearrowleft$-consistent with respect to g and $\mathcal{R}_c$.

By construction, $w \in \mathcal{L}(\mathcal{P})$ *iff* there is a run of $\mathcal{P}$ over w of the form $((\emptyset, \mathcal{R}_1, \ell_1), \beta_1) \xrightarrow{w(1)} ((\mathcal{R}_1, \mathcal{R}_2, \ell_2), \beta_2) \ldots \xrightarrow{w(n)} ((\mathcal{R}_n, \mathcal{R}_{n+1}, \ell_{n+1}), \beta_{n+1}) \ldots$ such that the following hold:

– $\mathcal{R}_1, \mathcal{R}_2, \ldots$ is a infinite sequence of regions which is good with respect to w;
– for all $i \geq 1$, $\ell_i = 1$ iff i is a dominant position of w;
– for infinitely many $j \geq 1$, $\mathcal{R}_j = (R_j, \emptyset, f_j)$, $\ell_{j+1} = 1$, and $\mathcal{R}_{j+1} = (R_{j+1}, R_{j+1} \setminus Acc_{j+1}, f_{j+1})$, where Acc_{j+1} is the set of accepting states of $\mathcal{A}$ with respect f_{j+1}.

By Theorem 4, it follows that $\mathcal{L}(\mathcal{P}) = \mathcal{L}(\mathcal{A})$. Moreover, the number of regions is at most $2^{2|Q|} \cdot 2^{|\mathsf{M}| \cdot \log(2k)}$, where k is the size of the largest non-trivial coBüchi stratum of $\mathcal{A}$ if such a stratum exists, and the constant 1 otherwise. Hence, Theorem 5 follows. □

5 Translation of VRE in subclasses of AJA on finite words

In the translation of VLTL formulas into SAJA, we use two subclasses of AJA over finite words (for which we give different acceptance notions). These two classes are introduced to handle the VRE associated with the future and past temporal operators. The approach proposed here substantially differs from the alternating automata-theoretic approach for RLTL, which is crucially based on the use of *non-deterministic* automata for handling the regular expressions of the temporal modalities[4].

The rest of this section is organized as follows. First, we introduce two basic classes of AJA over finite words, namely, *forward and backward AJA with main states* (Definition 9). Then, we introduce in Definition 10 some semantic constraints on the runs of the above two classes of AJA which are crucial for allowing a correct translation of VLTL formulas into equivalent SAJA. We show that these semantic requirements can be syntactically captured by defining two suitable subclasses of forward and backward AJA, that we call *forward and backward MAJA* (Definition 11). We establish that VRE can be translated in polynomial time into equivalent forward and backward MAJA. Moreover, in Subsection 5.1, we demonstrate that for the restricted class of well-formed VRE, the above translation can be done compositionally and the resulting MAJA has a number of states which is linear in the size of the VRE.

Remark 3 Without loss of generality, for the runs of two-way AJT over finite words, we restrict ourselves to *memoryless* runs.

Definition 9 (Forward and backward AJA with main states) A *forward* AJA *with main states* is an AJT with main states $\mathcal{A} = \langle \mathsf{M} \cup \mathsf{S}, q_0, \delta, Acc \rangle$ augmented with a set Acc of accepting states such that:
- only moves (dir, q, q') with $dir \in \{\rightarrow, \curvearrowright, \curvearrowright\!\!\bullet\}$ are allowed;
- $\delta(q, \sigma) = \texttt{false}$ for all accepting main states q and $\sigma \in \Sigma$.

Similarly, a *backward* AJA with main states is such that:
- only moves (dir, q, q') with $dir \in \{\varepsilon, \leftarrow, \curvearrowleft, \curvearrowleft\!\!\bullet\}$ are allowed;
- $\delta(q, \sigma) = \texttt{false}$ for all accepting main states q and $\sigma \in \Sigma$.

If $\mathcal{A}$ is forward, then a run of $\mathcal{A}$ over a finite word w is *accepting* if for all vertices of the form $(|w| + 1, q)$, $q \in Acc$. Similarly, if $\mathcal{A}$ is backward, then a run is accepting if for all vertices of the form $(0, q)$, $q \in Acc$. The language $\mathcal{L}(\mathcal{A})$ of $\mathcal{A}$ is the set of *non-empty finite* words w on Σ such that there is an accepting $(1, q_0)$-run on w (or an accepting $(|w|, q_0)$-run if $\mathcal{A}$ is backward).

[4] AJA are strictly more expressive than their non-deterministic counterpart [10].

A *pseudo* run of $\mathcal{A}$ is defined as a run of $\mathcal{A}$ but for all accepting main states q and $\sigma \in \Sigma$, we replace the value `false` of $\delta(q, \sigma)$ with `true`. The notion of pseudo run is introduced for converting a run of a AJA $\mathcal{A}$ over a subword $w[i, j]$ of an infinite word w in a pseudo run of $\mathcal{A}$ over w. In order to correctly handle the VRE expressions in the translation of VLTL formulas into SAJA, we need to impose additional restrictions on the above two classes of AJA (which intuitively allow to simulate the behavior of non-deterministic automata), ensuring at the same time that these restrictions still allow to capture VRE efficiently. These restrictions in their *semantic form* are the following ones.

Definition 10 (Semantic constraints on AJA with main states)

J1. In each (pseudo) run starting from a main state, there is exactly one maximal path (the *main path*) from the initial vertex which visits only main states. Moreover, each vertex of the run which is not visited by the main path is associated with a secondary state.

J2. In a pseudo run over an infinite word, if the main path ends at a vertex (j, q), then either $j = 0$ or q is accepting.

J3. In a pseudo run over an infinite word, if the main path visits an accepting state, then there is no infinite path from a secondary vertex.

J4. Let the given AJA $\mathcal{A}$ be forward (resp., backward). Then, for all infinite words w on Σ and $1 \leq i \leq j$, $w[i, j] \in \mathcal{L}(\mathcal{A})$ *iff* there is a pseudo (i, q_0)-run (resp., pseudo (j, q_0)-run) of $\mathcal{A}$ over the infinite word w whose main path visits position $j + 1$ (resp., $i - 1$) in an accepting main state, the latter being obtained by a local move.

Intuitively, the main path simulates the unique path of a run in a non-deterministic automaton. The semantic requirements J2–J4 allow to deal with the sequencing and power operators in the translation of VLTL formulas into SAJA. Now, we show that requirements J1–J4 can be *syntactically* captured. In particular, these syntactical constraints also ensure that in a (pseudo) run, the secondary vertices are associated with positions inside minimally well-matched subwords of the input word. The forward AJA with main states satisfying these syntactical requirements are called *forward* AJA *with main paths* (MAJA). Backward MAJA is defined similarly, and we simply use MAJA when the direction is clear from the context.

Definition 11 (Forward and backward MAJA) A *forward* MAJA is a tuple $\mathcal{A} = \langle \mathsf{M} \cup \mathsf{S} \cup \{\bot_\mathsf{M}, \bot_\mathsf{S}\}, q_0, \delta, Acc\rangle$ such that $\langle \mathsf{M} \cup \mathsf{S} \cup \{\bot_\mathsf{M}, \bot_\mathsf{S}\}, q_0, \delta\rangle$ is an AJT with main states containing a special main state $\bot_\mathsf{M} \notin \mathsf{M}$ and a special secondary state $\bot_\mathsf{S} \notin \mathsf{S}$. Moreover, Acc is a set of accepting states, and for all states q and $\sigma \in \Sigma$, the following holds:

S1. $q \in \mathsf{M}$: if $\sigma \notin \Sigma_{call}$, then $\delta(q, \sigma)$ is a disjunction of moves of the form $(\rightarrow, p, p)$ where $p \in \mathsf{M}$; otherwise, $\delta(q, \sigma)$ is a disjunction of conjuncts θ, where *either* $\theta = (dir, p, \bot_\mathsf{M}) \wedge (\rightarrow, p', p')$ for some $dir \in \{\curvearrowright, \curvearrowright\!\!\bullet\}$, $p \in \mathsf{M} \setminus Acc$ and $p' \in \mathsf{S}$, *or* $\theta = (\rightarrow, p, p)$ for some $p \in \mathsf{M}$. Moreover, if $q \in Acc$, then $\delta(q, \sigma) = \mathtt{false}$.

S2. $q \in \mathsf{S}$: (i) if $\sigma \in \Sigma_{call}$, then $\delta(q,\sigma)$ is a disjunction of conjuncts of the form $(\curvearrowright_\bullet, p, \bot_\mathsf{S}) \wedge (\rightarrow, p', p')$, where $p, p' \in \mathsf{S}$, (ii) if $\sigma \in \Sigma_{int}$, then $\delta(q,\sigma)$ is a disjunction of moves of the form $(\rightarrow, p, p)$ where $p \in \mathsf{S}$, and (iii) if $\sigma \in \Sigma_{ret}$, then $\delta(q,\sigma) \in \{\mathtt{true}, \mathtt{false}\}$.
S3. $q \in \{\bot_\mathsf{M}, \bot_\mathsf{S}\}$: $\delta(q,\sigma) = \mathtt{false}$.
S4. $\mathsf{S} \cup \{\bot_\mathsf{S}\} \subseteq Acc$, $q_0 \in \mathsf{M} \setminus Acc$, $\bot_\mathsf{M} \notin Acc$, and no moves lead to q_0.

Note that the empty disjunction is `false`. A *backward* MAJA is defined similarly but we switch calls and returns, $\rightarrow$-moves for $\leftarrow$-moves, $\curvearrowright$-moves for $\curvearrowleft$-moves, and $\curvearrowright_\bullet$-moves for $\curvearrowleft_\bullet$-moves.

Note that MAJA correspond to a syntactical subclass of AJA with main states. Conditions S1 and S2 in Definition 11 (and their variants for backward MAJA) ensure that in each (pseudo) run of a MAJA $\mathcal{A}$ starting from a main state, there is exactly one maximal path π (the *main path*) from the initial vertex which visits only vertices associated with main states. Moreover, each vertex of the run which is not visited by the main path is associated with a secondary state. Hence, the fulfillment of requirement J1 of Definition 10 follows. By Definition 11, for all main states q and input symbols σ, $\delta(q,\sigma) \neq \mathtt{true}$. Therefore, by J1 and definition of pseudo run, the fulfillment of the semantic requirement J2 follows.

With regard to the semantic requirements J3-J4, fix a forward MAJA $\mathcal{A}$ with initial state q_0 (the backward case is similar). Requirement S4 in Definition 11 ensures that the empty string ε is not accepted by $\mathcal{A}$, and the fulfillment of the acceptance condition depends only on the main path. Hence, a run of $\mathcal{A}$ over a finite word is accepting iff the main path visits an accepting state. Now, let us consider a (pseudo) run G of the forward MAJA $\mathcal{A}$ over a finite or infinite word w starting from a main vertex. Assume that the main path of G visits an accepting main state, which also implies that the main path ends at an accepting vertex. Then, conditions S1–S4 in Definition 11 ensure that for every path π of G from a *secondary* vertex (i,p), the following holds: $p \neq \bot_\mathsf{S}$ and the *next unmatched return* position j along the suffix of w from i is always defined, and π is finite and never visits positions after j. In particular, there are two *non-accepting main* vertices (j_c, q_c) and (k, q_r) such that j_c is a matched call position with matching return j_r, $k \in \{j_r, j_r+1\}$ and $j_c \leq i \leq j_r$. Hence, the semantic requirement J3 holds. Moreover, it follows that for all infinite words w and positions $1 \leq i \leq j$, there is a $(1,q_0)$-run of $\mathcal{A}$ over the finite word $w[i,j]$ whose main path visits an accepting vertex *iff* there is a pseudo (i,q_0)-run of $\mathcal{A}$ over the infinite word w whose main path visits position $j+1$ in an accepting main state (note that Definition 11 ensures that this accepting main state is obtained by a local move). Thus, since a run of $\mathcal{A}$ over a finite word is accepting iff the main path visits an accepting state, the fulfillment of the semantic requirement J4 follows. Consequently, we obtain the following result.

Proposition 2 *MAJA satisfy the semantic requirements* J1–J4 *of Definition 10.*

Remark 4 MAJA with no secondary states correspond to standard finite-state non-deterministic automata.

By an adaptation of known results, we show that in the general case, VRE can be translated in polynomial time into equivalent MAJA.

Theorem 6 (From unrestricted VRE to MAJA) *Given a VRE α, one can build in polynomial time a forward (resp., backward) MAJA $\mathcal{A}$ with $O(|\alpha|)$ main states and $O(|\alpha|^4)$ secondary states such that $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\alpha) \setminus \{\varepsilon\}$.*

Proof First, we introduce the notion of *reverse* NVPA. A reverse NVPA is defined as an NVPA with the difference that the automaton pushes onto the stack on reading returns, pops the stack on reading calls, and does not use the stack on internal actions. Formally, a reverse NVPA is a tuple $\langle Q, Q_0, \Gamma, \Delta, F\rangle$, where Q, Q_0, Γ and F are defined as for NVPA and $\Delta \subseteq (Q \times \Sigma_{ret} \times Q \times \Gamma) \cup (Q \times \Sigma_{call} \times (\Gamma \cup \{\bot\}) \times Q) \cup (Q \times \Sigma_{int} \times Q)$. Essentially, a reverse NVPA over a pushdown alphabet Σ corresponds to an ordinary NVPA operating over the pushdown alphabet obtained from Σ by switching call and returns. The quadratic-time translation from VRE to NVPA given in [11,12] can be easily adapted for obtaining a quadratic-time translation from VRE α to reverse NVPA accepting the reverse of $\mathcal{L}(\alpha)$. Hence, the following holds:

Claim 4. Given a VRE α, one can build in quadratic time an NVPA $\mathcal{P}$ and a reverse NVPA $\mathcal{P}_r$ accepting $\mathcal{L}(\alpha)$ and the reverse of $\mathcal{L}(\alpha)$, respectively, with $O(|\alpha|)$ states and $O(|\alpha|^2)$ stack symbols.

By Theorem 4 in [10], given an NVPA $\mathcal{P}$ with n states and s stack symbols, one can construct an equivalent one-way AJA $\mathcal{A}$ over finite words with $O(n^2 s)$ states. The constructed AJA $\mathcal{A}$ in [10] is a variant of a forward MAJA, where $\curvearrowright$-moves are not allowed. It is easy to convert in linear-time $\mathcal{A}$ into a forward MAJA accepting $\mathcal{L}(\mathcal{P}) \setminus \{\varepsilon\}$ with $O(n)$ main states and $O(n^2 s)$ secondary states. The same construction can be used to convert a reverse NVPA $\mathcal{P}_r$ with n states and s stack symbols into a backward MAJA with $O(n)$ main states and $O(n^2 s)$ secondary states, accepting the reverse of $\mathcal{L}(\mathcal{P}_r) \setminus \{\varepsilon\}$ by just switching calls and returns, and forward directions with the corresponding backward directions. By the claim above, the theorem follows. □

5.1 Compositional translation of well-formed VRE into MAJA

In this section, we show that well-formed VRE can be compositionally translated in polynomial time into equivalent forward and backward MAJA having a number of states which is linear in the size of the given VRE. Our approach exploits an additional syntactical subclass of MAJA that captures more efficiently the restricted class of *well-matched* VRE. The additional syntactical constraints are used to implement in an efficient way M-substitution and S-closure in well-matched VRE. Note that thanks to the fulfillment of the semantic requirements J1–J4 of Definition 10, the concatenation and the Kleene

closure operators can be handled in a way analogous to the standard translation of regular expressions in non-deterministic automata.

For a MAJA $\mathcal{A}$ with transition function δ, we say that a main state q *does not distinguish return symbols* if whenever there is a return $r \in \Sigma_{ret}$ such that $\delta(q, r)$ contains occurrences of accepting main states, then all returns $r' \in \Sigma_{ret}$ $\delta(q, r')$ contain occurrences of accepting main states as well. Similarly, we say that a main state q *does not distinguish call symbols*, if whenever there is a call $c \in \Sigma_{call}$ such that $\delta(q, c)$ contains occurrences of accepting main states, then all calls $c' \in \Sigma_{call}$, $\delta(q, c')$ contain occurrences of accepting main states as well.

Definition 12 A *matched-call forward* MAJA $\mathcal{A} = \langle \mathsf{M} \cup \mathsf{S} \cup \{\bot_\mathsf{M}, \bot_\mathsf{S}\}, q_0, \delta, Acc\rangle$ is a forward MAJA satisfying the following additional syntactical constraints for all $c \in \Sigma_{call}$:

1. for all $q \in \mathsf{M}$, $\delta(q, c)$ does not contain atoms of the form $(\rightarrow, p, p)$ with $p \in \mathsf{M}$;
2. for each atom $(\curvearrowright, p, \bot_\mathsf{M})$ in $\delta(q_0, c)$, the main state p does not distinguish *return* symbols.

A *matched-return backward* MAJA is a backward MAJA which satisfies similar constraints but switching calls and returns, switching $\rightarrow$-moves with $\leftarrow$-moves, and switching $\curvearrowright$-moves with $\curvearrowleft$-moves.

By Definitions 11 and 12, a matched-call forward MAJA accepts only words with no pending calls, and a matched-call backward MAJA accepts only words with no pending returns. The additional syntactical constraint given by Definition 12(2) (and its version for matched-return backward MAJA) is used to implement in an efficient way M-substitution and S-closure in well-matched VRE. The polynomial-time translation of well-formed VRE into MAJA is based on the following two Lemmata 5 and 7, which show that the considered subclass of MAJA is *efficiently* closed under union, concatenation, Kleene closure, M-substitution, and S-closure. By "efficiently" we mean that for each operation, the construction outputs a MAJA whose size is linear in the sizes of the input MAJA. In order to obtain these results, we first make some observations on the behavior of MAJA which easily follow from Definition 11. Note that the use of $\curvearrowright_\bullet$-moves instead of $\curvearrowright$-moves from secondary states associated with call positions in forward MAJA (and similarly for backward MAJA), as established by requirement S2 in Definition 11, is crucial for ensuring Property 2 of the following Lemma 4. On the other hand, Property 2 of Lemma 4 allows us to apply simple constructions (see Lemma 7) for showing that forward and backward MAJA are efficiently closed under M-substitution and S-closure.

Lemma 4 *Let $\mathcal{A}$ be a forward or backward MAJA and G be an accepting (pseudo) (i, q)-run of $\mathcal{A}$ over a finite word w, where q is a main state and $i \in Pos(w)$. Then, the following holds.*

1. *For all G-vertices $v_m = (j_m, p_m)$ and $v_s = (j_s, p_s)$ such that p_m is a main state, p_s is a secondary state, and v_s is reachable from v_m in G, it holds that $p_s \neq \bot_\mathsf{S}$ and:*

- *If $\mathcal{A}$ is forward then j_m is a matched-call in w with matching return j_r and $j_m < j_s \leq j_r$. Moreover, v_m has* either *a main successor of the form (j_r, p_r), which is obtained by applying a move of the form $(\curvearrowright, p_r, \bot_{\mathsf{M}})$,* or *a main successor of the form $(j_r{+}1, p_r)$, which is obtained by applying a move of the form $(\curvearrowright_{\bullet}, p_r, \bot_{\mathsf{M}})$.*
- *If $\mathcal{A}$ is backward then j_m is a matched-return in w with matching call j_c and $j_c \leq j_s < j_m$. Moreover, v_m has* either *a main successor of the form (j_c, p_c), which is obtained by applying a move of the form $(\curvearrowleft, p_c, \bot_{\mathsf{M}})$,* or *a main successor of the form $(j_c{-}1, p_c)$, which is obtained by applying a move of the form $(\curvearrowleft_{\bullet}, p_c, \bot_{\mathsf{M}})$.*

2. *For every position j along w, it holds that*
   - *$\mathcal{A}$ is forward: if j is not a return position, then there is at most one G-vertex associated with j. If, instead, j is a return position, there is at most one main G-vertex and at most one secondary G-vertex associated with j. Moreover, if $i \leq j$ and there is a G-vertex associated with a position $k \geq j$, then there is a G-vertex associated with position j.*
   - *$\mathcal{A}$ is backward: if j is not a call position, then there is at most one G-vertex associated with j. If, instead, j is a call position, there is at most one main G-vertex and at most one secondary G-vertex associated with position j. Moreover, if $j \leq i$ and there is a G-vertex associated with a position $k \leq j$, then there is a G-vertex associated with position j.*

We first show that MAJA are closed under union, concatenation, and Kleene closure. Moreover, as for the class of non-deterministic finite-state automata, the constructions just require the use of an additional state.

Lemma 5 *Let $\mathcal{A}$ and $\mathcal{A}'$ be two forward (resp., backward) MAJA with k and k' states respectively. Then, one can construct in linear time*
- Union: *a forward (resp., backward) MAJA accepting $\mathcal{L}(\mathcal{A}) \cup \mathcal{L}(\mathcal{A}')$ with at most $k + k' + 1$ states;*
- Kleene closure: *a forward (resp., backward) MAJA accepting $[\mathcal{L}(\mathcal{A})]^* \setminus \{\varepsilon\}$ with $k + 1$ states;*
- Concatenation: *three forward (resp., backward) MAJA accepting $\mathcal{L}(\mathcal{A}) \cdot \mathcal{L}(\mathcal{A}')$, $(\mathcal{L}(\mathcal{A}) \cup \{\varepsilon\}) \cdot \mathcal{L}(\mathcal{A}')$, and $\mathcal{L}(\mathcal{A}) \cdot (\mathcal{L}(\mathcal{A}') \cup \{\varepsilon\})$, respectively, with at most $k + k' + 1$ states.*

Moreover, the constructed MAJA are matched-call (resp., matched-return) if $\mathcal{A}$ and $\mathcal{A}'$ are matched-call (resp., matched-return).

Proof We illustrate the constructions one by one.

Construction for Union. The construction for forward MAJA, which holds for backward MAJA as well, is given in Fig. 1, where $q_{\cup}^0$ is a fresh main state. Since $\mathcal{A}$ and $\mathcal{A}'$ are forward MAJA, by construction, $\mathcal{A}_{\cup}$ is a forward MAJA accepting $\mathcal{L}(\mathcal{A}) \cup \mathcal{L}(\mathcal{A}')$, which is matched-call if $\mathcal{A}$ and $\mathcal{A}'$ are matched-call. The construction for backward MAJA is analogous.

Construction for Kleene closure. For forward MAJA, the construction is illustrated in Fig. 1, where q_*^0 is a fresh main state. For backward MAJA, we simply

$$\mathcal{A} = \langle \mathsf{M} \cup \mathsf{S} \cup \{\bot_\mathsf{M}, \bot_\mathsf{S}\}, q_0, \delta, Acc\rangle \quad \mathcal{A}' = \langle \mathsf{M}' \cup \mathsf{S}' \cup \{\bot_\mathsf{M}, \bot_\mathsf{S}\}, q_0', \delta', Acc'\rangle$$
$$(\mathsf{M} \cup \mathsf{S}) \cap (\mathsf{M}' \cup \mathsf{S}') = \emptyset$$

Union: $\mathcal{A}_\cup = \langle \mathsf{M} \cup \mathsf{S} \cup \mathsf{M}' \cup \mathsf{S}' \cup \{\bot_\mathsf{M}, \bot_\mathsf{S}\} \cup \{q_\cup^0\}, q_\cup^0, \delta_\cup, Acc \cup Acc'\rangle$

$$\delta_\cup(q,\sigma) = \begin{cases} \delta(q_0,\sigma) \vee \delta'(q_0',\sigma) & \text{if } q = q_\cup^0 \\ \delta(q,\sigma) & \text{if } q \in \mathsf{M} \cup \mathsf{S} \\ \delta'(q,\sigma) & \text{if } q \in \mathsf{M}' \cup \mathsf{S}' \\ \texttt{false} & \text{if } q \in \{\bot_\mathsf{M}, \bot_\mathsf{S}\} \end{cases}$$

Kleene closure: $\mathcal{A}_* = \langle \mathsf{M} \cup \mathsf{S} \cup \{\bot_\mathsf{M}, \bot_\mathsf{S}, q_*^0\}, q_*^0, \delta_*, Acc\rangle$

$$\delta_*(q,\sigma) = \begin{cases} \mathit{Update}(\delta(q_0,\sigma)) & \text{if } q = q_*^0 \\ \mathit{Update}(\delta(q,\sigma)) & \text{if } q \in \mathsf{M} \cup \{\bot_\mathsf{M}\} \\ \delta(q,\sigma) & \text{if } q \in \mathsf{S} \cup \{\bot_\mathsf{S}\} \end{cases}$$

$\mathit{Update}(\delta(q,\sigma)) \stackrel{\text{def}}{=} \delta(q,\sigma)$ if there is no atom $(\rightarrow, q_{acc}, q_{acc})$ in $\delta(q,\sigma)$ with $q_{acc} \in \mathsf{M} \cap Acc$;
otherwise, $\mathit{Update}(\delta(q,\sigma)) \stackrel{\text{def}}{=} \delta(q,\sigma) \vee (\rightarrow, q_0, q_0)$

Concatenation: $\mathcal{A} \cdot \mathcal{A}' = \langle \mathsf{M} \cup \mathsf{S} \cup \mathsf{M}' \cup \mathsf{S}' \cup \{\bot_\mathsf{M}, \bot_\mathsf{S}\}, q_0, \delta \cdot \delta', Acc' \cup \mathsf{S}\rangle$

$$\delta \cdot \delta'(q,\sigma) = \begin{cases} \mathit{Update}(\delta(q,\sigma)) & \text{if } q \in \mathsf{M} \\ \delta(q,\sigma) & \text{if } q \in \mathsf{S} \\ \delta'(q,\sigma) & \text{if } q \in \mathsf{M}' \cup \mathsf{S}' \\ \texttt{false} & \text{if } q \in \{\bot_\mathsf{M}, \bot_\mathsf{S}\} \end{cases}$$

$\mathit{Update}(\delta(q,\sigma)) \stackrel{\text{def}}{=} \delta(q,\sigma)$ if there is no atom $(\rightarrow, q_{acc}, q_{acc})$ in $\delta(q,\sigma)$ with $q_{acc} \in \mathsf{M} \cap Acc$;
otherwise, $\mathit{Update}(\delta(q,\sigma)) \stackrel{\text{def}}{=} \delta(q,\sigma) \vee (\rightarrow, q_0', q_0')$

Fig. 1 Constructions for union, Kleene closure, and concatenation of forward MAJA.

replace the $\rightarrow$-direction with the $\leftarrow$-direction. Note that if the MAJA $\mathcal{A}$ has no secondary states, then $\mathcal{A}$ is a non-deterministic finite-state automaton and the construction corresponds to the classical one used for regular languages.

By construction, $\mathcal{A}_*$ is a forward MAJA if $\mathcal{A}$ is forward, and backward if $\mathcal{A}$ is backward. We describe the proof for forward MAJA (the proof for backward MAJA is analogous). By Definition 12(1), for all calls c and main states q, $\mathit{Update}(\delta(q,c)) = \delta(q,c)$ if $\mathcal{A}$ is matched-call. By Definition 12, it follows that $\mathcal{A}_*$ is also matched-call if $\mathcal{A}$ is matched-call It remains to show that $\mathcal{L}(\mathcal{A}_*) = [\mathcal{L}(\mathcal{A})]^* \setminus \{\varepsilon\}$.

Inclusion $\mathcal{L}(\mathcal{A}_*) \subseteq [\mathcal{L}(\mathcal{A})]^* \setminus \{\varepsilon\}$. Let $w \in \mathcal{L}(\mathcal{A}_*)$. There is an accepting $(1, q_0)$-run G_* of $\mathcal{A}_*$ over the non-empty finite word w. Let $N(G_*)$ be the number of main vertices of G_* which are obtained by applying the unique new move $(\rightarrow, q_0, q_0)$. We prove by induction on $N(G_*)$ that $w \in [\mathcal{L}(\mathcal{A})]^*$. The base case follows immediately. Assume now that $N(G_*) > 0$. Then, since $q_0 \notin Acc$ (Requirement S4 in Definition 11), by construction, there is a position $1 < i \leq |w|$ such that the main path π of G_* visits a vertex v of the form $v = (i-1, q)$ for some main state q of $\mathcal{A}$ and the vertex $v' = (i, q_0)$. Moreover, v' is the unique successor of v in G_*, v' is the last vertex along the main path obtained

by applying a new move, and $\delta(q, w(i-1))$ contains a singleton conjunct of the form $(\rightarrow, q_{acc}, q_{acc})$ for some accepting main state q_{acc} of $\mathcal{A}$. Let V' be the set of vertices reachable from v', G be the subgraph of G_* obtained by removing all vertices in V' and adding the edge from v to the new vertex (i, q_{acc}), and G' be the subgraph of G_* with initial vertex v' corresponding to the restriction of G_* to V'. Since $N(G') = 0$, G' is an accepting (i, q_0)-run of the forward MAJA $\mathcal{A}$ over w. Hence $w[i, |w|] \in \mathcal{L}(\mathcal{A})$. Moreover, by Lemma 4(1), G is an accepting $(1, q_0)$-run of $\mathcal{A}_*$ over $w[1, i-1]$. Since $N(G) < N(G_*)$, by the induction hypothesis, $w[1, i-1] \in [\mathcal{L}(\mathcal{A})]^*$. Thus, since $w[i, |w|] \in \mathcal{L}(\mathcal{A})$, we obtain that $w \in [\mathcal{L}(\mathcal{A})]^*$, and the result follows.

Converse Inclusion $[\mathcal{L}(\mathcal{A})]^* \setminus \{\varepsilon\} \subseteq \mathcal{L}(\mathcal{A}_*)$. Let $w \in [\mathcal{L}(\mathcal{A})]^* \setminus \{\varepsilon\}$. Consequently, there is $k \geq 1$ such that $w = w_1 \cdot \ldots \cdot w_k$ and $w_i \in \mathcal{L}(\mathcal{A})$ for all $1 \leq i \leq k$. We prove by induction on k that $w \in \mathcal{L}(\mathcal{A}_*)$. The base case $k = 1$ is obvious since by construction, $\mathcal{L}(\mathcal{A}) \subseteq \mathcal{L}(\mathcal{A}_*)$. Now, assume that $k > 1$. By the induction hypothesis, there is an accepting $(1, q_0)$-run G of $\mathcal{A}_*$ over $w' = w_1 \cdot \ldots \cdot w_{k-1}$ and there is an accepting $(1, q_0)$-run G' of $\mathcal{A}_*$ over w_k. Since for a main state q of the MAJA $\mathcal{A}_*$, $\delta_*(q, \sigma)$ is never `true`, the main path π of G must lead to a vertex v of the form $v = (|w'| + 1, q_{acc})$ for some accepting main state q_{acc}. Let v' be the vertex that precedes v along π. Since $\mathcal{A}_*$ is a forward MAJA, by Definition 11, v' is of the form $v' = (|w'|, q)$ and v is obtained from v' by applying the move $(\rightarrow, q_{acc}, q_{acc})$. Moreover, v is the unique successor of v'. Also, since $q_0 \notin Acc$, by construction $(\rightarrow, q_{acc}, q_{acc})$ is a singleton conjunct in $\delta(q, w'(|w'|))$. Let $G'_{+|w'|}$ and G_* be the graphs defined as follows:

- $G'_{+|w'|}$ is obtained from G' by replacing every G'-vertex (i, q) with the vertex $(i + |w'|, q)$;
- G_* is obtained from the merging of G and $G'_{+|w'|}$ by additionally removing the vertex v and by replacing the edge (v', v) with the edge (v', v_0), where $v_0 = (|w'| + 1, q_0)$ is the initial vertex of $G'_{+|w'|}$.

By construction and Lemma 4(1), G_* is an accepting $(1, q_0)$-run of $\mathcal{A}_*$ over $w = w' \cdot w_k$. Hence, the result follows.

Construction for Concatenation. For forward MAJA $\mathcal{A}$ and $\mathcal{A}'$, the construction of the forward MAJA $\mathcal{A} \cdot \mathcal{A}'$ accepting $\mathcal{L}(\mathcal{A}) \cdot \mathcal{L}(\mathcal{A}')$ is illustrated in Fig. 1. For backward MAJA, we consider the construction of $\mathcal{A}' \cdot \mathcal{A}$ for forward MAJA (note that the roles of $\mathcal{A}$ and $\mathcal{A}'$ are switched) and we replace the $\rightarrow$-direction with the $\leftarrow$-direction. Note that if $\mathcal{A}$ and $\mathcal{A}'$ have no secondary states, then $\mathcal{A}$ and $\mathcal{A}'$ are non-deterministic finite-state automata and the construction corresponds to the classical construction used for regular languages. The construction can be proven to be correct by reasoning as for the case of the Kleene closure.

Finally, for the construction of the forward MAJA accepting $(\mathcal{L}(\mathcal{A}) \cup \{\varepsilon\}) \cdot \mathcal{L}(\mathcal{A}')$ and $\mathcal{L}(\mathcal{A}) \cdot (\mathcal{L}(\mathcal{A}') \cup \{\varepsilon\})$, we slightly modify the construction given for the language $\mathcal{L}(\mathcal{A}) \cdot \mathcal{L}(\mathcal{A}')$. Again, the construction of the backward MAJA

is symmetric. The forward MAJA accepting $(\mathcal{L}(\mathcal{A}) \cup \{\varepsilon\}) \cdot \mathcal{L}(\mathcal{A}')$ is obtained from $\mathcal{A} \cdot \mathcal{A}'$ in Fig. 1 by adding extra moves from the initial (main) state q_0 to mimic the initial moves of $\mathcal{A}'$ from q_0'. The forward MAJA accepting $\mathcal{L}(\mathcal{A}) \cdot (\mathcal{L}(\mathcal{A}') \cup \{\varepsilon\})$ is instead obtained from $\mathcal{A} \cdot \mathcal{A}'$ in Fig. 1 by updating the set of accepting states to $Acc \cup Acc'$. This concludes the proof of Lemma 5. □

Next, we show that matched-call forward and matched-return backward MAJA are efficiently closed under M-substitution and S-closure. For this, we need the following preliminar result.

Lemma 6 *Let $\mathcal{A} = \langle \mathsf{M} \cup \mathsf{S} \cup \{\bot_\mathsf{M}, \bot_\mathsf{S}\}, q_0, \delta, Acc\rangle$ be a matched-call forward (resp., matched-return backward) MAJA. Then, one can build in linear time a matched-call forward (resp., matched-return backward) MAJA, denoted by $MWM(\mathcal{A})$, of the form $\langle \mathsf{M} \cup \mathsf{S} \cup \{\bot_\mathsf{M}, \bot_\mathsf{S}\}, q_0, \delta', Acc\rangle$ accepting $MWM(\mathcal{L}(\mathcal{A}))$. Moreover, $MWM(\mathcal{A})$ satisfies the following:*

1. *If $MWM(\mathcal{A})$ is matched-call forward: for all calls $c \in \Sigma_{call}$, returns $r \in \Sigma_{ret}$, and main states $q \neq \bot_\mathsf{M}$ occurring in $\delta'(q_0, c)$, $\delta'(q, r)$ contains occurrences of accepting main states.*
2. *If $MWM(\mathcal{A})$ is matched-call backward: for all calls $c \in \Sigma_{call}$, returns $r \in \Sigma_{ret}$, and main states $q \neq \bot_\mathsf{M}$ occurring in $\delta'(q_0, r)$, $\delta'(q, c)$ contains occurrences of accepting main states.*

Proof We assume that $\mathcal{A} = \langle \mathsf{M} \cup \mathsf{S} \cup \{\bot_\mathsf{M}, \bot_\mathsf{S}\}, q_0, \delta, Acc\rangle$ is a matched-call forward MAJA (the other case being similar). Then, $MWM(\mathcal{A}) = \langle \mathsf{M} \cup \mathsf{S} \cup \{\bot_\mathsf{M}, \bot_\mathsf{S}\}, q_0, \delta', Acc\rangle$, where the transition function $\delta'(q, \sigma)$ is defined as follows for all $q \in \mathsf{M} \cup \mathsf{S} \cup \{\bot_\mathsf{M}, \bot_\mathsf{S}\}$ and $\sigma \in \Sigma$:

- $q \in \mathsf{S} \cup \{\bot_\mathsf{M}, \bot_\mathsf{S}\}$: $\delta'(q, \sigma) = \delta(q, \sigma)$;
- $q = q_0$: $\delta'(q_0, \sigma) = \texttt{false}$ if $\sigma \notin \Sigma_{call}$. Otherwise, if $\sigma \in \Sigma_{call}$, then $\delta'(q_0, \sigma)$ is defined as follows. For each conjunct θ occurring in $\delta(q_0, \sigma)$, we remove θ unless $\theta = (\curvearrowright, p, \bot_\mathsf{M}) \wedge (\rightarrow, p', p')$ and for some $r \in \Sigma_{ret}$, $\delta(p, r)$ contains accepting main states (note that $p \neq q_0$).
- $q \in \mathsf{M}$ and $q \neq q_0$: $\delta'(q, \sigma) = \texttt{false}$ if $\sigma \notin \Sigma_{ret}$. Otherwise, if $\sigma \in \Sigma_{ret}$, then $\delta'(q, \sigma)$ is defined as follows. For each conjunct θ occurring in $\delta(q, \sigma)$ (note that θ is of the form $(\rightarrow, p, p)$ for some $p \in M$), we remove θ *iff* the associated main state p is *not* accepting.

By construction, $MWM(\mathcal{A})$ is a matched-call forward MAJA which accepts only minimally well-matched words in $\mathcal{L}(\mathcal{A})$. Moreover, since $\mathcal{A}$ is a matched-call forward MAJA, for an accepting $(1, q_0)$-run over a minimally well-matched word w, the main path visits just the first position of w, the last position of w, and an accepting main vertex associated with position $|w| + 1$. Hence, by construction, every word in $MWM(\mathcal{L}(\mathcal{A}))$ is accepted by $MWM(\mathcal{A})$ as well. Finally, by Definition 12(2), it follows that $MWM(\mathcal{A})$ satisfies Condition 1 in the lemma, which concludes the proof. □

$$MWM(\mathcal{A}) = \langle \mathsf{M} \cup \mathsf{S} \cup \{\bot_\mathsf{M}, \bot_\mathsf{S}\}, q_0, \delta, Acc\rangle$$

For all calls c,
$\Upsilon_\mathcal{A}(c) \stackrel{\text{def}}{=} \{p \in \mathsf{S} \mid \delta(q_0, c)$ contains some conjunct of the form $(\curvearrowright, q, \bot_\mathsf{M}) \wedge (\rightarrow, p, p)\}$

S-closure: $\mathcal{A}^{\circlearrowleft\square} = \langle \mathsf{M} \cup \mathsf{S} \cup \{\bot_\mathsf{M}, \bot_\mathsf{S}\}, q_0, \delta^{\circlearrowleft\square}, Acc\rangle$

$$\delta^{\circlearrowleft\square}(q,\sigma) = \begin{cases} \delta(q,\sigma) & \text{if } q \in \mathsf{M} \cup \{\bot_\mathsf{M}, \bot_\mathsf{S}\} \\ \delta(q,\sigma) & \text{if } q \in \mathsf{S} \text{ and } \sigma \notin \Sigma_{call} \\ Update(\delta(q,\sigma)) & \text{otherwise} \end{cases}$$

For all states $q \in \mathsf{S}$ and calls c,
$Update(\delta(q,c)) \stackrel{\text{def}}{=}$ for all $(\rightarrow, p, p)$-moves from q on reading $\square$ – note that $p \in \mathsf{S}$ – and for all $p' \in \Upsilon_\mathcal{A}(c)$, we add disjunctively in $\delta(q,c)$ the conjunct $(\curvearrowright\!\!\bullet, p, \bot_\mathsf{S}) \wedge (\rightarrow, p', p')$.

Fig. 2 Construction for S-closure of a matched-call forward MAJA $\mathcal{A}$.

Lemma 7 *Let $\square \in \Sigma_{int}$ and $\mathcal{A}$ and $\mathcal{A}'$ be matched-call forward (resp., matched-return backward) MAJA with k and k' states, respectively. Then, one can build in linear time two matched-call forward (resp., matched-return backward) MAJA, one accepting $[\mathcal{L}(\mathcal{A})]^{\circlearrowleft\square}$ with k states, and the other one accepting $\mathcal{L}(\mathcal{A}) \curvearrowright_\square \mathcal{L}(\mathcal{A}')$ with $k + k' + 1$ states.*

Proof We consider the case when $\mathcal{A}$ and $\mathcal{A}'$ are matched-call forward MAJA. The constructions for matched-return backward MAJA are similar by switching calls and returns, and forward directions with the dual backward directions.

Construction for S-closure. Let $MWM(\mathcal{A})$ be the matched-call forward MAJA of Lemma 6 associated with $\mathcal{A}$ and accepting $MWM(\mathcal{L}(\mathcal{A}))$. First, we informally describe the construction of the matched-call forward MAJA $\mathcal{A}^{\circlearrowleft\square}$ accepting $[\mathcal{L}(\mathcal{A})]^{\circlearrowleft\square}$. For each call c, let $\Upsilon_\mathcal{A}(c)$ be as defined in Fig. 2. Essentially, $\mathcal{A}^{\circlearrowleft\square}$ simulates $MWM(\mathcal{A})$ but when a copy of $MWM(\mathcal{A})$ can performs an internal move of the form $(\rightarrow, p, p)$ upon reading $\square$ from the current secondary state q, then the copy of $\mathcal{A}^{\circlearrowleft\square}$ can choose to either process $\square$ as the current copy of $MWM(\mathcal{A})$, or recursively process instead some guessed word $w \in [\mathcal{L}(\mathcal{A})]^{\circlearrowleft\square}$ as follows. The copy of $\mathcal{A}^{\circlearrowleft\square}$ guesses a call c that is the initial symbol of w, chooses a secondary state $p' \in \Upsilon(c)$, and splits into two copies. The first copy moves to the next position in state p' using a $(\rightarrow, p', p')$-move. The second copy jumps to the position following the last position of w in state p by a $(\curvearrowright\!\!\bullet, p, \bot_\mathsf{S})$-move. The first copy ensures recursively that $w \in [\mathcal{L}(\mathcal{A})]^{\circlearrowleft\square}$, while the second copy can restart the simulation of the considered copy of $MWM(\mathcal{A})$ from the desired control state p.

The formal definition of $\mathcal{A}^{\circlearrowleft\square}$ is given in Fig. 2. Note that by Lemma 6, for every minimally well-matched word $w = c \cdot w' \cdot r$, $w \in \mathcal{L}(\mathcal{A})$ iff $w \in \mathcal{L}(MWM(\mathcal{A}))$ iff there is $p \in \Upsilon_\mathcal{A}(c)$ and a $(2,p)$-run of $MWM(\mathcal{A})$ over w.

By construction, it follows that $\mathcal{A}^{\circlearrowleft\square}$ is a matched-call forward. We only need to prove the construction is correct, that is, $\mathcal{L}(\mathcal{A}^{\circlearrowleft\square}) = (\mathcal{L}(\mathcal{A}))^{\circlearrowleft\square}$.

Inclusion $\mathcal{L}(\mathcal{A}^{\circlearrowleft\square}) \subseteq [\mathcal{L}(\mathcal{A})]^{\circlearrowleft\square}$. Let $w \in \mathcal{L}(\mathcal{A}^{\circlearrowleft\square})$. Consequently, there is an accepting $(1, q_0)$-run G of $\mathcal{A}^{\circlearrowleft\square}$ over w. Let $N(G)$ be the number of vertices v in G such that v has successors obtained by applying some *new* move. We prove by induction on $N(G)$ that $w \in [\mathcal{L}(MWM(\mathcal{A}))]^{\circlearrowleft\square}$, which implies the desired result. If $N(G) = 0$, then $w \in \mathcal{L}(MWM(\mathcal{A}))$ and the result follows. Now, assume that $N(G) > 0$. By construction, there is a vertex $v = (i, q)$ such that $q \in \mathsf{S}$, v has two successors $v_1 = (j + 1, p)$ and $v_2 = (i + 1, p')$, $w[i, j]$ is a minimally well-matched word, the atom $(\rightarrow, p, p)$ occurs in $\delta(q, \square)$, and $p' \in \Upsilon_{\mathcal{A}}(w(i))$. By construction (see Fig. 2), for some main state q', $\delta(q_0, w(i))$ contains the conjunct $(\curvearrowright, q', \bot_{\mathsf{M}}) \wedge (\rightarrow, p', p')$ and $\delta(q', w(j))$ contains some accepting main state (Lemma 6). We can assume that the vertex v is chosen in such a way that all the (secondary) states reachable from v_2 are obtained by $MWM(\mathcal{A})$-moves. It follows that there is an accepting pseudo (i, q_0)-run of $MWM(\mathcal{A})$ over w whose main path visits position $j+1$ in an accepting state. By Lemma 4(1), it follows that $w[i, j] \in \mathcal{L}(MWM(\mathcal{A}))$ and every G-vertex which is a descendant of v_2 is associated with a position in $[i + 1, j]$. Let G' be the graph defined as follows:

- G' is obtained from G by removing vertex v_2 and all its descendants, and replacing each remaining vertex $v' = (j', q'')$ such that $j' > j$ with the vertex $(j' - (j - i), q'')$.

We claim that G' is an accepting $(1, q_0)$-run of $\mathcal{A}^{\circlearrowleft\square}$ on $w[1, i-1]\cdot\square\cdot w[j+1, |w|]$. Hence, since $N(G') < N(G)$ and $w[i, j] \in \mathcal{L}(MWM(\mathcal{A}))$, by the induction hypothesis, the result follows. Note that for each call position i' such that $i' < i$ and its matched return position j' along w satisfies $j' > j$, $j' - (j - i)$ is the matching return position of i' along $w[1, i - 1] \cdot \square \cdot w[j + 1, |w|]$. Moreover, by Lemma 4(2), v is the unique vertex in G associated with the call position i along w. It follows that there is no vertex v' in G such that v' is not reachable from v and at the same time v' is associated with a position in $[i, j]$. Therefore, the claim follows.

Converse inclusion $[\mathcal{L}(\mathcal{A})]^{\circlearrowleft\square} \subseteq \mathcal{L}(\mathcal{A}^{\circlearrowleft\square})$. Let $w \in [\mathcal{L}(\mathcal{A})]^{\circlearrowleft\square}$. In this case, $w \in [\mathcal{L}(MWM(\mathcal{A}))]^{\circlearrowleft\square}$. We show by induction on $|w|$ that $w \in \mathcal{L}(\mathcal{A}^{\circlearrowleft\square})$, and the result follows. Note that $|w| \geq 2$. If $w \in \mathcal{L}(MWM(\mathcal{A}))$, the result follows immediately, since by construction, every run of $MWM(\mathcal{A})$ is also a run of $\mathcal{A}^{\circlearrowleft\square}$. This includes the base case $|w| = 2$.

Assume now that $w \notin \mathcal{L}(MWM(\mathcal{A}))$. Then, there is $u \in \mathcal{L}(MWM(\mathcal{A}))$ such that w is of the form $w = w' \cdot u \cdot w''$ and $w' \cdot \square \cdot w'' \in [\mathcal{L}(MWM(\mathcal{A}))]^{\circlearrowleft\square}$. By the induction hypothesis, $w' \cdot \square \cdot w'' \in \mathcal{L}(\mathcal{A}^{\circlearrowleft\square}) \subseteq MWM(\Sigma)$. Hence, there is an accepting $(1, q_0)$-run G of $\mathcal{A}^{\circlearrowleft\square}$ over $w' \cdot \square \cdot w''$. By Lemma 4(2), there is a unique vertex $v = (|w'| + 1, q)$ associated with the position $|w'| + 1$ (the '$\square$-position'). Moreover, since $\mathcal{A}^{\circlearrowleft\square}$ is a matched-call forward MAJA and $w' \cdot \square \cdot w'' \in MWM(\Sigma)$, by construction, $q \in \mathsf{S}$, v has a unique successor v_1 of the form $(|w'| + 2, p)$ and $(\rightarrow, p, p)$ is a move of $MWM(\mathcal{A})$ from q on reading $\square$. Since $u \in \mathcal{L}(MWM(\mathcal{A}))$, u is of the form $c \cdot u' \cdot r$ for some call c and return r and there is a $(2, p')$-run G' of $MWM(\mathcal{A})$ on u for some $p' \in \Upsilon_{\mathcal{A}}(c)$. By

$\mathcal{A} = \langle \mathsf{M} \cup \mathsf{S} \cup \{\bot_\mathsf{M}, \bot_\mathsf{S}\}, q_0, \delta, Acc\rangle \quad MWM(\mathcal{A}') = \langle \mathsf{M}' \cup \mathsf{S}' \cup \{\bot_\mathsf{M}, \bot_\mathsf{S}\}, q_0', \delta', Acc'\rangle$

$(\mathsf{M} \cup \mathsf{S}) \cap (\mathsf{M}' \cup \mathsf{S}') = \emptyset$ and $\Box \notin \mathcal{L}(\mathcal{A}')$

For all calls c,
$\Upsilon_{\mathcal{A}'}(c) \stackrel{\text{def}}{=} \{p \in \mathsf{S}' \mid \delta'(q_0', c)$ contains some conjunct of the form $(\curvearrowright, q, \bot_\mathsf{M}) \wedge (\rightarrow, p, p)\}$

M-substitution: $\mathcal{A} \curvearrowright_\Box \mathcal{A}' = \langle \mathsf{M} \cup \mathsf{S} \cup \mathsf{M}' \cup \mathsf{S}' \cup \{\#, \bot_\mathsf{M}, \bot_\mathsf{S}\}, q_0, \delta'', Acc \cup \mathsf{S}'\rangle$

$$\delta''(q,\sigma) = \begin{cases} \delta'(q,\sigma) & \text{if } q \in \mathsf{M}' \cup \mathsf{S}' \cup \{\bot_\mathsf{M}, \bot_\mathsf{S}\} \\ \delta(q,\sigma) & \text{if } q \in \mathsf{M} \cup \mathsf{S},\ \sigma \notin \Sigma_{call}, \text{ and } \sigma \neq \Box \\ \mathit{Update}(\delta(q,\sigma)) & \text{if } q \in \mathsf{M} \cup \mathsf{S} \text{ and } \sigma \in \Sigma_{call} \\ \bigvee_{p \in \mathsf{M} \cap Acc} (\rightarrow, p, p) & \text{if } q = \# \text{ and } \sigma \in \Sigma_{ret} \\ \texttt{false} & \text{otherwise} \end{cases}$$

For all $q \in \mathsf{M}$, $p \in \mathsf{S}$ and calls c,

- $\mathit{Update}(\delta(p,c)) \stackrel{\text{def}}{=}$ for all $(\rightarrow, p', p')$-moves of $\mathcal{A}$ from p on reading $\Box$ – note that $p' \in \mathsf{S}$ – and for all $p'' \in \Upsilon_{\mathcal{A}'}(c)$, we add in $\delta(p,c)$ the conjunct $(\curvearrowright_\bullet, p', \bot_\mathsf{S}) \wedge (\rightarrow, p'', p'')$.
- $\mathit{Update}(\delta(q,c)) \stackrel{\text{def}}{=}$ for all $(\rightarrow, q', q')$-moves of $\mathcal{A}$ from q on reading $\Box$ – note that $q' \in \mathsf{M}$ – and for all $p' \in \Upsilon_{\mathcal{A}'}(c)$, we add in $\delta(q,c)$ the conjunct $(\curvearrowright_\bullet, q', \bot_\mathsf{M}) \wedge (\rightarrow, p', p')$ if q' is not accepting, and the conjunct $(\curvearrowright, \#, \bot_\mathsf{M}) \wedge (\rightarrow, p', p')$ otherwise.

Fig. 3 Construction for M-substitution of matched-call forward MAJA $\mathcal{A}$ and $\mathcal{A}'$.

construction, $(\curvearrowright_\bullet, p, \bot_\mathsf{S}) \wedge (\rightarrow, p', p')$ is a conjunct in $\delta^{\circlearrowleft_\Box}(q,c)$. Let G_u, $G'_{+|w'|}$, and G'' be the graphs defined as follows:

- G_u is obtained from G by replacing each vertex (i', q') such that $i' > |w'|+1$ with the vertex $(i' + |u| - 1, q')$;
- $G'_{+|w'|}$ is obtained from G' by replacing each vertex (j,t) of G' with $(j + |w'|, t)$;
- G'' is obtained by merging G_u and $G'_{+|w'|}$, and adding an edge from the vertex v of G_u to the initial vertex of $G'_{+|w'|}$.

By reasoning as for the inclusion $\mathcal{L}(\mathcal{A}^{\circlearrowleft_\Box}) \subseteq [\mathcal{L}(\mathcal{A})]^{\circlearrowleft_\Box}$, we obtain that G'' is an accepting $(1, q_0)$-run of $\mathcal{A}^{\circlearrowleft_\Box}$ on $w = w' \cdot u \cdot w''$. Hence, the result follows.

Construction for M-substitution. The construction is given in Fig. 3, where $\#$ is a fresh non-accepting main state and we assume that $\Box \notin \mathcal{L}(\mathcal{A}')$ (the other case is similar). Note that one can check whether $\Box \notin \mathcal{L}(\mathcal{A}')$. The proof of correctness is analogous to the proof of S-closure.

This concludes the proof of Lemma 7. $\Box$

Polynomial translation of well-formed VRE into MAJA with linear size. We are ready to prove the desired result.

Theorem 7 *For a well-formed VRE α, one can* compositionally *construct in polynomial time a forward MAJA and a backward MAJA both accepting $\mathcal{L}(\alpha) \setminus \{\varepsilon\}$ and having at most $O(|\alpha|)$ states.*

Proof For a basic sub-expression α in $\Sigma \cup \{call, ret, int, \emptyset, \varepsilon\}$, one can directly construct a forward (resp., backward) MAJA with a constant number of states accepting $\mathcal{L}(\mathcal{A}) \setminus \{\varepsilon\}$. Moreover, for VRE α_1 and α_2, it holds that

$$\begin{aligned}
\mathcal{L}(\alpha_1 \curvearrowleft_{\square} \alpha_2) \setminus \{\varepsilon\} &= (\mathcal{L}(\alpha_1) \setminus \{\varepsilon\}) \curvearrowleft_{\square} (\mathcal{L}(\alpha_2) \setminus \{\varepsilon\}) \\
\mathcal{L}(\alpha_1^{\circlearrowleft_{\square}}) \setminus \{\varepsilon\} &= (\mathcal{L}(\alpha_1) \setminus \{\varepsilon\})^{\circlearrowleft_{\square}} \\
\mathcal{L}(\alpha_1^*) \setminus \{\varepsilon\} &= (\mathcal{L}(\alpha_1) \setminus \{\varepsilon\})^* \setminus \{\varepsilon\} \\
\mathcal{L}(\alpha_1 \cdot \alpha_2) \setminus \{\varepsilon\} &= \begin{cases} (\mathcal{L}(\alpha_1) \setminus \{\varepsilon\}) \cdot (\mathcal{L}(\alpha_2) \setminus \{\varepsilon\}) & \text{if } \varepsilon \notin \mathcal{L}(\alpha_1) \cup \mathcal{L}(\alpha_2) \\ \mathcal{L}(\alpha_1) \cdot (\mathcal{L}(\alpha_2) \setminus \{\varepsilon\}) & \text{if } \varepsilon \in \mathcal{L}(\alpha_1) \\ (\mathcal{L}(\alpha_1) \setminus \{\varepsilon\}) \cdot \mathcal{L}(\alpha_2) & \textit{otherwise} \end{cases}
\end{aligned}$$

Moreover, one can check in linear time whether for a given VRE α, $\varepsilon \in \mathcal{L}(\alpha)$. Thus, by Lemma 5, it suffices to show that given a *well-matched* VRE α, one can construct in polynomial time a matched-call forward MAJA with $O(|\alpha|)$ states which accepts $\mathcal{L}(\alpha) \setminus \{\varepsilon\}$ (similarly for a matched-return backward). The proof is by structural induction on α. The induction step easily follows from Lemmata 5 and 7. In particular, note that by Lemmata 5 and 7, each non-basic step adds at most a new state. For the base case, α can be either ε, or $\emptyset$, or $\square$, or $c \cdot \xi \cdot r$, for some $\square \in \Sigma_{int}$, $c \in \Sigma_{call}$, $r \in \Sigma_{ret}$, and regular expression ξ over Σ_{int}. The unique non-trivial case is when $\alpha = c \cdot \xi \cdot r$. Kleene's theorem [18] guarantees that one can construct compositionally and in polynomial time two non-deterministic finite-state automata (NFA) $\mathcal{A}_\xi$ and $\mathcal{A}_\xi^R$ over Σ_{int} accepting the language $\mathcal{L}(\xi)$ and the reverse of $\mathcal{L}(\xi)$, respectively, and having at most $O(|\xi|)$ states. Then, the construction of the matched-call forward MAJA $\mathcal{A}$ and the matched-return MAJA $\mathcal{A}^R$ accepting $\mathcal{L}(c \cdot \xi \cdot r)$ are based on the NFA $\mathcal{A}_\xi$ and $\mathcal{A}_\xi^R$. Here, we focus on the matched-call forward MAJA $\mathcal{A}$. Let $\mathcal{A}_\xi = \langle Q, q_0, \Delta, F \rangle$. Then, $\mathcal{A} = \langle \mathsf{M} \cup \mathsf{S} \cup \{\bot_\mathsf{M}, \bot_\mathsf{S}\}, c, \delta, \mathsf{S} \cup \{\bot_\mathsf{S}, q_{acc}\} \rangle$, where $\mathsf{M} = \{c, \#, q_{acc}\}$, $\mathsf{S} = Q$, and δ is defined as follows.

$$\delta(q, \sigma) = \begin{cases} (\curvearrowright, \#, \bot_\mathsf{M}) \wedge (\rightarrow, q_0, q_0) & \text{if } q = c \text{ and } \sigma = c \\ (\rightarrow, q_{acc}, q_{acc}) & \text{if } q = \# \text{ and } \sigma \in \Sigma_{ret} \\ \bigvee_{q' \in \mathsf{S} : (q, \sigma, q') \in \Delta} (\rightarrow, q', q') & \text{if } q \in \mathsf{S} \text{ and } \sigma \in \Sigma_{int} \\ \texttt{true} & \text{if } \sigma = r \text{ and for some } (q, r, q') \in \Delta,\ q' \in F \\ \texttt{false} & \text{otherwise} \end{cases}$$

This concludes the proof of the theorem. $\square$

6 Decision Procedures for VLTL

In this section, we study the satisfiability and visibly pushdown model checking problems for VLTL. Based on Lemma 2 and Theorems 6 and 7, we derive a polynomial-time compositional translation of VLTL formulas into SAJA, which provides an automata-theoretic approach to these decision problems.

Theorem 8 (From VLTL to SAJA) *Given a VLTL formula φ, one can construct in polynomial time a SAJA $\mathcal{A}_\varphi$ such that $\mathcal{L}_p(\mathcal{A}_\varphi) = \mathcal{L}_p(\varphi)$. Moreover, $\mathcal{A}_\varphi$ satisfies the following:*

1. *there are no ε-moves from the initial state;*
2. *the size of the largest non-trivial coBüchi stratum of $\mathcal{A}$ is linear in the size of the largest VRE associated with a weak future power operator in φ which is in the scope of an odd number of negations;*
3. *the set of main states is $O(|\varphi|)$ (where $|\varphi|$ is the size of φ), while the set of secondary states is $O(|\varphi|^4)$ for the general case, and it is just $O(|\varphi|)$ if either φ is well-formed, or the VRE occurring in φ have a fixed size.*

Proof The translation is described by induction on the structure of the given VLTL formula φ. The base case $\varphi = \mathtt{true}$ is trivial (in particular, the SAJA has only one state and the stratum associated to such a state is transient). For the induction step, given two VLTL formulas φ_1 and φ_2, assume that $\mathcal{A}_{\varphi_1} = \langle \mathsf{M}_1 \cup \mathsf{S}_1, q_1^0, \delta_1, \mathsf{F}_1 \rangle$ and $\mathcal{A}_{\varphi_2} = \langle \mathsf{M}_2 \cup \mathsf{S}_2, q_2^0, \delta_2, \mathsf{F}_2 \rangle$ are the SAJA associated with the VLTL formulas φ_1 and φ_2, which accept the ω-pointed languages $\mathcal{L}_p(\varphi_1)$ and $\mathcal{L}_p(\varphi_2)$. Assume also that $\mathcal{A}_{\varphi_1}$ and $\mathcal{A}_{\varphi_2}$ satisfy the invariant given by Conditions 1–3 in the theorem. Given a VLTL formula φ obtained from φ_1 and φ_2 by applying one of the VLTL operators we illustrate the construction of the SAJA $\mathcal{A}_\varphi = \langle \mathsf{M} \cup \mathsf{S}, q_0, \delta, \mathsf{F} \rangle$ associated with the formula φ. Then, at the end of the proof we show that $\mathcal{A}_\varphi$ accepts $\mathcal{L}_p(\varphi)$ and satisfies Conditions 1–3 in the theorem as well. In the construction, for the VRE α associated with a temporal modality, we use the MAJA $\mathcal{A}_\alpha = \langle \mathsf{M}_\alpha \cup \mathsf{S}_\alpha \cup \{\bot_\mathsf{M}, \bot_\mathsf{S}\}, q_\alpha, \delta_\alpha, Acc_\alpha \rangle$ associated with α of Theorem 6 or Theorem 7 (depending on whether α is well-formed or not). Recall that $\mathsf{M}_\alpha \cup \{\bot_\mathsf{M}\}$ is the set of main states containing the special main state $\bot_\mathsf{M} \notin \mathsf{M}_\alpha$, and $\mathsf{S}_\alpha \cup \{\bot_\mathsf{S}\}$ is the set of secondary states containing the special secondary state $\bot_\mathsf{S} \notin \mathsf{S}_\alpha$ (see Definition 11). In the following, in order to simplify the notation, we write M_α instead of $\mathsf{M}_\alpha \cup \{\bot_\mathsf{M}\}$, and S_α instead of $\mathsf{S}_\alpha \cup \{\bot_\mathsf{S}\}$. In particular, for all input symbols σ and states q of $\mathcal{A}_\alpha$, we use the notation $q \rightarrow_\sigma Acc_\alpha$ to mean that q is a main state and $\delta_\alpha(q, \sigma)$ contains atoms θ of the form (dir, q, q_{acc}) or (dir, q_{acc}, q) for some accepting main state q_{acc}. Similarly, $q \not\rightarrow_\sigma Acc_\alpha$ represents that $\delta_\alpha(q, \sigma)$ does not contains such atoms. Note that since $\mathcal{A}_\alpha$ is a MAJA, by Definition 11, it follows that θ is a singleton conjunct in $\delta_\alpha(q, \sigma)$. Moreover, $\theta = (dir, q_{acc}, q_{acc})$ and $dir = \rightarrow$ if $\mathcal{A}_\alpha$ is forward, and $dir = \leftarrow$ if $\mathcal{A}_\alpha$ is backward.

Disjunction: $\varphi = \varphi_1 \vee \varphi_2$**.** The initial state q_0 of $\mathcal{A}_\varphi$ is a fresh state and:

$$\mathsf{M} = \mathsf{M}_1 \cup \mathsf{M}_2 \cup \{q_0\},\ \mathsf{S} = \mathsf{S}_1 \cup \mathsf{S}_2$$

$$\delta(q, \sigma) = \begin{cases} \delta_1(q_1^0, \sigma) \vee \delta_2(q_2^0, \sigma) & \text{if } q = q_0 \\ \delta_1(q, \sigma) & \text{if } q \in \mathsf{M}_1 \cup \mathsf{S}_1 \\ \delta_2(q, \sigma) & \text{if } q \in \mathsf{M}_2 \cup \mathsf{S}_2 \end{cases}$$

$$\mathsf{F} = \mathsf{F}_1 \cup \mathsf{F}_2 \cup \{\langle \mathtt{t}, \{q_0\}, \emptyset \rangle\}$$

Negation: $\varphi = \neg\varphi_1$**.** $\mathcal{A}_\varphi$ is the SAJA-dual of $\mathcal{A}_{\varphi_1}$.

Sequencing: $\varphi = \alpha;\varphi_1$ **or** $\varphi = \varphi_1;\alpha$**.** Let $\mathcal{A}_\alpha = \langle \mathsf{M}_\alpha \cup \mathsf{S}_\alpha, q_\alpha, \delta_\alpha, Acc_\alpha\rangle$ be the *forward* (resp., *backward*) MAJA for the VRE α if $\varphi = \alpha;\varphi_1$ (resp., $\varphi = \varphi_1;\alpha$), and such that $(\mathsf{M}_\alpha \cup \mathsf{S}_\alpha) \cap (\mathsf{M}_1 \cup \mathsf{S}_1) = \emptyset$.

$$\mathsf{M} = \mathsf{M}_1 \cup \mathsf{M}_\alpha,\ \mathsf{S} = \mathsf{S}_1 \cup \mathsf{S}_\alpha,\ \text{and } q_0 = q_\alpha$$

$$\delta(q,\sigma) = \begin{cases} \delta_1(q,\sigma) & \text{if } q \in \mathsf{M}_1 \cup \mathsf{S}_1 \\ \delta_\alpha(q,\sigma) & \text{if } q \in \mathsf{S}_\alpha \text{ or } q \not\rightarrow_\sigma Acc_\alpha \text{ or } q = q_\alpha \\ \delta_\alpha(q,\sigma) \vee (\varepsilon, q_1^0, q_1^0) & \text{if } q \rightarrow_\sigma Acc_\alpha \text{ and } q \neq q_\alpha \end{cases}$$

$$\mathsf{F} = \begin{cases} \mathsf{F}_1 \cup \{\langle \mathsf{B}, \mathsf{M}_\alpha \cup \mathsf{S}_\alpha, \emptyset\rangle\} & \text{if } \varphi = \alpha;\varphi_1 \\ \mathsf{F}_1 \cup \{\langle -, \mathsf{M}_\alpha \cup \mathsf{S}_\alpha, \mathsf{S}_\alpha\rangle\} & \text{if } \varphi = \varphi_1;\alpha \end{cases}$$

Future Power and Future Weak Power. Let $\varphi = \varphi_1 | \alpha \rangle\!\rangle \varphi_2$ or $\varphi = \varphi_1 | \alpha \rangle \varphi_2$. Let $\mathcal{A}_\alpha = \langle \mathsf{M}_\alpha \cup \mathsf{S}_\alpha, q_\alpha, \delta_\alpha, Acc_\alpha\rangle$ be the *forward* MAJA for the VRE α and such that $(\mathsf{M}_\alpha \cup \mathsf{S}_\alpha) \cap (\mathsf{M}_1 \cup \mathsf{S}_1) = \emptyset$ and $(\mathsf{M}_\alpha \cup \mathsf{S}_\alpha) \cap (\mathsf{M}_2 \cup \mathsf{S}_2) = \emptyset$. Then, the initial state q_0 of $\mathcal{A}_\varphi$ is a fresh state and:

$$\mathsf{M} = \mathsf{M}_1 \cup \mathsf{M}_2 \cup \mathsf{M}_\alpha \cup \{q_0\} \text{ and } \mathsf{S} = \mathsf{S}_1 \cup \mathsf{S}_2 \cup \mathsf{S}_\alpha$$

$$\delta(q,\sigma) = \begin{cases} \delta_2(q_2^0,\sigma) \vee (\delta_1(q_1^0,\sigma) \wedge \delta_\alpha(q_\alpha,\sigma)) & \text{if } q = q_0 \\ \delta_1(q,\sigma) & \text{if } q \in \mathsf{M}_1 \cup \mathsf{S}_1 \\ \delta_2(q,\sigma) & \text{if } q \in \mathsf{M}_2 \cup \mathsf{S}_2 \\ \delta_\alpha(q,\sigma) & \text{if } q \in \mathsf{S}_\alpha \text{ or } q \not\rightarrow_\sigma Acc_\alpha \\ \delta_\alpha(q,\sigma) \vee (\varepsilon, q_0, q_0) & \text{if } q \rightarrow_\sigma Acc_\alpha \end{cases}$$

$$\mathsf{F} = \begin{cases} \mathsf{F}_1 \cup \mathsf{F}_2 \cup \{\langle \mathsf{B}, \mathsf{M}_\alpha \cup \mathsf{S}_\alpha \cup \{q_0\}, \emptyset\rangle\} & \text{if } \varphi = \varphi_1 | \alpha \rangle\!\rangle \varphi_2 \\ \mathsf{F}_1 \cup \mathsf{F}_2 \cup \{\langle \mathsf{B}, \mathsf{M}_\alpha \cup \mathsf{S}_\alpha \cup \{q_0\}, \{q_0\}\rangle\} & \text{if } \varphi = \varphi_1 | \alpha \rangle \varphi_2 \end{cases}$$

Past Power and Past Weak Power. Let $\varphi = \varphi_1 \langle\!\langle \alpha | \varphi_2$ or $\varphi = \varphi_1 \langle \alpha | \varphi_2$. Let $\mathcal{A}_\alpha = \langle \mathsf{M}_\alpha \cup \mathsf{S}_\alpha, q_\alpha, \delta_\alpha, Acc_\alpha\rangle$ be the *backward* MAJA for the VRE α and such that $(\mathsf{M}_\alpha \cup \mathsf{S}_\alpha) \cap (\mathsf{M}_1 \cup \mathsf{S}_1) = \emptyset$ and $(\mathsf{M}_\alpha \cup \mathsf{S}_\alpha) \cap (\mathsf{M}_2 \cup \mathsf{S}_2) = \emptyset$. Then, the initial state q_0 of $\mathcal{A}_\varphi$ is a fresh state and:

$$\mathsf{M} = \mathsf{M}_1 \cup \mathsf{M}_2 \cup \mathsf{M}_\alpha \cup \{q_0\} \text{ and } \mathsf{S} = \mathsf{S}_1 \cup \mathsf{S}_2 \cup \mathsf{S}_\alpha$$

$$\delta(q,\sigma) = \begin{cases} \delta_2(q_2^0,\sigma) \vee (\delta_1(q_1^0,\sigma) \wedge \delta_\alpha(q_\alpha,\sigma)) & \text{if } q = q_0 \text{ and } \varphi = \varphi_1 \langle\!\langle \alpha | \varphi_2 \\ \delta_2(q_2^0,\sigma) \vee & \\ \Big(\delta_1(q_1^0,\sigma) \wedge (\delta_\alpha(q_\alpha,\sigma) \vee (\leftarrow, \bot_{\mathsf{M}}, \bot_{\mathsf{S}}))\Big) & \text{if } q = q_0 \text{ and } \varphi = \varphi_1 \langle \alpha | \varphi_2 \\ \delta_1(q,\sigma) & \text{if } q \in \mathsf{M}_1 \cup \mathsf{S}_1 \\ \delta_2(q,\sigma) & \text{if } q \in \mathsf{M}_2 \cup \mathsf{S}_2 \\ \delta_\alpha(q,\sigma) & \text{if } q \in \mathsf{S}_\alpha \text{ or } q \not\rightarrow_\sigma Acc_\alpha \\ \delta_\alpha(q,\sigma) \vee (\varepsilon, q_0, q_0) & \text{if } q \rightarrow_\sigma Acc_\alpha \end{cases}$$

$$\mathsf{F} = \mathsf{F}_1 \cup \mathsf{F}_2 \cup \{\langle -, \mathsf{M}_\alpha \cup \mathsf{S}_\alpha \cup \{q_0\}, \mathsf{S}_\alpha\rangle\}$$

Correctness of the construction. We say that an occurrence of a subformula of φ has positive polarity if it is in the scope of an even number of negations. Otherwise we say that φ has negative polarity. Note that each step of the construction adds *at most* a new transient stratum with just one state if the

associated VLTL operator is a Boolean connective. For steps associated with a temporal modality with VRE α, the step adds a new non-transient stratum whose number of main states is at most $|\mathsf{M}_\alpha|+1$, and whose of secondary states is at most $|\mathsf{S}_\alpha|$. We can also assume that the SAJA $\mathcal{A}_{\varphi_1}$ and $\mathcal{A}_{\varphi_2}$ of the sub-formulas φ_1 and φ_2 share the strata associated with the common sub-formula occurrences at the same polarity in φ_1 and φ_2. By construction it follows that $\mathcal{A}_\varphi$ is a SAJA satisfying Conditions 1–3 in the theorem. We finally show that $\mathcal{L}_p(\mathcal{A}_\varphi) = \mathcal{L}_p(\varphi)$. For the case of negation, the result directly follows from the induction hypothesis and Lemma 2. For the other cases, we illustrate the future power operator and the past power operator. The remaining cases are similar or simpler.

Correctness for the case $\varphi = \varphi_1 | \alpha \rangle\!\rangle \varphi_2$ (future power operator). The acceptance condition of $\mathcal{A}_\varphi$ ensures that every accepting path in a run does not contain infinitely many occurrences of states belonging to the new stratum. Hence, since a forward MAJA satisfies the semantic requirements J1-J3 of Definition 10, it easily follows that

- $(w,i) \in \mathcal{L}_p(\mathcal{A}_\varphi)$ *iff* there are positions $j_1 < \ldots < j_n$ of w such that $j_1 = i$, $(w, j_n) \in \mathcal{L}_p(\mathcal{A}_{\varphi_2})$, and for all $1 \leq k < n$, $(w, j_k) \in \mathcal{L}_p(\mathcal{A}_{\varphi_1})$ and there is a pseudo (j_k, q_α)-run of $\mathcal{A}_\alpha$ over w whose main path visits an accepting main state at position $j_{k+1} + 1$.

Since a forward MAJA satisfies the semantic requirement J4 of Definition 10, there is a pseudo (j_k, q_α)-run of $\mathcal{A}_\alpha$ over w whose main path visits an accepting main state at position $j_{k+1} + 1$ iff $w[j_k, j_{k+1}] \in \mathcal{L}(\mathcal{A}_\alpha)$. Thus, since $\mathcal{L}(\mathcal{A}_\alpha) = \mathcal{L}(\alpha)$, by applying the induction hypothesis, we obtain that $(w,i) \in \mathcal{L}_p(\mathcal{A})$ *iff* there are positions $j_1 < \ldots < j_n$ of w such that $j_1 = i$, $(w, j_n) \models \varphi_2$, and for all $1 \leq k < n$, $(w, j_k) \models \varphi_1$ and $w[j_k, j_{k+1}] \in \mathcal{L}(\alpha)$. This means that $(w,i) \in \mathcal{L}_p(\mathcal{A}_\varphi)$ iff $(w,i) \models \varphi$, and the result follows.

Correctness for the case $\varphi = \varphi_1 \langle\!\langle \alpha | \varphi_2$ (past power operator). The acceptance condition of $\mathcal{A}_\varphi$ ensures that every vertex $(0, q)$ in an accepting run of $\mathcal{A}_\varphi$ satisfies that $q \notin \mathsf{M}_\alpha \cup \{q_0\}$. Thus, by construction and since a backward MAJA fulfills the semantic requirements J1 and J2 of Definition 10, it easily follows that

(*) $(w,i) \in \mathcal{L}_p(\mathcal{A}_\varphi)$ iff *either* $(w,i) \in \mathcal{L}_p(\mathcal{A}_{\varphi_2})$, *or* $i > 1$, $(w,i) \in \mathcal{L}_p(\mathcal{A}_{\varphi_1})$, and for some $1 \leq j < i$, $(w,j) \in \mathcal{L}_p(\mathcal{A}_\varphi)$ and there is a pseudo (i, q_α)-run of $\mathcal{A}_\alpha$ over w whose main path visits an accepting main state at position $j - 1$.

Since a backward MAJA satisfies the semantic requirement J4 of Definition 10, there is a pseudo (i, q_α)-run of $\mathcal{A}_\alpha$ over w whose main path visits an accepting main state at position $j - 1$ iff $w[j, i] \in \mathcal{L}(\mathcal{A}_\alpha)$. Since $\mathcal{L}(\mathcal{A}_\alpha) = \mathcal{L}(\alpha)$, by iterating the above equivalence (*) and applying the induction hypothesis, we obtain that $(w,i) \in \mathcal{L}_p(\mathcal{A})$ iff there are positions $j_1 < \ldots < j_n$ of w such that $j_n = i$, $(w, j_1) \models \varphi_2$, and for all $1 < k \leq n$, $(w, j_k) \models \varphi_1$ and $w[j_{k-1}, j_k] \in \mathcal{L}(\alpha)$. This means that $(w,i) \in \mathcal{L}_p(\mathcal{A}_\varphi)$ iff $(w,i) \models \varphi$, as desired.

This concludes the proof of Theorem 8. □

By Theorems 5 and 8, we obtain the following corollary.

Corollary 1 *For a well-formed VLTL formula φ, one can build a Büchi NVPA $\mathcal{P}$ accepting $\mathcal{L}(\varphi)$ with $2^{O(|\varphi| \cdot \log(k))}$ states and stack symbols, where k is the size of the largest VRE associated with a weak future power operator in φ which is in the scope of an odd number of negations if there is such a VRE, and a fixed constant $c > 1$ otherwise.*

Checking whether $\mathcal{L}(\mathcal{P}) \subseteq \mathcal{L}(\varphi)$ for a pushdown system $\mathcal{P}$ and a VLTL formula φ, reduces to checking emptiness of $\mathcal{L}(\mathcal{P}) \cap \mathcal{L}(\neg\varphi)$. Thus, since checking emptiness for the intersection of ω-VPL by Büchi NVPA is in PTIME [4], and satisfiability and visibly pushdown model checking for CaRet are EXPTIME-complete [3], by Theorems 3, 5, and 8, we obtain the following result.

Corollary 2 *Satisfiability and visibly pushdown model checking for the logic VLTL are EXPTIME-complete.*

Note that our automata-theoretic approach, based on the use of SAJA as an intermediate step, can be conveniently used also for less expressive logical frameworks. In particular, by Theorems 3, 5, and 8, any CaRet or NWTL$^+$ formula φ can be translated into equivalent Büchi NVPA of size $2^{O(|\varphi|)}$, which matches the upper bounds for the known direct translations [3,2]. Note that the size of the alphabet has been considered constant in the given bounds, and that the number of symbols occurring in a formula is bounded by the size of the formula.

Analogously, our approach can also be used to convert formulas φ of RLTL with past into equivalent Büchi non-deterministic finite-state automata of size $2^{O(|\varphi| \cdot \log(k))}$, where k is the size of the largest regular expression associated with a weak future power operator in φ. This follows from Theorem 5 and the fact that the SAJA obtained from φ has only local moves and no secondary states. The recent upper bounds for RLTL [29] tackled only future operators leaving RLTL with past as an open problem.

7 Concluding Remarks

We have introduced and investigated the linear-time temporal logical VLTL which provides a new characterization of the well-known class of ω-VPL. The logic VLTL can be seen as an extension of standard LTL, where the temporal modalities are guarded by VPL over finite words specified by means of Visibly Rational Expressions (the atomic blocks in a VLTL formula). We have made some choices in presenting the VLTL framework. In particular, we have adopted an action-based approach where the pushdown alphabet Σ is explicitly given. However, in formal verification, one usually considers a finite set AP of atomic propositions which represent predicates over the states of the given system. Moreover, for verifying recursive programs, one fixes three additional propositions, here denoted by *call*, *ret*, and *int*: *call* denotes the

invocation of a procedure, *ret* denotes the return from a procedure, and *int* denotes internal actions of the current procedure. Thus, the (observable) properties about a state s of the system can be represented by a pair of the form (τ, P) where $\tau \in \{call, ret, int\}$ and P is a Boolean formula built from atomic propositions in AP. Hence, the set AP induces a pushdown alphabet $\Sigma_{AP} = \Sigma_{call} \cup \Sigma_{ret} \cup \Sigma_{int}$, where $\Sigma_{call} = \{call\} \times 2^{AP}$, $\Sigma_{ret} = \{ret\} \times 2^{AP}$, and $\Sigma_{int} = \{int\} \times 2^{AP}$. By adopting this propositional-based approach, where the pushdown alphabet is implicitly given, the syntax and the semantics of VLTL remain as presented in this paper.

Additionally, the automata-theoretic machinery exploited for the action-based setting can be used for propositional alphabets as well. Even if the size of Σ_{AP} is exponential in the size of AP, one can use a symbolic representation of the transition relations and adapt the constructions in this paper to maintain a succinct encoding of the transition functions of the resulting automata. Therefore, all results of this paper hold for the propositional-based approach as well. For clarity and space considerations we omit the details of these adapted constructions.

Another important observation concerns our choice of definition of the size of a VLTL formula. It is standard to adopt as the size of a formula the number of distinct subformulas, because this corresponds to a succinct encoding of the input. Our choice follows this trend except for the size of the VRE blocks in a VLTL formula, where we use the length of the string describing the VRE, and not the number of distinct VRE subexpressions. This last choice was followed for technical convenience in order to ease the presentation of Section 5.1 regarding the compositional translation of well-formed VRE into forward and backward MAJA (Theorem 7). In particular, for the expressions $\alpha \cup \alpha'$, $\alpha \cdot \alpha'$, and $\alpha \curvearrowleft_{\Box} \alpha'$, we assume that the MAJA $\mathcal{A}$ and $\mathcal{A}'$ resulting from the two operands α and α' have no state in common (see Figures 1 and 3). This assumption can be relaxed and the MAJA $\mathcal{A}$ and $\mathcal{A}'$ can share the states associated with the MAJA resulting from the common subexpressions of α and α'.

As future work, we aim to investigate the relative expressive power of fragments of VLTL and to capture *minimal* expressively complete VLTL fragments. Recent results [32] show that the future fragment of VLTL closed under Boolean connectives and using only the future sequencing operator is already expressively-complete for the class of ω-VPL.

Acknowledgment

We would like to acknowledge the anonymous reviewers for their feedback and numerous suggestions about how to improve this paper.

A From NVPA to well-formed VRE

In this section we prove the following result.

Theorem 9 *Given an NVPA $\mathcal{P}$, one can construct a well-formed VRE α such that $\mathcal{L}(\alpha) = \mathcal{L}(\mathcal{P})$.*

In order to prove Theorem 9, we need additional definitions. We fix an NVPA $\mathcal{P} = \langle Q, Q_0, \Gamma, \Delta, F\rangle$ over the pushdown alphabet Σ. Given $q, p \in Q$, a *summary* of $\mathcal{P}$ from q to p is a run π of $\mathcal{P}$ over some word $w \in MWM(\Sigma)$ from a configuration of the form (q, β) to a configuration of the form (p, β') for some stack contents β and β'. Observe that if π is such a run over $w \in MWM(\Sigma)$ from (q, β) to (p, β'), then $\beta' = \beta$ and the portion of the stack corresponding to β is never read in π. In particular, there is also a run of $\mathcal{P}$ from $(q, \bot)$ to $(p, \bot)$ over w which uses the same transitions used by π. Given a run π of $\mathcal{P}$ over some word w and $\mathcal{S} \subseteq Q \times Q$, we say that the run π uses only *sub-summaries* from $\mathcal{S}$ whenever for all $q, p \in Q$, if there is sub-run of π which is a summary from q to p, then $(q, p) \in \mathcal{S}$.

Let Λ be the following alphabet, disjoint from Σ, given by

$$\Lambda \stackrel{\text{def}}{=} \{\Box_{qp} \mid q, p \in Q\}$$

For each $\Lambda' \subseteq \Lambda$, we denote by $\Sigma_{\Lambda'}$ the pushdown alphabet obtained by adding to Σ the additional symbols in Λ', which are interpreted as internal actions. Moreover, we define $\mathcal{P}_\Lambda = \langle Q, Q_0, \Gamma, \Delta_\Lambda, F\rangle$ as the NVPA over Σ_Λ obtained from $\mathcal{P}$ by adding for each $(q, p) \in Q \times Q$, the internal transition $(q, \Box_{qp}, p)$.

We prove that one can construct a well-formed VRE α over Σ_Λ denoting $\mathcal{L}(\mathcal{P})$ (the additional symbols in Λ are used only as parameters for intermediate substitutions). The proof consists of the following two Lemmata 8 and 9.

Lemma 8 *Assume that for all $(q, p) \in Q \times Q$, one can construct a well-matched VRE $\alpha(q, p)$ such that $\mathcal{L}(\alpha(q, p))$ is the set of words in $MWM(\Sigma)$ so that there is a summary of $\mathcal{P}$ from q to p over w. Then, one can construct a well-formed VRE α such that $\mathcal{L}(\alpha) = \mathcal{L}(\mathcal{P})$.*

Proof Fix an ordering $\{(q_1, p_1), \ldots, (q_n, p_n)\}$ of $Q \times Q$. Given a regular expression ξ over Σ_Λ, let

$$\xi[\Box_{q_1p_1} \leftarrow \alpha(q_1, p_1), \ldots, \Box_{q_np_n} \leftarrow \alpha(q_n, p_n)] \tag{1}$$

be the VRE obtained from ξ by simultaneously replacing each occurrence of $\Box_{q_ip_i}$ with the well-matched VRE $\alpha(q_i, p_i)$. The following follows immediately:

Claim 5. The VRE given by Equation (1) is well-formed and denotes the language

$$\mathcal{L}(\xi) \curvearrowleft_{\Box_{q_1p_1}} \mathcal{L}(\alpha(q_1, p_1)) \ldots \curvearrowleft_{\Box_{q_np_n}} \mathcal{L}(\alpha(q_n, p_n))$$

Let $\mathcal{A}_1 = \langle Q, Q_0, \Delta_1, F\rangle$ and $\mathcal{A}_2 = \langle Q, Q_0, \Delta_2, F\rangle$ be the non-deterministic finite-state automata (NFA) over Σ_Λ, where Δ_1 and Δ_2 are defined as follows:

- Δ_1 is obtained from the transition relation of $\mathcal{P}_\Lambda$ by removing all the push and pop transitions, and by adding for each pop transition of $\mathcal{P}_\Lambda$ of the form $(q, r, \bot, p)$, the transition (q, r, p);
- Δ_2 is obtained from the transition relation of $\mathcal{P}_\Lambda$ by removing all the push and pop transitions, and by adding for each push transition (q, c, p, γ) of $\mathcal{P}_\Lambda$, the transition (q, c, p).

Let

$$\mathcal{L}_{reg} = \bigcup_{(q_0,q,p)\in Q_0\times Q\times F} \mathcal{L}_{1,q_0,q} \cdot \mathcal{L}_{2,q,p}$$

where for all $q, p \in Q$, $\mathcal{L}_{1,q,p}$ and $\mathcal{L}_{2,q,p}$ are the set of words w such that there is a run of the NFA $\mathcal{A}_1$ and NFA $\mathcal{A}_2$ over w from q to p. Then,

$$\mathcal{L}(\mathcal{P}) = \mathcal{L}_{reg} \curvearrowleft_{\Box_{q_1p_1}} \mathcal{L}(\alpha(q_1, p_1)) \ldots \curvearrowleft_{\Box_{q_np_n}} \mathcal{L}(\alpha(q_n, p_n))$$

Moreover, from Kleene's Theorem for regular expressions [18], $\mathcal{L}_{reg}$ can be effectively captured by a regular expression ξ. By Claim 5 above, we obtain that the well-formed VRE

$$\xi[\Box_{q_1p_1} \leftarrow \alpha(q_1,p_1), \ldots, \Box_{q_np_n} \leftarrow \alpha(q_n,p_n)]$$

which denotes the language $\mathcal{L}(\mathcal{P})$, as desired. □

Lemma 9 *For all $(q,p) \in Q \times Q$, one can construct a well-matched VRE denoting the set of words in $MWM(\Sigma)$ so that there is a summary of $\mathcal{P}$ from q to p over w.*

Proof Given $(q,p) \in Q \times Q$, $\mathcal{S} \subseteq Q \times Q$, and $\Lambda' \subseteq \Lambda$, we define the following two languages:

- $S(q,p,\mathcal{S},\Lambda') \stackrel{\text{def}}{=}$ the set of words $w \in MWM(\Sigma_{\Lambda'})$ such that there is a summary of $\mathcal{P}_\Lambda$ over w from q to p which uses only sub-summaries from $\mathcal{S}$.
- $WM(q,p,\mathcal{S},\Lambda') \stackrel{\text{def}}{=}$ the set of words $w \in WM(\Sigma_{\Lambda'})$ such that there is a run of $\mathcal{P}_\Lambda$ over w from $(q,\bot)$ to some configuration of the form (p,β) which uses only sub-summaries from $\mathcal{S}$ (note that $\beta = \bot$).

We show by induction on the cardinality of the finite set $\mathcal{S}$ that the languages $S(q,p,\mathcal{S},\Lambda')$, $WM(q,p,\mathcal{S},\Lambda')$, and $\{c\} \cdot WM(q,p,\mathcal{S},\Lambda') \cdot \{r\}$ with $c \in \Sigma_{call}$ and $r \in \Sigma_{ret}$ can be effectively denoted by well-matched VRE over $\Sigma_{\Lambda'}$. The Lemma follows from this claim.

Base case: $\mathcal{S} = \emptyset$. The result for $S(q,p,\emptyset,\Lambda')$ follows immediately because $S(q,p,\emptyset,\Lambda') = \emptyset$. We only need to prove that the language $WM(q,p,\emptyset,\Lambda')$ can be effectively denoted by a regular expression over the set of internal actions given by $\Sigma_{int} \cup \Lambda'$. Let $\mathcal{A} = \langle Q, Q_0, \Delta', F\rangle$ be the non-deterministic finite-state automaton (NFA) over $\Sigma_{int} \cup \Lambda'$, where Δ' is obtained from the transition relation of $\mathcal{P}_\Lambda$ by removing all push and pop transitions, and all internal transitions $(q', \Box_{q'p'}, p')$ with $\Box_{q'p'} \notin \Lambda'$. Since no word in $WM(q,p,\emptyset,\Lambda')$ contains occurrences of calls or occurrences of returns, $WM(q,p,\emptyset,\Lambda')$ is the set of words w over $\Sigma_{int} \cup \Lambda'$ such that there is a run of $\mathcal{A}$ over w from q to p. Kleene's Theorem [18] guarantees that one can construct a regular expression over $\Sigma_{int} \cup \Lambda'$ denoting $WM(q,p,\emptyset,\Lambda')$.

Induction step: $\mathcal{S} \neq \emptyset$. Let $(q',p') \in \mathcal{S}$ and $P_{q' \to p'}$ be the set:

$$\{(s,c,r,s') \in Q \times \Sigma_{call} \times \Sigma_{ret} \times Q \mid \text{ there is } \gamma \in \Gamma.\, (q',c,s,\gamma), (s',r,\gamma,p') \in \Delta\}$$

So, $P_{q' \to p'}$ is the set of tuples (s,c,r,s') such that there is a push transition from q' to s reading the call c and a matching pop transition from s' to p' reading r. It easily follows that

$$\begin{aligned}
&S(q',p',\mathcal{S},\Lambda') = \\
&\Big(\big[\bigcup_{(s,c,r,s') \in P_{q' \to p'}} \{c\} \cdot WM(s,s',\mathcal{S} \setminus \{(q',p')\}, \Lambda' \cup \{\Box_{q'p'}\}) \cdot \{r\}\big]^{\curvearrowright \Box_{q'p'}}\Big) \curvearrowright \Box_{q'p'} \\
&\big[\bigcup_{(s,c,r,s') \in P_{q' \to p'}} \{c\} \cdot WM(s,s',\mathcal{S} \setminus \{(q',p')\}, \Lambda') \cdot \{r\}\big]
\end{aligned}$$

By the induction hypothesis, the sets $\{c\} \cdot WM(s,s',\mathcal{S} \setminus \{(q,'p')\}, \Lambda' \cup \{\Box_{q'p'}\}) \cdot \{r\}$ and $\{c\} \cdot WM(s,s',\mathcal{S} \setminus \{(q,'p')\}, \Lambda') \cdot \{r\}$ can be effectively denoted by well-matched VRE. Thus, for each $(q',p') \in \mathcal{S}$, the set $S(q',p',\mathcal{S},\Lambda')$ can be effectively denoted by a well-matched VRE.

Fix an ordering $\{(p_1,q_1), \ldots, (p_n,q_n)\}$ of $\mathcal{S}$ and let $\Lambda_\mathcal{S}$ be the subset of Λ given by $\{\Box_{q_1p_1}, \ldots, \Box_{q_np_n}\}$. Now, we observe that for all calls c and returns r,

$$\begin{aligned}
WM(q,p,\mathcal{S},\Lambda') = WM(q,p,\emptyset,\Lambda' \cup \Lambda_\mathcal{S}) &\curvearrowright \Box_{q_1p_1} (S(q_1,p_1,\mathcal{S},\Lambda') \cup \Lambda') \ldots \\
&\ldots \curvearrowright \Box_{q_np_n} (S(q_n,p_n,\mathcal{S},\Lambda') \cup \Lambda')
\end{aligned}$$

$$\begin{aligned}
\{c\} \cdot WM(q,p,\mathcal{S},\Lambda') \cdot \{r\} = \{c\} \cdot WM(q,p,\emptyset,\Lambda' \cup \Lambda_\mathcal{S}) \cdot \{r\} &\curvearrowright \Box_{q_1p_1} \\
(S(q_1,p_1,\mathcal{S},\Lambda') \cup \Lambda') \ldots &\curvearrowright \Box_{q_np_n} (S(q_n,p_n,\mathcal{S},\Lambda') \cup \Lambda')
\end{aligned}$$

Moreover, note that if $(q,p) \notin \mathcal{S}$, then $S(q,p,\mathcal{S},\Lambda') = \emptyset$. Hence, the result follows. □

References

1. R. Alur. Marrying words and trees. In *Proc. 26th PODS*, pages 233–242. ACM, 2007.
2. R. Alur, M. Arenas, P. Barceló, K. Etessami, N. Immerman, and L. Libkin. First-order and temporal logics for nested words. In *Proc. 22nd LICS*, pages 151–160. IEEE Computer Society, 2007.
3. R. Alur, K. Etessami, and P. Madhusudan. A temporal logic of nested calls and returns. In *Proc. 10th TACAS*, volume 2988 of *LNCS*, pages 467–481. Springer, 2004.
4. R. Alur and P. Madhusudan. Visibly pushdown languages. In *Proc. 36th STOC*, pages 202–211. ACM, 2004.
5. R. Alur and P. Madhusudan. Adding nesting structure to words. *J. ACM*, 56(3), 2009.
6. M. Arenas, P. Barceló, and L. Libkin. Regular languages of nested words: Fixed points, automata, and synchronization. In *Proc. 34th ICALP*, volume 4596 of *LNCS*, pages 888–900. Springer, 2007.
7. R. Armoni, L. Fix, A. Flaisher, R. Gerth, B. Ginsburg, T. Kanza, A. Landver, S. Mador-Haim, E. Singerman, A. Tiemeyer, and M. Y. Vardi. The ForSpec temporal logic: A new temporal property-specification language. In *Proc. TACAS*, volume 2280 of *LNCS*, pages 296–211. Springer, 2002.
8. T. Ball and S. K. Rajamani. Bebop: a symbolic model checker for boolean programs. In *Proc. 7th SPIN*, volume 1885 of *LNCS*, pages 113–130. Springer, 2000.
9. B. Bollig, A. Cyriac, P. Gastin, and M. Zeitoun. Temporal logics for concurrent recursive programs: Satisfiability and model checking. In *Proc. 36th MFCS*, volume 6907 of *LNCS*, pages 132–144. Springer, 2011.
10. L. Bozzelli. Alternating automata and a temporal fixpoint calculus for visibly pushdown languages. In *Proc. 18th CONCUR*, volume 4703 of *LNCS*, pages 476–491. Springer, 2007.
11. L. Bozzelli and C. Sánchez. Visibly rational expressions. In *Proc. FSTTCS*, LIPIcs 18, pages 211–223, 2012.
12. L. Bozzelli and C. Sánchez. Visibly rational expressions. *Acta Inf.*, 51(1):25–49, 2014.
13. C. Dax and F. Klaedtke. Alternation elimination for automata over nested words. In *Proc. 14th FOSSACS*, volume 6604 of *LNCS*, pages 168–183. Springer, 2011.
14. C. Dax, F. Klaetke, and M. Lange. On regular temporal logic with past. *Acta Informatica*, 47:251–277, 2010.
15. M. J. Fischer and R. E. Ladner. Propositional dynamic logic of regular programs. *J. Comput. Syst. Sci.*, 18(2):194–211, 1979.
16. J. G. Henriksen and P. S. Thiagarajan. Dynamic linear time temporal logic. *Ann. Pure Appl. Logic*, 96(1-3):187–207, 1999.
17. IEEE Standard 1850–2010. *IEEE Standard for Property Specification Language (PSL)*, Apr. 2010.
18. S. C. Kleene. Representation of events in nerve nets and finite automata. *Automata Studies*, 34:3–41, 1956.
19. O. Kupferman and M. Y. Vardi. Weak alternating automata are not that weak. *ACM Transactions on Computational Logic*, 2(3):408–429, 2001.
20. O. Kupferman, M. Y. Vardi, and P. Wolper. An Automata-Theoretic Approach to Branching-Time Model Checking. *J. ACM*, 47(2):312–360, 2000.
21. M. Leucker and C. Sánchez. Regular linear temporal logic. In *Proc. 4th ICTAC*, volume 4711 of *LNCS*, pages 291–305. Springer, 2007.
22. C. Löding and O. Serre. Propositional dynamic logic with recursive programs. In *Proc. 9th FoSSaCS*, volume 3921 of *LNCS*, pages 292–306. Springer, 2006.
23. P. Madhusudan and M. Viswanathan. Query automata for XML nested words. In *Proc. 34th MFCS*, volume 5734 of *LNCS*, pages 561–573. Springer, 2009.
24. K. Mehlhorn. Pebbling moutain ranges and its application of dcfl-recognition. In *Proc. 7th ICALP*, volume 85 of *LNCS*, pages 422–435. Springer, 1980.
25. S. Miyano and T. Hayashi. Alternating finite automata on ω-words. *Theoretical Computer Science*, 32:321–330, 1984.
26. D. Muller and P. Schupp. Alternating Automata on Infinite Trees. *Theoretical Computer Science*, 54:267–276, 1987.

27. A. Pnueli. The temporal logic of programs. In *Proc. 18th FOCS*, pages 46–57. IEEE Computer Society, 1977.
28. C. Sánchez and M. Leucker. Regular linear temporal logic with past. In *Proc. 11th VMCAI*, volume 5944 of *LNCS*, pages 295–311. Springer, 2010.
29. C. Sánchez and J. Samborski-Forlese. Efficient regular linear temporal logic using dualization and stratification. In *Proc. 19th TIME*, pages 13–20, 2012.
30. L. J. Stockmeyer and A. R. Meyer. Word problems requiring exponential time: Preliminary report. In *Proc. 5th STOC*, pages 1–9. ACM, 1973.
31. M. Y. Vardi and P. Wolper. Reasoning about infinite computations. *Information and Computation*, 115:1–37, 1994.
32. A. Weinert and M. Zimmermann. Visibly linear dynamic logic. In *Proc. 36th FSTTCS (To appear)*, LIPIcs, 2016.
33. P. Wolper. Temporal logic can be more expressive. *Information and Control*, 56:72–99, 1983.
34. W. Zielonka. Infinite games on finitely coloured graphs with applications to automata on infinite trees. *Theoretical Computer Science*, 200(1-2):135–183, 1998.