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

Visibly Rational Expressions

Laura Bozzelli · César Sánchez

Received: date / Accepted: date

Abstract Regular Expressions (RE) are an algebraic formalism for expressing regular languages, widely used in string search and as a specification language in verification. In this paper, we introduce and investigate Visibly Rational Expressions (VRE), an extension of RE for the class of Visibly Pushdown Languages (VPL). We show that VRE capture precisely the class of VPL. Moreover, we identify an *equally expressive* fragment of VRE which admits a quadratic time compositional translation into the automata acceptors of VPL. We also prove that, for this fragment, universality, inclusion and language equivalence are EXPTIME-complete. Finally, we provide an extension of VRE for VPL over infinite words.

Keywords Visibly Pushdown Languages · Context-free specifications · Regular expressions · Algebraic characterization

1 Introduction

Visibly Pushdown Languages (VPL), introduced by Alur et al. [4,5], represent a robust and widely investigated subclass of context-free languages which includes strictly the class of regular languages. A VPL consists of *nested words*,

A preliminary version of this paper appears in the Proceedings of the IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS'12), 15-17 December 2012, Hyderabad, India.

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

that is, words over an alphabet (pushdown alphabet) which is partitioned into three 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 Visibly Pushdown Automata (VPA), a subclass of pushdown automata which push onto the stack only when a call is read, pops the stack only at returns, and do not use the stack on reading internal symbols. Hence, the input controls the kind of operations permissible on the stack, and thus the stack depth at every position [4]. This restriction makes the class of VPL very similar in tractability and robustness to that of regular languages. In particular, VPL are closed under intersection, union, complementation, renaming, concatenation and Kleene closure [4]. Moreover, VPA (over finite words) are determinizable, and decision problems like universality, equivalence and inclusion – which are undecidable for context-free languages – become EXPTIME-complete for VPL. 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 polynomial-time translations from VPL into regular tree languages over tree-encodings of nested words, and vice versa. Furthermore, various alternative and constructive characterizations of VPL have been given in terms of operational and descriptive formalisms: logical characterizations by standard MSO over nested words extended with a binary matching-predicate (MSO_μ) [4] or by fixpoint logics [10], a context-free grammar based characterization [4], alternating automata based characterizations [6,10], and a congruence-based characterization [3].

The theory of VPL has relevant applications in the formal verification and synthesis of sequential recursive programs with finite data types modeled by pushdown systems [7,5,2,19]. Runs in these programs can be seen as nested words capturing the nested calling structure, where the call to and return from procedures capture the nesting. Additionally, VPL have applications 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 [16,21,17,20,1,22]. Examples include type-checking (validation) and dynamic typing of XML documents against schema specifications [17], and evaluation of MSO_μ queries on streaming XML documents [20].

Contribution. Regular Expressions [15,14] (RE) are an algebraic formalism for describing regular languages. RE are widely adopted as a descriptive specification language, for example in string search [23], and for extensions of temporal logics for hardware model checking [13,18]. In this paper, we introduce and investigate a similar algebraic formalism for the class of VPL, that we call *Visibly Regular Expressions* (VRE). VRE extend RE by adding two novel non-regular operators which are parameterized by an internal action: (1) the binary *Minimally Well-Matched Substitution* operator (M-substitution for short), which allows to substitute occurrences of the designated internal action by *min-*

imally well-matched (MWM) words;[1] (2) and the unary *Strict Mimimally Well-Matched Closure* operator (S-closure for short), which corresponds to the (unbounded) iteration of the M-substitution operation. We also consider a third operator which can be expressed in terms of M-substitution and S-closure. The class of *pure* VRE is obtained by disallowing the explicit use of this operator. Intuitively, M-substitution and S-closure, when applied to languages $\mathcal{L}$ of MWM words, correspond to classical tree language concatenation and Kleene closure applied to the tree language encoding of the (nested) word languages $\mathcal{L}$ (in accordance with the standard encoding of MWM words by ordered unranked finite trees [1]).

Our results are as follows. First, we establish that VRE capture exactly the class of VPL. Like the classical Kleene theorem [15], the translation from automata (VPA) to expressions (VRE) involves a singly exponential blow-up. For the converse direction (from VRE to VPA), the proposed construction requires again single exponential time (it is an open question if this exponential blow-up can be avoided). On the other hand, we show that pure VRE – whose expressive power is equivalent to unrestricted VRE – can be compositionally converted in quadratic time into equivalent VPA. The key of this translation is given by a novel subclass of VPA. Next, we prove that universality, inclusion, and language equivalence for pure VRE are EXPTIME-complete. Finally, we also provide an algebraic characterization of VPL over infinite words.

A potential application of our algebraic formalism is as a schema specification language for semi-structured data such as XML documents. In fact, usually, XML schema specifications are context-free grammars and their derivation trees give the tree representation of the associated sets of XML documents. So, these specifications are typically compiled into tree automata. However, it has been shown [17,20,1,22] that VPA are often more natural (and sometime exponentially more succinct) than tree automata, and moreover preferable in the streaming processing of XML documents.

Related Work. Another algebraic characterization of VPL has been given in [8], where regular expressions are extended with an infinite family of operators, which are implicit least fix-points. In fact, these operators encode in a linear way a subclass of context-free grammars. In comparison, we introduce just two operators which make our formalism really more lightweight and intuitive to use. In [12], a characterization of VPL is given by morphisms to suitable algebraic structures. Moreover, [12] introduces an extension of the Kleene-closure free fragment of regular expressions obtained by adding a variant of our substitution operator, and shows that this extension captures exactly the first-order fragment of MSO_μ over well-matched words.

The rest of the paper is structured as follows. In Section 2, we recall the framework of visibly pushdown automata and visibly pushdown languages. Next, in Section 3, we introduce and investigate the class of Visibly Rational

[1] MWM words are words whose first symbol is a call and whose last symbol is the matching return.

Expressions. Then, in Section 4, we describe a compositional quadratic-time translation from pure VRE to VPA, and in Section 5, we provide an algebraic characterization of ω-VPL in terms of ω-VRE. Finally, in Section 6, we give some concluding remarks.

2 Visibly pushdown languages

In this section, we recall the class of visibly pushdown automata and visibly pushdown languages [4].

Pushdown Alphabet. A *pushdown alphabet* is a tuple $\widetilde{\Sigma} = \langle \Sigma_{call}, \Sigma_{ret}, \Sigma_{int} \rangle$ consisting of three disjoint finite alphabets: Σ_{call} is a finite set of *calls*, Σ_{ret} is a finite set of *returns*, and Σ_{int} is a finite set of *internal actions*. For any such $\widetilde{\Sigma}$, the *support* of $\widetilde{\Sigma}$ is $\Sigma = \Sigma_{call} \cup \Sigma_{ret} \cup \Sigma_{int}$. Therefore, Σ^* is the set of finite words over support alphabet. We will use $c, c_1, c_i, \ldots$ for elements of Σ_{call}, $r, r_1, r_i, \ldots$ for elements of Σ_{ret}, $\Box, \Box_1, \Box_i, \ldots$ for elements of Σ_{int}, and $\sigma, \sigma_1, \sigma_i, \ldots$ for arbitrary elements of Σ.

Visibly Pushdown Automata and Visibly Pushdown Languages. A *Nondeterministic Visibly Pushdown Automaton on finite words* (NVPA) [4] *over* $\widetilde{\Sigma} = \langle \Sigma_{call}, \Sigma_{ret}, \Sigma_{int} \rangle$ is a tuple $\mathcal{P} = \langle Q, q_{in}, \Gamma, \Delta, F \rangle$, where Q is a finite set of (control) states, $q_{in} \in Q$ is the initial state, 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. A transition of the form $(q, c, q', \gamma) \in Q \times \Sigma_{call} \times Q \times \Gamma$ is a push transition, where on reading the call c, the symbol $\gamma \neq \bot$ is pushed onto the stack and the control changes from q to q'. A transition of the form $(q, r, \gamma, q') \in Q \times \Sigma_{ret} \times (\Gamma \cup \{\bot\}) \times Q$ is a pop transition, where on reading the return r, γ is read from the top of the stack and popped, and the control changes from q to q' (if the top of the stack is $\bot$, then it is read but not popped). Finally, on reading an internal action $\Box$, $\mathcal{P}$ 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 $\sigma_1 \ldots \sigma_{n-1} \in \Sigma^*$ is a finite sequence of the form $\pi = (q_1, \beta_1) \xrightarrow{\sigma_1} (q_2, \beta_2) \ldots \xrightarrow{\sigma_{n-1}} (q_n, \beta_n)$ such that (q_i, β_i) is a configuration for all $1 \leq i \leq n$, 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 $\beta_{i+1} = \beta_i$.

For all $1 \leq i \leq j \leq n$, the subsequence of π given by $\pi_{ij} = (q_i, \beta_i) \xrightarrow{\sigma_i} \ldots \xrightarrow{\sigma_{j-1}} (q_j, \beta_j)$ is called *subrun of* π (note that π_{ij} is a run of $\mathcal{P}$ over $\sigma_i \ldots \sigma_{j-1}$). The run π is *initialized* if $q_1 = q_{in}$ and $\beta_1 = \bot$. Moreover, the run π is *accepting* if the last state is accepting, that is, if $q_n \in F$. For two configurations (q, β) and (q', β') and $w \in \Sigma^*$, we write $(q, \beta) \xrightarrow{w} (q', \beta')$ to mean that there is a run of

$\mathcal{P}$ over w starting at (q, β) and leading to (q', β'). The language $\mathcal{L}(\mathcal{P})$ of $\mathcal{P}$ is the set of finite words $w \in \Sigma^*$ such that there is an initialized accepting run of $\mathcal{P}$ on w. A language of finite words $\mathcal{L} \subseteq \Sigma^*$ is a *visibly pushdown language* (VPL) *with respect to* $\widetilde{\Sigma}$ if there is an NVPA $\mathcal{P}$ over $\widetilde{\Sigma}$ such that $\mathcal{L} = \mathcal{L}(\mathcal{P})$.

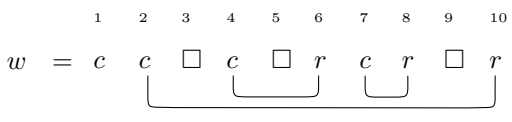
We also consider *Visibly Pushdown Automata on infinite words* (ω-NVPA). Formally (see [4]), a Büchi ω-NVPA *over* $\widetilde{\Sigma}$ is a tuple $\mathcal{P} = \langle Q, q_{in}, \Gamma, \Delta, F \rangle$, where $Q, q_{in}, \Gamma, \Delta$, and F are defined as for NVPA over $\widetilde{\Sigma}$. A run π over an infinite word $\sigma_1\sigma_2 \ldots \in \Sigma^\omega$ is an infinite sequence $\pi = (q_1, \beta_1) \xrightarrow{\sigma_1} (q_2, \beta_2) \ldots$ that is defined using the natural extension of the definition of runs on finite words. The run is *accepting* if for infinitely many $i \geq 1$, $q_i \in F$. The notions of initialized run and (finite) subrun are defined as for NVPA. The ω-language $\mathcal{L}(\mathcal{P})$ of $\mathcal{P}$ is the set of infinite words $w \in \Sigma^\omega$ such that there is an initialized accepting run of $\mathcal{P}$ on w. An ω-language $\mathcal{L} \subseteq \Sigma^\omega$ is an *ω-visibly pushdown language* (ω-VPL) *with respect to* $\widetilde{\Sigma}$ if there is a Büchi ω-NVPA $\mathcal{P}$ over $\widetilde{\Sigma}$ such that $\mathcal{L} = \mathcal{L}(P)$.

Matched calls and returns. Fix a pushdown alphabet $\widetilde{\Sigma} = \langle \Sigma_{call}, \Sigma_{ret}, \Sigma_{int} \rangle$. For a finite or infinite word w over Σ, $|w|$ is the length of w (we set $|w| = \omega$ if w is infinite). For all $1 \leq i \leq |w|$, $w(i)$ is the i^{th} symbol of w. A position $1 \leq i \leq |w|$ of w is a call (resp., return, internal) position if $w(i) \in \Sigma_{call}$ (resp., $w(i) \in \Sigma_{ret}$, $w(i) \in \Sigma_{int}$). The empty word is denoted by ε.

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

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 $w(i+1) \ldots w(j-1)$ is a well-matched word (note that j is uniquely determined if it exists), we say that j is the *matching return* of i along w, and i is the *matching call* of j along w. The set $MWM(\widetilde{\Sigma})$ 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 $MWM(\mathcal{L}) \stackrel{\text{def}}{=} \mathcal{L} \cap MWM(\widetilde{\Sigma})$, 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 subword $w[2] \ldots w[10]$ is minimally well-matched.



3 Visibly Rational Expressions (VRE)

In this section, we introduce and investigate the class of *Visibly Rational Expressions* (VRE), an extension of regular expressions obtained by adding two novel non-regular operators: the binary M-substitution operator and the unary S-closure operator. First, we define the corresponding operations on languages of finite words and show that VPL are effectively closed under these operations. Then, in Subsection 3.1, we introduce VRE and establish their effective language equivalence to the class of NVPA.

Let us fix a pushdown alphabet $\widetilde{\Sigma} = \langle \Sigma_{call}, \Sigma_{ret}, \Sigma_{int} \rangle$. For two languages $\mathcal{L}, \mathcal{L}' \subseteq \Sigma^*$ of finite words on Σ, we use $\mathcal{L} \cdot \mathcal{L}'$ to denote the standard concatenation of $\mathcal{L}$ and $\mathcal{L}'$, and $\mathcal{L}^*$ to denote the standard Kleene closure of $\mathcal{L}$.

Definition 1 (M-substitution) 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 follows:

$$\mathcal{L} \curvearrowleft_\Box \mathcal{L}' \stackrel{\text{def}}{=} \bigcup_{w \in \mathcal{L}} w \curvearrowleft_\Box \mathcal{L}'$$

Note that $\curvearrowleft_\Box$ is associative. Moreover, if $\{\Box\} \cap \mathcal{L} = \emptyset$, then $\{\Box\} \curvearrowleft_\Box \mathcal{L} = MWM(\mathcal{L})$.

Example 2 Let $\Sigma_{call} = \{c_1, c_2\}$, $\Sigma_{ret} = \{r\}$, and $\Sigma_{int} = \{\Box\}$. Let us consider the languages $\mathcal{L} = \{c_1^n \,\Box\, \Box\, r^n \mid n \geq 1\}$ and $\mathcal{L}' = \{c_2\}^* \cdot \{r\}^*$. Then $\mathcal{L} \curvearrowleft_\Box \mathcal{L}'$ is given by $\{c_1^n\, c_2^m\, r^m\, c_2^k\, r^k\, r^n \mid n, m, k \geq 1\}$.

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

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

The *S-closure of $\mathcal{L}$ through $\Box$*, denoted by $\mathcal{L}^{\circlearrowleft_\Box}$, is defined as follows:

$$\mathcal{L}^{\circlearrowleft_\Box} \stackrel{\text{def}}{=} (MWM(\mathcal{L}))^{\curvearrowleft_\Box}$$

Note that $\mathcal{L}^{\circlearrowleft_\Box}$ is contained in $MWM(\widetilde{\Sigma})$. The M-closure operator is a derived operator since it can be expressed in terms of S-closure and M-substitution as follows:

$$\mathcal{L}^{\curvearrowleft_\Box} = \mathcal{L} \curvearrowleft_\Box (\mathcal{L}^{\circlearrowleft_\Box} \cup \{\Box\})$$

Example 3 Let $\Sigma_{call} = \{c_1, c_2\}$, $\Sigma_{ret} = \{r_1, r_2\}$, and $\Sigma_{int} = \{\Box\}$. Let us consider the languages $\mathcal{L} = \{\Box, c_1 \Box r_1,\ c_2 \Box r_2\}$ and $\mathcal{L}' = \{c_1\, r_1,\ c_2\, r_2\}$. Then, $\mathcal{L}^{\curvearrowleft\Box}\ \curvearrowleft_{\Box}\ \mathcal{L}' = \{c_{i_1}\, c_{i_2} \ldots c_{i_n}\, r_{i_n} \ldots r_{i_2}\, r_{i_1} \mid n \geq 1,\ i_1, \ldots, i_n \in \{1, 2\}\}$. It is interesting to observe the following, where $\mathcal{L}_{vis}$ is a shortcut for $\mathcal{L}^{\curvearrowleft\Box}\ \curvearrowleft_{\Box}\ \mathcal{L}'$.

Claim: there is no regular language $\mathcal{L}_{reg}$ such that $MWM(\mathcal{L}_{reg}) = \mathcal{L}_{vis}$.
Proof of the claim: By contradiction, assume there is a finite-state automaton $\mathcal{A} = \langle Q, q_0, \Delta, F\rangle$ over Σ such that $\mathcal{L}_{vis} = MWM(\mathcal{L}(\mathcal{A}))$. Let $K \geq 1$ be such that $2^K > |Q|$ and $\mathcal{L}_K$ be given by

$$\mathcal{L}_K = \{c_{i_1}\, c_{i_2} \ldots c_{i_K}\, r_{i_K} \ldots r_{i_2}\, r_{i_1} \mid i_1, \ldots, i_K \in \{1, 2\}\}$$

Note that $\mathcal{L}_K$ consists of 2^K words of length $2K$. Fix an ordering $w_1, \ldots, w_{2^K}$ of such words, and for each $1 \leq i \leq 2^K$, let w'_i (resp., w''_i) be the prefix (resp., suffix) of w_i of length K. Note that $w_i = w'_i \cdot w''_i$ and for each $j \neq i$, $w'_i \cdot w''_j \in MWM(\widetilde{\Sigma}) \setminus \mathcal{L}_{vis}$. Since $\mathcal{L}_K \subseteq \mathcal{L}_{vis}$ and $2^K > |Q|$, by hypothesis, there must be two distinct words w_i and w_j, $q \in Q$, and two accepting runs π_i and π_j of $\mathcal{A}$ over w_i and w_j of the forms $\pi_i = q_0 \xrightarrow{w'_i} q \xrightarrow{w''_i} q_{acc}$ and $\pi_j = q_0 \xrightarrow{w'_j} q \xrightarrow{w''_j} q'_{acc}$. Hence, we deduce that $w'_i \cdot w''_j \in \mathcal{L}(\mathcal{A})$. Since $w'_i \cdot w''_j \in MWM(\widetilde{\Sigma}) \setminus \mathcal{L}_{vis}$, we obtain a contradiction, and we are done. □

Now we show that VPL are closed under M-substitution, M-closure and S-closure. First, we need the following preliminary result.

Lemma 1 *Given an NVPA $\mathcal{P} = \langle Q, q_{in}, \Gamma, \Delta, F\rangle$, one can build in linear time an NVPA $\mathcal{P}'$ accepting $MWM(\mathcal{L}(\mathcal{P}))$ of the form $\mathcal{P}' = \langle Q', q'_{in}, \Gamma \cup \widehat{\Gamma}, \Delta', F'\rangle$, where $\widehat{\Gamma}$ is a fresh copy of Γ, $|Q'| = |Q| + 2$ and $Q' \supseteq Q$. Moreover, the push transitions from q'_{in} push symbols in $\widehat{\Gamma}$, and each internal transition in Δ' is also in Δ.*

Proof $Q' = Q \cup \{q'_{in}, q'_{acc}\}$, where q'_{in} and q'_{acc} are two distinct fresh states, $F' = \{q'_{acc}\}$, and Δ' is obtained from Δ by adding the following transitions, where for each $\gamma \in \Gamma$, $\widehat{\gamma}$ denotes the copy of γ in $\widehat{\Gamma}$:

- for every push transition $(q_{in}, c, q, \gamma) \in \Delta$ from the initial state of $\mathcal{P}$, we add the push transition $(q'_{in}, c, q, \widehat{\gamma})$;
- for every pop transition $(q, r, \gamma, q_{acc}) \in \Delta$ leading to an accepting state $q_{acc} \in F$ and popping $\gamma \neq \bot$, we add the pop transition $(q, r, \widehat{\gamma}, q'_{acc})$.

Note that there are no transitions of $\mathcal{P}'$ leading to the initial state q'_{in} and there are no transitions of $\mathcal{P}'$ outgoing from the unique accepting state q'_{acc}. It remains to show that $\mathcal{L}(\mathcal{P}') = MWM(\mathcal{L}(\mathcal{P}))$. We consider the inclusion $\mathcal{L}(\mathcal{P}') \subseteq MWM(\mathcal{L}(\mathcal{P}))$ (the converse inclusion is trivial). Let $w \in \mathcal{L}(\mathcal{P}')$. We need to show that $w \in MWM(\mathcal{L}(\mathcal{P}))$. By construction, there are only push transitions (resp., pop transitions) from the initial state (resp., leading to the unique accepting state) in $\mathcal{P}'$. Hence, w is of the form $w = c \cdot w' \cdot r$ and there is an initialized accepting run of $\mathcal{P}'$ over w of the form $\pi =$

$(q'_{in}, \bot) \xrightarrow{c} (q, \widehat{\gamma} \cdot \bot) \xrightarrow{w'} (p, \widehat{\gamma'} \cdot \alpha) \xrightarrow{r} (q'_{acc}, \alpha)$ such that $(q_{in}, c, q, \gamma) \in \Delta$ and $(p, r, \gamma', q_{acc}) \in \Delta$ for some $q_{acc} \in F$. Let us consider the subrun π' of π corresponding to $(q, \widehat{\gamma} \cdot \bot) \xrightarrow{w'} (p, \widehat{\gamma'} \cdot \alpha)$. By construction, symbols in $\widehat{\Gamma}$ cannot be neither popped nor pushed along π', and π' uses only transitions of $\mathcal{P}$. It follows that $\gamma' = \gamma$, $\alpha = \bot$, $w' \in WM(\widetilde{\Sigma})$, and there is also a run of $\mathcal{P}$ over w' of the form $(q, \gamma \cdot \bot) \xrightarrow{w'} (p, \gamma \cdot \bot)$. Hence, $w \in MWM(\widetilde{\Sigma})$ and there is an initialized accepting run of $\mathcal{P}$ over w. This means that $w \in MWM(\mathcal{L}(\mathcal{P}))$, which concludes the proof of the lemma. □

Theorem 1 (S-closure) *Let $\Box \in \Sigma_{int}$ and $\mathcal{P} = \langle Q, q_{in}, \Gamma, \Delta, F \rangle$ be an NVPA over $\widetilde{\Sigma}$. Then, one can construct in polynomial time an NVPA accepting $(\mathcal{L}(\mathcal{P}))^{\circlearrowleft \Box}$ with $|Q| + 2$ states and $|\Gamma| \cdot (|Q| + 2)$ stack symbols.*

Proof Let $\mathcal{P} = \langle Q, q_{in}, \Gamma, \Delta, F \rangle$ be a NVPA over $\widetilde{\Sigma}$ and $\mathcal{P}' = \langle Q', q'_{in}, \Gamma \cup \widehat{\Gamma}, \Delta', F' \rangle$ be the NVPA accepting $MWM(\mathcal{L}(P))$ given by Lemma 1, where $Q' \supseteq Q$ and $|Q'| = |Q| + 2$. The result follows from the construction of an NVPA $\mathcal{P}''$ accepting $(\mathcal{L}(\mathcal{P}'))^{\circlearrowleft \Box}$ with $|Q'|$ states and stack alphabet $\Gamma \cup \widehat{\Gamma} \cup Q \times \widehat{\Gamma}$. First, we informally describe the construction of $\mathcal{P}''$. Essentially, $\mathcal{P}''$ simulates $\mathcal{P}'$ step by step, but when $\mathcal{P}'$ performs an internal transition of the form $(q, \Box, p)$, then from the current state q, $\mathcal{P}''$ can choose to either process $\Box$ as $\mathcal{P}'$, or recursively process instead some guessed word $w \in \mathcal{L}(\mathcal{P}')^{\circlearrowleft \Box}$ as follows. $\mathcal{P}''$ guesses a call c that is the initial symbol of w, and chooses a push transition $(q'_{in}, c, q', \gamma) \in \Delta'$ from the initial state of $\mathcal{P}'$ (the construction of $\mathcal{P}'$ ensures that $\gamma \in \widehat{\Gamma}$). In the same step, $\mathcal{P}''$ pushes onto the stack both γ and the target state $p \in Q$ of the internal transition $(q, \Box, p)$ of $\mathcal{P}'$, and moves to state q'. This compound step allows $\mathcal{P}''$ to guarantee that when the stack is popped on reading the matching return of c (corresponding to the last symbol of the guessed word w), $\mathcal{P}''$ can restart the simulation of $\mathcal{P}'$ from the desired control state p. Moreover, when the matching return of c is read, $\mathcal{P}''$ guarantees that the pair (p, γ) is popped from the stack if and only if from the current state of $\mathcal{P}''$, there is some pop transition of $\mathcal{P}'$ which pops γ and leads to an accepting state q'_{acc} from F'. Note that since $\mathcal{P}'$ accepts only minimally well-matched words, the pair (p, γ) pushed onto the stack is eventually popped. Formally, the NVPA $\mathcal{P}''$ is given by $\mathcal{P}'' = \langle Q', q'_{in}, \Gamma \cup \widehat{\Gamma} \cup (Q \times \widehat{\Gamma}), \Delta'', F' \rangle$, where Δ'' is obtained from Δ' by adding the following transitions:

- *New Push transitions*: for each internal transition $(q, \Box, p) \in \Delta'$ – note that $p \in Q$ – and for each push transition from the initial state of the form $(q'_{in}, c, q', \gamma) \in \Delta'$ – note that $\gamma \in \widehat{\Gamma}$ –, we add the new push transition $(q, c, q', (p, \gamma))$.
- *New Pop transitions*: for each pop transition $(q, r, \gamma, q_{acc}) \in \Delta'$ which pops $\gamma \in \widehat{\Gamma}$ and leads to an accepting state $q_{acc} \in F'$, we add for each $p \in Q$, the new pop transition $(q, r, (p, \gamma), p)$.

It remains to show that the construction is correct, i.e., $\mathcal{L}(\mathcal{P}'') = (\mathcal{L}(\mathcal{P}'))^{\circlearrowleft \Box}$.

Inclusion $\mathcal{L}(\mathcal{P}'') \subseteq [\mathcal{L}(\mathcal{P}')]^{\circlearrowleft\square}$**:** let $w \in \mathcal{L}(\mathcal{P}'')$. Hence, there is an initialized accepting run π of $\mathcal{P}''$ over w. Let $N(\pi)$ be the number of steps along π obtained by executing a *new* push transition. We prove by induction on $N(\pi)$ that $w \in [\mathcal{L}(\mathcal{P}')]^{\circlearrowleft\square}$, hence, the result follows.

Base case: $N(\pi) = 0$. Since $\mathcal{L}(\mathcal{P}') \subseteq [\mathcal{L}(\mathcal{P}')]^{\circlearrowleft\square}$, it suffices to show that π is also an initialized accepting run of $\mathcal{P}'$ over w. This holds since the initial stack content in π is empty and the new push (resp., pop) transitions push (resp., pop) symbols which are not in the stack alphabet $\Gamma \cup \widehat{\Gamma}$ of $\mathcal{P}'$.

Induction step: $N(\pi) > 0$. By construction, π can be written in the form

$$\pi = (q'_{in}, \bot) \xrightarrow{w'} (q, \alpha) \xrightarrow{c} (q', (p, \gamma) \cdot \alpha) \xrightarrow{w''} (q_{acc}, \beta) \quad \text{with } w = w' \cdot c \cdot w'' \quad (1)$$

such that $q_{acc} \in F'$, $(q, c, q', (p, \gamma))$ is a new push transition, and the suffix π'' of π over w'' does *not* use new push transitions. We claim that the top (p, γ) of the initial stack content $(p, \gamma) \cdot \alpha$ of π'' is popped along π'', i.e., the first position in the word $c \cdot w''$ is a *matched* call position. We assume the contrary and derive a contradiction. It follows from this assumption that $c \cdot w'' \notin MWM(\widetilde{\Sigma})$ and by construction, π'' cannot use new transitions. Hence, we deduce that there is also a run of $\mathcal{P}'$ over w'' of the form $(q', \gamma \cdot \bot) \xrightarrow{w''} (q_{acc}, \beta')$ for some stack content β'. Since $(q, c, q', (p, \gamma))$ is a new push transition, by construction $(q'_{in}, c, q', \gamma) \in \Delta'$. Hence, we obtain that $c \cdot w'' \in \mathcal{L}(\mathcal{P}')$. This is a contradiction since $\mathcal{L}(\mathcal{P}') \subseteq MWM(\widetilde{\Sigma})$ and $c \cdot w'' \notin MWM(\widetilde{\Sigma})$. Thus, the claim holds, i.e. the initial stack content $(p, \gamma) \cdot \alpha$ of π'' is popped along π''. Therefore, π'' can be written in the form

$$\pi'' = (q', (p, \gamma) \cdot \alpha) \xrightarrow{u} (q'', (p, \gamma) \cdot \alpha) \xrightarrow{r} (p, \alpha) \xrightarrow{v} (q_{acc}, \beta), \;\; w'' = u \cdot r \cdot v \quad (2)$$

such that the prefix of π'' over u does not use *new* transitions, $u \in WM(\widetilde{\Sigma})$, and $(q'', r, (p, \gamma), p)$ is a new pop transition. By construction, $(q'', r, \gamma, q'_{acc}) \in \Delta'$ for some $q'_{acc} \in F'$. Since $u \in WM(\widetilde{\Sigma})$, we deduce that there is also a run of $\mathcal{P}'$ over u of the form $(q', \gamma \cdot \bot) \xrightarrow{u} (q'', \gamma \cdot \bot)$. Thus, since $(q'_{in}, c, q', \gamma) \in \Delta'$ (by construction and the fact that $(q, c, q', (p, \gamma))$ is a new push transition), it follows that $c \cdot u \cdot r \in \mathcal{L}(\mathcal{P}')$. Moreover, since $(q, \square, p) \in \Delta' \subseteq \Delta''$ (by construction and the fact that $(q, c, q', (p, \gamma))$ is a new push transition), by (1) and (2), there is an initialized accepting run π' of $\mathcal{P}''$ over $w' \cdot \square \cdot v$ such that $N(\pi') < N(\pi)$. By the induction hypothesis, $w' \cdot \square \cdot v \in [\mathcal{L}(\mathcal{P}')]^{\circlearrowleft\square}$. Thus, since $c \cdot u \cdot r \in \mathcal{L}(\mathcal{P}') \subseteq MWM(\widetilde{\Sigma})$ and $w = w' \cdot c \cdot u \cdot r \cdot v$, we obtain that $w \in [\mathcal{L}(\mathcal{P}')]^{\circlearrowleft\square}$ as we wanted to show.

Converse inclusion $[\mathcal{L}(\mathcal{P}')]^{\circlearrowleft\square} \subseteq \mathcal{L}(\mathcal{P}'')$**:** let $w \in [\mathcal{L}(\mathcal{P}')]^{\circlearrowleft\square}$. We show by induction on $|w|$ that $w \in \mathcal{L}(\mathcal{P}'')$ (note that $|w| \geq 2$), hence the result follows. If $w \in \mathcal{L}(\mathcal{P}') \subseteq MWM(\widetilde{\Sigma})$, the result is obvious, since $\Delta' \subseteq \Delta''$ (note that this case includes the base case $|w| = 2$). Now, assume that $w \notin \mathcal{L}(\mathcal{P}')$. Then, there is $u \in \mathcal{L}(\mathcal{P}') \subseteq MWM(\widetilde{\Sigma})$ such that w is of the form $w = w' \cdot u \cdot w''$ and $w' \cdot \square \cdot w'' \in [\mathcal{L}(\mathcal{P}')]^{\circlearrowleft\square}$. By the induction hypothesis, $w' \cdot \square \cdot w'' \in \mathcal{L}(\mathcal{P}'')$. Hence, there is an initialized accepting run of $\mathcal{P}''$ over $w' \cdot \square \cdot w''$ of the form

$(q'_{in}, \bot) \xrightarrow{w'} (q, \alpha) \xrightarrow{\Box} (p, \alpha) \xrightarrow{w''} (q_{acc}, \beta)$ such that $(q, \Box, p) \in \Delta'$. Thus, it is sufficient to show that there is a run of $\mathcal{P}''$ over u of the form $(q, \bot) \xrightarrow{u} (p, \bot)$. Since $u \in \mathcal{L}(\mathcal{P}') \subseteq MWM(\widetilde{\Sigma})$, and $(q, \Box, p) \in \Delta'$, by construction, the result follows. This concludes the proof of the theorem. □

Theorem 2 (M-substitution and M-closure) *Let $\mathcal{P} = \langle Q, q_{in}, \Gamma, \Delta, F\rangle$ and $\mathcal{P}' = \langle Q', q'_{in}, \Gamma', \Delta', F'\rangle$ be two NVPA over $\widetilde{\Sigma}$, and $\Box \in \Sigma_{int}$. Then, one can construct in polynomial time:*

1. *an NVPA accepting $\mathcal{L}(\mathcal{P}) \curvearrowleft_{\Box} \mathcal{L}(\mathcal{P}')$ with $|Q| + |Q'|$ states and at most $|\Gamma \cup \Gamma'| + |\Gamma'| \cdot |Q|$ stack symbols.*
2. *an NVPA accepting $(\mathcal{L}(\mathcal{P}))^{\curvearrowright \Box}$ with $2|Q|+2$ states and $2|\Gamma| \cdot (|Q|+1)$ stack symbols.*

Proof **Proof of Condition 1:** let $\Box \in \Sigma_{int}$ and $\mathcal{P} = \langle Q, q_{in}, \Gamma, \Delta, F\rangle$ and $\mathcal{P}' = \langle Q', q'_{in}, \Gamma', \Delta', F'\rangle$ as in the statement of the theorem. Without loss of generality we assume that Q and Q' are disjoint, and we denote by Γ'_0 the set of symbols pushed by the push transitions of $\mathcal{P}'$ from the initial state. Assume that $\Box \notin \mathcal{L}(\mathcal{P}')$ (the other case being similar). Note that this condition can be trivially checked. Let Δ_0 obtained from Δ by removing every internal transition of the form $(q, \Box, p)$. We construct an NVPA $\mathcal{P}''$ over $\widetilde{\Sigma}$ accepting $\mathcal{L}(\mathcal{P}) \curvearrowleft_{\Box} \mathcal{L}(\mathcal{P}')$ as follows: $\mathcal{P}'' = \langle Q \cup Q', q_{in}, \Gamma \cup \Gamma' \cup (Q \times \Gamma'_0), \Delta'', F\rangle$, where Δ'' is obtained from $\Delta_0 \cup \Delta'$ by adding the following transitions.

- *New Push transitions*: for each internal transition of $\mathcal{P}$ of the form $(q, \Box, p) \in \Delta$ and for each push transition of $\mathcal{P}'$ from the initial state of the form $(q'_{in}, c, q', \gamma') \in \Delta'$ – note that $\gamma' \in \Gamma'_0$ –, we add the new push transition $(q, c, q', (p, \gamma'))$.
- *New Pop transitions*: for each pop transition $(q', r, \gamma', q'_{acc}) \in \Delta'$ of $\mathcal{P}'$ which pops $\gamma' \in \Gamma'_0$ and leads to an accepting state $q'_{acc} \in F'$, we add for each $p \in Q$, the new pop transition $(q', r, (p, \gamma'), p)$.

The proof of correctness is a simplified version of that given in Theorem 1, so we omit it.

Proof of Condition 2: let $\Box \in \Sigma_{int}$ and $\mathcal{P} = \langle Q, q_{in}, \Gamma, \Delta, F\rangle$ be an NVPA. By the proof of Theorem 1, one can construct an NVPA $\mathcal{P}'$ accepting $\mathcal{L}(\mathcal{P})^{\circlearrowleft \Box}$ with $|Q| + 2$ states and stack alphabet $\Gamma \cup \widehat{\Gamma} \cup (Q \times \widehat{\Gamma})$, where $\widehat{\Gamma}$ is a fresh copy of Γ. Moreover, each push transition of $\mathcal{P}'$ from the initial state pushes a symbol in $\widehat{\Gamma}$. Thus, by the proof given for Condition 1, it follows that one can construct a NVPA $\mathcal{P}''$ with $2|Q| + 2$ states and $2|\Gamma| \cdot (|Q| + 1)$ stack symbols accepting $\mathcal{L}(\mathcal{P}) \curvearrowleft_{\Box} ([\mathcal{L}(\mathcal{P})]^{\circlearrowleft \Box} \cup \{\Box\}) = [\mathcal{L}(\mathcal{P})]^{\curvearrowright \Box}$. This concludes the proof of the theorem. □

3.1 VRE and Equivalence Between VRE and NVPA

Definition 3 The syntax of *Visibly Rational Expressions* (VRE) E over the pushdown alphabet $\widetilde{\Sigma}$ is inductively defined as follows:

$$E := \emptyset \mid \varepsilon \mid \sigma \mid (E \cup E) \mid (E \cdot E) \mid E^* \mid (E \curvearrowleft_{\Box} E) \mid E^{\circlearrowleft \Box} \mid E^{\curvearrowright \Box}$$

where $\sigma \in \Sigma$ and $\Box \in \Sigma_{int}$. A *pure* VRE is a VRE which does not contain occurrences of the M-closure operator $^{\curvearrowleft\Box}$. A VRE E denotes a language $\mathcal{L}(E)$ of finite words over Σ, defined as follows:

- $\mathcal{L}(\emptyset) = \emptyset$,
- $\mathcal{L}(\varepsilon) = \{\varepsilon\}$,
- $\mathcal{L}(\sigma) = \{\sigma\}$ for each $\sigma \in \Sigma$,
- $\mathcal{L}(E_1 \cup E_2) = \mathcal{L}(E_1) \cup \mathcal{L}(E_2)$,
- $\mathcal{L}(E_1 \cdot E_2) = \mathcal{L}(E_1) \cdot \mathcal{L}(E_2)$,
- $\mathcal{L}(E^*) = [\mathcal{L}(E)]^*$,
- $\mathcal{L}(E_1 \curvearrowleft_\Box E_2) = \mathcal{L}(E_1) \curvearrowleft_\Box \mathcal{L}(E_2)$,
- $\mathcal{L}(E^{\circlearrowleft\Box}) = [\mathcal{L}(E)]^{\circlearrowleft\Box}$, and
- $\mathcal{L}(E^{\curvearrowleft\Box}) = [\mathcal{L}(E)]^{\curvearrowleft\Box}$.

As usual, the size $|E|$ of a VRE E is the length of the string describing E.

Remark 1 By Definition 2, the M-closure operator is a derived operator. Hence, pure VRE and unrestricted VRE capture the same class of languages.

It is known [1] that for the class of regular languages (over a pushdown alphabet), NVPA can be exponentially more succinct than nondeterministic finite-state automata (NFA). We establish an analogous result for VRE and regular expressions.

Theorem 3 *There are a pushdown alphabet $\widetilde{\Sigma}$ and a family $\{\mathcal{L}_n\}_{n\geq 1}$ of regular languages over $\widetilde{\Sigma}$ such that for each $n \geq 1$, $\mathcal{L}_n$ can be denoted by a VRE of size $O(n)$ and every regular expression denoting $\mathcal{L}_n$ has size at least $2^{\Omega(n)}$.*

Proof Let $\widetilde{\Sigma} = \langle \Sigma_{call}, \Sigma_{ret}, \{\Box\}\rangle$ with $\Sigma_{call} = \{c_1, c_2\}$ and $\Sigma_{ret} = \{r_1, r_2\}$. For $n \geq 1$, let $\mathcal{L}_n$ be the finite (hence, regular) language $\{c_{i_1}c_{i_2}\ldots c_{i_n}r_{i_n}\ldots r_{i_2}r_{i_1} \mid i_1, \ldots, i_n \in \{1,2\}\}$. Evidently, $\mathcal{L}_n$ can be expressed by the VRE of size $O(n)$ given by $\underbrace{E \curvearrowleft_\Box E \curvearrowleft_\Box \ldots \curvearrowleft_\Box E}_{n-1 \text{ times}} \curvearrowleft_\Box (c_1 \cdot r_1 \cup c_2 \cdot r_2)$, where $E = (c_1 \cdot \Box \cdot r_1 \cup c_2 \cdot \Box \cdot r_2)$. However, as shown in [1], any NFA accepting $\mathcal{L}_n$ requires at least 2^n states. Thus, since regular expressions can be converted in linear time into equivalent NFA, the result follows. □

In the following, we show that VRE and NVPA are effectively language equivalent. First, we recall the following known result [4].

Theorem 4 (From [4]) *Let $\mathcal{P} = \langle Q, q_{in}, \Gamma, \Delta, F\rangle$ and $\mathcal{P}' = \langle Q', q'_{in}, \Gamma', \Delta', F'\rangle$ be two NVPA over $\widetilde{\Sigma}$. Then, one can construct in linear time:*

1. *an NVPA accepting $\mathcal{L}(\mathcal{P}) \cup \mathcal{L}(\mathcal{P}')$ (resp. $\mathcal{L}(\mathcal{P}) \cdot \mathcal{L}(\mathcal{P}')$) with $|Q| + |Q'|$ states and $|\Gamma| + |\Gamma'|$ stack symbols.*
2. *an NVPA accepting $[\mathcal{L}(\mathcal{P})]^*$ with $2|Q|$ states and $2|\Gamma|$ stack symbols.*

By Theorems 1, 2, and 4, a given VRE can be effectively and compositionally translated into an equivalent NVPA. However, due to Condition 2 in

Theorem 2 (concerning the M-closure operator) and Condition 2 in Theorem 4 (concerning the Kleene closure operator), the translation can involve a singly exponential blow-up. In the next section, we show that this exponential blow-up is due essentially to the presence of the M-closure operator (in particular, we propose a quadratic time translation of *pure* VRE into equivalent NVPA).

Corollary 1 *Given a VRE E, one can construct in* singly exponential time *an NVPA accepting $\mathcal{L}(E)$.*

Now, we show that any NVPA can be converted into an equivalent VRE. First, we need some additional notation. Let $\mathcal{P} = \langle Q, q_{in}, \Gamma, \Delta, F\rangle$ be an NVPA over a pushdown alphabet $\widetilde{\Sigma}$. Given $p, p' \in Q$, a *summary* of $\mathcal{P}$ from p to p' is a run π of $\mathcal{P}$ over some word $w \in MWM(\widetilde{\Sigma})$ from a configuration of the form (p, 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 $w \in MWM(\widetilde{\Sigma})$ from (p, 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 $(p, \bot)$ 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 $q, q' \in Q$, if there is subrun of π which is a summary from q to q', then $(q, q') \in \mathcal{S}$. Given a finite alphabet Λ disjoint from Σ, we denote by $\widetilde{\Sigma}_\Lambda$ the pushdown alphabet $\langle \Sigma_{call}, \Sigma_{ret}, \Sigma_{int} \cup \Lambda\rangle$ obtained by interpreting the elements in Λ as internal actions.

Theorem 5 *Given an NVPA $\mathcal{P}$, one can construct in single exponential time a VRE E such that $\mathcal{L}(E) = \mathcal{L}(\mathcal{P})$.*

Proof Let $\mathcal{P} = \langle Q, q_{in}, \Gamma, \Delta, F\rangle$. We construct a finite alphabet Λ disjoint from Σ and a VRE E over $\widetilde{\Sigma}_\Lambda$ denoting $\mathcal{L}(\mathcal{P})$; the additional symbols in Λ are used only as parameters for intermediate substitutions. The alphabet Λ is given by $\{\Box_{pp'} \mid p, p' \in Q\}$. Moreover, we define $\mathcal{P}_\Lambda = \langle Q, q_{in}, \Gamma, \Delta_\Lambda, F\rangle$ as the NVPA over $\widetilde{\Sigma}_\Lambda$ obtained from $\mathcal{P}$ by adding for each $(p, p') \in Q \times Q$, the internal transition $(p, \Box_{pp'}, p')$. Given $q, q' \in Q$, $\mathcal{S} \subseteq Q \times Q$, and $\Lambda' \subseteq \Lambda$, we define $R(q, q', \mathcal{S}, \Lambda')$ as the language of finite words w over $\Sigma_{\Lambda'}$ ($\Sigma_{\Lambda'}$ is the support of $\widetilde{\Sigma}_{\Lambda'}$) such that there is a run of $\mathcal{P}_\Lambda$ over w from $(q, \bot)$ to some configuration of the form (q', β) which uses only sub-summaries from $\mathcal{S}$.[2] By construction, $\mathcal{L}(\mathcal{P})$ is the union of the sets $R(q, q', Q \times Q, \emptyset)$ such that $q = q_{in}$ and $q' \in F$. Thus, Theorem 5 is a consequence of the following fact: for all $q, q' \in Q$, $\mathcal{S} \subseteq Q \times Q$, and $\Lambda' \subseteq \Lambda$, the languages $R(q, q', \mathcal{S}, \Lambda')$ and $WM(R(q, q', \mathcal{S}, \Lambda'))$ can be effectively denoted by VRE of sizes singly exponential in the size of $\mathcal{P}$, where $WM(R(q, q', \mathcal{S}, \Lambda')) \stackrel{\text{def}}{=} R(q, q', \mathcal{S}, \Lambda') \cap WM(\widetilde{\Sigma})$. The proof of this fact proceeds by induction on the cardinality of the finite set $\mathcal{S}$.

Base case: $\mathcal{S} = \emptyset$. We show that the languages $WM(R(q, q', \emptyset, \Lambda'))$ and $R(q, q', \emptyset, \Lambda')$ can be effectively denoted by regular expressions over $\Sigma_{\Lambda'}$ of

[2] note that if $w \in WM(\widetilde{\Sigma}_{\Lambda'})$, then $\beta = \bot$.

sizes singly exponential in the size of $\mathcal{P}$. The result follows because a regular expression is also a VRE. For each $i = 0, 1, 2$, let $\mathcal{A}_i = \langle Q, q_{in}, \Delta_i, F\rangle$ be the nondeterministic finite-state automaton (NFA) over $\Sigma_{\Lambda'}$, where Δ_i is defined as follows:

- Δ_0 is obtained from the transition relation of $\mathcal{P}_\Lambda$ by removing all the push and pop transitions, and the internal transitions $(p, \square_{pp'}, p')$ with $\square_{pp'} \notin \Lambda'$;
- Δ_1 is obtained from Δ_0 by adding for each pop transition of $\mathcal{P}$ of the form $(q, r, \bot, q')$, the transition (q, r, q');
- Δ_2 is obtained from Δ_0 by adding for each push transition (q, c, q', γ) of $\mathcal{P}$, the transition (q, c, q').

First, consider $WM(R(q, q', \emptyset, \Lambda'))$. Note that no word in $WM(R(q, q', \emptyset, \Lambda'))$ contains occurrences of calls or occurrences of returns. Hence, $WM(R(q, q', \emptyset, \Lambda'))$ is the set of words w such that there is a run of $\mathcal{A}_0$ over w from q to q'. Kleene's theorem [15] for regular expressions over words guarantees that $WM(R(q, q', \emptyset, \Lambda'))$ can be *effectively* denoted by a regular expression over $\Sigma_{\Lambda'}$ of size singly exponential in the size of $\mathcal{P}$. Hence, the result for $WM(R(q, q', \emptyset, \Lambda'))$ follows. Now, let us consider the language $R(q, q', \emptyset, \Lambda')$. Note that each word $w \in R(q, q', \emptyset, \Lambda')$ is of the form $w = w' \cdot w''$ such that w' does not contain occurrences of calls and w'' does not contain occurrences of returns. Hence, $R(q, q', \emptyset, \Lambda') = \bigcup_{p\in Q} \mathcal{L}_{1,q,p} \cdot \mathcal{L}_{2,p,q'}$, where $\mathcal{L}_{1,q,p}$ (resp., $\mathcal{L}_{2,p,q'}$) is the set of words w such that there is a run of the NFA $\mathcal{A}_1$ (resp., NFA $\mathcal{A}_2$) over w from q to p (resp., p to q'). Thus, from the Kleene theorem for regular expressions over words, the result for $R(q, q', \emptyset, \Lambda')$ holds as well.

Induction step: $\mathcal{S} = \mathcal{S}' \cup \{(p, p')\}$ with $(p, p') \notin \mathcal{S}'$. Let $P_{p\to p'}$ be the set:

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

So, $P_{p\to p'}$ is the set of tuples (s, c, r, s') such that there is a push transition from p to s reading the call c and a matching pop transition from s' to p' reading r. Moreover, let $S(p, p', \mathcal{S}' \cup \{(p, p')\}, \Lambda')$ be the language over $\Sigma_{\Lambda'}$ defined as follows:

$$\begin{aligned} S(p, p', \mathcal{S}' \cup \{(p, p')\}, \Lambda') \stackrel{\text{def}}{=} \\ \Big(\Big[\bigcup_{(s,c,r,s')\in P_{p\to p'}} \{c\} \cdot WM(R(s, s', \mathcal{S}', \Lambda' \cup \{\square_{pp'}\})) \cdot \{r\}\Big]^{\curvearrowleft \square_{pp'}}\Big) \curvearrowleft \square_{pp'} \\ \Big[\bigcup_{(s,c,r,s')\in P_{p\to p'}} \{c\} \cdot WM(R(s, s', \mathcal{S}', \Lambda')) \cdot \{r\}\Big] \end{aligned}$$

Note that $S(p, p', \mathcal{S}' \cup \{(p, p')\}, \Lambda')$ represents the set of words $w \in MWM(\widetilde{\Sigma}_{\Lambda'})$ such that there is a summary of $\mathcal{P}_\Lambda$ over w from p to p' which uses only sub-summaries from $\mathcal{S}' \cup \{(p, p')\}$. Moreover, by the induction hypothesis, the sets $WM(R(s, s', \mathcal{S}', \Lambda' \cup \{\square_{pp'}\}))$ and $WM(R(s, s', \mathcal{S}', \Lambda'))$ used in the definition of $S(p, p', \mathcal{S}' \cup \{(p, p')\}, \Lambda')$ can be effectively denoted by VRE. Thus, one can

construct a VRE over $\widetilde{\Sigma}_\Lambda$ denoting the language $S(p, p', \mathcal{S}' \cup \{(p, p')\}, \Lambda')$. Now, we observe that:

$$WM(R(q, q', \mathcal{S}' \cup \{(p, p')\}, \Lambda')) = WM(R(q, q', \mathcal{S}', \Lambda')) \cup WM(R(q, q', \mathcal{S}', \Lambda' \cup \{\Box_{pp'}\})) \curvearrowleft_{\Box_{pp'}} S(p, p', \mathcal{S}' \cup \{(p, p')\}, \Lambda')$$

$$R(q, q', \mathcal{S}' \cup \{(p, p')\}, \Lambda') = R(q, q', \mathcal{S}', \Lambda') \cup R(q, q', \mathcal{S}', \Lambda' \cup \{\Box_{pp'}\}) \curvearrowleft_{\Box_{pp'}} S(p, p', \mathcal{S}' \cup \{(p, p')\}, \Lambda')$$

Thus, by the induction hypothesis, the languages $WM(R(q, q', \mathcal{S}' \cup \{(p, p')\}, \Lambda'))$ and $R(q, q', \mathcal{S}' \cup \{(p, p')\}, \Lambda')$ can be effectively denoted by VRE. Moreover, by expanding recursively the above equalities until the base case (the number of iterations is at most $|Q|^2$), we deduce that each of the constructed VRE has size singly exponential in the size of the NVPA $\mathcal{P}$. This concludes the proof of the theorem. □

Corollary 1 and Theorem 5 precisely identify the expressive power of VRE.

Corollary 2 *(Pure) Visibly Rational Expressions capture the class of VPL.*

4 Pure VRE

In this section we study pure VRE expressions.

First, we show that pure VRE can be compositionally translated in *quadratic time* into equivalent NVPA. The key of the proposed efficient and elegant translation is represented by a subclass of NVPA, that we call *strong* NVPA. Then, in Subsection 4.1, we establish the exact complexity of some language decision problems for pure VRE. In the reminder of this section, we fix a pushdown alphabet $\widetilde{\Sigma}$ and we use an additional special stack symbol $\widehat{\perp}$.

Definition 4 A *strong* NVPA over $\widetilde{\Sigma}$ is an NVPA $\mathcal{P} = \langle Q, q_{in}, \Gamma, \Delta, F \rangle$ over $\widetilde{\Sigma}$ such that $\widehat{\perp} \in \Gamma$ and the following holds:

- *Initial State Requirement*: $q_{in} \notin F$ and there are no transitions leading to q_{in}.
- *Final State Requirement*: there are no transitions from accepting states.
- *Push Requirement*: every push transition from the initial state q_{in} pushes onto the stack the special symbol $\widehat{\perp}$.
- *Pop Requirement*: for all $q, p \in Q$ and $r \in \Sigma_{ret}$, $(q, r, \perp, p) \in \Delta$ if and only if $(q, r, \widehat{\perp}, p) \in \Delta$.
- *Well-formed (semantic) Requirement*: for all $w \in \mathcal{L}(\mathcal{P})$, every initialized accepting run of $\mathcal{P}$ over w leads to a configuration whose stack content is in $\{\widehat{\perp}\}^* \cdot \perp$.

Note that the initial state requirement implies that $\varepsilon \notin \mathcal{L}(\mathcal{P})$.

The push requirement is used in particular to implement in an efficient way M-substitution and S-closure. The pop requirement ensures that pop operations which pop the special stack symbol $\widehat{\perp}$ have the same effect as popping the empty stack (i.e., the stack containing just the special bottom symbol $\perp$). This requirement and the well-formed requirement are used in particular to implement in an efficient way concatenation and Kleene closure. We first show efficient constructions for strong NVPA for the operations of union, concatenation, and Kleene closure.

Theorem 6 *Let $\mathcal{P} = \langle Q, q_{in}, \Gamma, \Delta, F\rangle$ and $\mathcal{P}' = \langle Q', q'_{in}, \Gamma', \Delta', F'\rangle$ be two strong NVPA over $\widetilde{\Sigma}$. Then, one can construct in linear time*

1. *a strong NVPA accepting $[\mathcal{L}(\mathcal{P})]^* \setminus \{\varepsilon\}$ with $|Q|+1$ states and $|\Gamma|$ stack symbols, and*
2. *a strong NVPA accepting $\mathcal{L}(\mathcal{P})\cup\mathcal{L}(\mathcal{P}')$ (resp., $\mathcal{L}(\mathcal{P})\cdot\mathcal{L}(\mathcal{P}')$) with $|Q|+|Q'|+1$ (resp., $|Q|+|Q'|$) states and $|\Gamma|+|\Gamma'|-1$ stack symbols.*

Proof **Proof of Condition 1.** Let $\mathcal{P} = \langle Q, q_{in}, \Gamma, \Delta, F\rangle$ be a strong NVPA over $\widetilde{\Sigma}$ an q'_{in} be a fresh control state. Define $\mathcal{P}' = \langle Q \cup \{q'_{in}\}, q'_{in}, \Gamma, \Delta', F\rangle$, where Δ' is obtained from Δ by adding new transitions as follows. First, for each transition $t \in \Delta$ leading to an accepting state, we add the transition obtained from t by replacing the target state of t with the previous initial state q_{in}. Let Δ_0 be the resulting set of transitions. Then, Δ' is obtained from Δ_0 by adding the following transitions: for each transition $t \in \Delta_0$ from the previous initial state q_{in}, we add the transition obtained from t by replacing the source state of t with the new initial state q'_{in}. Note that the construction is identical to the classical construction for regular languages. We need to prove that $\mathcal{P}'$ is a *strong* NVPA accepting $[\mathcal{L}(\mathcal{P})]^* \setminus \{\varepsilon\}$. Since $\mathcal{P}$ is a strong NVPA, by construction, $\mathcal{P}'$ satisfies the initial and final state requirements and the push and pop requirements of Definition 4. Thus, it only remains to be shown that $\mathcal{L}(\mathcal{P}') = [\mathcal{L}(\mathcal{P})]^* \setminus \{\varepsilon\}$ and $\mathcal{P}'$ satisfies the well-formed requirement.

First, we show that $\mathcal{L}(\mathcal{P}') \subseteq [\mathcal{L}(\mathcal{P})]^* \setminus \{\varepsilon\}$ and $\mathcal{P}'$ satisfies the well-formed requirement. Let $w \in \mathcal{L}(\mathcal{P}')$ (note that $w \neq \varepsilon$ since $\mathcal{P}'$ satisfies the initial state requirement) and π be an initialized accepting run of $\mathcal{P}'$ over w of the form $(q'_{in}, \perp) \xrightarrow{w} (q_{acc}, \beta)$ for some stack content β and $q_{acc} \in F$. We need to show that $w \in [\mathcal{L}(\mathcal{P})]^* \setminus \{\varepsilon\}$ and $\beta \in \{\widehat{\perp}\}^* \cdot \perp$. By construction, w is of the form $w = w_1 \cdot \ldots \cdot w_n$ for some $n \geq 1$, such that $w_1, \ldots, w_n$ are non-empty and there are runs $\pi_1, \ldots, \pi_n$ of $\mathcal{P}$ over $w_1, \ldots, w_n$, respectively, of the form

$$\pi_1 = (q_{in}, \perp) \xrightarrow{w_1} (p_1, \beta_1), \;\; \pi_2 = (q_{in}, \beta_1) \xrightarrow{w_2} (p_2, \beta_2), \;\; \ldots \;\; ,$$
$$\pi_n = (q_{in}, \beta_{n-1}) \xrightarrow{w_n} (p_n, \beta_n)$$

where $p_i \in F$ for each $1 \leq i \leq n$, and $\beta = \beta_n$. We show by induction on i that $w_i \in \mathcal{L}(\mathcal{P})$ and $\beta_i \in \{\widehat{\perp}\}^* \cdot \perp$ for all $1 \leq i \leq n$. Since π_1 is an initialized accepting run of $\mathcal{P}$ over w_1 and $\mathcal{P}$ satisfies the well-formed requirement, the result for the base case holds. For the induction step, let us consider the run $\pi_i = (q_{in}, \beta_{i-1}) \xrightarrow{w_i} (p_i, \beta_i)$ with $i > 1$. By the induction hypothesis, $\beta_{i-1} \in \{\widehat{\perp}\}^* \cdot \perp$. Moreover, β_i is of the form $\beta_i = \beta'_i \cdot \{\widehat{\perp}\}^m \cdot \perp$ for some $m \geq 0$,

where β'_i consists of the symbols pushed on the stack along π_i on reading the unmatched call positions of w_i. Since $\mathcal{P}$ satisfies the pop requirement, we easily deduce that there is also an *initialized accepting* run of $\mathcal{P}$ over w_i of the form $\pi_i = (q_{in}, \bot) \xrightarrow{w_i} (p_i, \beta'_i \cdot \bot)$. Since $\mathcal{P}$ satisfies the well-formed requirement, $\beta'_i \in \{\widehat{\bot}\}^* \cdot \bot$, and we are done.

It remains to show that $[\mathcal{L}(\mathcal{P})]^* \setminus \{\varepsilon\} \subseteq \mathcal{L}(\mathcal{P}')$. Let $w \in [\mathcal{L}(\mathcal{P})]^* \setminus \{\varepsilon\}$. We need to prove that $w \in \mathcal{L}(\mathcal{P}')$. Since $\mathcal{P}$ satisfies the well-formed requirement, by construction, w can be written in the form $w = w_1 \cdot \ldots \cdot w_n$ for some $n \geq 1$, such that there are runs $\pi_1, \ldots, \pi_n$ of $\mathcal{P}'$ over $w_1, \ldots, w_n$, respectively, of the form

$$\pi_1 = (q_{in}, \bot) \xrightarrow{w_1} (q_{in}, \beta_1 \cdot \bot), \;\; \pi_2 = (q_{in}, \bot) \xrightarrow{w_2} (q_{in}, \beta_2 \cdot \bot), \;\; \ldots \;\; , \quad \pi_n = (q_{in}, \bot) \xrightarrow{w_n} (q_{acc}, \beta_n \cdot \bot)$$

where $q_{acc} \in F$, and $\beta_i \in \{\widehat{\bot}\}^*$ for each $1 \leq i \leq n$. Since $\mathcal{P}'$ satisfies the pop requirement, we easily deduce that for all $1 < i \leq n$ and $\beta \in \{\widehat{\bot}\}^* \cdot \bot$, there are $\beta' \in \{\widehat{\bot}\}^* \cdot \bot$ and a run of $\mathcal{P}'$ over w_i of the form $(q_{in}, \beta) \xrightarrow{w_i} (p, \beta_i \cdot \beta')$ where $p = q_{in}$ if $i < n$, and $p = q_{acc}$ otherwise. Hence, the existence of an initialized accepting run of $\mathcal{P}'$ over $w = w_1 \cdot \ldots \cdot w_n$ follows, which concludes the proof of Condition 1 of Theorem 6.

Proof of Condition 2. Let $\mathcal{P} = \langle Q, q_{in}, \Gamma, \Delta, F \rangle$ and $\mathcal{P}' = \langle Q', q'_{in}, \Gamma', \Delta', F' \rangle$ be two strong NVPA over $\widetilde{\Sigma}$. Without loss of generality, we assume that Q and Q' are disjoint. Let q''_{in} be a fresh control state. The strong NVPA accepting $\mathcal{L}(\mathcal{P}) \cup \mathcal{L}(\mathcal{P}')$ is given by $\mathcal{P}' = \langle Q \cup Q' \cup \{q''_{in}\}, q''_{in}, \Gamma \cup \Gamma', \Delta'', F \cup F' \rangle$, where Δ'' is obtained from $\Delta \cup \Delta'$ by adding the following transitions: for each transition $t \in \Delta$ (resp., $t \in \Delta'$) from the initial state q_{in} (resp., q'_{in}), we add the transition obtained from t by replacing the source state of t with the new initial state q''_{in}. Correctness of the construction follows. Finally, the strong NVPA accepting $\mathcal{L}(\mathcal{P}) \cdot \mathcal{L}(\mathcal{P}')$ is given by $\mathcal{P}' = \langle Q \cup Q', q_{in}, \Gamma \cup \Gamma', \Delta'', F' \rangle$, where Δ'' is obtained from $\Delta \cup \Delta'$ by adding the following transitions: for each transition $t \in \Delta$ leading to an accepting state of $\mathcal{P}$, we add the transition obtained from t by replacing the target state of t with the initial state q'_{in} of $\mathcal{P}'$. The proof of correctness is a simplified version of that given for Condition 1. Note that the given constructions are identical to the classical constructions for regular languages. This concludes the proof of Theorem 6. □

Next we show that strong NVPA are "efficiently" closed under M-substitution and S-closure. For this, we need the following preliminary result.

Lemma 2 *Let $\mathcal{P} = \langle Q, q_{in}, \Gamma, \Delta, F \rangle$ be a strong NVPA over $\widetilde{\Sigma}$. Then, one can construct in linear time a strong NVPA over $\widetilde{\Sigma}$ accepting $MWM(\mathcal{L}(\mathcal{P}))$ with $|Q|$ states and $|\Gamma| + 1$ stack symbols.*

Proof Fix a strong NVPA $\mathcal{P} = \langle Q, q_{in}, \Gamma, \Delta, F \rangle$ over $\widetilde{\Sigma}$. We first remove those transitions from the initial state to some accepting state, and all pop and internal transitions outgoing from the initial state. These transitions can never be used along runs over words from $MWM(\mathcal{L}(\mathcal{P}))$. Let γ_0 be a fresh stack

symbol. We construct a strong NVPA $\mathcal{P}'$ over $\widetilde{\Sigma}$ as follows: $\mathcal{P}' = \langle Q, q_{in}, \Gamma \cup \{\gamma_0\}, \Delta', F\rangle$, where Δ' is obtained from Δ as follows:

1. For every transition $t \in \Delta$ whose target state is accepting, we remove t, unless t is a pop transition which pops either $\bot$ or $\widehat{\bot}$.
2. For every pop transition $t \in \Delta$ which pops $\bot$ and whose target is *not* accepting we remove t from Δ'.
3. For every pop transition $t \in \Delta$ which pops $\widehat{\bot}$ and whose target state is *not* accepting we replace t with the transition obtained from t by replacing the popped symbol $\widehat{\bot}$ with γ_0.
4. For every push transition $t \in \Delta$ which pushes $\widehat{\bot}$ and such that the source state is *not* q_{in} and the target state is *not accepting*, we replace t with the transition obtained from t by replacing the pushed symbol $\widehat{\bot}$ with γ_0.

Since $\mathcal{P}$ is a strong NVPA, by construction $\mathcal{P}'$ satisfies the initial and final state requirements and the push and the pop requirements of Definition 4. It suffices to show that $\mathcal{L}(\mathcal{P}') = MWM(\mathcal{L}(\mathcal{P}))$, because the well-formed requirement of $\mathcal{P}'$ follows from this equality. We first consider the inclusion $\mathcal{L}(\mathcal{P}') \subseteq MWM(\mathcal{L}(\mathcal{P}))$. Let $w \in \mathcal{L}(\mathcal{P}')$. Since there are no $\mathcal{P}'$-transitions from the initial state to an accepting state and $q_{in} \notin F$, it follows that $|w| > 1$. Since $\mathcal{P}'$ satisfies the push requirement and there are only push transitions in $\mathcal{P}'$ from the initial state, by Condition 1 in the construction, there is an initialized accepting run of $\mathcal{P}'$ over w of the form

$$\pi = (q_{in}, \bot) \xrightarrow{c} (q, \widehat{\bot} \cdot \bot) \xrightarrow{w'} (p, \alpha) \xrightarrow{r} (q_{acc}, \alpha') \qquad \text{with } w = c \cdot w' \cdot r$$

such that $(q_{in}, c, q, \widehat{\bot}) \in \Delta$ and the last step $(p, \alpha) \xrightarrow{r} (q_{acc}, \alpha')$ of π is obtained by popping either $\bot$ or $\widehat{\bot}$. Since $\mathcal{P}'$ satisfies the initial and final state requirements, the subrun π' of π corresponding to $(q, \widehat{\bot} \cdot \bot) \xrightarrow{w'} (p, \alpha)$ does not visit neither the initial state nor any accepting states. Hence, in particular, by Conditions 2 – 4 in the construction, the stack symbol $\widehat{\bot}$ is not popped or pushed along π'. Thus, we deduce that $w \in MWM(\widetilde{\Sigma})$, $\alpha = \widehat{\bot} \cdot \bot$, $\alpha' = \bot$, and $(p, r, \widehat{\bot}, q_{acc}) \in \Delta$. Moreover, by Conditions 2 – 4 in the construction, there is also a run of $\mathcal{P}$ over w' from $(q, \widehat{\bot} \cdot \bot)$ to $(p, \widehat{\bot} \cdot \bot)$. Thus, we obtain that $w \in MWM(\mathcal{L}(\mathcal{P}))$, and $\mathcal{L}(\mathcal{P}') \subseteq MWM(\mathcal{L}(\mathcal{P}))$, as desired.

For the converse inclusion $MWM(\mathcal{L}(\mathcal{P})) \subseteq \mathcal{L}(\mathcal{P}')$, let $w \in MWM(\mathcal{L}(\mathcal{P}))$. Then, w is of the form $w = c \cdot w' \cdot r$, where $c \in \Sigma_{call}$, $r \in \Sigma_{ret}$, and w' is a well-matched word. Moreover, $\mathcal{P}$ is a strong NVPA (in particular, $\mathcal{P}$ satisfies the push requirement), and consequently there is an initialized accepting run of $\mathcal{P}$ over w of the form $\pi = (q_{in}, \bot) \xrightarrow{c} (q, \widehat{\bot} \cdot \bot) \xrightarrow{w'} (p, \widehat{\bot} \cdot \bot) \xrightarrow{r} (q_{acc}, \bot)$ such that the subrun π' corresponding to $(q, \widehat{\bot} \cdot \bot) \xrightarrow{w'} (p, \widehat{\bot} \cdot \bot)$ does not visit the initial state or any accepting states. By construction, $(q_{in}, c, q, \widehat{\bot}) \in \Delta'$ and $(p, r, \widehat{\bot}, q_{acc}) \in \Delta'$. Moreover, w' does not contain unmatched returns, so the subrun π' of $\mathcal{P}$ does not use transitions which pop $\bot$. Thus, by Conditions 2 – 4 in the construction, there is also a run of $\mathcal{P}'$ over w' from $(q, \widehat{\bot} \cdot \bot)$ to

$(p, \widehat{\perp} \cdot \perp)$. Hence, we obtain that $w \in \mathcal{L}(\mathcal{P}')$, which concludes the proof of Lemma 2. □

Theorem 7 (S-closure) *Let $\mathcal{P} = \langle Q, q_{in}, \Gamma, \Delta, F\rangle$ be a strong NVPA over $\widetilde{\Sigma}$ and $\Box \in \Sigma_{int}$. One can construct in linear time a strong NVPA over $\widetilde{\Sigma}$ accepting $(\mathcal{L}(\mathcal{P}))^{\circlearrowleft\Box}$ with $|Q|$ states and $|Q| + |\Gamma| + 1$ stack symbols.*

Proof By Lemma 2, it suffices to show that given a strong NVPA $\mathcal{P} = \langle Q, q_{in}, \Gamma, \Delta, F\rangle$ over $\widetilde{\Sigma}$ such that $\mathcal{L}(\mathcal{P}) \subseteq MWM(\widetilde{\Sigma})$, one can construct a strong NVPA accepting $(\mathcal{L}(\mathcal{P}))^{\circlearrowleft\Box}$ with $|Q|$ states and $|Q| + |\Gamma|$ stack symbols. Fix such a strong NVPA $\mathcal{P} = \langle Q, q_{in}, \Gamma, \Delta, F\rangle$, where $\mathcal{L}(\mathcal{P}) \subseteq MWM(\widetilde{\Sigma})$. Without loss of generality we assume that Q and Γ are disjoint and all the transitions from the initial state are push transitions. We construct an NVPA $\mathcal{P}'$ over $\widetilde{\Sigma}$ as follows: $\mathcal{P}' = \langle Q, q_{in}, \Gamma \cup Q, \Delta', F\rangle$, where Δ' is obtained from Δ by adding the following transitions.

- *New Push transitions*: for each internal transition $(q, \Box, p) \in \Delta$ – note that $q \neq q_{in}$ and $p \neq q_{in}$ – and for each push transition from the initial state $(q_{in}, c, q', \widehat{\perp}) \in \Delta$, we add the new push transition (q, c, q', p).
- *New Pop transitions*: for each pop transition $(q, r, \widehat{\perp}, q_{acc}) \in \Delta$ which pops the special stack symbol $\widehat{\perp}$ and leads to an accepting state $q_{acc} \in F$, we add for each $p \in Q \setminus \{q_{in}\}$, the new pop transition (q, r, p, p).

It remains to be shown that $\mathcal{P}'$ is a strong NVPA over $\widetilde{\Sigma}$ accepting $(\mathcal{L}(\mathcal{P}))^{\circlearrowleft\Box}$. Since $\mathcal{P}$ is a strong NVPA, by construction, it easily follows that $\mathcal{P}'$ satisfies the initial and final state requirements, and the push and pop requirements of Definition 4. Thus, it remains to show that $\mathcal{P}'$ satisfies the well-formed requirement and $\mathcal{L}(\mathcal{P}') = (\mathcal{L}(\mathcal{P}))^{\circlearrowleft\Box}$. Since $(\mathcal{L}(\mathcal{P}))^{\circlearrowleft\Box} \subseteq MWM(\widetilde{\Sigma})$, the well-formed requirement for $\mathcal{P}'$ directly follows from the equality $\mathcal{L}(\mathcal{P}') = (\mathcal{L}(\mathcal{P}))^{\circlearrowleft\Box}$. The proof of $\mathcal{L}(\mathcal{P}') = (\mathcal{L}(\mathcal{P}))^{\circlearrowleft\Box}$ is analogous to the proof of Theorem 1. □

Theorem 8 (M-substitution) *Let $\Box \in \Sigma_{int}$ and $\mathcal{P} = \langle Q, q_{in}, \Gamma, \Delta, F\rangle$ and $\mathcal{P}' = \langle Q', q'_{in}, \Gamma', \Delta', F'\rangle$ be two strong NVPA over $\widetilde{\Sigma}$. Then, one can construct in linear time a strong NVPA over $\widetilde{\Sigma}$ accepting $\mathcal{L}(\mathcal{P})\curvearrowleft_{\Box}\mathcal{L}(\mathcal{P}')$ with $|Q| + |Q'|$ states and $|\Gamma| + |\Gamma'| + |Q|$ stack symbols.*

Proof Note that one can trivially check whether $\Box$ is in the language of an NVPA. Thus, by Lemma 2, it suffices to show that given two strong NVPA $\mathcal{P} = \langle Q, q_{in}, \Gamma, \Delta, F\rangle$ and $\mathcal{P}' = \langle Q', q'_{in}, \Gamma', \Delta', F'\rangle$ over $\widetilde{\Sigma}$ such that $\mathcal{L}(\mathcal{P}') \subseteq MWM(\widetilde{\Sigma})$, one can construct two strong NVPA accepting $\mathcal{L}(\mathcal{P})\curvearrowleft_{\Box}\mathcal{L}(\mathcal{P}')$ and $\mathcal{L}(\mathcal{P})\curvearrowleft_{\Box}(\mathcal{L}(\mathcal{P}') \cup \{\Box\})$, respectively, and both having $|Q| + |Q'|$ states and $|\Gamma| + |\Gamma'| + |Q| - 1$ stack symbols. We illustrate the construction for the strong NVPA accepting $\mathcal{L}(\mathcal{P})\curvearrowleft_{\Box}\mathcal{L}(\mathcal{P}')$ (the other case being similar). Without loss of generality, we assume that the sets Q, Q', $\Gamma \setminus \{\widehat{\perp}\}$, and $\Gamma' \setminus \{\widehat{\perp}\}$ are pairwise disjoint. Let Δ_0 be obtained from Δ by removing every internal transition of the form $(q, \Box, p)$. We construct an NVPA $\mathcal{P}''$ over $\widetilde{\Sigma}$ as follows: $\mathcal{P}'' = \langle Q \cup Q', q_{in}, \Gamma \cup \Gamma' \cup Q, \Delta'', F\rangle$, where Δ'' is obtained from $\Delta_0 \cup \Delta'$ by adding the following transitions.

- *New Push transitions*: for each internal transition of $\mathcal{P}$ of the form $(q, \Box, p) \in \Delta$ – note that $p \neq q_{in}$ – and for each push transition of $\mathcal{P}'$ from the initial state of the form $(q'_{in}, c, q', \widehat{\perp}) \in \Delta'$, we add the new push transition (q, c, q', p) if $q \neq q_{in}$ and the new push transition $(q_{in}, c, q', \widehat{\perp})$ otherwise.
- *New Pop transitions*: for each pop transition of $\mathcal{P}'$ of the form $(q', r, \widehat{\perp}, q'_{acc}) \in \Delta'$ which pops the special stack symbol $\widehat{\perp}$ and leads to an accepting state $q'_{acc} \in F'$, we add the following pop transitions: (i) for each $p \in Q \setminus \{q_{in}\}$, the new pop transition (q', r, p, p), and (ii) for each internal transition $(q_{in}, \Box, p) \in \Delta$, the new pop transitions $(q', r, \widehat{\perp}, p)$ and $(q', r, \perp, p)$ (this last transition is used just to ensure that the pop requirement is satisfied).

In order to conclude the proof of Theorem 8, we need to show the following.

Claim: $\mathcal{P}''$ is a strong NVPA over $\widetilde{\Sigma}$ accepting $\mathcal{L}(\mathcal{P}) \circlearrowleft_{\Box} \mathcal{L}(\mathcal{P}')$.

Proof of the claim: since $\mathcal{P}$ and $\mathcal{P}'$ are strong NVPA, by construction, it follows that $\mathcal{P}''$ satisfies the initial and final state requirements, and the push and pop requirements in Definition 4. It remains to show that $\mathcal{P}''$ satisfies the well-formed requirement and $\mathcal{L}(\mathcal{P}'') = \mathcal{L}(\mathcal{P}) \circlearrowleft_{\Box} \mathcal{L}(\mathcal{P}')$. First, let us consider the inclusion $\mathcal{L}(\mathcal{P}) \circlearrowleft_{\Box} \mathcal{L}(\mathcal{P}') \subseteq \mathcal{L}(\mathcal{P}'')$. Every transition of $\mathcal{P}$ which is not labeled by the internal action $\Box$ is also a transition of $\mathcal{P}''$. Moreover, $\mathcal{L}(\mathcal{P}') \subseteq MWM(\widetilde{\Sigma})$. Hence, it is sufficient to show that for all $(q, \Box, p) \in \Delta$ and $w \in \mathcal{L}(\mathcal{P}')$, there is a run of $\mathcal{P}''$ over w from $(q, \perp)$ to $(p, \perp)$, which follows directly by construction.

It remains to show that $\mathcal{L}(\mathcal{P}'') \subseteq \mathcal{L}(\mathcal{P}) \circlearrowleft_{\Box} \mathcal{L}(\mathcal{P}')$ and $\mathcal{P}''$ satisfies the well-formed requirement. In fact, we prove a stronger result. Let $w \in \Sigma^*$ and π be an initialized run of $\mathcal{P}''$ having the form $\pi = (q_{in}, \perp) \xrightarrow{w} (s, \beta)$ for some stack content β and $s \in Q$. Let $N(\pi)$ be the number of steps along π obtained by executing a *new* push transition. We prove by induction on $N(\pi)$ that there exists an initialized run of $\mathcal{P}$ having the form $(q_{in}, \perp) \xrightarrow{H(w)} (s, \beta)$ over a word $H(w)$ such that $w \in H(w) \circlearrowleft_{\Box} \mathcal{L}(\mathcal{P}')$. Hence, since $\mathcal{P}$ satisfies the well-formed requirement, the result follows.

Base case: $N(\pi) = 0$. Since the new pop transitions start at states which are not in Q, we deduce that π is an initialized run of $\mathcal{P}$ over w which does not use internal transitions of the form $(q, \Box, p) \in \Delta$. Hence, w does not contain occurrences of $\Box$. Thus, by setting $H(w) = w$, the result holds in this case.

Induction step: $N(\pi) > 0$. Let (q, c, q', p) be the last new push transition used by π, where $p \in Q \cup \{\widehat{\perp}\}$. Then, π can be written in the form

$$\pi = (q_{in}, \perp) \xrightarrow{w'} (q, \alpha) \xrightarrow{c} (q', p \cdot \alpha) \xrightarrow{w''} (s, \beta) \qquad \text{with } w = w' \cdot c \cdot w''$$

where the suffix π'' of π over w'' does *not* use new push transitions. Since $q \in Q$, by the induction hypothesis, there is an initialized run of $\mathcal{P}$ having the form $(q_{in}, \perp) \xrightarrow{H(w')} (q, \alpha)$ over a word $H(w') \in \Sigma^*$ such that $w' \in H(w') \circlearrowleft_{\Box} \mathcal{L}(\mathcal{P}')$. Thus, it suffices to show that there is a run of $\mathcal{P}$ having the form $(q, \alpha) \xrightarrow{H(c \cdot w'')} (s, \beta)$ over a word $H(c \cdot w'') \in \Sigma^*$ such that $c \cdot w'' \in$

$H(c \cdot w'')\circlearrowleft_{\Box}\mathcal{L}(\mathcal{P}')$. Since $q' \in Q'$ and $s \in Q$, by construction it follows that there is exactly one step of π'' which uses a new pop transition. Moreover, such a step pops p from the initial stack content $p \cdot \alpha$ of π'' (otherwise, by construction, it easily follows that the initial stack content of π'' is not popped along π'' and there is a prefix $u \cdot r$ of w'' such that $c \cdot u \cdot r \in \mathcal{L}(\mathcal{P}')$ and $c \cdot u \cdot r \notin MWM(\widetilde{\Sigma})$, which is a contradiction). Thus, there is a new pop transition (q'', r, p, p') matching the new push transition (q, c, q', p), and π'' can be written in the form

$$\pi'' = (q', p \cdot \alpha) \xrightarrow{u} (q'', p \cdot \alpha) \xrightarrow{r} (p', \alpha) \xrightarrow{v} (s, \beta) \qquad \text{with } w'' = u \cdot r \cdot v$$

where $u \in WM(\widetilde{\Sigma})$. Moreover, there is also a run of $\mathcal{P}'$ over the well-matched word u of the form $(q', \widehat{\perp} \cdot \perp) \xrightarrow{u} (q'', \widehat{\perp} \cdot \perp)$, and the suffix of π'' over v corresponds to a run of $\mathcal{P}$ which does not use internal transitions labeled by $\Box$. Hence, v does not contain occurrences of $\Box$. By construction $(q'_{in}, c, q', \widehat{\perp}) \in \Delta'$, $(q'', r, \widehat{\perp}, q'_{acc}) \in \Delta'$ for some $q'_{acc} \in F'$, and either $p = \widehat{\perp}$, $q = q_{in}$, and $(q_{in}, \Box, p') \in \Delta$, or $p = p' \in Q$ and $(q, \Box, p) \in \Delta$. Hence, $(q, \Box, p') \in \Delta$. It follows that $c \cdot u \cdot r \in \mathcal{L}(\mathcal{P}')$, there is a run of $\mathcal{P}$ over $\Box \cdot v$ of the form $(q, \alpha) \xrightarrow{\Box \cdot v} (s, \beta)$, and $c \cdot w'' = c \cdot u \cdot r \cdot v \in \Box \cdot v\circlearrowleft_{\Box}\mathcal{L}(\mathcal{P}')$, and we are done. This concludes the proof of the claim, and hence of Theorem 8. □

Now, we can prove the main result of this section.

Theorem 9 *Let E be a* pure *VRE. Then, one can construct in quadratic time an NVPA $\mathcal{P}$ accepting $\mathcal{L}(E)$ with at most $|E|+1$ states and $|E|^2$ stack symbols.*

Proof Since one can trivially check in linear time whether $\varepsilon \in \mathcal{L}(E)$, it suffices to show that one can construct in quadratic time a *strong* NVPA accepting $\mathcal{L}(E) \setminus \{\varepsilon\}$ with at most $|E| + 1$ states and $|E|^2$ stack symbols. The proof is by induction on $|E|$. The base case holds immediately. For the induction step, the result follows from the induction hypothesis and Theorems 6, 7 and 8. As an example, we illustrate the case $E = E_1\circlearrowleft_{\Box}E_2$. By the induction hypothesis, one can construct two strong NVPA $\mathcal{P}_1 = \langle Q_1, q^1_{in}, \Gamma_1, \Delta_1, F_1\rangle$ and $\mathcal{P}_2 = \langle Q_2, q^2_{in}, \Gamma_2, \Delta_2, F_2\rangle$ accepting $\mathcal{L}(E_1) \setminus \{\varepsilon\}$ and $\mathcal{L}(E_2) \setminus \{\varepsilon\}$, respectively. Moreover, $|Q_1| \leq |E_1| + 1$, $|Q_2| \leq |E_2| + 1$, $|\Gamma_1| \leq |E_1|^2$, and $|\Gamma_2| \leq |E_2|^2$. By Theorem 8, one can construct in linear time a strong NVPA $\mathcal{P} = \langle Q, q_{in}, \Gamma, \Delta, F\rangle$ accepting $(\mathcal{L}(E_1) \setminus \{\varepsilon\})\circlearrowleft_{\Box}(\mathcal{L}(E_2) \setminus \{\varepsilon\}) = \mathcal{L}(E) \setminus \{\varepsilon\}$. Moreover, $|Q| = |Q_1| + |Q_2|$ and $|\Gamma| = |\Gamma_1| + |\Gamma_2| + |Q_1|$. Hence, $|\Gamma| \leq |E_1|^2 + |E_2|^2 + |E_1| + 1 \leq (|E_1| + |E_2| + 1)^2 = |E|^2$ and $|Q| \leq |E_1| + |E_2| + 2 = |E| + 1$, and the result follows.

For concatenation $E = E_1 \cdot E_2$ one starts with the two NVPA $\mathcal{P}_1$ and $\mathcal{P}_2$ accepting $\mathcal{L}(E_1)$ and $\mathcal{L}(E_2)$ and modifies slightly the construction in the proof of Theorem 6 if either E_1 or E_2 accept ε. In the first case (if $\varepsilon \in \mathcal{L}(E_1)$) one adds extra transitions from the fresh new state q_{in} to mimic the inital transitions of $\mathcal{P}_2$. In the second case (if $\varepsilon \in \mathcal{L}(E_2)$), one replicates those transitions of $\mathcal{P}_1$ that reach final states modified to reach final states of $\mathcal{P}_2$ (and hence, final states of the resulting strong NVPA as well). It is routinary to check that the resulting automatong accepts $\mathcal{L}(E_1 \cdot E_2) \setminus \{\varepsilon\}$. □

4.1 Decision Problems for pure VRE

In this subsection, we show the following result.

Theorem 10 *The universality, inclusion, and language equivalence problems for pure VRE are EXPTIME-complete.*

The decision problems of universality, inclusion, and equivalence for NVPA are EXPTIME-complete (see [4]). Consequently, the construction in Theorem 9 directly implies the upper bounds stated in Theorem 10. For the matching lower bounds, it is sufficient to show EXPTIME-hardness for the universality problem because universality can be reduced in linear time to equivalence and language inclusion. The hardness result for universality is proved by a polynomial time reduction from the word problem for polynomial space bounded alternating Turing Machines (TM) with a binary branching degree, which is a well-known EXPTIME-complete problem [11]. Fix such a machine $\mathcal{M} = \langle A, Q, Q_\forall, Q_\exists, q_0, \delta, F\rangle$, where A is the input alphabet, $Q = Q_\forall \cup Q_\exists$ is the finite set of states, which is partitioned into a set $Q_\exists$ of existential states and a set $Q_\forall$ of universal states, $q_0 \in Q$ is the initial state, $F \subseteq Q$ is the set of accepting states, and $\delta : Q \times A \to (Q \times A \times \{\leftarrow, \rightarrow\}) \times (Q \times A \times \{\leftarrow, \rightarrow\})$ is the transition function. In each step, $\mathcal{M}$ overwrites the tape cell being scanned, and the tape head moves one position to the left ($\leftarrow$) or to the right ($\rightarrow$). Configurations of $\mathcal{M}$ are words in $A^* \cdot (Q \times A) \cdot A^*$. A TM configuration $C = \alpha \cdot (q, a) \cdot \alpha'$ denotes that the tape content is $\alpha \cdot a \cdot \alpha'$, the current state is q, and the reading head is at position $|\alpha| + 1$. For the TM configuration C, the *left* (resp., *right*) successor of C is the successor of C obtained by choosing the left (resp., the right) triple in $\delta(q, a)$. Since $\mathcal{M}$ is polynomial space bounded, there is an integer constant $k \geq 1$ such that for each $\alpha \in A^*$, the space needed by $\mathcal{M}$ on the input α is bounded by $|\alpha|^k$. Fix an input α and let $n = |\alpha|$. Without loss of generality we can assume that $k = 1$, $n > 1$, and each (reachable) TM configuration (from the fixed input α) has length exactly n. The initial TM configuration C_α is given by $(q_0, a_1), \ldots, a_n$, where $\alpha = a_1, \ldots, a_n$. Now, we recall the notion of acceptance. A *pseudo computation tree* (over the fixed input α) is a *finite* binary tree T where each node is labeled by a TM configuration (of length n); T is *initialized* if the root is labeled by the initial TM configuration C_α, and T is *accepting* if each leaf is labeled by an accepting TM configuration (i.e., the associated state is in F). A *computation tree* (over α) is a pseudo computation tree satisfying the following additional requirements: each internal node labeled by an universal TM configuration (i.e., the associated state is in $Q_\forall$) has two children, labeled by the left and right successors of C, respectively, while each internal node labeled by an existential TM configuration (i.e., the associated state is in $Q_\exists$) has a unique child, labeled by some successor of C. $\mathcal{M}$ accepts α if there is an initialized accepting computation tree of $\mathcal{M}$ over α.

In the following, we construct in polynomial time in the size of n and the size of $\mathcal{M}$ a pushdown alphabet $\widetilde{\Sigma}$ and a pure VRE E over $\widetilde{\Sigma}$ such that $\mathcal{L}(E) =$

Σ^* *iff* $\mathcal{M}$ does not accept α. Hence, EXPTIME-hardness of the universality problem for pure VRE follows. The pushdown alphabet $\widetilde{\Sigma}$ is given by

$$\widetilde{\Sigma} = \langle \{call\} \times B, \{ret\} \times B, \{\Box\} \rangle \quad \text{where } B = A \cup Q \times A \cup \{\exists_L, \exists_R, \forall_L, \forall_R, end\}$$

Intuitively, the symbols $\exists_L$ and $\exists_R$ (resp., $\forall_L$ and $\forall_R$) are used to tag the left and right successors of an existential (resp., universal) configuration, while the extra symbol *end* is used to tag TM configurations associated with leaf nodes of (pseudo) computation trees. For a word w over B, we denote by $(call, w)$ (resp., (ret, w)) the word over Σ obtained from w by replacing each letter $b \in B$ occurring in w with $(call, b)$ (resp., (ret, b)). For two words w and w', w' is a *subword* of w if w is of the form $w = u \cdot w' \cdot v$ for some words u and v.

Encoding of (pseudo) computation trees. We use a standard encoding of (pseudo) computation trees by minimally well-matched words (over $\widetilde{\Sigma}$) [9,4]. This encoding corresponds to a slight variant of the well-known nested word encoding of finite binary labeled trees, where the given tree is processed in depth-first order as follows: for each node x, we first visit the subtree associated with the left child (if any), and successively, the subtree associated with the right child (if any). Note that each internal node x is visited exactly twice: the first time is when we enter the node x coming from its parent node (in case x is the root, then x is the first node to be examined), and the second time is when we reach x from its right child if it exists, and from its left child otherwise. Moreover, we assume that each leaf is visited twice as well. Thus, the encoding $w_T \in MWM(\widetilde{\Sigma})$ of a pseudo computation tree T is obtained as follows. When a node x with TM configuration C is visited for the first time, we write the subword $(call, d \cdot C \cdot d')$ (consisting of calls), where

- $d = \exists_L$ if x is the root, and d is defined as follows otherwise, where C' is the configuration of the parent node y of x: $d \in \{\exists_L, \exists_R\}$ if C' is existential, and $d = \forall_L$ (resp., $d = \forall_R$) if C' is universal and x is the left (resp., right) child of y;
- $d' = end$ if x is a leaf-node, and d' is empty otherwise.

Finally, when we visit the node x for the last time, then we write the subword $(ret, (d \cdot C \cdot d')^{rev})$ (consisting of returns),[3] which matches the subword associated with the first visit of x. This encoding ensures the following crucial property: for all nodes x and y of T labeled by TM configurations C_x and C_y such that y is the child of x, there is a subword of w_T encoding $C_x \cdot C_y$ or its reverse. More precisely, any subword $(call, w_C)$ of w_T, which encodes the first visit of an internal node x_C with TM configuration C, is followed by a subword $(call, w_L)$ corresponding to the *left* child of x_C in T if C is an universal configuration, and to the unique child of x_C in T otherwise. Moreover, if C is an universal configuration, then the subword $(ret, (w_R)^{rev})$ of w corresponding to the last visit of the *right* child x_R of x_C in T is followed by the subword $(ret, (w_C)^{rev})$ corresponding to the last visit of x_C.

[3] for a word w, w^{rev} denotes the reverse of w.

Let $Codes(\alpha)$ be the set of words $w \in \Sigma^*$ encoding initialized accepting computation trees (over α). We construct in time polynomial in n and the size of $\mathcal{M}$ a pure VRE over $\widetilde{\Sigma}$ which denotes the language $\Sigma^* \setminus Codes(\alpha)$.

Construction of a pure VRE over $\widetilde{\Sigma}$ denoting $\Sigma^* \setminus Codes(\alpha)$. First, we define some languages of finite words over Σ as follows:

- $\mathcal{L}_{in}$: the set of words $w \in \Sigma^*$ of the form $w = (call, \exists_L \cdot C_\alpha) \cdot w'$ for some w' (*initialization requirement*).
- $\mathcal{L}_{acc}$: the set of words $w \in \Sigma^*$ such that for each subword $w' \cdot (call, end)$ of w with $w' \in \Sigma^n$, w' contains some symbol of the form $(call, (q, a))$ with $q \in F$ (*acceptance requirement*).
- $\mathcal{L}_{pseudo}$: the set of words $w \in \Sigma^*$ which encode *pseudo* computation trees. In particular, $\mathcal{L}_{pseudo}$ is the intersection of the following languages:
  - $\mathcal{L}_{pseudo,1}$: the set of words $w \in MWM(\widetilde{\Sigma})$ whose first symbol is $(call, \exists_L)$ and such that w does not contain occurrences of $\Box$ and for each call symbol of the form $(call, b)$ occurring in w, the matching return is (ret, b).
  - $\mathcal{L}_{pseudo,2}$: the set of words $w \in \Sigma^*$ such that each occurrence of a call $(call, \mathcal{Q})$, where $\mathcal{Q} \in \{\exists_L, \exists_R, \forall_L, \forall_R\}$, is followed by a subword of the form $(call, C{\cdot}b)$, where C is a TM configuration and $b \in \{\exists_L, \exists_R, \forall_L, end\}$. Moreover, if $b \neq end$, then $b \in \{\exists_L, \exists_R\}$ if C is existential, and $b = \forall_L$ otherwise.
  - $\mathcal{L}_{pseudo,3}$: the set of words $w \in \Sigma^*$ such that each occurrence of $(call, end)$ is followed by (ret, end), and each occurrence of a return symbol (ret, b) with $b \notin \{\exists_L, \exists_R, \forall_L, \forall_R\}$ is *not* followed by a call symbol.
  - $\mathcal{L}_{pseudo,4}$: the set of words $w \in \Sigma^*$ such that if a return $(ret, \mathcal{Q})$ occurs in a non-terminal position, where $\mathcal{Q} \in \{\exists_L, \exists_R, \forall_R\}$, then this occurrence is followed by a return of the form (ret, b) with $b \in A \cup (Q \times A)$.
  - $\mathcal{L}_{pseudo,5}$: the set of words $w \in \Sigma^*$ such that each occurrence of the return $(ret, \forall_L)$ is followed by the call $(call, \forall_R)$.

  Note that in the encoding of pseudo computation trees T, the language $\mathcal{L}_{pseudo,1}$ corresponds to the requirement that the subword associated with the second visit of a node x of T is the reverse of the subword associated with the first visit of x. The language $\mathcal{L}_{pseudo,2}$ ensures that the first visit of an internal node x of T must be followed by the first visit of node y, where *either* y is the unique child of x, *or* x has two children and y is the left child of x. Moreover, $\mathcal{L}_{pseudo,2}$ ensures that if x is labeled by an existential (resp., universal) configuration, then the subword encoding the first visit of y is tagged by a symbol in $\{\exists_L, \exists_R\}$ (resp., $\forall_L$). The language $\mathcal{L}_{pseudo,4}$ corresponds to the requirement that if a non-root node x is either the right child of the parent node y (if y has two children) or the unique child of y, then the last visit of x must be followed by the last visit of the parent node. Finally, the la nguage $\mathcal{L}_{pseudo,5}$ ensures that the last visit of a node x, which is the left child of a node having two children, is followed by the first visit of the right child of the parent node.
- $\mathcal{L}_{faithful}$: it is the intersection of the following two languages

- $\mathcal{L}_{faithful,1}$: the set of words $w \in \Sigma^*$ such that for each subword $(call, C_1) \cdot (call, \mathcal{Q}) \cdot (call, C_2)$ of w for which $\mathcal{Q} \in \{\exists_L, \forall_L, \exists_R\}$ and C_1 and C_2 are two TM configurations, the following holds:
  - if $Q \in \{\exists_L, \forall_L\}$ then C_2 is the left successor of C_1 in $\mathcal{M}$.
  - if $Q = \exists_R$ then C_2 is the right successor of C_1 in $\mathcal{M}$.
- $\mathcal{L}_{faithful,2}$: the set of words $w \in \Sigma^*$ s.t. for each subword $(ret, (C_1)^{rev}) \cdot (ret, \mathcal{Q}) \cdot (ret, (C_2)^{rev})$ of w for which $\mathcal{Q} \in \{\exists_L, \exists_R, \forall_R\}$ and C_1 and C_2 are two TM configurations, the following holds:
  - if $\mathcal{Q} = \exists_L$ then C_1 is the left successor of C_2 in $\mathcal{M}$.
  - if $\mathcal{Q} = \{\exists_R, \forall_R\}$ then C_1 is the right successor of C_2 in $\mathcal{M}$.

Note that the proposed encoding ensures that for each word $w \in \mathcal{L}_{pseudo}$ (i.e., w encodes a *pseudo* computation tree), $w \in \mathcal{L}_{faithful}$ if and only if w encodes a computation tree.

Thus,

$$\mathit{Codes}(\alpha) = \mathcal{L}_{in} \cap \mathcal{L}_{acc} \cap \bigcap_{i=1}^{i=5} \mathcal{L}_{pseudo,i} \cap \mathcal{L}_{faithful,1} \cap \mathcal{L}_{faithful,2}$$

and then

$$\Sigma^* \setminus \mathit{Codes}(\alpha) = \Sigma^* \setminus \mathcal{L}_{in} \cup \Sigma^* \setminus \mathcal{L}_{acc} \cup \bigcup_{i=1}^{i=5} \Sigma^* \setminus \mathcal{L}_{pseudo,i} \cup \Sigma^* \setminus \mathcal{L}_{faithful,1} \cup \Sigma^* \setminus \mathcal{L}_{faithful,2}$$

Hence, it suffices to show that for all $\mathcal{L} \in \{\mathcal{L}_{in}, \mathcal{L}_{acc}, \mathcal{L}_{pseudo,1}, \ldots, \mathcal{L}_{pseudo,5}, \mathcal{L}_{faithful,1}, \mathcal{L}_{faithful,2}\}$, it is possible to build in polynomial time a pure VRE over $\widetilde{\Sigma}$ denoting the language $\Sigma^* \setminus \mathcal{L}$. The constructions for the languages $\Sigma^* \setminus \mathcal{L}$ with $\mathcal{L} \in \{\mathcal{L}_{in}, \mathcal{L}_{acc}, \mathcal{L}_{pseudo,3}, \mathcal{L}_{pseudo,4}, \mathcal{L}_{pseudo,5}\}$ are straightforward. Moreover, the constructions for $\Sigma^* \setminus \mathcal{L}_{faithful,1}$ and $\Sigma^* \setminus \mathcal{L}_{faithful,2}$ are very similar. Thus, here, we focus on the languages $\Sigma^* \setminus \mathcal{L}_{pseudo,1}$, $\Sigma^* \setminus \mathcal{L}_{pseudo,2}$, and $\Sigma^* \setminus \mathcal{L}_{faithful,1}$. We use some simplifying notation: for $A, B \subseteq \Sigma$, we write A to denote the VRE $\bigcup_{\sigma \in A} \sigma$, and $A \setminus B$ to denote the VRE $\bigcup_{\sigma \in A \setminus B} \sigma$.

Pure VRE for $\Sigma^* \setminus \mathcal{L}_{pseudo,1}$: first, we construct a pure VRE E denoting the language $\Sigma^* \setminus \mathit{MWM}(\widetilde{\Sigma})$ as follows

$$E \stackrel{\text{def}}{=} (\Sigma \setminus \Sigma_{call}) \cdot \Sigma^* \quad \cup \quad \Sigma_{call} \cdot [(\Box \curvearrowleft_{\Box} \Sigma^*)^* \cdot (\Sigma_{call} \cup \Box)^*]^*$$
$$\bigcup$$
$$\bigcup_{c \in \Sigma_{call},\, r \in \Sigma_{ret}} [\Box \curvearrowleft_{\Box} (c \cdot \Sigma^* \cdot r)] \cdot \Sigma^+$$

Then, the pure VRE denoting the language $\Sigma^* \setminus \mathcal{L}_{pseudo,1}$ is given by

$$(\Sigma \setminus \{(call, \exists_L)\}) \cdot \Sigma^* \ \cup \ \Sigma^* \cdot \Box \cdot \Sigma^* \ \cup \ E$$

$$\bigcup$$

$$\Sigma^* \ \cdot \bigcup_{(call,b)\in\Sigma_{call},\,(ret,b')\in\Sigma_{ret},\,b'\neq b} [\Box \curvearrowleft_{\Box} ((call, b) \cdot \Sigma^* \cdot (ret, b'))] \ \cdot \ \Sigma^*$$

Pure VRE for $\Sigma^* \setminus \mathcal{L}_{pseudo,2}$:

$$\Sigma^* \cdot (\{call\} \times \{\exists_L, \exists_R, \forall_L, \forall_R\}) \cdot \Big[$$

$$\Sigma^n \cdot (\Sigma \setminus \{call\} \times \{\exists_L, \exists_R, \forall_L, end\})$$

$$\bigcup$$

$$\bigcup_{i=1}^{i=n} [\Sigma^{i-1} \cdot (\Sigma \setminus \{call\} \times (A \cup Q \times A)) \cdot \Sigma^{n-i+1}]$$

$$\bigcup$$

$$\bigcup_{i=1}^{i=n} [\Sigma^{i-1} \cdot \{call\} \times (Q_\exists \times A) \cdot \Sigma^{n-i} \cdot (\Sigma \setminus \{call\} \times \{\exists_L, \exists_R, end\})]$$

$$\bigcup$$

$$\bigcup_{i=1}^{i=n} [\Sigma^{i-1} \cdot \{call\} \times (Q_\forall \times A) \cdot \Sigma^{n-i} \cdot (\Sigma \setminus \{call\} \times \{\forall_L, end\})]$$

$$\bigcup$$

$$\big(\Sigma \setminus (\{call\} \times (Q \times A))\big)^n$$

$$\bigcup$$

$$\bigcup_{i=1}^{i=n} \bigcup_{j=i}^{j=n} \Sigma^{i-1} \cdot (\{call\} \times (Q \times A)) \cdot \Sigma^{j-i+1} \cdot (\{call\} \times (Q \times A)) \cdot \Sigma^{n-j+1}\Big] \cdot \Sigma^*$$

The last two lines correspond to C not being a valid configuration (no state or more than one state, resp.)

Pure VRE for $\Sigma^* \setminus \mathcal{L}_{faithful,1}$: let $C = u_1 \dots u_n$ be a TM configuration. For each $1 \leq i \leq n$, the value u'_i of the i^{th} cell of the left (resp., right) successor of C is completely determined by the values u_{i-1}, u_i and u_{i+1} (taking u_{i+1} for $i = n$ and u_{i-1} for $i = 1$ to be some special symbol). We denote by $N_L(u_{i-1}, u_i, u_{i+1})$ (resp., $N_R(u_{i-1}, u_i, u_{i+1})$) our expectation for u'_i (these functions can be trivially obtained from the transition function δ of $\mathcal{M}$). The pure VRE for $\Sigma^* \setminus \mathcal{L}_{faithful,1}$ is defined as follows:

$$\bigcup_{i=1}^{i=n} \bigcup_{u_1,u_2,u_3\in\Lambda} \bigcup_{dir\in\{L,R\}} \Sigma^* \cdot E_{i,u_1,u_2,u_3} \ \cdot \ \{call\} \times \{\exists_{dir}, \forall_{dir}\} \cdot [\{call\} \times \Lambda]^{i-1} \cdot$$

$$\cdot (\Sigma \setminus \{(call, N_{dir}(u_1, u_2, u_3))\}) \ \cdot \ [\{call\} \times \Lambda]^{n-i} \ \cdot \ \Sigma^*$$

where $\Lambda = A \cup Q \times A$ and E_{i,u_1,u_2,u_3} is defined as follows. We assume that $1 < i < n$ (the other cases being similar)

$$E_{i,u_1,u_2,u_3} \stackrel{\text{def}}{=} [\{call\} \times \Lambda]^{i-2} \cdot (call, u_1) \cdot (call, u_2) \cdot (call, u_3) \cdot [\{call\} \times \Lambda]^{n-i-1}$$

5 ω-Visibly Rational Expressions

In this section, we introduce the class of ω-Visibly Rational Expressions (ω-VRE) and provide a Büchi-like theorem for ω-VPL in terms of ω-VRE. Fix a pushdown alphabet $\widetilde{\Sigma}$. For a language $\mathcal{L}$ of finite words over Σ, we denote by $\mathcal{L}^\omega$ the standard ω-Kleene closure of $\mathcal{L}$.

Definition 5 The syntax of ω-VRE I over $\widetilde{\Sigma}$ is inductively defined as follows:

$$I := (E)^\omega \;\mid\; (I \cup I) \;\mid\; (E \cdot I)$$

where E is a VRE over $\widetilde{\Sigma}$. Note that ω-VRE are defined similarly to ω-regular expressions. An ω-VRE I is *pure* if every VRE subexpression is pure. An ω-VRE I denotes a language of infinite words over Σ, written $\mathcal{L}(I)$, defined as follows: $\mathcal{L}(E^\omega) = [\mathcal{L}(E)]^\omega$, $\mathcal{L}(I \cup I') = \mathcal{L}(I) \cup \mathcal{L}(I')$, and $\mathcal{L}(E \cdot I) = \mathcal{L}(E) \cdot \mathcal{L}(I)$.

We show that ω-VRE capture the class of ω-VPL. For this, we need the following preliminary result establishing that ω-VPL can be expressed in terms of VPL in the same way as ω-regular languages can be expressed in terms of regular languages.

Theorem 11 *Let $\mathcal{L}$ be an ω-VPL with respect to $\widetilde{\Sigma}$. Then, there are $n \geq 1$ and VPL $\mathcal{L}_1, \mathcal{L}'_1, \ldots, \mathcal{L}_n, \mathcal{L}'_n$ with respect to $\widetilde{\Sigma}$ such that $\mathcal{L} = \bigcup_{i=1}^{i=n} \mathcal{L}_i \cdot (\mathcal{L}'_i)^\omega$. Moreover, the characterization is constructive.*

Proof Fix a Büchi NVPA $\mathcal{P} = \langle Q, q_{in}, \Gamma, \Delta, F \rangle$ accepting $\mathcal{L}$. Let $MR(\widetilde{\Sigma})$ be the set of all *finite* words where every return position has a matching call position, and $MC(\widetilde{\Sigma})$ be the set of all *finite* words where every call position has a matching return position. Note that $MR(\widetilde{\Sigma})$ and $MC(\widetilde{\Sigma})$ are effectively NVPA representable. Now, for all states $q \in Q$, we define the following languages of finite words over Σ as follows:

- $\mathcal{L}_{in,q}$ is the set of finite words w such that there is a run of $\mathcal{P}$ over w of the form $(q_{in}, \bot) \xrightarrow{w} (q, \beta)$ for some stack content β;
- $\mathcal{L}_q$ is the set of finite words $w \neq \varepsilon$ such that there is a run of $\mathcal{P}$ over w of the form $(q, \bot) \xrightarrow{w} (q, \beta)$ which visits some state in F;
- $MC(\mathcal{L}_{in,q}) \stackrel{\text{def}}{=} \mathcal{L}_{in,q} \cap MC(\widetilde{\Sigma})$ and $MC(\mathcal{L}_q) \stackrel{\text{def}}{=} \mathcal{L}_q \cap MC(\widetilde{\Sigma})$;
- $MR(\mathcal{L}_q) \stackrel{\text{def}}{=} \mathcal{L}_q \cap MR(\widetilde{\Sigma})$.

It is easy to show that $\mathcal{L}_q$ is effectively NVPA representable. Thus, since $\mathcal{L}_{in,q}$, $MR(\widetilde{\Sigma})$, and $MC(\widetilde{\Sigma})$ are effectively NVPA representable, and NVPA are effectively closed under conjunction [4], we have that all the languages defined above are effectively NVPA representable. Therefore, the theorem directly follows from the following claim.

Claim: $\mathcal{L} = \mathcal{L}_\cup$ where $\mathcal{L}_\cup \stackrel{\text{def}}{=} \bigcup_{q \in Q} \mathcal{L}_{in,q} \cdot [MR(\mathcal{L}_q)]^\omega \cup \bigcup_{q \in Q} MC(\mathcal{L}_{in,q}) \cdot [MC(\mathcal{L}_q)]^\omega$.

Proof of the claim:

Inclusion $\mathcal{L} \subseteq \mathcal{L}_\cup$: let $w \in \mathcal{L}$. We show that $w \in \mathcal{L}_\cup$ by distinguishing two cases:

– There are *infinitely* many *unmatched* return positions in w. Hence, every call position has a matching return in w and $w \in (MC(\widetilde{\Sigma}))^\omega$. Since $\mathcal{P}$ accepts w and $MC(\widetilde{\Sigma})$ is closed under concatenation, it follows that there are $q \in Q$ and an initialized accepting run of $\mathcal{P}$ over w of the form $(q_{in}, \bot) \xrightarrow{w_0} (q, \beta_1) \xrightarrow{w_1} (q, \beta_2) \xrightarrow{w_2} (q, \beta_3) \dots$ such that: $w_0 \in MC(\widetilde{\Sigma})$, and for all $i \geq 1$, $w_i \in MC(\widetilde{\Sigma}) \setminus \{\varepsilon\}$ and the subrun $(q, \beta_i) \xrightarrow{w_i} (q, \beta_{i+1})$ of $\mathcal{P}$ over w_i visits some state in F. Since $w_i \in MC(\widetilde{\Sigma})$ for all $i \geq 0$, we deduce that $\beta_i = \bot$. It follows that $w_0 \in MC(\mathcal{L}_{in,q})$ and $w_i \in MC(\mathcal{L}_q)$ for all $i \geq 1$. Thus, since $w = w_0 \cdot w_1 \cdot w_2 \cdot \dots$, we obtain that $w \in MC(\mathcal{L}_{in,q}) \cdot [MC(\mathcal{L}_q)]^\omega \subseteq \mathcal{L}_\cup$, and the result holds in this case.
– There are *finitely* many unmatched return positions in w. Hence, $w \in \Sigma^* \cdot (MR(\widetilde{\Sigma}))^\omega$. Since $\mathcal{P}$ accepts w and $MR(\widetilde{\Sigma})$ is closed under concatenation, it follows that there are $q \in Q$ and an initialized accepting run of $\mathcal{P}$ over w of the form $(q_{in}, \bot) \xrightarrow{w_0} (q, \beta_1) \xrightarrow{w_1} (q, \beta_2) \xrightarrow{w_2} (q, \beta_3) \dots$ such that for all $i \geq 1$, $w_i \in MR(\widetilde{\Sigma}) \setminus \{\varepsilon\}$ and the subrun $(q, \beta_i) \xrightarrow{w_i} (q, \beta_{i+1})$ of $\mathcal{P}$ over w_i visits some state in F. Since $w_i \in MR(\widetilde{\Sigma})$ for $i \geq 1$, the portion of the stack corresponding to β_i is never read in the subrun $(q, \beta_i) \xrightarrow{w_i} (q, \beta_{i+1})$. Hence, there is also a run of $\mathcal{P}$ over w_i from $(q, \bot)$ to a configuration of the form (q, β) which visits some state in F. Hence, $w_i \in MR(\mathcal{L}_q)$ for all $i \geq 1$. Since $w = w_0 \cdot w_1 \cdot w_2 \cdot \dots$, we obtain that $w \in \mathcal{L}_{in,q} \cdot [MR(\mathcal{L}_q)]^\omega \subseteq \mathcal{L}_\cup$, and the result holds in this case as well.

Converse inclusion $\mathcal{L}_\cup \subseteq \mathcal{L}$: let $w \in \mathcal{L}_\cup$. We show that $w \in \mathcal{L}$. By definition of $\mathcal{L}_\cup$, there are two cases:

– $w \in MC(\mathcal{L}_{in,q}) \cdot [MC(\mathcal{L}_q)]^\omega$ for some $q \in Q$. Hence, w can be written in the form $w_0 \cdot w_1 \cdot w_2 \dots$ such that $w_0 \in MC(\widetilde{\Sigma})$, there is a run of $\mathcal{P}$ over w_0 of the form $(q_{in}, \bot) \xrightarrow{w_0} (q, \beta_0)$, and for all $i \geq 1$, $w_i \in MC(\widetilde{\Sigma}) \setminus \{\varepsilon\}$ and there is a run of $\mathcal{P}$ over w_i of the form $(q, \bot) \xrightarrow{w_i} (q, \beta_i)$ which visits some state in F. Since $w_i \in MC(\widetilde{\Sigma})$ for all $i \geq 0$, it follows that $\beta_i = \bot$. Hence, there is an initialized accepting run of $\mathcal{P}$ over w, i.e. $w \in \mathcal{L}$.
– $w \in \mathcal{L}_{in,q} \cdot [MR(\mathcal{L}_q)]^\omega$ for some $q \in Q$. Hence, w can be written in the form $w_0 \cdot w_1 \cdot w_2 \dots$ such that there is a run of $\mathcal{P}$ over w_0 of the form $(q_{in}, \bot) \xrightarrow{w_0} (q, \beta_0 \cdot \bot)$ and for all $i \geq 1$, $w_i \in MR(\widetilde{\Sigma}) \setminus \{\varepsilon\}$ and there is a run of $\mathcal{P}$ over w_i of the form $(q, \bot) \xrightarrow{w_i} (q, \beta_i \cdot \bot)$ which visits some state in F. Since $w_i \in MR(\widetilde{\Sigma})$ for all $i \geq 1$, it follows that for all $\beta \in \Gamma^*$, there is also a run of $\mathcal{P}$ over w_i of the form $(q, \beta \cdot \bot) \xrightarrow{w_i} (q, \beta_i \cdot \beta \cdot \bot)$ which visits some state in F. Thus, we deduce that there is an initialized accepting run of $\mathcal{P}$ over w of the form $(q_{in}, \bot) \xrightarrow{w_0} (q, \beta_0 \cdot \bot) \xrightarrow{w_1} (q, \beta_1 \cdot \beta_0 \cdot \bot) \xrightarrow{w_2} (q, \beta_2 \cdot \beta_1 \cdot \beta_0 \cdot \bot) \dots$. Hence, $w \in \mathcal{L}$.

This concludes the proof of Theorem 11. □

Since ω-VPL are effectively closed under ω-Kleene closure and under (left) concatenation with VPL (and the constructions can be done in linear time) [4], by Corollary 1, it follows that ω-VRE can be converted into equivalent ω-NVPA

in single exponential time. Moreover, by using *strong* NVPA and constructions very similar to those used in the proof of Theorem 6, one can show that *pure* ω-VRE can be converted into equivalent Büchi ω-NVPA in quadratic time. Thus, by Theorem 11 we obtain the following result.

Theorem 12 (*Pure*) *ω-VRE capture the class of ω-VPL. Moreover, pure ω-VRE can be converted in quadratic time into equivalent Büchi ω-NVPA.*

6 Conclusion

In this paper, we have provided a Kleene theorem for VPL and a Büchi theorem for ω-VPL. From a theoretical point of view, there are some interesting open questions. For example, while is well-known that NFA are exponentially more succinct than regular expressions, it remains an open problem to capture the precise succinctness gap between VRE and NVPA. From a practical viewpoint, it remains to be empirically evaluated whether VRE are useful as a specification language for nested word search and for XML schemas. Another line of future work is the combination of VRE with temporal logics for nested words (like CaRet [2]), as done for word regular languages [13,18].

References

1. R. Alur. Marrying words and trees. In *Proc. of 26th ACM Symposium on Principles of Database Systems (PODS'07)*, pages 233–242. ACM, 2007.
2. R. Alur, K. Etessami, and P. Madhusudan. A Temporal Logic of Nested Calls and Returns. In *Proc. of 10th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS'04)*, volume 2988 of LNCS, pages 467–481. Springer, 2004.
3. R. Alur, V. Kumar, P. Madhusudan, and M. Viswanathan. Congruences for Visibly Pushdown Languages. In *Proc. of 32nd International Colloquium on Automata, Languages and Programming (ICALP'05)*, volume 3580 of LNCS, pages 1102–1114. Springer, 2005.
4. R. Alur and P. Madhusudan. Visibly Pushdown Languages. In *Proc. of 36th Annual ACM Symposium on Theory of Computing (STOC'04)*, pages 202–211. ACM, 2004.
5. R. Alur and P. Madhusudan. Adding nesting structure to words. *Journal of the ACM*, 56(3), 2009.
6. M. Arenas, P. Barceló, and L. Libkin. Regular Languages of Nested Words: Fixed Points, Automata, and Synchronization. In *Proc. of 34th International Colloquium on Automata, Languages and Programming (ICALP'07)*, volume 4596 of LNCS, pages 888–900. Springer, 2007.
7. T. Ball and S.K. Rajamani. Bebop: a symbolic model checker for boolean programs. In *Proc. of 7th International Workshop on Model Checking and Software Verification (SPIN'00)*, volume 1885 of LNCS, pages 113–130. Springer, 2000.
8. C. Bolduc and B. Ktari. Visibly Pushdown Kleene Algebra and Its Use in Interprocedural Analysis of (Mutually) Recursive Programs. In *Proc. of 11th International Conference on Relational Methods in Computer Science (RelMiCS'09)*, volume 5827 of LNCS, pages 44-58. Springer, 2009.
9. A. Bouajjani, J. Esparza, and O. Maler. Reachability analysis of pushdown automata: application to model-checking. In *Proc. of 8th International Conference on Concurrency Theory (CONCUR'97)*, volume 1243 of LNCS, pages 135–150. Springer, 1997.

10. L. Bozzelli. Alternating automata and a temporal fixpoint calculus for visibly pushdown languages. In *Proc. of 18th International Conference on Concurrency Theory (CONCUR'07)*, volume 4703 of LNCS, pages 476–491. Springer, 2007.
11. A.K. Chandra, D. Kozen, and L.J. Stockmeyer. Alternation. *Journal of the ACM*, 28(1):114–133, 1981.
12. A. Cyriac. Temporal Logics for Concurrent Recursive Programs. Rapport de Master, Master Parisien de Recherche en Informatique, Paris, France, September 2010.
13. D. Fisman, C. Eisner, and J. Havlicek. Formal syntax and Semantics of PSL: Appendix B of Accellera Property Language Reference Manual, Version 1.1. March, 2004.
14. J.E. Hopcroft and J.D. Ullman. *Introduction to automata theory, languages and computation*. Addison-Wesley, 1979.
15. S.C. Kleene. Representation of Events in Nerve Nets and Finite Automata. In *Automata Studies*, volume 34, pages 3–41. Princeton University Press, 1956.
16. C. Koch and S. Scherzinger. Attribute Grammars for Scalable Query Processing on XML Streams. In *Proc. of the 9th International Workshop on Database Programming Languages (DBPL'03)*, volume 2921 of LNCS, pages 233–256. Springer, 2004.
17. V. Kumar, P. Madhusudan, and M. Viswanathan. Visibly pushdown automata for streaming XML. In *Proc. of 16th International Conference on World Wide Web (WWW'07)*, pages 1053–1062. ACM, 2007.
18. M. Leucker and C. Sánchez. Regular Linear Temporal Logic. In *Proc. of 4th International Colloquium on Theoretical Aspects of Computing (ICTAC'07)*, volume 4711 of LNCS, pages 291–305. Springer, 2007.
19. C. Löding, P. Madhusudan, and O. Serre. Visibly Pushdown Games. In *Proc. of 24th International Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS'04)*, volume 3328 of LNCS, pages 408–420. Springer, 2004.
20. P. Madhusudan and M. Viswanathan. Query Automata for XML Nested Words. In *Proc. of 34th International Symposium on Mathematical Foundations of Computer Science (MFCS'09)*, volume 5734 of LNCS, pages 561–573. Springer, 2009.
21. C. Pitcher. Visibly Pushdown Expression Effects for XML Stream Processing. In *Proc. of ACM Workshop on Programming Language Technologies for XML (PLAN-X'05)*, pages 1–14. ACM, 2005.
22. A. Thomo, S. Venkatesh, and Y.Y. Ye. Visibly Pushdown Transducers for Approximate Validation of Streaming XML. In *Proc. of 5th International Symposium on Foundations of Information and Knowledge Systems (FoIKS'08)*, volume 4932 of LNCS, pages 219-238. Springer, 2008.
23. K. Thompson. Regular expression search algorithm. *Communications of the ACM*, 11(6):419–422, 1968.