LTL with the Freeze Quantifier and Register Automata

Stéphane Demri *
LSV, CNRS & ENS Cachan & INRIA Futurs, France

Ranko Lazić †
Department of Computer Science, University of Warwick, UK

Abstract

Temporal logics, first-order logics, and automata over data words have recently attracted considerable attention. A data word is a word over a finite alphabet, together with a datum (an element of an infinite domain) at each position. Examples include timed words and XML documents. To refer to the data, temporal logics are extended with the freeze quantifier, first-order logics with predicates over the data domain, and automata with registers or pebbles.

We investigate relative expressiveness and complexity of standard decision problems for LTL with the freeze quantifier (LTL$^{\downarrow}$), 2-variable first-order logic (FO^2) over data words, and register automata. The only predicate available on data is equality. Previously undiscovered connections among those formalisms, and to counter automata with incrementing errors, enable us to answer several questions left open in recent literature.

We show that the future-time fragment of LTL$^{\downarrow}$ which corresponds to FO^2 over finite data words can be extended considerably while preserving decidability, but at the expense of non-primitive recursive complexity, and that most of further extensions are undecidable. We also prove that surprisingly, over infinite data words, LTL$^{\downarrow}$ without the 'until' operator, as well as nonemptiness of one-way universal register automata, are undecidable even when there is only 1 register.

1. Introduction

Being able to store a value in some register/variable and to test it later in a different context, is a common feature of many recently studied logical formalisms. The following are the most prominent examples:

Timed logics. The freeze quantifier in timed logics was introduced in the logic TPTL (e.g. [2]), where the formula $x \cdot \phi(x)$ binds the variable x to the time t of the current state. Depending on the semantics, x is interpreted as a real number or a natural number and the formula is semantically equivalent to $\phi(t)$.

Hybrid logics. In [13], the formula $\downarrow_x \phi(x)$ holds whenever $\phi(x)$ holds in the variant Kripke structure where the propositional variable x is interpreted as a singleton containing exactly the current state.

Modal logics. Predicate λ-abstraction is presented in [11] to solve the problem of interpreting constants in first-order modal logics: $\langle \lambda x \cdot \mathtt{F}\ P(x) \rangle(c)$ states that the current value of the constant c satisfies the predicate P eventually in the future.

Logics with forgettable past. In [15], $\mathtt{Now}\ \phi$ holds whenever ϕ holds in a linear structure in which the origin is updated to the current position (ϕ may contain past-time operators). Equivalently, the register containing the position of the origin of time is assigned the current position.

Interestingly, the same general mechanism is central to the notion of register automata [14, 26, 6, 21], which recognise words over infinite alphabets. Indeed, a letter can be stored in a register and tested later against the current letter. Similarly, in Alur-Dill timed automata (e.g. [1]), resetting a clock c to 0 is equivalent to storing the current time as the time when c was last reset.

The ability to store and test is powerful, since many problems are undecidable in its presence [14, 1, 6, 8, 17]. However, searching for decidable fragments or subproblems, and determining their complexity, is well-motivated by the fact that logical and automata formalisms with such features are helpful for querying semi-structured data [21, 7, 3], verifying timed systems [2, 1], model checking constrained automata [8], and verifying dynamic systems with resources [17], quoting a few examples.

In this paper, we consider logics and automata over finite and infinite data words. In a data word, at each index, there is a letter from a finite alphabet Σ, and an element of an infinite domain D. As in [14, 26, 21, 7, 4, 8, 17], elements of D can only be compared for equality, so it is equivalent and simpler to define a data word as a word over Σ

*Supported by the ACI "Sécurité et Informatique" CORTOS.

†Supported by an invited professorship from ENS Cachan, and by grants from the EPSRC (GR/S52759/01) and the Intel Corporation. Also affiliated to the Mathematical Institute, Serbian Academy of Sciences and Arts, Belgrade.

equipped with an equivalence relation on its indices: $i \sim j$ iff the elements of D at indices i and j are equal. In common with [7, 4], we take this latter approach. To be able to consider languages of words over Σ obtained by projecting data words, we do not eliminate the finite alphabets from the definition of data words, although such eliminations are possible by encodings as in [14, 26, 21, 8].

We study linear temporal logic extended by the freeze quantifier (LTL$^{\downarrow}$). The formula $\downarrow_r \phi$ holds at an index i of a data word iff ϕ holds with i stored in the register r. Within the scope of the freeze quantifier $\downarrow_r$, the atomic formula $\uparrow_r \sim$ is true at an index j iff $i \sim j$, i.e. the data value at the index in r is equal to the data value at the current index.

LTL$^{\downarrow}$ is the core of Constraint LTL with the freeze quantifier [8], and of the linear temporal logics with predicate λ-abstraction [17]. Moreover, Repeating Hybrid LTL considered in [12] is exactly the fragment of LTL$^{\downarrow}$ with the temporal operators $\mathtt{X}$, $\mathtt{X}^{-1}$, $\mathtt{F}$ and $\mathtt{F}^{-1}$.

We show that the first-order logic with 2 variables FO$^2(\sim, <, +1)$ studied in [7, 4] is equivalent to a natural fragment of LTL$^{\downarrow}$ with only 1 register. That extends the equivalence between FO$^2(<, +1)$ and unary LTL in the setting without data values [10]. In [4], satisfiability for FO$^2(\sim, <, +1)$ is proved decidable and as hard as reachability for Petri nets, over finite and over infinite data words.

The automata formalism we consider is register automata (RA) over data words. As in LTL$^{\downarrow}$, an RA can store the current data word index in a register r, and subsequently test whether the index in r is in the same class of $\sim$ as the current index. Over finite data words, essentially the same automata were studied in [14, 26, 21], and one-way nondeterministic RA are a subclass of data automata [6, 5]. We consider two-way alternating RA, as well as subclasses obtained by restricting directionality and/or control. In contrast to finite automata on words over finite alphabets, a number of separation results for such subclasses were obtained in [14, 21]. For infinite data words, we focus on weak parity RA, a subclass of both Büchi and co-Büchi RA.

The second expressiveness result in the paper is a translation in logarithmic space from sentences of LTL$^{\downarrow}$ to equivalent RA. For the future fragment, one-way RA are sufficient.

The central part of the paper consists of a systematic investigation of decidability and complexity of standard decision problems for fragments of LTL$^{\downarrow}$ and classes of RA. Most of the results are based on surprising translations to and from counter automata (CA), where we consider both standard (Minsky) CA whose computations are exact, and faulty (Incrementing) CA whose computations may contain errors which increase one or more counters.

We show that, over finite and infinite data words, with only 1 register, satisfiability for the future fragment of LTL$^{\downarrow}$, as well as nonemptiness of one-way alternating RA, are reducible in polynomial space to nonemptiness of Incrementing CA. In the finitary case, that gives us decidability of the former two problems. The only decidable fragment of LTL$^{\downarrow}$ previously known is the flat fragment in [8], in which use of the freeze quantifier is heavily restricted. Interestingly, the translation from one-way alternating RA to Incrementing CA consists of broadly similar steps as the translation from one-way alternating timed automata to faulty channel machines in [24].

For the next main result, we first adapt the recent results in [27, 24] on faulty channel machines to obtain that finitary [resp. infinitary] nonemptiness of Incrementing CA is not primitive recursive [resp. Π^0_1-hard]. Then, through encoding runs of Incrementing CA as data words, we obtain that, even with only 1 register, finitary [resp. infinitary] satisfiability for the future fragment of LTL$^{\downarrow}$ without the $\mathtt{U}$ operator, as well as nonemptiness of one-way universal RA (equivalently, nonuniversality of one-way nondeterministic RA), are not primitive recursive [resp. Π^0_1-hard].

By translating from Minsky CA instead of Incrementing CA, we show that several other satisfiability problems for fragments of LTL$^{\downarrow}$ and nonemptiness problems for classes of RA are Σ^0_1-hard over finite data words, and Σ^1_1-hard over infinite data words.

Taken together, the decision problem results in this paper provide a tighter demarkation of the decidability border than was known in the literature, in terms of finite versus infinite data words, the number of registers, sets of temporal operators, and automata directionality and control. We answer several questions posed in [14, 21, 12, 8, 17]. In particular, the undecidability result for nonemptiness of one-way universal RA with only 1 register shows that it is impossible to extend the developments in [14] to infinite words, which was a challenge posed in that paper.

Along the way, we obtain several other results, including a characterisation of projections onto the finite alphabet of languages of sentences from the future fragment of LTL$^{\downarrow}$ with 1 register. Surprisingly, the characterisation is the same with or without the $\mathtt{U}$ operator.

More detailed comparisons with related work can be found throughout the paper.

Sections 2 and 3 contain the definitions, and warm-up results on closure properties of register automata and on nonemptiness of counter automata. The results on relative expressiveness are in Section 4. The central part of the paper is in Sections 5 and 6, which are mainly on decidability and complexity of decision problems.

2. Preliminaries

2.1. LTL over data words

LTL$^{\downarrow}(\sim; \mathcal{O})$ will denote the linear temporal logic with the freeze quantifier, the predicate $\sim$, and temporal oper-

ators in the set $\mathcal{O}$. Each formula is over a finite alphabet Σ. Atomic propositions a are elements of Σ, r ranges over $\mathbb{N} \setminus \{0\}$, and $\mathtt{O}$ ranges over $\mathcal{O}$.

$$\phi \ ::= \ \top \mid a \mid \uparrow_r \sim \mid \neg\phi \mid \phi \wedge \phi \mid \mathtt{O}(\phi, \ldots, \phi) \mid \downarrow_r \phi$$

An occurence of $\uparrow_r \sim$ within the scope of some freeze quantifier $\downarrow_r$ is bound by it; otherwise, it is free. A sentence is a formula with no free occurence of any $\uparrow_r \sim$.

We consider the set $\{\mathtt{X}, \mathtt{X}^{-1}, \mathtt{F}, \mathtt{F}^{-1}, \mathtt{U}, \mathtt{U}^{-1}\}$ of temporal operators, and its subsets. As $\mathtt{F}\phi$ is equivalent to $\top \mathtt{U} \phi$, $\mathtt{F}$ can be omitted from any set which contains $\mathtt{U}$, and the same is true for $\mathtt{F}^{-1}$ and $\mathtt{U}^{-1}$. As usual, we regard $\mathtt{G}$ and $\mathtt{G}^{-1}$ as abbreviations for $\neg\mathtt{F}\neg$ and $\neg\mathtt{F}^{-1}\neg$.

Let $\mathrm{LTL}^{\downarrow}_n(\sim; \mathcal{O})$ be the fragment of $\mathrm{LTL}^{\downarrow}(\sim; \mathcal{O})$ with n registers, i.e. where $r \in \{1, \ldots, n\}$.

Models of $\mathrm{LTL}^{\downarrow}(\sim; \mathcal{O})$ are *data words*. A data word σ over a finite alphabet Σ is a nonempty word in $\Sigma^{<\omega}$ or Σ^{ω}, together with an interpretation of $\sim$ as an equivalence relation on word indices. We write $|\sigma|$ for the length of the word, $\sigma(i)$ for its letters where $0 \leq i < |\sigma|$, and $\sim^{\sigma}$ for the interpretation of $\sim$. Let $\Sigma^{<\omega}(\sim)$ and $\Sigma^{\omega}(\sim)$ denote the sets of all such finite and infinite (respectively) data words. For a data word σ, let $\mathrm{str}(\sigma)$ be the underlying word in $\Sigma^{\leq\omega}$.

A *register valuation* v for a data word σ is a finite partial map from $\mathbb{N} \setminus \{0\}$ to the indices of σ. An undefined register value in an atomic formula will make the latter false. Undefined register values will be used for initial automata states. The satisfaction relation $\models$ is defined as follows. Temporal operators are interpreted as in LTL, so we show only one. We also omit the Boolean cases.

$$\begin{aligned}
\sigma, i \ &\models_v \ a \quad &&\overset{\text{def}}{\Leftrightarrow} \quad \sigma(i) = a \\
\sigma, i \ &\models_v \ \uparrow_r \sim \quad &&\overset{\text{def}}{\Leftrightarrow} \quad r \in \mathrm{dom}(v) \text{ and } v(r) \sim^{\sigma} i \\
\sigma, i \ &\models_v \ \mathtt{X}\phi \quad &&\overset{\text{def}}{\Leftrightarrow} \quad i + 1 < |\sigma| \text{ and } \sigma, i+1 \ \models_v \ \phi \\
\sigma, i \ &\models_v \ \downarrow_r \phi \quad &&\overset{\text{def}}{\Leftrightarrow} \quad \sigma, i \ \models_{v[r \mapsto i]} \ \phi
\end{aligned}$$

2.2. First-order logic over data words

As defined in [4], $\mathrm{FO}(\sim, <, +1, \ldots, +m)$ denotes first-order logic over data words, in which variables range over word indices. We use variable names $x_0, x_1, \ldots$ The predicates $x_i < x_j$ and $x_i = x_j + k$ are interpreted as expected. Each formula has an alphabet Σ, and it may contain unary predicates $P_a(x_i)$ which are satisfied by a data word iff the letter at index x_i is a. When we write $\phi(x_{i_1}, \ldots, x_{i_N})$, it means that at most $x_{i_1}, \ldots, x_{i_N}$ occur free in ϕ.

$\mathrm{FO}^n(\sim, <, +1, \ldots, +m)$ has variables $x_0, \ldots, x_{n-1}$.

2.3. Register automata

Suppose Q is a finite set of locations, and $n \in \mathbb{N}$. The set $\Phi(Q, n)$ of all *transition formulae* with respect to Q and n is defined as follows, where $r \in \{1, \ldots, n\}$ and $q \in Q$:

$$\begin{aligned}
\varphi \quad ::= \quad & \top \mid \bot \mid \uparrow_r \sim \mid \uparrow_r \not\sim \mid \mathtt{beg} \mid \mathtt{nbeg} \mid \mathtt{end} \mid \mathtt{nend} \mid \\
& \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \downarrow_r \varphi \mid \langle q, 1 \rangle \mid \langle q, -1 \rangle
\end{aligned}$$

A transition formula is *locationless* iff it has no subformula of the form $\langle q, o \rangle$. Otherwise, it is *locationful*. For any locationless transition formula φ, let $\bar{\varphi}$ denote its dual, obtained by replacing any atomic subformula by its negation, and interchanging $\wedge$ and $\vee$.

A two-way alternating *register automaton* $\mathcal{A}$ is a tuple $\langle \Sigma, Q, q_I, n, \delta, \rho \rangle$ such that:

- Σ is a finite alphabet;
- Q is a finite set of locations;
- $q_I \in Q$ is the initial location;
- $n \in \mathbb{N}$ is the number of registers (given in unary);
- $\delta : Q \times \Sigma \to \Phi(Q, n)$ is the transition function;
- $\rho : Q \to \mathbb{N}$ specifies ranks such that, whenever $\langle q', o \rangle$ is a subformula of $\delta(q, a)$, we have $\rho(q') \leq \rho(q)$.

Suppose $\sigma \in \Sigma^{\leq\omega}(\sim)$. To define runs of $\mathcal{A}$ over σ, we first define a state of $\mathcal{A}$ for σ to be a triple $\langle i, q, v \rangle$ where i is an index of σ, q is a location of $\mathcal{A}$, and v is a register valuation for σ.

Next, we interpret the transition function δ by means of a satisfaction relation $S \models_v^{\sigma, i} \varphi$ where S is a finite set of states for σ and φ is a transition formula. The cases for Boolean constants and operators are standard, and treatments of dual clauses are as expected:

$$\begin{aligned}
S \models_v^{\sigma,i} \uparrow_r \sim \quad &\overset{\text{def}}{\Leftrightarrow} \quad v(r) \sim^{\sigma} i \text{ and } v(r) \text{ is defined} \\
S \models_v^{\sigma,i} \mathtt{beg} \quad &\overset{\text{def}}{\Leftrightarrow} \quad i = 0 \\
S \models_v^{\sigma,i} \mathtt{end} \quad &\overset{\text{def}}{\Leftrightarrow} \quad i = |\sigma| - 1 \\
S \models_v^{\sigma,i} \downarrow_r \varphi \quad &\overset{\text{def}}{\Leftrightarrow} \quad S \models_{v[r \mapsto i]}^{\sigma,i} \varphi \\
S \models_v^{\sigma,i} \langle q, o \rangle \quad &\overset{\text{def}}{\Leftrightarrow} \quad \langle i + o, q, v \rangle \in S
\end{aligned}$$

Now, a run of $\mathcal{A}$ of length $0 < \kappa \leq \omega$ over σ is a directed acyclic graph G consisting of:

- for each $0 \leq j < \kappa$, a finite set $G(j)$ (called a level) of states of $\mathcal{A}$ for σ, and
- for each j with $j+1 < \kappa$, an edge relation $\to_j$ between $G(j)$ and $G(j+1)$,

such that:

(i) $G(0) = \{\langle 0, q_I, \emptyset \rangle\}$, where $\emptyset$ is the empty register valuation;

(ii) for any j with nonempty $G(j)$, we have $j + 1 < \kappa$, $G(j+1) = \bigcup_{\langle i,q,v \rangle \in G(j)} S_{\langle i,q,v \rangle}$ where each $S_{\langle i,q,v \rangle}$ is some minimal set satisfying $S_{\langle i,q,v \rangle} \models_v^{\sigma,i} \delta(q, \sigma(i))$, and $\langle i, q, v \rangle \to_j \langle i', q', v' \rangle$ iff $\langle i', q', v' \rangle \in S_{\langle i,q,v \rangle}$.

Observe that, as a consequence of (ii), any finite run is complete, in the sense that its last level $G(\kappa-1)$ is necessarily empty. Note also that, in any valid run, for every $\langle i,q,v\rangle$ in any level, some $S_{\langle i,q,v\rangle}$ as in (ii) must exist.

Along any path π of a run, the location ranks are nonincreasing. Therefore, if π is infinite, the location ranks eventually have the same value, which we denote by $\rho(\pi)$. A data word is accepted iff it has a run such that, for each infinite path π, $\rho(\pi)$ is even. [1]

An automaton as above is one-way iff it contains no transition subformula of the form $\langle q,-1\rangle$. For one-way automata, any successor state of a state $\langle i,q,v\rangle$ is of the form $\langle i+1,q,v\rangle$. Over a finite data word σ, any run of a one-way automaton is finite. Hence, σ is accepted iff it has a run, i.e. the ranking function ρ is irrelevant.

An automaton is nondeterministic (i.e. existential) iff any transition subformula which is a conjunction of locationful formulae is of the form $(\varphi\vee\varphi')\wedge(\bar{\varphi}\vee\varphi'')$ where φ is locationless. For nondeterministic automata, any successor set of any state has size at most 1. Thus, any run is a sequence of states.

An automaton is universal iff any transition subformula which is a disjunction of locationful formulae is of the form $(\varphi\wedge\varphi')\vee(\bar{\varphi}\wedge\varphi'')$ where φ is locationless. For universal automata, any state has at most one successor set. Thus, any data word has at most one run.

An automaton is deterministic iff it is both nondeterministic and universal.

The classes of register automata above will be denoted by $dC\mathrm{RA}(\sim)$, where $d\in\{1,2\}$ and $C\in\{\mathrm{A},\mathrm{N},\mathrm{U},\mathrm{D}\}$ specify any restrictions on directionality and control. Let $dC\mathrm{RA}_n(\sim)$ denote the subclass with n registers.

Note 1 The definition of register automata above is suited to the uses in this paper, and has similarities with definitions in [6, 5, 16, 23]. Thus, it differs in some technical details from the definition of register automata in [14, 26, 21]. With the latter definition, in any state, the registers contain mutually distinct data values (or the default value $\sharp$), and the previous data value is always held in a register. It can be checked that, for any automaton with $n+1$ registers in the sense of [14, 26, 21], we can construct an equivalent automaton with $n+1$ registers and an equivalent alternating automaton with n registers in the sense of this paper.

2.4. Counter automata

A *counter automaton* (CA), with ε transitions and zero testing, is a tuple $\langle\Sigma,Q,q_I,n,\delta,F\rangle$ where Σ is a finite alphabet, Q is a finite set of locations, q_I is the initial location, $n\in\mathbb{N}$ is the number of counters (given in unary), $\delta\subseteq Q\times(\Sigma\uplus\{\varepsilon\})\times L\times Q$ is the transition relation over the instruction set $L=\{\mathtt{inc},\mathtt{dec},\mathtt{ifzero}\}\times\{1,\dots,n\}$, and $F\subseteq Q$ is the set of accepting locations.

A counter valuation is a function $\{1,\dots,n\}\to\mathbb{N}$. An error-free run over $w\in\Sigma^{<\omega}$ [resp. $w\in\Sigma^{\omega}$] is a finite [resp. infinite] sequence $\langle q_0,v_0\rangle\xrightarrow{w_0,l_0}\langle q_1,v_1\rangle\xrightarrow{w_1,l_1}\cdots$ observing the standard interpretation of the instructions ($\langle\mathtt{dec},c\rangle$ can be performed only if c is nonzero), where $q_0=q_I$, $v_0,v_1,\dots$ are counter valuations, v_0 assigns 0 to each counter, and $w=w_0w_1\dots$.

A finite run is accepting iff it ends with an accepting location. An infinite run is accepting iff it contains an accepting location infinitely often.

A *Minsky* CA has error-free runs. For Minsky CA, finitary nonemptiness is in Σ^0_1, and infinitary nonemptiness is in Σ^1_1. Already with 2 counters, infinitary nonemptiness is Σ^1_1-hard [2, Lemma 8], and finitary nonemptiness of deterministic Minsky CA is Σ^0_1-hard [19].

An *Incrementing* CA is defined as a Minsky CA except that its runs may contain errors which increase one or more counters. Formally, for counter valuations v and $v_\dagger$, we write $v\le v_\dagger$ iff, for all c, $v(c)\le v_\dagger(c)$. Runs of Incrementing CA are defined by replacing the relation $\xrightarrow{w,l}$ with the following: $\langle q,v\rangle\xrightarrow{w,l}_\dagger\langle q',v'\rangle$ iff there exist $v_\dagger$ and $v'_\dagger$ such that $v\le v_\dagger$, $\langle q,v_\dagger\rangle\xrightarrow{w,l}\langle q',v'_\dagger\rangle$, and $v'_\dagger\le v'$. When it is clear from the context that we are considering an Incrementing CA, we may write simply $\xrightarrow{w,l}$ instead of $\xrightarrow{w,l}_\dagger$.

Theorem 2 *For Incrementing CA, finitary nonemptiness is decidable and not primitive recursive;*[2] *infinitary nonemptiness is Π^0_1-complete.*

Proof. Decidability of finitary nonemptiness follows from the decidability of the coverability problem for Reset Petri nets [9] by reversing the computations, and non-primitive recursiveness from an adaptation of the result of [27]. Π^0_1-completeness of infinitary nonemptiness follows by adapting the proof of [24, Theorem 2], and by Π^0_1 membership of the recurrent-state problem for Insertion Channel Machines with Emptiness Testing [25]. □

2.5. Languages and decision problems

For an $\mathrm{LTL}^{\downarrow}(\sim;\mathcal{O})$ sentence ϕ over an alphabet Σ, let $\mathrm{L}^{\alpha}_{\Sigma}(\phi)=\{\sigma\in\Sigma^{\alpha}(\sim)\,:\,\sigma,0\models\phi\}$.

For an $\mathrm{FO}(\sim,<,+1)$ sentence ϕ over an alphabet Σ, let $\mathrm{L}^{\alpha}_{\Sigma}(\phi)=\{\sigma\in\Sigma^{\alpha}(\sim)\,:\,\sigma\models\phi\}$.

For a register automaton $\mathcal{A}$ whose alphabet is Σ, let $\mathrm{L}^{<\omega}(\mathcal{A})$ [resp. $\mathrm{L}^{\omega}(\mathcal{A})$] denote the set of all finite [resp. infinite] data words over Σ accepted by $\mathcal{A}$.

[1] Recall that we consider only nonempty data words. It is straightforward to extend the definitions to enable acceptance of the empty data word, but that is not necessary in this paper.

[2] Recall the Ritchie-Cobham property [22, page 297]: a decision problem (i.e. a set) is primitive recursive iff it is solvable in primitive recursive time/space.

For a counter automaton $\mathcal{C}$ whose alphabet is Σ, let $\mathrm{L}^{<\omega}(\mathcal{C})$ [resp. $\mathrm{L}^{\omega}(\mathcal{C})$] denote the set of all finite [resp. infinite] words over Σ accepted by $\mathcal{C}$.

The decision problems of *satisfiability* for a logical fragment, and *nonemptiness* for a class of automata, are defined as usual in terms of the languages above. A problem is *finitary* or *infinitary*, which specifies word length.

A sentence or register automaton is said to be *equivalent* to another one iff they are over the same alphabet and have the same finitary and infinitary languages.

3. Closure properties

Proposition 3 *Over finite and over infinite data words, and for any $d \in \{1, 2\}$, $dDRA_n(\sim)$ and $dARA_n(\sim)$ are closed under complement, and $dNRA_n(\sim)$ is dual to $dURA_n(\sim)$.*

Proof. It can be shown that the dual automaton recognises the complement language, by adapting complementation of alternating automata and using determinacy of weak parity games (see [18]). □

Proposition 4 *Over finite and over infinite data words:*

(a) Each automata class $1CRA(\sim)$ is closed under intersection and union. For intersections of universal or alternating automata, and for unions of nondeterministic or alternating automata, the maximum of the two numbers of registers suffices. Otherwise, their sum suffices.

(b) $2URA(\sim)$ is closed under intersection, $2NRA(\sim)$ is closed under union, and $2ARA(\sim)$ is closed under intersection and union. The maximum of the two numbers of registers suffices.

Proof. By branching and product constructions. □

4. Expressiveness

Suppose $m \geq 1$. We write $\mathtt{X}^m$ [resp. $\mathtt{X}^{-m}$] to denote the temporal operator made of $m \geq 1$ successive operators $\mathtt{X}$ [resp. $\mathtt{X}^{-1}$]. The operators $\mathtt{X}^m\mathtt{F}$ and $\mathtt{X}^{-m}\mathtt{F}^{-1}$ are defined analogously. Let $\mathcal{O}_m$ denote the following set of temporal operators: $\{\mathtt{X}, \mathtt{X}^{-1}, \ldots, \mathtt{X}^m, \mathtt{X}^{-m}, \mathtt{X}^{m+1}\mathtt{F}, \mathtt{X}^{-(m+1)}\mathtt{F}^{-1}\}$.

An $\mathrm{LTL}_1^{\downarrow}(\sim; \mathcal{O}_m)$ formula is said to be *simple* iff any occurrence of a temporal operator is immediately preceded by $\downarrow_1$ (and there are no other occurences of $\downarrow_1$).

Example 5 Since the Boolean operators and the freeze quantifier commute, the simple fragment of $\mathrm{LTL}_1^{\downarrow}(\sim; \mathcal{O}_1)$ can express $\downarrow_1 \mathtt{F}$, $\downarrow_1 \mathtt{G}$ and $\downarrow_1 \mathtt{XG}$. Hence, there is a simple formula in $\mathrm{LTL}_1^{\downarrow}(\sim; \mathcal{O}_1)$ equivalent to the 'nonces' sentence $\mathtt{G} \downarrow_1 \mathtt{XG}\neg \uparrow_1 \sim$ which states that no class of $\sim$ contains more than one element.

An $\mathrm{LTL}_1^{\downarrow}(\sim; \mathcal{O}_m)$ sentence ϕ is said to be equivalent to an $\mathrm{FO}^2(\sim, <, +1, \ldots, +m)$ formula $\phi'(x_j)$ iff they are over the same alphabet Σ and, for every $\sigma \in \Sigma^{\leq\omega}(\sim)$ and index i of σ, we have $\sigma, i \models_{\emptyset} \phi \Leftrightarrow \sigma \models_{[x_j \mapsto i]} \phi'(x_j)$.

Proposition 6 *(a) For any simple $\mathrm{LTL}_1^{\downarrow}(\sim; \mathcal{O}_m)$ sentence, an equivalent $\mathrm{FO}^2(\sim, <, +1, \ldots, +m)$ formula is computable in logspace.*

(b) For any $\mathrm{FO}^2(\sim, <, +1, \ldots, +m)$ formula $\phi(x_j)$, an equivalent simple $\mathrm{LTL}_1^{\downarrow}(\sim; \mathcal{O}_m)$ formula is computable in pspace.

Proof. First, we introduce some convenient notation. Let $\mathtt{O}^0 = \downarrow_1$, $\mathtt{O}^k = \downarrow_1 \mathtt{X}^k$ for $k \in \{-m, \ldots, -1, 1, \ldots, m\}$, $\mathtt{O}^{m+1} = \downarrow_1 \mathtt{X}^{m+1}\mathtt{F}$, and $\mathtt{O}^{-(m+1)} = \downarrow_1 \mathtt{X}^{-(m+1)}\mathtt{F}^{-1}$. For $j \in \{0, 1\}$, let

$$\begin{aligned}
\chi_0^j &\stackrel{\text{def}}{=} x_{1-j} = x_j \\
\chi_k^j &\stackrel{\text{def}}{=} x_{1-j} = x_j + k \ (1 \leq k \leq m) \\
\chi_{-k}^j &\stackrel{\text{def}}{=} x_j = x_{1-j} + k \ (1 \leq k \leq m) \\
\chi_{m+1}^j &\stackrel{\text{def}}{=} x_j < x_{1-j} \wedge \textstyle\bigwedge_{1 \leq k \leq m} \neg x_{1-j} = x_j + k \\
\chi_{-(m+1)}^j &\stackrel{\text{def}}{=} x_{1-j} < x_j \wedge \textstyle\bigwedge_{1 \leq k \leq m} \neg x_j = x_{1-j} + k
\end{aligned}$$

(The equality predicate can be expressed using $<$.)

We have (a) by the following translations T_j which map simple $\mathrm{LTL}_1^{\downarrow}(\sim; \mathcal{O}_m)$ formulae to $\mathrm{FO}^2(\sim, <, +1, \ldots, +m)$ formulae. Any sentence ϕ will be equivalent to $T_j(\phi)$ which will contain at most x_j free. T_j are defined by structural recursion, by encoding the semantics of simple formulae into first-order logic, and by recycling variables (to use only two variables). The Boolean clauses are omitted.

$$T_j(a) \stackrel{\text{def}}{=} P_a(x_j) \qquad T_j(\uparrow_1 \sim) \stackrel{\text{def}}{=} x_{1-j} \sim x_j$$
$$T_j(\mathtt{O}^k\psi) \stackrel{\text{def}}{=} \exists x_{1-j}(\chi_k^j \wedge T_{1-j}(\psi))$$

For (b), we proceed by adapting the proof of [10, Theorem 1]. We define recursively translations T'_j from $\mathrm{FO}^2(\sim, <, +1, \ldots, +m)$ formulae $\phi(x_j)$ to equivalent simple $\mathrm{LTL}_1^{\downarrow}(\sim; \mathcal{O}_m)$ sentences. The cases of Boolean operators and one-variable atomic formulae are straightforward. The remaining case is when $\phi(x_j)$ is of the form

$$\exists x_{1-j}\, \beta(\alpha_1(x_0, x_1), \ldots, \alpha_L(x_0, x_1), \\ \xi_1(x_j), \ldots, \xi_N(x_j), \zeta_1(x_{1-j}), \ldots, \zeta_M(x_{1-j}))$$

where β is a Boolean formula, and each $\alpha_i(x_0, x_1)$ is a $\sim$, $<$ or $+k$ atomic formula. Now, for any $-(m+1) \leq k \leq m+1$ and $\bowtie \in \{\sim, \not\sim\}$, let $\alpha_i^{k,\bowtie}$ denote the truth value of $\alpha_i(x_0, x_1)$ under the assumption $\chi_k^j \wedge x_j \bowtie x_{1-j}$. Also, for any $X \subseteq \{1, \ldots, N\}$, let $\xi_i^X = \top$ if $i \in X$, and $\xi_i^X = \bot$ otherwise. $T'_j(\phi(x_j))$ is then computed as

$$\textstyle\bigvee_{-(m+1) \leq k \leq m+1} \bigvee_{\bowtie \in \{\sim, \not\sim\}} \bigvee_{X \subseteq \{1, \ldots, N\}} \\ \big(\bigwedge_{i \in \{1, \ldots, N\}} T'_j(\xi_i(x_j)) \Leftrightarrow \xi_i^X\big) \wedge \\ \mathtt{O}^k(\uparrow_1 \bowtie \wedge\ \beta(\alpha_1^{k,\bowtie}, \ldots, \alpha_L^{k,\bowtie}, \\ \xi_1^X, \ldots, \xi_N^X, T'_{1-j}(\zeta_1(x_{1-j})), \ldots, T'_{1-j}(\zeta_M(x_{1-j}))))$$

The size of the equivalent simple $\mathrm{LTL}_1^{\downarrow}(\sim;\mathcal{O}_m)$ formula is exponential in $|\phi|$, because the length of the stack of recursive calls is linear and generalized conjunctions and disjunctions have at most exponentially many arguments. For the same reasons, polynomial space is sufficient for the computation. □

Corollary 7 *Over finite and over infinite data words, simple $\mathrm{LTL}_1^{\downarrow}(\sim;\mathcal{O}_m)$ satisfiability is logspace reducible to $FO^2(\sim,<,+1,\ldots,+m)$ satisfiability, and there is a pspace reduction in the reverse direction.*

We now turn to translating temporal formulae to register automata.

Theorem 8 *For any $\mathrm{LTL}_n^{\downarrow}(\sim;\mathtt{X},\mathtt{X}^{-1},\mathtt{U},\mathtt{U}^{-1})$ sentence, an equivalent automaton in $2ARA_n(\sim)$ is constructible in logspace. For formulae with only future-time operators, the automata are one-way.*

Proof. By a standard logspace reduction, we can assume that ϕ over an alphabet Σ is a sentence in $\mathrm{LTL}_n^{\downarrow}(\sim;\mathtt{X},\mathtt{X}^{-1},\mathtt{U},\mathtt{U}^{-1})$ extended by the duals $\bot$, $\vee$, $\bar{\mathtt{X}}$, $\bar{\mathtt{X}}^{-1}$, $\bar{\mathtt{U}}$, $\bar{\mathtt{U}}^{-1}$ of $\top$, $\wedge$, $\mathtt{X}$, $\mathtt{X}^{-1}$, $\mathtt{U}$, $\mathtt{U}^{-1}$ (respectively), and that ϕ is in negation normal form, i.e. such that $\neg$ occurs only in front of atomic formulae.

To construct an equivalent $\mathcal{A}_\phi^\Sigma = \langle \Sigma, Q, n, q_I, \delta, \rho\rangle$ in $\mathrm{2ARA}_n(\sim)$, let $Q = \mathrm{cl}(\phi)$, where $\mathrm{cl}(\phi)$ is the set of all subformulae of ϕ, and let $q_I = \phi$.

The transition function is defined as follows. It is homomorphic for $\top$, $\bot$, $\wedge$ and $\vee$ formulae, and the clauses for $\bar{\mathtt{U}}$, $\mathtt{U}^{-1}$ and $\bar{\mathtt{U}}^{-1}$ are similar to the clause for $\mathtt{U}$.

$$\begin{aligned}
\delta(b,a) &= \text{truth value of } b = a\\
\delta(\neg b,a) &= \text{truth value of } b \neq a\\
\delta(\uparrow_r\sim,a) &= \uparrow_r\sim\\
\delta(\neg\uparrow_r\sim,a) &= \uparrow_r\not\sim\\
\delta(\mathtt{X}\psi,a) &= \langle\psi,1\rangle\\
\delta(\bar{\mathtt{X}}\psi,a) &= \mathtt{end}\vee\langle\psi,1\rangle\\
\delta(\mathtt{X}^{-1}\psi,a) &= \langle\psi,-1\rangle\\
\delta(\bar{\mathtt{X}}^{-1}\psi,a) &= \mathtt{beg}\vee\langle\psi,-1\rangle\\
\delta(\psi\mathtt{U}\psi',a) &= \delta(\psi',a)\vee(\delta(\psi,a)\wedge\langle\psi\mathtt{U}\psi',1\rangle)\\
\delta(\downarrow_r\psi,a) &= \downarrow_r\delta(\psi,a)
\end{aligned}$$

To complete the construction, for every $\psi \in \mathrm{cl}(\phi)$, let $\rho(\psi) = 2\times|\psi|+1$ if the outermost construct in ψ is $\mathtt{U}$, and let $\rho(\psi) = 2\times|\psi|$ otherwise. As required, we have that, whenever $\langle q', o\rangle$ is a subformula of $\delta(q,a)$, $\rho(q') \le \rho(q)$.

Showing $\mathrm{L}_\Sigma^{\le\omega}(\phi) = \mathrm{L}^{\le\omega}(\mathcal{A}_\phi^\Sigma)$ is routine. □

Example 9 Let $\phi = \mathtt{G}\downarrow_1\mathtt{XG}\neg\uparrow_1\sim$ be the nonces sentence. Recalling that $\mathtt{G}\psi$ is equivalent to $\bot\bar{\mathtt{U}}\psi$, the transition formulae $\delta(\mathtt{G}\psi,a)$ in the construction of Theorem 8 simplify to $\delta(\psi,a)\wedge(\mathtt{end}\vee\langle\mathtt{G}\psi,1\rangle)$. Figure 1 represents the resulting one-way register automaton. The forks are read as conjunctions, and edge labels $\mathtt{nend}$ and $\uparrow_1\sim$ as implication premises. Thus, the edges from location $\mathtt{G}\neg\uparrow_1\sim$ represent the transition formula

$$(\uparrow_1\sim\Rightarrow\bot)\wedge(\mathtt{nend}\Rightarrow\langle\mathtt{G}\neg\uparrow_1\sim,1\rangle)$$

or equivalently $\uparrow_1\not\sim\wedge(\mathtt{end}\vee\langle\mathtt{G}\neg\uparrow_1\sim,1\rangle)$. The rank of location ϕ is larger than the rank of location $\mathtt{G}\neg\uparrow_1\sim$, and both are even. Observe that the automaton is universal, as it does not contain a disjunction between locationful transition subformulae.

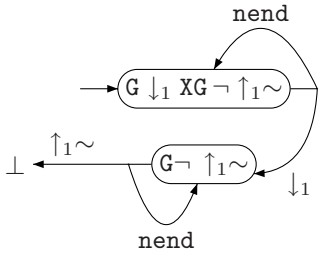

Figure 1. A one-way universal RA

5. Upper complexity bounds

The following warm-up theorem contains basic membership results. In some cases, they will be matched by the hardness results in Section 6.

Theorem 10 *Over finite data words, satisfiability for $\mathrm{LTL}^{\downarrow}(\sim;\mathtt{X},\mathtt{X}^{-1},\mathtt{U},\mathtt{U}^{-1})$ and nonemptiness for $2ARA(\sim)$ are in Σ_1^0. Over infinite data words, satisfiability for $\mathrm{LTL}^{\downarrow}(\sim;\mathtt{X},\mathtt{X}^{-1},\mathtt{U},\mathtt{U}^{-1})$ and nonemptiness for $2ARA(\sim)$ are in Σ_1^1, and nonemptiness for $2NRA(\sim)$ is in Σ_2^0.*

Proof. For encoding runs of alternating automata, we use the following observation: a run is accepting iff, at infinitely many levels we have that, for each state of odd rank, all its successors at some subsequent level have strictly smaller ranks. We also note that, for any finite data word, any automaton has finitely many states. □

In [26], finitary nonemptiness for $1\mathrm{NRA}(\sim)$ was shown to be in NP, but their proof does not carry over to the definition of register automata in this paper: see Note 1 and Theorem 14.

Theorem 11 *Finitary nonemptiness and infinitary nonemptiness for $1NRA(\sim)$ are in* PSPACE.

Proof. By a logspace reduction to satisfiability for CLTL$(\mathbb{N},=)$, which is PSPACE-complete [8]. □

By Corollary 7 and [4], finitary and infinitary satisfiability for simple $\mathrm{LTL}_1^{\downarrow}(\sim;\mathtt{X},\mathtt{X}^{-1},\mathtt{XXF},\mathtt{X}^{-1}\mathtt{X}^{-1}\mathtt{F}^{-1})$ are reducible to reachability for Petri nets. The latter problem is decidable, so the former also are. It is not known whether reachability for Petri nets is elementary.

It was shown in [14, Appendix A] that, over finite data words, for $\mathcal{A}$ in $1\mathrm{NRA}_1(\sim)$ and $\mathcal{A}'$ in $1\mathrm{NRA}(\sim)$, containment of $\mathcal{A}'$ in $\mathcal{A}$ is decidable.[3] By Theorem 2 and Proposition 3, the result in [14, Appendix A] is subsumed by the following theorem. The proofs in [14, Appendix A] and of Theorem 2 both involve well-quasi-orders.

Theorem 12 *For $\mathcal{A}$ in $1ARA_1(\sim)$ and $\mathcal{A}'$ in $1NRA(\sim)$, finitary [resp. infinitary] nonemptiness of the intersection of $\mathcal{A}$ and $\mathcal{A}'$ is pspace reducible to finitary [resp. infinitary] nonemptiness of an Incrementing CA.*

Proof. Suppose that $\mathcal{A} = \langle \Sigma, Q, q_I, 1, \delta, \rho\rangle$ and $\mathcal{A}' = \langle \Sigma', Q', q'_I, n, \delta', \rho'\rangle$. We can assume $\Sigma = \Sigma'$.

We first consider finitary nonemptiness. Suppose $\sigma \in \Sigma^{<\omega}(\sim)$, $0 \le i < |\sigma|$, and G and G' are partial runs of $\mathcal{A}$ and $\mathcal{A}'$ (respectively) up to ith level. Any element of $G(i)$ is of the form $\langle i, q, v\rangle$, and $G'(i)$ is either a singleton $\{\langle i, q', v'\rangle\}$ or empty.

Observe that the intersection of $\mathcal{A}$ and $\mathcal{A}'$ is nonempty iff, from some such $\langle \sigma, i, G(i), G'(i)\rangle$, an empty $(i+1)$th level is reachable.

Now, quadruples $\langle \sigma, i, G(i), G'(i)\rangle$ as above will be represented by *configurations*, which are tuples of the form

$$\langle c, Q_u, q', E, f, \varphi_b, \varphi_e\rangle$$

where c is a function from $\mathcal{P}^+(Q)$ (the set of all nonempty subsets of Q) to $\mathbb{N}$, $Q_u \in \mathcal{P}(Q)$, $q' \in Q'$ or $q' = \emptyset$, E is an equivalence relation on a subset of $\{1,\ldots,n\}$, f is a function from classes of E to $\mathcal{P}(Q)$, $\varphi_b \in \{\mathtt{beg}, \mathtt{nbeg}\}$, and $\varphi_e \in \{\mathtt{end}, \mathtt{nend}\}$.

A configuration $\langle c, Q_u, q', E, f, \varphi_b, \varphi_e\rangle$ represents a quadruple $\langle \sigma, i, G(i), G'(i)\rangle$ as above iff:

- for any $Q^\dagger \in \mathcal{P}^+(Q)$, $c(Q^\dagger)$ is the number of classes S of $\approx$ such that $\pi_2(S) = Q^\dagger$ and $\pi_3(S)$ is not in the same class of $\sim^\sigma$ as any $v'(r)$ as below, where $\approx$ is the equivalence relation on the set of all $\langle i, q, v\rangle \in G(i)$ with $v \neq \emptyset$ defined by

  $$\langle i, q_1, [1 \mapsto j_1]\rangle \approx \langle i, q_2, [1 \mapsto j_2]\rangle \;\Leftrightarrow\; j_1 \sim^\sigma j_2$$

  and π_k is kth-component tuple projection;
- $Q_u = \{q : \langle i, q, \emptyset\rangle \in G(i)\}$;
- if $G'(i) = \emptyset$ then $q' = \emptyset$, otherwise we have $G'(i) = \{\langle i, q', v'\rangle\}$, $r_1\, E\, r_2$ iff $r_1, r_2 \in \mathrm{dom}(v')$ and $v'(r_1) \sim^\sigma v'(r_2)$, and $f(\bar r) = Q^\dagger$ if there exists a class S of $\approx$ such that $\pi_2(S) = Q^\dagger$ and $\pi_3(S)$ is in the same class of $\sim^\sigma$ as $v'(r)$, otherwise $f(\bar r) = \emptyset$;
- $\varphi_b = \mathtt{beg}$ iff $i = 0$, and $\varphi_e = \mathtt{end}$ iff $i = |\sigma| - 1$.

For φ a transition formula in $\mathcal{A}$, $\varphi_b \in \{\mathtt{beg}, \mathtt{nbeg}\}$, $\varphi_e \in \{\mathtt{end}, \mathtt{nend}\}$, $\bowtie \in \{\sim, \not\sim\}$, and $Q^=, Q^\neq \in \mathcal{P}(Q)$, we write $\varphi \rightsquigarrow_{\bowtie}^{\varphi_b,\varphi_e} Q^=, Q^\neq$ iff there exist $\sigma \in \Sigma^{<\omega}(\sim)$, $0 \le i < |\sigma|$, a partial function v_1 from $\{1\}$ to $\{0,\ldots,i-1\}$, and a minimal S such that $S \models_{v_1}^{\sigma,i} \varphi$, $\varphi_b = \mathtt{beg}$ iff $i = 0$, $\varphi_e = \mathtt{end}$ iff $i = |\sigma| - 1$, $v_1(1) \bowtie^\sigma i$,

$$\begin{aligned} Q^= &= \{q : \langle i+1, q, [1 \mapsto i]\rangle \in S\} \\ Q^\neq &= \{q : \langle i+1, q, [1 \mapsto j]\rangle \in S \wedge j \neq i\} \end{aligned}$$

Observe that it suffices to consider σ with $|\sigma| \le 3$.

Similarly, for φ' a transition formula in $\mathcal{A}'$, $\varphi_b \in \{\mathtt{beg}, \mathtt{nbeg}\}$, $\varphi_e \in \{\mathtt{end}, \mathtt{nend}\}$, $R_1, R_2 \subseteq \{1,\ldots,n\}$, and $q' \in Q'$, we write $\varphi' \rightsquigarrow_{R_1}^{\varphi_b,\varphi_e} q', R_2$ iff there exist $\sigma \in \Sigma^{<\omega}(\sim)$, $0 \le i < |\sigma|$, a partial function v'_1 from $\{1,\ldots,n\}$ to $\{0,\ldots,i-1\}$, and a minimal S such that $S \models_{v'_1}^{\sigma,i} \varphi'$, $\varphi_b = \mathtt{beg}$ iff $i = 0$, $\varphi_e = \mathtt{end}$ iff $i = |\sigma| - 1$, $R_1 = \{r : v'_1(r) \sim^\sigma i\}$, and $S = \{\langle i+1, q', v'_2\rangle\}$ with $R_2 = \{r : i = v'_2(r)\}$. Whenever the above is satisfied with $S = \emptyset$, we write $\varphi' \rightsquigarrow_{R_1}^{\varphi_b,\varphi_e} \emptyset$. Observe that, in any case, it suffices to consider σ with $|\sigma| \le 3$.

We are now ready to begin constructing an Incrementing CA $\mathcal{C}$ with the alphabet Σ. $\mathcal{C}$ will have $2 \times (2^{|Q|} - 1)$ counters, which we denote by $c(Q^\dagger)$ and $\hat c(Q^\dagger)$ for $Q^\dagger \in \mathcal{P}^+(Q)$. The locations, initial location, transition relation and accepting locations are constructed so that $\mathcal{C}$ performs the following:

(1) A configuration $\langle c, Q_u, q', E, f, \varphi_b, \varphi_e\rangle$ is kept in each state of $\mathcal{C}$. Initially, $Q_u = \{q_I\}$, $q' = q'_I$, $E = f = \emptyset$, $\varphi_b = \mathtt{beg}$, and φ_e is chosen (nondeterministically, by ε transitions).

(2) At this point, $\mathcal{C}$ performs an $a \in \Sigma$ transition, chooses

$$C \in \{Q^\dagger : c(Q^\dagger) > 0\} \cup \{\bar r : q' \neq \emptyset \wedge r\, E\, r\} \cup \{\not\sim\}$$

which specifies whether the current index is regarded as belonging to a class represented in the current configuration, and proceeds to compute a successor configuration $\langle \hat c, \hat Q_u, \hat q', \hat E, \hat f, \hat\varphi_b, \hat\varphi_e\rangle$.

The successor configurations will represent exactly quadruples $\langle \sigma, i+1, G(i+1), G'(i+1)\rangle$ such that the current configuration represents $\langle \sigma, i, G(i), G'(i)\rangle$, and either $C = \pi^2(S)$ for some class S of $\approx$ whose register values are in the class of i and where the register values in $G'(i)$ are not in the class of i, or C consists of those registers whose values in $G'(i)$ are in the class

[3] The exact statement allows $\mathcal{A}$ to have 2 registers, but see Note 1.

of i, or $C = \not\sim$ and there are no register values in $G(i)$ or $G'(i)$ which are in the class of i.

(3) If $q' = \emptyset$, then $\hat{q}' = \emptyset$. Otherwise, let $R_1 = \emptyset$ if $C = Q^\dagger$ or $C = \not\sim$, and $R_1 = C$ if $C = \bar{r}$. $\mathcal{C}$ then chooses a completion of $\delta'(q', a) \rightsquigarrow_{R_1}^{\varphi_b,\varphi_e}$: either $\hat{q}', R_2$ or $\emptyset$.

- If $\hat{q}', R_2$ was chosen, $\mathcal{C}$ obtains $\hat{E}$ from E by making the registers in $R_1 \cup R_2$ related only to each other.
- Otherwise, $\hat{q}' = \emptyset$.

An auxiliary $Q^\sim \in \mathcal{P}(Q)$ is initialised to $\emptyset$. At the end of (6), $Q^\sim$ will be equal to $\{q : \langle i+1, q, [1 \mapsto j]\rangle \in G(i+1) \wedge j \sim i\}$.

(4) While there exists $Q^\dagger \in \mathcal{P}^+(Q)$ such that $c(Q^\dagger) > 0$, $\mathcal{C}$ chooses and processes a successor set from each location in $Q^\dagger$ with respect to a and C:

- If $C = Q^\dagger$, $\mathcal{C}$ chooses, for each $q \in Q^\dagger$, $Q_q^=$ and $Q_q^{\neq}$ with $\delta(q, a) \rightsquigarrow_{\sim}^{\varphi_b,\varphi_e} Q_q^=, Q_q^{\neq}$. $Q^\sim$ is then updated to $Q^\sim \cup \bigcup_{q\in Q^\dagger} Q_q^= \cup Q_q^{\neq}$, and C to $\not\sim$.
- Otherwise, $\mathcal{C}$ chooses, for each $q \in Q^\dagger$, $Q_q^=$ and $Q_q^{\neq}$ with $\delta(q, a) \rightsquigarrow_{\not\sim}^{\varphi_b,\varphi_e} Q_q^=, Q_q^{\neq}$. $Q^\sim$ is then updated to $Q^\sim \cup \bigcup_{q\in Q^\dagger} Q_q^=$, and if $\bigcup_{q\in Q^\dagger} Q_q^{\neq} \neq \emptyset$, $\hat{c}(\bigcup_{q\in Q^\dagger} Q_q^{\neq})$ is incremented.

Now, $c(Q^\dagger)$ is decremented, and (4) is repeated.

(5) For each $q \in Q_u$, $\mathcal{C}$ chooses $Q_q^=$ and $Q_q^{\neq}$ such that $\delta(q, a) \rightsquigarrow_{\not\sim}^{\varphi_b,\varphi_e} Q_q^=, Q_q^{\neq}$. $Q^\sim$ is then updated to $Q^\sim \cup \bigcup_{q\in Q_u} Q_q^=$, and $\hat{Q}_u$ is set to $\bigcup_{q\in Q_u} Q_q^{\neq}$.

(6) For each $\bar{r}$ such that $f(\bar{r}) \neq \emptyset$, $\mathcal{C}$ chooses and processes a successor set from each location in $f(\bar{r})$ with respect to a and C:

- If $C = \bar{r}$, $\mathcal{C}$ chooses, for each $q \in f(\bar{r})$, $Q_q^=$ and $Q_q^{\neq}$ with $\delta(q, a) \rightsquigarrow_{\sim}^{\varphi_b,\varphi_e} Q_q^=, Q_q^{\neq}$. $Q^\sim$ is then updated to $Q^\sim \cup \bigcup_{q\in f(\bar{r})} Q_q^= \cup Q_q^{\neq}$.
- Otherwise ($R_1 = \emptyset$), $\mathcal{C}$ chooses, for each $q \in f(\bar{r})$, $Q_q^=$ and $Q_q^{\neq}$ with $\delta(q, a) \rightsquigarrow_{\not\sim}^{\varphi_b,\varphi_e} Q_q^=, Q_q^{\neq}$. $Q^\sim$ is then updated to $Q^\sim \cup \bigcup_{q\in f(\bar{r})} Q_q^=$. If $\hat{q}' \neq \emptyset$ and $\bar{r} \not\subseteq R_2$, then $\hat{f}(\bar{r} \setminus R_2)$ is set to $\bigcup_{q\in f(\bar{r})} Q_q^{\neq}$. Otherwise, and if $\bigcup_{q\in f(\bar{r})} Q_q^{\neq} \neq \emptyset$, $\hat{c}(\bigcup_{q\in f(\bar{r})} Q_q^{\neq})$ is incremented.

(7) If $\hat{q}' \neq \emptyset$ and $R_1 \cup R_2 \neq \emptyset$, $\hat{f}(R_1 \cup R_2)$ is set to $Q^\sim$. Otherwise, and if $Q^\sim \neq \emptyset$, $\hat{c}(Q^\sim)$ is incremented.

(8) If $\hat{c}(Q^\dagger) = 0$ for each $Q^\dagger \in \mathcal{P}^+(Q)$, $\hat{Q}_u = \emptyset$, and $\hat{q}' = \emptyset$, $\mathcal{C}$ stops at an accepting location if $\varphi_e = \texttt{end}$, or goes to a location from which it accepts all nonempty words if $\varphi_e = \texttt{nend}$. Otherwise, and if $\varphi_e = \texttt{nend}$, then $\hat{\varphi}_b$ is set to $\texttt{nbeg}$, $\hat{\varphi}_e$ is chosen, $\langle c, Q_u, q', E, f, \varphi_b, \varphi_e\rangle$ is replaced by $\langle \hat{c}, \hat{Q}_u, \hat{q}', \hat{E}, \hat{f}, \hat{\varphi}_b, \hat{\varphi}_e\rangle$, and $\mathcal{C}$ repeats from (2).

Now, suppose the intersection of $\mathcal{A}$ and $\mathcal{A}'$ contains a finite data word σ. Let G and G' be runs of $\mathcal{A}$ and $\mathcal{A}'$ (respectively) over σ. As above, the levels of G and G' are represented by some configurations $\langle c_i, Q_u^i, q_i', E_i, f_i, \varphi_b^i, \varphi_e^i\rangle$ for $i = 0, \ldots, |\sigma| - 1$. By the construction of $\mathcal{C}$, it has an error-free accepting finite run, such that the configurations whenever (2) is begun are exactly $\langle c_i, Q_u^i, q_i', E_i, f_i, \varphi_b^i, \varphi_e^i\rangle$ for $i = 0, \ldots, |\sigma| - 1$.

Conversely, suppose $\mathcal{C}$ has an accepting finite run. Let $\langle \tilde{c}_i, \tilde{Q}_u^i, \tilde{q}_i', \tilde{E}_i, \tilde{f}_i, \tilde{\varphi}_b^i, \tilde{\varphi}_e^i\rangle$ for $i = 0, \ldots, k-1$ be the configurations whenever (2) is begun. By the construction of $\mathcal{C}$, it has an error-free accepting finite run. More precisely, the configurations whenever (2) is begun are some $\langle c_i, Q_u^i, q_i', E_i, f_i, \varphi_b^i, \varphi_e^i\rangle$ for $i = 0, \ldots, k-1$ such that, for each i, we have $c_i \sqsubseteq \tilde{c}_i$, $Q_u^i = \tilde{Q}_u^i$, $q_i' = \tilde{q}_i'$, $E_i = \tilde{E}_i$, $f_i(\bar{r}) \subseteq \tilde{f}_i(\bar{r})$ for all $\bar{r}$, $\varphi_b^i = \tilde{\varphi}_b^i$ and $\varphi_e^i = \tilde{\varphi}_e^i$, where $c_i \sqsubseteq \tilde{c}_i$ iff there exists an injection

$$\iota : \{\langle Q^\dagger, j\rangle : Q^\dagger \in \mathcal{P}^+(Q) \wedge j \in \{1, \ldots, c_i(Q^\dagger)\}\} \to \{\langle \tilde{Q}^\dagger, j\rangle : \tilde{Q}^\dagger \in \mathcal{P}^+(Q) \wedge j \in \{1, \ldots, \tilde{c}_i(\tilde{Q}^\dagger)\}\}$$

such that, whenever $\iota(\langle Q^\dagger, j\rangle) = \langle \tilde{Q}^\dagger, \tilde{j}\rangle$, we have $Q^\dagger \subseteq \tilde{Q}^\dagger$. It follows that $\mathcal{A}$ and $\mathcal{A}'$ have runs G and G' (respectively) over a finite data word σ of length k, whose levels are represented by the configurations $\langle c_i, Q_u^i, q_i', E_i, f_i, \varphi_b^i, \varphi_e^i\rangle$.

To complete the proof for finitary nonemptiness, polynomial space suffices for the construction of $\mathcal{C}$ because: the number of counters is exponential in $|Q|$; each branching or iteration in $\mathcal{C}$ is at most polynomial in $|\Sigma|$ or $|Q'|$, or exponential in $|Q|$ or n, and their nesting is bounded by a constant; for each of the relations $\varphi \rightsquigarrow_{\bowtie}^{\varphi_b,\varphi_e} Q^=, Q^{\neq}$, $\varphi' \rightsquigarrow_{R_1}^{\varphi_b,\varphi_e} q', R_2$ and $\varphi' \rightsquigarrow_{R_1}^{\varphi_b,\varphi_e} \emptyset$, membership is in PSPACE.

Infinitary nonemptiness is handled in the same way, except that the construction is extended in order to transform alternating weak parity acceptance into nondeterministic Büchi acceptance, similarly as in the proof of [20, Theorem 5.1]. □

Corollary 13 *Finitary [resp. infinitary] satisfiability for $\mathrm{LTL}_1^{\downarrow}(\sim; \mathtt{X}, \mathtt{U})$ is pspace reducible to finitary [resp. infinitary] nonemptiness of Incrementing CA.*

Proof. By Theorems 8 and 12. □

6. Lower complexity bounds

The following result matches PSPACE membership shown in Theorem 11, even for deterministic automata.

Theorem 14 *Finitary nonemptiness and infinitary nonemptiness for 1DRA$(\sim)$ are* PSPACE*-hard.*

Proof. By reducing from QBF. □

By [4] and Corollary 7, reachability for Petri nets is PTIME reducible to finitary and infinitary satisfiability for simple $\mathrm{LTL}_1^{\downarrow}(\sim; \mathtt{X}, \mathtt{X}^{-1}, \mathtt{XXF}, \mathtt{X}^{-1}\mathtt{X}^{-1}\mathtt{F}^{-1})$. The former problem is EXPSPACE-hard, so the latter also are.

We now turn to hardness results for fragments of $\mathrm{LTL}^{\downarrow}(\sim; \mathtt{X}, \mathtt{X}^{-1}, \mathtt{U}, \mathtt{U}^{-1})$ without the simplicity restriction, and for register automata which are not one-way and nondeterministic.

The following shows that the problems in Theorem 12 in Corollary 13 are also not easier than nonemptiness of Incrementing CA (see Theorem 2), already for nonemptiness of universal automata and with F instead of U.

Theorem 15 *In both finitary and infinitary cases, nonemptiness of Incrementing CA is logspace reducible to satisfiability for $\mathrm{LTL}_1^{\downarrow}(\sim; \mathtt{X}, \mathtt{F})$ and nonemptiness for 1URA$_1(\sim)$.*

Proof. For the finitary case, suppose $\mathcal{C} = \langle \Sigma, Q, q_I, n, \delta, F \rangle$ is an Incrementing CA. Let $L = \{\mathtt{inc}, \mathtt{dec}, \mathtt{ifzero}\} \times \{1, \ldots, n\}$, and $\hat{\Sigma} = Q \times (\Sigma \cup \{\varepsilon\}) \times L \times Q$. For any $\sigma \in \hat{\Sigma}^{<\omega}(\sim)$, where $\mathrm{str}(\sigma) = \langle q_0, w_0, l_0, q_0' \rangle \langle q_1, w_1, l_1, q_1' \rangle \cdots$, let $\bar{\sigma} = w_0 w_1 \cdots$.

To ensure that $\sigma \in \hat{\Sigma}^{<\omega}(\sim)$ corresponds to a run of $\mathcal{C}$, we constrain the equivalence relation $\sim^{\sigma}$. Firstly, there must not be two $\langle \mathtt{inc}, c \rangle$ transitions, or two $\langle \mathtt{dec}, c \rangle$ transitions (with the same c) in the same class. For an $\langle \mathtt{ifzero}, c \rangle$ transition to be correct, whenever it is preceded by $\langle \mathtt{inc}, c \rangle$, there must be an intermediate $\langle \mathtt{dec}, c \rangle$ in the same class. Incrementing errors may occur because a $\langle \mathtt{dec}, c \rangle$ transition may be preceded by no $\langle \mathtt{inc}, c \rangle$ in the same class. Such a $\langle \mathtt{dec}, c \rangle$ transition corresponds to a faulty decrement which leaves c unchanged. Now, it is easy to check that, for any run of $\mathcal{C}$, there is a run which differs at most in counter values and whose only incrementing errors are such faulty decrements.

More precisely, $w \in \Sigma^{<\omega}$ is accepted by $\mathcal{C}$ iff $w = \bar{\sigma}$ for some $\sigma \in \hat{\Sigma}^{<\omega}(\sim)$ which satisfies the following, where $\mathrm{str}(\sigma) = \langle q_0, w_0, l_0, q_0' \rangle \langle q_1, w_1, l_1, q_1' \rangle \cdots$:

(1) for each i, $\langle q_i, w_i, l_i, q_i' \rangle \in \delta$;

(2) $q_0 = q_I$, and for each $i > 0$, $q_{i-1}' = q_i$;

(3) for the maximum i, $q_i' \in F$;

(4) there are no c and $i < j$ such that $l_i = l_j = \langle \mathtt{inc}, c \rangle$ and $i \sim^{\sigma} j$;

(5) there are no c and $i < j$ such that $l_i = l_j = \langle \mathtt{dec}, c \rangle$ and $i \sim^{\sigma} j$;

(6) for any c and i such that $l_i = \langle \mathtt{inc}, c \rangle$, it is not the case that, there is $j > i$ with $l_j = \langle \mathtt{ifzero}, c \rangle$ but there is no $k > i$ with $l_k = \langle \mathtt{dec}, c \rangle$ and $i \sim^{\sigma} k$;

(7) there are no c and $i < j < k$ such that $l_i = \langle \mathtt{inc}, c \rangle$, $l_j = \langle \mathtt{ifzero}, c \rangle$, $l_k = \langle \mathtt{dec}, c \rangle$ and $i \sim^{\sigma} k$;

An $\mathrm{LTL}_1^{\downarrow}(\sim; \mathtt{X}, \mathtt{F})$ sentence over $\hat{\Sigma}$ which expresses the conjunction of (1)–(7) can be constructed from $\mathcal{C}$ in logarithmic space. (1)–(3) are straightforward. Among (4)–(7), the most interesting is (7), and the rest can be expressed similarly. Note how (6) and (7) were formulated to avoid using the U operator. The following sentence expresses (7):

$$\neg \bigvee_{c=1}^{n} \mathtt{F} \left(\bigvee_{q,w,q'} \langle q, w, \langle \mathtt{inc}, c \rangle, q' \rangle \right) \wedge$$
$$\downarrow_1 \mathtt{XF} \left(\bigvee_{q,w,q'} \langle q, w, \langle \mathtt{ifzero}, c \rangle, q' \rangle \right) \wedge$$
$$\mathtt{XF} \left(\bigvee_{q,w,q'} \langle q, w, \langle \mathtt{dec}, c \rangle, q' \rangle \right) \wedge \uparrow_1 \sim$$

A 1URA$_1(\sim)$ automaton which accepts exactly those $\sigma \in \hat{\Sigma}^{<\omega}(\sim)$ which satisfy (1)–(7) can also be constructed in logarithmic space. By Propositions 3 and 4, it suffices, for each of (1)–(7), to construct in logarithmic space a 1NRA$_1(\sim)$ automaton which accepts a data word iff it fails the condition. In fact, (6) and (7) can be treated together, and this automaton is the most interesting. It makes sure that σ has a run iff it contains an $\langle \mathtt{inc}, c \rangle$ instruction, followed by no occurence of $\langle \mathtt{dec}, c \rangle$ in the same class until $\langle \mathtt{ifzero}, c \rangle$ occurs. Formally, it has locations $\{0, 1, \ldots, n\}$ where 0 is initial, and the following transition function. (As the automaton is one-way, we omit the offsets of 1.)

$$\begin{aligned}
(0, \langle q, w, \langle \mathtt{inc}, c \rangle, q' \rangle) &\mapsto 0 \vee \downarrow_1 c \\
(0, \langle q, w, l, q' \rangle) &\mapsto 0, \text{ otherwise} \\
(c, \langle q, w, \langle \mathtt{dec}, c \rangle, q' \rangle) &\mapsto \uparrow_1 \not\sim \wedge c \\
(c, \langle q, w, \langle \mathtt{ifzero}, c \rangle, q' \rangle) &\mapsto \top \\
(c, \langle q, w, l, q' \rangle) &\mapsto c, \text{ otherwise}
\end{aligned}$$

The infinitary case is obtained similarly, where (3) is replaced by:

(3') for infinitely many i, $q_i \in F$, and for infinitely many i, $w_i \neq \varepsilon$. □

Corollary 16 *Over finite and over infinite words, the following sets of languages are the same:*

(i) of the form $\mathrm{L}(\mathcal{C})$*, where $\mathcal{C}$ is an Incrementing CA;*

(ii) of the form $f(\mathrm{str}(\mathrm{L}(\phi)))$*, where f is a string homomorphism and ϕ is an* $\mathrm{LTL}_1^{\downarrow}(\sim; \mathtt{X}, \mathtt{F})$ *sentence;*

(iii) of the form $f(\mathrm{str}(\mathrm{L}(\phi)))$*, where f is a string homomorphism and ϕ is an* $\mathrm{LTL}_1^{\downarrow}(\sim; \mathtt{X}, \mathtt{U})$ *sentence.*

Proof. By the proofs of Corollary 13 and Theorem 15. □

Our final result shows that it is impossible, without causing Σ_1^0 or Σ_1^1 hardness (see Section 2.4), to extend the problems in Theorem 12 in Corollary 13 by adding backward

transitions, the $\mathtt{F}^{-1}$ operator, or one more register, even if we restrict to nonemptiness of universal automata and replace $\mathtt{U}$ with $\mathtt{F}$. The result should also be compared with Theorem 10.

The theorem below is stronger than [12, Corollary 1] and [8, Theorem 3], which showed Σ^1_1-hardness of infinitary satisfiability for $\mathrm{LTL}^{\downarrow}_2(\sim; \mathtt{X}, \mathtt{X}^{-1}, \mathtt{F}, \mathtt{F}^{-1})$ and $\mathrm{LTL}^{\downarrow}_2(\sim; \mathtt{X}, \mathtt{U})$. Also, together with Proposition 3, it implies [21, Theorem 5.1] where finitary nonuniversality for $1\mathrm{NRA}(\sim)$ was shown undecidable. Undecidability of finitary nonemptiness for $2\mathrm{DRA}_1(\sim)$ was shown in [7, Section 7.3], using a different encoding.

Theorem 17 *In both finitary and infinitary cases, nonemptiness of Minsky CA is logspace reducible to satisfiability for $\mathrm{LTL}^{\downarrow}_1(\sim; \mathtt{X}, \mathtt{F}, \mathtt{F}^{-1})$ and $\mathrm{LTL}^{\downarrow}_2(\sim; \mathtt{X}, \mathtt{F})$, and nonemptiness for $1URA_2(\sim)$. In the finitary [resp. infinitary] case, we also have a logspace reduction to nonemptiness of $2DRA_1(\sim)$ [resp. $2URA_1(\sim)$].*

Proof. The reductions to problems with $\mathtt{F}^{-1}$ or backward transitions use encodings as in the proof of Theorem 15. The reductions to problems with 2 registers use encodings similar to that in [17, Section 4]. □

7. Concluding remarks

A summary of the results in this paper on the complexity of satisfiability for fragments of $\mathrm{LTL}^{\downarrow}(\sim; \mathtt{X}, \mathtt{X}^{-1}, \mathtt{U}, \mathtt{U}^{-1})$ can be found in boldface in the table below. The remaining entries were shown in [12, 8]. R \ PR means decidable and not primitive recursive, and 'co.' abbreviates 'complete'.

	$\mathrm{SAT}^{<\omega}$		SAT^{ω}	
registers	1	2	1	2
$\mathtt{X}, \mathtt{F}$	**R \ PR**	$\mathbf{\Sigma^0_1}$**-co.**	$\mathbf{\Pi^0_1}$**-co.**	$\mathbf{\Sigma^1_1}$**-co.**
$\mathtt{X}, \mathtt{U}$	**R \ PR**	Σ^0_1-co.	$\mathbf{\Pi^0_1}$**-co.**	Σ^1_1-co.
$\mathtt{X}, \mathtt{F}, \mathtt{F}^{-1}$	$\mathbf{\Sigma^0_1}$**-co.**	$\mathbf{\Sigma^0_1}$**-co.**	$\mathbf{\Sigma^1_1}$**-co.**	Σ^1_1-co.

We are grateful to Claire David, Massimo Franceschet, Marcin Jurdziński, Anca Muscholl, David Nowak, Joël Ouaknine, Philippe Schnoebelen and Luc Segoufin for helpful discussions.

References

[1] R. Alur and D. Dill. A theory of timed automata. *TCS*, 126:183–235, 1994.

[2] R. Alur and T. Henzinger. A really temporal logic. *JACM*, 41(1):181–204, 1994.

[3] M. Bojańczyk, C. David, A. Muscholl, T. Schwentick, and L. Segoufin. Two-variable on Data Trees and XML reasoning. In *PODS*, 2006.

[4] M. Bojańczyk, A. Muscholl, T. Schwentick, L. Segoufin, and C. David. Two-variable logic on words with data. In *LICS*. IEEE, 2006.

[5] P. Bouyer. A logical characterization of data languages. *IPL*, 84(2):75–85, 2002.

[6] P. Bouyer, A. Petit, and D. Thérien. An algebraic approach to data languages and timed languages. *I & C*, 182(2):137–162, 2003.

[7] C. David. Mots et données infinies. Master's thesis, LIAFA, 2004.

[8] S. Demri, R. Lazić, and D. Nowak. On the freeze quantifier in constraint LTL: decidability and complexity. Technical Report 05-3, LSV, 2005. Extended abstract appeared in Proc. of TIME'05.

[9] C. Dufourd, A. Finkel, and P. Schnoebelen. Reset nets between decidability and undecidability. In *ICALP*, volume 1443 of *LNCS*, pages 103–115. Springer, 1998.

[10] K. Etessami, M. Vardi, and T. Wilke. First-order logic with two variables and unary temporal logic. *I & C*, 179(2):279–295, 2002.

[11] M. Fitting. Modal logic between propositional and first-order. *JLC*, 12(6):1017–1026, 2002.

[12] T. French. Quantified propositional temporal logic with repeating states. In *TIME-ICTL*, pages 155–165. IEEE, 2003.

[13] V. Goranko. Hierarchies of modal and temporal logics with references pointers. *JoLLI*, 5:1–24, 1996.

[14] M. Kaminski and N. Francez. Finite-memory automata. *TCS*, 134(2):329–363, 1994.

[15] F. Laroussinie, N. Markey, and P. Schnoebelen. Temporal logic with forgettable past. In *LICS*, pages 383–392. IEEE, 2002.

[16] S. Lasota and I. Walukiewicz. Alternating timed automata. In *FOSSACS*, volume 3441 of *LNCS*, pages 250–265. Springer, 2005.

[17] A. Lisitsa and I. Potapov. Temporal logic with predicate λ-abstraction. In *TIME*, pages 147–155. IEEE, 2005.

[18] C. Löding and W. Thomas. Alternating automata and logics over infinite words. In *IFIP TCS*, volume 1878 of *LNCS*, pages 521–535. Springer, 2000.

[19] M. Minsky. *Computation, Finite and Infinite Machines*. Prentice Hall, 1967.

[20] S. Miyano and T. Hayashi. Alternating finite automata on ω-words. *TCS*, 32:321–330, 1984.

[21] F. Neven, T. Schwentick, and V. Vianu. Finite state machines for strings over infinite alphabets. *ACM TOCL*, 5(3):403–435, 2004.

[22] P. Odifreddi. *Classical Recursion Theory II*, volume 143. Elsevier, 1999.

[23] J. Ouaknine and J. Worrell. On the decidability of metric temporal logic. In *LICS*, pages 188–197, 2005.

[24] J. Ouaknine and J. Worrell. On metric temporal logic and faulty Turing machines. In *FOSSACS*, volume 3921 of *LNCS*, pages 217–230. Springer, 2006.

[25] J. Ouaknine and J. Worrell. Personal communication, March 2006.

[26] H. Sakamoto and D. Ikeda. Intractability of decision problems for finite-memory automata. *TCS*, 231(2):297–308, 2000.

[27] P. Schnoebelen. Verifying lossy channel systems has nonprimitive recursive complexity. *IPL*, 83(5):251–261, 2002.