Safety Alternating Automata on Data Words

RANKO LAZIĆ
Department of Computer Science, University of Warwick, UK

A data word is a sequence of pairs of a letter from a finite alphabet and an element from an infinite set, where the latter can only be compared for equality. We consider safety one-way alternating automata with one register on infinite data words, and show that nonemptiness is ExpSpace-complete and inclusion is decidable but not primitive recursive. Generalising to the weak acceptance mechanism, two-way automata, or two registers, each cause undecidability.

Categories and Subject Descriptors: F.4.1 [Mathematical Logic and Formal Languages]: Formal Languages—Decision problems; F.1.1 [Computation by Abstract Devices]: Models of Computation—Automata

General Terms: Algorithms, Verification

1. INTRODUCTION

Context. Logics and automata for words and trees over finite alphabets are relatively well-understood. Motivated partly by the need for formal verification and synthesis of infinite-state systems, and the search for automated reasoning techniques for XML, there is an active and broad research programme on logics and automata for words and trees which have richer structure.

Segoufin's recent survey [Segoufin 2006] summarises the substantial progress made on reasoning about data words and data trees. A data word is a word over a finite alphabet, with an equivalence relation on word positions. Implicitly, every word position is labelled by an element (“datum”) from an infinite set (“data domain”), but since the infinite set is equipped only with the equality predicate, it suffices to know which word positions are labelled by equal data, and that is what the equivalence relation represents. Similarly, a data tree is a tree (countable, unranked and ordered) whose every node is labelled by a letter from a finite alphabet, with an equivalence relation on the set of its nodes.

It has been nontrivial to find satisfactory specification formalisms even for data words. First-order logic was considered in [Bojańczyk et al. 2006; David 2004], and related automata were studied further in [Björklund and Schwentick 2007]. The logic has variables which range over word positions (\{0, \ldots, l-1\} or \mathbb{N}), a unary predicate for each letter from the finite alphabet, and a binary predicate \( x \sim y \)
for the equivalence relation that represents equality of data labels. \( \text{FO}^2(\sim, <, +1) \) denotes such a logic with two variables and binary predicates \( x + 1 = y \) and \( x < y \). Over finite and over infinite data words, satisfiability for \( \text{FO}^2(\sim, <, +1) \) was proved decidable and at least as hard as reachability for Petri nets. The latter problem is \( \text{ExpSpace}\)-hard [Lipton 1976], but whether it is elementary has been open for many years. Elementary complexity of satisfiability can be obtained at the price of substantially reducing the navigational power: over finite data words, \( \text{NExpSpace} \)-completeness was established for \( \text{FO}^2(\sim, <) \) and \( 2\text{NExpSpace} \)-membership for \( \text{FO}^2(\sim, +1) \). In the other direction, if \( \text{FO}^2(\sim, <, +1) \) is extended by one more variable, \( +1 \) becomes expressible using \( < \), but satisfiability was shown undecidable.

An alternative approach to reasoning about data words is based on automata with registers [Kaminski and Francez 1994]. A register is used for storing a datum for later equality comparisons (i.e. an equivalence class for later membership testing). Nonemptiness of one-way nondeterministic register automata over finite data words has relatively low complexity: \( \text{NP} \)-complete [Sakamoto and Ikeda 2000] or \( \text{PSpace} \)-complete [Demri and Lazić 2008], depending on technical details of their definition. Unfortunately, such automata fail to provide a satisfactory notion of regular language of finite data words, as they are not closed under complement [Kaminski and Francez 1994] and their nonuniversality is undecidable [Neven et al. 2004]. To overcome those limitations, one-way alternating automata with 1 register (for short, \( 1\text{ARA} \)) were proposed in [Demri and Lazić 2008]: they are closed under Boolean operations, their nonemptiness over finite data words is decidable, and future-time fragments of temporal logics such as LTL or the modal \( \mu \)-calculus extended by 1 register are easily translatable to such automata. However, nonemptiness for \( 1\text{ARA} \) turned out to be not primitive recursive over finite data words, and undecidable (more precisely, \( \Pi_1^{0} \)-hard) over infinite ones with the weak acceptance mechanism [Muller et al. 1986] and thus also with Büchi or co-Büchi acceptance.

**Contribution.** We consider one-way alternating automata with 1 register with the safety acceptance mechanism over infinite data words (i.e. data \( \omega \)-words). The languages of such automata are safety properties [Alpern and Schneider 1987]: every rejected data \( \omega \)-word has a finite prefix such that every other data \( \omega \)-word which extends it is also rejected. (Over finite data words, safety is not a restriction.)

The main result in the paper is that nonemptiness of safety \( 1\text{ARA} \) is \( \text{ExpSpace} \)-complete. The upper bound is surprising since even decidability is fragile: from the proof of [Demri and Lazić 2008, Theorem 5.4], nonemptiness is \( \Pi_1^{0} \)-hard for automata which are two-way or have 2 registers. Moreover, nonemptiness of safety forward (i.e. downward and rightward) alternating automata with 1 register on data trees was recently shown decidable but not elementary [Jurdziński and Lazić 2007]. Decidability [Ouaknine and Worrell 2006] and nonelementarity [Bouyer et al. 2008] also hold for the safety fragment of metric temporal logic on timed \( \omega \)-words.

The proof of \( \text{ExpSpace} \)-membership is in two stages. The first consists of translating a given safety \( 1\text{ARA} \), \( A \) to a nondeterministic automaton with faulty counters \( C_A \) which is on \( \omega \)-words over the alphabet of \( A \) and which is nonempty iff \( A \) is. The counters of \( C_A \) are faulty in the sense that they are subject to incrementing errors, i.e. they can spontaneously increase at any time. Although a nonemptiness-preserving translation from weak \( 1\text{ARA} \) to counter automata with incrementing
errors was given in [Demri and Lazić 2008], applying it to safety 1ARA produces automata with the Büchi acceptance mechanism, where the latter ensures that certain loops cannot repeat infinitely due to incrementing errors. To obtain safety automata, we enrich the instruction set by nondeterministic transfers. When applied to a counter $c$ and a set of counters $C$, such an instruction transfers the value of $c$ to the counters in $C$, nondeterministically splitting it. Thus we obtain $C_A$ whose nonemptiness amounts to existence of an infinite computation from the initial state. However, a further observation on the resulting automata is required: the counters of such an automaton are nonempty subsets of a certain set (essentially, the set of locations of the given safety 1ARA), and it suffices to use nondeterministic transfers which are simultaneous for all counters and which have a certain distributivity property in terms of the partial-order structure of the set of all counters.

The second stage of the proof is then an inductive counting argument which shows that $C_A$ is nonempty iff it has a computation from the initial state of length doubly exponential in the size of $A$. Some of the techniques are also used in the proof that termination of channel machines with occurrence testing and insertion errors is primitive recursive [Bouyer et al. 2008]. Although counters are simpler resources than channels, the class of machines considered do not have instructions which correspond to the nondeterministic transfers, and the sets of channels and messages (which are counterparts to the sets of counters) have no special structure.

We also show decidability of language inclusion between two safety 1ARA. Since safety 1ARA are trivially closed under intersections and unions, it follows that nonemptiness is decidable for Boolean combinations of safety 1ARA. The latter is thus a competing formalism to FO$^2(\sim, <, +1)$ over data $\omega$-words. They are incomparable in expressiveness: there exist properties involving the past (e.g. 'every $b$ is preceded by an $a$ with the same datum') which are expressible in FO$^2(\sim, <, +1)$ but not by a Boolean combination of safety 1ARA, and the reverse is true of some constraints involving more than 2 word positions at a time (e.g. ‘whenever $a$ is followed by $b$ with the same datum, $c$ does not occur in between’). However, already nonuniversality of safety 1ARA is not primitive recursive, whereas it is not known whether satisfiability for FO$^2(\sim, <, +1)$ is elementary.

Finally, we remark that safety fragments of future-time temporal logics such as LTL or the modal $\mu$-calculus extended by 1 register are easily translatable to safety 1ARA (cf. [Demri and Lazić 2008; Jurdziński and Lazić 2007]), so satisfiability of Boolean combinations of their sentences is decidable.

2. PRELIMINARIES

2.1 Data $\omega$-Words

A *data $\omega$-word* $\sigma$ over a finite alphabet $\Sigma$ is an $\omega$-word $\text{str}(\sigma)$ over $\Sigma$ together with an equivalence relation $\sim^\sigma$ on $\mathbb{N}$. For $i \in \mathbb{N}$, we write $\sigma(i)$ for the letter at position $i$, and $\lfloor i \rfloor_{\sim^\sigma}$ for the class that contains $i$. When $\sigma$ is understood, we may write simply $\sim$ instead of $\sim^\sigma$. We shall sometimes refer to classes of $\sim$ as ‘data’.

2.2 Safety Games

The automata that will be introduced in the next section will be alternating and able to recognise safety properties, so we shall use the following class of zero-sum
two-player finitely branching games to define acceptance by such automata. A safety game $G$ is a tuple \( \langle P, P_1, P_2, \rightarrow \rangle \) such that:

- \( P \) is a set of all positions;
- \( P_1 \) and \( P_2 \) disjointly partition \( P \) into positions of players 1 and 2 (respectively);
- \( \rightarrow \subseteq P \times P \) is a successor relation with respect to which every position has finitely many successors.

A play \( \pi \) of \( G \) is a sequence \( p_0 p_1 \ldots \) of positions of \( G \) such that \( p_i \rightarrow p_{i+1} \) for each \( i \). We say that \( \pi \) is complete iff either it ends with a position without successors or it is infinite. For such \( \pi \), we consider it winning for player 1 iff either it ends with a position of player 2 or it is infinite. Otherwise, i.e. iff it ends with a position of player 1, \( \pi \) is winning for player 2.

A strategy for player \( l \) from a position \( p \) of \( G \) is a tree \( \tau \subseteq P^<\omega \) of finite plays of \( G \) such that:

(i) \( p \in \tau \) and it is the root;
(ii) whenever \( \pi \in \tau \) ends with a position \( p \) of player \( l \) which has at least one successor, it has a unique child;
(iii) whenever \( \pi \in \tau \) ends with a position \( p \) of the other player, it has all children \( \pi p' \) with \( p \rightarrow p' \).

We say that \( \tau \) is positional iff the choices of successors in (ii) depend only on the ending positions \( p \).

Now, a play by \( \tau \) is either an element of \( \tau \) or an infinite sequence whose every nonempty prefix is an element of \( \tau \). We say that \( \tau \) is winning iff each complete play by \( \tau \) is winning for player \( l \).

The following well-known result is a corollary of e.g. [Demri and Lažič 2008, Theorem 2.4].

**Lemma 2.1.** Every safety game is positionally determined, i.e. from every position, one of the players has a positional winning strategy.

### 2.3 Register Automata

We now define one-way alternating automata with a register for recognising safety languages of data \( \omega \)-words. The details are based on the more general definition of weak two-way alternating register automata in [Demri and Lažič 2008].

A state of such an automaton for a data \( \omega \)-word will consist of a word position, an automaton location and a register value. From it, according to the transition function, one of the following is performed:

- branching to another location depending on whether the current letter equals a specified letter, or the current datum equals the register value;
- storing the current datum into the register;
- conjunctive or disjunctive branching to a pair of locations;
- acceptance or rejection;
- moving to the next word position.
We shall consider only automata which cannot perform infinitely many transitions while remaining at the same word position. That constraint simplifies some proofs without reducing expressiveness.

Formally, the set $\Delta(\Sigma, Q)$ of all transition formulae over a finite alphabet $\Sigma$ and a finite set $Q$ of locations is defined as

$$\{ q \overset{a}{\rightarrow} q', q \overset{\uparrow}{\rightarrow} q', q \wedge q', q \vee q', \top, \bot, Xq : a \in \Sigma, q, q' \in Q \}$$

A safety one-way alternating automaton with 1 register (shortly, safety 1ARA$_1$) $A$ is a tuple $\langle \Sigma, Q, q_I, \delta \rangle$ as follows:

- $\Sigma$ is a finite alphabet;
- $Q$ is a finite set of locations, and $q_I \in Q$ is the initial location;
- $\delta : Q \rightarrow \Delta(\Sigma, Q)$ is a transition function such that, whenever $q_0, \ldots, q_{k-1} \in Q$ and $q_{i+1} \mod k$ occurs in $\delta(q_i)$ for each $i$, we have $\delta(q_i) = Xq_{i+1} \mod k$ for some $i$.

For a data $\omega$-word $\sigma$ over $\Sigma$, the acceptance game $G_{A, \sigma} = \langle P, P_1, P_2, \rightarrow \rangle$ is the safety game defined below. Player 1 ("automaton") will be resolving the disjunctive branchings given by the transition function of $A$, winning a finite play if it ends with an accepting state, and winning every infinite play. Dually, player 2 ("pathfinder") will be resolving the conjunctive branchings and winning at rejecting states.

- $P$ is the set of all states of $A$ for $\sigma$: triples $\langle i, q, D \rangle$ where $i \in \mathbb{N}$, $q \in Q$, and $D$ is either $\emptyset$ (denoting that the register is undefined) or a class of $\sim$.  
- The partition of $P$ into $P_1$ and $P_2$, and the successor relation, are given by the table in Figure 1. The ownership of states with unique successors has not been specified because it is irrelevant. The table omits dual transition formulae, which are treated by swapping the ownerships.

A run of $A$ over $\sigma$ is a strategy $\tau$ in $G_{A, \sigma}$ for player 1 from the initial state $\langle 0, q_I, \emptyset \rangle$. We say that $\tau$ is accepting iff it is winning, and that $A$ accepts $\sigma$ iff $A$ has an accepting run over $\sigma$.

Since player 2 loses infinite plays, it only has winning strategies which are finite trees. Recalling Lemma 2.1, it follows that the language of $A$ is safety.

**Example 2.2.** A safety 1ARA$_1$ whose alphabet contains letters $a$ and $b$ is depicted in Figure 2. The automaton rejects a data $\omega$-word iff there is an occurrence of $a$ and a subsequent occurrence of $b$ with the same datum such that $a$ with the same datum does not occur between them.
2.4 Counter Automata

We introduce below a class of nondeterministic automata on $\omega$-words which have $\varepsilon$ transitions and $\mathbb{N}$-valued counters. The set of counters of such an automaton will have structure: there will be a finite set called the basis of the automaton, and each counter will be a nonempty subset of the basis. In the course of a transition, the automaton will be able either to increment a counter, or to decrement a counter if nonzero, or to perform a simultaneous nondeterministic transfer with respect to a mapping $f$ from counters to sets of counters. The latter transfers the value of each counter $c$ to the counters in $f(c)$, nondeterministically splitting it. However, only mappings which satisfy a distributivity constraint in terms of the structure of the set of counters may be used.

We shall only consider automata with no cycles of $\varepsilon$ transitions, and they will recognise safety languages, so every infinite run will accept some $\omega$-word.

The automata will be faulty in the sense that their counters may erroneously increase at any time.

Formally, for a finite set $X$ and $C \subseteq \mathcal{P}(X) \setminus \{\emptyset\}$, let $L(C)$ be the set of all instructions:

- $\langle \text{inc}, c \rangle$ and $\langle \text{dec}, c \rangle$ for $c \in C$;
- $\langle \text{transf}, f \rangle$ for mappings $f : C \to \mathcal{P}(C)$ which are distributive as follows: whenever $c \in C$, $c \subseteq \bigcup_{i=1}^{k} c_i$, and $c'_i \in f(c_i)$ for each $i = 1, \ldots, k$, there exists $c' \in f(c)$ such that $c' \subseteq \bigcup_{i=1}^{k} c'_i$.

A safety powerset counter automaton with nondeterministic transfers and incrementing errors (shortly, safety IPCANT) $\mathcal{C}$ is a tuple $\langle \Sigma, Q, q_I, X, C, \delta \rangle$ such that:

- $\Sigma$ is a finite alphabet;
- $Q$ is a finite set of locations, and $q_I$ is the initial location;
- $X$ is a finite set called the basis, and $C \subseteq \mathcal{P}(X) \setminus \{\emptyset\}$ is the set of counters;
- $\delta \subseteq Q \times (\Sigma \cup \{\varepsilon\}) \times L(C) \times Q$ is a transition relation which does not contain a cycle of $\varepsilon$ transitions.

A state of $\mathcal{C}$ is a pair $\langle q, v \rangle$, where $q \in Q$ and $v$ is a counter valuation, i.e. $v : C \to \mathbb{N}$. We say that $\langle q, v \rangle$ has an error-free transition labelled by $w \in \Sigma \cup \{\varepsilon\}$ and performing $l \in L(C)$ to $\langle q', v' \rangle$, and we write $\langle q, v \rangle \xrightarrow{w,l} \langle q', v' \rangle$, iff $\langle q, w, l, q' \rangle \in \delta$ and $v'$ can be obtained from $v$ by $l$. The latter is defined as follows:

- instructions $\langle \text{inc}, c \rangle$ and $\langle \text{dec}, c \rangle$ have the standard interpretations, where $\langle \text{dec}, c \rangle$ is firable iff $v(c) > 0$;
accepts exactly the string projections of data $\omega$

This section contains a two-stage proof that nonemptiness of safety 1ARA and ExpSpace is polynomially (in fact, linearly) bounded. Nonemptiness is preserved, since Kőnig’s Lemma that the language of every safety IPCANT is safety.

For counter valuations $v$ and $v'$, we write $v \leq v'$ iff, for all $c$, $v(c) \leq v'(c)$. To allow transitions of $C$ to contain incrementing errors, we define $\langle q, v \rangle \overset{w, l}{\rightarrow} \langle q', v' \rangle$ to mean that there exist $v'$ and $v'$ with $v \leq v'$, $\langle q, v, v' \rangle \overset{w, l}{\rightarrow} \langle q', v', v' \rangle$ and $v' \leq v'$.

We say that $C$ accepts an $\omega$-word $w$ over $\Sigma$ iff $C$ has a run $\langle q_0, v_0 \rangle \overset{w_0, l_0}{\rightarrow} \langle q_1, v_1 \rangle \overset{w_1, l_1}{\rightarrow} \cdots$ where $(q_0, v_0)$ is the initial state $(q, 0)$ and $w = w_0w_1 \ldots$.

Example 2.3. Given $Y \subseteq X$, let $f_Y(c) = \emptyset$ if $c \cap Y \neq \emptyset$, and $f_Y(c) = \{c\}$ otherwise. Observe that $f_Y$ is distributive. The instruction $\langle\text{transf}, f_Y\rangle$ is firable iff each counter which intersects $Y$ is zero, and it does not change the value of any counter. Hence, we may write $\langle\text{transf}, Y\rangle$ instead of $\langle\text{transf}, f_Y\rangle$.

Suppose $C = \{\{x\} : x \in X\}$, i.e. the set of counters has no structure. The instruction $\langle\text{transf}, Y\rangle$ is firable iff each counter $\{x\}$ for $x \in Y$ is zero. Observe that every $f : C \rightarrow \mathcal{P}(C)$ is distributive. For instance, given $c \in C$ and nonempty $C' \subseteq C$, let $f_{c, C'}(c) = C'$ and $f_{c, C'}(c') = \{c'\}$ for $c' \neq c$. The instruction $\langle\text{transf}, f_{c, C'}\rangle$ nondeterministically distributes the value of $c$ to the counters in $C'$.

For $C$ as above, let us say that a transition $\langle q, v \rangle \overset{w, l}{\rightarrow} \langle q', v' \rangle$ is lazy iff either $\langle q, v \rangle \overset{w, l}{\rightarrow} \langle q', v' \rangle$, or $l$ is of the form $\langle\text{dec}, c\rangle$, $v(c) = 0$ and $v' = v$. Thus, in lazy transitions, only incrementing errors which enable decrements of counters with value 0 may occur. The following straightforward lemma shows that restricting to lazy transitions does not affect the languages of safety IPCANTS. Since from every state only finitely many lazy transitions are possible, it also shows together with König’s Lemma that the language of every safety IPCANT is safety.

Lemma 2.4. Whenever $\langle q, v \rangle \overset{w, l}{\rightarrow} \langle q', v' \rangle$ is a transition of a safety IPCANT $C$ and $v_1 \leq v$, there exists a lazy transition $\langle q, v_1 \rangle \overset{w, l}{\rightarrow} \langle q', v'_1 \rangle$ of $C$ such that $v'_1 \leq v'$.

3. UPPER BOUND FOR NONEMPTINESS

This section contains a two-stage proof that nonemptiness of safety 1ARA$_1$ is in ExpSpace. The first theorem below shows that each such automaton $A$ is translatable to a safety IPCANT $C_A$ of at most exponential size, but whose basis size is polynomially (in fact, linearly) bounded. Nonemptiness is preserved, since $C_A$ accepts exactly the string projections of data $\omega$-words in the language of $A$. By the second theorem, nonemptiness of $C_A$ is decidable in space exponential in its basis size and logarithmic in its alphabet size and number of locations, so space exponential in the size of $A$ suffices overall.

Theorem 3.1. Given a safety 1ARA$_1$ $A$, a safety IPCANT $C_A$ is computable in polynomial space, such that $C_A$ and $A$ have the same alphabet, the basis size of $C_A$ is linear in the number of locations of $A$, and $L(C_A) = \{\str(\sigma) : \sigma \in L(A)\}$.

Proof. The proof is an adaptation of the proof of [Demri and Lazić 2008, Theorem 4.4], where it was shown by the following steps how to translate in polynomial
space weak 1ARA₁ to Büchi nondeterministic counter automata with ε transitions and incrementing errors, and whose instructions are increments, decrements and zero tests of individual counters:

(i) replace two-player acceptance games for weak 1ARA₁ by one-player games whose positions are built from sets of states, and whose successors are “big step” in the sense that they correspond to following strategies for the automaton until first moves to the next word position;

(ii) combine the one-player acceptance games with searching for a data word to be accepted, resulting in one-player nonemptiness games;

(iii) show how to construct counter automata which guess and check winning plays in the nonemptiness games;

(iv) show that allowing incrementing errors in computations of the counter automata does not increase their languages.

Let \( A = (\Sigma, Q, q_0, \delta) \). We show below that, since \( A \) is safety, the construction in step (ii) can be modified using nondeterministic transfers with a suitable basis and set of counters, so that zero tests of individual counters, cycles of ε transitions and the Büchi acceptance condition are eliminated, resulting in a safety IPCANT.

We first define a one-player nonemptiness game for \( A \), whose set of positions \( H_A \) consists of \( \emptyset \) and all “abstract sets” of the form \( \langle a, Q_\emptyset, Q, \sharp \rangle \) where \( a \in \Sigma, Q_\emptyset \subseteq Q, \sharp : \mathcal{P}(Q) \setminus \{\emptyset\} \to \mathbb{N} \), and either \( Q_\emptyset \neq \emptyset \) or \( Q_\emptyset \neq \emptyset \) or \( \sharp(Q_1) > 0 \) for some \( Q_1 \). As in the proof of [Demri and Lazić 2008, Theorem 4.4], each such \( \langle a, Q_\emptyset, Q, \sharp \rangle \) represents any nonempty set \( P \) of states of \( A \) for a data \( \omega \)-word \( \sigma \) over \( \Sigma \) such that:

---

some \( i \in \mathbb{N} \) is the first component of every state in \( P \);

\(-a = \sigma(i), Q_\emptyset = \{ q : (i, q, [i]_\sim) \in P \} \) and \( Q_\emptyset = \{ q : (i, q, \emptyset) \in P \} \);

\(-\sharp(Q_1) = |\{ D \neq [i]_\sim : \{ q : (i, q, D) \in P \} = Q_1 \}| \) for all nonempty \( Q_1 \subseteq Q \).

In particular, \( \sharp(Q_1) \) is the number of distinct data \( D \) which are not the class of \( i \) and for which the set of all \( q \) with \( (i, q, D) \in P \) equals \( Q_1 \).

To specify a big-step successor relation between abstract sets, for \( a \in \Sigma, uu \in \{ \top, \bot \} \) and \( q \in Q \), let \( \langle a, uu, q \rangle \) be the set of pairs of sets of locations that is defined in Figure 3, where well-definedness relies on the absence of cycles of non-\( X \) transitions. Given a map \( f : X \to \mathcal{P}(Y_1) \times \mathcal{P}(Y_2) \), let

\[
\bigcup \ f = \bigcup \{ Z_1 : (Z_1, Z_2) \in f(X) \} \quad \bigcup \ f = \bigcup \{ Z_2 : (Z_1, Z_2) \in f(X) \}
\]

For \( h, h' \in H_A \), we write \( h \Rightarrow h' \) iff \( h \) is of the form \( \langle a, Q_\emptyset, Q, \sharp \rangle \) and there exist maps \( q \in Q_\emptyset \mapsto f_\emptyset(q) \in \langle a, \top, q \rangle \), \( q \in Q_\emptyset \mapsto f_0(q) \in \langle a, \bot, q \rangle \) and \( q \in Q_1 \mapsto f_{Q_1,j}(q) \in \langle a, \bot, q \rangle \) for each nonempty \( Q_1 \subseteq Q \) and \( j \in \{1, \ldots, \sharp(Q_1)\} \) such that:

---

either \( h' = \emptyset, \bigcup \ f_\emptyset = \emptyset, \) and \( \sharp'(Q'_1) = 0 \) for all \( Q'_1 \),

or \( h' \) is of the form \( \langle a', 0, \bigcup \ f_\emptyset, \sharp' \rangle \),

or \( h' \) is of the form \( \langle a', Q'_\emptyset, \bigcup \ f_\emptyset, \sharp'[Q'_\emptyset \mapsto \sharp'(Q'_\emptyset) - 1] \rangle \),

ACM Journal Name, Vol. V, No. N, Month 20YY.
where, for each nonempty $Q'_i \subseteq Q$, $\sharp'(Q'_i)$ is defined as

$$\sharp'(Q'_i) = \|\{\langle q, q' \rangle : \cup f_{Q_{1}, j} = Q'_i\}| + \begin{cases} 1, & \text{if } \cup f_2 = \cup f_{\emptyset} = \cup f_{Q_{1}, j} = Q'_i \\ 0, & \text{otherwise} \end{cases}$$

The one-player game thus defined has the following key property:

(*) $\mathcal{A}$ accepts some data $\omega$-word $\sigma$ with $\text{str}(\sigma) = a_0 a_1 \ldots$ if there exists a sequence $h_0 \Rightarrow h_1 \Rightarrow \cdots$ of elements of $H_{\mathcal{A}}$ such that $h_0 = \langle a_0, \emptyset, \{q_f\} \rangle, Q_{f} \mapsto 0$, the sequence either ends by $\emptyset$ or is infinite, and for each $i > 0$ with $h_i \neq \emptyset$, the first component of $h_i$ is $a_i$.

Let $Q = \{q : q \in Q\}$ and $\overline{Q} = \{q : q \in Q\}$. The alphabet of $\mathcal{C}_{\mathcal{A}}$ is $\Sigma$, and its basis is $\{\#\} \cup Q \cup \{\overline{\#}\} \cup Q \cup \overline{Q}$. The locations and transition relation of $\mathcal{C}_{\mathcal{A}}$ are constructed so that $\mathcal{A}$ guesses and checks a sequence $h_0 \Rightarrow h_1 \Rightarrow \cdots$ as in (*), storing at most two consecutive members in any state. To store an abstract set $\langle a, Q_{=}, Q_{\emptyset}, \sharp \rangle$, locations of $\mathcal{C}_{\mathcal{A}}$ are used for the first three components, and $\sharp$ is stored by means of $2^{\sharp}|Q| - 1$ counters $\{\#\} \cup Q_1$ for $\emptyset \neq Q_1 \subseteq Q$. $\mathcal{C}_{\mathcal{A}}$ also has $4^{\sharp}|Q| + 1$ auxiliary counters: $\{\#\}$ and $\{\overline{\#}\}$ for $Q_{\#}, Q_{\overline{\#}} \subseteq Q$. The nontrivial part of $\mathcal{C}_{\mathcal{A}}$ is, given $a, Q_{=}, Q_{\emptyset}$, and $\sharp$, which is stored by the counters $\{\#\} \cup Q_1$, to guess maps $f_{=}, f_{\emptyset}$ and $f_{Q_{1}, j}$, and set the counters $\{\#\} \cup Q_1$ so that they store $\sharp'$ as defined above. Figure 4 contains pseudo-code for that computation, which is by $\varepsilon$ transitions. It assumes that each counter $\{\overline{\#}\} \cup Q_{\#} \cup Q_{\overline{\#}}$ is zero at the beginning, and ensures that the same holds at the end. The choices of maps are nondeterministic. If a map cannot be chosen because a corresponding set $\langle a, uu, q \rangle$ is empty, the computation blocks. Using notation from Example 2.3, the conditional is implemented by a nondeterministic choice among $\text{ifz}^\wedge(\langle \# \rangle)$ and $\text{dec}(\langle \overline{\#}\rangle \cup Q_{\#} \cup Q_{\overline{\#}}); \text{inc}(\langle \overline{\#}\rangle \cup Q_{\#} \cup Q_{\overline{\#}}); Q_1 := Q_1 \cup \{q\}$ for all $Q_{\#}$ and $Q_{\overline{\#}}$ with $q \in Q_{\overline{\#}}$.

By essentially the same argument as in the proof of [Demri and Lazić 2008, Theorem 4.4], incrementing errors do not increase the language of $\mathcal{C}_{\mathcal{A}}$: such errors in an infinite computation amount to introducing superfluous states of $\mathcal{A}$ from which winning strategies for the automaton are then found.

To compute $\mathcal{C}_{\mathcal{A}}$ in polynomial space, the pseudo-code in Figure 4 is implemented so that any state stores (by means of its location) at most one component of the maps $q \in Q_1 \mapsto f(q) \in \langle a, \bot, q \rangle$, $q \in Q_{=} \mapsto f_{=} (q) \in \langle a, \top, q \rangle$, and $q \in Q_{\emptyset} \mapsto f_{\emptyset} (q) \in \langle a, \bot, q \rangle$. The definition in Figure 3 provides a nondeterministic algorithm.
which, given \( a \in \Sigma \), \( uu \in \{T, \perp\} \), \( q \in Q \) and \( Q_{q'}^1, Q_{q'}^2 \subseteq Q \), checks whether \( \langle Q_{q'}^1, Q_{q'}^2 \rangle \in \langle a, uu, q \rangle \) in space polynomial in the size of \( \mathcal{A} \).

**Theorem 3.2.** Nonemptiness of safety IPCANT is decidable in space exponential in basis size and logarithmic in alphabet size and number of locations.

**Proof.** Suppose \( \mathcal{C} = \langle \Sigma, Q, q_1, X, C, \delta \rangle \) is a safety IPCANT. By Lemma 2.4, \( \mathcal{C} \) is nonempty iff it has an infinite sequence of lazy transitions from the initial state.

We define positive integers \( \alpha_i \) and \( U_i \) for \( i = 0, \ldots, |X| \) as follows:

\[
\begin{align*}
\alpha_0 &= |Q| \\
U_0 &= 1 \\
\alpha_{i+1} &= 2(|X| - i)\alpha_i U_i^{[C]} \\
U_{i+1} &= 3\alpha_i U_i^{[C]}
\end{align*}
\]

Let \( m = 2\alpha_{|X|} U_i^{[C]} \). We shall show that, if \( \mathcal{C} \) has a sequence of lazy transitions of length \( m \) from the initial state, then it has an infinite sequence. In such a sequence \( S \), each counter is at most \( m - 1 \). For guessing \( S \), it suffices to store only one transition at a time. Since \( m < 2^{2|X|^2 + |X| \log(3|Q|)} \), it follows by Savitch’s Theorem that nonemptiness of \( \mathcal{C} \) is decidable in space \( 2^{O(|X|^2 \log(|\Sigma||Q|))} \).

Suppose \( \mathcal{C} \) has a sequence of lazy transitions \( S = \langle q_1, v_1 \rangle \xrightarrow{w_1,j_1} \cdots \xrightarrow{w_{m-1},j_{m-1}} \langle q_m, v_m \rangle \) from the initial state, but no infinite sequence. Let \( q \in Q \) and \( J_0 \subseteq \{1, \ldots, m\} \) be such that \( |J_0| = m/\alpha_0 U_i^{[C]} \) and \( q_j = q \) for each \( j \in J_0 \). We claim:

There exists an enumeration \( x_1, \ldots, x_{|X|} \) of \( X \), and for \( i = 1, \ldots, |X| \), mappings \( u_i : C_i \to \{0, \ldots, U_i - 1\} \) where \( C_i = \{c \in C : x_i \in c \land x_1, \ldots, x_{i-1} \notin c\} \), and subsets \( J_i \) of \( \{1, \ldots, m\} \) of size \( m/\alpha_i U_i^{[C]} \), such that for each \( 0 \leq i \leq |X| \) and \( j \in J_i \), we have \( q_j = q \) and \( v_j(c) = u_i(c) \) for all \( c \in C_i \) and \( 1 \leq i' \leq i \).

The claim holds for \( i = 0 \). Assume that \( 0 \leq i < |X| \) and that \( x_{i'}, u_{i'} \) and \( J_{i'} \) for \( 1 \leq i' \leq i \) have been picked so that the claim holds for \( i \). Let us call a subsequence of \( S \) an \textit{i-subsequence} if there exist consecutive \( j, j' \in J_i \) (i.e. where there is no \( j'' \in J_i \) with \( j < j'' < j' \)) such that the subsequence begins at \( \langle q_j, v_j \rangle \) and ends at \( \langle q_{j'}, v_{j'} \rangle \). Consider the \( m/2\alpha_i U_i^{[C]} \) shortest \( i \)-subsequences, and let \( J_i' \subseteq J_i \) consist of their beginning positions. The length of the longest of these \( i \)-subsequences must be at most \( 2\alpha_i U_i^{[C]} \). Let \( S' = \langle q_j, v_j \rangle \xrightarrow{w_j,j_j} \cdots \xrightarrow{w_{j'-1},j_{j'-1}} \langle q_{j'}, v_{j'} \rangle \) be an \( i \)-subsequence with \( j \in J_i' \). We have \( q_j = q_{j'} = q \), \( v_j(c) = v_{j'}(c) = u_i(c) \) for all \( c \in C_i \) and \( 1 \leq i' \leq i \), and \( j' - j \leq 2\alpha_i U_i^{[C]} \). In particular, \( \sum_{i=1}^{j'} \sum_{c \in C_i} v_j(c) \leq \sum_{i=1}^{j'} |C_i| U_i \).

Assume that, for each \( x' \neq x_1, \ldots, x_i \), there exists \( c_{x'} \) such that \( x' \in c_{x'} \), \( x_1, \ldots, x_i \notin c_{x'} \), and \( v_j(c_{x'}) > 2\alpha_i U_i^{[C]} + \sum_{i' = 1}^{j'} |C_{i'}| U_{i'} \). Let \( H \) be a directed...
acyclic graph on \( \{j, \ldots, j'\} \times C \), defined by letting the successors of \( \langle j^i, c^i \rangle \) be:

- \( \emptyset \), if \( j^i = j' \);
- \( \{j^i + 1, c^i\} : c^i \in f(c^i) \}, \) if \( j^i \) is of the form \( \text{transf}, f \);
- \( \{j^i + 1, c^i\} \), otherwise.

Now, for \( c \in C \) and \( j^i \in \{j, \ldots, j'\} \), let \( H(c, j^i) \) be the set of all \( c^i \) such that \( \langle j^i, c^i \rangle \) is reachable in \( H \) from \( \langle j, c \rangle \). We have \( \sum_{c^i \in H(c, j^i)} v_j(c^i) \geq v_j(c) - (j^i - j) \) by induction on \( j^i \). In particular, for each \( x^i \neq x_1, \ldots, x_i \), we have \( \sum_{c^i \in H(c, j^i)} v_j(c^i) \geq v_j(c^i) - (j^i - j) > \sum_{c^i = \emptyset} v_j(c^i) \), so \( H \) contains a path \( H_{c^i} \) from \( \langle j, c^i \rangle \) to some \( \langle j', c^i \rangle \) with \( x_1, \ldots, x_i \notin c^i \).

Consider any \( c \) with \( x_1, \ldots, x_i \notin c \). We have \( c \subseteq \bigcup \{c^i : x^i \in c \} \). By distributivity of nondeterministic transfer mappings and the definition of \( H \), there exists a path \( H_c \) from \( \langle j, c \rangle \) to some \( \langle j', c' \rangle \) with \( x_1, \ldots, x_i \notin c' \). For \( j^i \in \{j, \ldots, j'\} \), let \( H_c(j^i) \) be the counter at position \( j^i \) in \( H_c \).

Given any \( v'_j \) such that \( v'_j(c) = v_j(c) \) for all \( c \in C \) and \( 1 \leq i' \leq i \), by induction we can find \( \langle q_j, v'_j \rangle \xrightarrow{w, \delta} \langle q_j', v'_j \rangle \) such that \( v'_j(c^i) = v_j(c^i) \) for all \( j^i \in \{j + 1, \ldots, j'\} \) and \( i' \notin \{H_c(j^i) : x_1, \ldots, x_i \notin c \} \). Since \( v_j(c) = v_j(c) \) for all \( c \in C \) and \( 1 \leq i' \leq i \), it follows that \( C \) has an infinite sequence of transitions, obtained by following \( S \) to position \( j \), and then repeatedly simulating \( S' \). That is a contradiction, so there exists \( x^i \neq x_1, \ldots, x_i \) such that \( v_j(c) \leq 2a \cdot U_i^{|C|} + \sum_{i' = 1}^{j^i} |C| \cdot U_i < U_{i+1} \) for all \( c \in C \) and \( 1 \leq i \leq i \).

Let \( x_{i+1} \neq x_1, \ldots, x_i \) be such that there exists \( J_{i+1} \subseteq J_i \) with \( |J_{i+1}| = m/\alpha + 1 \) and \( v(c) < U_{i+1} \) for all \( j \in J_{i+1} \) and \( c \in C \). Then let \( u_{i+1} : C_{i+1} \rightarrow \{0, \ldots, U_{i+1} - 1\} \) be such that there exists \( J_{i+1} \subseteq J_i \) with \( |J_{i+1}| = m/\alpha + 1 \cdot U_i^{|C|} \) and \( v(c) = u_{i+1}(c) \) for all \( j \in J_{i+1} \) and \( c \in C_{i+1} \). That completes the inductive proof of the claim.

Since \( m = 2a |X| U_i^{|C|} \), we have from the claim above that \( S \) contains two equal states, so \( C \) has an infinite sequence of transitions from the initial state. That is a contradiction, completing the proof.

4. LOWER BOUND FOR NONEMPTINESS

THEOREM 4.1. Nonemptiness of safety IARA is ExpSpace-hard.

PROOF. We shall show ExpSpace-hardness by reducing from the halting problem for Turing machines with exponentially long tapes. More precisely, a Turing machine \( M \) is a tuple of \( \langle \Sigma, a_B, Q, q_I, \delta \rangle \) such that:

- \( \Sigma \) is a finite alphabet, and \( a_B \in \Sigma \) denotes the blank symbol;
- \( Q \) is a finite set of locations, and \( q_I \in Q \) is the initial location;
- \( \delta : Q \times \Sigma \rightarrow Q \times \Sigma \times \{-1, 1\} \) is the transition function.

If the size of \( M \) is \( n \), we consider its computation on a tape of length \( 2^n \). More formally, a state of \( M \) is of the form \( \langle q, i, w \rangle \) where \( q \in Q \) is the machine location, \( 0 \leq i < 2^n \) is the head position, and \( w \in \Sigma^{2^n} \) is the tape contents. The initial state is \( \langle q_I, 0, a_B^{2^n} \rangle \). A state \( \langle q, i, w \rangle \) has a transition iff \( 0 \leq i + \alpha < 2^n \) where \( \langle q', a, \alpha \rangle = \delta(q, w(i)) \). In that case, we write \( \langle q, i, w \rangle \rightarrow \langle q', i + \alpha, w[i \rightarrow a] \rangle \). Since
\( M \) can terminate by requesting to move the head off an edge of the tape, it does not need to have a special terminal location.

The following problem is \( \text{ExpSpace}-\text{complete} \): given \( M = \langle \Sigma, a_B, Q, q_I, \delta \rangle \) of size \( n \), is the computation from the initial state with tape length \( 2^n \) is infinite? We shall show that a safety \( 1\text{ARA}_1 \) \( A_M \) is computable in space logarithmic in \( n \), such that the answer to the decision problem is ‘yes’ if \( A_M \) is nonempty.

Let \( \hat{\Sigma} = \{ \hat{a} : a \in \Sigma \} \). The alphabet of \( A_M \) will be \( \tilde{\Sigma} = Q \cup \{0_d, 1_d : d \in \{1, \ldots, n\}\} \cup \Sigma \cup \hat{\Sigma} \). A state \( \langle q, i, w \rangle \) is encoded by the word
\[
q 0_1 \cdots 0_{n-1} 0_n w(0, i) 0_1 \cdots 0_{n-1} 1_n w(1, i) \cdots 1_1 \cdots 1_{n-1} 1_n w(2^n - 1, i)
\]
where \( w(i, i) = \hat{w}(i) \), and \( w(j, i) = w(j) \) for \( j \neq i \).

The computation of \( M \) from the initial state with tape length \( 2^n \) is infinite iff there exists a data \( \omega \)-word \( \sigma \) over \( \tilde{\Sigma} \) such that:

(i) \( \text{str}(\sigma) \) is a sequence of encodings of states of \( M \);
(ii) \( \text{str}(\sigma) \) begins with the encoding of the initial state \( \langle q_I, 0, 0 \rangle \);
(iii) for every two consecutive encodings in \( \text{str}(\sigma) \) of states \( \langle q, i, w \rangle \) and \( \langle q', i', w' \rangle \), we have \( \langle q, i, w \rangle \to \langle q', i', w' \rangle \).

Hence, it suffices to construct \( A_M \) which accepts \( \sigma \) iff (i)–(iii) hold and:

(iv) for every encoding in \( \sigma \) of a tape position, all the letters \( b_d \) and \( w(j, i) \) are in the same class;
(v) for every two encodings in \( \sigma \) of tape positions \( j \) and \( j' \) (occurring in one or two state encodings), their classes are the same iff \( j = j' \).

The purpose of (iv) and (v) is to enable \( A_M \) whose size will be only polynomial in \( n \) to navigate through \( \sigma \) for checking (i)–(iii).

Since safety \( 1\text{ARA}_1 \) are trivially closed under intersection, we can construct an automaton for each of (i)–(v) separately. For (i), we can split it into the following constraints, each of which is straightforward to express:

—the first letter is a location of \( M \);
—every location of \( M \) is succeeded by \( 0_1 \cdots 0_{n-1} 0_n \);
—every \( b_n \) is succeeded by an element of \( \Sigma \cup \hat{\Sigma} \);
—for every \( b_d \) not succeeded by \( 1_{d+1} \cdots 1_n, b_d \) occurs \( n + 1 \) positions later;
—for every \( 0_d \) succeeded by \( 1_{d+1} \cdots 1_n, 1_d 0_{d+1} \cdots 0_n \) occurs \( n + 1 \) positions later;
—\( 1_1 \cdots 1_{n-1} 1_n \) and then an element of \( \Sigma \cup \hat{\Sigma} \) are succeeded by a location of \( M \);
—between every two consecutive occurrences of locations of \( M \), there is exactly one occurrence of an element of \( \hat{\Sigma} \).

Properties (ii) and (iv) are also straightforward. Before (iii), let us consider (v), which is equivalent to the following conjunction:

(v.1) in the encoding of the first state, no two tape positions have the same class;
(v.2) for every encoding of a tape position, some tape position in the next state encoding has the same class.
(v.3) for every encoding of tape position $j$, its class is distinct from the classes for all tape positions $j' \neq j$ in the next state encoding.

The most involved is (v.3). However, assuming (v.1) and (v.2), it amounts to requiring that, for all $d \in \{1, \ldots, n\}$ and $b \in \{0, 1\}$, it is not the case that there is an occurrence of $b_d$ and a subsequent occurrence of $(1 - b)_d$ with the same datum such that $b_d$ with the same datum does not occur between them. The latter is expressed by the automaton depicted in Figure 2, with $a$ and $b$ substituted by $b_d$ and $(1 - b)_d$ (respectively).

Property (iii) is now equivalent to asserting that the following hold for all $q \in Q$ and $a \in \Sigma$, where $(q', a', o) = \delta(q, a)$:

(iii.1) whenever $q$ occurs with $\hat{a}$ in the same state encoding, the next occurrence of a location of $M$ is $q'$;

(iii.2) for every occurrence of some $b \in \Sigma$ in a state encoding which contains $q$ and $\hat{a}$, the next occurrence in the same class of an element of $\Sigma \cup \hat{\Sigma}$ is an occurrence of $b$ or $\hat{b}$;

(iii.3) for every occurrence of $\hat{a}$ in a state encoding containing $q$, the next occurrence in the same class of an element of $\Sigma \cup \hat{\Sigma}$ is an occurrence of $a'$, and $n$ positions earlier (if $o = -1$) or later (if $o = 1$) there is an occurrence of an element of $\hat{\Sigma}$.

The most involved is (iii.3), and the two cases of $o = -1$ and $o = 1$ are similar. A safety 1ARA for the former case is depicted in Figure 5, where $a_1, \ldots, a_k - 1$ enumerates $\Sigma \setminus \{a\}$, and $X^n$ abbreviates $n$ consecutive locations labelled by $X$.

5. COMPLEXITY OF INCLUSION

Using well-quasi-orderings, the proof of Theorem 3.2, and that universality over finite data words of one-way nondeterministic automata with 1 register is not primitive recursive [Demri and Lazic 2008, Theorem 5.2], we obtain the result below.

We remark that, in a similar manner, one can show that the following “model-checking” problem is decidable and not primitive recursive: whether the language of a Büchi one-way nondeterministic register automaton (with any number of registers) is included in the language of a safety 1ARA.

**Theorem 5.1.** Inclusion between two safety 1ARA is decidable and not primitive recursive.
PROOF. For decidability, suppose \( A \) and \( B \) are safety 1ARA\(_1\) with alphabet \( \Sigma \), and we need to determine whether \( L(A) \subseteq L(B) \).

Let \( \overline{B} \) be the dual automaton to \( B \), obtained by replacing every transition formula \( q \wedge q' \) by \( q \vee q' \), every \( \top \) by \( \bot \), and vice versa. Acceptance games for \( \overline{B} \) are defined in the same way as for safety 1ARA\(_1\), except that they are co-safety: player 1 wins at accepting states, and player 2 wins at rejecting states and wins every infinite play.

By [Demri and Lazic 2008, Theorem 2.7 (a)], \( L(\overline{B}) \) is the complement of \( L(B) \).

Since player 1 in acceptance games for \( \overline{B} \) only has winning strategies which are finite trees, the construction in the proof of Theorem 3.1 computes a safety IPCANT \( C_{A,\overline{B}} \) with alphabet \( \Sigma \) and a subset \( Y \) of its basis such that:

the intersection of \( L(A) \) and \( L(\overline{B}) \) is nonempty iff, from the initial state,

\( C_{A,\overline{B}} \) can reach a state \( \langle q, v \rangle \) which satisfies:

(i) \( v(c) = 0 \) for each counter \( c \) which intersects \( Y \);

(ii) there exists an infinite computation from \( \langle q, v \rangle \).

We define \( \preceq \) to be the following quasi-ordering on states of \( C_{A,\overline{B}} \): \( \langle q, v \rangle \preceq \langle q', v' \rangle \) iff \( q = q' \) and \( v \preceq v' \). By Dickson’s Lemma [Dickson 1913], \( \preceq \) is a well-quasi-ordering: for every infinite sequence \( s_0, s_1, \ldots \), there exist \( i < j \) such that \( s_i \preceq s_j \).

Now, consider the following procedure:

1. Let \( S \) consist of the initial state of \( C_{A,\overline{B}} \).
2. Let \( S' \) be the set of all successors of states in \( S \) by lazy transitions.
3. If for all \( s' \in S' \) there exists \( s \in S \) with \( s \preceq s' \), then stop. Otherwise, set \( S \) to \( S \cup S' \), and repeat from (2).

Since \( \preceq \) is a well-quasi-ordering, the procedure terminates. The last \( S \) is a finite set, and by Lemma 2.4, its upward closure \( \uparrow S = \{ s' : \exists s \in S(s \preceq s') \} \) is the set of all states which \( C_{A,\overline{B}} \) can reach from the initial state. Observing that the set of all \( \langle q, v \rangle \) which satisfy (i) and (ii) above is downwards closed, it remains to check whether (i) and (ii) are true of some state in \( S \). For \( \langle q, v \rangle \in S \), checking (i) is trivial, and whether (ii) holds can be determined by replacing the initial state in the procedure in the proof of Theorem 3.2 by \( \langle q, v \rangle \).

Non-primitive recursiveness holds already for universality of safety 1ARA\(_1\), even with restricting the automata to nondeterministic (i.e. without transition formulae of the form \( q \wedge q' \)). Namely, by [Demri and Lazic 2008, Theorem 5.2], universality over finite data words of one-way nondeterministic automata with 1 register is not primitive recursive. Given such an automaton \( B \) with alphabet \( \Sigma \), let \( \hat{B} \) be a safety 1ARA\(_1\) with alphabet \( \Sigma \cup \hat{\Sigma} \) which accepts a data \( \omega \)-word \( \sigma \) iff:

—either \( \sigma \) contains no letter of \( \hat{\Sigma} \),
—or \( B \) accepts the prefix of \( \sigma \) up to and including the first occurrence of a letter of \( \hat{\Sigma} \), with the latter replaced by the corresponding letter of \( \Sigma \).

We have that \( \hat{B} \) is computable in logarithmic space and it is universal iff \( A \) is. \( \Box \)

6. CONCLUDING REMARKS

Satisfiability (over timed \( \omega \)-words) for the safety fragment of metric temporal logic (MTL) was shown decidable in [Ouaknine and Worrell 2006], and nonelementary in
by reducing from termination of channel machines with emptiness testing and insertion errors. It would be interesting to investigate whether ideas in the proof of Theorem 3.2 above can be combined with those in the proof of primitive recursiveness of termination of channel machines with occurrence testing and insertion errors [Bouyer et al. 2008] to obtain that satisfiability for safety MTL is primitive recursive.

Another open question is whether nonemptiness of safety forward alternating tree automata with 1 register [Jurdziński and Lazić 2007] is primitive recursive.

ACKNOWLEDGMENTS
I am grateful to Stéphane Demri and James Worrell for helpful discussions.

REFERENCES
Alpern, B. and Schneider, F. B. 1987. Recognizing safety and liveness. Distr. Comput. 2, 3, 117–126.
Björklund, H. and Schwentick, T. 2007. On notions of regularity for data languages. In Fundamentals of Comput. Theory (FCT), 16th Int. Symp. Lect. Notes Comput. Sci., vol. 4639. Springer, 88–99.
Bojańczyk, M., Muscholl, A., Schwentick, T., Segoufin, L., and David, C. 2006. Two-variable logic on words with data. In 21th IEEE Symp. on Logic in Comput. Sci. (LICS). IEEE Comput. Soc., 7–16.
Bouyer, P., Markey, N., Ouaknine, J., Schnoebelen, P., and Worrell, J. 2008. On termination for faulty channel machines. In 25th Int. Symp. on Theor. Asp. of Comput. Sci. (STACS). IBFI, Schloss Dagstuhl, Germany, 121–132.
David, C. 2004. Mots et données infinies. M.S. thesis, Laboratoire d’Informatique Algorithmique: Fondements et Applications, Paris.
Demri, S. and Lazić, R. 2008. LTL with the freeze quantifier and register automata. ACM Trans. On Comp. Logic. Provisionally accepted, revised version is available at http://arxiv.org/abs/cs.LO/0610027.
Dickson, L. 1913. Finiteness of the odd perfect and primitive abundant numbers with distinct factors. Amer. J. Math. 35, 413–422.
Jurdziński, M. and Lazić, R. 2007. Alternation-free modal mu-calculus for data trees. In 22nd IEEE Symp. on Logic in Comput. Sci. (LICS). IEEE Comput. Soc., 131–140.
Kaminski, M. and Francez, N. 1994. Finite-memory automata. Theor. Comput. Sci. 134, 2, 329–363.
Lazić, R. 2006. Safely freezing LTL. In FSTTCS: Found. of Softw. Technology and Theor. Comput. Sci., 26th Int. Conf. Lect. Notes Comput. Sci., vol. 4337. Springer, 381–392.
Lipton, R. J. 1976. The reachability problem requires exponential space. Tech. Rep. 62, Yale University.
Muller, D. E., Saoudi, A., and Schupp, P. E. 1986. Alternating automata, the weak monadic theory of the tree, and its complexity. In Automata, Lang. and Program., 13th Int. Coll. (ICALP). Lect. Notes Comput. Sci., vol. 226. Springer, 275–283.
Neven, F., Schwentick, T., and Vianu, V. 2004. Finite state machines for strings over infinite alphabets. ACM Trans. On Comp. Logic 5, 3, 403–435.
Ouaknine, J. and Worrell, J. 2006. Safety Metric temporal logic is fully decidable. In Tools and Algorithms for the Constr. and Anal. of Systems (TACAS), 12th Int. Conf. Lect. Notes Comput. Sci., vol. 3920. Springer, 411–425.
Sakamoto, H. and Ikeda, D. 2000. Intractability of decision problems for finite-memory automata. Theor. Comput. Sci. 231, 2, 297–308.
Segoufin, L. 2006. Automata and logics for words and trees over an infinite alphabet. In Comput. Sci. Logic (CSL), 20th Int. Works. Lect. Notes Comput. Sci., vol. 4207. Springer, 41–57.