Two Results on Discontinuous Input Processing

Vojtěch Vorel(✉)

Faculty of Mathematics and Physics, Charles University,
Malostranské nám. 25, Prague, Czech Republic
vorel@ktiml.mff.cuni.cz

Abstract. First, we show that universality and other properties of general jumping finite automata are undecidable, which answers questions asked by Meduna and Zemek in 2012 [12]. Second, we close a study started by Černo and Mráz in 2010 [3] by proving that a clearing restarting automaton using contexts of length two can accept a binary non-context-free language.

1 Introduction

In 2012, Meduna and Zemek [12,13] introduced *general jumping finite automata* as a model of discontinuous information processing in modern software. A general jumping finite automaton (GJFA) is described by a finite set Q of states, a finite alphabet Σ, a finite set R of *rules* from $Q \times \Sigma^* \times Q$, an initial state $q_0 \in Q$, and a set $F \subseteq Q$ of final states. In a step of computation, the automaton switches from a state r to a state s using a rule $(r, v, s) \in R$ and deletes a factor equal to v from any part of the input word. A rule (r, v, s) and an occurrence of the factor v are chosen nondeterministically (in other words, the read head can *jump* to any position). A word $w \in \Sigma^*$ is accepted if the GJFA can reduce w to the empty word while passing from the initial state to an accepting state. The boldface term **GJFA** refers to the class of languages accepted by GJFA. The initial work [12,13] deals mainly with closure properties of **GJFA** and its relations to classical language classes (the publications [12,13] contain flaws, see [17]). It turns out that the class **GJFA** is not closed under operations related to continuous processing (concatenation, Kleene star, homomorphism, inverse homomorphism, shuffle) nor some Boolean closure operations (complementation, intersection). The class is incomparable with both regular and context-free languages. It is a proper subclass of both context-sensitive languages and of the class NP, while there exist NP-complete **GJFA**languages (see [5], which is an extended version of [6]).

On the other hand, the concept of *restarting automata* [10,14] is motivated by reduction analysis and grammar checking of natural language sentences. In 2010, Černo and Mráz [3] introduced a subclass named *clearing restarting automata*

Research supported by the Czech Science Foundation grant GA14-10799S and the GAUK grant No. 52215.

C. Câmpeanu et al. (Eds.): DCFS 2016, LNCS 9777, pp. 205–216, 2016.
DOI: 10.1007/978-3-319-41114-9_16

(cl-RA) in order to describe systems that use only very basic types of reduction rules (see also [2]). Clearing restarting automata may delete factors according to contexts and endmarks, but, unlike GJFA and classical restarting automata, they are not controlled by states and rules. A key property of a cl-RA is the maximum length k of context used. For $k \geq 0$, a *k-clearing restarting automaton* (k-cl-RA) is described by a finite alphabet Σ and a finite set I of instructions of the form $(u_{\mathrm{L}}, v, u_{\mathrm{R}})$, where $v \in \Sigma^*$, $u_{\mathrm{L}} \in \Sigma^k \cup ¢\Sigma^{k-1}$, and $u_{\mathrm{R}} \in \Sigma^k \cup \Sigma^{k-1}\$$. The words $u_{\mathrm{L}}, u_{\mathrm{R}}$ specify the left and right context for consuming a factor v, while ¢ and \$ stand for the left and right end of input, respectively. A word is accepted by a cl-RA if it may be completely consumed using a series of instructions. The class of languages accepted by cl-RA is not closed under complementation, intersection, or union [3]. It forms a superset of regular languages, a subset of context-sensitive languages, and is incomparable with context-free languages [3].

Tough both the formalisms are defined as acceptors, they may be equivalently treated as generative systems. Moreover, they share important properties with *insertion systems* [16] (possibly *graph-controlled* [1]) and semi-contextual grammars [15] (possibly using *regular control without appearance checking* [11]), as we briefly discuss in the conclusion. The present paper consists of two main parts:

In Sect. 3 we show that, given a GJFA M with an alphabet Σ, it is undecidable whether M accepts the universal language Σ^*. In other words, *universality* of GJFA is undecidable. As a direct consequence, the more general problems of *equivalence* and *inclusion* are undecidable for GJFA as well. Decidability of these tasks was listed as an open problem in [12,13].

In Sect. 4 we deal with expressive power of cl-RA with short contexts and small alphabets, as it was addressed in [3]. The authors showed that a language accepted by a 2-cl-RA may not be context-free, but the example automata required at least six-letter alphabets, so they asked what is the least sufficient alphabet size. We provide a binary example, which forms a tight bound.

2 Preliminaries

We use the notion of *insertion* as it was defined, e.g., in [4,7,9]:

Definition 1. *Let $K, L \subseteq \Sigma^*$ be languages. The* insertion *of K to L is*

$$L \leftarrow K = \{u_1 v u_2 \mid u_1 u_2 \in L, v \in K\}.$$

More generally, for each $k \geq 1$ we denote

$$L \leftarrow^k K = \left(L \leftarrow^{k-1} K\right) \leftarrow K,$$
$$L \leftarrow^* K = \bigcup_{i \geq 0} L \leftarrow^i K,$$

where $L \leftarrow^0 K$ stands for L. In expressions with $\leftarrow$ and $\leftarrow^*$, a singleton set $\{w\}$ may be replaced by w.

A chain $L_1 \leftarrow L_2 \leftarrow \cdots \leftarrow L_d$ of insertions is evaluated from the left, e.g., $L_1 \leftarrow L_2 \leftarrow L_3$ means $(L_1 \leftarrow L_2) \leftarrow L_3$. The empty word is denoted by ϵ.

As described above, a GJFA is a quintuple $M = (Q, \Sigma, R, q_0, F)$. For a rule $(r, v, s) \in R$ with $r, s \in Q$, the word $v \in \Sigma^*$ is called the *label* of the rule. A sequence

$$(r_1, v_1, s_1), (r_2, v_2, s_2), \ldots, (r_k, v_k, s_k)$$

of rules from R is a *path* if $k \geq 1$ and $s_i = r_{i+1}$ for $1 \leq i \leq k-1$. The sequence $v_1, v_2, \ldots, v_k$ is the *labeling* of the path. The path is *accepting* if $r_1 = q_0$ and $s_k \in F$. The original definition [12,13] of the language $L(M)$ accepted by M is based on *configurations* that specify positions of the read head (i.e., starting positions of the factor to be erased in the next step). For our proofs, this type of configurations is useless, whence we directly use the following generative characterization [17, Corollary 1] of $L(M)$ as a definition:

Definition 2. *Let $M = (Q, \Sigma, R, s, F)$ be a GJFA and $w \in \Sigma^*$. Then $w \in \mathrm{L}(M)$ if and only if $w = \epsilon$ and $s \in F$, or*

$$w \in \epsilon \leftarrow v_d \leftarrow v_{d-1} \leftarrow \cdots \leftarrow v_2 \leftarrow v_1, \tag{1}$$

where $d \geq 1$ and $v_1, v_2, \ldots, v_d$ is a labeling of an accepting path in M.

If a GJFA $M = (Q, \Sigma, R, s, F)$ is clear, we write $(r, w) \curvearrowright (s, u)$ for $r, s \in Q$ and $u, v \in \Sigma^*$ if $w \in u \leftarrow v$ for some $(r, v, s) \in R$.

In the case of clearing restarting automata we include the original definition, which builds on *context rewriting systems* [3]:

Definition 3. *For $k \geq 0$, a k-*context rewriting system *is a tuple $M = (\Sigma, \Gamma, I)$, where Σ is an input alphabet, $\Gamma \supseteq \Sigma$ is a working alphabet not containing the special symbols ¢ and \$, called* sentinels*, and I is a finite set of* instructions *of the form*

$$(u_\mathrm{L}, v \to t, u_\mathrm{R}),$$

where u_L is a left context*, $u_\mathrm{L} \in \Gamma^k \cup ¢\Gamma^{k-1}$, u_R is a* right context*, $u_\mathrm{R} \in \Gamma^k \cup \Gamma^{k-1}\$$, and $v \to t$ is a* rule*, $v, t \in \Gamma^*$. A word $w = u_1 v u_2$ can be rewritten into $u_1 t u_2$ (denoted by $u_1 v u_2 \to_M u_1 t u_2$) if and only if there exists an instruction $(u_\mathrm{L}, v \to t, u_\mathrm{R}) \in I$ such that u_L is a suffix of $¢u_1$ and u_R is a prefix of $u_2\$$.*

We use the star in $\curvearrowright^*, \to^*, \dashv^*$ and other symbols to denote reflexive-transitive closures of binary relations.

Definition 4. *For $k \geq 0$, a k-*clearing restarting automaton *(k-cl-RA) is a system $M = (\Sigma, I)$, where $M' = (\Sigma, \Sigma, I)$ is a k-context rewriting system such that for each $\mathbf{i} = (u_\mathrm{L}, v \to t, u_\mathrm{R}) \in I$ it holds that $v \in \Sigma^+$ and $t = \epsilon$. Since t is always the empty word, the notation $\mathbf{i} = (u_\mathrm{L}, v, u_\mathrm{R})$ is used. A k-cl-RA M accepts the language*

$$L(M) = \{w \in \Sigma^* \mid w \vdash_M^* \epsilon\},$$

where $\vdash_M$ denotes the rewriting relation $\to_{M'}$ of M'. The term $\mathcal{L}(k\text{-cl-RA})$ denotes the class of languages accepted by k-cl-RA.

The generative approach is formalized by writing $w_2 \dashv w_1$ instead of $w_1 \vdash w_2$.

3 Undecidability in General Jumping Finite Automata

Theorem 5. *Given a GJFA $M = (Q, \Sigma, R, s, F)$, it is undecidable whether $L(M) = \Sigma^*$.*

Let us prove the theorem. Given a context-free grammar G with terminal alphabet Σ_{T}, it is undecidable whether $L(G) = \Sigma_{\mathrm{T}}^*$ [8]. We present a reduction from this problem to the universality of GJFA. Assume that the given grammar G

- has non-terminal alphabet Σ_{N} and a start symbol $A_{\mathrm{S}} \in \Sigma_{\mathrm{N}}$,
- accepts the empty word ϵ, and
- is given in Greibach normal form [8], i.e., the rules are $A_S \to \epsilon$ and $A_i \to u_i$, where $A_i \in \Sigma_{\mathrm{N}}$ and $u_i \in \Sigma_{\mathrm{T}}\Sigma_{\mathrm{N}}^*$ for $i \in \{1, \ldots, m\}$, $m \geq 0$.

Note that any context-free grammar that accepts ϵ can be algorithmically converted to the form above. Next, we construct a GJFA $M_G = (Q, \Gamma, R, s, F)$ as follows, denoting $\Sigma_{\mathrm{B}} = \{b_1, \ldots, b_m\}$:

$$\begin{aligned} Q &= \{q_0, q_1, q_2, q_3, q_4\}, \\ \Gamma &= \Sigma_{\mathrm{T}} \cup \Sigma_{\mathrm{N}} \cup \Sigma_{\mathrm{B}}, \end{aligned}$$

$s = q_0$, $F = \{q_2, q_4\}$. The set R of rules is defined in Fig. 1. In this figure, each arrow labeled with a finite set $S \subseteq \Gamma^*$ stands for $|S|$ rules, each labeled with a word $v \in S$. The following finite sets are used:

$$\begin{aligned} P_{\mathrm{BU}} &= \{b_i u_i \mid i = 1, \ldots, m\}, \\ P_{\mathrm{NB}} &= \{A_i b_i \mid i = 1, \ldots, m\}, \end{aligned}$$

$$\begin{aligned} P_{\mathrm{C}} = &\{x A_1 \mid x \in \Sigma_{\mathrm{T}}\} \\ &\cup \{A_i b_i \mid i = 1, \ldots, m\} \\ &\cup \{b_i A_{i+1} \mid i = 1, \ldots, m-1\} \\ &\cup \{b_m x \mid x \in \Sigma_{\mathrm{T}}\}. \end{aligned}$$

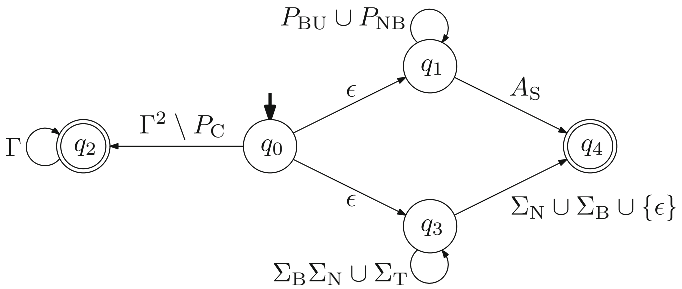

Fig. 1. The GJFA M_G corresponding to a context-free grammar G

For a word $w \in \Gamma^*$ we denote with w_{T} and $w_{\mathrm{N,B}}$ the projections of w to subalphabets Σ_{T} and $\Sigma_{\mathrm{N}} \cup \Sigma_{\mathrm{B}}$ respectively[1] Let us show that $L(G) = \Sigma_{\mathrm{T}}^*$ if and only if $L(M_G) = \Gamma^*$.

First, suppose that $L(G) = \Sigma_{\mathrm{T}}^*$ and take an arbitrary $w \in \Gamma^*$. Describe a derivation of w_{T} by G using $v_0, v_1, \ldots, v_d \in (\Sigma_{\mathrm{T}} \cup \Sigma_{\mathrm{N}})^*$, $d \geq 1$, where

$$\begin{aligned} v_0 &= A_{\mathrm{S}}, \\ v_d &= w_{\mathrm{T}}, \\ v_k &= v_{\mathbf{p},k} A_{i_k} v_{\mathbf{s},k}, \\ v_{k+1} &= v_{\mathbf{p},k} u_{i_k} v_{\mathbf{s},k} \end{aligned}$$

for each $k \in \{0, \ldots, d-1\}$. For $k \in \{0, \ldots, d\}$, we define inductively a word $w_k \in \Gamma^*$ and a mapping σ_k from each occurrence of $x \in \Sigma_{\mathrm{N}}$ in v_k to an occurrence of the same x in w_k. First, $w_0 = A_{\mathrm{S}}$ and σ_0 is trivial. Next, take $0 \leq k \leq d-1$ and write $w_k = w_{\mathbf{p},k} A_{i_k} w_{\mathbf{s},k}$ such that the A_{i_k} right after $w_{\mathbf{p},k}$ is the σ_k-image of the A_{i_k} right after $v_{\mathbf{p},k}$ in v_k. Then define

$$w_{k+1} = w_{\mathbf{p},k} A_{i_k} b_{i_k} u_{i_k} w_{\mathbf{s},k}$$

and let σ_{k+1} extend σ_k with mapping the occurrences of $x \in \Sigma_{\mathrm{N}}$ within the factor u_{i_k} in v_{k+1} to the corresponding occurrences within the same factor in w_{k+1}. Informally, the words $w_0, \ldots, w_d$ describe the derivation of w_{T} with keeping all the used nonterminals, i.e., A_{i_k} is rewritten with $A_{i_k} b_{i_k} u_{i_k}$ instead of u_{i_k}. Observe that $(q_1, w_d) \curvearrowright^* (q_1, A_{\mathrm{S}})$ using the rules labeled with words from P_{BU}. Also observe that, due to Greibach normal form, $w_d \in (\Sigma_{\mathrm{T}} \cup \Sigma_{\mathrm{T}} \Sigma_{\mathrm{N}} \Sigma_{\mathrm{B}})^*$, i.e., the factors from $\Sigma_{\mathrm{N}} \Sigma_{\mathrm{B}}$ are always separated with letters from Σ_{T}.

Distinguish the following cases:

– If w does not have a factor from $\Gamma^2 \backslash P_{\mathrm{C}}$, all two-letter factors of w belong to P_{C}, which implies that w is a factor of a word from $(\Sigma_{\mathrm{T}} t)^*$, where

$$t = A_1 b_1 A_2 b_2 \cdots A_m b_m. \tag{2}$$

  - If w starts with a letter from $\Sigma_{\mathrm{T}} \cup \Sigma_{\mathrm{N}}$ and ends with a letter from $\Sigma_{\mathrm{T}} \cup \Sigma_{\mathrm{B}}$, then $(q_1, w) \curvearrowright^* (q_1, w_d)$ using the rules labeled with words from P_{NB}. Because $(q_1, w_d) \curvearrowright^* (q_1, A_{\mathrm{S}})$, we conclude that $w \in L(M_G)$.
  - Otherwise, w starts with a letter from Σ_{B} or ends with a letter from Σ_{N}. Then
  $$w_{\mathrm{N,B}} \in \Sigma_{\mathrm{B}} \left(\Sigma_{\mathrm{N}} \Sigma_{\mathrm{B}}\right)^* \cup \left(\Sigma_{\mathrm{N}} \Sigma_{\mathrm{B}}\right)^* \Sigma_{\mathrm{N}} \cup \Sigma_{\mathrm{B}} \left(\Sigma_{\mathrm{N}} \Sigma_{\mathrm{B}}\right)^* \Sigma_{\mathrm{N}}$$
  and we observe that $(q_0, w) \curvearrowright (q_3, w) \curvearrowright^* (q_3, w_{\mathrm{N,B}}) \curvearrowright (q_3, u)$ for some $u \in \Sigma_{\mathrm{N}} \cup \Sigma_{\mathrm{B}} \cup \{\varepsilon\}$. As $(q_3, u) \curvearrowright (q_4, \epsilon)$, we get $w \in L(M_G)$.

– If w has a factor $u \in \Gamma^2 \backslash P_{\mathrm{C}}$, write $w = w_{\mathbf{p}} u w_{\mathbf{s}}$ and observe

$$(q_0, w_{\mathbf{p}} u w_{\mathbf{s}}) \curvearrowright (q_2, w_{\mathbf{p}} w_{\mathbf{s}}) \curvearrowright^* (q_2, \epsilon),$$

implying $w \in L(M_G)$.

[1] A *projection* to $\Gamma' \subseteq \Gamma$ is given by the homomorphism that maps $x \in \Gamma$ to x if $x \in \Gamma'$ or to ϵ otherwise.

Second, suppose that $L(M_G) = \Gamma^*$ and take an arbitrary $v = x_1 x_2 \cdots x_n \in \Sigma_{\mathrm{T}}^*$ with $x_1, \ldots, x_n \in \Sigma_{\mathrm{T}}$. Let $w = (x_1 t)\,(x_2 t) \cdots (x_{n-1} t)\,(x_n t)$, with t defined in (2). We have $w \in L(M_G)$. Observe that:

– The word w does not contain a factor from $\Gamma^2 \backslash P_{\mathrm{C}}$.
– By deleting factors from $\Sigma_{\mathrm{B}} \Sigma_{\mathrm{N}} \cup \Sigma_{\mathrm{T}}$, the word w cannot become a word from $\Sigma_{\mathrm{N}} \cup \Sigma_{\mathrm{B}} \cup \{\epsilon\}$.

Thus, w is accepted by M using a path through the state q_1 ending in the state q_4. In other words, w can be obtained by inserting words from $P_{\mathrm{BU}} \cup P_{\mathrm{NB}}$ to A_{S}. During that process, once an occurrence of b_i fails to be preceded by A_i, this situation lasts to the very end, which is a contradiction. It follows that $b_i u_i \in P_{\mathrm{BU}}$ can be inserted only to the right of an occurrence of A_i that is not followed by b_i. This corresponds to rewriting A_i with u_i, so we can observe that the whole looping on q_1 (viewed backwards) corresponds to generating $w_{\mathrm{T}} = v$ from A_{S} using the rules of G. □

Because it is easy to construct a GJFA accepting Σ^*, universality is a special case of both equivalence and inclusion. Thus, the following claim is trivial:

Corollary 6. *Given GJFA M_1 and M_2, it is undecidable both whether $L(M_1) = L(M_2)$ and whether $L(M_1) \subseteq L(M_2)$.*

4 Clearing Restarting Automata with Small Contexts

Recall that the following facts were formulated and proved in [3]:

1. For each $k \geq 3$, the class $\mathcal{L}(k\text{-cl-RA})$ contains a binary language that is not context-free.
2. The class $\mathcal{L}(2\text{-cl-RA})$ contains a language $L \subseteq \Sigma^*$ with $|\Sigma| = 6$ that is not context-free.
3. The class $\mathcal{L}(1\text{-cl-RA})$ contains only context-free languages.

Moreover, for each $k \geq 1$, all the unary languages lying in $\mathcal{L}(k\text{-cl-RA})$ are regular [3]. The present section is devoted to proving the following theorem, which completes the results listed above.

Theorem 7. *The class $\mathcal{L}(2\text{-cl-RA})$ contains a binary language that is not context-free.*

In order to prove Theorem 7, we define two particular rewriting systems:

1. A 1-context rewriting system $R_{\mathrm{uV}} = (\{\mathrm{u}, \mathrm{V}\}, \{\mathrm{u}, \mathrm{V}\}, I_{\mathrm{uV}})$. The set I_{uV} is listed in Table 1.
2. A 2-clearing restarting automaton $R_{01} = (\{0, 1\}, I_{01})$. The set I_{01} is listed in Table 2.

Note that headings of the tables provide identifiers of rules. We write $\rightarrow_{\mathrm{uV}}$ for the rewriting relation of R_{uV} and $\dashv_{01}$ for the "generative" relation of R_{01}.

The key feature of the system R_{uV} is:

Table 1. The rules I_{uV}

0	$(¢, \epsilon \to \mathrm{uu}, \$)$
1	$(¢, \mathrm{u} \to \mathrm{uuV}, \epsilon)$
2	$(\epsilon, \mathrm{Vu} \to \mathrm{uuuV}, \epsilon)$
3	$(\epsilon, \mathrm{Vu} \to \mathrm{uuuu}, \$)$

Table 2. The rules I_{01}

	a	b	c	d
0	$(¢, 00, \$)$	-	-	-
1	$(¢, 10, 00)$	$(¢, 00, 10)$	-	-
2	$(01, 10, 00)$	$(00, 11, 01)$	$(11, 00, 10)$	$(10, 01, 11)$
3	$(01, 10, 0\$)$	$(00, 11, 0\$)$	-	-

Lemma 8. *Let $w \in L(R_{\mathrm{uV}}) \cap \{\mathrm{u}\}^*$. Then $|w| = 2 \cdot 3^n$ for some $n \geq 0$.*

The proof is postponed to Sect. 4.1. Next, we define:

1. A length-preserving mapping $\varphi : \{0,1\}^* \to \{\mathrm{u}, \mathrm{V}\}^*$ as $\varphi(x_1 \dots x_n) = \overline{x}_1 \dots \overline{x}_n$, where

$$\overline{x}_k = \begin{cases} \mathrm{V} & \text{if } 1 < k < n \text{ and } x_{k-1} = x_{k+1} \\ \mathrm{u} & \text{otherwise} \end{cases}$$

   for each $k \in \{1, \dots, n\}$.
2. A regular language $K \subseteq \{0,1\}^*$:

$$K = \left\{ w \in \{0,1\}^* \mid w \text{ has none of the factors } 000, 010, 101, 111 \right\}.$$

The following is a trivial property of φ and K. Informally, $\varphi(u)$ marks by V the positions where a *defect* occurs in $u \in \{0,1\}^*$. A defect is a position that violates the form $\dots 00110011 \dots$, i.e., a position whose neighbours are equal:

Lemma 9. *Let $u \in \{0,1\}^*$. Then $u \in K$ if and only if $\varphi(u) \in \{\mathrm{u}\}^*$.*

We index the rules from I_{uV} and I_{01} by the rows of Tables 1 and 2, i.e., by *types* 0 to 3. For a string $w = x_1 x_2 \dots x_d$, where $x_1, x_2, \dots, x_d$ are letters, and for integers i, j with $1 \leq i \leq j \leq d$, we denote $w[i,j] = x_i x_{i+1} \dots x_j$ and $w[i, \dots] = w[i, d]$.

The next lemma describes how the systems R_{01} and R_{uV} are related. Informally, a rule of the type 2 from I_{01} can be applied only right after a defect in $u \in \{0,1\}^*$. This creates another defect on the right, i.e., a factor $x_1 x_2 y_1 y_2$ of u with defect on x_2 is replaced with $x_1 x_2 z_1 z_2 y_1 y_2$ with defect on y_1. This corresponds to applying the rule $\mathrm{Vu} \to \mathrm{uuuV}$ to the defect markers. A rule of the type 1 from I_{01} can introduce a new defect near the beginning of $u \in \{0,1\}^*$, while a rule of type 3 from I_{01} can remove a defect near to the end:

Lemma 10. *Let $u, v \in \{0,1\}^*$. If $u \dashv_{01} v$, then $\varphi(u) \to_{\mathrm{uV}} \varphi(v)$.*

Proof. For $u = v$ the claim is trivial, so we suppose $u \neq v$. Denote $m = |u|$. As u can be rewritten to v using a single rule of R_{01}, we can distinguish which of the rule types is used:

(0) If the rule 0 is used, we have $u = \epsilon$ and $v = 00$. Thus $\varphi(u) = \epsilon$ and $\varphi(v) = \mathrm{uu}$.

(1) If a rule $(¢, z_1z_2, y_1y_2)$ of the type 1 is used, we see that v has some of the prefixes $1000, 0010$ and so $\varphi(v)$ starts with uuV. Trivially, $\varphi(u)$ starts with u. Because $u[1,\dots] = v[3,\dots]$, we have $\varphi(u)[2,\dots] = \varphi(v)[4,\dots]$ and we conclude that applying the rule $(¢, \mathrm{u} \to \mathrm{uuV}, \epsilon)$ rewrites $\varphi(u)$ to $\varphi(v)$.

(2) If a rule (x_1x_2, z_1z_2, y_1y_2) of the type 2 is used, we have

$$u[k, k+3] = x_1x_2y_1y_2,$$
$$v[k, k+5] = x_1x_2z_1z_2y_1y_2$$

for some $k \in \{1, \dots, m-3\}$. As $x_1x_2y_1y_2$ equals some of the factors 0100, 0001, 1110, 1011, we have

$$\varphi(u)[k+1, k+2] = \mathrm{Vu}.$$

As $x_1x_2z_1z_2y_1y_2$ equals some of the factors 011000, 001101, 110010, 100111, we have

$$\varphi(v)[k+1, k+4] = \mathrm{uuuV}.$$

Because $u[1, k+1] = v[1, k+1]$ and $u[k+2, \dots] = v[k+4, \dots]$, we have

$$\varphi(u)[1, k] = \varphi(v)[1, k]\,,$$
$$\varphi(u)[k+3, \dots] = \varphi(v)[k+5, \dots]\,.$$

Now it is clear that the rule $(\epsilon, \mathrm{Vu} \to \mathrm{uuuV}, \epsilon)$ rewrites $\varphi(u)$ to $\varphi(v)$.

(3) If a rule $(x_1x_2, z_1z_2, y\$)$ of the type 3 is used, we have

$$u[m-2, m] = x_1x_2y,$$
$$v[m-2, m+2] = x_1x_2z_1z_2y.$$

As x_1x_2y equals some of the factors 010, 000, we have

$$\varphi(u)[m-1, m] = \mathrm{Vu}.$$

As $x_1x_2z_1z_2y$ equals some of the factors 01100, 00110, we have

$$\varphi(v)[m-1, m+2] = \mathrm{uuuu}.$$

Because $u[1, m-1] = v[1, m-1]$, we have

$$\varphi(u)[1, m-2] = \varphi(v)[1, m-2]\,,$$

Now it is clear that the rule $(\epsilon, \mathrm{Vu} \to \mathrm{uuuu}, \$)$ rewrites $\varphi(u)$ to $\varphi(v)$. □

Corollary 11. *If $u \in L(R_{01})$, then $\epsilon \to^*_{\mathrm{uV}} \varphi(u)$.*

Proof. Follows from the fact that $\varphi(\epsilon) = \epsilon$ and a trivial inductive use of Lemma 10. □

Note that $L(R_{01})$ contains, e.g., 00 and 100110. Informally, the claims above imply that $L(R_{01})$ contains only words without defects and that each word from $L(R_{01})$ is obtained from 00 by adding defects to the beginning and pushing them to the end, while the length of the word is tripled for each processed defect. It remains to show that a defect can be always avoided. It turns out to be convenient to describe simultaneous processing of two defects that are close to each other.

The last part of the proof of Theorem 7 relies on the following lemma, whose proof is postponed to Sect. 4.2:

Lemma 12. *For each $\alpha \geq 0$ and $\beta \geq 1$ it holds that*

$$00\,(1100)^{\alpha}\,10\,(0011)^{\beta}\,00 \dashv_{01}^{*} 00\,(1100)^{\alpha+9}\,10\,(0011)^{\beta-1}\,00.$$

Corollary 13. *For each $\gamma \geq 0$ it holds that*

$$0010\,(0011)^{\gamma}\,00 \dashv_{01}^{*} 00\,(1100)^{9\gamma}\,1000.$$

Proof. As the left-hand side equals $00\,(1100)^{0}\,10\,(0011)^{\gamma}\,00$ and the right-hand side equals $00\,(1100)^{9\gamma}\,10\,(0011)^{0}\,00$, the claim follows from Lemma 12 applied γ times. $\square$

Corollary 14. *The language $L(R_{01}) \cap K$ is infinite.*

Proof. We show that for each $k \geq 0$,

$$00\,(1100)^{\frac{2\cdot 9^{k}-2}{4}} \in L(R_{01})\,.$$

In the case of $k = 0$ we just check that $00 \in L(R_{01})$. Next, we suppose that the claim holds for a fixed $k \geq 0$ and show that

$$00\,(1100)^{\frac{2\cdot 9^{k}-2}{4}} \dashv_{01}^{*} 00\,(1100)^{\frac{2\cdot 9^{k+1}-2}{4}}\,.$$

Using the rules 1a and 1b we get

$$00\,(1100)^{\frac{2\cdot 9^{k}-2}{4}} \dashv_{01} 1000\,(1100)^{\frac{2\cdot 9^{k}-2}{4}} \dashv_{01} 001000\,(1100)^{\frac{2\cdot 9^{k}-2}{4}}\,,$$

while Corollary 13 continues with

$$0010\,(0011)^{\frac{2\cdot 9^{k}-2}{4}}\,00 \dashv_{01}^{*} 00\,(1100)^{\frac{2\cdot 9^{k+1}-18}{4}}\,1000.$$

Finally, denoting $p = 00\,(1100)^{\frac{2\cdot 9^{k+1}-18}{4}}$, using rules 3b, 2a, 2b, 2d, 2c, and 3a respectively, we get

$$p1000 \dashv_{01} p100\underline{11}0 \dashv_{01} p1\underline{10}00110 \dashv_{01} p\,(1100)\,\underline{11}0110 \dashv_{01} p\,(1100)\,110\underline{01}110 \dashv_{01}$$
$$\dashv_{01} p\,(1100)\,(1100)\,11\underline{00}10 \dashv_{01} p\,(1100)\,(1100)\,(1100)\,1\underline{10}0 = 00\,(1100)^{\frac{2\cdot 9^{k+1}-2}{4}}\,.$$

$\square$

We conclude the proof of Theorem 7 by pointing out that Lemmas 8, 9, and 10 say that for each $w \in \{0,1\}^*$ we have

$$w \in L(R_{01}) \cap K \;\Rightarrow\; \varphi(w) \in L(R_{\mathrm{uV}}) \cap \{\mathrm{u}\}^* \;\Rightarrow\; (\exists n \geq 0)\ |w| = 2 \cdot 3^n .$$

This, together with the pumping lemma for context-free languages and the infiniteness of $L(R_{01}) \cap K$, implies that $L(R_{01}) \cap K$ is not a context-free language. As the class of context-free languages is closed under intersections with regular languages, $L(R_{01})$ is not context-free either.

4.1 Proof of Lemma 8

We should show that $w \in L(R_{\mathrm{uV}}) \cap \{\mathrm{u}\}^*$ implies $|w| = 2 \cdot 3^n$ for some $n \geq 0$. Let $\Phi : \{\mathrm{u}, \mathrm{V}\}^* \to \mathbb{N}$ be defined inductively as follows:

$$\begin{aligned} \Phi(\epsilon) &= 0, \\ \Phi(\mathrm{u}^k w) &= k + \Phi(w)\,, \\ \Phi(\mathrm{V}w) &= 1 + 3 \cdot \Phi(w) \end{aligned}$$

for each $k \geq 1$ and $w \in \{\mathrm{u}, \mathrm{V}\}^*$. Observe that we have assigned a unique value of Φ to each word from $\{\mathrm{u}, \mathrm{V}\}^*$. Next, we describe effects of the rules of R_{uV} to the value of Φ.

(0) The rule 0 can only rewrite $w_1 = \epsilon$ to $w_2 = \mathrm{uu}$. We have $\Phi(w_1) = 0$ and $\Phi(w_2) = 2$.
(1) The rule 1 rewrites $w_1 = \mathrm{u}w$ to $w_2 = \mathrm{uuV}w$ for some $w \in \{\mathrm{u}, \mathrm{V}\}^*$. We have $\Phi(w_1) = 1 + \Phi(w)$ and $\Phi(w_2) = 3 + 3 \cdot \Phi(w)$. Thus, $\Phi(w_2) = 3 \cdot \Phi(w_1)$.
(2) The rule 2 rewrites $w_1 = \overline{w}\mathrm{Vu}w$ to $w_2 = \overline{w}\mathrm{uuuV}w$ for some $w, \overline{w} \in \{\mathrm{u}, \mathrm{V}\}^*$. We have

$$\Phi(\mathrm{Vu}w) = \Phi(\mathrm{uuuV}w) = 4 + 3 \cdot \Phi(w)\,.$$

It follows that $\Phi(w_1) = \Phi(w_2)$.
(3) The rule 3 rewrites $w_1 = \overline{w}\mathrm{Vu}$ to $w_2 = \overline{w}\mathrm{uuuu}$ for some $\overline{w} \in \{\mathrm{u}, \mathrm{V}\}^*$. We have $\Phi(\mathrm{Vu}) = \Phi(\mathrm{uuuu}) = 4$ and thus $\Phi(w_1) = \Phi(w_2)$.

Together, each $w \in L(R_{\mathrm{uV}})$ has $\Phi(w) = 2 \cdot 3^n$ for some $n \geq 0$. As $\Phi(w) = |w|$ for each $w \in \{\mathrm{u}\}^*$, the proof is complete. □

4.2 Proof of Lemma 12

We should prove that

$$00\,(1100)^{\alpha}\,10\,(0011)^{\beta}\,00 \dashv_{01}^{*} 00\,(1100)^{\alpha+9}\,10\,(0011)^{\beta-1}\,00$$

for $\alpha \geq 0, \beta \geq 1$. Let $p = 00\,(1100)^{\alpha}$, $q = (0011)^{\beta-1}\,00$, and derive the claim as follows:

$$\begin{aligned} p10\,(0011)\,q &\dashv_{\mathrm{b}} p100\underline{11}011q & \dashv_{\mathrm{a}} \\ p1\underline{10}0011011q &\dashv_{\mathrm{b}} p\,(1100)\,\underline{11}011011q & \dashv_{\mathrm{d}} \\ p\,(1100)\,110\underline{01}11011q &\dashv_{\mathrm{d}} p\,(1100)^2\,1110\underline{01}11q & \dashv_{\mathrm{c}} \\ p\,(1100)^2\,11\underline{00}100111q &\dashv_{\mathrm{a}} p\,(1100)^3\,1\underline{10}00111q & \dashv_{\mathrm{b}} \\ p\,(1100)^4\,\underline{11}0111q &\dashv_{\mathrm{c}} p\,(1100)^4\,11011\underline{00}1q & \dashv_{\mathrm{d}} \end{aligned}$$

$$\begin{array}{llll} p\,(1100)^4\,110\underline{01}11001q & \dashv_{\rm c} & p\,(1100)^5\,11\underline{00}1001q & \dashv_{\rm a} \\ p\,(1100)^6\,1\underline{10}001q & \dashv_{\rm a} & p\,(1100)^7\,01\underline{10}q & \dashv_{\rm b} \\ p\,(1100)^7\,\underline{11}0110q & \dashv_{\rm d} & p\,(1100)^7\,110\underline{01}110q & \dashv_{\rm c} \\ p\,(1100)^8\,11\underline{00}10q, & & & \end{array}$$

where uses of particular rules of the type 2 are indicated by typing $\dashv_{\rm a}, \dashv_{\rm b}, \dashv_{\rm c}, \dashv_{\rm d}$ instead of $\dashv_{01}$. □

5 Conclusions and Remarks

We made a progress in studying basic properties of two recently introduced formalisms. Even if these particular models do not find application in practice, our results may be of key importance for designing suitable modifications.

The maximum length of labels is a key property of a GJFA. It remains open whether our undecidability results hold if restricted to GJFA with labels of a fixed maximum length. In *jumping finite automata*, i.e., GJFA with labels of length one, the problems become decidable (see [5] for a thorough survey).

Note that there is a group of older models that can be, in fact, put to a common framework with GJFA and cl-RA, immediately sharing some properties following from our new results:

- *Insertion systems* [16] were introduced in the scope of DNA computing. They generate sequences by inserting factors according to contexts of restricted lengths. Their generalization to *graph-controlled* [1] insertion systems together with contexts of zero length corresponds to the expressive power of GJFA. Using the notation of [1], we have $\mathrm{LStP}_*\left(\mathrm{ins}_*^{0,0}\right) = \mathbf{GJFA}$. Another (historical) work introduces *regular control semi-contextual grammars without appearance checking* [11]. Again, the variant with forbidden contexts (with a language class denoted by $\mathcal{C}_0$) is equivalent to GJFA. Our results imply that universality, inclusion, and equivalence are undecidable for these models as well.
- Up to explicit endmarking, insertion systems and the basic variant of semi-contextual grammars [15], both with contexts bounded by some $k \geq 1$, are equivalent to k-cl-RA. More precisely, each language from the class denoted by INS_*^k or $\mathcal{J}_k$ is accepted by a k-cl-RA, while for each k-cl-RA M, the language $\cent L(M)\,\$$ lies in $\mathrm{INS}_*^k = \mathcal{J}_k$. Thus, we can conclude that the class $\mathrm{INS}_*^2 = \mathcal{J}_2$ contains non-context-free binary languages.

The remarks above are hard to present in more depth because the original definitions of insertions systems and semi-contextual grammars use non-compatible notational paradigms. Once these definitions are understood, the claims are very easy to check (see [17]).

References

1. Alhazov, A., Krassovitskiy, A., Rogozhin, Y., Verlan, S.: Small size insertion and deletion systems. In: Martin-Vide, C. (ed.) Scientific Applications of Language Methods, pp. 459–524. Imperial College Press (2010)

2. Černo, P.: Clearing restarting automata and grammatical inference. In: Heinz, J., Colin de la Higuera, T.O. (eds.) Proceedings of the Eleventh International Conference on Grammatical Inference. JMLR Workshop and Conference Proceedings, vol. 21, pp. 54–68 (2012)
3. Černo, P., Mráz, F.: Clearing restarting automata. Fund. Inf. **104**(1), 17–54 (2010)
4. Ehrenfeucht, A., Haussler, D., Rozenberg, G.: On regularity of context-free languages. Theoret. Comput. Sci. **27**(3), 311–332 (1983)
5. Fernau, H., Paramasivan, M., Schmid, M., Vorel, V.: Characterization andcomplexity results on jumping finite automata. Accepted to Theoretical Computer Science (2015). http://arxiv.org/abs/1512.00482
6. Fernau, H., Paramasivan, M., Schmid, M.L.: Jumping finite automata: characterizations and complexity. In: Drewes, F. (ed.) CIAA 2015. LNCS, vol. 9223, pp. 89–101. Springer, Heidelberg (2015)
7. Haussler, D.: Insertion languages. Inf. Sci. **31**(1), 77–89 (1983)
8. Hopcroft, J.E., Motwani, R., Ullman, J.D.: Introduction to Automata Theory, Languages, and Computation, 2nd edn. Addison-Wesley (2003)
9. Ito, M., Kari, L., Thierrin, G.: Insertion and deletion closure of languages. Theoret. Comput. Sci. **183**(1), 3–19 (1997)
10. Jančar, P., Mráz, F., Plátek, M., Vogel, J.: Restarting automata. In: Reichel, H. (ed.) FCT 1995. LNCS, vol. 965, pp. 283–292. Springer, Heidelberg (1995)
11. Marcus, M., Păun, G.: Regulated Galiukschov semicontextual grammars. Kybernetika **26**(4), 316–326 (1990)
12. Meduna, A., Zemek, P.: Jumping finite automata. Int. J. Found. Comput. Sci. **23**(7), 1555–1578 (2012)
13. Meduna, A., Zemek, P.: Chapter 17 Jumping Finite Automata. In: Regulated Grammars and Automata, pp. 567–585. Springer, New York (2014)
14. Mráz, F., Plátek, M., Vogel, J.: Restarting automata with rewriting. In: Král, J., Bartosek, M., Jeffery, K. (eds.) SOFSEM 1996. LNCS, vol. 1175, pp. 401–408. Springer, Heidelberg (1996)
15. Păun, G.: Two theorems about Galiukschov semicontextual languages. Kybernetika **21**(5), 360–365 (1985)
16. Păun, G., Rozenberg, G., Salomaa, A.: Insertion-deletion systems. In: DNA Computing: New Computing Paradigms. Texts in Theoretical Computer Science, pp. 187–215. Springer, Heidelberg (1998)
17. Vorel, V.: On basic properties of jumping finite automata. Int. J. Found. Comput. Sci., conditionally accepted in 2015. http://arxiv.org/abs/1511.08396