On CD-Systems of Stateless Deterministic Two-Phase RR(1)-Automata

Martin Kutrib[1] and Friedrich Otto[2]

[1] Institut für Informatik, Universität Giessen,
Arndtstr. 2, 35392 Giessen, Germany
kutrib@informatik.uni-giessen.de
[2] Fachbereich Elektrotechnik/Informatik, Universität Kassel,
34109 Kassel, Germany
otto@theory.informatik.uni-kassel.de

Abstract. We study stateless deterministic two-phase RR-automata of window size one: stl-det-2-RR(1)-automata. While general deterministic RR-automata of window size one characterize the regular languages, it turns out that the class of languages accepted by the stateless two-phase variants is subregular. Therefore we combine stl-det-2-RR(1)-automata into computationally stronger cooperating distributed systems, obtaining the stl-det-local-CD-2-RR(1)-systems. By limiting their inherent nondeterminism, two further variants are derived. The relations between the different classes and some well-known language families are investigated, and it is shown that the classes defined here form a finite hierarchy whose levels are incomparable to several well-known language families. Further, closure properties and decision problems are studied for these classes.

1 Introduction

One of the fundamental concepts of computing models and automata is that of internal states which evolve at discrete time steps. Accordingly, the number of these states can be seen as a parameter of such systems. By reducing this number as much as possible, we obtain types of automata that only have a single internal state. Thus, the behavior of these automata does not depend on their internal state at all and, therefore, these devices are called *stateless*. It is easily seen that the computational power of *stateless* finite automata is strictly weaker than that of general finite automata. On the other hand, it is well known that already stateless nondeterministic pushdown automata accept all context-free languages [5]. Thus, for nondeterministic pushdown automata, the resource 'pushdown store' can compensate for the absence of states. Generally speaking, it is a natural and interesting question of how resources given to finite automata relate to the absence or presence of internal states. Given some computational model, are states necessary at all?

Inspired by biologically motivated models of computing related studies were initiated in [6,18], as it is difficult and even unrealistic to maintain a global state for a massively parallel group of objects appearing in natural phenomena of cell

H. Bordihn, M. Kutrib, and B. Truthe (Eds.): Dassow Festschrift 2012, LNCS 7300, pp. 111–137, 2012.

evolutions and chemical reactions. The study of stateless multi-head finite automata and stateless multi-counter systems in [18] and the successor paper [6] shows that the resource 'heads' cannot compensate for the absence of states. Recently, also stateless two-pushdown automata have been investigated [7], and it has been shown that for shrinking as well as for length-reducing deterministic and nondeterministic two-pushdown automata states are not needed. Further, also stateless variants of restarting automata have been studied. In [7] so-called R-automata with combined rewrite/restart operations are considered, while in [8] restarting automata which, after executing a rewrite step may continue to read their tape before performing a restart, so-called RR-automata, are of main interest. Thus, even after executing a rewrite step an RR-automaton has still the option to accept or to reject instead of performing a restart. In particular, in [8] the *two-phase* RR-*automaton* has been introduced, which is a stateless RR-automaton that can distinguish between the two parts of each cycle: the first part, which ends with an application of a rewrite (that is, delete) operation, and the second part, which ends with an execution of a restart operation.

Here we study the influence of the size k of the read/write window on the expressive power of *stateless deterministic two-phase* RR-*automata*, abbreviated as stl-det-2-RR(k)-automata. We will see that based on the size k, we obtain an infinite strict hierarchy of language classes that, however, are incomparable to the class REG of regular languages with respect to inclusion. In particular, it turns out that the class of languages accepted by the stateless two-phase RR-automata of window size one is subregular, while general deterministic RR-automata of window size one characterize the regular languages [9].

Then, in analogy to the work presented in [14,15] we introduce *cooperating distributed systems* (CD-systems) of stl-det-2-RR(1)-automata, the so-called stl-det-local-CD-2-RR(1)-systems. These systems are an adaptation of the notion of *cooperating distributed grammar system* with external control (see, for example, [1,3]) to the setting of stl-det-2-RR(1)-automata. As it turns out these systems are strictly more expressive than the CD-systems of stateless deterministic R(1)-automata (the so-called stl-det-local-CD-R(1)-systems) studied in [14]. On the other hand, the class of languages $\mathscr{L}_{=1}$(stl-det-local-2-RR(1)) accepted by the stl-det-local-CD-2-RR(1)-systems is incomparable under inclusion to the classes of (deterministic) context-free languages, linear languages, Church-Rosser languages and growing context-sensitive languages.

Although all the component automata of a stl-det-local-CD-2-RR(1)-system are deterministic, the system itself is not. Therefore, also two types of *deterministic* CD-systems of stl-det-2-RR(1)-automata are defined: the *strictly deterministic* CD-systems and the *globally deterministic* CD-systems. We compare the resulting classes of languages to each other and to the class of regular languages, and we establish closure and non-closure properties for them.

The paper is organized as follows. First we describe in short the two-phase restarting automaton and derive a few fundamental results on them. In Section 3, CD-systems of stateless deterministic 2-RR(1)-automata are introduced and investigated. Then the two variants without nondeterminism are defined and

studied in Section 4. It turns out that the strictly deterministic CD-systems define a language class that forms a non-reversal and non-intersection closed anti-AFL, which is quite surprising for a deterministic automaton model. Although anti-AFLs are sometimes referred to as "unfortunate families of languages," there is linguistical evidence that such language families might be of crucial importance, since in [2] it was shown that the family of natural languages is an anti-AFL. Decidability problems are the main aspect of Section 5. The results on the relations between the different language classes are summarized in Figure 1, and Table 1 summarizes the closure and non-closure properties. Finally, we conclude and present some open and untouched questions in Section 6.

2 Two-Phase Restarting Automata

A *stateless deterministic two-phase* RR-*automaton*, stl-det-2-RR-automaton for short, is described by a 6-tuple $M = (\Sigma, ¢, \$, k, \delta_1, \delta_2)$, where Σ is a finite input alphabet, $¢$ and $\$$ are additional symbols that serve as markers for the left and right border of the input tape, $k \geq 1$ is the size of the read/write window, and δ_1 and δ_2 are the transition functions that associate a *transition step* to each possible content u of the window. There are four types of transition steps: A *move-right step* (MVR) causes M to shift the window one position to the right. However, the window cannot be shifted beyond the right border marker $\$$. A *rewrite step* causes M to delete at least one and at most all symbols of the content u of the window, thereby replacing u by v and shortening the tape. Subsequently, the window is placed immediately to the right of v. Some additional restrictions apply in that the border markers $¢$ and $\$$ must not disappear from the tape. Hence, if u ends with the symbol $\$$, then so does v, and in this situation the window is placed on the $\$$. An *accept step* causes M to halt and accept, and a *restart step* causes M to place the window again over the left end of the tape, so that the first symbol it contains is the left border marker $¢$. If the transition step is undefined for the current situation, then M necessarily halts and rejects.

A computation of M consists of cycles followed by a tail computation. A *cycle* begins with the window scanning the left border marker. It consists of a sequence of MVR steps which is followed by a rewrite step that completes the first phase of the cycle. The behavior of M during the first phase is determined by δ_1. After the rewrite step, the second phase controlled by δ_2 starts. It consists of further MVR steps followed by a restart step that completes the cycle. A computation of M ends by a *tail computation*, which is an incomplete cycle ending with an accept step or a reject. Accept instructions can occur in both δ_1 and δ_2.

With M we associate two languages – the *simple language*

$$S(M) = \{\, w \in \Sigma^* \mid M \text{ accepts } w \text{ in a tail computation} \,\}$$

and the *language*

$$L(M) = \{\, w \in \Sigma^* \mid \exists z \in S(M) : w \vdash_M^{c^*} z \,\}$$

of words accepted by M. Here $\vdash_M^c$ denotes the reduction relation on Σ^* that is induced by the cycles of M. In order to clarify our notion we give a first short example.

Example 1. The non-regular language $\{ w \in \{a,b\}^* \mid |w|_a = |w|_b \}$ is accepted by the stateless deterministic two-phase RR-automaton $M = (\{a,b\}, ¢, \$, 2, \delta_1, \delta_2)$, where $\delta_1(¢\$) = \mathsf{Accept}$, $\delta_1(u) = \varepsilon$ for all $u \in \{ab, ba\}$, $\delta_1(u) = \mathsf{MVR}$ for all $u \in \{¢a, ¢b, aa, bb\}$, and $\delta_2(u) = \mathsf{Restart}$ for all $u \in \{ab, ba, aa, bb, a\$, b\$, \$\}$. □

In the example above, the stl-det-2-RR(2)-automaton restarts after a rewrite in any case. This particular behavior led to the definition of the so-called R-*automata* that cannot continue to read the input after a rewrite, that is, rewrite and restart steps are combined. Therefore, for these automata the transition function δ_2 can be omitted.

For each $k \geq 1$, stl-det-2-RR(k) denotes the class of stateless deterministic two-phase RR-automata with window of size k, and $\mathscr{L}$(stl-det-2-RR(k)) denotes the class of languages that are accepted by stl-det-2-RR(k)-automata. Similarly for R-automata. For devices with states it is evident that RR-automata are at least as powerful as R-automata. But this cannot be derived from the definition for stateless variants. Nevertheless, we have the following result.

Lemma 2. *For each $k \geq 1$ and each* stl-det-R(k)*-automaton M, there exists a* stl-det-2-RR(k)*-automaton M' such that the reduction relations $\vdash_M^c$ and $\vdash_{M'}^c$ coincide, $S(M) = S(M')$ and, thus, $L(M) = L(M')$.*

Proof. Let $M = (\Sigma, ¢, \$, k, \delta)$ be a stl-det-R(k)-automaton. We obtain a stl-det-2-RR(k)-automaton $M' = (\Sigma, ¢, \$, k, \delta_1, \delta_2)$ by taking $\delta_1 = \delta$ and $\delta_2(u) = \mathsf{Restart}$ for all u that can occur as the contents of the window of M. Then the cycles of M' and of M correspond to each other, and $S(M') = S(M)$ holds. □

Example 3. For $k \geq 1$ and $\Sigma = \{a,b\}$, we define the language $L_k = b^* \cdot (a^k \cdot b^+)^*$.

Claim. $L_k \in \mathscr{L}$(stl-det-2-RR(k)).

Proof (of claim). We define a stl-det-2-RR(k)-automaton $M_k = (\Sigma, ¢, \$, k, \delta_1, \delta_2)$ as follows:

$$\begin{aligned}
\delta_1(¢b^i\$) &= \mathsf{Accept}, \text{ for all } 0 \leq i \leq k-2,\\
\delta_1(¢b^i a^{k-1-i}) &= \mathsf{MVR}, \quad \text{for all } 0 \leq i \leq k-1,\\
\delta_1(b^i a^{k-i}) &= \mathsf{MVR}, \quad \text{for all } 1 \leq i \leq k,\\
\delta_1(a^k) &= \varepsilon,\\
\delta_1(b^{k-1}\$) &= \mathsf{Accept};\\
\delta_2(b^i a^{k-i}) &= \mathsf{Restart}, \text{ for all } 1 \leq i \leq k,\\
\delta_2(b^i\$) &= \mathsf{Restart}, \text{ for all } 1 \leq i \leq k-1.
\end{aligned}$$

Then $S(M) = b^*$, and $b^i a^k u \vdash_M^c b^i u$ for all $i \geq 0$ and all words u such that $u \in b^+$ or $u = b^r a^s u'$ for some $r, s \geq 1$ such that $r + s \geq k$. Thus, it is easily seen that $L(M) = L_k$ holds. □

Claim. $L_k \notin \mathscr{L}$(stl-det-R(k)).

Proof (of claim). Assume to the contrary that $M = (\Sigma, ¢, \$, k, \delta)$ is a stl-det-R(k)-automaton such that $L(M) = L_k$ holds. The word $w_1 = a^k b$ belongs to L_k, that is, M accepts on input w_1. Now we consider the function δ. Obviously, $\delta(¢a^{k-1})$ must be defined. It cannot be an accept instruction, and it cannot be a rewrite instruction, as the prefix a^{k-1} of w_1 cannot be replaced by any shorter word without obtaining a word that is not a member of L_k. Thus, it follows that $\delta(¢a^{k-1}) = \mathsf{MVR}$. If $\delta(a^k) = \mathsf{MVR}$, then we must consider $\delta(a^{k-1}b)$. The suffix $a^{k-1}b$ of w_1 cannot be replaced by a shorter word without obtaining a word that is not a member of L_k, and so it follows that $\delta(a^{k-1}b) = \mathsf{MVR}$. Finally, $\delta(a^{k-2}b\$)$ must be an accept instruction. However, then together with w_1, M also accepts the word $a^{k+1}b \notin L_k$. This contradiction shows that $\delta(a^k) = \varepsilon$ must hold. But then $a^k a^k b \vdash_M^c a^k b \vdash_M^* \mathsf{Accept}$, and M accepts $a^k a^k b \notin L_k$. □

Together with Lemma 2, Example 3 yields the following proper inclusions.

Corollary 4. *For all $k \geq 1$, $\mathscr{L}$(stl-det-R(k)) $\subsetneq \mathscr{L}$(stl-det-2-RR(k)).*

In [14] it is shown that the regular language $L'_k = \{ (ab^k)^i \mid i \geq 0 \}$ separates the language class $\mathscr{L}$(stl-det-R(k)) from the class $\mathscr{L}$(stl-det-R($k+1$)). From Lemma 2 we see that L'_k is also accepted by a stl-det-2-RR($k+1$)-automaton.

Lemma 5. *The language L'_k is not accepted by any* stl-det-2-RR(k)*-automaton.*

Proof. Assume to the contrary that $M = (\Sigma, ¢, \$, k, \delta_1, \delta_2)$ is a stl-det-2-RR(k)-automaton that accepts the language L'_k. Then on input $ab^k ab^k$, M will have to accept. However, as M has a window of size k only, it cannot accept the word $ab^k ab^k$ in a tail computation without accepting some word not belonging to L'_k. Hence, the accepting computation of M on input $ab^k ab^k$ begins with a cycle $ab^k ab^k \vdash_M^c z$. Then $|z| < 2k + 2$, and as M can delete at most k symbols in a single cycle, we have $|z| \geq k + 2$. This implies, however, that $z \notin L'_k$. So, M cannot accept $ab^k ab^k$ without accepting z as well. It follows that L'_k is not accepted by any stl-det-2-RR(k)-automaton. □

Recall from [9] that $\mathscr{L}$(det-RR(1)) coincides with the class of regular languages, and from Example 1 that $\mathscr{L}$(stl-det-2-RR(2)) includes a non-regular language. Thus, together with Lemma 5 this yields the following results.

Corollary 6. (a) *For all $k \geq 1$, $\mathscr{L}$(stl-det-2-RR(k)) $\subsetneq \mathscr{L}$(stl-det-2-RR($k+1$)).* (b) *The class $\mathscr{L}$(stl-det-2-RR(1)) is properly contained in the class* REG *of regular languages.* (c) *For all $k \geq 2$, the class $\mathscr{L}$(stl-det-2-RR(k)) is incomparable under inclusion to the class* REG.

3 CD-Systems of stl-det-2-RR(1)-Automata

Cooperating distributed systems (CD-systems) of restarting automata were introduced and studied in [12]. Here we study CD-systems of stateless deterministic 2-RR(1)-automata, comparing them in particular to the CD-systems of stateless deterministic R(1)-automata of [14].

A CD-system of stateless deterministic 2-RR(1)-automata consists of a finite collection $\mathcal{M} = ((M_i, \sigma_i)_{i \in I}, I_0)$ of stateless deterministic 2-RR(1)-automata $M_i = (\Sigma, ¢, \$, 1, \delta_1^{(i)}, \delta_2^{(i)})$ $(i \in I)$, *successor relations* $\sigma_i \subseteq I$ $(i \in I)$, and a subset $I_0 \subseteq I$ of *initial indices*. Here it is required that $I_0 \neq \emptyset$, and that $\sigma_i \neq \emptyset$ for all $i \in I$. For the CD-systems of stl-det-R(1)-automata introduced in [14] it was required in addition that $i \notin \sigma_i$ for all $i \in I$, but this requirement is easily met by using two isomorphic copies of each component automaton. Therefore, we abandon it here in order to simplify the presentation.

Various modes of operation have been introduced and studied for CD-systems of restarting automata, but here we are only interested in mode = 1 computations. A computation of $\mathcal{M}$ in mode = 1 on an input word w proceeds as follows. First an index $i_0 \in I_0$ is chosen nondeterministically. Then the 2-RR-automaton M_{i_0} starts the computation with the initial configuration $¢w\$$, and executes a single cycle. Thereafter an index $i_1 \in \sigma_{i_0}$ is chosen nondeterministically, and M_{i_1} continues the computation by executing a single cycle. This continues until, for some $l \geq 0$, the automaton M_{i_l} accepts. Such a computation will be denoted as $(i_0, w) \vdash_{\mathcal{M}}^c (i_1, w_1) \vdash_{\mathcal{M}}^c \cdots \vdash_{\mathcal{M}}^c (i_l, w_l) \vdash_{M_{i_l}}^* \mathsf{Accept}$. Should at some stage the chosen automaton M_{i_l} be unable to execute a cycle or to accept, then the computation fails. By $L_{=1}(\mathcal{M})$ we denote the language that the system $\mathcal{M}$ accepts in mode = 1, and by $\mathscr{L}_{=1}$(stl-det-local-CD-2-RR(1)) we denote the class of languages that are accepted by mode = 1 computations of stl-det-local-CD-2-RR(1)-systems, that is, by CD-systems of stateless deterministic 2-RR(1)-automata.

From Lemma 2 we immediately obtain that $\mathscr{L}_{=1}$(stl-det-local-CD-R(1)) is contained in $\mathscr{L}_{=1}$(stl-det-local-CD-2-RR(1)). Below we will see that this inclusion is actually a proper one.

Recall from [4] or from [14] that a language $L \subseteq \Sigma^*$ is called a *rational trace language* if there exists a reflexive and transitive binary relation D on Σ (a *dependency relation*) such that $L = \bigcup_{w \in R} [w]_D$ for some regular language R on Σ. Here $[w]_D$ denotes the congruence class of w with respect to the congruence $\equiv_D = \{ (uabv, ubav) \mid u, v \in \Sigma^*, a, b \in \Sigma, (a, b) \notin D \}$. In [14] it is shown that the stl-det-local-CD-R(1)-systems accept all rational trace languages. Thus, we see that also the stl-det-local-CD-2-RR(1)-systems accept all rational trace languages. Further, it is shown in [14] that one can extract a finite-state acceptor A from a stl-det-local-CD-R(1)-system $\mathcal{M}$ such that A accepts a sublanguage of $L_{=1}(\mathcal{M})$ that is letter-equivalent to $L_{=1}(\mathcal{M})$. Below we prove that this result does not carry over to stl-det-local-CD-2-RR(1)-systems.

Example 7. Let $\mathcal{M} = ((M_i, \sigma_i)_{i \in \{1,2\}}, \{1\})$ be the CD-system of stl-det-2-RR(1)-automata on $\Sigma = \{a, b\}$ that is specified by $\sigma_1 = \{2\}$, $\sigma_2 = \{1\}$, and

$$M_1 : \delta_1^{(1)} : ¢ \mapsto \mathsf{MVR},\ a \mapsto \varepsilon,\ \$ \mapsto \mathsf{Accept};\ \delta_2^{(1)} : a \mapsto \mathsf{Restart},\ b \mapsto \mathsf{Restart};$$
$$M_2 : \delta_1^{(2)} : ¢ \mapsto \mathsf{MVR},\ b \mapsto \varepsilon,\ a \mapsto \mathsf{MVR};\quad \delta_2^{(2)} : b \mapsto \mathsf{MVR},\quad \$ \mapsto \mathsf{Restart}.$$

Then $\mathcal{M}$ accepts the empty word. If $w \in \Sigma^+$ is accepted, then we see from the definition of $\mathcal{M}$ that $w = a^n b^m$ for some $n, m \geq 1$. In fact, as M_1

and M_2 alternate in every computation of $\mathcal{M}$, we have $n = m$ and, therefore, $L_{=1}(\mathcal{M}) = \{ a^n b^n \mid n \geq 0 \}$. □

Actually, the following stronger result can be derived.

Proposition 8. *For all $m \geq 1$,*

$$L_m = \{ a_1^n a_2^n \dots a_m^n \mid n \geq 0 \} \in \mathscr{L}_{=1}(\text{stl-det-local-CD-2-RR(1)}).$$

The language L_m ($m \geq 2$) does not contain a regular sublanguage that is letter-equivalent to L_m. It follows that this language is not accepted by any stl-det-local-CD-R(1)-systems. So we obtain the following proper inclusion.

Corollary 9. $\mathscr{L}_{=1}(\text{stl-det-local-CD-R(1)}) \subsetneq \mathscr{L}_{=1}(\text{stl-det-local-CD-2-RR(1)})$.

In order to determine the computational capacity of stl-det-local-CD-2-RR(1)-systems we continue with an example that shows that these systems accept a language that is not even growing context-sensitive.

Example 10. Let $\Sigma = \{a, b, \tilde{a}, \tilde{b}\}$. For any word $w = x_1 x_2 \cdots x_n \in \{a, b\}^*$, we set $\tilde{w} = \tilde{x}_1 \tilde{x}_2 \cdots \tilde{x}_n \in \{\tilde{a}, \tilde{b}\}^*$, and consider $L_{tc} = \{ aw\tilde{a}\tilde{w} \mid w \in \{a, b\}^* \}$ over Σ.

The language L_{tc} is not growing context-sensitive, as the growing context-sensitive languages are closed under union and ε-free homomorphisms, and the copy language is not growing context-sensitive [10]. However, it is accepted by the stl-det-local-CD-2-RR(1)-system $\mathcal{M} = ((M_i, \sigma_i)_{i \in \{0,1,2,3,4\}}, \{0\})$ that is specified by $\sigma_0 = \{1\}$, $\sigma_1 = \{0, 2, 4\}$, $\sigma_2 = \{3\}$, $\sigma_3 = \{0, 2, 4\}$, $\sigma_4 = \{4\}$, and

$$\begin{aligned}
&\delta_1^{(0)} : ¢ \mapsto \mathsf{MVR}, \quad a \mapsto \varepsilon;\\
&\delta_2^{(0)} : a \mapsto \mathsf{Restart}, \quad b \mapsto \mathsf{Restart}, \quad \tilde{a} \mapsto \mathsf{Restart};\\
&\delta_1^{(1)} : ¢ \mapsto \mathsf{MVR}, \quad a \mapsto \mathsf{MVR}, \quad b \mapsto \mathsf{MVR}, \quad \tilde{a} \mapsto \varepsilon;\\
&\delta_2^{(1)} : \tilde{a} \mapsto \mathsf{Restart}, \quad \tilde{b} \mapsto \mathsf{Restart}, \quad \$ \mapsto \mathsf{Restart};\\
&\delta_1^{(2)} : ¢ \mapsto \mathsf{MVR}, \quad b \mapsto \varepsilon;\\
&\delta_2^{(2)} : a \mapsto \mathsf{Restart}, \quad b \mapsto \mathsf{Restart}, \quad \tilde{b} \mapsto \mathsf{Restart};\\
&\delta_1^{(3)} : ¢ \mapsto \mathsf{MVR}, \quad a \mapsto \mathsf{MVR}, \quad b \mapsto \mathsf{MVR}, \quad \tilde{b} \mapsto \varepsilon;\\
&\delta_2^{(3)} : \tilde{a} \mapsto \mathsf{Restart}, \quad \tilde{b} \mapsto \mathsf{Restart}, \quad \$ \mapsto \mathsf{Restart};\\
&\delta_1^{(4)} : ¢ \mapsto \mathsf{MVR}, \quad \$ \mapsto \mathsf{Accept}.
\end{aligned}$$

Initially, component 0 deletes the first input symbol if it is an a, otherwise the input is rejected. Then component 1 searches for the first occurrence of an input letter from $\{\tilde{a}, \tilde{b}\}$. It is deleted if it is $\tilde{a}$, otherwise the input is rejected. In subsequent cycles corresponding symbols a and $\tilde{a}$ or b and $\tilde{b}$ are deleted by the components 0 and 1 or 2 and 3. After deleting an a, component 0 rejects if the next input symbol is $\tilde{b}$ or \$. In all other cases it restarts. The following component 1 deletes the first occurrence of an input letter from $\{\tilde{a}, \tilde{b}\}$ if it is $\tilde{a}$, otherwise the input is rejected. Moreover, component 1 restarts only if the deleted symbol is followed by another symbol from $\{\tilde{a}, \tilde{b}\}$ or by \$. Similarly, for

the components 2 and 3, and b and $\tilde{b}$. Whenever a pair of corresponding symbols has been deleted, system $\mathcal{M}$ guesses of which type the next pair is, or whether all pairs have been deleted. In the first case either component 0 or 2 is chosen to continue the computation. In the latter case, component 4 is used to verify that in fact all symbols have been deleted. Only in this situation it accepts. It follows that $L_{=1}(\mathcal{M}) = L_{tc}$. □

The power of stl-det-local-CD-2-RR(1)-systems is only deployed for languages over an alphabet with at least two symbols. For unary languages we have the following characterization.

Theorem 11. *A language $L \subseteq \{a\}^*$ is accepted by a* stl-det-local-CD-2-RR(1)*-system if and only if it is regular.*

Proof. As already stl-det-local-CD-R(1)-systems accept all regular languages [14], we see from Corollary 9 that the implication from right to left holds. To prove the reverse implication let $\mathcal{M} = ((M_i, \sigma_i)_{i \in I}, I_0)$ be a CD-system of stateless deterministic 2-RR(1)-automata on $\Sigma = \{a\}$. For all $i \in I$, if $\delta_1^{(i)}(¢)$ is undefined, then each computation of $\mathcal{M}$ that activates M_i fails, and if $\delta_1^{(i)}(¢) = \mathsf{Accept}$, then each computation of $\mathcal{M}$ that activates M_i accepts. Thus, in the former case M_i can be seen as a trap "state," while in the latter case it can be seen as an accepting "state" that keeps on digesting a's. Now assume that $\delta_1^{(i)}(¢) = \mathsf{MVR}$. If also $\delta_1^{(i)}(a) = \mathsf{MVR}$, then M_i can only execute tail computations. In fact, either M_i accepts all words from Σ^* in tail computations, and this is the case if $\delta_1^{(i)}(\$) = \mathsf{Accept}$, or it rejects all words from Σ^* in tail computations, and this is the case if $\delta_1^{(i)}(\$)$ is undefined. Also if $\delta_1^{(i)}(a) = \varepsilon$ and $\delta_2^{(i)}(a) \neq \mathsf{Restart}$ and $\delta_2^{(i)}(\$) \neq \mathsf{Restart}$, then M_i can only execute tail computations. Hence, again M_i can be seen as a trap "state" or as an accepting "state."

Now we can construct a finite-state acceptor $A = (Q, \Sigma, S, F, \delta_A)$ from $\mathcal{M}$ that accepts the language $L = L_{=1}(\mathcal{M})$. Essentially the states of A correspond to the component automata M_i of $\mathcal{M}$, with certain component automata becoming trap states and others becoming accepting states. For each $i \in I$, if M_i can execute a cycle (that is, $\delta_1^{(i)}(¢) = \mathsf{MVR}$, $\delta_1^{(i)}(a) = \varepsilon$, and $\delta_2^{(i)}(a) = \mathsf{Restart}$ or $\delta_2^{(i)}(a) = \mathsf{MVR}$ and $\delta_2^{(i)}(\$) = \mathsf{Restart}$), then A has an a-transition from the state corresponding to M_i to all states that correspond to component automata M_j with $j \in \sigma_i$. It is now easy to set up the transition relation δ_A in such a way that $L(A) = L_{=1}(\mathcal{M})$ holds. □

Since, for example, the unary language $\{ a^{2^n} \mid n \geq 0 \}$ belongs to the class of Church-Rosser languages [11], which in turn is a proper subset of the growing context-sensitive languages, we obtain the following incomparability results.

Corollary 12. *The language class $\mathscr{L}_{=1}$(*stl-det-local-CD-2-RR(1)*) is incomparable under inclusion to the classes* CRL *of Church-Rosser languages and* GCSL *of growing context-sensitive languages.*

Now we turn to consider the closure properties of the language class $\mathscr{L}_{=1}(\mathsf{stl\text{-}det\text{-}local\text{-}CD\text{-}2\text{-}RR(1)})$. Closure under certain operations indicates a certain robustness of the language families considered, while non-closure properties may serve, for example, as a valuable basis for extensions. We start to explore the closure properties under the Boolean operations union, intersection, and complementation. The first result is immediate.

Lemma 13. *The class* $\mathscr{L}_{=1}(\mathsf{stl\text{-}det\text{-}local\text{-}CD\text{-}2\text{-}RR(1)})$ *is closed under union.*

Proof. Given two $\mathsf{stl\text{-}det\text{-}local\text{-}CD\text{-}2\text{-}RR(1)}$-systems $\mathcal{M} = ((M_i, \sigma_i)_{i\in I}, I_0)$ and $\mathcal{M}' = ((M'_i, \sigma'_i)_{i\in I'}, I'_0)$, we can assume without loss of generality that I and I' are disjoint. So, it suffices to construct a new $\mathsf{stl\text{-}det\text{-}local\text{-}CD\text{-}2\text{-}RR(1)}$-system $\mathcal{M}'' = ((M_i, \sigma_i)_{i\in I\cup I'}, I_0 \cup I'_0)$ that consists of the components of $\mathcal{M}$ and $\mathcal{M}'$. Initially, $\mathcal{M}''$ guesses a starting component from the union $I_0 \cup I'_0$, that is, whether to simulate $\mathcal{M}$ or $\mathcal{M}'$. □

In order to show non-closure under intersection with regular sets we give the following example.

Example 14. Let D_1 denote the Dyck language on $\Sigma = \{a, b\}$, $\varphi : \Sigma^* \to \Sigma^*$ be the homomorphism that is induced by $a \mapsto a$ and $b \mapsto ba$, and $D_\varphi = \varphi(D_1)$. Then $w \in \Sigma^+$ belongs to D_φ if and only if $w \in \{a, ba\}^+$ and there exists an $n \geq 1$ such that $(w = a^n baz) \wedge (a^{n-1}z \in D_\varphi)$.

The $\mathsf{stl\text{-}det\text{-}local\text{-}CD\text{-}2\text{-}RR(1)}$-system $\mathcal{M} = ((M_i, \sigma_i)_{i\in\{1,2,3\}}, \{1\})$ is specified by $\sigma_1 = \{2\}$, $\sigma_2 = \{3\}$, $\sigma_3 = \{1\}$, and

$$
\begin{array}{lll}
M_1 : \delta_1^{(1)}(¢) = \mathsf{MVR}, & M_2 : \delta_1^{(2)}(¢) = \mathsf{MVR}, & M_3 : \delta_1^{(3)}(¢) = \mathsf{MVR}, \\
\quad \delta_1^{(1)}(a) = \varepsilon, & \quad \delta_1^{(2)}(a) = \mathsf{MVR}, & \quad \delta_1^{(3)}(a) = \varepsilon; \\
\quad \delta_1^{(1)}(\$) = \mathsf{Accept}; & \quad \delta_1^{(2)}(b) = \varepsilon; & \quad \delta_2^{(3)}(a) = \mathsf{Restart}, \\
\quad \delta_2^{(1)}(a) = \mathsf{Restart}, & \quad \delta_2^{(2)}(a) = \mathsf{Restart}; & \quad \delta_2^{(3)}(b) = \mathsf{Restart}, \\
\quad \delta_2^{(1)}(b) = \mathsf{Restart}; & & \quad \delta_2^{(3)}(\$) = \mathsf{Restart}.
\end{array}
$$

Obviously, $\mathcal{M}$ accepts on input ε. Now let $w = a^n baz$ such that $a^{n-1}z \in D_\varphi$. Then on input w, $\mathcal{M}$ proceeds as follows:

$$(1, w) = (1, a^n baz) \vdash^c_{M_1} (2, a^{n-1}baz) \vdash^c_{M_2} (3, a^{n-1}az) \vdash^c_{M_3} (1, a^{n-1}z).$$

By induction it follows that $\mathcal{M}$ accepts input $a^{n-1}z$, which shows $w \in L_{=1}(\mathcal{M})$. Thus, $D_\varphi \subseteq L_{=1}(\mathcal{M})$.

Conversely, assume that $w \in L_{=1}(\mathcal{M})$. If $w = \varepsilon$, then $w \in D_\varphi$. Otherwise, the accepting computation of $\mathcal{M}$ on input w looks as follows:

$$(1, w) \vdash^c_{M_1} (2, w_1) \vdash^c_{M_2} (3, w_2) \vdash^c_{M_3} (1, w_3) \vdash^*_{\mathcal{M}} \mathsf{Accept},$$

where $w = aw_1$ for $w_1 \neq \varepsilon$, $w_1 = a^m baz$ for some $z \in \Sigma^*$, $w_2 = a^m az$, and $w_3 = a^m z \in L_{=1}(\mathcal{M})$. By induction it follows that $a^m z \in D_\varphi$, which implies that $w = aw_1 = a^{m+1}baz$ belongs to D_φ. Thus, $L_{=1}(\mathcal{M}) = D_\varphi$ holds. □

Theorem 15. *The class $\mathscr{L}_{=1}$(stl-det-local-CD-2-RR(1)) is not closed under intersection (with regular sets), complementation, and ε-free homomorphisms.*

Proof. By Example 14, the language D_φ is accepted by a stl-det-local-CD-2-RR(1)-system. We take $R = a^* \cdot (ba)^*$ and show that the intersection $D_\varphi \cap R$ does not belong to the class $\mathscr{L}_{=1}$(stl-det-local-CD-2-RR(1)).

Claim. $D_\varphi \cap R \notin \mathscr{L}_{=1}$(stl-det-local-CD-2-RR(1)).

Proof (of claim). Assume that $\mathcal{M} = ((M_i, \sigma_i)_{i\in I}, I_0)$ is a stl-det-local-CD-2-RR(1)-system such that $L_{=1}(\mathcal{M}) = D_\varphi \cap R$. Then, for each $n \geq 1$, $\mathcal{M}$ has an accepting computation on input $w_{n+1} = a^{n+1}(ba)^{n+1}$. Let $n > |I|$, and let

$$(i_0, w_{n+1}) \vdash^c_{M_{i_0}} (i_1, z_1) \vdash^c_{M_{i_1}} \cdots \vdash^c_{M_{i_{m-1}}} (i_m, z_m) \vdash^*_{M_{i_m}} \mathsf{Accept}$$

be an accepting computation of $\mathcal{M}$ on input w_{n+1}. We now analyze this computation. Assume that there exists an index $k < n$ such that

$$(i_0, w_{n+1}) \vdash^{c^k}_{\mathcal{M}} (i_k, a^{n+1-k}(ba)^{n+1}) \vdash^c_{M_{i_k}} (i_{k+1}, a^{n+1-k}a(ba)^n) \vdash^*_{\mathcal{M}} \mathsf{Accept}$$

holds, that is, in each of the first $k < n$ cycles, an occurrence of the letter a is deleted, while in the $(k+1)$-st cycle the first occurrence of the letter b is deleted. Then $\mathcal{M}$ would also perform the following computation:

$$(i_0, a^n baa(ba)^n) \vdash^{c^k}_{\mathcal{M}} (i_k, a^{n-k}baa(ba)^n) \vdash^c_{M_{i_k}} (i_{k+1}, a^{n-k}aa(ba)^n) \vdash^*_{\mathcal{M}} \mathsf{Accept},$$

which shows that $\mathcal{M}$ accepts on input $a^n baa(ba)^n$ as well. However, since $a^n baa(ba)^n \notin D_\varphi \cap R$, this contradicts our assumption on $\mathcal{M}$.

Thus, during the first k cycles, in the accepting computation above the prefix a^k is deleted, for a $k \geq n$. As $n > |I|$, this means that there exist integers j and $\ell > 0$ such that $j + \ell \leq n$ and $i_j = i_{j+\ell}$. Hence, the accepting computation above has the following form:

$$(i_0, w_{n+1}) \vdash^{c^j}_{\mathcal{M}} (i_j, a^{n+1-j}(ba)^{n+1}) \vdash^{c^\ell}_{\mathcal{M}} (i_j, a^{n+1-j-\ell}(ba)^{n+1}) \vdash^*_{\mathcal{M}} \mathsf{Accept}.$$

But then $\mathcal{M}$ will also execute the following accepting computation:

$$(i_0, a^{n+1-\ell}(ba)^{n+1}) \vdash^{c^j}_{\mathcal{M}} (i_j, a^{n+1-j-\ell}(ba)^{n+1}) \vdash^*_{\mathcal{M}} \mathsf{Accept},$$

which shows that it accepts on input $a^{n+1-\ell}(ba)^{n+1} \notin D_\varphi \cap R$. Again this contradicts our assumption on $\mathcal{M}$. It follows that $D_\varphi \cap R$ is not accepted by any stl-det-local-CD-2-RR(1)-system working in mode $= 1$. □

So the class $\mathscr{L}_{=1}$(stl-det-local-CD-2-RR(1)) is not closed under intersection even with regular sets. By Lemma 13 it is closed under union. Since closure under complementation and union implies closure under intersection, it cannot be closed under complementation, either.

By Example 7, the language $L_2 = \{\, a^n b^n \mid n \geq 0 \,\}$ is accepted by a stl-det-local-CD-2-RR(1)-system. Let $h : \{a, b\}^* \to \{a, b\}^*$ be the ε-free homomorphism defined by $h(a) = a$ and $h(b) = ba$. Then $h(L_2) = D_\varphi \cap R$ does not belong to $\mathscr{L}_{=1}$(stl-det-local-CD-2-RR(1)), which shows the non-closure under ε-free homomorphisms. □

The proof of Theorem 15 together with Example 10 reveal further incomparabilities. Since the language $D_\varphi \cap R$ belongs to the intersection of deterministic and linear context-free languages, we have the following corollary.

Corollary 16. *The language class* $\mathscr{L}_{=1}$(stl-det-local-CD-2-RR(1)) *is incomparable under inclusion to the classes* CFL *of context-free languages,* LIN *of linear, and* DCFL *of deterministic context-free languages.*

We continue with further (non-)closure properties.

Example 17. Let $\Sigma = \{a, b, c, d\}$ and define the language

$$L_{dc} = \{\, wc^m dc^n \mid w \in \{a, b\}^*, m = |w|_a, n = |w|_b \,\}.$$

The language L_{dc} is accepted by the stl-det-local-CD-2-RR(1)-system $\mathcal{M} = ((M_i, \sigma_i)_{i\in\{0,1,2,3,4,5\}}, \{0, 2\})$ that is specified by $\sigma_0 = \{1\}$, $\sigma_1 = \{0, 2\}$, $\sigma_2 = \{3, 5\}$, $\sigma_3 = \{4\}$, $\sigma_4 = \{3, 5\}$, $\sigma_5 = \{5\}$, and

$$\begin{aligned}
&\delta_1^{(0)} : ¢ \mapsto \mathsf{MVR}, \quad b \mapsto \mathsf{MVR}, \quad a \mapsto \varepsilon;\\
&\delta_2^{(0)} : a \mapsto \mathsf{Restart}, \quad b \mapsto \mathsf{Restart}, \quad c \mapsto \mathsf{Restart};\\
&\delta_1^{(1)} : ¢ \mapsto \mathsf{MVR}, \quad a \mapsto \mathsf{MVR}, \quad b \mapsto \mathsf{MVR}, \quad c \mapsto \varepsilon;\\
&\delta_2^{(1)} : c \mapsto \mathsf{Restart}, \quad d \mapsto \mathsf{Restart};\\
&\delta_1^{(2)} : ¢ \mapsto \mathsf{MVR}, \quad b \mapsto \mathsf{MVR}, \quad d \mapsto \varepsilon;\\
&\delta_2^{(2)} : c \mapsto \mathsf{Restart}, \quad \$ \mapsto \mathsf{Restart};\\
&\delta_1^{(3)} : ¢ \mapsto \mathsf{MVR}, \quad b \mapsto \varepsilon;\\
&\delta_2^{(3)} : b \mapsto \mathsf{Restart}, \quad c \mapsto \mathsf{Restart};\\
&\delta_1^{(4)} : ¢ \mapsto \mathsf{MVR}, \quad b \mapsto \mathsf{MVR}, \quad c \mapsto \varepsilon;\\
&\delta_2^{(4)} : c \mapsto \mathsf{Restart}, \quad \$ \mapsto \mathsf{Restart};\\
&\delta_1^{(5)} : ¢ \mapsto \mathsf{MVR}, \quad \$ \mapsto \mathsf{Accept}.
\end{aligned}$$

Basically, the idea of the construction is that components 0 and 1 are used to delete one a from the prefix w and, subsequently, one c from the first block of c's. When all a's and c's have been deleted, component 2 is used to delete the sole symbol d. The input is rejected if component 2 sees an a or a c before reaching the d. Next, components 3 and 4 are used to delete successively the remaining b's from the prefix and the c's from the second block. Finally, component 5 checks that all symbols have been deleted. Only in this situation it accepts. □

Theorem 18. *The class* $\mathscr{L}_{=1}$(stl-det-local-CD-2-RR(1)) *is not closed under inverse homomorphisms.*

Proof. By Example 17, the language L_{dc} is accepted by a stl-det-local-CD-2-RR(1)-system. Let $h : \{a, c, d\}^* \to \{a, b, c, d\}^*$ be the homomorphism that is defined by $h(a) = ab$, $h(c) = c$, and $h(d) = d$. Then $h^{-1}(L_{dc}) = \{\, a^n c^n dc^n \mid n \geq 0 \,\}$.

Assume that $\mathcal{M} = ((M_i, \sigma_i)_{i\in I}, I_0)$ is a stl-det-local-CD-2-RR(1)-system such that $L_{=1}(\mathcal{M}) = h^{-1}(L_{dc})$.

First we note that in any accepting computation none of the c's following the sole d can be deleted as long as there is at least one c left before the d.

Let $n > |I|$. Clearly, $\mathcal{M}$ cannot accept the input $w_n = a^n c^n d c^n$ in a tail computation. So, there exist integers j and $\ell > 0$ with $j + \ell \leq n$ and $i_j = i_{j+\ell}$, and integers k_1, k_2, ℓ_1, ℓ_2 with $k_1 + \ell_1 + k_2 + \ell_2 = j + \ell$ such that, assuming that the sole d is not deleted during the first $j + \ell$ cycles, the accepting computation on w_n has the following form:

$$(i_0, w_n) \vdash_{\mathcal{M}}^{c^j} (i_j, a^{n-k_1} c^{n-k_2} d c^n) \vdash_{\mathcal{M}}^{c^\ell} (i_j, a^{n-k_1-\ell_1} c^{n-k_2-\ell_2} d c^n) \vdash_{\mathcal{M}}^{*} \mathsf{Accept}.$$

But then $\mathcal{M}$ will also execute the following accepting computation:

$$(i_0, a^{n-\ell_1} c^{n-\ell_2} d c^n) \vdash_{\mathcal{M}}^{c^j} (i_j, a^{n-k_1-\ell_1} c^{n-k_2-\ell_2} d c^n) \vdash_{\mathcal{M}}^{*} \mathsf{Accept},$$

which shows that it accepts the input $a^{n-\ell_1} c^{n-\ell_2} d c^n$ not belonging to $h^{-1}(L_{dc})$.

Now assume that the sole d is deleted during the first $j + \ell$ cycles. Then we obtain immediately a contradiction since the input $a^n c^{n+1} d c^{n-1} \notin h^{-1}(L_{dc})$ is accepted as well. It follows that $h^{-1}(L_{dc})$ is not accepted by any stl-det-local-CD-2-RR(1))-system, which proves the non-closure under inverse homomorphisms. □

Theorem 19. *$\mathscr{L}_{=1}$(stl-det-local-CD-2-RR(1)) is not closed under reversal.*

Proof. By Example 14, the language D_φ is accepted by a stl-det-local-CD-2-RR(1)-system. We show the theorem by proving that the reversal D_φ^R does not belong to $\mathscr{L}_{=1}$(stl-det-local-CD-2-RR(1)).

In contrast to the assertion assume that $\mathcal{M} = ((M_i, \sigma_i)_{i \in I}, I_0)$ is a stl-det-local-CD-2-RR(1)-system such that $L_{=1}(\mathcal{M}) = D_\varphi^R$. We consider accepting computations on inputs of the form $w_n = (ab)^n a^n$, for n large enough.

First we note that each component that deletes a symbol has to delete the leftmost occurrence of that symbol. Therefore, none of the components can delete an a from the suffix a^n as long as there is at least one a left in the prefix $(ab)^n$. Moreover, it is not hard to see that $\mathcal{M}$ cannot accept without deleting some a's from the suffix. Consider the tape inscription before the cycle in which the first symbol a from the suffix is deleted. It must be of the form $b^k a^n$, and k is determined by the prefix $(ab)^n$. Furthermore, for a fixed k, there are less than $|I|$ different values n such that $b^k a^n$ is the tape inscription in that situation. This implies that k is not bounded, that is, for any $k \geq 0$ we can find an n such that $(ab)^n a^n$ is transformed into $b^{k'} a^n$, where $k' \geq k$. We choose an n large enough such that k is large enough as well. Therefore, during the computation on the prefix there must occur two cycles in which the same component deletes an a such that the number of b's preceding the a is larger in the second cycle. More precisely, there exist integers j and $\ell > 0$ with $j + \ell \leq n$ and $i_j = i_{j+\ell}$, and integers $k_1 \geq 0$, k_2, m_1, m_2 with $m_2 - m_1 \geq 1$, $k_2 - m_2 - 1 \geq 0$ such that the accepting computation on $w_n = (ab)^n a^n$ has the following form:

$$(i_0, w_n) \vdash_{\mathcal{M}}^{c^j} (i_j, b^{k_1}(ab)^{k_2}a^n)$$
$$\vdash_{\mathcal{M}}^{c} (i_{j+1}, b^{k_1}b(ab)^{k_2-1}a^n) \vdash_{\mathcal{M}}^{c^{\ell-1}} (i_j, b^{k_1-m_1+m_2}(ab)^{k_2-m_2}a^n)$$
$$\vdash_{\mathcal{M}}^{c} (i'_{j+1}, b^{k_1-m_1+m_2}b(ab)^{k_2-m_2-1}a^n) \vdash_{\mathcal{M}}^{*} \mathsf{Accept}.$$

But then $\mathcal{M}$ will also execute the following accepting computation:

$$(i_0, (ab)^{n-k_2}abb^{m_2-m_1}(ab)^{k_2-m_2-1}a^n) \vdash_{\mathcal{M}}^{c^j} (i_j, b^{k_1}abb^{m_2-m_1}(ab)^{k_2-m_2-1}a^n)$$
$$\vdash_{\mathcal{M}}^{c} (i'_{j+1}, b^{k_1}bb^{m_2-m_1}(ab)^{k_2-m_2-1}a^n) \vdash_{\mathcal{M}}^{*} \mathsf{Accept},$$

which shows that it accepts the input $(ab)^{n-k_2+1}b^{m_2-m_1}(ab)^{k_2-m_2-1}a^n \notin D_{\varphi}^R$. It follows that D_{φ}^R is not accepted by any stl-det-local-CD-2-RR(1)-system. □

4 Deterministic CD-Systems of stl-det-2-RR(1)-Automata

Although all the component automata of a stl-det-local-CD-2-RR(1)-system are deterministic, the system itself is not. Indeed, the initial component with which to begin a particular computation is chosen nondeterministically from the set I_0 of all initial components, and after each cycle the component for executing the next cycle is chosen nondeterministically from among all the successors of the previously active component. Here we define two types of *deterministic* CD-systems of stl-det-2-RR(1)-automata: the *strictly deterministic* CD-systems and the *globally deterministic* CD-systems.

4.1 Strictly Deterministic CD-Systems of stl-det-2-RR(1)-Automata

Here we introduce and study a first type of CD-system of stateless deterministic 2-RR(1)-automata that is completely deterministic. The idea and the notation is taken from [13], where a corresponding notion was introduced for CD-systems of general restarting automata.

A CD-system $\mathcal{M} = ((M_i, \sigma_i)_{i\in I}, I_0)$ of stateless deterministic 2-RR(1)-automata is called *strictly deterministic* if $|I_0| = 1$ and $|\sigma_i| = 1$ for all $i \in I$. Then, for each word $w \in \Sigma^*$, $\mathcal{M}$ has a unique computation that begins with the initial configuration corresponding to input w. By $\mathscr{L}_{=1}$(stl-det-strict-CD-2-RR(1)) we denote the class of languages that are accepted by strictly deterministic stateless CD-2-RR(1)-systems. Note that the CD-systems in Examples 7, 14, and Proposition 8 are strictly deterministic. On the other hand, we have the following simple but useful observation on the weakness of stl-det-strict-CD-2-RR(1)-systems.

Lemma 20. *Let $\mathcal{M} = ((M_i, \sigma_i)_{i\in I}, \{i_0\})$ be a* stl-det-strict-CD-2-RR(1)*-system that accepts a language over the alphabet Σ, where $\delta_1^{(i_0)}(¢) = \mathsf{MVR}$. For all $w \in \Sigma^*$ and all $x, y \in \Sigma$, if $\delta_1^{(i_0)}(x) = \delta_1^{(i_0)}(y) = \varepsilon$, then $xw \in L_{=1}(\mathcal{M})$ if and only if $yw \in L_{=1}(\mathcal{M})$.*

Lemma 21. *The finite language $L_0 = \{aaa, bb\}$ is not accepted by any strictly deterministic stateless* CD-2-RR(1)*-system.*

Proof. Assume that $\mathcal{M} = ((M_i, \sigma_i)_{i\in I}, I_0)$ is a strictly deterministic stateless CD-2-RR(1)-system such that $L_{=1}(\mathcal{M}) = L_0$, and let $I_0 = \{i_0\}$. Since L_0 is neither $\{a, b\}^*$ nor empty, we have $\delta_1^{(i_0)}(¢) = \mathsf{MVR}$. Similarly, $L_0 \cap a^+$ is neither a^+ nor empty and, thus, we see that $\delta_1^{(i_0)}(a) = \varepsilon$. Analogously it follows that $\delta_1^{(i_0)}(b) = \varepsilon$. So we see from Lemma 20 that $aaa \in L_{=1}(\mathcal{M})$ if and only if $baa \in L_{=1}(\mathcal{M})$, a contradiction. □

We obtain the following consequences.

Corollary 22. $\mathscr{L}_{=1}$(stl-det-strict-CD-2-RR(1)) *is incomparable under inclusion to the language classes* FIN *of finite languages,* REG *of regular languages, and* CFL *of context-free languages. In particular, it follows that the inclusion* $\mathscr{L}_{=1}$(stl-det-strict-CD-2-RR(1)) $\subseteq$ $\mathscr{L}_{=1}$(stl-det-local-CD-2-RR(1)) *is proper.*

Further, we see that $\mathscr{L}_{=1}$(stl-det-strict-CD-2-RR(1)) is incomparable under inclusion to the language class $\mathscr{L}_{=1}$(stl-det-local-CD-R(1)). For future reference we consider another finite example language.

Lemma 23. *The finite language $L_0' = \{aaaa, abb\}$ is not accepted by any strictly deterministic stateless* CD-2-RR(1)*-system.*

Proof. Assume that $\mathcal{M} = ((M_i, \sigma_i)_{i\in I}, I_0)$ is a strictly deterministic stateless CD-2-RR(1)-system such that $L_{=1}(\mathcal{M}) = L_0'$, let $I_0 = \{i_0\}$, and let $\sigma_{i_0} = \{i_1\}$ and $\sigma_{i_1} = \{i_2\}$. Obviously, we have $\delta_1^{(i_0)}(¢) = \mathsf{MVR}$, and $\delta_1^{(i_0)}(a) = \varepsilon$. Further, it holds that $\delta_1^{(i_1)}(¢) = \mathsf{MVR}$, and $\delta_1^{(i_1)}(a) = \delta_1^{(i_1)}(b) = \varepsilon$. Now $(i_0, aaaa) \vdash_{\mathcal{M}}^c (i_1, aaa) \vdash_{\mathcal{M}}^c (i_2, aa)$, which leads to acceptance, while $(i_0, abaa) \vdash_{\mathcal{M}}^c (i_1, baa) \vdash_{\mathcal{M}}^c (i_2, aa)$ should lead to rejection, which is a contradiction. Thus, L_0' is not accepted by any strictly deterministic stateless CD-2-RR(1)-system working in mode = 1. □

From Lemma 21 we immediately obtain several non-closure properties for the class $\mathscr{L}_{=1}$(stl-det-strict-CD-2-RR(1)). In fact, we can derive the following result.

Theorem 24. *The language class* $\mathscr{L}_{=1}$(stl-det-strict-CD-2-RR(1)) *is not closed under union, intersection with regular sets, ε-free homomorphisms, and inverse homomorphisms.*

Proof. The languages $\{aaa\}$, $\{bb\}$, and $\{a, b\}^*$ are all accepted by stl-det-strict-CD-2-RR(1)-systems. As $\{aaa\} \cup \{bb\} = \{aaa, bb\} = \{aaa, bb\} \cap \{a, b\}^*$, Lemma 21 shows that this language class is neither closed under union nor under intersection with regular sets.

The languages $\{c, d\}$ and $\{c^6\}$ are accepted by stl-det-strict-CD-2-RR(1)-systems. Let $h_1 : \{c, d\}^* \to \{a, b\}^*$ be the homomorphism defined by $c \mapsto aaa$ and $d \mapsto bb$, and let $h_2 : \{a, b\}^* \to \{c\}^*$ be the homomorphism defined by $a \mapsto c^2$ and $b \mapsto c^3$. Then $h_1(\{c, d\}) = \{aaa, bb\} = h_2^{-1}(\{c^6\})$, and hence, Lemma 21 shows that this language class is neither closed under ε-free homomorphisms nor under inverse homomorphisms. □

Proposition 25. *The class* $\mathscr{L}_{=1}$(stl-det-strict-CD-2-RR(1)) *is* (a) *closed under complementation and* (b) *not closed under intersection.*

Proof. (a) Let $\mathcal{M} = ((M_i, \sigma_i)_{i\in I}, \{i_0\})$ be a stl-det-strict-CD-2-RR(1)-system on Σ such that $L_{=1}(\mathcal{M}) = L$. By interchanging accept transitions and undefined transitions within each function $\delta_1^{(i)}$ and $\delta_2^{(i)}$, we obtain a stl-det-strict-CD-2-RR(1)-system $\mathcal{M}' = ((M_i', \sigma_i)_{i\in I}, \{i_0\})$ that executes exactly the same cycles as $\mathcal{M}$, but for each index $i \in I$, the accepting tail computations of M_i' correspond to rejecting tail computations of M_i, and vice versa. Hence, it follows that $L_{=1}(\mathcal{M}') = \Sigma^* \setminus L = \overline{L}$.
(b) Closure under complementation and non-closure under union yield immediately that $\mathscr{L}_{=1}$(stl-det-strict-CD-2-RR(1)) is not closed under intersection. □

Proposition 26. *The class* $\mathscr{L}_{=1}$(stl-det-strict-CD-2-RR(1)) *is* (a) *not closed under commutative closure and* (b) *not closed under reversal.*

Proof. (a) From Example 7 we know that the language $L_2 = \{ a^n b^n \mid n \geq 0 \}$ is accepted by a stl-det-strict-CD-2-RR(1)-system. Its commutative closure is the language $L_= = \{ w \in \{a,b\}^* \mid |w|_a = |w|_b \geq 0 \}$.

Assume that $\mathcal{M} = ((M_i, \sigma_i)_{i\in I}, I_0)$ is a stl-det-strict-CD-2-RR(1)-system accepting the language $L_=$, and assume that $I_0 = \{i_0\}$. Then $\delta_1^{(i_0)}(¢) = \mathsf{MVR}$, and as $\varepsilon \in L_=$, we also have $\delta_1^{(i_0)}(\$) = \mathsf{Accept}$. As $a \notin L_=$, we see that $\delta_1^{(i_0)}(a) = \varepsilon$, and as $b \notin L_=$, we also have $\delta_1^{(i_0)}(b) = \varepsilon$. Further, it holds that $\delta_2^{(i_0)}(b) = \mathsf{Restart}$, or $\delta_2^{(i_0)}(b) = \mathsf{MVR}$ and $\delta_2^{(i_0)}(\$) = \mathsf{Restart}$, as $ab \in L_=$, while $abb \notin L_=$. Hence, $\mathcal{M}$ performs the following computations, where $\sigma_{i_0} = \{i_1\}$:

$$(i_0, ab) \vdash_{\mathcal{M}}^{c} (i_1, b) \vdash_{\mathcal{M}}^{*} \mathsf{Accept} \text{ and } (i_0, bb) \vdash_{\mathcal{M}}^{c} (i_1, b) \vdash_{\mathcal{M}}^{*} \mathsf{Accept}.$$

As bb does not belong to $L_=$, this contradicts our assumption on $\mathcal{M}$. Hence, $L_=$ is not accepted by any stl-det-strict-CD-2-RR(1)-system, which means that $\mathscr{L}_{=1}$(stl-det-strict-CD-2-RR(1)) is not closed under the operation of commutative closure.
(b) Let $L = \{ aaw \mid w \in \{a,b\}^* \}$. Then L is accepted by the following stl-det-strict-CD-2-RR(1)-system $\mathcal{M} = ((M_0, \{1\}), (M_1, \{1\}), \{0\})$, where M_0 and M_1 are defined as follows:

$$M_0 : \delta_1^{(0)} : ¢ \mapsto \mathsf{MVR},\ a \mapsto \varepsilon;\ \delta_2^{(0)} : a \mapsto \mathsf{Restart};$$
$$M_1 : \delta_1^{(1)} : ¢ \mapsto \mathsf{MVR},\ a \mapsto \varepsilon;\ \delta_2^{(1)} : a \mapsto \mathsf{MVR}, \quad b \mapsto \mathsf{MVR},\ \$ \mapsto \mathsf{Accept}.$$

Assume that $\mathcal{M} = ((M_i, \sigma_i)_{i\in I}, I_0)$ is a stl-det-strict-CD-2-RR(1)-system such that $L_{=1}(\mathcal{M}) = L^R$. Without loss of generality we can assume that $I = \{0, 1, \ldots, m\}$, that $I_0 = \{0\}$ and that $\sigma_0 = \{1\}$. Obviously, $\delta_1^{(0)}(¢) = \mathsf{MVR}$, and $\delta_1^{(0)}(a) = \varepsilon$, as $a \notin L^R$, while $a^2 \in L^R$. If $\delta_1^{(0)}(b) = \varepsilon$ as well, then with $aa \in L^R$, $\mathcal{M}$ would also accept $ba \notin L^R$. Hence, $\delta_1^{(0)}(b) = \mathsf{MVR}$. It remains to consider the function $\delta_2^{(0)}$.

If $\delta_2^{(0)}(a) = \mathsf{Accept}$, then $\mathcal{M}$ would accept the word $aab \notin L^R$. Also if $\delta_2^{(0)}(\$) = \mathsf{Accept}$, then $\mathcal{M}$ would accept the word $a \notin L^R$. Hence, $\delta_2^{(0)}(a) = \mathsf{Restart}$, or $\delta_2^{(0)}(a) = \mathsf{MVR}$ and $\delta_2^{(0)}(\$) = \mathsf{Restart}$. Further, as $abaa \in L^R$, while $abab \notin L^R$, it follows that $\delta_2^{(0)}(b) \in \{\mathsf{MVR}, \mathsf{Restart}\}$ holds, too. But then $\mathcal{M}$ executes the following computations:

$$(0, baa) \vdash_{\mathcal{M}}^{c} (1, ba) \vdash_{\mathcal{M}}^{*} \mathsf{Accept} \text{ and } (0, aba) \vdash_{\mathcal{M}}^{c} (1, ba) \vdash_{\mathcal{M}}^{*} \mathsf{Accept}.$$

As $aba \notin L^R$, this again contradicts our assumption on $\mathcal{M}$. Thus, L^R is not accepted by any stl-det-strict-CD-2-RR(1)-system, and it follows that $\mathscr{L}_{=1}$(stl-det-strict-CD-2-RR(1)) is not closed under reversal. □

For showing that the class $\mathscr{L}_{=1}$(stl-det-strict-CD-2-RR(1)) is an anti-AFL, it remains to be proven that this class is not closed under concatenation and Kleene star, either. Let L_p be the language $L_p = a^+ \cdot b \cdot a^+$ on $\Sigma_2 = \{a, b\}$.

Lemma 27. $L_p \in \mathscr{L}_{=1}$(stl-det-strict-CD-2-RR(1)).

Proof. The language L_p is accepted by the stl-det-strict-CD-2-RR(1)-system $\mathcal{M}_p = ((M_i, \sigma_i)_{i\in\{0,1,2\}}, \{0\})$, where $\sigma_0 = \{1\}$, $\sigma_1 = \{2\}$, $\sigma_2 = \{0\}$, and the stl-det-2-RR(1)-automata M_0, M_1 and M_2 are defined as follows:

$M_0 : \delta_1^{(0)} : ¢ \mapsto \mathsf{MVR}$, $a \mapsto \varepsilon$; $\quad \delta_2^{(0)} : a \mapsto \mathsf{Restart}$, $b \mapsto \mathsf{Restart}$;

$M_1 : \delta_1^{(1)} : ¢ \mapsto \mathsf{MVR}$, $a \mapsto \mathsf{MVR}$, $b \mapsto \varepsilon$; $\quad \delta_2^{(1)} : a \mapsto \mathsf{Restart}$;

$M_2 : \delta_1^{(2)} : ¢ \mapsto \mathsf{MVR}$, $a \mapsto \mathsf{MVR}$, $\$ \mapsto \mathsf{Accept}$. □

Non-closure under concatenation for $\mathscr{L}_{=1}$(stl-det-strict-CD-2-RR(1)) will follow from the following negative result.

Lemma 28. $L_p \cdot L_p \notin \mathscr{L}_{=1}$(stl-det-strict-CD-2-RR(1)).

Proof. Obviously,

$$L_p^2 = L_p \cdot L_p = a^+ \cdot b \cdot a \cdot a^+ \cdot b \cdot a^+ = \{\, a^m b a^n b a^p \mid m, p \geq 1,\, n \geq 2 \,\}.$$

We claim that the language L_p^2 is not accepted by any stl-det-strict-CD-2-RR(1)-system.

Assume to the contrary that $\mathcal{M} = ((M_i, \sigma_i)_{i\in I}, I_0)$ is a stl-det-strict-CD-2-RR(1)-system such that $L_{=1}(\mathcal{M}) = L_p^2$. Without loss of generality we may assume that $I = \{0, 1, \ldots, r\}$, and that $I_0 = \{0\}$.

We first analyze the transition functions of M_0. Obviously, $\delta_1^{(0)}(¢) = \mathsf{MVR}$. If $\delta_1^{(0)}(a) = \mathsf{MVR}$, then $\delta_1^{(0)}(b) = \varepsilon$, as $L_{=1}(\mathcal{M})$ is neither empty nor the set Σ_2^*. But then $\mathcal{M}$ cannot distinguish between the input $abaaba \in L_p^2$ and the input $aababa \notin L_p^2$, contradicting our assumption above. Hence, we conclude that $\delta_1^{(0)}(a) = \varepsilon$.

If $\delta_1^{(0)}(b) = \mathsf{Accept}$, then $\mathcal{M}$ would accept all words beginning with the letter b. If $\delta_1^{(0)}(b) = \mathsf{MVR}$, then $\mathcal{M}$ cannot distinguish between the input $abaaba \in L_p^2$

and the input $baaaba \notin L_p^2$. It follows that $\delta_1^{(0)}(b) = \emptyset$. Further, as in the second phase of the first cycle, M_0 cannot possibly ensure that the remaining tape contents is of the form $a^* \cdot b \cdot a \cdot a^+ \cdot b \cdot a^+$, we see that M_0 executes a restart operation after deleting the first a.

Let $\sigma_0 = \{1\}$. We continue by analyzing the transition functions of M_1. Obviously, $\delta_1^{(1)}(¢) = \mathsf{MVR}$.

Assume first that $\delta_1^{(1)}(a) = \varepsilon$. If also $\delta_1^{(1)}(b) = \varepsilon$, then $\mathcal{M}$ cannot distinguish between the input $aabaaba \in L_p^2$ and the input $abbaaba \notin L_p^2$, which contradicts our assumption above. If $\delta_1^{(1)}(b) = \mathsf{MVR}$, then we have the following partial computations of $\mathcal{M}$, where $\sigma_1 = \{2\}$ is taken:

$$(0, abaaba) \vdash_{\mathcal{M}}^{c} (1, baaba) \vdash_{\mathcal{M}}^{c} (2, baba),$$

and

$$(0, aababa) \vdash_{\mathcal{M}}^{c} (1, ababa) \vdash_{\mathcal{M}}^{c} (2, baba).$$

Hence, $\mathcal{M}$ cannot distinguish between the input $abaaba \in L_p^2$ and the input $aababa \notin L_p^2$. It follows that $\delta_1^{(1)}(a) = \mathsf{MVR}$. But then $\delta_1^{(1)}(b) = \varepsilon$ follows, which in turn means that $\mathcal{M}$ executes the following partial computations:

$$(0, abaaba) \vdash_{\mathcal{M}}^{c} (1, baaba) \vdash_{\mathcal{M}}^{c} (2, aaba)$$

and

$$(0, aababa) \vdash_{\mathcal{M}}^{c} (1, ababa) \vdash_{\mathcal{M}}^{c} (2, aaba).$$

This again shows that $\mathcal{M}$ cannot distinguish between the input $abaaba \in L_p^2$ and the input $aababa \notin L_p^2$. In conclusion we see that L_p^2 is not accepted by any stl-det-strict-CD-2-RR(1)-system. □

In fact, it can be shown that each stl-det-strict-CD-2-RR(1)-system that accepts all words from L_p^2 also accepts some words from $(a^* \cdot b \cdot a^*)^*$ that do not belong to the language L_p^*. Hence, it follows that L_p^* (and also L_p^+) is not accepted by any stl-det-strict-CD-2-RR(1)-system. Thus, we have the following additional non-closure results.

Corollary 29. *The language class* $\mathscr{L}_{=1}$(stl-det-strict-CD-2-RR(1)) *is not closed under concatenation, Kleene plus and Kleene star.*

Thus, we see that $\mathscr{L}_{=1}$(stl-det-strict-CD-2-RR(1)) is an anti-AFL.

4.2 Globally Deterministic CD-Systems of stl-det-2-RR(1)-Automata

In a globally deterministic CD-system of stateless deterministic R(1)-automata, each rewrite operation of each component automaton is associated with a particular successor index. Thus, if M_{i_1} is the active component, and if it executes a cycle involving the deletion of the letter $a \in \Sigma$, then the component $i_2 \in \sigma_{i_1}$

that is associated with the delete operation $\delta_{i_1}(a) = \varepsilon$ is activated. Hence, the choice of the successor component is based on the symbol deleted.

In a computation of a CD-system of stateless deterministic 2-RR(1)-automata, a successor component is chosen whenever the active component executes a restart operation. Accordingly, for these CD-systems we associate a particular successor index with each restart operation.

Let $((M_i, \sigma_i)_{i \in I}, I_0)$ be a CD-system of stateless deterministic 2-RR(1)-automata over Σ such that $|I_0| = 1$. For each $i \in I$, let $\Sigma_{\mathrm{rs}}^{(i)}$ be the set of symbols that cause the component automaton M_i to perform a restart operation, that is,

$$\Sigma_{\mathrm{rs}}^{(i)} = \{\, a \in \Sigma \mid \delta_2^{(i)}(a) = \mathsf{Restart} \,\} \cup \{\, \$ \mid \delta_2^{(i)}(\$) = \mathsf{Restart} \,\}.$$

Further, let $\delta : \bigcup_{i \in I}(\{i\} \times \Sigma_{\mathrm{rs}}^{(i)}) \to I$ be a mapping that assigns to each pair $(i, a) \in \{i\} \times \Sigma_{\mathrm{rs}}^{(i)}$ an element $j \in \sigma_i$. Then δ is called a *global successor function*. It assigns a successor component $j \in \sigma_i$ to the active component i based on the symbol $a \in \Sigma_{\mathrm{rs}}^{(i)}$ that causes M_i to perform a restart operation in the current cycle. It follows that, for each input word $w \in \Sigma^*$, the system $\mathcal{M} = ((M_i, \sigma_i)_{i \in I}, I_0, \delta)$ has a unique computation that starts from the initial configuration corresponding to input w. Accordingly we call $\mathcal{M}$ a *globally deterministic stateless* CD-2-RR(1)-*system*, and by $\mathscr{L}_{=1}$(stl-det-global-CD-2-RR(1)) we denote the class of languages that are accepted by these systems.

Obviously, each strictly deterministic stateless CD-2-RR(1)-system is globally deterministic. However, the globally deterministic stateless CD-2-RR(1)-systems are more expressive than the strictly deterministic ones.

Example 30. Let $\mathcal{M} = ((M_i, \sigma_i)_{i \in I}, I_0, \delta)$ be the globally deterministic CD-system of stateless deterministic 2-RR(1)-automata over $\Sigma = \{a, b\}$ that is defined as follows:
$I = \{0, 1, 2, 3, 4, 5\}$, $I_0 = \{0\}$, $\sigma_0 = \{1, 4\}$, $\sigma_1 = \{2\}$, $\sigma_2 = \{3\}$, $\sigma_3 = \{5\} = \sigma_4$, $\sigma_5 = \{1\}$, and M_0 to M_5 are the stateless deterministic 2-RR(1)-automata that are given by the following transition functions:

$$\begin{aligned}
M_0 &: \delta_1^{(0)} : ¢ \mapsto \mathsf{MVR},\ a \mapsto \varepsilon;\ \delta_2^{(0)} : a \mapsto \mathsf{Restart},\ b \mapsto \mathsf{Restart};\\
M_1 &: \delta_1^{(1)} : ¢ \mapsto \mathsf{MVR},\ a \mapsto \varepsilon;\ \delta_2^{(1)} : a \mapsto \mathsf{Restart};\\
M_2 &: \delta_1^{(2)} : ¢ \mapsto \mathsf{MVR},\ a \mapsto \varepsilon;\ \delta_2^{(2)} : a \mapsto \mathsf{Restart};\\
M_3 &: \delta_1^{(3)} : ¢ \mapsto \mathsf{MVR},\ a \mapsto \varepsilon;\ \delta_2^{(3)} : \$ \mapsto \mathsf{Accept};\\
M_4 &: \delta_1^{(4)} : ¢ \mapsto \mathsf{MVR},\ b \mapsto \varepsilon;\ \delta_2^{(4)} : b \mapsto \mathsf{Restart};\\
M_5 &: \delta_1^{(5)} : ¢ \mapsto \mathsf{MVR},\ b \mapsto \varepsilon;\ \delta_2^{(5)} : \$ \mapsto \mathsf{Accept}.
\end{aligned}$$

and δ is defined by $\delta(0, a) = 1$, $\delta(0, b) = 4$, $\delta(1, a) = 2$, $\delta(2, a) = 3$, $\delta(4, b) = 5$. Then it is easily seen that $L_{=1}(\mathcal{M}) = \{aaaa, abb\}$, which is not accepted by any strictly deterministic stateless CD-2-RR(1)-system by Lemma 23. □

Thus, we have the following proper inclusion.

Corollary 31

$$\mathscr{L}_{=1}(\text{stl-det-strict-CD-2-RR(1)}) \subsetneq \mathscr{L}_{=1}(\text{stl-det-global-CD-2-RR(1)}).$$

Further, we relate the stl-det-global-CD-2-RR(1)-systems to the stl-det-local-CD-2-RR(1)-systems.

Proposition 32

$$\mathscr{L}_{=1}(\text{stl-det-global-CD-2-RR(1)}) \subseteq \mathscr{L}_{=1}(\text{stl-det-local-CD-2-RR(1)}).$$

Proof Let $\mathcal{M} = ((M_i, \sigma_i)_{i\in I}, \{i_0\}, \delta)$ be a stl-det-global-CD-2-RR(1)-system on alphabet Σ and, for each $i \in I$, $\Sigma_{\mathrm{rs}}^{(i)}$ as defined above. From $\mathcal{M}$ we now construct a stl-det-local-CD-2-RR(1)-system $\mathcal{M}' = ((M'_j, \sigma'_j)_{j\in J}, J_0)$ satisfying $L_{=1}(\mathcal{M}') = L_{=1}(\mathcal{M})$. For all $i \in I$, let $S^{(i)} = \Sigma_{\mathrm{rs}}^{(i)}$, if $\Sigma_{\mathrm{rs}}^{(i)} \neq \emptyset$, and $S^{(i)} = \{+\}$, otherwise. Now let $J = \{\, (i,a) \mid i \in I,\, a \in S^{(i)} \,\}$, let $J_0 = \{\, (i_0, a) \mid a \in S^{(i_0)} \,\}$, and for all $i \in I$, take

$$\begin{aligned} \sigma'_{(i,a)} &= \{\, (j,b) \mid j = \delta(i,a),\, b \in S^{(j)} \,\} \text{ for all } a \in \Sigma_{\mathrm{rs}}^{(i)}, \\ \sigma'_{(i,+)} &= J_0, \qquad\qquad\qquad\qquad \text{if } \Sigma_{\mathrm{rs}}^{(i)} = \emptyset. \end{aligned}$$

Finally, we define the stateless deterministic 2-RR(1)-automata $M'_{(i,a)}$ as follows, where $i \in I$, $a \in S^{(i)}$, and $b \in \Sigma$:

$$\begin{aligned} M'_{(i,a)} : \delta_1^{(i,a)}(x) &= \delta_1^{(i)}(x) \quad \text{for all } x \in \Sigma \cup \{¢, \$\}; \\ \delta_2^{(i,a)}(x) &= \delta_2^{(i)}(x) \quad \text{for all } x \in (\Sigma \cup \{\$\}) \smallsetminus \Sigma_{\mathrm{rs}}, \\ \delta_2^{(i,a)}(a) &= \mathsf{Restart}, \text{ if } a \in \Sigma_{\mathrm{rs}}, \\ \delta_2^{(i,a)}(b) &= \emptyset, \qquad \text{for all } b \in \Sigma_{\mathrm{rs}} \smallsetminus \{a\}. \end{aligned}$$

Let $w = a_1 a_2 \cdots a_n \in \Sigma^*$, where $n \geq 0$ and $a_1, \ldots, a_n \in \Sigma$. Assume that the computation of $\mathcal{M}$ on input w has the following form:

$$\begin{aligned} (i_0, w) = (i_0, u_0 b_0 v_0) \vdash_{\mathcal{M}}^{c} (i_1, u_0 v_0) &= (i_1, u_1 b_1 v_1) \vdash_{\mathcal{M}}^{c} \cdots \\ \vdash_{\mathcal{M}}^{c} (i_r, u_{r-1} v_{r-1}) &= (i_r, w_r), \end{aligned}$$

and that starting with the configuration (i_r, w_r), the component automaton M_{i_r} performs a tail computation. Then $\mathcal{M}'$ can simulate this sequence of cycles by guessing, in each step, on which letter the next restart operation of $\mathcal{M}$ will be executed. Thus, we conclude that $L_{=1}(\mathcal{M}) \subseteq L_{=1}(\mathcal{M}')$ holds.

Conversely, if $\mathcal{M}'$ has an accepting computation on input $w \in \Sigma^*$, then it follows easily from the above construction of $\mathcal{M}'$ that $\mathcal{M}$ will also accept on input w. Thus, we see that $L_{=1}(\mathcal{M}') = L_{=1}(\mathcal{M})$, which completes the proof. □

Since all rational trace languages are accepted by stl-det-local-CD-2-RR(1)-systems, the inclusion result above raises the question of whether all rational trace languages are accepted by stl-det-global-CD-2-RR(1)-systems as well. The following result answers this question in the negative.

Proposition 33. *The rational trace language*

$$L_\vee = \{\, w \in \{a,b\}^* \mid \exists n \geq 0 : |w|_a = n \text{ and } |w|_b \in \{n, 2n\} \,\}$$

is not accepted by any globally deterministic stateless CD-2-RR(1)*-system.*

Proof. As $L_\vee$ is the commutative closure of the regular language $(ab)^* \cup (abb)^*$, it is obviously a rational trace language.

It remains to be proven that $L_\vee \notin \mathscr{L}_{=1}(\text{stl-det-global-CD-2-RR(1)})$. Assume to the contrary that $\mathcal{M} = ((M_i, \sigma_i)_{i \in I}, I_0, \delta)$ is a stl-det-global-CD-2-RR(1)-system such that $L_{=1}(\mathcal{M}) = L_\vee$. Without loss of generality we can assume that $I = \{0, 1, \ldots, m-1\}$ and that $I_0 = \{0\}$.

Let $n > 2m$, and let $w = a^n b^n \in L_\vee$. Then the computation of $\mathcal{M}$ on input w is accepting, that is, it is of the form

$$(0, a^n b^n) \vdash^c_{\mathcal{M}} (i_1, w_1) \vdash^c_{\mathcal{M}} \cdots \vdash^c_{\mathcal{M}} (i_r, w_r) \vdash^*_{M_{i_r}} \mathsf{Accept},$$

where M_{i_r} accepts the tape contents $¢w_r\$$ in a tail computation. Let $i = |w_r|_a$ and $j = |w_r|_b$.

If $j > 1$, then M_{i_r} would also accept the tape contents $w_r b^k = a^i b^{j+k}$ for any $k > 0$, and therewith $\mathcal{M}$ would accept the input $wb^{2n} = a^n b^{3n}$. As this word is not contained in $L_\vee$, this contradicts our assumption that $L_{=1}(\mathcal{M}) = L_\vee$. Hence, we conclude that $j = |w_r|_b \leq 1$.

Analogously, if $i > 1$, then M_{i_r} would also accept the tape contents $a^{i+k} b^j$ for any $k > 0$, and therewith $\mathcal{M}$ would accept the input $a^n w = a^{2n} b^n \notin L_\vee$. Hence, we conclude that $i = |w_r|_a \leq 1$. Thus, $|w_r| = i + j \leq 2$, which shows that in the above computation at least the first $n-1$ occurrences of the letter a and the first $n-1$ occurrences of the letter b are deleted letter by letter, and then M_{i_r} accepts the word w_r of length at most two.

As $n > m$, there exists an index $i \in I$ such that the component automaton M_i is used twice within the above sequence of cycles. Thus, there are integers $s, t, k, \ell \geq 0$, $m \geq s + t \geq 0$ and $m \geq k + \ell > 0$, such that the above computation can be written as follows:

$$(0, a^n b^n) \vdash^{c^*}_{\mathcal{M}} (i, a^{n-s} b^{n-t}) \vdash^{c^+}_{\mathcal{M}} (i, a^{n-s-k} b^{n-t-\ell}) \vdash^{c^*}_{\mathcal{M}} (i_r, w_r) \vdash^*_{M_{i_r}} \mathsf{Accept}.$$

Obviously, $\mathcal{M}$ will also execute the following shortened computation:

$$(0, a^{n-k} b^{n-\ell}) \vdash^{c^*}_{\mathcal{M}} (i, a^{n-s-k} b^{n-t-\ell}) \vdash^{c^*}_{\mathcal{M}} (i_r, w_r) \vdash^*_{M_{i_r}} \mathsf{Accept},$$

that is, $\mathcal{M}$ accepts on input $a^{n-k} b^{n-\ell}$. From our assumption that $L_{=1}(\mathcal{M}) = L_\vee$ we can therefore conclude that $k = \ell$, as $n > 2m$.

Now consider the computation of $\mathcal{M}$ on input $a^n b^{2n}$. As $a^n b^{2n} \in L_\vee$, this computation is accepting, that is, it has the following form:

$$(0, a^n b^{2n}) \vdash^{c^*}_{\mathcal{M}} (i, a^{n-s} b^{2n-t}) \vdash^{c^+}_{\mathcal{M}} (i, a^{n-s-k} b^{2n-t-k}) \vdash^{c^*}_{\mathcal{M}} (i', z') \vdash^*_{M_{i'}} \mathsf{Accept}$$

for some $i' \in I$ and some word $z' \in \Sigma^*$. But then $\mathcal{M}$ will also execute the following computation:

$$(0, a^{n-k} b^{2n-k}) \vdash^{c^*}_{\mathcal{M}} (i, a^{n-s-k} b^{2n-t-k}) \vdash^{c^*}_{\mathcal{M}} (i', z') \vdash^*_{M_{i'}} \mathsf{Accept},$$

that is, it accepts on input $a^{n-k}b^{2n-k} \notin L_\vee$. It follows that $L_{=1}(\mathcal{M}) \neq L_\vee$, that is, $L_\vee$ is not accepted by any globally deterministic stateless CD-2-RR(1)-system working in mode $= 1$. □

This yields the following consequence.

Corollary 34.

$$\mathscr{L}_{=1}(\text{stl-det-global-CD-2-RR(1)}) \subsetneq \mathscr{L}_{=1}(\text{stl-det-local-CD-2-RR(1)}).$$

The Dyck language D_1 is not a rational trace language, but it is accepted by a strictly deterministic stateless CD-2-RR(1)-system as can be shown easily (see Example 14). Thus, we have the following consequence.

Corollary 35. *The two language classes* $\mathscr{L}_{=1}$(stl-det-strict-CD-2-RR(1)) *and* $\mathscr{L}_{=1}$(stl-det-global-CD-2-RR(1)) *are incomparable under inclusion to the class of rational trace languages.*

In a stl-det-global-CD-R(1)-system, the choice of the successor component is based on the letter removed in the current cycle, while in a stl-det-global-CD-2-RR(1)-system, this choice is based on the letter on which the currently active component automaton executes the restart that completes the current cycle. This raises the question of whether each stl-det-global-CD-R(1)-system can be simulated by a stl-det-global-CD-2-RR(1)-system. In order to answer this question we first note that Lemma 20 applies also to stl-det-global-CD-2-RR(1)-systems. Hence, from Lemma 21 we adapt the following negative result.

Corollary 36. *The finite language* $L_0 = \{aaa, bb\}$ *is not accepted by any globally deterministic stateless* CD-2-RR(1)*-system.*

Since all regular languages are accepted by stl-det-global-CD-R(1)-systems, the corollary implies that the class $\mathscr{L}_{=1}$(stl-det-global-CD-R(1)) is not contained in $\mathscr{L}_{=1}$(stl-det-global-CD-2-RR(1)). On the other hand, Example 7 shows that already stl-det-strict-CD-2-RR(1)-systems accept some languages that are not accepted by stl-det-local-CD-R(1)-systems. Hence, we have the following incomparability results.

Corollary 37. *The language classes*

$$\mathscr{L}_{=1}(\text{stl-det-strict-CD-2-RR(1)}) \text{ and } \mathscr{L}_{=1}(\text{stl-det-global-CD-2-RR(1)})$$

are incomparable under inclusion to the classes

$$\mathscr{L}_{=1}(\text{stl-det-global-CD-R(1)}) \text{ and } \mathscr{L}_{=1}(\text{stl-det-local-CD-R(1)}).$$

Even though stl-det-global-CD-2-RR(1)-systems cannot accept all finite languages, they seem to be powerful devices. In particular, the language of Example 10, which is not even growing context-sensitive, can be shown to belong to the class $\mathscr{L}_{=1}$(stl-det-global-CD-2-RR(1)) by adding a corresponding global successor function to the stl-det-local-CD-2-RR(1))-system of Example 10.

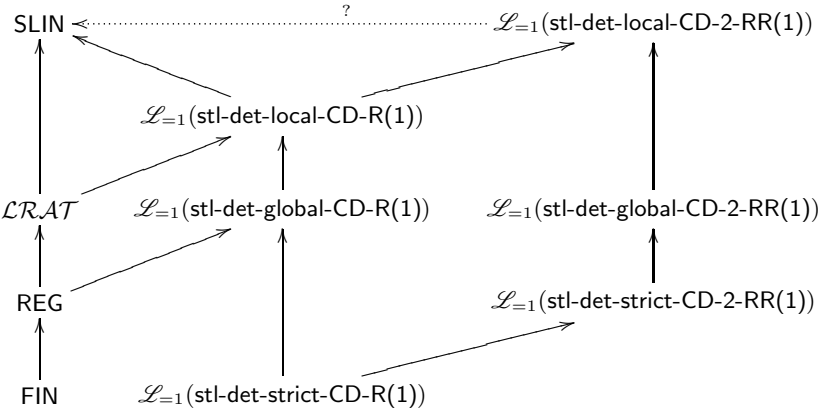

Fig. 1. Hierarchy of language classes accepted by various types of CD-R(1)- and CD-2-RR(1)-systems. Here SLIN denotes the class of *semi-linear languages*, $\mathcal{LRAT}$ is the class of *rational trace languages*, and FIN is the class of all finite languages. Each arrow represents a proper inclusion, the dotted arrow represents an inclusion that is still open, and all other classes that are not connected by a sequence of arrows are incomparable under inclusion.

Theorem 38. *The language class* $\mathscr{L}_{=1}$(stl-det-global-CD-2-RR(1)) *is incomparable under inclusion to the classes* GCSL *of growing context-sensitive languages,* CRL *of Church-Rosser languages,* CFL *of context-free languages,* LIN *of linear languages,* DCFL *of deterministic context-free languages,* REG *of regular languages as well as to the class* FIN *of finite languages.*

The diagram in Figure 1 summarizes our inclusion results. We have the following results on closure and non-closure properties for the language class $\mathscr{L}_{=1}$(stl-det-global-CD-2-RR(1)).

Proposition 39. *The class* $\mathscr{L}_{=1}$(stl-det-global-CD-2-RR(1)) *is* (a) *closed under complementation,* (b) *not closed under union or intersection,* (c) *not closed under intersection with regular languages, concatenation, ε-free homomorphisms, and inverse homomorphisms, and* (d) *not closed under commutative closure and reversal.*

Proof. (a) Let $\mathcal{M} = ((M_i, \sigma_i)_{i \in I}, \{i_0\}, \delta)$ be a stl-det-global-CD-2-RR(1)-system on Σ such that $L_{=1}(\mathcal{M}) = L$. By interchanging accept transitions and undefined transitions within each function $\delta_1^{(i)}$ and $\delta_2^{(i)}$, we obtain a stl-det-global-CD-2-RR(1)-system $\mathcal{M}' = ((M_i', \sigma_i)_{i \in I}, \{i_0\}, \delta)$ that executes exactly the same cycles as $\mathcal{M}$, but for each index $i \in I$, the accepting tail computations of M_i' correspond to rejecting tail computations of M_i, and vice versa. Hence, it follows that $L_{=1}(\mathcal{M}') = \Sigma^* \setminus L = \overline{L}$.

(b) By Corollary 36 the finite language $L_0 = \{aaa, bb\}$ is not accepted by any stl-det-global-CD-2-RR(1)-system. It is easily seen that the languages

$L_{0,1} = \{aaa\}$ and $L_{0,2} = \{bb\}$ are accepted by such systems and, thus, the class $\mathscr{L}_{=1}(\mathsf{stl\text{-}det\text{-}global\text{-}CD\text{-}2\text{-}RR(1)})$ is not closed under union. Together with closure under complementation this also yields non-closure under intersection.

(c) As $\{a,b\}^*$ is accepted by a stl-det-global-CD-2-RR(1)-system, and as L_0 is regular, we see from the claim above and the fact that $L_0 = \{a,b\}^* \cap L_0$ that the class $\mathscr{L}_{=1}(\mathsf{stl\text{-}det\text{-}global\text{-}CD\text{-}2\text{-}RR(1)})$ is not closed under intersection with regular languages.

It is not hard to see that the languages $\{\varepsilon, aa\}$ and $\{\varepsilon, bb\}$ are accepted by stl-det-global-CD-2-RR(1)-systems. By an application of Lemma 20 their concatenation $\{\varepsilon, aa\} \cdot \{\varepsilon, bb\} = \{\varepsilon, aa, bb, aabb\}$ is not.

The languages $\{c,d\}$ and $\{c^6\}$ are accepted by stl-det-global-CD-2-RR(1)-systems. Let $h_1 : \{c,d\}^* \to \{a,b\}^*$ be the homomorphism defined by $c \mapsto aaa$ and $d \mapsto bb$, and let $h_2 : \{a,b\}^* \to \{c\}^*$ be the homomorphism defined by $a \mapsto c^2$ and $b \mapsto c^3$. Then $h_1(\{c,d\}) = \{aaa, bb\} = h_2^{-1}(\{c^6\})$, and hence, Corollary 36 shows that the language class $\mathscr{L}_{=1}(\mathsf{stl\text{-}det\text{-}global\text{-}CD\text{-}2\text{-}RR(1)})$ is neither closed under ε-free homomorphisms nor under inverse homomorphisms.

(d) Since the regular language $(ab)^* \cup (abb)^*$ can be accepted by some stl-det-global-CD-2-RR(1)-system, Proposition 33 implies that the language class $\mathscr{L}_{=1}(\mathsf{stl\text{-}det\text{-}global\text{-}CD\text{-}2\text{-}RR(1)})$ is not closed under commutative closure.

From Example 30 we know that the language $L_0' = \{aaaa, abb\}$ is accepted by a stl-det-global-CD-2-RR(1)-system. In analogy it can be shown that also the language $L_1' = \{caaa, cbb\}$ is accepted by a stl-det-global-CD-2-RR(1)-system. Here we claim that the language ${L_1'}^R = \{aaac, bbc\}$ is not accepted by any stl-det-global-CD-2-RR(1)-system.

Assume that $\mathcal{M} = ((M_i, \sigma_i)_{i \in I}, I_0, \delta)$ is a stl-det-global-CD-2-RR(1)-system accepting the language ${L_1'}^R$. Without loss of generality we can assume that $I = \{0, 1, \ldots, n\}$, and that $I_0 = \{0\}$. Obviously, $\delta_1^{(0)}(¢) = \mathsf{MVR}$. We now consider various cases.

(i) If $\delta_1^{(0)}(a) = \delta_1^{(0)}(b) = \mathsf{MVR}$, then necessarily $\delta_1^{(0)}(c) = \varepsilon$ and $\delta_2^{(0)}(\$) = \mathsf{Restart}$ follow. Let $\delta(0, \$) = 1$. Then the system $\mathcal{M}' = ((M_i, \sigma_i)_{i \in I}, \{1\}, \delta)$ accepts the language $L_0 = \{aaa, bb\}$. This, however, contradicts Corollary 36.

(ii) If $\delta_1^{(0)}(a) = \delta_1^{(0)}(b) = \varepsilon$, Lemma 20 shows that ${L_1'}^R$ is not accepted by $\mathcal{M}$.

(iii) If $\delta_1^{(0)}(a) = \mathsf{MVR}$ and $\delta_1^{(0)}(b) = \varepsilon$, then $\delta_1^{(0)}(c) = \varepsilon$ and $\delta_2^{(0)}(\$) = \mathsf{Restart}$. Then $\mathcal{M}$ executes the following accepting computation:

$$(0, aaac) \vdash_{\mathcal{M}}^{c} (1, aaa) \vdash_{\mathcal{M}}^{*} \mathsf{Accept},$$

where $\delta(0, \$) = 1$. But then $\mathcal{M}$ also executes the following computation:

$$(0, aaab) \vdash_{\mathcal{M}}^{c} (1, aaa) \vdash_{\mathcal{M}}^{*} \mathsf{Accept},$$

which again contradicts our assumption on $\mathcal{M}$, as $aaab \notin {L_1'}^R$.

(iv) If $\delta_1^{(0)}(a) = \varepsilon$ and $\delta_1^{(0)}(b) = \mathsf{MVR}$, then $\delta_1^{(0)}(c) = \varepsilon$ and $\delta_2^{(0)}(\$) = \mathsf{Restart}$. Then $\mathcal{M}$ executes the following accepting computation:

$$(0, bbc) \vdash_{\mathcal{M}}^{c} (1, bb) \vdash_{\mathcal{M}}^{*} \mathsf{Accept},$$

where $\delta(0,\$) = 1$. But then $\mathcal{M}$ also executes the following computation:

$$(0, bba) \vdash^{c}_{\mathcal{M}} (1, bb) \vdash^{*}_{\mathcal{M}} \mathsf{Accept},$$

which again contradicts our assumption on $\mathcal{M}$, as $bba \notin {L'_1}^R$.

As this covers all cases, we see that indeed ${L'_1}^R$ is not accepted by any stl-det-global-CD-2-RR(1)-system. □

5 Decidability Problems

In this section we turn to investigate decidability problems for the classes $\mathscr{L}_{=1}$(stl-det-local-CD-R(1)) and $\mathscr{L}_{=1}$(stl-det-local-CD-2-RR(1)). Basically it turns out that all undecidable problems are not even semidecidable. For these problems, it suffices to show the results for $\mathscr{L}_{=1}$(stl-det-local-CD-R(1)). We will use a reduction of Post's Correspondence Problem (PCP) (see, for example, [17]).

Let Σ be an alphabet. An instance of the PCP is given by two lists $\alpha = \alpha_1, \alpha_2, \ldots, \alpha_n$ and $\beta = \beta_1, \beta_2, \ldots, \beta_n$ of words from Σ^+. It is well known that it is undecidable whether a PCP has a solution [16], that is, whether there is a nonempty finite sequence of indices $i_1, i_2, \ldots, i_k$ such that $\alpha_{i_1}\alpha_{i_2}\cdots\alpha_{i_k} = \beta_{i_1}\beta_{i_2}\cdots\beta_{i_k}$. In particular, it is semidecidable whether a PCP has a solution, but is *not* semidecidable whether it has *no* solution. In the sequel we call $i_1, i_2, \ldots, i_k$ as well as $\alpha_{i_1}\alpha_{i_2}\cdots\alpha_{i_k}$ a solution of the PCP.

Theorem 40. *Regularity, context-freeness, equivalence, and inclusion are not semidecidable for* $\mathscr{L}_{=1}$(stl-det-local-CD-R(1)) *and* $\mathscr{L}_{=1}$(stl-det-local-CD-2-RR(1)).

Proof. Let an instance of the PCP be given by the lists $\alpha = \alpha_1, \alpha_2, \ldots, \alpha_n$ and $\beta = \beta_1, \beta_2, \ldots, \beta_n$ of nonempty words over some alphabet $\Sigma = \{a_1, a_2, \ldots, a_m\}$. Further, let $H = \{1, 2, \ldots, n\}$, $\alpha_j = \alpha_{j,1}\alpha_{j,2}\cdots\alpha_{j,|\alpha_j|}$, $\beta_j = \beta_{j,1}\beta_{j,2}\cdots\beta_{j,|\beta_j|}$, define $\tilde{\Sigma} = \{\, \tilde{a} \mid a \in \Sigma \,\}$ to be a disjoint copy of Σ, and set $\tilde{\beta}_j = \tilde{\beta}_{j,1}\tilde{\beta}_{j,2}\cdots\tilde{\beta}_{j,|\beta_j|}$, for $1 \leq j \leq n$.

In order to construct a language that meets our purposes we start with the set $E = \{\, x_1x'_1x_2x'_2\cdots x_\ell x'_\ell \mid \ell \geq 0, x_i \in \Sigma, x'_i \in \tilde{\Sigma}, 1 \leq i \leq \ell, \text{ and } x'_i = \tilde{x}_i \,\}$, that is, the set of words in which symbols from Σ and $\tilde{\Sigma}$ occur alternatingly (!) so that each symbol from $\tilde{\Sigma}$ is the copy of its left neighbor from Σ. Now define

$$L_1 = ((\Sigma \cup \tilde{\Sigma})^* \setminus E) \sqcup\!\sqcup H^*,$$

that is, the complement of E with respect to $\Sigma \cup \tilde{\Sigma}$ shuffled with indices from H, and

$$L_2 = (E \sqcup\!\sqcup H^*) \cap \{\, (w \sqcup\!\sqcup \tilde{w}) \sqcup\!\sqcup v \mid v = v_1v_2\cdots v_k \in H^+ \text{ such that } w = \alpha_{v_1}\alpha_{v_2}\cdots\alpha_{v_k}, \tilde{w} = \tilde{\beta}_{v_1}\tilde{\beta}_{v_2}\cdots\tilde{\beta}_{v_k} \,\}.$$

To conclude the construction of the language set $L_P = L_1 \cup L_2$. It can be shown that L_P is accepted by a stl-det-local-CD-R(1)-system $\mathcal{M}$. Essentially $\mathcal{M}$ is the union of two subsystems, the one of which accepts the language L_1, while the other accepts a superset of the language L_2.

Now assume that the PCP has no solution. Then L_2 is empty and $L_P = L_1$ is regular and, thus, context-free. Conversely, if L_P is context-free, then $L_P \cap (E \sqcup\!\sqcup H^*) = L_2$ is context free. A straightforward application of the pumping lemma on words of the form $(\Sigma \cup \tilde{\Sigma})^* \cdot H^*$ that belong to L_2 shows a contradiction. Therefore, the non-semidecidability of regularity and context-freeness follows by the non-semidecidability of the unsolvability of the PCP.

As mentioned above, $\mathcal{M}$ includes a subsystem for accepting L_1. Therefore, the semidecidability of equivalence implies the semidecidability of whether $\mathcal{M}$ accepts L_1, that is, whether L_2 is empty and, thus, the semidecidability of the unsolvability of the PCP. Finally, if inclusion is semidecidable then so is equivalence. □

Theorem 41. *Universality and cofiniteness are not semidecidable for the classes* $\mathscr{L}_{=1}$(stl-det-local-CD-R(1)) *and* $\mathscr{L}_{=1}$(stl-det-local-CD-2-RR(1)).

Proof. Given an instance of the PCP, a stl-det-local-CD-2-RR(1)-system can be constructed for the language $\overline{L_2}$ (which includes L_1). Since L_2 is empty if and only if the PCP has no solution, $\overline{L_2} = (\Sigma \cup \tilde{\Sigma} \cup H)^*$ if and only if the PCP has no solution. Therefore, universality is non-semidecidable.

A PCP has either no solution or infinitely many solutions. So $\overline{L_2}$ is cofinite if and only if the PCP has no solution, which completes the proof. □

6 Conclusions

We have investigated cooperating distributed systems of stateless deterministic two-phase RR-automata of window size one. The main interest was on the computational power and the closure properties of the language classes induced by the systems considered. The proven inclusion relations are depicted in Figure 1, while Table 1 summarizes the closure and non-closure properties obtained. Moreover, we considered decidability problems for the classes $\mathscr{L}_{=1}$(stl-det-local-CD-R(1)) and $\mathscr{L}_{=1}$(stl-det-local-CD-2-RR(1)). However, several questions remain unanswered: (1) Do the systems in question only accept semilinear languages? (2) Is the class $\mathscr{L}_{=1}$(stl-det-local-CD-2-RR(1)) not closed under concatenation or Kleene plus? Similarly, is $\mathscr{L}_{=1}$(stl-det-global-CD-2-RR(1)) not closed under Kleene plus? (3) What are the remaining algorithmic properties of the language class $\mathscr{L}_{=1}$(stl-det-local-CD-2-RR(1))? Is emptiness or finiteness decidable? How about the decidability problems for $\mathscr{L}_{=1}$(stl-det-strict-CD-2-RR(1)) and $\mathscr{L}_{=1}$(stl-det-global-CD-2-RR(1))?

Table 1. "+" denotes the fact that the corresponding class is closed under the given operation, "−" denotes the fact that it is not closed, and "?" indicates that the status of this property is still open. $\cup$ denotes union, $\overline{\ }$ complementation, $\cap$ intersection, $\cap_{REG}$ intersection with regular languages, $\cdot$ concatenation, $^+$ Kleene plus, h_ε ε-free homomorphism, h^{-1} inverse homomorphism, *com* commutative closure, and ^{R} denotes reversal.

Types of CD-Systems	**Operations**									
	$\cup$	$\overline{\ }$	$\cap$	$\cap_{REG}$	$\cdot$	$^+$	h_ε	h^{-1}	*com*	^{R}
stl-det-local-CD-2-RR(1)	+	−	−	−	?	?	−	−	?	−
stl-det-global-CD-2-RR(1)	−	+	−	−	−	?	−	−	−	−
stl-det-strict-CD-2-RR(1)	−	+	−	−	−	−	−	−	−	−

References

1. Csuhaj-Varjú, E., Dassow, J., Kelemen, J., Păun, G.: Grammar Systems. A Grammatical Approach to Distribution and Cooperation. Gordon and Breach, London (1994)
2. Culy, C.: Formal properties of natural language and linguistic theories. Linguistics and Philosophy 19, 599–617 (1996)
3. Dassow, J., Păun, G., Rozenberg, G.: Grammar systems. In: Rozenberg, G., Salomaa, A. (eds.) Handbook of Formal Languages, vol. 2, pp. 155–213. Springer, Berlin (1997)
4. Diekert, V., Rozenberg, G.: The Book of Traces. World Scientific, Singapore (1995)
5. Harrison, M.A.: Introduction to Formal Language Theory. Addison-Wesley, Reading (1978)
6. Ibarra, O., Karhumäki, J., Okhotin, A.: On stateless multihead automata: Hierarchies and the emptiness problem. Theoret. Comput. Sci. 411, 581–593 (2009)
7. Kutrib, M., Messerschmidt, H., Otto, F.: On stateless two-pushdown automata and restarting automata. Int. J. Found. Comput. Sci. 21, 781–798 (2010)
8. Kutrib, M., Messerschmidt, H., Otto, F.: On stateless deterministic restarting automata. Acta Inform. 47, 391–412 (2010)
9. Kutrib, M., Reimann, J.: Succinct description of regular languages by weak restarting automata. Inform. Comput. 206, 1152–1160 (2008)
10. Lautemann, C.: One pushdown and a small tape. In: Dirk Siefkes zum 50. Geburtstag, pp. 42–47. TU Berlin and Universität Augsburg (1988)
11. McNaughton, R., Narendran, P., Otto, F.: Church-Rosser Thue systems and formal languages. J. ACM 35, 324–344 (1988)
12. Messerschmidt, H., Otto, F.: Cooperating distributed systems of restarting automata. Int. J. Found. Comput. Sci. 18, 1333–1342 (2007)
13. Messerschmidt, H., Otto, F.: Strictly Deterministic CD-Systems of Restarting Automata. In: Csuhaj-Varjú, E., Ésik, Z. (eds.) FCT 2007. LNCS, vol. 4639, pp. 424–434. Springer, Heidelberg (2007)
14. Nagy, B., Otto, F.: CD-Systems of Stateless Deterministic R(1)-Automata Accept All Rational Trace Languages. In: Dediu, A.-H., Fernau, H., Martín-Vide, C. (eds.) LATA 2010. LNCS, vol. 6031, pp. 463–474. Springer, Heidelberg (2010)

15. Nagy, B., Otto, F.: Globally Deterministic CD-Systems of Stateless R(1)-Automata. In: Dediu, A.-H., Inenaga, S., Martín-Vide, C. (eds.) LATA 2011. LNCS, vol. 6638, pp. 390–401. Springer, Heidelberg (2011)
16. Post, E.L.: A variant of a recursively unsolvable problem. Bull. AMS 52, 264–268 (1946)
17. Salomaa, A.: Formal Languages. Academic Press, New York (1973)
18. Yang, L., Dang, Z., Ibarra, O.: On stateless automata and P systems. In: Workshop on Automata for Cellular and Molecular Computing, pp. 144–157. MTA SZTAKI (2007)