On Stateless Restarting Automata

Martin Kutrib\textsuperscript{1}, Hartmut Messerschmidt\textsuperscript{2}, Friedrich Otto\textsuperscript{3}

\textsuperscript{1}Institut für Informatik, Universität Giessen
D-35392 Giessen

\textsuperscript{2}Technologie-Zentrum Informatik, Universität Bremen
D-28359 Bremen

\textsuperscript{3}Fachbereich Elektrotechnik/Informatik, Universität Kassel
D-34109 Kassel

35th International Conference on
Current Trends in Theory and Practice of Computer Science
Špindlerův Mlýn, Czech Republic, Jan. 24–30, 2009
1 Introduction
2 Restarting Automata
3 Stateless RWW-Automata
4 Stateless RRWW-Automata
5 Stateless Two-Phase RRWW-Automata
6 Concluding Remarks
Traditional automata:

- Finite state control
- Input tape
- Memory

Diagram representation of a traditional automata with finite-state control, input tape, and memory.
Unconventional computing models:

P-systems:

DNA computing: multiset of DNA-strands

No global state.
Study traditional models of automata *without* states:

- How do other additional resources (heads, memory) given to finite automata relate to the absence or presence of states?
- For which computational models are states really necessary?
A stateless finite-state automaton with input alphabet $\Sigma$ accepts $\Sigma_0^*$ for a subset $\Sigma_0$ of $\Sigma$.

A language is accepted by a stateless pushdown automaton iff it is context-free.

A language is accepted by a stateless dpda iff it is a simple language [Har78].

Stateless multihead finite automata and stateless multicounter systems have been investigated by O. Ibarra and his co-authors [IKO2007, YDI2007].

**Theorem 1**

$\forall k \geq 1 \exists$ finite language $L \subseteq \{a\}^*$ : $L$ is not accepted by any stateless two-way nondeterministic finite automaton with $k$ heads.

Here: stateless restarting automata
2. Restarting Automata

An **RRWW-automaton** is defined as $M = (Q, \Sigma, \Gamma, \triangleleft, \triangleright, q_0, k, \delta)$, where

- $Q$ is a finite set of states,
- $\Sigma$ is a finite input alphabet,
- $\Gamma$ is a finite tape alphabet, $\Gamma \supseteq \Sigma$,
- $\triangleleft, \triangleright \notin \Gamma$ are the delimiters for the tape,
- $q_0 \in Q$ is the initial state,
- $k \in \mathbb{N}_+$ is the size of the read/write window,
- $\delta$ is the transition relation:
  $\delta : \triangleleft \cdot \Gamma^{k-1} \cup \Gamma^k \cup \Gamma^{\leq k-1} \cdot \triangleright \cup \triangleleft \cdot \Gamma^{\leq k-2} \cdot \triangleright \rightarrow P_{\text{fin}}(\text{Ops})$. 

A restarting automaton

<table>
<thead>
<tr>
<th>$\triangleleft$</th>
<th>$a_1$</th>
<th>$a_2$</th>
<th>$a_3$</th>
<th>$a_4$</th>
<th>$a_5$</th>
<th>$\ldots$</th>
<th>$\triangleright$</th>
<th>flexible tape</th>
</tr>
</thead>
</table>

read/write window

finite-state control
An RRWW-automaton can execute the following operations:

- A move-right step \((q', \text{MVR}) \in \delta(q, u)\)
- A rewrite step \((q', v) \in \delta(q, u), \text{where } |v| < |u|:\)

- A restart step:

- An accept step \(\text{Accept} \in \delta(q, u)\)
In each computation rewrite and restart steps alternate.

$M$ works in cycles:

$q_0 \vdash xuyz \leftarrow \ast \vdash xquyz \vdash \vdash xvq_1yz \leftarrow \ast \vdash xvyq_2z \leftarrow q_0 \vdash xvyz$

restarting configuration

Notation: $xuyz \vdash^c_M xvyz$. 

M. Kutrib, H. Messerschmidt, F. Otto () On Stateless Restarting Automata 9 / 24
Input configuration for \( w \in \Sigma^* : q_0 \$ w \$

\[ L(M) = \{ w \in \Sigma^* \mid M \text{ accepts on input } w \} \] is the input language of \( M \),

\[ L_C(M) = \{ x \in \Gamma^* \mid M \text{ accepts starting from } q_0 \$ x \$ \} \]

is the characteristic language of \( M \).

\( M \) is deterministic, if \( \delta \) is a function.

\( M \) is monotone, if in any computation of \( M \), the distance of the place of rewriting to the right delimiter \( \$ \) does not increase from one cycle to the next.
2. Restarting Automata

Input configuration for $w \in \Sigma^*$ : $q_0 \# w$

$L(M) = \{ w \in \Sigma^* \mid M$ accepts on input $w \}$ is the input language of $M$,

$L_C(M) = \{ x \in \Gamma^* \mid M$ accepts starting from $q_0 \# x \}$ is the characteristic language of $M$,

$M$ is deterministic, if $\delta$ is a function.

$M$ is monotone, if in any computation of $M$, the distance of the place of rewriting to the right delimiter $\#$ does not increase from one cycle to the next.
2. Restarting Automata

$M$ is an **RWW-automaton**, if each rewrite step is combined with a restart step.

$M$ is an **R(R)W-automaton**, if $\Sigma = \Gamma$ (i.e., no auxiliary symbols), and

$M$ is an **R(R)-automaton**, if $\nu$ is a (scattered) subword of $u$ for all $(q', \nu) \in \delta(q, u)$.

---

**Theorem 2** (Jančar, Mráz, Plátek, Vogel et al.)

(a) $\mathcal{L}($det-mon-$R(R)) = \mathcal{L}($det-mon-$R(R)WW) = DCFL$

(b) $\mathcal{L}(mon-R(R)WW) = CFL$

(c) $\mathcal{L}($det-$R(R)WW) = CRL \subsetneq GCSL \subsetneq \mathcal{L}(RWW) \subsetneq \mathcal{L}(RRWW)$
2. Restarting Automata

$M$ is an **RWW-automaton**, if each rewrite step is combined with a restart step.

$M$ is an **R(R)W-automaton**, if $\Sigma = \Gamma$ (i.e., no auxiliary symbols), and

$M$ is an **R(R)-automaton**, if $\nu$ is a (scattered) subword of $u$ for all $(q', \nu) \in \delta(q, u)$.

**Theorem 2** (Jančar, Mráz, Plátek, Vogel et al.)

| (a) $\mathcal{L}(\text{det-mon-R(R)}) = \mathcal{L}(\text{det-mon-R(R)WW}) = \text{DCFL}$ |
| (b) $\mathcal{L}(\text{mon-R(R)WW}) = \text{CFL}$ |
| (c) $\mathcal{L}(\text{det-R(R)WW}) = \text{CRL} \subsetneq \text{GCSL} \subsetneq \mathcal{L}(\text{RWW}) \subseteq \mathcal{L}(\text{RRWW})$ |
An RWW-autom. \( M = (Q, \Sigma, \Gamma, \phi, \$, q_0, k, \delta) \) is **stateless** if \( Q = \{ q_0 \} \).

Notation: \( M = (\Sigma, \Gamma, \phi, \$, k, \delta) \).

**Example**

\( \Sigma = \{ a, b \} \), \( \delta \) defined as follows:

- \( \delta(\phi \$) = \text{Accept} \), \( \delta(\phi \text{aaa}) = \text{MVR} \), \( \delta(\text{aaaa}) = \text{MVR} \),
- \( \delta(\phi \text{ab}\$) = \text{Accept} \), \( \delta(\phi \text{aab}) = \text{MVR} \), \( \delta(\text{aab}) = \text{MVR} \),
- \( \delta(\text{aabb}) = \text{ab} \).

\( M \) is a stateless R-automaton, it is deterministic and monotone, and 
\( L(M) = \{ a^n b^n \mid n \geq 0 \} \).
Proposition 3

Given a stateless RWW-automaton $M = (\Sigma, \Gamma, \phi, \$, k, \delta)$, a stateless RWW-automaton $M' = (\Sigma, \Gamma, \phi, \$, k + 1, \delta')$ can be constructed such that the following properties hold:

(1) $M'$ executes accept instructions only at the right end of the tape,
(2) $L_C(M') = L_C(M),$
(3) if all rewrite instructions of $M$ are deletions, then this also holds for $M'$, and
(4) if $M$ is deterministic, then so is $M'$. 
**Proposition 4**

$\text{REG} \subsetneq \mathcal{L}(\text{stl-det-mon-R})$

$L_d = \{ ca^n b^n | n \geq 0 \} \cup \{ da^n b^{2n} | n \geq 0 \}$

$L_d$ is a deterministic linear language, that is, it is accepted by a deterministic one-turn PDA.

**Lemma 5**

$L_d$ is not accepted by any stateless RW-automaton.
Proposition 4

\[ \text{REG} \subsetneq \mathcal{L}(\text{stl-det-mon-R}) \]

\[ L_d = \{ ca^n b^n \mid n \geq 0 \} \cup \{ da^n b^{2n} \mid n \geq 0 \} \]

\( L_d \) is a deterministic linear language, that is, it is accepted by a deterministic one-turn PDA.

Lemma 5

\( L_d \) is not accepted by any stateless RW-automaton.
3. Stateless RWW-Automata

\[ \mathcal{L}(\text{stl-wmon-RWW}) = GCSL \]

\[ \mathcal{L}(\text{stl-det-RWW}) \]

\[ \mathcal{L}(\text{stl-det-RW}) \]

\[ \mathcal{L}(\text{stl-det-R}) \]

\[ \mathcal{L}(\text{stl-det-mon-RWW}) \]

\[ \mathcal{L}(\text{stl-det-mon-RW}) \]

\[ \mathcal{L}(\text{stl-det-mon-R}) \]

\[ \mathcal{L}(\text{stl-det-RW}) \]

\[ \mathcal{L}(\text{stl-det-R}) \]

\[ \mathcal{L}(\text{stl-mon-RWW}) \]

\[ \mathcal{L}(\text{stl-mon-RW}) \]

\[ \mathcal{L}(\text{stl-mon-R}) \]

\[ \mathcal{L}(\text{stl-mon-RW}) \]

\[ \mathcal{L}(\text{stl-mon-R}) \]

\[ \mathcal{L}(\text{stl-mon-RW}) \]

\[ \mathcal{L}(\text{stl-mon-R}) \]

\[ \mathcal{L}(\text{stl-mon-RW}) \]

\[ \mathcal{L}(\text{stl-mon-R}) \]

\[ \mathcal{L}(\text{stl-mon-RW}) \]

\[ \mathcal{L}(\text{stl-mon-R}) \]

\[ \mathcal{L}(\text{stl-mon-RW}) \]

\[ \mathcal{L}(\text{stl-mon-R}) \]

\[ \mathcal{L}(\text{stl-mon-RW}) \]

\[ \mathcal{L}(\text{stl-mon-R}) \]
4. Stateless RRWW-Automata

Let $M = (\Sigma, \Gamma, \$, $, k, \delta)$ be a stateless RRWW-automaton. Any attempt to perform a second rewrite step within a cycle is interpreted as a reject step.

Lemma 6

(a) \{ $a^n b^n \mid n \geq 0$ \} \in \mathcal{L}(\text{stl-det-RR}).

(b) $L_{aba} = \{ a^m b^{m+n} a^n \mid m, n \geq 0 \} \notin \mathcal{L}(\text{stl-det-RRW}).$

Proof idea for (b):

$a^m b^m \in L_{aba} : \delta(a^\ell b^{k-\ell}) = a^{\ell-i} b^{k-\ell-i}$

$b^n a^n \in L_{aba} : \delta(b^{\ell'} a^{k-\ell'}) = b^{\ell'-i'} a^{k-\ell'-i'}$

$a^m b^{m+n} a^n \in L_{aba} : a^m b^{m+n} a^n \vdash a^{m-i} b^{m-i+n} a^n \vdash^*$

$a^{m-i} b^{m-i} b^n a^n \vdash a^{m-i} b^{m-i} b^{n-i'} a^{n-i'} : \text{Reject.}$
4. Stateless RRWW-Automata

Let $M = (\Sigma, \Gamma, \iota, \$, k, \delta)$ be a stateless RRWW-automaton. Any attempt to perform a second rewrite step within a cycle is interpreted as a reject step.

Lemma 6

(a) $\{ a^n b^n \mid n \geq 0 \} \in \mathcal{L}(\text{stl-det-RR}).$
(b) $L_{aba} = \{ a^m b^{m+n} a^n \mid m, n \geq 0 \} \notin \mathcal{L}(\text{stl-det-RRW}).$

Proof idea for (b):

$a^m b^m \in L_{aba} : \delta(a^\ell b^{k-\ell}) = a^{\ell-i} b^{k-\ell-i}$
$b^n a^n \in L_{aba} : \delta(b^{\ell'} a^{k-\ell'}) = b^{\ell'-i'} a^{k-\ell'-i'}$

$a^m b^{m+n} a^n \in L_{aba} : a^m b^{m+n} a^n \vdash a^{m-i} b^{m-i+n} a^n \vdash^* a^{m-i} b^{m-i} a^n \vdash a^{m-i} b^{m-i} b^{n-i'} a^{n-i'} : \text{Reject.}$
4. Stateless RRWW-Automata

Let $M = (\Sigma, \Gamma, \emptyset, \$, k, \delta)$ be a stateless RRWW-automaton. Any attempt to perform a second rewrite step within a cycle is interpreted as a reject step.

**Lemma 6**

(a) $\{ a^n b^n | n \geq 0 \} \in \mathcal{L}(\text{stl-det-RR})$.  

(b) $L_{aba} = \{ a^m b^{m+n} a^n | m, n \geq 0 \} \notin \mathcal{L}(\text{stl-det-RRW})$.

**Proof idea for (b):**

$\begin{align*}
a^m b^m & \in L_{aba} : \quad \delta(a^\ell b^{k-\ell}) = a^{\ell-i} b^{k-\ell-i} \\
b^n a^n & \in L_{aba} : \quad \delta(b^{\ell'} a^{k-\ell'}) = b^{\ell'-i'} a^{k-\ell'-i'} \\
a^m b^{m+n} a^n & \in L_{aba} : \quad a^{m-i} b^{m-i+n} a^n \vdash a^{m-i} b^{m-i} b^{n-i'} a^{n-i'} : \text{Reject}
\end{align*}$
Corollary 7

\[ \text{REG} \subsetneq \mathcal{L}(\text{stl-det-RR}) \]

Proof outline:

Let \( L \in \text{REG}(\Sigma) \), and let \( A = (Q, \Sigma, q_0, F, \varphi) \) be a complete DFA for \( L \). If \( m = |Q| \), then each word \( w \in \Sigma^m \) can be written as \( w = w_1 w_2 w_3 \) such that \( |w_2| \geq 1 \) and \( \varphi(q_0, w_1 w_2) = \varphi(q_0, w_1) \).

Hence, for all \( z \in \Sigma^* \), \( wz \in L \) iff \( w_1 w_3 z \in L \).

Define a stl-det-RR-automaton \( M \) on \( \Sigma \) with window size \( k = m + 1 \):

1. \( \delta(\$w\$) = \text{Accept} \) for all \( w \in \Sigma^m \cap L \),
2. \( \delta(\$w\$) = \$w_1 w_3 \) for all \( w \in \Sigma^m \), where \( w = w_1 w_2 w_3 \) is the chosen factorization of \( w \),
3. \( \delta(w) = \text{MVR} \) for all \( w \in \Sigma^{m+1} \),
4. \( \delta(w\$) = \text{RESTART} \) for all \( w \in \Sigma^m \).

Then \( M \) is monotone, and \( L(M) = L \). \( \square \)
Theorem 8

\[ \mathcal{L}(\text{stl-det-mon-RRWW}) = \text{DCFL} \text{ and } \mathcal{L}(\text{stl-mon-RRWW}) = \text{CFL}. \]

Proof outline:

(a) Let \( L \subseteq \Sigma^+ \) be context-free. Then \( L = L(P) \), where \( P = (Q, \Sigma, \Gamma, \delta, q_0, z_0) \) is a PDA without \( \lambda \)-transitions that accepts by empty pushdown.

(b) Here we can assume that \( L \) is accepted by a DPDA \( P \) for which each \( \lambda \)-transition simply pops a symbol from the pushdown [Autebert, Berstel, Boasson 97].

Theorem 9

\[ \mathcal{L}(\text{stl-det-RRWW}) = \text{CRL}. \]
Theorem 8

\[ \mathcal{L}(\text{stl-det-mon-RRWW}) = \text{DCFL} \quad \text{and} \quad \mathcal{L}(\text{stl-mon-RRWW}) = \text{CFL}. \]

Proof outline:

(a) Let \( L \subseteq \Sigma^+ \) be context-free. Then \( L = L(P) \), where 
\( P = (Q, \Sigma, \Gamma, \delta, q_0, z_0) \) is a PDA without \( \lambda \)-transitions that accepts by empty pushdown.

(b) Here we can assume that \( L \) is accepted by a DPDA \( P \) for which each \( \lambda \)-transition simply pops a symbol from the pushdown [Autebert, Berstel, Boasson 97].

Theorem 9

\[ \mathcal{L}(\text{stl-det-RRWW}) = \text{CRL}. \]
A stateless two-phase RRWW-automaton (stl-2-RRWW) is described by a 7-tuple $M = (\Sigma, \Gamma, \phi, $, $, k, \delta_1, \delta_2)$:

$\delta_1$ : transition relation for the first phase of a cycle,
$\delta_2$ : transition relation for the second phase of a cycle.

**Lemma 10**

(a) $\mathcal{L}(\text{stl-(det)-R}(W)(W)) \subseteq \mathcal{L}(\text{stl-(det)-2-RR}(W)(W))$

(b) $\mathcal{L}(\text{stl-(det)-RR}(W)(W)) \subseteq \mathcal{L}(\text{stl-(det)-2-RR}(W)(W))$
A **stateless two-phase RRWW-automaton** (stl-2-RRWW) is described by a 7-tuple $M = (\Sigma, \Gamma, \phi, \$, $k, \delta_1, \delta_2)$:

- $\delta_1$: transition relation for the first phase of a cycle,
- $\delta_2$: transition relation for the second phase of a cycle.

**Lemma 10**

(a) $\mathcal{L}(\text{stl-(det)-R(W)(W)}) \subseteq \mathcal{L}(\text{stl-(det)-2-RR(W)(W)})$

(b) $\mathcal{L}(\text{stl-(det)-RR(W)(W)}) \subseteq \mathcal{L}(\text{stl-(det)-2-RR(W)(W)})$
Theorem 11

(a) \( \mathcal{L}(\text{stl-det-mon-2-RRWW}) = \text{DCFL} \)

(b) \( \mathcal{L}(\text{stl-mon-2-RRWW}) = \text{CFL} \)

(c) \( \mathcal{L}(\text{stl-det-2-RRWW}) = \text{CRL} \)

\[ L_d = \{ ca^n b^n \mid n \geq 0 \} \cup \{ da^n b^{2n} \mid n \geq 0 \} \not\in \mathcal{L}(\text{stl-2-RRW}) \]

Corollary 12

\( \mathcal{L}(\text{stl-(det)-2-RR(W)}) \subset \mathcal{L}((\text{det})-\text{RR(W)}) \)

\[ L_{\text{expo}}^{(1)} = \{ a^{2^n} \mid n \geq 0 \} \cup \{ a^i b a^j \mid i, j \geq 0, \exists m \geq 1 : i + 2 \cdot j = 2^m \} \]

Lemma 13

\( L_{\text{expo}}^{(1)} \in \mathcal{L}(\text{stl-det-2-RRW}) \)
Theorem 11

(a) \( \mathcal{L}(\text{stl-det-mon-2-RRWW}) = \text{DCFL} \)
(b) \( \mathcal{L}(\text{stl-mon-2-RRWW}) = \text{CFL} \)
(c) \( \mathcal{L}(\text{stl-det-2-RRWW}) = \text{CRL} \)

\[ L_d = \{ ca^n b^n \mid n \geq 0 \} \cup \{ da^n b^{2n} \mid n \geq 0 \} \not\in \mathcal{L}(\text{stl-2-RRW}) \]

Corollary 12

\( \mathcal{L}(\text{stl-(det)-2-RR(W)}) \subset \mathcal{L}((\text{det})-\text{RR(W)}) \)

\[ L_{\text{expo}}^{(1)} = \{ a^{2^n} \mid n \geq 0 \} \cup \{ a^i b a^i \mid i, j \geq 0, \exists m \geq 1 : i + 2 \cdot j = 2^m \} \]

Lemma 13

\( L_{\text{expo}}^{(1)} \in \mathcal{L}(\text{stl-det-2-RRW}) \)
Theorem 11

(a) $\mathcal{L}(\text{stl-det-mon-2-RRWW}) = \text{DCFL}$
(b) $\mathcal{L}(\text{stl-mon-2-RRWW}) = \text{CFL}$
(c) $\mathcal{L}(\text{stl-det-2-RRWW}) = \text{CRL}$

$L_d = \{ ca^n b^n \mid n \geq 0 \} \cup \{ da^n b^{2n} \mid n \geq 0 \} \not\in \mathcal{L}(\text{stl-2-RRW})$

Corollary 12

$\mathcal{L}(\text{stl-(det)-2-RR(W)}) \subset \mathcal{L}((\text{det})-\text{RR(W)})$

$L_{\text{expo}}^{(1)} = \{ a^{2n} \mid n \geq 0 \} \cup \{ a^i b a^i \mid i, j \geq 0, \exists m \geq 1 : i + 2 \cdot j = 2^m \}$

Lemma 13

$L_{\text{expo}}^{(1)} \in \mathcal{L}(\text{stl-det-2-RRW})$
Lemma 14

\[ L^{(1)}_{\text{expo}} \notin \mathcal{L}(\text{stl-RW}) \cup \mathcal{L}(\text{stl-det-RRW}) \]

Proof idea:

Let \( M \) be a stl-RW-automaton for \( L^{(1)}_{\text{expo}} \):

If \( M \) accepts \( ba^{2^n} \) in a tail, then it also accepts \( ba^{2^n+1} \notin L^{(1)}_{\text{expo}} \).

If \( ba^{2^n} \models_{M} w \), then \( w = a^{2^n} \), and \( ba^{2^n} ba^{2^n-1} \models_{M} a^{2^n} ba^{2^n-1} \in L^{(1)}_{\text{expo}} \).

Let \( M \) be a stl-det-RRW-automaton for \( L^{(1)}_{\text{expo}} \):

If \( M \) accepts \( ba^{2^n} \) in a tail, then it also accepts \( ba^{2^n+1} \notin L^{(1)}_{\text{expo}} \).

If \( ba^{2^n} \models_{M} w \), then \( w = a^{2^n} \), and \( M \) performs a restart on \( a^{2^n-k} \).

Also \( a^{2^n} \models_{M} a^{2^n-2i} ba^{i} \), i.e., \( M \) performs a rewrite on \( a^{2^n-k} \).

This contradicts the determinism of \( M \).
Lemma 14

\[ L_{\text{expo}}^{(1)} \not\in \mathcal{L}(\text{stl-RW}) \cup \mathcal{L}(\text{stl-det-RRW}) \]

Proof idea:

Let \( M \) be a stl-RW-automaton for \( L_{\text{expo}}^{(1)} \):

If \( M \) accepts \( ba^{2^n} \) in a tail, then it also accepts \( ba^{2^n+1} \not\in L_{\text{expo}}^{(1)} \).

If \( ba^{2^n} \vdash_{M} w \), then \( w = a^{2^n} \), and \( ba^{2^n} ba^{2^n-1} \vdash_{M} a^{2^n} ba^{2^n-1} \in L_{\text{expo}}^{(1)} \).

Let \( M \) be a stl-det-RRW-automaton for \( L_{\text{expo}}^{(1)} \):

If \( M \) accepts \( ba^{2^n} \) in a tail, then it also accepts \( ba^{2^n+1} \not\in L_{\text{expo}}^{(1)} \).

If \( ba^{2^n} \vdash_{M} w \), then \( w = a^{2^n} \), and \( M \) performs a restart on \( a^{2^n-k} \).

Also \( a^{2^n} \vdash_{M} a^{2^n-2i} ba^i \), i.e., \( M \) performs a rewrite on \( a^{2^n-k} \).

This contradicts the determinism of \( M \).
Lemma 14

\[ L^{(1)}_{\text{expo}} \notin \mathcal{L}(\text{stl-RW}) \cup \mathcal{L}(\text{stl-det-RRW}) \]

Proof idea:

Let \( M \) be a stl-RW-automaton for \( L^{(1)}_{\text{expo}} \):

If \( M \) accepts \( ba^{2n} \) in a tail, then it also accepts \( ba^{2n+1} \notin L^{(1)}_{\text{expo}} \).

If \( ba^{2n} \vdash_M w \), then \( w = a^{2n} \), and \( ba^{2n} \vdash_M a^{2n}ba^{2n-1} \in L^{(1)}_{\text{expo}} \).

Let \( M \) be a stl-det-RRW-automaton for \( L^{(1)}_{\text{expo}} \):

If \( M \) accepts \( ba^{2n} \) in a tail, then it also accepts \( ba^{2n+1} \notin L^{(1)}_{\text{expo}} \).

If \( ba^{2n} \vdash_M w \), then \( w = a^{2n} \), and \( M \) performs a restart on \( a^{2n-k} \).

Also \( a^{2n} \vdash_M a^{2n-2i}ba^i \), i.e., \( M \) performs a rewrite on \( a^{2n-k} \).

This contradicts the determinism of \( M \).
Lemma 13 and Lemma 14 imply the following.

**Corollary 15**

(a) \( \mathcal{L}(\text{stl-det-RW}) \subsetneq \mathcal{L}(\text{stl-det-2-RRW}) \)

(b) \( \mathcal{L}(\text{stl-det-RRW}) \subsetneq \mathcal{L}(\text{stl-det-2-RRW}) \)

**Corollary 16**

(a) \( \mathcal{L}(\text{stl-det-R}) \subsetneq \mathcal{L}(\text{stl-det-2-RR}) \)

(b) \( \mathcal{L}(\text{stl-det-RR}) \subsetneq \mathcal{L}(\text{stl-det-2-RR}) \)

**Lemma 17**

\( \overline{L}_{\text{expo}}^{(1)} := \{a, b\}^* \setminus L_{\text{expo}}^{(1)} \) is not accepted by any stateless deterministic 2-RRW-automaton that executes accept instructions only at the right end of the tape.
Lemma 13 and Lemma 14 imply the following.

**Corollary 15**

(a) \( \mathcal{L}(\text{stl-det-RW}) \subsetneq \mathcal{L}(\text{stl-det-2-RRW}) \)

(b) \( \mathcal{L}(\text{stl-det-RRW}) \subsetneq \mathcal{L}(\text{stl-det-2-RRW}) \)

**Corollary 16**

(a) \( \mathcal{L}(\text{stl-det-R}) \subsetneq \mathcal{L}(\text{stl-det-2-RR}) \)

(b) \( \mathcal{L}(\text{stl-det-RR}) \subsetneq \mathcal{L}(\text{stl-det-2-RR}) \)

**Lemma 17**

\( \overline{L}_{\text{expo}}^{(1)} := \{a, b\}^* \setminus L_{\text{expo}}^{(1)} \) is not accepted by any stateless deterministic 2-RRW-automaton that executes accept instructions only at the right end of the tape.
Lemma 13 and Lemma 14 imply the following.

**Corollary 15**

(a) \( \mathcal{L}(\text{stl-det-RW}) \subsetneq \mathcal{L}(\text{stl-det-2-RRW}) \)

(b) \( \mathcal{L}(\text{stl-det-RRW}) \subsetneq \mathcal{L}(\text{stl-det-2-RRW}) \)

**Corollary 16**

(a) \( \mathcal{L}(\text{stl-det-R}) \subsetneq \mathcal{L}(\text{stl-det-2-RR}) \)

(b) \( \mathcal{L}(\text{stl-det-RR}) \subsetneq \mathcal{L}(\text{stl-det-2-RR}) \)

**Lemma 17**

\( \bar{L}_{\text{expo}}^{(1)} := \{a, b\}^* \setminus L_{\text{expo}}^{(1)} \) is not accepted by any stateless deterministic 2-RRW-automaton that executes accept instructions only at the right end of the tape.
6. Concluding Remarks

The following (non-) closure properties have been obtained for the language families that are specified by stateless deterministic restarting automata:

<table>
<thead>
<tr>
<th>Language Family</th>
<th>Operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{L}$(stl-det-R)</td>
<td>$\cup$</td>
</tr>
<tr>
<td>$\mathcal{L}$(stl-det-RW)</td>
<td>$\cap$</td>
</tr>
<tr>
<td>$\mathcal{L}$(stl-det-RR)</td>
<td>$\sim$</td>
</tr>
<tr>
<td>$\mathcal{L}$(stl-det-RRW)</td>
<td>$\cap_{reg}$</td>
</tr>
<tr>
<td>$\mathcal{L}$(stl-det-2-RR)</td>
<td>$R$</td>
</tr>
<tr>
<td>$\mathcal{L}$(stl-det-2-RRW)</td>
<td>$\cdot$</td>
</tr>
<tr>
<td></td>
<td>$*$</td>
</tr>
<tr>
<td></td>
<td>$h^{-1}$</td>
</tr>
<tr>
<td></td>
<td>$h_\lambda$</td>
</tr>
</tbody>
</table>

The table shows the closure properties for the specified language families.
Open Problems:

■ What is the exact relationship between $\mathcal{L}(stl\text{-}det\text{-}R(W))$ and $\mathcal{L}(stl\text{-}det\text{-}RR(W))$?

■ Is the expressive power of stl-RRWW-automata changed, if we require them to perform accept steps only at the right end of the tape?

■ Closure properties for language families accepted by non-deterministic types of stateless restarting automata?

■ What is the expressive power of stateless RRWW-automata that are allowed to perform several rewrite steps per cycle?