LIMITED CONTEXT RESTARTING AUTOMATA AND MCNAUGHTON FAMILIES OF LANGUAGES

Friedrich Otto
Peter Černo, František Mráz
Introduction

• **Part I**: Introduction,
• **Part II**: Clearing and $\Delta$-Clearing Restarting Automata,
• **Part III**: Limited Context Restarting Automata,
• **Part IV**: Confluent Limited Context Restarting Automata,
• **Part V**: Concluding Remarks.
Part I: Introduction

- **Restarting Automata**:  
  - Model for the linguistic technique of *analysis by reduction*.  
  - Many different types have been defined and studied intensively.

- **Analysis by Reduction**:  
  - Method for checking [non-]correctness of a sentence.  
  - Iterative application of simplifications.  
  - Until the input cannot be simplified anymore.

- **Restricted Models**:  
  - Clearing, Δ-Clearing and Δ*-Clearing Restarting Automata,  
  - Limited Context Restarting Automata.
Part II: Clearing Restarting Automata

• Let $k$ be a nonnegative integer.

• **$k$–context rewriting system ($k$-CRS)**

• Is a triple $M = (\Sigma, \Gamma, I)$:
  - $\Sigma$ … *input alphabet*, $\emptyset, \$ \notin \Sigma$,
  - $\Gamma$ … *working alphabet*, $\Gamma \supseteq \Sigma$,
  - $I$ … finite set of *instructions* $(x, z \rightarrow t, y)$:
    - $x \in \{\emptyset, \lambda\}.\Gamma^*$, $|x| \leq k$ (left context)
    - $y \in \Gamma^*.\{\emptyset, \}$, $|y| \leq k$ (right context)
    - $z \in \Gamma^+, z \neq t \in \Gamma^*$.
  - $\emptyset$ and $\$ … *sentinels.*
Rewriting

- $u_z v \vdash_M u t v$ iff $\exists (x, z \to t, y) \in I$:
- $x$ is a suffix of $\cdot u$ and $y$ is a prefix of $v$.

- $L(M) = \{w \in \Sigma^* / w \vdash^*_M \lambda\}$.
- $L_C(M) = \{w \in \Gamma^* / w \vdash^*_M \lambda\}$. 
Empty Word

- **Note**: For every \( k\text{-CRS } M: \lambda \vdash^* M \lambda \), hence \( \lambda \in L(M) \).
- Whenever we say that a \( k\text{-CRS } M \) recognizes a language \( L \), we always mean that \( L(M) = L \cup \{\lambda\} \).
- We simply **ignore the empty word** in this setting.
Clearing Restarting Automata

• **$k$–Clearing Restarting Automaton** ($k$-cl-RA)
  - Is a $k$-CRS $M = (\Sigma, \Sigma, I)$ such that:
  - For each $(x, z \rightarrow t, y) \in I$: $z \in \Sigma^+$, $t = \lambda$.

• **$k$–$\Delta$–Clearing Restarting Automaton** ($k$-$\Delta$-cl-RA)
  - Is a $k$-CRS $M = (\Sigma, \Gamma, I)$ such that:
    - $\Gamma = \Sigma \cup \{\Delta\}$ where $\Delta$ is a new symbol, and
    - For each $(x, z \rightarrow t, y) \in I$: $z \in \Gamma^+$, $t \in \{\lambda, \Delta\}$.

• **$k$–$\Delta^*$–Clearing Restarting Automaton** ($k$-$\Delta^*$-cl-RA)
  - Is a $k$-CRS $M = (\Sigma, \Gamma, I)$ such that:
    - $\Gamma = \Sigma \cup \{\Delta\}$ where $\Delta$ is a new symbol, and
    - For each $(x, z \rightarrow t, y) \in I$: $z \in \Gamma^+$, $t = \Delta^i$, $0 \leq i \leq |z|$.
Example 1

- \( L_1 = \{a^n b^n \mid n > 0\} \cup \{\lambda\} \):
- 1-cl-RA \( M = (\{a, b\}, I) \),
- Instructions \( I \) are:
  - \( R1 = (a, ab \rightarrow \lambda, b) \),
  - \( R2 = (\$, ab \rightarrow \lambda, $) \).

- Note:
  - We did not use \( \Delta \).
Example 2

- \( L_2 = \{a^n cb^n \mid n > 0\} \cup \{\lambda\} \):
- 1-\(\Delta\)-cl-RA \( M = (\{a, b, c\}, I) \),
- **Instructions** \( I \) are:
  - \( R1 = (a, c \rightarrow \Delta, b) \),
  - \( R2 = (a, a\Delta b \rightarrow \Delta, b) \),
  - \( R3 = (\$, a\Delta b \rightarrow \lambda, \$) \).

**Note:**
- We **must use** \( \Delta \).
Clearing Restarting Automata

- **Clearing Restarting Automata:**
  - Accept all regular and even some non-context-free languages.
  - They do not accept all context-free languages ($\{a^n cb^n | n > 0\}$).

- **$\Delta$-Clearing and $\Delta^*$-Clearing Restarting Automata:**
  - Accept all context-free languages.
  - The exact expressive power remains open.

- Here we establish an upper bound by showing that **Clearing, $\Delta$- and $\Delta^*$-Clearing Restarting Automata** only accept languages that are growing context-sensitive [Dahlhaus, Warmuth].
Clearing Restarting Automata

**Theorem:** $\mathcal{L}(\Delta^*-cl$-$RA) \subseteq GCSL$.

**Proof.**
- Let $M = (\Sigma, \Gamma, I)$ be a $k$-$\Delta^*$-$cl$-$RA$ for some $k \geq 0$.
- Let $\Omega = \Gamma \cup \{\$, $\#, Y\}$, where $Y$ is a new letter.
- Let $S(M)$ be the following string-rewriting system over $\Omega$:
  \[ S(M) = \{ xzy \rightarrow xty \mid (x, z \rightarrow t, y) \in I \} \cup \{ \$ \$ \rightarrow Y \} \]
- Let $g$ be a weight function: $g(\Delta) = 1$ and $g(a) = 2$ for all $a \neq \Delta$.

**Claim:** $L(M)$ coincides with the McNaughton language [Beaudry, Holzer, Niemann, Otto] specified by $(S(M), \$, $\#, Y)$.

As $S(M)$ is a finite weight-reducing system, it follows that the McNaughton language $L(M)$ is a growing context-sensitive language, that is, $L(M) \in GCSL$. ■
Clearing Restarting Automata
Part III: Limited Context RA

- **Limited Context Restarting Automaton (lc-RA):**
  - Is defined exactly as *Context Rewriting Systems*, except that:
  - There is *no upper bound* $k$ on the length of contexts.
  - The instructions are usually written as: $(x/z \rightarrow t/y)$.
  - There is a *weight function* $g$ such that $g(z) > g(t)$ for all instructions $(x/z \rightarrow t/y)$ of the automaton.
Limited Context Restarting Automata

- **Restricted types:** \( lc\text{-}RA M = (\Sigma, \Gamma, I) \) is of **type**:
  - \( \mathcal{R}_0' \), if \( I \) is an arbitrary finite set of (weight-reducing) instructions,
  - \( \mathcal{R}_1' \), if \( |t| \leq 1 \),
  - \( \mathcal{R}_2' \), if \( |t| \leq 1 \), \( x \in \{\emptyset, \lambda\}, y \in \{\lambda, \$\} \),
  - \( \mathcal{R}_3' \), if \( |t| \leq 1 \), \( x \in \{\emptyset, \lambda\}, y = $ \), for all \( (x/z \rightarrow t/y) \in I \).

- **Moreover,** \( lc\text{-}RA M = (\Sigma, \Gamma, I) \) is of **type**:
  - \( \mathcal{R}_0 \), (\( \mathcal{R}_1, \mathcal{R}_2, \mathcal{R}_3 \), respectively) if it is of type:
  - \( \mathcal{R}_0' \), (\( \mathcal{R}_1', \mathcal{R}_2', \mathcal{R}_3' \), respectively) and all instructions of \( M \) are length-reducing (i.e. \( |z| > |t| \) for all \( (x/z \rightarrow t/y) \in I \)).

- We use the notation \( lc\text{-}RA[\mathcal{R}_i'], lc\text{-}RA[\mathcal{R}_i] \) to denote the corresponding **class** of the restricted \( lc\text{-}RA \)s.
**lc-RA[\mathcal{R}_0']** and **lc-RA[\mathcal{R}_0]**

**Theorem:** \( \mathcal{L}(lc-RA[\mathcal{R}_0']) = \mathcal{L}(lc-RA[\mathcal{R}_0]) = GCSL \).

**Proof.**

- For each lc-RA \( M = (\Sigma, I, I) \) we can associate a **finite weight-reducing string-rewriting system** \( S(M) \) such that \( L(M) \) is the **McNaughton language** specified by the four-tuple \( (S(M), \$0, \$, Y) \).

\[
S(M) = \{ xzy \rightarrow xty \mid (x \mid z \rightarrow t \mid y) \in I \} \cup \{ \$_0 \rightarrow Y \}.
\]

- It follows that \( L(M) \in GCSL \).

- **On the other hand**, each growing context-sensitive language is accepted by an lc-RA[\mathcal{R}_0].}
Theorem: \( \mathcal{L}(\text{lR-RA}[\mathcal{R}_1']) = \text{GCSL} \).

Proof.

Let \( G = (N, T, S, P) \) be a *weight-increasing context-sensitive grammar*. By taking:

\[
I(G) = \{(u | x \rightarrow A | v) | (uAv \rightarrow uxv) \in P\} \cup \{(\$ | r \rightarrow \lambda | \$) | (S \rightarrow r) \in P\},
\]

we obtain an \( \text{lR-RA}[\mathcal{R}_1'] M(G) = (T, N \cup T, I(G)) \) such that

\[
L(M(G)) = L(G) \cup \{\lambda\}.
\]

The class of languages generated by *weight-increasing context-sensitive grammars*, which is known as the class \( \text{ACSL} (\text{acyclic context-sensitive languages}) \), coincides with the class \( \text{GCSL} [\text{Niemann, Woinowski}] \).

Thus, \( \mathcal{L}(\text{lR-RA}[\mathcal{R}_1']) \supseteq \text{GCSL} \).
Theorem: $\mathcal{L}(lc\text{-}RA[\mathcal{R}_1]) = \text{GACSL}$.

Proof.

Let $lc\text{-}RA M = (\Sigma, \Gamma, I)$ be of type $\mathcal{R}_1$.

For all $(x/z \to t/y) \in I : |z| > |t|$ and $|t| \leq 1$.

Lemma: It is possible to obtain an equivalent $lc\text{-}RA M$ such that:

For all $(x/z \to t/y) \in I : |z| > |t|$ and $|t| = 1$ if $x \neq \$ or $y \neq $.

From string-rewriting system: $R = \{ xty \to xzy | (x/z \to t/y) \in I \}$.

We construct a length-increasing context-sensitive grammar:

$G = (I, \Sigma, S, R)$ such that $L(G) = \$ . L(M) . $.

The class of languages generated by length-increasing context-sensitive grammars is known as the class $\text{GACSL (growing acyclic context-sensitive languages)}$. $\text{GACSL} \subseteq \text{ACSL} = \text{GCSL}$.

$\$ . L(M) . $ \in \text{GACSL}$, i.e. $L(M) \in \text{GACSL}$ [Buntrock]. Similarly $\supseteq$.
lc-RA[$\mathcal{R}_2'$] and lc-RA[$\mathcal{R}_2$]

**Theorem:** $\mathcal{L}$(lc-RA[$\mathcal{R}_2'$]) = $\mathcal{L}$(lc-RA[$\mathcal{R}_2$]) = CFL.

**Proof.**

1. Let lc-RA $M = (\Sigma, \Gamma, I)$ be of type $\mathcal{R}_2'$.
2. For all $(x/z \to t/y) \in I : |t| \leq 1$, $x \in \{\epsilon, \lambda\}$, $y \in \{\lambda, \$\}$.
3. We split $R(M) = \{ xzy \to xty / (x/z \to t/y) \in I \}$ into 4 subsystems:

   (a) $R_{bif} = \{ \epsilon x\$ \to \epsilon y\$ | (\epsilon | x \to y | \$) \in I \}$, the bifix rules of $R(M)$,
   (b) $R_{pre} = \{ \epsilon x \to \epsilon y | (\epsilon | x \to y | \lambda) \in I \}$, the prefix rules of $R(M)$,
   (c) $R_{suf} = \{ x\$ \to y\$ | (\lambda | x \to y | \$) \in I \}$, the suffix rules of $R(M)$,
   (d) $R_{inf} = \{ x \to y | (\lambda | x \to y | \lambda) \in I \}$, the infix rules of $R(M)$.

4. Take $A(M) = \{ \alpha \in \Gamma^* | \epsilon \alpha \$ \in \text{dom}(R_{bif}) \text{ and } \epsilon \alpha \$ \Rightarrow^*_{R(M)} \epsilon \$ \}$
5. Then $A(M)$ is a finite set. Let $R' = R_{pre} \cup R_{suf} \cup R_{inf}$. Then $L(M) = \{ w \in \Sigma^* | \epsilon w\$ \Rightarrow^*_{R(M)} \epsilon \$ \} = \{ w \in \Sigma^* | \exists \alpha \in A(M) \cup \{\lambda\} : \epsilon w\$ \Rightarrow^*_{R'} \epsilon \alpha \$ \}$
Proof. (Continued).

Consider a **mixed rewriting system**: \( P(M) = P_{pre} \cup P_{suf} \cup P_{inf} \)
- **Prefix-rewriting system**: \( P_{pre} = \{ x \rightarrow y \mid (\exists x \rightarrow \exists y) \in R_{pre} \} \)
- **Suffix-rewriting system**: \( P_{suf} = \{ x \rightarrow y \mid (x\$ \rightarrow y\$) \in R_{suf} \} \)
- **String-rewriting system**: \( P_{inf} = R_{inf} \)

The rules of a **prefix-rewriting system** (**suffix-rewriting system**) are only applied to the **prefix** (**suffix**) of a word.

**Apparently**: \( L(M) = \nabla_{P(M)}^*(A(M) \cup \{\lambda\}) \cap \Sigma^* \)

As \( P(M) \) only contains **generalized monadic rules**, it follows that the language \( L(M) \) is **context-free** [Leupold, Otto].

Moreover, it is easy to obtain from a given **context-free grammar** an equivalent \( lc\text{-}RA M = (\Sigma, \Gamma, I) \) of the type \( \mathcal{R}_2 \).

Thus we have: \( CFL \subseteq \mathcal{L}(lc\text{-}RA[\mathcal{R}_2]) \subseteq \mathcal{L}(lc\text{-}RA[\mathcal{R}_2']) \subseteq CFL \).
**Theorem:** \( \mathcal{L}(lc\text{-}RA[\mathcal{R}_3']) = \mathcal{L}(lc\text{-}RA[\mathcal{R}_3]) = \text{REG} \).

**Proof.**

- Let \( lc\text{-}RA M = (\Sigma, \Gamma, I) \) be of type \( \mathcal{R}_3' \).
- For all \( (x \mid z \rightarrow t \mid y) \in I : |t| \leq 1, x \in \{\emptyset, \lambda\}, y = \$ \).
- We split \( R(M) = \{ xzy \rightarrow xty \mid (x \mid z \rightarrow t \mid y) \in I \} \) into 2 subsystems:
  (a) \( R_{bif} = \{ \emptyset x\$ \rightarrow \emptyset y\$ \mid (\emptyset \mid x \rightarrow y \mid \$) \in I \} \), the bifix rules of \( R(M) \),
  (b) \( R_{suf} = \{ x\$ \rightarrow y\$ \mid (\lambda \mid x \rightarrow y \mid \$) \in I \} \), the suffix rules of \( R(M) \).
- Now we take only the suffix-rewriting system \( P(M) = P_{suf} \), where:
  \( P_{suf} = \{ y \rightarrow x \mid (x\$ \rightarrow y\$) \in R_{suf} \} \)
- **Apparently:** \( L(M) = \Delta^*_{P(M)}(A(M) \cup \{\lambda\}) \cap \Sigma^* \) is **regular**.
- Again, it is easy to obtain from a given **regular grammar** an equivalent \( lc\text{-}RA M = (\Sigma, \Gamma, I) \) of the type \( \mathcal{R}_3 \).
- Thus we have: \( \text{REG} \subseteq \mathcal{L}(lc\text{-}RA[\mathcal{R}_3]) \subseteq \mathcal{L}(lc\text{-}RA[\mathcal{R}_3']) \subseteq \text{REG} \). ■
Limited Context Restarting Automata

- **Hierarchy of Language Classes:**

\[
\mathcal{L}(lc-R[\mathcal{R}_0]) = \mathcal{L}(lc-R[\mathcal{R}'_0]) = GCSL
\]
\[
\mathcal{L}(lc-R[\mathcal{R}'_1]) \uparrow ? \downarrow \mathcal{L}(lc-R[\mathcal{R}_1]) = GACSL
\]
\[
\mathcal{L}(lc-R[\mathcal{R}_2]) = \mathcal{L}(lc-R[\mathcal{R}'_2]) = CFL
\]
\[
\mathcal{L}(lc-R[\mathcal{R}_3]) = \mathcal{L}(lc-R[\mathcal{R}'_3]) = REG
\]
Part IV: Confluent lc-RA

- Since *lc-RA* *M* is a **nondeterministic** device, it is **difficult** to decide the membership in *L(M)*.
- Here we are interested in *lc-RA* *M* = (*Σ*, *Γ*, *I*) for which **all computations** from *¢ w $* lead to *¢ $*, if *w* ∈ *L(M)*.
- The **reduction relation** *⊢* *M* corresponds to the **single-step reduction relation** *⇒* *R(M)* induced by the **string-rewriting system** *R(M) = \{ xzy → xty | (x | z → t | y) ∈ I \}* on *¢ Γ* $*$.
- As it is **undecideable** whether *R(M)* is confluent on the congruence class *[¢ $]* *R(M)\]*, we consider only **confluence**.
- An *lc-RA* *M* = (*Σ*, *Γ*, *I*) is called **confluent** if the corresponding **string-rewriting system** *R(M)* is **confluent**.
- We use the **prefix** con- to denote **confluent lc-RA**.
lc-RA[\textit{con-}\mathcal{R}_0'] and lc-RA[\textit{con-}\mathcal{R}_0]

**Theorem:** \(\mathcal{L}(\text{lc-RA[con-}\mathcal{R}_0']) = \mathcal{L}(\text{lc-RA[con-}\mathcal{R}_0]) = \text{CRL}.\)

**Proof.**

- For each \textit{lc-RA[con-}\mathcal{R}_0'] \(M = (\Sigma, \Gamma, I) : S(M) = R(M) \cup \{ \text{}\rightarrow Y \} \) is a \textit{finite weight-reducing string-rewriting system} that is \textit{confluent}.
- \(L(M)\) is the \textit{McNaughton language} specified by \((S(M), \text{} , $, Y)\), i.e.
- \(L(M)\) is a \textit{Church-Rosser language} [McNaughton, Narendran, Otto].
- On the other hand, each \textit{Church-Rosser language} \(L\) is accepted by a \textit{length-reducing deterministic two-pushdown automaton} \(A\) [Niemann, Otto].
- Based on \(A\) it is possible to construct a \textit{confluent lc-RA} of type \(\mathcal{R}_0\) recognizing the language \(L\). ■
lc-RA[ con-ℛ₃’] and lc-RA[ con-ℛ₃]

**Theorem:** \( \mathcal{L}(lc-RA[\text{con-ℛ₃’}]) = \mathcal{L}(lc-RA[\text{con-ℛ₃}]) = \text{REG} \).

**Proof.**

- Apparently, \( \mathcal{L}(lc-RA[\text{con-ℛ₃’}]) \subseteq \mathcal{L}(lc-RA[\text{ℛ₃’}]) = \text{REG} \).

- Conversely, if \( L \subseteq \Sigma^* \) is regular then there exists DFA \( A = (Q, \Sigma, q_0, F, \delta) \) that accepts \( L^R \). We define \( lc-RA\ M = (\Sigma, \Sigma \cup Q, I) \), where \( I = \{ (\emptyset | ab \to q | \lambda) | \delta(q_0, ab) = q \} \cup \{ (\emptyset | qa \to q’ | \lambda) | \delta(q, a) = q’ \} \cup \{ (\emptyset | q \to \lambda | $) | q \in F \} \cup \{ (\emptyset | a \to \lambda | $) | a \in \Sigma \cap L^R \} \).

- It is easy to see that \( L(M) = L^R \), and that the string-rewriting system \( R(M) \) is confluent. By taking \( M’ = (\Sigma, \Sigma \cup Q, I’) \), where:
  \[
  I’ = \{ (\lambda | u^R \to v^R | $) \} \cup \{ (\emptyset | u \to v | \lambda) \in I \} \cup \{ (\emptyset | u^R \to v^R | $) \} \cup \{ (\emptyset | u \to v | $) \in I \},
  \]

- We obtain a confluent lc-RA of type \( \text{ℛ₃} \) that accepts \( L \). ■
lc-RA[ con-ℛ₂’] and lc-RA[ con-ℛ₂]

- For other classes we have no characterization results.
- We have only some preliminary results.
- **Lemma**: \( \mathcal{L}(lc-RA[ con-ℛ₂’]) \subseteq DCFL \cap DCFL^R. \)
- **Proof Idea**.
  - Consider the leftmost derivation, which can be realized by a deterministic pushdown automaton. ■
- **Lemma**: The deterministic context-free language
  \[
  L_u = \{ ca^n b^n c \mid n \geq 1 \} \cup \{ da^m b^{2m} d \mid m \geq 1 \}
  \]
  is not accepted by any lc-RA[ con-ℛ₂’].
- **Note**: Both \( L_u \) and \( L_u^R \) are DLIN languages.
- **Corollary**: \( \mathcal{L}(lc-RA[ con-ℛ₂’]) \subset DCFL \cap DCFL^R. \)
**lc-RA[ con-\(\mathcal{R}_2\)’] and lc-RA[ con-\(\mathcal{R}_2\)]**

**Lemma**: The nonlinear language \(\{a^n b^n c^m d^m \mid n, m \geq 1\}\) is accepted by a **confluent lc-RA of type \(\mathcal{R}_2\)**.

**Corollary**: The class of languages accepted by **confluent lc-RA of type \(\mathcal{R}_2\)’** is **incomparable** to **DLIN** and **LIN**.

• These results also hold for the class of languages that are accepted by **lc-RA[ con-\(\mathcal{R}_2\)]**.

• The exact relationship of these classes of languages to the class of confluent **[generalized]** monadic McNaughton languages [Leupold, Otto] remains open.
Lemma: The language $L_{\text{expo5}} = \{ a^{5n} | n \geq 0 \}$ is accepted by an $lc{-}\text{RA}[\text{con-}\mathcal{R}_1]$.

Proof. Take $\Sigma = \{a\}$, $\Gamma = \{a, b, A, B, C, D\}$, and $M = (\Sigma, \Gamma, I)$, where $I$:

<table>
<thead>
<tr>
<th>I</th>
<th>R(M)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1) $(\phi a^2 \mid a^2 \rightarrow B \mid a)$</td>
<td>$\phi a^5 \rightarrow \phi a^2 Ba$</td>
</tr>
<tr>
<td>(2) $(\phi \mid a^2 \rightarrow b \mid Ba)$</td>
<td>$\phi a^2 Ba \rightarrow \phi bBa$</td>
</tr>
<tr>
<td>(3) $(Aa^2 \mid a^2 \rightarrow B \mid a)$</td>
<td>$Aa^5 \rightarrow Aa^2 Ba$</td>
</tr>
<tr>
<td>(4) $(\lambda \mid Aa^2 \rightarrow b \mid Ba)$</td>
<td>$Aa^2 Ba \rightarrow bBa$</td>
</tr>
<tr>
<td>(5) $(b \mid Ba \rightarrow A \mid \lambda)$</td>
<td>$bBa \rightarrow bA$</td>
</tr>
<tr>
<td>(6) $(\lambda \mid A \rightarrow \lambda \mid $)</td>
<td>$A$ \rightarrow $</td>
</tr>
<tr>
<td>(7) $(\lambda \mid b^2 \rightarrow C \mid b^3$)</td>
<td>$b^5$ \rightarrow $Cb^3$</td>
</tr>
<tr>
<td>(8) $(Cb \mid b^2 \rightarrow a \mid $)</td>
<td>$Cb^3$ \rightarrow $Cb a$</td>
</tr>
<tr>
<td>(9) $(\lambda \mid Cb \rightarrow D \mid a)$</td>
<td>$Cba \rightarrow Da$</td>
</tr>
<tr>
<td>(10) $(\lambda \mid b^2 \rightarrow C \mid b^3D)$</td>
<td>$b^5 D \rightarrow Cb^3 D$</td>
</tr>
<tr>
<td>(11) $(Cb \mid b^2 D \rightarrow a \mid \lambda)$</td>
<td>$Cb^3 D \rightarrow Cb a$</td>
</tr>
<tr>
<td>(12) $(\phi \mid D \rightarrow \lambda \mid \lambda)$</td>
<td>$\phi D \rightarrow \phi$</td>
</tr>
<tr>
<td>(13) $(\phi \mid a \rightarrow \lambda \mid $)</td>
<td>$\phi a$ \rightarrow $\phi$</td>
</tr>
<tr>
<td>(14) $(\phi \mid b \rightarrow \lambda \mid $)</td>
<td>$\phi b$ \rightarrow $\phi$</td>
</tr>
</tbody>
</table>
$lc$-$RA[\text{con-}\mathcal{R}_1']$ and $lc$-$RA[\text{con-}\mathcal{R}_1]$

- As the language $L_{\text{expo5}}$ is **not context-free**, we obtain:
- **Corollary**: The class of languages accepted by **confluent $lc$-$RA$ of type $\mathcal{R}_1$** is **incomparable** to $CFL$.
- **In particular**, $lc$-$RA[\text{con-}\mathcal{R}_1] \supset lc$-$RA[\text{con-}\mathcal{R}_2]$.
- These results also hold for the class of languages that are accepted by $lc$-$RA[\text{con-}\mathcal{R}_1']$. 
Confluent lc-RA

- Hierarchy of Language Classes:
Part V: Concluding Remarks

- The class $GCSL$ forms an upper bound for all types of limited context restarting automata considered.
- Under the additional requirement of confluence, the Church-Rosser languages form an upper bound.
- For the most restricted types of $lc$-$RA$ we obtain regular languages, both in confluent and non-confluent case.
- For the intermediate systems, the question for an exact characterization of the corresponding classes of languages remains open.
- For the intermediate systems it even remains open whether the weight-reducing $lc$-$RA$ are more expressive than the corresponding length reducing $lc$-$RA$. 
References

• BASOVNÍK, Learning restricted restarting automata using genetic algorithm.

• BASOVNÍK, MRÁZ, Learning limited context restarting automata by genetic algorithms.

• BEAUDRY, HOLZER, NIEMANN, OTTO, McNaughton families of languages.


• BÜCHI, Regular canonical systems.
  - Arch. f. Math. Logik Grundlagenf. 6 (1964), 91-111.

G. BUNTROCK, Wachsende kontext-sensitive Sprachen.
  - Habilitationsschrift, Fakultät für Mathematik und Informatik, Universität at Würzburg, 1996.

• BUNTROCK, OTTO, Growing context-sensitive languages and Church-Rosser languages.


• ČERNO, MRÁZ, Δ-clearing restarting automata and CFL.

• DAHLHAUS, WARMUTH, Membership for growing context-sensitive grammars is polynomial.

• HOBFBAUER, WALDMANN, Deleting string rewriting systems preserve regularity.

• JANČAR, MRÁZ, PLÁTEK, VÖGEL, Restarting automata.

• LEUPOLD, OTTO, On McNaughton families of languages that are specified by some variants of monadic string-rewriting systems.
  - Fund. Inf. 112 (2011), 219-238.

• MCNAUGHTON, NARENDRAN, OTTO, Church-Rosser Thue systems and formal languages.

• NIEMANN, OTTO, The Church-Rosser languages are the deterministic variants of the growing context-sensitive languages.

• NIEMANN, WOINOWSKI, The growing context-sensitive languages are the acyclic context-sensitive languages.

• OTTO, On deciding the congruence of a finite string-rewriting system on a given congruence class.

• OTTO, Restarting automata.
Thank You!

- This presentation is available on the following website: