Diploma Thesis

Clearing Restarting Automata

Peter Černo
RNDr. František Mráz, CSc.

Clearing Restarting Automata

- **Represent** a **new restricted model** of **restarting automata**.
- **Can** be **learned very efficiently** from positive examples and the extended model enables to learn effectively a **large class of languages**.
- **In the thesis** we relate the class of languages recognized by these automata to **Chomsky hierarchy** and study their formal properties.

Diploma Thesis Outline

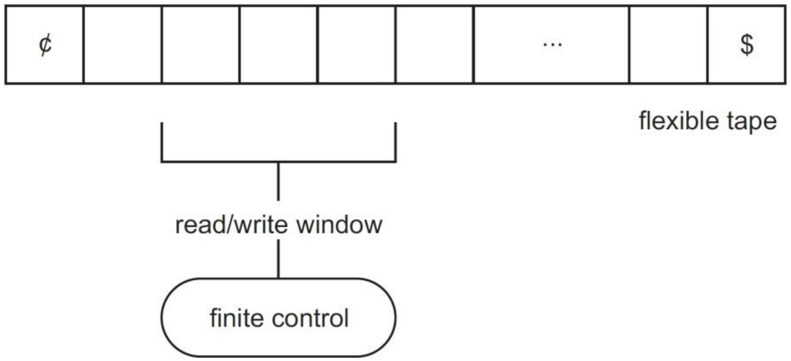
- **Chapter 1** gives a short **introduction** to the theory of automata and formal languages.
- **Chapter 2** gives an overview of several **selected models** related to our model.
- **Chapter 3** introduces our model of **clearing restarting automata**.
- **Chapter 4** describes two **extended models** of clearing restarting automata.
- **Conclusion** gives some **open problems**.

Selected Models

- **Contextual Grammars** by Solomon Marcus:
  - Are based on adjoining (inserting) pairs of strings/contexts into a word according to a selection procedure.
- **Pure grammars** by Mauer et al.:
  - Are similar to Chomsky grammars, but they do not use auxiliary symbols – nonterminals.
- **Church-Rosser string rewriting systems**:
  - Recognize words which can be reduced to an auxiliary symbol Y. Each maximal sequence of reductions ends with the same irreducible string.
- **Associative language descriptions** by Cherubini et al.:
  - Work on so-called stencil trees which are similar to derivation trees but without nonterminals. The inner nodes are marked by an auxiliary symbol Δ.

Selected Models

- **Restarting Automata** by Jančar et al., 1995:
    - Introduced in order to model the so-called analysis by reduction - a technique used in linguistics to analyze sentences of natural languages that have free word order.



Formal Definition

- Let *k* be a *positive integer*.
- ***k*-clearing restarting automaton** (*k-cl-RA*-automaton) is a couple $M = (\Sigma, I)$:
  - Σ is a finite nonempty *alphabet*, $¢, \$ \notin \Sigma$.
  - I is a finite set of *instructions* (x, z, y), $z \in \Sigma^+$,
    - $x \in LC_k = \Sigma^k \cup ¢.\Sigma^{\leq k-1}$ (left context)
    - $y \in RC_k = \Sigma^k \cup \Sigma^{\leq k-1}.\$$ (right context)
  - The special symbols: ¢ and \$ are called *sentinels*.

Formal Definition

- A word $w = uzv$ can be *rewritten* to uv: ($u\underline{z}v \vdash_M uv$) if and only if there exist an instruction $i = (x, z, y) \in I$ such that:
  - x is a *suffix* of $¢.u$
  - y is a *prefix* of $v.\$$
- A word w is *accepted* if and only if $w \vdash^*_M \lambda$ where $\vdash^*_M$ is reflexive and transitive closure of the reduction relation $\vdash_M$.
- The *k-cl-RA*-automaton M *recognizes* the language $L(M) = \{w \in \Sigma^* \mid M$ *accepts* $w\}$.

Formal Definition

- By *cl-RA* we denote the class of all clearing restarting automata.
- $\mathcal{L}(k\text{-}cl\text{-}RA)$ denotes the class of all languages accepted by *k-cl-RA*-automata.
- Similarly $\mathcal{L}(cl\text{-}RA)$ denotes the class of all languages accepted by *cl-RA*-automata.
- $\mathcal{L}(cl\text{-}RA) = \bigcup_{k\geq 1}\mathcal{L}(k\text{-}cl\text{-}RA)$.
- **Note**: For every *cl-RA M*: $\lambda \vdash^*_M \lambda$ hence $\lambda \in L(M)$. If we say that *cl-RA M recognizes a language L*, we always mean that $L(M) = L \cup \{\lambda\}$.

Motivation

- This model was originally inspired by the *Associative Language Descriptions* model:
  - By Alessandra Cherubini, Stefano Crespi-Reghizzi, Matteo Pradella, Pierluigi San Pietro.
- The simplicity of *cl-RA* model implies that the investigation of its properties is not so difficult and also the learning of languages is easy.
- Another important advantage of this model is that the instructions are human readable.

Example

- The language $L = \{a^n b^n \mid n \geq 0\}$ **is recognized** by the *1-cl-RA*-automaton $M = (\{a, b\}, I)$, where the instructions I are:
    - $R1 = (a, \underline{ab}, b)$,
    - $R2 = (¢, \underline{ab}, \$)$.
- For instance:
    - $aaa\underline{ab}bbb \vdash^{R1} aa\underline{ab}bb \vdash^{R1} a\underline{ab}b \vdash^{R1} \underline{ab} \vdash^{R2} \lambda$.
- Now we see that the word *aaaabbbb* is accepted.

Question to the Audience

- **What if** we used only the instruction:
  - $R = (\lambda, \underline{ab}, \lambda)$.

Question to the Audience

- **What if** we used only the instruction:
  - $R = (\lambda, \underline{ab}, \lambda)$.
- **Answer**: we would get a **Dyck language** of correct parentheses generated by the following context-free grammar:
  - $S \rightarrow \lambda \mid SS \mid aSb$.

Set Notation

- **However**, in the definition of *cl-RA*-automata we allowed only contexts with positive length.
- **Therefore** we introduce the following notation:
  - Let $X \subseteq LC_k$, $Y \subseteq RC_k$, $Z \subseteq \Sigma^+$. Then:
    $(X, Z, Y) = \{ (x, z, y) \mid x \in X, z \in Z, y \in Y \}$.
- **Now** we can represent $R = (\lambda, \underline{ab}, \lambda)$ as the set:
  - $(\{¢, a, b\}, \underline{ab}, \{a, b, \$\})$
  - Instead of $\{w\}$ we use only w.

Infinite Hierarchy

- This idea can be easily **generalized**:
  - By increasing the length of contexts we can only increase the power of *cl-RA*-automata.
- **Moreover**:
  - $\mathcal{L}(k\text{-}cl\text{-}RA) \subset \mathcal{L}((k+1)\text{-}cl\text{-}RA)$, for all $k \geq 1$.
  - **Proof**. The following language:

    $\{ (c^k a c^k)^n (c^k b c^k)^n \mid n \geq 0 \}$

    belongs to the $\mathcal{L}((k+1)\text{-}cl\text{-}RA) - \mathcal{L}(k\text{-}cl\text{-}RA)$. ■

Simple Observations

- **Error preserving property**:
  Let $M = (\Sigma, I)$ be a *cl-RA*-automaton and $u \vdash^*_M v$.
  If $u \notin L(M)$ then $v \notin L(M)$.
  - **Proof**. $v \vdash^*_M \lambda \Rightarrow u \vdash^*_M v \vdash^*_M \lambda$. ■
- **Lemma**: For each finite language L there exists a *1-cl-RA*-automaton M such that $L(M) = L \cup \{\lambda\}$.
  - **Proof**. For $L = \{w_1, \ldots, w_n\}$ consider:
    $I = \{(¢, w_1, \$), \ldots, (¢, w_n, \$)\}$. ■

Regular Languages

- **Theorem**:
  All regular languages can be recognized by clearing restarting automata using **only** instructions with left contexts starting with ¢.
- **Theorem**:
  If $M = (\Sigma, I)$ is a *k-cl-RA*-automaton such that for each $(x, z, y) \in I$: ¢ is a prefix of x **or** $\$$ is a suffix of y then $L(M)$ is a regular language.

Context-Free Languages

- **Theorem**:
  Over **one-letter alphabet**, clearing restarting automata recognize **exactly** all context-free languages containing the empty word.
- **Theorem**:
  Over **general alphabet**, the family of languages recognized by *1-cl-RA*-automata is **strictly included** in the family of context-free languages containing the empty word.

Non-Context-Free Languages

- **Theorem**:
  2-cl-RA-automata **can** recognize some **non-**context-free languages.
- **In the following** we give a **technique** which was used to prove that *4-cl-RA*-automaton **can** recognize a **non**-context-free language.
- **How?**
  Let the *cl-RA*-automaton **learn the language**!

Learning Meta-Algorithm

- **Let** $u_i \vdash_M v_i$, $i = 1 \ldots n$ be a list of reductions.
- A **meta-algorithm** for machine learning of unknown clearing restarting automaton:

Step 1: $k := 1$.

Step 2: For each reduction $u_i \vdash_M v_i$ choose (nondeterministically) a **factorization** of u_i, such that $u_i = x_i z_i y_i$ and $v_i = x_i y_i$.

Learning Meta-Algorithm

Step 3: **Construct** a *k-cl-RA* $M = (\Sigma, I)$, where:
$I = \{ (Suff_k(¢.x_i), z_i , Pref_k(y_i.\$)) \mid i = 1 \ldots n \}$.
Step 4: **Test** the automaton M using any available information.

Step 5: If the automaton **passed** all the tests, return M. Otherwise try another factorization of the known reductions and continue by Step 3. If all possible factorizations have been tried, then increase k and continue by Step 2.

Learning Non-CFL

- **Idea**: We try to create a *k-cl-RA*-automaton *M* such that $L(M) \cap \{(ab)^n \mid n>0\} = \{(ab)^{2^m} \mid m \geq 0\}$.
- If *L(M)* is a **CFL** then also the intersection with a regular language is a **CFL**. However, in our case the intersection is **not** a **CFL**.
- Next we give a sample computation showing **how to recognize** words $(ab)^{2^m}$ by means of clearing restarting automata.

Sample Computation

- **Consider**:

$$¢\ ababababababab\underline{a}b\ \$ \vdash_M ¢\ abababababab\underline{a}babb\ \$ \vdash_M$$
$$¢\ ababab\underline{a}babbabb\ \$ \vdash_M ¢\ ab\underline{a}babbabbabb\ \$ \vdash_M$$
$$¢\ abbabbabba\underline{b}b\ \$ \vdash_M ¢\ abbabba\underline{b}bab\ \$ \vdash_M$$
$$¢\ abba\underline{b}babab\ \$ \vdash_M ¢\ a\underline{b}bababab\ \$ \vdash_M$$
$$¢\ ababab\underline{a}b\ \$ \vdash_M ¢\ ab\underline{a}babb\ \$ \vdash_M$$
$$¢\ abba\underline{b}b\ \$ \vdash_M ¢\ a\underline{b}bab\ \$ \vdash_M$$
$$¢\ ab\underline{a}b\ \$ \vdash_M ¢\ a\underline{b}b\ \$ \vdash_M ¢\ \underline{ab}\ \$ \vdash_M ¢\ \lambda\ \$\ accept.$$

- From this sample computation **we can collect** 15 reductions with unambiguous factorizations.

Inferring the Automaton

- The only variable we have to choose is *k* - the length of the context of the instructions.
- **Let us try**:
- **For** $k = 1$ we get the following instructions:

  (b, <u>a</u>, b), (a, <u>b</u>, b), (¢, <u>ab</u>, $).

  But then the automaton would accept the word *ababab* which **does not** belong to *L*:

  abab<u>a</u>b $\vdash_M$ *ab<u>a</u>bb* $\vdash_M$ *a<u>b</u>bb* $\vdash_M$ *a<u>b</u>b* $\vdash_M$ *<u>ab</u>* $\vdash_M$ λ.

Inferring the Automaton

- **For** $k = 2$ we get the following instructions:

  $(ab, \underline{a}, \{b\$, ba\}), (\{\text{¢}a, ba\}, \underline{b}, \{b\$, ba\}), (\text{¢}, \underline{ab}, \$).$

  But then the automaton would accept the word *ababab* which **does not** belong to L:

  $abab\underline{a}b \vdash_M aba\underline{b}b \vdash_M ab\underline{a}b \vdash_M a\underline{b}b \vdash_M \underline{ab} \vdash_M \lambda.$

- **For** $k = 3$ we get the following instructions:

  $(\{\text{¢}ab, bab\}, \underline{a}, \{b\$, bab\}), (\{\text{¢}a, bba\}, \underline{b}, \{b\$, bab\}), (\text{¢}, \underline{ab}, \$).$

  And again we get:

  $ab\underline{a}bab \vdash_M aba\underline{b}b \vdash_M ab\underline{a}b \vdash_M a\underline{b}b \vdash_M \underline{ab} \vdash_M \lambda.$

Inferring the Automaton

- **Finally, for** *$k = 4$* we get the required *4-cl-RA*-automaton *M*.

¢ab abab	<u>a</u>	b\$ babb	¢a abba	<u>b</u>	b\$ bab\$ baba
¢	<u>ab</u>	\$			

- For this *4-cl-RA*-automaton *M* it can be shown, that: $L(M) \cap \{(ab)^n \mid n>0\} = \{(ab)^{2^m} \mid m \geq 0\}$.

Problem with *cl-RA*-automata

- **Theorem**:
  The language $L_1 = \{a^n c b^n \mid n \geq 0\} \cup \{\lambda\}$
  is **not recognized** by any *cl-RA*-automaton.
- **Similarly**:
  Let $L_2 = \{a^n b^n \mid n \geq 0\}$ and $L_3 = \{a^n b^{2n} \mid n \geq 0\}$
  be two sample languages. Both L_2 and L_3
  are recognized by *1-cl-RA*-automata.
- **But** languages $L_2 \cup L_3$ and $L_2 . L_3$
  are **not recognized** by any *cl-RA*-automaton.

(Non-)closure Properties

- **Theorem**:
  The class *$\mathcal{L}$(cl-RA)* is **not closed** under:
  - Union
  - Intersection
  - Intersection with regular language
  - Set difference
  - Concatenation
  - Morphism

Extended Models

- ***Δ*-clearing restarting automata**
  - **Can** leave a mark – a symbol *Δ* – at the place of deleting besides rewriting into the empty word.
  - **Can** recognize Greibach's hardest context-free language.
- ***Δ**-clearing restarting automata**
  - **Can** rewrite a subword w into Δ^k where $k \leq |w|$.
  - **Can** recognize **all** context-free languages.

Example

- The language $L_1 = \{a^n cb^n \mid n \geq 0\} \cup \{\lambda\}$ **is recognized** by the *1-Δcl-RA*-automaton $M = (\{a, b, c\}, I)$, where the instructions I are:
  - $Rc1 = (a, \underline{c} \to \Delta, b)$, $Rc2 = (¢, \underline{c} \to \lambda, \$)$
  - $R\Delta 1 = (a, \underline{a\Delta b} \to \Delta, b)$, $R\Delta 2 = (¢, \underline{a\Delta b} \to \lambda, \$)$
- For instance:
  - $aa\underline{acb}bb \vdash^{Rc1} a\underline{a\Delta b}b \vdash^{R\Delta 1} \underline{a\Delta b} \vdash^{R\Delta 2} \lambda$.
- Now we see that the word *aaacbbb* is accepted.

Greibach's Hardest CFL

- **As we have seen**, not all **CFLs** are recognized by original clearing restarting automata.
- **We can** still characterize **CFL** using Δ-clearing restarting automata, inverse homomorphism and Greibach's hardest context-free language H.
  - **Any** context-free language L can be parsed in whatever time or space it takes to recognize H.
  - **Any** context-free language L can be obtained from H by an inverse homomorphism.

Greibach's Hardest CFL Definition

- Let $\Sigma = \{a_1, a_2, \underline{a}_1, \underline{a}_2, \#, c\}$, $d \notin \Sigma$.
- Let D_2 be *Semi-Dyck language* on $\{a_1, a_2, \underline{a}_1, \underline{a}_2\}$. generated by the context-free grammar:
  $S \rightarrow \lambda \,/\, SS \,/\, a_1 S \underline{a}_1 \,/\, a_2 S \underline{a}_2$.
- Then Greibach's hardest **CFL** $H = \{\lambda\} \cup \{ \prod_{i=1..n} x_i c y_i c z_i d \mid n \geq 1, y_1 y_2 \ldots y_n \in \#D_2, x_i, z_i \in \Sigma^* \}$,
  - $y_1 \in \# . \{a_1, a_2, \underline{a}_1, \underline{a}_2\}^*$,
  - $y_i \in \{a_1, a_2, \underline{a}_1, \underline{a}_2\}^*$ for all $i > 1$.

Greibach's Hardest CFL and *Δcl-RA*

- **Theorem**:
  Greibach's Hardest **CFL** *H*
  is **not recognized** by any *cl-RA*-automaton.
  is **recognized** by a *1-Δcl-RA*-automaton.
  - **Idea**. Suppose that we have $w \in H$:
    $w = ¢\, x_1 c y_1 c z_1 d\ x_2 c y_2 c z_2 d \ldots x_n c y_n c z_n d\ \$$
  - In the ***first phase*** we start with deleting letters (from $\Sigma = \{a_1, a_2, \underline{a}_1, \underline{a}_2, \#, c\}$) from the right side of ¢ and from the left and right sides of the letters *d*.

Greibach's Hardest CFL and *Δcl-RA*

- As soon as we think that we have the word:

  ¢ cy_1cd cy_2cd... cy_ncd $

  we **introduce the *Δ* symbols**:

  ¢ $\Delta y_1 \Delta y_2 \Delta \dots \Delta y_n \Delta$ $

- In the ***second phase*** we check if $y_1y_2 \dots y_n \in \#D_2$.

• **However**, there is no such thing as a *first phase* or a *second phase*.

• **We have only instructions**.

Greibach's Hardest CFL and *Δcl-RA*

- **Nevertheless**, the following holds: Suppose
  $\Sigma = \{a_1, a_2, \underline{a}_1, \underline{a}_2, \#, c\}$, $d \notin \Sigma$, $\Gamma = \Sigma \cup \{d, \Delta\}$.

First phase instructions:	Second phase instructions:
(1) $(¢, \Sigma \to \lambda, \Sigma)$	(7) $(\Gamma, a_1\underline{a}_1 \to \lambda, \Gamma - \{\#\})$
(2) $(\Sigma, \Sigma \to \lambda, d)$	(8) $(\Gamma, a_2\underline{a}_2 \to \lambda, \Gamma - \{\#\})$
(3) $(d, \Sigma \to \lambda, \Sigma)$	(9) $(\Gamma, a_1\Delta\underline{a}_1 \to \Delta, \Gamma - \{\#\})$
(4) $(¢, c \to \Delta, \Sigma \cup \{\Delta\})$	(10) $(\Gamma, a_2\Delta\underline{a}_2 \to \Delta, \Gamma - \{\#\})$
(5) $(\Sigma \cup \{\Delta\}, cdc \to \Delta, \Sigma \cup \{\Delta\})$	(11) $(\Sigma - \{c\}, \Delta \to \lambda, \Delta)$
(6) $(\Sigma \cup \{\Delta\}, cd \to \Delta, \$)$	(12) $(¢, \Delta\#\Delta \to \lambda, \$)$

- **Theorem**:
  $L(M) = H$.

CFL and *Δ*cl-RA*-automata

- ***Δ*cl-RA*-automata** differ from *Δcl-RA*-automata in the ability to leave **more than one** symbol *Δ*.
- The **only constraint** is that they can replace a subword *z* by at most */z/* symbols *Δ*.
- **Theorem**:

  For **each** context-free language *L* there exists a *1-Δ*cl-RA*-automaton *M* recognizing $L \cup \{\lambda\}$.
  - **Idea**. We **code nonterminals** by sequences of symbols *Δ*.

Open Problems

- What is the difference between *ℒ(Δcl-RA)* and *ℒ(Δ*cl-RA)* ?
- Can *Δcl-RA*-automata recognize all context-free languages ?
- What is the relation between *ℒ(Δcl-RA)* and:
  - One counter languages,
  - Simple context-sensitive languages,
  - Growing context-sensitive languages,
  - etc.

Conclusion

- The **main goal** of the thesis was **successfully achieved**.
- The **results** of the thesis were **presented in**:
  - **ABCD workshop**, Prague, March 2009
  - **NCMA workshop**, Wroclaw, August 2009
  - An extended version of the paper from the NCMA workshop was accepted for publication in **Fundamenta Informaticae**.

Thank You

http://www.petercerno.wz.cz/ra.html