Languages Defined by Recurrent Circuits

Christian Reitwießner

Department of Computer Science
University of Würzburg

Walkshop, 1. Oktober 2007
Outline

1 Motivation
   - Regular Expressions as Non-Recurrent Circuits
   - Definition of Recurrent Circuits

2 Investigation of Recurrent Circuit Classes
   - Equivalences to Known Classes
   - The Class $RC(\cdot)$
Examples of Regular Expression

\[ L_2 \overset{\text{df}}{=} (aa \cup b)^* \]
(words containing \(a\)-blocks of even length)

\[ L_3 \overset{\text{df}}{=} (aaa \cup b)^* \]
(\ldots of length multiple of three)

\[ L_6 \overset{\text{df}}{=} L_2 \cap L_3 = (aa \cup b)^* \cap (aaa \cup b)^* \]
(\ldots of length multiple of six)
Examples of Regular Expression

- \( L_2 \overset{\text{df}}{=} (aa \cup b)^* \)
  (words containing \( a \)-blocks of even length)

- \( L_3 \overset{\text{df}}{=} (aaa \cup b)^* \)
  (\ldots of length multiple of three)

- \( L_6 \overset{\text{df}}{=} L_2 \cap L_3 = (aa \cup b)^* \cap (aaa \cup b)^* \)
  (\ldots of length multiple of six)
Examples of Regular Expression

- $L_2 \overset{df}{=} (aa \cup b)^*$
  (words containing $a$-blocks of even length)
- $L_3 \overset{df}{=} (aaa \cup b)^*$
  (… of length multiple of three)
- $L_6 \overset{df}{=} L_2 \cap L_3 = (aa \cup b)^* \cap (aaa \cup b)^*$
  (… of length multiple of six)
Parse Tree of a Regular Expression

\[(aa \cup b)^* = L_2 \]

\[L_6 = (aa \cup b)^* \cap (aaa \cup b)^*\]
Parse Tree of a Regular Expression

\[ (aa \cup b)^* = L_2 \]

\[ * \]

\[ \cap \]

\[ L_6 = (aa \cup b)^* \cap (aaa \cup b)^* \]

\[ L_3 = (aaa \cup b)^* \]
Motivation
Investigation of Recurrent Circuit Classes
Summary and Open Problems

Regular Expressions as Non-Recurrent Circuits
Definition of Recurrent Circuits

Parse Tree of a Regular Expression

\[(aa \cup b)^* = L_2\]
\[L_2 = (aa \cup b)^* \cap (aaa \cup b)^*\]

\[L_3 = (aaa \cup b)^*\]
Regular expressions can be seen as combinatoric circuits with
- letters as inputs and
- language-operations in the computing gates.

What class of languages do we get if we allow non-combinatoric circuits?
Regular expressions can be seen as combinatoric circuits with
- letters as inputs and
- language-operations in the computing gates.

What class of languages do we get if we allow non-combinatoric circuits?
For \( \mathcal{O} \subseteq \{\cup, \cap, \cdot\} \) a recurrent \( \mathcal{O} \)-circuit \( C \) over the alphabet \( \Sigma \) is:

- a directed graph \((V, E)\) with ordered edges
- where to each gate in \( V \) is assigned:
  - one operation from \( \mathcal{O} \) and
  - an initial set that can be
- output gates \( V' \subseteq V \)
For \( \mathcal{O} \subseteq \{ \cup, \cap, \cdot \} \) a recurrent \( \mathcal{O} \)-circuit \( C \) over the alphabet \( \Sigma \) is

- a directed graph \((V, E)\) with ordered edges
- where to each gate in \( V \) is assigned:
  - one operation from \( \mathcal{O} \) and
  - an initial set that can be \( \emptyset \) for some \( w \in \Sigma^* \)
- output gates \( V' \subseteq V \)
For $O \subseteq \{\cup, \cap, \cdot\}$ a recurrent $O$-circuit $C$ over the alphabet $\Sigma$ is

- a directed graph $(V, E)$ with ordered edges
- where to each gate in $V$ is assigned:
  - one operation from $O$ and
  - an initial set that can be $\{w\}$ for some $w \in \Sigma^*$
- output gates $V' \subseteq V$
For $\mathcal{O} \subseteq \{\cup, \cap, \cdot\}$ a recurrent $\mathcal{O}$-circuit $C$ over the alphabet $\Sigma$ is

- a directed graph $(V, E)$ with ordered edges
- where to each gate in $V$ is assigned:
  - one operation from $\mathcal{O}$ and
  - an initial set that can be
    - $\{w\}$ for some $w \in \Sigma^*$,
    - $\emptyset$
- output gates $V' \subseteq V$
Definition of the Syntax

For $\mathcal{O} \subseteq \{\cup, \cap, \cdot\}$ a recurrent $\mathcal{O}$-circuit $C$ over the alphabet $\Sigma$ is

- a directed graph $(V, E)$ with ordered edges
- where to each gate in $V$ is assigned:
  - one operation from $\mathcal{O}$ and
  - an initial set that can be
    - $\{w\}$ for some $w \in \Sigma^*$,
    - $\emptyset$

- output gates $V' \subseteq V$
For $\mathcal{O} \subseteq \{\cup, \cap, \cdot\}$ a recurrent $\mathcal{O}$-circuit $C$ over the alphabet $\Sigma$ is

- a directed graph $(V, E)$ with ordered edges
- where to each gate in $V$ is assigned:
  - one operation from $\mathcal{O}$ and
  - an initial set that can be
    - $\{w\}$ for some $w \in \Sigma^*$,
    - $\emptyset$
- output gates $V' \subseteq V$
Definition of the Semantics

Let $C$ be a recurrent circuit. For each node $v$ we define:

- $C(v, 0) \overset{df}{=} \text{initial set of } v$
- If $u_1, u_2, \ldots, u_n$ are the ordered predecessors of $v$ and $o$ is the operation of $v$ then
  \[ C(v, t + 1) \overset{df}{=} C(v, t) \cup \begin{cases} 
    \bigcup_{i=1}^{n} C(u_i, t) & \text{if } o = \cup \\
    \bigcap_{i=1}^{n} C(u_i, t) & \text{if } o = \cap \\
    C(u_1, t) \cdot \ldots \cdot C(u_n, t) & \text{if } o = \cdot 
  \end{cases} \]
- $C(v) \overset{df}{=} \bigcup_{t=0}^{\infty} C(v, t)$
Let $C$ be a recurrent circuit. For each node $v$ we define:

- $C(v, 0) \overset{\text{df}}{=} \text{initial set of } v$
- If $u_1, u_2, \ldots, u_n$ are the ordered predecessors of $v$ and $o$ is the operation of $v$ then
  \[
  C(v, t + 1) \overset{\text{df}}{=} C(v, t) \cup \begin{cases} 
  \bigcup_{i=1}^{n} C(u_i, t) & \text{if } o = \cup \\
  \bigcap_{i=1}^{n} C(u_i, t) & \text{if } o = \cap \\
  C(u_1, t) \cdot \ldots \cdot C(u_n, t) & \text{if } o = \cdot 
  \end{cases}
  \]
- $C(v) \overset{\text{df}}{=} \bigcup_{t=0}^{\infty} C(v, t)$
Let $C$ be a recurrent circuit. For each node $v$ we define:

- $C(v, 0) \overset{df}{=} \text{initial set of } v$
- If $u_1, u_2, \ldots, u_n$ are the ordered predecessors of $v$ and $o$ is the operation of $v$ then
  $$C(v, t + 1) \overset{df}{=} C(v, t) \cup \begin{cases} \bigcup_{i=1}^{n} C(u_i, t) & \text{if } o = \cup \\ \bigcap_{i=1}^{n} C(u_i, t) & \text{if } o = \cap \\ C(u_1, t) \cdot \ldots \cdot C(u_n, t) & \text{if } o = \cdot \end{cases}$$
- $C(v) \overset{df}{=} \bigcup_{t=0}^{\infty} C(v, t)$
Definition of Recurrent Circuit Classes

Definition

- For a recurrent circuit $C$ with output gates $V'$ we define $L(C) \overset{df}{=} \bigcup_{v \in V'} C(v)$.
- For any $\mathcal{O} \subseteq \{\cup, \cap, \cdot\}$ we define $\text{RC}(\mathcal{O}) \overset{df}{=} \{L \mid L = L(C) \text{ for some recurrent } \mathcal{O}\text{-circuit } C\}$. 
Overview of the Classes

\[
\begin{align*}
\text{CSL} & \quad \text{RC}(\cup, \cap, \cdot) = \text{CCFL} \\
\text{RC}(\cap, \cdot) & \quad \text{RC}(\cup, \cdot) = \text{CFL} \\
\text{RC}(\cdot) & \quad \text{REG} \\
\text{RC}(\cap) & \quad \text{RC}(\cup) = \text{RC}(\cup, \cap)
\end{align*}
\]
Overview of the Classes

\[
\begin{align*}
\text{CSL} & \quad \text{RC}(\cup, \cap, \cdot) = \text{CCFL} \\
\quad & \quad \text{RC}(\cap, \cdot) \quad \text{RC}(\cup, \cdot) = \text{CFL} \\
\quad & \quad \text{RC}(\cdot) \\
\quad & \quad \text{REG} \\
\quad & \quad \text{RC}(\cap) = \text{RC}(\cup) = \text{RC}(\cup, \cap)
\end{align*}
\]
First Observations

Theorem

For every gate $v$ of the circuit $C$ with operation $o$ and
predecessors $u_1, u_2, \ldots, u_n$:

$$C(v) = C(v, 0) \cup \begin{cases} \bigcup_{i=1}^{n} C(u_i) & \text{if } o = \cup \\ \bigcap_{i=1}^{n} C(u_i) & \text{if } o = \cap \\ C(u_1) \cdot \ldots \cdot C(u_n) & \text{if } o = \cdot \end{cases}$$

Theorem

$RC(\mathcal{O})$ is closed under any operation in $\mathcal{O}$ for $\mathcal{O} \subseteq \{\cup, \cap, \cdot\}$. 
First Observations

**Theorem**

For every gate $v$ of the circuit $C$ with operation $o$ and predecessors $u_1, u_2, \ldots, u_n$:

$$C(v) = C(v, 0) \bigcup \begin{cases} \bigcup_{i=1}^{n} C(u_i) & \text{if } o = \bigcup \\ \bigcap_{i=1}^{n} C(u_i) & \text{if } o = \bigcap \\ C(u_1) \cdot \ldots \cdot C(u_n) & \text{if } o = \cdot \end{cases}$$

**Theorem**

$RC(\mathcal{O})$ is closed under any operation in $\mathcal{O}$ for $\mathcal{O} \subseteq \{\bigcup, \bigcap, \cdot\}$. 
Equivalence to Context Free Languages

**Theorem**

\[ \text{RC}(\cup, \cdot) = \text{CFL} \]

**Sketch of Construction.**

Gates and nonterminals correspond in a natural way:
- Concatenation gate \( A \) with predecessors \( A_1, A_2, \ldots, A_n \) corresponds to production \( A \rightarrow A_1 A_2 \cdots A_n \).
- Union gate \( B \) with predecessors \( B_1, B_2, \ldots, B_n \) corresponds to production \( B \rightarrow B_1 | B_2 | \cdots | B_n \).
Theorem
\[ \text{RC}(\cup, \cdot) = \text{CFL} \]

Sketch of Construction.
Gates and nonterminals correspond in a natural way:
Concatenation gate \( A \) with predecessors \( A_1, A_2, \ldots, A_n \)
corresponds to production \( A \rightarrow A_1 A_2 \ldots A_n \).
Union gate \( B \) with predecessors \( B_1, B_2, \ldots, B_n \) corresponds
to production \( B \rightarrow B_1 | B_2 | \cdots | B_n \).
Conjunctive Context Free Languages (CCFL)

- Defined by Okhotin in 2001.
- Context free grammars with intersection.
- Contain productions of the form $A \rightarrow w_1 \& w_2 \& \ldots \& w_n$ with $w_i \in (N \cup \Sigma)^*$.

Definition

$x \in (N \cup \Sigma)^*$ can be derived from $w_1 \& w_2 \& \ldots \& w_n$ iff it can be derived from all $w_i$, $i = 1, 2, \ldots, n$. 
**Theorem**

\[ \text{RC}(\cup, \cap, \cdot) = \text{CCFL} \]

**Sketch of Construction.**

Additionally to proof of \( \text{RC}(\cup, \cdot) = \text{CFL} \):

Intersection gate \( C \) with predecessors \( C_1, C_2, \ldots, C_n \) corresponds to production \( C \rightarrow C_1 \& C_2 \& \ldots \& C_n \).
**Motivation**
Investigation of Recurrent Circuit Classes

**Summary and Open Problems**

**Equivalences to Known Classes**

The Class $\text{RC}(\cdot)$

$\text{RC}(\cup, \cap, \cdot) = \text{CCFL}$

**Theorem**

$\text{RC}(\cup, \cap, \cdot) = \text{CCFL}$

**Sketch of Construction.**

Additionally to proof of $\text{RC}(\cup, \cdot) = \text{CFL}$:

Intersection gate $C$ with predecessors $C_1, C_2, \ldots, C_n$ corresponds to production $C \rightarrow C_1 \& C_2 \& \ldots \& C_n$. 

Christian Reitwießner
Languages Defined by Recurrent Circuits
Facts about CCFL by Okhotin

- Efficient parsing algorithm: Adaption of CYK, also $O(n^3)$.
- This implies $\text{CCFL} \subsetneq \text{P}$.
- There is a P-complete language in CCFL.
- $\Gamma_n(\text{CFL}) \subsetneq \text{CCFL} \subseteq \text{CSL}$
- There are non-regular unary CCFL [Jeż, 2007], as opposed to unary CFL.
Facts about CCFL by Okhotin

- Efficient parsing algorithm: Adaption of CYK, also $O(n^3)$.
- This implies $\text{CCFL} \varsubsetneq \text{P}$.
- There is a $\text{P}$-complete language in $\text{CCFL}$.
  - $\Gamma_n(\text{CFL}) \varsubsetneq \text{CCFL} \subseteq \text{CSL}$
- There are non-regular unary $\text{CCFL}$ [Jeż, 2007], as opposed to unary $\text{CFL}$. 
Facts about CCFL by Okhotin

- Efficient parsing algorithm: Adaption of CYK, also $O(n^3)$.
- This implies $\text{CCFL} \nsubseteq \text{P}$.
- There is a P-complete language in CCFL.
- $\Gamma \cap \text{(CFL)} \nsubseteq \text{CCFL} \subseteq \text{CSL}$
- There are non-regular unary CCFL [Jeż, 2007], as opposed to unary CFL.
Facts about CCFL by Okhotin

- Efficient parsing algorithm: Adaption of CYK, also $O(n^3)$.
- This implies $\text{CCFL} \subsetneq \text{P}$.
- There is a P-complete language in CCFL.
- $\Gamma \cap (\text{CFL}) \subsetneq \text{CCFL} \subseteq \text{CSL}$
- There are non-regular unary CCFL [Jeż, 2007], as opposed to unary CFL.
Overview of the Classes

\[
\begin{align*}
\text{CSL} \\
\text{RC}(\cup, \cap, \cdot) &= \text{CCFL} \\
\text{RC}(\cap, \cdot) &\quad \text{RC}(\cup, \cdot) = \text{CFL} \\
\text{RC}(\cdot) &\quad \text{REG} \\
\text{RC}(\cap) &= \text{RC}(\cup) = \text{RC}(\cup, \cap)
\end{align*}
\]
The Class $\text{RC}(\cdot)$

- can be defined by restriction of PDAs.
- is incomparable to deterministic PDAs.
- is neither closed under $\cap$ nor under $\neg$.
- is closed under $\cup, \cdot, \ast$ and homomorphisms.
**RC(·)**

- can be defined by restriction of PDAs.
- is incomparable to deterministic PDAs.
- is neither closed under $\cap$ nor under $\setminus$.
- is closed under $\cup$, $\cdot$, $*$ and homomorphisms.
RC(·)

- can be defined by restriction of PDAs.
- is incomparable to deterministic PDAs.
- is neither closed under $\cap$ nor under $\neg$.
- is closed under $\cup$, $\cdot$, $^*$ and homomorphisms.
RC(·)

- can be defined by restriction of PDAs.
- is incomparable to deterministic PDAs.
- is neither closed under $\cap$ nor under $\neg$.
- is closed under $\cup$, $\cdot$, $\ast$ and homomorphisms.
Theorem

RC(·) is not closed under intersection with regular languages.

Sketch of Proof.

Theorem of Chomsky and Schützenberger:

\[ L \in CFL \iff L = h(D_n^* \cap R), \text{ for some homomorphism } h, \]
\[ n \in \mathbb{N} \text{ and } R \in \text{REG} \]

\( D_n^* \) is the (one-sided) Dyck language with \( n \) bracket-types).

Since \( D_n^* \in \text{RC(·)} \), this would imply \( \text{RC(·)} = \text{CFL} \), but \( \text{RC(·)} \nsubseteq \text{CFL} \).
**Nice (Non-)Closure Proofs for RC(\emptyset)**

**Theorem**

RC(\cdot) is not closed under intersection with regular languages.

**Sketch of Proof.**

Theorem of Chomsky and Schützenberger:

\[ L \in \text{CFL} \iff L = h(D_n^* \cap R), \text{ for some homomorphism } h, \]
\[ n \in \mathbb{N} \text{ and } R \in \text{REG} \]

\(D_n^*\) is the (one-sided) Dyck language with \(n\) bracket-types.

Since \(D_n^* \in \text{RC(\cdot)}\), this would imply RC(\cdot) = CFL, but RC(\cdot) \nsubseteq\ CFL.
Other Nice (Non-)Closure Proofs for $\text{RC}(\cdot)$

**Theorem**

$\text{RC}(\cdot)$ is not closed under inverse homomorphisms.

**Sketch of Proof.**

Theorem from AFL-Theory:
Let $\mathcal{K}$ be an $\varepsilon$-free language class.
$\mathcal{K}$ closed under $\cdot$, $\varepsilon$-free $h$ and $h^{-1} \Rightarrow \mathcal{K}$ closed under $\cap \text{REG}$.
But the conclusion has just been disproved.
Theorem

$\text{RC}(\cdot)$ is not closed under inverse homomorphisms.

Sketch of Proof.

Theorem from AFL-Theory:
Let $\mathcal{K}$ be an $\varepsilon$-free language class.
$\mathcal{K}$ closed under $\cdot$, $\varepsilon$-free $h$ and $h^{-1} \Rightarrow \mathcal{K}$ closed under $\cap \text{REG}$.
But the conclusion has just been disproved.
**RC(\cap, \cdot) is Not Closed Under Homomorphisms**

**Theorem**

RC(\cap, \cdot) is not closed under homomorphisms.

**Sketch of Proof.**

Chomsky-Schützenberger: CFL = h(D^* \land \text{REG})

Thus CFL \subseteq \Gamma_{h,\cap}(RC(\cdot)) \Rightarrow \Gamma_{h,\cap}(CFL) \subseteq \Gamma_{h,\cap}(RC(\cdot))

Theorem of Ginsburg, Greibach, Harrison: RE = \Gamma_{h,\cap}(CFL)

\Rightarrow RE \subseteq RC(\cap, \cdot), contradiction.
The Class \( RC(\cap, \cdot) \) is Not Closed Under Homomorphisms

**Theorem**

\( RC(\cap, \cdot) \) is not closed under homomorphisms.

**Sketch of Proof.**

Chomsky-Schützenberger: \( CFL = h(D^* \land \text{REG}) \)

Thus \( CFL \subseteq \Gamma_{h,\cap}(RC(\cdot)) \Rightarrow \Gamma_{h,\cap}(CFL) \subseteq \Gamma_{h,\cap}(RC(\cdot)) \)

Theorem of Ginsburg, Greibach, Harrison: \( \text{RE} = \Gamma_{h,\cap}(CFL) \)

\( \Rightarrow \text{RE} \subseteq RC(\cap, \cdot) \), contradiction.
Interesting language classes characterized by recurrent circuits:

- $\text{RC}(\cup, \cap, \cdot)$, an extension of context free grammars.
- $\text{RC}(\cup, \cdot)$, context free grammars.
- $\text{RC}(\cap, \cdot)$, incomparable to context free grammars.
- $\text{RC}(\cdot)$, a restriction of context free grammars.

Open problems:

- Which of the inclusions $\Gamma_\cap(\text{RC}(\cdot)) \subseteq \text{RC}(\cap, \cdot) \subseteq \text{RC}(\cup, \cap, \cdot)$ is strict?
- Closure of $\text{RC}(\cap, \cdot)$ and $\text{RC}(\cap, \cup, \cdot)$ under $\varepsilon$-free homomorphisms and complementation.
- Are there other reasonable possibilities for the initial sets (finite sets, regular languages, ...)?
Interesting language classes characterized by recurrent circuits:
- $\text{RC}(\cup, \cap, \cdot)$, an extension of context free grammars.
- $\text{RC}(\cup, \cdot)$, context free grammars.
- $\text{RC}(\cap, \cdot)$, incomparable to context free grammars.
- $\text{RC}(\cdot)$, a restriction of context free grammars.

Open problems:
- Which of the inclusions $\Gamma_\cap(\text{RC}(\cdot)) \subseteq \text{RC}(\cap, \cdot) \subseteq \text{RC}(\cup, \cap, \cdot)$ is strict?
- Closure of $\text{RC}(\cap, \cdot)$ and $\text{RC}(\cap, \cup, \cdot)$ under $\varepsilon$-free homomorphisms and complementation.
- Are there other reasonable possibilities for the initial sets (finite sets, regular languages, ...)?
Interesting language classes characterized by recurrent circuits:

- $\text{RC}(\cup, \cap, \cdot)$, an extension of context free grammars.
- $\text{RC}(\cup, \cdot)$, context free grammars.
- $\text{RC}(\cap, \cdot)$, incomparable to context free grammars.
- $\text{RC}(\cdot)$, a restriction of context free grammars.

Open problems:

- Which of the inclusions $\Gamma_n(\text{RC}(\cdot)) \subseteq \text{RC}(\cap, \cdot) \subseteq \text{RC}(\cup, \cap, \cdot)$ is strict?
- Closure of $\text{RC}(\cap, \cdot)$ and $\text{RC}(\cap, \cup, \cdot)$ under $\varepsilon$-free homomorphisms and complementation.

Are there other reasonable possibilities for the initial sets (finite sets, regular languages, ...)?
Interesting language classes characterized by recurrent circuits:

- $\text{RC}(\cup, \cap, \cdot)$, an extension of context free grammars.
- $\text{RC}(\cup, \cdot)$, context free grammars.
- $\text{RC}(\cap, \cdot)$, incomparable to context free grammars.
- $\text{RC}(\cdot)$, a restriction of context free grammars.

Open problems:

- Which of the inclusions $\Gamma_\cap(\text{RC}(\cdot)) \subseteq \text{RC}(\cap, \cdot) \subseteq \text{RC}(\cup, \cap, \cdot)$ is strict?
- Closure of $\text{RC}(\cap, \cdot)$ and $\text{RC}(\cap, \cup, \cdot)$ under $\varepsilon$-free homomorphisms and complementation.

Are there other reasonable possibilities for the initial sets (finite sets, regular languages, …)?