Jumping Grammars

Zbyněk Křivka
krivka@fit.vutbr.cz
Brno University of Technology
Faculty of Information Technology
Czech Republic

Seminar of FM Research Group at FIT BUT,
March 31, 2015

Outline

Introduction

Motivation

- Typical grammars and automata work strictly continuously

Motivation

- Typical grammars and automata work strictly continuously
- Adaptation of classical models to work on words discontinuously

Motivation

- Typical grammars and automata work strictly continuously
- Adaptation of classical models to work on words discontinuously
- Models structure unchanged; only the computation is adapted

Motivation

- Typical grammars and automata work strictly continuously
- Adaptation of classical models to work on words discontinuously
- Models structure unchanged; only the computation is adapted
- Jumping Finite Automata – ideas applied to Grammars

Motivation

- Typical grammars and automata work strictly continuously
- Adaptation of classical models to work on words discontinuously
- Models structure unchanged; only the computation is adapted
- Jumping Finite Automata – ideas applied to Grammars

Motivation

- Typical grammars and automata work strictly continuously
- Adaptation of classical models to work on words discontinuously
- Models structure unchanged; only the computation is adapted
- Jumping Finite Automata – ideas applied to Grammars

Possible application fields?

Note: Just theoretical study right now!

- applied mathematics
- computational linguistics
- bioinformatics (DNA computing)
- strongly-scattered information processing

Basic Idea of Jumping Grammars

- We take a grammar of some type (Chomsky classification, etc.) with productions of form

$$x \rightarrow y$$

Basic Idea of Jumping Grammars

- We take a grammar of some type (Chomsky classification, etc.) with productions of form

$$x \rightarrow y$$

- Starting from *starting nonterminal*, we repeatedly rewrites strings to get a sentence.

Basic Idea of Jumping Grammars

- We take a grammar of some type (Chomsky classification, etc.) with productions of form

$$x \rightarrow y$$

- Starting from *starting nonterminal*, we repeatedly rewrites strings to get a sentence.
- Classical grammars:

  Let $z = uxv$. By using $x \rightarrow y$, G rewrites uxv to uyv.

Basic Idea of Jumping Grammars

- We take a grammar of some type (Chomsky classification, etc.) with productions of form

$$x \to y$$

- Starting from *starting nonterminal*, we repeatedly rewrites strings to get a sentence.
- Classical grammars:

  Let $z = uxv$. By using $x \to y$, G rewrites uxv to uyv.
- Jumping grammars:

  Let $z = uxv$. By using $x \to y$, G performs:

Basic Idea of Jumping Grammars

- We take a grammar of some type (Chomsky classification, etc.) with productions of form

$$x \to y$$

- Starting from *starting nonterminal*, we repeatedly rewrites strings to get a sentence.
- Classical grammars:

  Let $z = uxv$. By using $x \to y$, G rewrites uxv to uyv.
- Jumping grammars:

  Let $z = uxv$. By using $x \to y$, G performs:

  1. selects an occurrence of x in z;

Basic Idea of Jumping Grammars

- We take a grammar of some type (Chomsky classification, etc.) with productions of form
$$x \to y$$
- Starting from *starting nonterminal*, we repeatedly rewrites strings to get a sentence.
- Classical grammars:
  Let $z = uxv$. By using $x \to y$, G rewrites uxv to uyv.
- Jumping grammars:
  Let $z = uxv$. By using $x \to y$, G performs:
  1. selects an occurrence of x in z;
  2. erase x from z;

Basic Idea of Jumping Grammars

- We take a grammar of some type (Chomsky classification, etc.) with productions of form
$$x \rightarrow y$$
- Starting from *starting nonterminal*, we repeatedly rewrites strings to get a sentence.
- Classical grammars:

  Let $z = uxv$. By using $x \rightarrow y$, G rewrites uxv to uyv.
- Jumping grammars:

  Let $z = uxv$. By using $x \rightarrow y$, G performs:
  1. selects an occurrence of x in z;
  2. erase x from z;
  3. G jumps anywhere in uv and inserts y there.

Trivial Example – DNA Computing

- DNA is a string over $\{G, A, T, C\}$. For instance,

$$GGGGAGTGGGATTGGGAGAGGGGTTTGCCCCGCTCCC$$

Trivial Example – DNA Computing

- DNA is a string over $\{G, A, T, C\}$. For instance,

$$GGGGAGTGGGATTGGGAGAGGGGTTTGCCCCGCTCCC$$

- We want to study all strings with the same number of Cs and Gs and the same number of As and Ts. For instance,

$$\color{green}{CGGCATCCGGTA}\color{black}{\text{, but }}\color{red}{CGCACCGGTA}$$

Trivial Example – DNA Computing

- DNA is a string over $\{G, A, T, C\}$. For instance,

$$GGGGAGTGGGATTGGGAGAGGGGTTTGCCCCGCTCCC$$

- We want to study all strings with the same number of Cs and Gs and the same number of As and Ts. For instance,

$$CGGCATCCGGTA, \text{ but } CGCACCGGTA$$

- Consider the jumping right-linear grammar with productions

$$1 \to C2,\ 2 \to G1,\ 1 \to 3,\ 3 \to A4,\ 4 \to T3,\ 3 \to \varepsilon$$

Similar devices

- Algebraic approach
  - Commutative language closure
  - Formal Macroset Theory - a sentence as a multiset of symbols, order of symbols is totally irrelevant (Kudlek & Martín-Vide & Păun, 2000)
- Accepting devices = Automata
  - Jumping Finite Automata (Meduna & Zemek, 2012)
- Generating devices = Grammars
  - Commutative Grammars (Crespi-Reghizzi & Mandrioli, 1976)
  - Insertion-Deletion Systems (Kari, 1991+, Verlan, 2000+)
  - Petri Nets

Definitions and Examples

Formal Language Theory - Basic Notions

- For an alphabet, V, V^* represents the free monoid generated by V under concatenation.
- Unit of V^* is denoted by ε.
- The *set of all permutations* of w, perm(w), is defined as perm$(w) = \{b_1 b_2 \cdots b_n \mid b_i \in$ alph(w) for all $i = 1, 2, \ldots, n$, and $(b_1, b_2, \ldots, b_n)$ is a permutation of $(a_1, a_2, \ldots, a_n)$ where $w = a_1 a_2 \cdots a_n\}$.

Definition 1 (General Grammars).

A general grammar (GG for short) is a quadruple, $G = (V, T, P, S)$, where

- V is an alphabet,
- $T \subseteq V$ is an alphabet of terminals, $N = V - T$ is an alphabet of nonterminals,
- P is a finite relation from $V^* - T^*$ to V^* (a member is called rule or production), we write $p\colon x \to y$, and
- $S \in V - T$ is the start nonterminal.

Definition 2 (Four modes of *derivation steps*).

Let $u, v \in V^*$. We define the four derivation relations over V^* as follows

(i) $u \ {}_s\!\Rightarrow v$ in G iff there exist $x \rightarrow y \in P$ and $w, z \in V^*$ such that $u = wxz$ and $v = wyz$;

(ii) $u \ {}_{lj}\!\Rightarrow v$ in G iff there exist $x \rightarrow y \in P$ and $w, t, z \in V^*$ such that $u = wtxz$ and $v = wytz$;

(iii) $u \ {}_{rj}\!\Rightarrow v$ in G iff there exist $x \rightarrow y \in P$ and $w, t, z \in V^*$ such that $u = wxtz$ and $v = wtyz$;

(iv) $u \ {}_j\!\Rightarrow v$ in G iff $u \ {}_{lj}\!\Rightarrow v$ or $u \ {}_{rj}\!\Rightarrow v$ in G.

- Transitive-reflexive and transitive closures of ${}_h\!\Rightarrow$ are denoted by ${}_h\!\Rightarrow^*$ and ${}_h\!\Rightarrow^+$, for $h \in \{s, lj, rj, j\}$.
- Let $k \geq 0$ and ${}_h\!\Rightarrow_k = \{(x, y) \mid (x, y) \in {}_h\!\Rightarrow$, occur$(N, x) \leq k$, occur$(N, y) \leq k\}$.

Definition 3 (Generated Language).

Let $G = (V, T, P, S)$ be a GG. Set

$$L(G, {}_h\Rightarrow) = \{x \in T^* \mid S \;{}_h\Rightarrow^* x\}.$$

$L(G, {}_h\Rightarrow)$ is said to be the *language that G generates by using* ${}_h\Rightarrow$.

For any $X \subseteq \Gamma_{GG}$, set

$$\mathscr{L}(X, {}_h\Rightarrow) = \{L(G, {}_h\Rightarrow) \mid G \in X\}.$$

Grammars Subclasses

Let G be a GG.

- G is a monotonous grammar (MONG) if every $x \to y \in P$ satisfies $|x| \leq |y|$.
- G is a context-sensitive grammar (CSG) if every $x \to y \in P$ satisfies $x = \alpha A \beta$ and $y = \alpha\gamma\beta$ such that $A \in N$, $\alpha, \beta \in V^*$, and $\gamma \in V^+$.
- G is a context-free grammar (CFG) if every $x \to y \in P$ satisfies $x \in N$.
- G is an ε-free context-free grammar (CFG$^{-\varepsilon}$) if G is a CFG and every $x \to y \in P$ satisfies $y \neq \varepsilon$.
- G is a linear grammar (LG) if G is a CFG and every $x \to y \in P$ satisfies $y \in T^*NT^* \cup T^*$.
- G is a right-linear grammar (RLG) if G is a CFG and every $x \to y \in P$ satisfies $y \in T^*N \cup T^*$.
- G is a regular grammar (RG) if G is a CFG and every $x \to y \in P$ satisfies $y \in TN \cup T$.

Language Families

Grammar Classes

Let Γ_X denote the set of all X grammars, for all $X \in$ {GG, MONG, CSG, CFG, CFG$^{-\varepsilon}$, LG, RLG, RG}.

Definition 4 (Well-known Language Families).

Set

- $\mathbf{REG} = \mathscr{L}(\Gamma_{RLG}, {}_s\Rightarrow)$,
- $\mathbf{LIN} = \mathscr{L}(\Gamma_{LG}, {}_s\Rightarrow)$,
- $\mathbf{CF} = \mathscr{L}(\Gamma_{CFG}, {}_s\Rightarrow)$,
- $\mathbf{CS} = \mathscr{L}(\Gamma_{MONG}, {}_s\Rightarrow)$, and
- $\mathbf{RE} = \mathscr{L}(\Gamma_{GG}, {}_s\Rightarrow)$.
- Let k be a positive integer. Set $\mathbf{CF}_k = \bigcup_{i \geq 1}^{k} \mathscr{L}(\Gamma_{CFG}, {}_s\Rightarrow_i)$ and $\mathbf{CF}_{fin} = \{L \mid L \in \mathbf{CF}_i$, for some $i \geq 1\}$ (grammars of finite index).

Recall $\mathbf{FIN} \subset \mathbf{REG} \subset \mathbf{LIN} \subset \mathbf{CF}_{fin} \subset \mathbf{CF} \subset \mathbf{CS} \subset \mathbf{RE}$

Jumping Grammars – Examples

Example 5 (Example of Jumping Regular Grammar).

Consider RG

$$G = \left(\{A, B, C, a, b, c\}, \Sigma = \{a, b, c\}, P, A\right)$$

where $P = \{A \to aB,\ B \to bC,\ C \to cA,\ C \to c\}$.

$$L(G, {}_s\!\Rightarrow) = \{abc\}\{abc\}^* \in \mathbf{REG}, \text{ but}$$

$$L(G, {}_j\!\Rightarrow) = \{w \in \Sigma^* \mid \text{occur}(\{a\}, w) = \text{occur}(\{b\}, w) = \text{occur}(\{c\}, w)\} \in \mathbf{CS}.$$

Jumping Grammars – Examples

Example 6 (Example of Jumping Context-Sensitive Grammar).

Consider CSG $G = (\{S, A, B, a, b\}, \{a, b\}, P, S)$ with productions:

$$\begin{aligned} S &\rightarrow aABb \\ S &\rightarrow ab \\ AB &\rightarrow AABB \\ aA &\rightarrow aa \\ Bb &\rightarrow bb \end{aligned}$$

$L(G, {}_s\Rightarrow) = \{a^n b^n \mid n \geq 1\}$.

Using ${}_j\Rightarrow$, we can make the following derivation sequence:

$S \; {}_j\Rightarrow aABb \; {}_j\Rightarrow aAABBb \; {}_j\Rightarrow aAABbb \; {}_j\Rightarrow aaABbb \; {}_j\Rightarrow aBbbaa \; {}_j\Rightarrow abbbaa$

Notice: $L(G, {}_s\Rightarrow) \in \mathbf{CF}$, but we cannot generate it by any jumping CFG, CSG or even MONG.

Results

Jumping grammars are weak with sequences

Lemma 7.

$\{a\}^*\{b\}^* \notin \mathscr{L}(\Gamma_{MONG}, {}_j\Rightarrow)$.

Proof Idea.

- Assume MONG $G = (V, T, P, S)$ such that $L(G, {}_j\Rightarrow) = \{a\}^*\{b\}^*$.

Jumping grammars are weak with sequences

Lemma 7.

$\{a\}^*\{b\}^* \notin \mathscr{L}(\Gamma_{MONG}, {}_j\Rightarrow)$.

Proof Idea.

- Assume MONG $G = (V, T, P, S)$ such that $L(G, {}_j\Rightarrow) = \{a\}^*\{b\}^*$.
- Let $p\colon\ x \to y \in P$ and $S\ {}_j\Rightarrow^*\ uxv\ {}_j\Rightarrow w\ [p]$ where $w \in L(G, {}_j\Rightarrow)$, $u, v \in T^*$ and $y \in \{a\}^+ \cup \{b\}^+ \cup \{a\}^+\{b\}^+$.

Jumping grammars are weak with sequences

Lemma 7.
$\{a\}^*\{b\}^* \notin \mathscr{L}(\Gamma_{MONG}, {}_j\Rightarrow)$.

Proof Idea.

- Assume MONG $G = (V, T, P, S)$ such that $L(G, {}_j\Rightarrow) = \{a\}^*\{b\}^*$.
- Let $p\colon\ x \to y \in P$ and $S\ {}_j\Rightarrow^*\ uxv\ {}_j\Rightarrow\ w\ [p]$ where $w \in L(G, {}_j\Rightarrow)$, $u, v \in T^*$ and $y \in \{a\}^+ \cup \{b\}^+ \cup \{a\}^+\{b\}^+$.
- In addition, assume that the sentential form uxv is longer than x such that $uv \in \{a\}^+\{b\}^+$.

Jumping grammars are weak with sequences

Lemma 7.

$\{a\}^*\{b\}^* \notin \mathscr{L}(\Gamma_{MONG}, {}_j\Rightarrow)$.

Proof Idea.

- Assume MONG $G = (V, T, P, S)$ such that $L(G, {}_j\Rightarrow) = \{a\}^*\{b\}^*$.
- Let $p\colon\ x \to y \in P$ and $S\ {}_j\Rightarrow^*\ uxv\ {}_j\Rightarrow w\ [p]$ where $w \in L(G, {}_j\Rightarrow)$, $u, v \in T^*$ and $y \in \{a\}^+ \cup \{b\}^+ \cup \{a\}^+\{b\}^+$.
- In addition, assume that the sentential form uxv is longer than x such that $uv \in \{a\}^+\{b\}^+$.
  (a) If y contains at least one symbol b, the last jumping derivation step can place y at the beginning of the sentence and create a string from $\{a, b\}^*\{b\}\{a, b\}^*\{a\}\{a, b\}^*$ that does not belong to $\{a\}^*\{b\}^*$.

Jumping grammars are weak with sequences

Lemma 7.

$\{a\}^*\{b\}^* \notin \mathscr{L}(\Gamma_{MONG}, {}_j\Rightarrow)$.

Proof Idea.

- Assume MONG $G = (V, T, P, S)$ such that $L(G, {}_j\Rightarrow) = \{a\}^*\{b\}^*$.
- Let $p\colon\ x \to y \in P$ and $S \ {}_j\Rightarrow^* \ uxv \ {}_j\Rightarrow \ w \ [p]$ where $w \in L(G, {}_j\Rightarrow)$, $u, v \in T^*$ and $y \in \{a\}^+ \cup \{b\}^+ \cup \{a\}^+\{b\}^+$.
- In addition, assume that the sentential form uxv is longer than x such that $uv \in \{a\}^+\{b\}^+$.
  (a) If y contains at least one symbol b, the last jumping derivation step can place y at the beginning of the sentence and create a string from $\{a, b\}^*\{b\}\{a, b\}^*\{a\}\{a, b\}^*$ that does not belong to $\{a\}^*\{b\}^*$.
  (b) By analogy, if y contains at least one symbol a, the last jumping derivation step can place y at the end of the sentence and therefore, place at least one a behind some bs.

Jumping grammars are weak with sequences

Lemma 7.

$\{a\}^*\{b\}^* \notin \mathscr{L}(\Gamma_{MONG}, {}_j\Rightarrow)$.

Proof Idea.

- Assume MONG $G = (V, T, P, S)$ such that $L(G, {}_j\Rightarrow) = \{a\}^*\{b\}^*$.
- Let $p\colon\ x \to y \in P$ and $S\ {}_j\Rightarrow^*\ uxv\ {}_j\Rightarrow\ w\ [p]$ where $w \in L(G, {}_j\Rightarrow)$, $u, v \in T^*$ and $y \in \{a\}^+ \cup \{b\}^+ \cup \{a\}^+\{b\}^+$.
- In addition, assume that the sentential form uxv is longer than x such that $uv \in \{a\}^+\{b\}^+$.
  (a) If y contains at least one symbol b, the last jumping derivation step can place y at the beginning of the sentence and create a string from $\{a, b\}^*\{b\}\{a, b\}^*\{a\}\{a, b\}^*$ that does not belong to $\{a\}^*\{b\}^*$.
  (b) By analogy, if y contains at least one symbol a, the last jumping derivation step can place y at the end of the sentence and therefore, place at least one a behind some bs.
- This is a contradiction.

Incomparability with regular and context-free languages

Corollary 8.

The following pairs of language families are incomparable, but not disjoint:

- $\mathbf{REG}$ *and* $\mathscr{L}(\Gamma_{MONG}, {}_j\Rightarrow)$;
- $\mathbf{CF}$ *and* $\mathscr{L}(\Gamma_{MONG}, {}_j\Rightarrow)$;
- $\mathbf{REG}$ *and* $\mathscr{L}(\Gamma_{RG}, {}_j\Rightarrow)$;
- $\mathbf{CF}$ *and* $\mathscr{L}(\Gamma_{RG}, {}_j\Rightarrow)$.

Proof.

- Since $\mathbf{REG} \subset \mathbf{CF}$, it is sufficient to prove that $\mathbf{REG} - \mathscr{L}(\Gamma_{MONG}, {}_j\Rightarrow)$, $\mathscr{L}(\Gamma_{RG}, {}_j\Rightarrow) - \mathbf{CF}$, and $\mathbf{REG} \cap \mathscr{L}(\Gamma_{RG}, {}_j\Rightarrow)$ are non-empty

Incomparability with regular and context-free languages

Corollary 8.

The following pairs of language families are incomparable, but not disjoint:

- $\mathbf{REG}$ *and* $\mathscr{L}(\Gamma_{MONG}, {}_j\Rightarrow)$;
- $\mathbf{CF}$ *and* $\mathscr{L}(\Gamma_{MONG}, {}_j\Rightarrow)$;
- $\mathbf{REG}$ *and* $\mathscr{L}(\Gamma_{RG}, {}_j\Rightarrow)$;
- $\mathbf{CF}$ *and* $\mathscr{L}(\Gamma_{RG}, {}_j\Rightarrow)$.

Proof.

- Since $\mathbf{REG} \subset \mathbf{CF}$, it is sufficient to prove that $\mathbf{REG} - \mathscr{L}(\Gamma_{MONG}, {}_j\Rightarrow)$, $\mathscr{L}(\Gamma_{RG}, {}_j\Rightarrow) - \mathbf{CF}$, and $\mathbf{REG} \cap \mathscr{L}(\Gamma_{RG}, {}_j\Rightarrow)$ are non-empty
- By previous lemma 7, $\{a\}^*\{b\}^* \in \mathbf{REG} - \mathscr{L}(\Gamma_{MONG}, {}_j\Rightarrow)$.

Incomparability with regular and context-free languages

Corollary 8.

The following pairs of language families are incomparable, but not disjoint:

- $\mathbf{REG}$ *and* $\mathscr{L}(\Gamma_{MONG}, {}_j\Rightarrow)$*;*
- $\mathbf{CF}$ *and* $\mathscr{L}(\Gamma_{MONG}, {}_j\Rightarrow)$*;*
- $\mathbf{REG}$ *and* $\mathscr{L}(\Gamma_{RG}, {}_j\Rightarrow)$*;*
- $\mathbf{CF}$ *and* $\mathscr{L}(\Gamma_{RG}, {}_j\Rightarrow)$*.*

Proof.

- Since $\mathbf{REG} \subset \mathbf{CF}$, it is sufficient to prove that $\mathbf{REG} - \mathscr{L}(\Gamma_{MONG}, {}_j\Rightarrow)$, $\mathscr{L}(\Gamma_{RG}, {}_j\Rightarrow) - \mathbf{CF}$, and $\mathbf{REG} \cap \mathscr{L}(\Gamma_{RG}, {}_j\Rightarrow)$ are non-empty
- By previous lemma 7, $\{a\}^*\{b\}^* \in \mathbf{REG} - \mathscr{L}(\Gamma_{MONG}, {}_j\Rightarrow)$.
- For $\mathscr{L}(\Gamma_{RG}, {}_j\Rightarrow) - \mathbf{CF} \neq \emptyset$, see Example 5.

Incomparability with regular and context-free languages

Corollary 8.

The following pairs of language families are incomparable, but not disjoint:

- $\mathbf{REG}$ *and* $\mathscr{L}(\Gamma_{MONG}, {}_j\Rightarrow)$;
- $\mathbf{CF}$ *and* $\mathscr{L}(\Gamma_{MONG}, {}_j\Rightarrow)$;
- $\mathbf{REG}$ *and* $\mathscr{L}(\Gamma_{RG}, {}_j\Rightarrow)$;
- $\mathbf{CF}$ *and* $\mathscr{L}(\Gamma_{RG}, {}_j\Rightarrow)$.

Proof.

- Since $\mathbf{REG} \subset \mathbf{CF}$, it is sufficient to prove that $\mathbf{REG} - \mathscr{L}(\Gamma_{MONG}, {}_j\Rightarrow)$, $\mathscr{L}(\Gamma_{RG}, {}_j\Rightarrow) - \mathbf{CF}$, and $\mathbf{REG} \cap \mathscr{L}(\Gamma_{RG}, {}_j\Rightarrow)$ are non-empty
- By previous lemma 7, $\{a\}^*\{b\}^* \in \mathbf{REG} - \mathscr{L}(\Gamma_{MONG}, {}_j\Rightarrow)$.
- For $\mathscr{L}(\Gamma_{RG}, {}_j\Rightarrow) - \mathbf{CF} \neq \emptyset$, see Example 5.
- Regular language $\{a\}^* \in \mathscr{L}(\Gamma_{RG}, {}_j\Rightarrow)$, so $\mathbf{REG} \cap \mathscr{L}(\Gamma_{RG}, {}_j\Rightarrow)$ is non-empty.

Open Problems

Since simple regular language such as $\{a\}^+\{b\}^+$ cannot be generated by jumping CSGs or even jumping MONGs, we pinpoint the following open problem:

Problem 9.

- *Is* $\mathscr{L}(\Gamma_{CFG}, {}_j\Rightarrow) \subseteq \mathscr{L}(\Gamma_{CSG}, {}_j\Rightarrow)$ *proper?*

Open Problems

Since simple regular language such as $\{a\}^+\{b\}^+$ cannot be generated by jumping CSGs or even jumping MONGs, we pinpoint the following open problem:

Problem 9.

- *Is* $\mathscr{L}(\Gamma_{CFG}, {}_j\Rightarrow) \subseteq \mathscr{L}(\Gamma_{CSG}, {}_j\Rightarrow)$ *proper?*
- *Is* $\mathscr{L}(\Gamma_{CSG}, {}_j\Rightarrow) \subseteq \mathscr{L}(\Gamma_{MONG}, {}_j\Rightarrow)$ *proper?*

Context-sensitive jumping is weaker than classical one

Theorem 10.
$\mathscr{L}(\Gamma_{MONG}, {}_j\Rightarrow) \subset \mathbf{CS}$.

Proof.

- By demonstrating transformation of any jumping MONG, $G = (V_G, T, P_G, S)$, to an equivalent MONG, $H = (V_H, T, P_H, S)$.

□

Context-sensitive jumping is weaker than classical one

Theorem 10.
$\mathscr{L}(\Gamma_{MONG}, {}_{j}\!\Rightarrow) \subset \mathbf{CS}$.

Proof.

- By demonstrating transformation of any jumping MONG, $G = (V_G, T, P_G, S)$, to an equivalent MONG, $H = (V_H, T, P_H, S)$.
- Set $V_H = N_H \cup T$ and $N_H = N_G \cup \{\bar{X} \mid X \in V_G\}$.

□

Context-sensitive jumping is weaker than classical one

Theorem 10.
$\mathscr{L}(\Gamma_{MONG}, {}_j\Rightarrow) \subset \mathbf{CS}$.

Proof.

- By demonstrating transformation of any jumping MONG, $G = (V_G, T, P_G, S)$, to an equivalent MONG, $H = (V_H, T, P_H, S)$.
- Set $V_H = N_H \cup T$ and $N_H = N_G \cup \{\bar{X} \mid X \in V_G\}$.
- Let π be the homomorphism from V_G^* to V_H^* defined by $\pi(X) = \bar{X}$ for all $X \in V_G$. Set $P_H = P_1 \cup P_2$ where

□

Context-sensitive jumping is weaker than classical one

Theorem 10.
$\mathscr{L}(\Gamma_{MONG}, {}_j\Rightarrow) \subset \mathbf{CS}$.

Proof.

- By demonstrating transformation of any jumping MONG, $G = (V_G, T, P_G, S)$, to an equivalent MONG, $H = (V_H, T, P_H, S)$.
- Set $V_H = N_H \cup T$ and $N_H = N_G \cup \{\bar{X} \mid X \in V_G\}$.
- Let π be the homomorphism from V_G^* to V_H^* defined by $\pi(X) = \bar{X}$ for all $X \in V_G$. Set $P_H = P_1 \cup P_2$ where
  - $P_1 = \bigcup_{\alpha \to \beta \in P_G} \{\alpha \to \pi(\beta), \pi(\beta) \to \beta\}$

Context-sensitive jumping is weaker than classical one

Theorem 10.
$\mathscr{L}(\Gamma_{MONG}, {}_j\Rightarrow) \subset \mathbf{CS}$.

Proof.

- By demonstrating transformation of any jumping MONG, $G = (V_G, T, P_G, S)$, to an equivalent MONG, $H = (V_H, T, P_H, S)$.
- Set $V_H = N_H \cup T$ and $N_H = N_G \cup \{\bar{X} \mid X \in V_G\}$.
- Let π be the homomorphism from V_G^* to V_H^* defined by $\pi(X) = \bar{X}$ for all $X \in V_G$. Set $P_H = P_1 \cup P_2$ where
  - $P_1 = \bigcup_{\alpha \to \beta \in P_G} \{\alpha \to \pi(\beta), \pi(\beta) \to \beta\}$
  - $P_2 = \bigcup_{\alpha \to \beta \in P_G} \{X\pi(\beta) \to \pi(\beta)X, \pi(\beta)X \to X\pi(\beta) \mid X \in V_G\}$

Context-sensitive jumping is weaker than classical one

Theorem 10.
$\mathscr{L}(\Gamma_{MONG}, {}_j\Rightarrow) \subset \mathbf{CS}$.

Proof.

- By demonstrating transformation of any jumping MONG, $G = (V_G, T, P_G, S)$, to an equivalent MONG, $H = (V_H, T, P_H, S)$.
- Set $V_H = N_H \cup T$ and $N_H = N_G \cup \{\bar{X} \mid X \in V_G\}$.
- Let π be the homomorphism from V_G^* to V_H^* defined by $\pi(X) = \bar{X}$ for all $X \in V_G$. Set $P_H = P_1 \cup P_2$ where
  - $P_1 = \bigcup_{\alpha \to \beta \in P_G} \{\alpha \to \pi(\beta), \pi(\beta) \to \beta\}$
  - $P_2 = \bigcup_{\alpha \to \beta \in P_G} \{X\pi(\beta) \to \pi(\beta)X, \pi(\beta)X \to X\pi(\beta) \mid X \in V_G\}$
- Clearly, $\{a\}^*\{b\}^* \in \mathbf{CS}$, so $\mathbf{CS} - \mathscr{L}(\Gamma_{MONG}, {}_j\Rightarrow)$ is non-empty. Hence, this theorem holds.

□

Dyck Language with Finite Index?

Example 11.

Consider Dyck language of all well-written arithmetic expression only with $($, $)$ and $[$, $]$.

By classical CFG G

$$E \to (E)E,\, E \to [E]E,\, E \to \varepsilon$$

But G is not of a finite index!

Dyck Language with Finite Index?

Example 11.

Consider Dyck language of all well-written arithmetic expression only with $(,)$ and $[,]$.

By classical CFG G

$$E \rightarrow (E)E,\, E \rightarrow [E]E,\, E \rightarrow \varepsilon$$

But G is not of a finite index!

By jumping RLG H

$$\begin{array}{lll} E & \rightarrow & ()E \\ E & \rightarrow & []E \\ E & \rightarrow & \varepsilon \end{array}$$

Observe that H is of index 1.

Jumping Finite Automata

Definition 12.

A general jumping finite automaton (GJFA) is a quintuple $M = (Q, \Sigma, R, s, F)$, where

- Q is finite set of *states*
- Σ is the *input alphabet*, $Q \cap \Sigma = \emptyset$,
- $R \subseteq Q \times \Sigma^* \times Q$ is finite, member are called *rules*, instead of $(p, y, q) \in R$, we write $py \rightarrow q \in R$,
- $s \in Q$ is the *start state*, and
- $F \subseteq Q$ is a set of *final states*.

Jumping Finite Automata

Definition 12.
A general jumping finite automaton (GJFA) is a quintuple
$M = (Q, \Sigma, R, s, F)$, where

- Q is finite set of *states*
- Σ is the *input alphabet*, $Q \cap \Sigma = \emptyset$,
- $R \subseteq Q \times \Sigma^* \times Q$ is finite, member are called *rules*, instead of $(p, y, q) \in R$, we write $py \to q \in R$,
- $s \in Q$ is the *start state*, and
- $F \subseteq Q$ is a set of *final states*.

If $py \to q \in R$ implies that $|y| \leq 1$, then M is a jumping finite automaton (JFA).

Jumping Finite Automata – Language

Definition 13.

A *configuration* of M is any string in $\Sigma^* Q \Sigma^*$. The binary *jumping relation*, symbolically denoted by $\curvearrowright$, over $\Sigma^* Q \Sigma^*$:

- Let $x, z, x', z' \in \Sigma^*$ such that $xz = x'z'$ and $py \rightarrow q \in R$; then, M makes a jump from $xpyz$ to $x'qz'$, symbolically written as $xpyz \curvearrowright x'qz'$.
- In the standard manner, we extent $\curvearrowright$ to $\curvearrowright^m$, where $m \geq 0$, $\curvearrowright^+$, and $\curvearrowright^*$.

The *language* accepted by M, denoted by $L(M)$, is defined as
$L(M) = \{uv \mid u, v \in \Sigma^*, usv \curvearrowright^* f, f \in F\}$.
GJFA and **JFA** denote the families of languages accepted by GJFAs and JFAs, respectively.

Recall known[1] results

$\mathbf{JFA} \subset \mathbf{GJFA}$, $\mathbf{FIN} \subset \mathbf{GJFA}$, and $\mathbf{FIN}$ and $\mathbf{JFA}$ are incomparable.

[1] See "A. Meduna and P. Zemek, Jumping Automata. *Int. J. Found. Comput. Sci.* **23**(2012) 1555–1578."

$\mathbf{GJFA} = \mathscr{L}(\Gamma_{\mathbf{RLG}}, {}_j\Rightarrow)$

Lemma 14.

GJFA $\subseteq \mathscr{L}(\Gamma_{RLG}, {}_j\Rightarrow)$.

Proof.

For every GJFA $M = (Q, \Sigma, R, s, F)$, we construct a RLG $G = (Q \cup \Sigma \cup \{S\}, \Sigma, P, S)$, where S is a new nonterminal, $S \notin Q \cup \Sigma$, such that $L(M) = L(G, {}_j\Rightarrow)$.

$$P = \{S \rightarrow f \mid f \in F\} \cup \{q \rightarrow xp \mid px \rightarrow q \in R\} \cup \{q \rightarrow x \mid sx \rightarrow q \in R\}$$

Basic Idea

- Principle: analogous to conversion from classical general (lazy) finite automata to equivalent RLGs

$\mathbf{GJFA} = \mathscr{L}(\Gamma_{\mathbf{RLG}}, {}_j\Rightarrow)$

Lemma 14.

GJFA $\subseteq \mathscr{L}(\Gamma_{RLG}, {}_j\Rightarrow)$.

Proof.

For every GJFA $M = (Q, \Sigma, R, s, F)$, we construct a RLG $G = (Q \cup \Sigma \cup \{S\}, \Sigma, P, S)$, where S is a new nonterminal, $S \notin Q \cup \Sigma$, such that $L(M) = L(G, {}_j\Rightarrow)$.

$$P = \{S \rightarrow f \mid f \in F\} \cup \{q \rightarrow xp \mid px \rightarrow q \in R\} \cup \{q \rightarrow x \mid sx \rightarrow q \in R\}$$

Basic Idea

- Principle: analogous to conversion from classical general (lazy) finite automata to equivalent RLGs
- First, S is nondeterministically rewritten to some f in G. Let $w = uv$.

$$usv \curvearrowright^* ypxy' \curvearrowright zqz'z'' \; [px \rightarrow q] \; \curvearrowright^* f \text{ in } M$$

is simulated in G by

$$S \;{}_j\Rightarrow f \;{}_j\Rightarrow^* zz'qz'' \;{}_j\Rightarrow yxpy' \; [q \rightarrow xp] \;{}_j\Rightarrow^* w, \text{ where } yy' = zz'z''.$$

$\mathbf{GJFA} = \mathscr{L}(\Gamma_{\mathbf{RLG}}, {}_j\Rightarrow)$

Lemma 15.

$\mathscr{L}(\Gamma_{RLG}, {}_j\Rightarrow) \subseteq$ **GJFA**.

Proof.

For every RLG $G = (V, T, P, S)$, we construct a GJFA $M = (N \cup \{\sigma\}, T, R, \sigma, \{S\})$, where σ is a new start state, $\sigma \notin V$ and $N = V - T$, such that $L(G, {}_j\Rightarrow) = L(M)$.

$$R = \{Bx \rightarrow A \mid A \rightarrow xB \in P, A, B \in N, x \in T^*\} \cup \\ \{\sigma x \rightarrow A \mid A \rightarrow x \in P, x \in T^*\}$$

Basic Idea

- The start nonterminal of G corresponds to the only final state of M.

$$S \;{}_j\Rightarrow^* yy'Ay'' \;{}_j\Rightarrow zxBz' \; [A \rightarrow xB] \;{}_j\Rightarrow^* w$$

is simulated by M's jumping moves as

$u\sigma v \curvearrowright^* zBxz' \curvearrowright yAy'y'' \; [Bx \rightarrow A] \; \curvearrowright^* S$, where $yy'y'' = zz'$ and $w = uv$.

Equivalence with Jumping Finite Automata

Theorem 16.
GJFA $= \mathscr{L}(\Gamma_{RLG}, {}_j\Rightarrow)$.

Proof.
This theorem holds by Lemmas 14 and 15. □

Theorem 17.
JFA $= \mathscr{L}(\Gamma_{RG}, {}_j\Rightarrow)$.

Proof.

- Consider jumping finite automata that processes only one input symbol in one move.

□

Equivalence with Jumping Finite Automata

Theorem 16.
GJFA $= \mathscr{L}(\Gamma_{RLG}, {}_j\Rightarrow)$.

Proof.
This theorem holds by Lemmas 14 and 15. □

Theorem 17.
JFA $= \mathscr{L}(\Gamma_{RG}, {}_j\Rightarrow)$.

Proof.

- Consider jumping finite automata that processes only one input symbol in one move.
- Proof is analogical to the proof of Theorem 16 with $x \in T$.

□

Right-Linear, Linear and Finite Index Jumping Grammars

Theorem 18.
$\mathscr{L}(\Gamma_{RLG}, {}_j\Rightarrow) = \mathscr{L}(\Gamma_{LG}, {}_j\Rightarrow) = \bigcup_{k\geq 1} \mathscr{L}(\Gamma_{CFG}, {}_j\Rightarrow_k)$.

Idea.

- Since $\mathscr{L}(\Gamma_{RLG}, {}_j\Rightarrow) \subseteq \mathscr{L}(\Gamma_{LG}, {}_j\Rightarrow) \subseteq \bigcup_{k\geq 1} \mathscr{L}(\Gamma_{CFG}, {}_j\Rightarrow_k)$ follows from the definitions, it suffices to proof that $\bigcup_{k\geq 1} \mathscr{L}(\Gamma_{CFG}, {}_j\Rightarrow_k) \subseteq \mathscr{L}(\Gamma_{RLG}, {}_j\Rightarrow)$ (transform G to H).

□

Right-Linear, Linear and Finite Index Jumping Grammars

Theorem 18.
$\mathscr{L}(\Gamma_{RLG}, {}_j\Rightarrow) = \mathscr{L}(\Gamma_{LG}, {}_j\Rightarrow) = \bigcup_{k\geq 1} \mathscr{L}(\Gamma_{CFG}, {}_j\Rightarrow_k)$.

Idea.

- Since $\mathscr{L}(\Gamma_{RLG}, {}_j\Rightarrow) \subseteq \mathscr{L}(\Gamma_{LG}, {}_j\Rightarrow) \subseteq \bigcup_{k\geq 1} \mathscr{L}(\Gamma_{CFG}, {}_j\Rightarrow_k)$ follows from the definitions, it suffices to proof that $\bigcup_{k\geq 1} \mathscr{L}(\Gamma_{CFG}, {}_j\Rightarrow_k) \subseteq \mathscr{L}(\Gamma_{RLG}, {}_j\Rightarrow)$ (transform G to H).
- $V_H = \{\langle x \rangle \mid x \in \bigcup_{i=1}^{k} (V_G - T)^i\} \cup T$

□

Right-Linear, Linear and Finite Index Jumping Grammars

Theorem 18.

$$\mathscr{L}(\Gamma_{RLG}, {}_j\Rightarrow) = \mathscr{L}(\Gamma_{LG}, {}_j\Rightarrow) = \bigcup_{k \geq 1} \mathscr{L}(\Gamma_{CFG}, {}_j\Rightarrow_k).$$

Idea.

- Since $\mathscr{L}(\Gamma_{RLG}, {}_j\Rightarrow) \subseteq \mathscr{L}(\Gamma_{LG}, {}_j\Rightarrow) \subseteq \bigcup_{k \geq 1} \mathscr{L}(\Gamma_{CFG}, {}_j\Rightarrow_k)$ follows from the definitions, it suffices to proof that $\bigcup_{k \geq 1} \mathscr{L}(\Gamma_{CFG}, {}_j\Rightarrow_k) \subseteq \mathscr{L}(\Gamma_{RLG}, {}_j\Rightarrow)$ (transform G to H).
- $V_H = \{\langle x \rangle \mid x \in \bigcup_{i=1}^{k} (V_G - T)^i\} \cup T$
- $P_H = \{\langle \alpha A \beta \rangle \rightarrow \tau(x)\langle \gamma \rangle \mid A \rightarrow x \in P_G, \alpha, \beta \in N^*, \gamma = \alpha\beta\eta(x), 1 \leq |\gamma| \leq k\} \cup \{\langle A \rangle \rightarrow x \mid A \rightarrow x \in P_G, x \in T^*\}$

□

Right-Linear, Linear and Finite Index Jumping Grammars

Theorem 18.

$\mathscr{L}(\Gamma_{RLG}, {}_j\Rightarrow) = \mathscr{L}(\Gamma_{LG}, {}_j\Rightarrow) = \bigcup_{k\geq 1} \mathscr{L}(\Gamma_{CFG}, {}_j\Rightarrow_k)$.

Idea.

- Since $\mathscr{L}(\Gamma_{RLG}, {}_j\Rightarrow) \subseteq \mathscr{L}(\Gamma_{LG}, {}_j\Rightarrow) \subseteq \bigcup_{k\geq 1} \mathscr{L}(\Gamma_{CFG}, {}_j\Rightarrow_k)$ follows from the definitions, it suffices to proof that $\bigcup_{k\geq 1} \mathscr{L}(\Gamma_{CFG}, {}_j\Rightarrow_k) \subseteq \mathscr{L}(\Gamma_{RLG}, {}_j\Rightarrow)$ (transform G to H).
- $V_H = \{\langle x\rangle \mid x \in \bigcup_{i=1}^{k} (V_G - T)^i\} \cup T$
- $P_H = \{\langle \alpha A\beta\rangle \rightarrow \tau(x)\langle\gamma\rangle \mid A \rightarrow x \in P_G, \alpha, \beta \in N^*, \gamma = \alpha\beta\eta(x),$ $1 \leq |\gamma| \leq k\} \cup \{\langle A\rangle \rightarrow x \mid A \rightarrow x \in P_G, x \in T^*\}$

□

Right-Linear, Linear and Finite Index Jumping Grammars

Theorem 18.
$\mathscr{L}(\Gamma_{RLG}, {}_j\Rightarrow) = \mathscr{L}(\Gamma_{LG}, {}_j\Rightarrow) = \bigcup_{k\geq 1} \mathscr{L}(\Gamma_{CFG}, {}_j\Rightarrow_k)$.

Idea.

- Since $\mathscr{L}(\Gamma_{RLG}, {}_j\Rightarrow) \subseteq \mathscr{L}(\Gamma_{LG}, {}_j\Rightarrow) \subseteq \bigcup_{k\geq 1} \mathscr{L}(\Gamma_{CFG}, {}_j\Rightarrow_k)$ follows from the definitions, it suffices to proof that $\bigcup_{k\geq 1} \mathscr{L}(\Gamma_{CFG}, {}_j\Rightarrow_k) \subseteq \mathscr{L}(\Gamma_{RLG}, {}_j\Rightarrow)$ (transform G to H).
- $V_H = \{\langle x\rangle \mid x \in \bigcup_{i=1}^{k} (V_G - T)^i\} \cup T$
- $P_H = \{\langle \alpha A\beta\rangle \to \tau(x)\langle\gamma\rangle \mid A \to x \in P_G, \alpha,\beta \in N^*, \gamma = \alpha\beta\eta(x),$ $1 \leq |\gamma| \leq k\} \cup \{\langle A\rangle \to x \mid A \to x \in P_G, x \in T^*\}$

□

Problem 19.
Is $\bigcup_{k\geq 1} \mathscr{L}(\Gamma_{CFG}, {}_j\Rightarrow_k) \subseteq \mathscr{L}(\Gamma_{CFG}, {}_j\Rightarrow)$ *proper?*

General Jumping Grammars are Turing Complete

Lemma 20.

$\mathbf{RE} \subseteq \mathscr{L}(\Gamma_{GG}, {}_j\Rightarrow)$.

Construction.

- For every GG $G = (V_G, T, P_G, S_G)$, we construct another GG $H = (V_H = V_G \cup \{S_H, \$, \#, \lfloor, \rfloor\}, T, P_H, S_H)$ such that $L(G, {}_s\Rightarrow) = L(H, {}_j\Rightarrow)$.

□

General Jumping Grammars are Turing Complete

Lemma 20.

$\mathbf{RE} \subseteq \mathscr{L}(\Gamma_{GG}, {}_j\Rightarrow)$.

Construction.

- For every GG $G = (V_G, T, P_G, S_G)$, we construct another GG $H = (V_H = V_G \cup \{S_H, \$, \#, \lfloor, \rfloor\}, T, P_H, S_H)$ such that $L(G, {}_s\Rightarrow) = L(H, {}_j\Rightarrow)$.
- $S_H, \$, \#, \lfloor$, and $\rfloor$ are new nonterminal symbols in H.

$$P_H = \{S_H \to \#S_G, \# \to \lfloor\$, \lfloor\,\rfloor \to \#, \# \to \varepsilon\} \cup \\ \{\$\alpha \to \rfloor\beta \mid \alpha \to \beta \in P_G\}.$$

□

General Jumping Grammars are Turing Complete

Lemma 20.

$\mathbf{RE} \subseteq \mathscr{L}(\Gamma_{GG}, {}_j\Rightarrow)$.

Construction.

- For every GG $G = (V_G, T, P_G, S_G)$, we construct another GG $H = (V_H = V_G \cup \{S_H, \$, \#, \lfloor, \rfloor\}, T, P_H, S_H)$ such that $L(G, {}_s\Rightarrow) = L(H, {}_j\Rightarrow)$.
- $S_H, \$, \#, \lfloor$, and $\rfloor$ are new nonterminal symbols in H.

$$P_H = \{S_H \to \#S_G, \# \to \lfloor\$, \lfloor\,\rfloor \to \#, \# \to \varepsilon\} \cup \{\$\alpha \to \rfloor\beta \mid \alpha \to \beta \in P_G\}.$$

- Idea: Every application of $\alpha \to \beta$ in G is simulated in H:

$$\ldots\#\ldots\alpha\ldots\ {}_j\Rightarrow\ \ldots\lfloor\$\alpha\ldots\ {}_j\Rightarrow\ldots\lfloor\rfloor\beta\ldots\ {}_j\Rightarrow\ \ldots\#\ldots\beta\ldots$$

□

General Jumping Grammars are Turing Complete

Lemma 20.

$\mathbf{RE} \subseteq \mathscr{L}(\Gamma_{GG}, {}_j\Rightarrow)$.

Construction.

- For every GG $G = (V_G, T, P_G, S_G)$, we construct another GG $H = (V_H = V_G \cup \{S_H, \$, \#, \lfloor, \rfloor\}, T, P_H, S_H)$ such that $L(G, {}_s\Rightarrow) = L(H, {}_j\Rightarrow)$.
- $S_H, \$, \#, \lfloor$, and $\rfloor$ are new nonterminal symbols in H.

$$P_H = \{S_H \rightarrow \#S_G, \# \rightarrow \lfloor\$, \lfloor\rfloor \rightarrow \#, \# \rightarrow \varepsilon\} \cup \{\$\alpha \rightarrow \rfloor\beta \mid \alpha \rightarrow \beta \in P_G\}.$$

- Idea: Every application of $\alpha \rightarrow \beta$ in G is simulated in H:

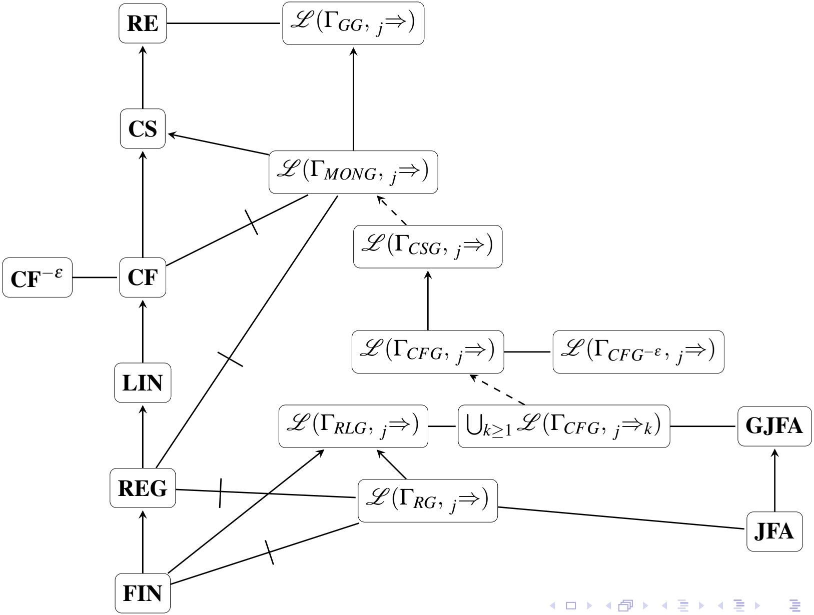
$$\ldots\#\ldots\alpha\ldots \;{}_j\Rightarrow\; \ldots\lfloor\$\alpha\ldots \;{}_j\Rightarrow\; \ldots\lfloor\rfloor\beta\ldots \;{}_j\Rightarrow\; \ldots\#\ldots\beta\ldots$$

□

Theorem 21.

$\mathscr{L}(\Gamma_{GG}, {}_j\Rightarrow) = \mathbf{RE}$.

Language Families Hierarchy - Results Summary



Semilinearity

Definition 22.

- Let $w \in V^*$ with $V = \{a_1, \ldots, a_n\}$.
- We define Parikh vector of w by $\psi_V(w) = (\text{occur}(a_1, w), \text{occur}(a_2, w), \ldots, \text{occur}(a_n, w))$.
- A set of vectors is called *semilinear* if it can be represented as a union of a finite number of sets of the form $\{v_0 + \sum_{i=1}^{m} \alpha_i v_i \mid \alpha_i \in \mathbb{N}, 1 \leq i \leq m\}$ where v_i for $0 \leq i \leq m$ is an n-dimensional vector.
- A language $L \subseteq V^*$ is called *semilinear* if the set $\psi_V(L) = \{\psi_V(w) \mid w \in L\}$ is a semilinear set.
- A language family is semilinear if all its languages are semilinear.

Semilineary of Context-Free Jumping Language

Lemma 23.
For $X \in \{$RG, RLG, LG, CFG$\}$, $\mathscr{L}(\Gamma_X, {}_j\Rightarrow)$ is semilinear.

Proof.

- By Parikh's Theorem, for each context-free language $L \subseteq V^*$, $\psi_V(L)$ is semilinear.

□

Semilineary of Context-Free Jumping Language

Lemma 23.

For $X \in \{$*RG, RLG, LG, CFG*$\}$, $\mathscr{L}(\Gamma_X, {}_j\Rightarrow)$ *is semilinear.*

Proof.

- By Parikh's Theorem, for each context-free language $L \subseteq V^*$, $\psi_V(L)$ is semilinear.
- Let G be a CFG such that $L(G, {}_s\Rightarrow) = L$.

□

Semilineary of Context-Free Jumping Language

Lemma 23.
For $X \in \{RG, RLG, LG, CFG\}$, $\mathscr{L}(\Gamma_X, {}_j\Rightarrow)$ *is semilinear.*

Proof.

- By Parikh's Theorem, for each context-free language $L \subseteq V^*$, $\psi_V(L)$ is semilinear.
- Let G be a CFG such that $L(G, {}_s\Rightarrow) = L$.
- From the definition of ${}_j\Rightarrow$ and CFG it follows that $\psi(L(G, {}_s\Rightarrow)) = \psi(L(G, {}_j\Rightarrow))$ therefore $\psi(L(G, {}_j\Rightarrow))$ is semilinear as well.

□

Multiset Grammar and Language

Definition 24.

Let $G = (V, T, P, S) \in \Gamma_{GG}$ be a grammar and $u, v \in V^*$; then, $u\ {}_m\!\Rightarrow v\ [x \rightarrow y]$ in G iff there exist $x \rightarrow y \in P$ and $t, t', z, z' \in V^*$ such that $txt' \in \text{perm}(u)$ and $zyz' \in \text{perm}(v)$. Then, $L(G,\ {}_m\!\Rightarrow)$ is called multiset language.

Lemma 25.

Let $G \in \Gamma_{GG}$; then, $w \in L(G,\ {}_m\!\Rightarrow)$ implies that perm$(w) \subseteq L(G,\ {}_m\!\Rightarrow)$.

Proof.

Consider Definition 24 with v representing every permutation of v in every $u\ {}_m\!\Rightarrow v$ in G to see that this lemma hold true. □

Non-semilinearity of Context-Sensitive Jumping Languages

Theorem 26.
$\mathscr{L}(\Gamma_{CSG}, {}_j\Rightarrow)$ is not semilinear. Neither is $\mathscr{L}(\Gamma_{MONG}, {}_j\Rightarrow)$.

Idea.

- Recall that $\mathscr{L}(\Gamma_{MONG}, {}_m\Rightarrow)$ contains non-semilinear languages[2] and

□

[2] See Theorem 1 in “M. Kudlek, C. Martín-Vide, and Gh. Păun, Toward FMT (Formal Macroset Theory), In: *Pre-proceedings of the Workshop on Multiset Processing* (Curtea de Arges, August 21-25, 2000), pages 149-158.”

Non-semilinearity of Context-Sensitive Jumping Languages

Theorem 26.
$\mathscr{L}(\Gamma_{CSG}, {}_j\Rightarrow)$ is not semilinear. Neither is $\mathscr{L}(\Gamma_{MONG}, {}_j\Rightarrow)$.

Idea.

- Recall that $\mathscr{L}(\Gamma_{MONG}, {}_m\Rightarrow)$ contains non-semilinear languages[2] and
- $\mathscr{L}(\Gamma_{CSG}, {}_j\Rightarrow) \subseteq \mathscr{L}(\Gamma_{MONG}, {}_j\Rightarrow)$ follows from the definition.

□

[2]See Theorem 1 in "M. Kudlek, C. Martín-Vide, and Gh. Păun, Toward FMT (Formal Macroset Theory), In: *Pre-proceedings of the Workshop on Multiset Processing* (Curtea de Arges, August 21-25, 2000), pages 149-158."

Non-semilinearity of Context-Sensitive Jumping Languages

Theorem 26.
$\mathscr{L}(\Gamma_{CSG}, {}_j\Rightarrow)$ is not semilinear. Neither is $\mathscr{L}(\Gamma_{MONG}, {}_j\Rightarrow)$.

Idea.

- Recall that $\mathscr{L}(\Gamma_{MONG}, {}_m\Rightarrow)$ contains non-semilinear languages[2] and
- $\mathscr{L}(\Gamma_{CSG}, {}_j\Rightarrow) \subseteq \mathscr{L}(\Gamma_{MONG}, {}_j\Rightarrow)$ follows from the definition.
- We only need to prove that $\mathscr{L}(\Gamma_{MONG}, {}_m\Rightarrow) \subseteq \mathscr{L}(\Gamma_{CSG}, {}_j\Rightarrow)$.

□

[2] See Theorem 1 in "M. Kudlek, C. Martín-Vide, and Gh. Păun, Toward FMT (Formal Macroset Theory), In: *Pre-proceedings of the Workshop on Multiset Processing* (Curtea de Arges, August 21-25, 2000), pages 149-158."

Non-semilinearity of Context-Sensitive Jumping Languages

Theorem 26.

$\mathscr{L}(\Gamma_{CSG}, {}_j\Rightarrow)$ *is not semilinear. Neither is* $\mathscr{L}(\Gamma_{MONG}, {}_j\Rightarrow)$.

Idea.

- Recall that $\mathscr{L}(\Gamma_{MONG}, {}_m\Rightarrow)$ contains non-semilinear languages[2] and
- $\mathscr{L}(\Gamma_{CSG}, {}_j\Rightarrow) \subseteq \mathscr{L}(\Gamma_{MONG}, {}_j\Rightarrow)$ follows from the definition.
- We only need to prove that $\mathscr{L}(\Gamma_{MONG}, {}_m\Rightarrow) \subseteq \mathscr{L}(\Gamma_{CSG}, {}_j\Rightarrow)$.

□

[2]See Theorem 1 in "M. Kudlek, C. Martín-Vide, and Gh. Păun, Toward FMT (Formal Macroset Theory), In: *Pre-proceedings of the Workshop on Multiset Processing* (Curtea de Arges, August 21-25, 2000), pages 149-158."

Non-semilinearity of Context-Sensitive Jumping Languages

Theorem 26.

$\mathscr{L}(\Gamma_{CSG}, {}_j\Rightarrow)$ *is not semilinear. Neither is* $\mathscr{L}(\Gamma_{MONG}, {}_j\Rightarrow)$.

Idea.

- Recall that $\mathscr{L}(\Gamma_{MONG}, {}_m\Rightarrow)$ contains non-semilinear languages[2] and
- $\mathscr{L}(\Gamma_{CSG}, {}_j\Rightarrow) \subseteq \mathscr{L}(\Gamma_{MONG}, {}_j\Rightarrow)$ follows from the definition.
- We only need to prove that $\mathscr{L}(\Gamma_{MONG}, {}_m\Rightarrow) \subseteq \mathscr{L}(\Gamma_{CSG}, {}_j\Rightarrow)$.

□

Corollary 27.

$\mathscr{L}(\Gamma_{CFG}, {}_j\Rightarrow) \subset \mathscr{L}(\Gamma_{CSG}, {}_j\Rightarrow)$.

---

[2] See Theorem 1 in "M. Kudlek, C. Martín-Vide, and Gh. Păun, Toward FMT (Formal Macroset Theory), In: *Pre-proceedings of the Workshop on Multiset Processing* (Curtea de Arges, August 21-25, 2000), pages 149-158."

Closure Properties of Jumping Grammars - Work in Progress

Operations	$\cup$	$\cap$	Complement	Reversal
$\mathscr{L}(\Gamma_{JRG}, {}_j\Rightarrow)$	+	+	+	+
$\mathscr{L}(\Gamma_{RLG}, {}_j\Rightarrow)$	+	-	-	+
$\mathscr{L}(\Gamma_{CFG}, {}_j\Rightarrow)$	+	-	-	+?
$\mathscr{L}(\Gamma_{CSG}, {}_j\Rightarrow)$	+	-	-	
$\mathscr{L}(\Gamma_{MONG}, {}_j\Rightarrow)$	+	-	-	
$\mathscr{L}(\Gamma_{GG}, {}_j\Rightarrow)$	+	-	-	+

Table: Empty cell = unknown

Conclusion

Extensions and Future

Jumping Grammars

- Closure properties

Extensions and Future

Jumping Grammars

- Closure properties
- Right and Left jumps

Extensions and Future

Jumping Grammars

- Closure properties
- Right and Left jumps
- Alternative Jumping Context-Sensitive Grammars

Extensions and Future

Jumping Grammars

- Closure properties
- Right and Left jumps
- Alternative Jumping Context-Sensitive Grammars
- Relationship with Formal Macroset Theory

Extensions and Future

Jumping Grammars

- Closure properties
- Right and Left jumps
- Alternative Jumping Context-Sensitive Grammars
- Relationship with Formal Macroset Theory

Extensions and Future

Jumping Grammars

- Closure properties
- Right and Left jumps
- Alternative Jumping Context-Sensitive Grammars
- Relationship with Formal Macroset Theory

New Jumping Grammars with Regulation

- Addition of regulating mechanism (matrix, random-context, scattered-context, ...)

Extensions and Future

Jumping Grammars

- Closure properties
- Right and Left jumps
- Alternative Jumping Context-Sensitive Grammars
- Relationship with Formal Macroset Theory

New Jumping Grammars with Regulation

- Addition of regulating mechanism (matrix, random-context, scattered-context, ...)
- Grammar systems with jumping components?

Extensions and Future

Jumping Grammars

- Closure properties
- Right and Left jumps
- Alternative Jumping Context-Sensitive Grammars
- Relationship with Formal Macroset Theory

New Jumping Grammars with Regulation

- Addition of regulating mechanism (matrix, random-context, scattered-context, ...)
- Grammar systems with jumping components?
- ...

Thanks for your attention!