

CFG and natural languages (3)

Context-free languages (CFLs)

- can be recognized in polynomial time ($\mathcal{O}(n^3)$);
- are accepted by push-down automata;
- have nice closure properties (e.g., closure under homomorphisms, intersection with regular languages ...);
- satisfy a pumping lemma;
- can describe nested dependencies ($\{ww^R \mid w \in T^*\}$).

[HU79]

Parsing Beyond CFG	5	Introduction
--------------------	---	--------------

CFG and natural languages (4)

Question: Is CFG powerful enough to describe all natural language phenomena?

Answer: No. There are constructions in natural languages that cannot be adequately described with a context-free grammar.

Example: cross-serial dependencies in Dutch and in Swiss German.

Dutch:

- (1)
- ... dat Wim Jan Marie de kinderen zag helpen leren zwemmen
 ... that Wim Jan Marie the children saw help teach swim
 ‘... that Wim saw Jan help Marie teach the children to swim’

Parsing Beyond CFG	6	Introduction
--------------------	---	--------------

CFG and natural languages (5)

Swiss German:

- (2)
- ... das mer em Hans es huus hälfed aastriche
 ... that we Hans_{Dat} house_{Acc} helped paint
 ‘... that we helped Hans paint the house’
- (3)
- ... das mer d’chind em Hans es huus lönd hälfe aastriche
 ... that we the children_{Acc} Hans_{Dat} house_{Acc} let help paint
 ‘... that we let the children help Hans paint the house’

Swiss German uses case marking and displays cross-serial dependencies.

[Shi85] shows that Swiss German is not context-free.

Parsing Beyond CFG	7	Introduction
--------------------	---	--------------

CFG and natural languages (6)

If closure under homomorphisms and intersection with regular languages is given, the following holds:

A formalism that can generate cross-serial dependencies can also generate the copy language $\{ww \mid w \in \{a, b\}^*\}$.

The **copy language** is **not context-free**.

Therefore we are interested in extensions of CFG in order to describe all natural language phenomena.

Parsing Beyond CFG	8	Introduction
--------------------	---	--------------

CFG and natural languages (7)

Idea [Jos85]: characterize the amount of context-sensitivity necessary for natural languages.

Mildly context-sensitive formalisms have the following properties:

1. They generate (at least) all CFLs.
2. They can describe a limited amount of cross-serial dependencies.

In other words, there is a $n \geq 2$ up to which the formalism can generate all string languages $\{w^n \mid w \in T^*\}$.

3. They are polynomially parsable.
4. Their string languages are of constant growth.

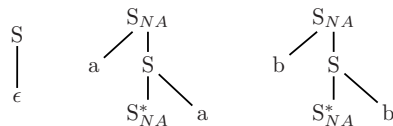
In other words, the length of the words generated by the grammar grows in a linear way, e.g., $\{a^{2^n} \mid n \geq 0\}$ does not have that property.

Polynomial extensions of CFG (1)

Tree Adjoining Grammars (TAG), [JLT75, JS97]:

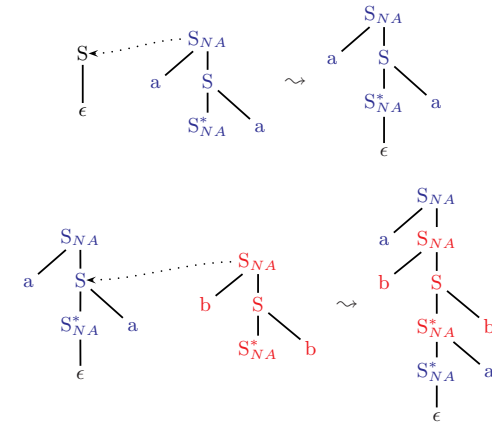
- Tree-rewriting grammar.
- Extension of CFG that allows to replace not only leaves but also internal nodes with new trees.
- Can generate the copy language.

Example: TAG for the copy language



Polynomial extensions of CFG (2)

Example: TAG derivation of $abab$:



Polynomial extensions of CFG (3)

Linear Context-free rewriting systems (LCFRS) and the equivalent Multiple Context-Free Grammars (MCFG), [VSWJ87, Wei88, SMFK91]

Idea: extension of CFG where non-terminals can span tuples of non-adjacent strings.

Example: $yield(A) = \langle a^n b^n, c^n d^n \rangle$, with $n \geq 1$.

The rewriting rules tell us how to compute the span of the lefthand side non-terminal from the spans of the righthand side non-terminals.

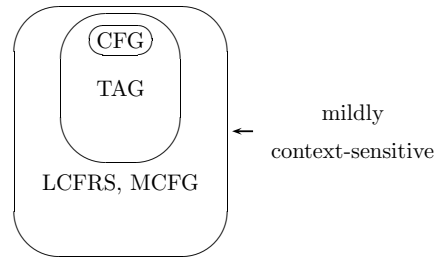
$$A(ab, cd) \rightarrow \varepsilon \quad A(aXb, cYd) \rightarrow A(X, Y) \quad S(XY) \rightarrow A(X, Y)$$

Generated string language: $\{a^n b^n c^n d^n \mid n \geq 1\}$.

LCFRS is more powerful than TAG but still mildly context-sensitive.

Polynomial extensions of CFG (6)

Summary:



In this course, we are interested in mildly context-sensitive formalisms.

Basic Definitions: Languages (1)**Definition 1 (Alphabet, word, language)**

1. An *alphabet* is a nonempty finite set X .
2. A string $x_1 \dots x_n$ with $n \geq 1$ and $x_i \in X$ for $1 \leq i \leq n$ is called a *nonempty word* on the alphabet X . X^+ is defined as the set of all nonempty words on X .
3. A new element $\varepsilon \notin X^+$ is added: $X^* := X^+ \cup \{\varepsilon\}$.

For each $w \in X^*$, the concatenation of w and ε is defined as follows: $w\varepsilon = \varepsilon w = w$.

ε is called the *empty word*, and each $w \in X^*$ is called a *word* on X .

4. A set L is called a *language* iff there is an alphabet X such that $L \subseteq X^*$.

Basic Definitions: Languages (2)**Definition 2 (Homomorphism)**

For two alphabets X and Y , a function $f : X^* \rightarrow Y^*$ is a *homomorphism* iff for all $v, w \in X^*$: $f(vw) = f(v)f(w)$.

Definition 3 (Length of a word) Let X be an alphabet, $w \in X^*$.

1. The *length* of w , $|w|$ is defined as follows: if $w = \varepsilon$, then $|w| = 0$. If $w = xw'$ for some $x \in X$, then $|w| = 1 + |w'|$.
2. For every $a \in X$, we define $|w|_a$ as the number of a s occurring in w : If $w = \varepsilon$, then $|w|_a = 0$, if $w = aw'$ then $|w|_a = |w'|_a + 1$ and if $w = bw'$ for some $b \in X \setminus \{a\}$, then $|w|_a = |w'|_a$.

Basic Definitions: CFG (1)**Definition 4 (Context-free grammar)**

A *context-free grammar (CFG)* is a tuple $G = \langle N, T, P, S \rangle$ such that

1. N and T are disjoint alphabets, the *nonterminals* and *terminals* of G ,
2. $P \subset N \times (N \cup T)^*$ is a finite set of *productions* (also called *rewriting rules*). A production $\langle A, \alpha \rangle$ is usually written $A \rightarrow \alpha$.
3. $S \in N$ is the *start symbol*.

Basic Definitions: CFG (2)**Definition 5 (Language of a CFG)**

Let $G = \langle N, T, P, S \rangle$ be a CFG. The (string) language $L(G)$ of G is the set $\{w \in T^* \mid S \xRightarrow{*} w\}$ where

- for $w, w' \in (N \cup T)^*$: $w \Rightarrow w'$ iff there is a $A \rightarrow \alpha \in P$ and there are $v, u \in (N \cup T)^*$ such that $w = vAu$ and $w' = v\alpha u$.
- $\xRightarrow{*}$ is the reflexive transitive closure of \Rightarrow :
 - $w \xRightarrow{0} w$ for all $w \in (N \cup T)^*$, and
 - for all $w, w' \in (N \cup T)^*$: $w \xRightarrow{n} w'$ iff there is a v such that $w \Rightarrow v$ and $v \xRightarrow{n-1} w'$.
 - for all $w, w' \in (N \cup T)^*$: $w \xRightarrow{*} w'$ iff there is a $i \in \mathbb{N}$ such that $w \xRightarrow{i} w'$.

A language L is called *context-free* iff there is a CFG G such that $L = L(G)$.

Basic Definitions: CFG (3)**Proposition 1 (Pumping lemma for context-free languages)**

Let L be a context-free language. Then there is a constant c such that for all $w \in L$ with $|w| \geq c$: $w = xv_1yv_2z$ with

- $|v_1v_2| \geq 1$,
- $|v_1yv_2| \leq c$, and
- for all $i \geq 0$: $xv_1^i y v_2^i z \in L$.

Basic Definitions: CFG (4)

Proposition 2 Context-free languages are closed under homomorphisms, i.e., for alphabets T_1, T_2 and for every context-free language $L_1 \subset T_1^*$ and every homomorphism $h : T_1^* \rightarrow T_2^*$, $h(L_1) = \{h(w) \mid w \in L_1\}$ is a context-free language.

Proposition 3 Context-free languages are closed under intersection with regular languages, i.e., for every context-free language L and every regular language L_r , $L \cap L_r$ is a context-free language.

Proposition 4 The copy language $\{ww \mid w \in \{a, b\}^*\}$ is not context-free.

Basic Definitions: Trees (1)**Definition 6 (Directed Graph)**

1. A *directed graph* is a pair $\langle V, E \rangle$ where V is a finite set of *vertices* and $E \subseteq V \times V$ is a set of *edges*.
2. For every $v \in V$, we define the *in-degree* of v as $|\{v' \in V \mid \langle v', v \rangle \in E\}|$ and the *out-degree* of v as $|\{v' \in V \mid \langle v, v' \rangle \in E\}|$.

E^+ is the transitive closure of E and E^* is the reflexive transitive closure of E .

Basic Definitions: Trees (2)**Definition 7 (Tree)**

A *tree* is a triple $\gamma = \langle V, E, r \rangle$ such that

- $\langle V, E \rangle$ is a directed graph and $r \in V$ is a special node, the *root node*.
- γ contains no cycles, i.e., there is no $v \in V$ such that $\langle v, v \rangle \in E^+$,
- only the root $r \in V$ has in-degree 0,
- every vertex $v \in V$ is accessible from r , i.e., $\langle r, v \rangle \in E^*$, and
- all nodes $v \in V - \{r\}$ have in-degree 1.

A vertex with out-degree 0 is called a *leaf*. The vertices in a tree are also called *nodes*.

Parsing Beyond CFG	21	Introduction
--------------------	----	--------------

Basic Definitions: Trees (3)

Definition 8 (Ordered Tree) A tree is *ordered* if it has an additional *linear precedence* relation $\prec \in V \times V$ such that

- \prec is irreflexive, antisymmetric and transitive,
- for all v_1, v_2 with $\{\langle v_1, v_2 \rangle, \langle v_2, v_1 \rangle\} \cap E^* = \emptyset$: either $v_1 \prec v_2$ or $v_2 \prec v_1$ and if there is either a $\langle v_3, v_1 \rangle \in E$ with $v_3 \prec v_2$ or a $\langle v_4, v_2 \rangle \in E$ with $v_1 \prec v_4$, then $v_1 \prec v_2$, and
- nothing else is in \prec .

We use Gorn addresses for nodes in ordered trees: The root address is ε , and the j th child of a node with address p has address pj .

Parsing Beyond CFG	22	Introduction
--------------------	----	--------------

Basic Definitions: Trees (4)

Definition 9 (Labeling) A *labeling* of a graph $\gamma = \langle V, E \rangle$ over a signature $\langle A_1, A_2 \rangle$ is a pair of functions $l : V \rightarrow A_1$ and $g : E \rightarrow A_2$ with A_1, A_2 possibly distinct.

Definition 10 (Syntactic tree) Let N and T be disjoint alphabets of non-terminal and terminal symbols. A *syntactic tree* (over N and T) is an ordered finite labeled tree such that $l(v) \in N$ for each vertex v with out-degree at least 1 and $l(v) \in (N \cup T \cup \{\varepsilon\})$ for each leaf v .

Parsing Beyond CFG	23	Introduction
--------------------	----	--------------

Basic Definitions: Trees (5)

Definition 11 (Tree Language of a CFG) Let $G = \langle N, T, P, S \rangle$ be a CFG.

1. A syntactic tree $\langle V, E, r \rangle$ over N and T is a *parse tree* in G iff
 - $l(v) \in (T \cup \{\varepsilon\})$ for each leaf v ,
 - for every $v_0, v_1, \dots, v_n \in V$, $n \geq 1$ such that $\langle v_0, v_i \rangle \in E$ for $1 \leq i \leq n$ and $\langle v_i, v_{i+1} \rangle \in \prec$ for $1 \leq i < n$, $l(v_0) \rightarrow l(v_1) \dots l(v_n) \in P$.
2. A parse tree $\langle V, E, r \rangle$ is a *derivation tree* in G iff $l(r) = S$.
3. The *tree language* of G is

$$L_T(G) = \{\gamma \mid \gamma \text{ is a derivation tree in } G\}$$

Parsing Beyond CFG	24	Introduction
--------------------	----	--------------

Basic Definitions: Trees (6)**Definition 12 (Weak and Strong Equivalence)**

Let F_1, F_2 be two grammar formalisms.

- F_1 and F_2 are *weakly equivalent* iff for each instance G_1 of F_1 there is an instance G_2 of F_2 that generates the same string language and vice versa.
- F_1 and F_2 are *strongly equivalent* iff for both formalisms the notion of a tree language is defined and, furthermore, for each instance G_1 of F_1 there is an instance G_2 of F_2 that generates the same tree language and vice versa.

References

- [HU79] John E. Hopcroft and Jeffrey D. Ullman. *Introduction to Automata Theory, Languages and Computation*. Addison Wesley, 1979.
- [JLT75] Aravind K. Joshi, Leon S. Levy, and Masako Takahashi. Tree Adjunct Grammars. *Journal of Computer and System Science*, 10:136–163, 1975.
- [Jos85] Aravind K. Joshi. Tree adjoining grammars: How much contextsensitivity is required to provide reasonable structural descriptions? In D. Dowty, L. Karttunen, and A. Zwicky, editors, *Natural Language Parsing*, pages 206–250. Cambridge University Press, 1985.
- [JS97] Aravind K. Joshi and Yves Schabes. Tree-Adjoining

Grammars. In G. Rozenberg and A. Salomaa, editors, *Handbook of Formal Languages*, pages 69–123. Springer, Berlin, 1997.

- [Kal10] Laura Kallmeyer. *Parsing Beyond Context-Free Grammars*. Cognitive Technologies. Springer, Heidelberg, 2010.
- [SBMSN87] Walter J. Savitch, Emmon Bach, William Marxh, and Gila Safran-Naveh, editors. *The Formal Complexity of Natural Language*. Studies in Linguistics and Philosophy. Reidel, Dordrecht, Holland, 1987.
- [Shi85] Stuart M. Shieber. Evidence against the context-freeness of natural language. *Linguistics and Philosophy*, 8:333–343, 1985. Reprinted in [SBMSN87].
- [SMFK91] Hiroyuki Seki, Takahashi Matsumura, Mamoru Fujii, and Tadao Kasami. On multiple context-free

grammars. *Theoretical Computer Science*, 88(2):191–229, 1991.

- [VSWJ87] K. Vijay-Shanker, David J. Weir, and Aravind K. Joshi. Characterizing structural descriptions produced by various grammatical formalisms. In *Proceedings of ACL*, Stanford, 1987.
- [Wei88] David J. Weir. *Characterizing Mildly Context-Sensitive Grammar Formalisms*. PhD thesis, University of Pennsylvania, 1988.