Mildly Context-Sensitive Grammar Formalisms:

LMG and RCG

Laura Kallmeyer
Heinrich-Heine-Universität Düsseldorf

Sommersemester 2011

Overview

1. Introduction
2. Literal Movement Grammar
3. Range Concatenation Grammar

Introduction (1)

- So far, when dealing with LCFRS, we required the grammar to be linear:

  Every variable in the left-hand side of a rule appears exactly once in its right-hand side and vice versa.
- If we drop this constraint, we obtain more general grammars.
- So far, it did not matter whether we instantiated the variables with strings or ranges.
- If we drop linearity, this difference matters.

  In Literal Movement Grammars, variables are instantiated with strings, while in Range Concatenation Grammars, variables are instantiated with ranges. In the latter, all range concatenations must be possible with respect to some word in the language.

Introduction (2)

Example:

- Grammar 1: rules

  $S(aXb) \to B(XX)$, $B(bX) \to B(X)$, $B(\varepsilon) \to \varepsilon$

  RCG string language: $\{ab\}$

  LMG string language: $\{ab^k \mid k \geq 1\}$
- Grammar 2: rules

  $S(XY) \to A(Xb)C(Y)$, $A(ab) \to \varepsilon$, $C(b) \to \varepsilon$, $C(c) \to \varepsilon$

  RCG string language: $\{ab\}$

  LMG string language: $\{ab, ac\}$

Introduction (3)

- Literal Movement Grammars were defined first, in [Groenink, 1995, Groenink, 1996, Groenink, 1997] (see also [Kracht, 2003] for an introduction to LMG).
- Range Concatenation Grammars were inspired by Groenink's work on LMG and were first defined in [Boullier, 1998a].

Introduction (4)

LMGs and RCGs are straightforward generalizations of LCFRSs:

Definition 1 (LMG and RCG) *A* Range Concatenation Grammar *and* Literal Movement Grammar *is a tuple* $G = \langle N, T, V, S, P\rangle$ *such that*

- N, T, V *and* S *are defined as in an LCFRS;*
- P *is a finite set of* clauses

$$A_0(\alpha_{01}, \dots, \alpha_{0a_0}) \to A_1(\alpha_{11}, \dots, \alpha_{1a_1}) \dots A_n(\alpha_{n1}, \dots, \alpha_{na_n})$$

with $n \geq 0$ *where* $A_i \in N, \alpha_{ij} \in (T \cup V)^*$ *and* a_i *the arity of* A_i*.*

Literal Movement Grammars (1)

LMG variables are instantiated wrt. strings:

Definition 2 (LMG clause instantiation) *Let* $G = \langle N, T, V, S, P\rangle$ *be a LMG.*

For a rule $c = A(\vec{\alpha}) \to A_1(\vec{\alpha_1}) \dots A_m(\vec{\alpha_m}) \in P$*, every function* $f : \{x \mid x \in V, x \text{ occurs in } c\} \to T^*$ *is an* instantiation *of* c.

We call $A(f(\vec{\alpha})) \to A_1(f(\vec{\alpha_1})) \dots A_m(f(\vec{\alpha_m}))$ *then an* instantiated clause *where* f *is extended as follows:*

1. $f(\varepsilon) = \varepsilon$;
2. $f(t) = t$ *for all* $t \in T$;
3. $f(xy) = f(x)f(y)$ *for all* $x, y \in T^*$;
4. $f(\langle\alpha_1, \dots, \alpha_m\rangle) = (\langle f(\alpha_1), \dots, f(\alpha_m)\rangle)$ *for all* $(\langle\alpha_1, \dots, \alpha_m\rangle) \in [(T \cup V)^*]^m$, $m \geq 1$.

Literal Movement Grammars (2)

Ex.: LMG for $L_{MIX} = \{w \mid w \in \{a, b, c\}^*, |w|_a = |w|_b = |w|_c\}$

$S(\varepsilon) \to \varepsilon$	$S(XaY) \to A(XY)$
$A(XbY) \to B(XY)$	$B(XcY) \to S(XY)$

Possible rule instantiations for $A(XbY) \to B(XY)$:

- $f(X) = ab, f(Y) = cc, \leadsto A(abbcc) \to B(abcc)$
- $f(X) = cba, f(Y) = c, \leadsto A(cbabc) \to B(cbac)$
- $f(X) = aaa, f(Y) = a, \leadsto A(aaaba) \to B(aaaa)$ (this one gets never used in an actual derivation)

The string language is defined as for LCFRSs.

Literal Movement Grammars (3)

Definition 3 (LMG string language) *Let $G = \langle N, T, V, S, P\rangle$ be a LMG.*

1. *The set $L_{pred}(G)$ of instantiated predicates $A(\vec{\tau})$ where $A \in N$ and $\vec{\tau} \in (T^*)^k$ for some $k \geq 1$ is defined by the following deduction rules:*

$$\frac{}{A(\vec{\tau})}\ A(\vec{\tau}) \rightarrow \varepsilon \textit{ is an instantiated clause}$$

$$\frac{A_1(\vec{\tau_1}) \dots A_m(\vec{\tau_m})}{A(\vec{\tau})}\ A(\vec{\tau}) \rightarrow A_1(\vec{\tau_1}) \dots A_m(\vec{\tau_m}) \textit{ is an instantiated clause}$$

2. *The string language of G is*

$$\{w \in T^* \mid S(w) \in L_{pred}(G)\}.$$

Literal Movement Grammars (4)

Ex: Derivation of $w = aabcbc$ (deduction of $S(aabcbc)$)

$$\frac{}{S(\varepsilon)}\ S(\varepsilon) \rightarrow \varepsilon \qquad \frac{S(\varepsilon)}{B(c)}\ B(XcY) \rightarrow S(XY)$$

$$\frac{B(c)}{A(bc)}\ A(XbY) \rightarrow B(XY) \qquad \frac{A(bc)}{S(abc)}\ S(XaY) \rightarrow A(XY)$$

$$\frac{S(abc)}{B(abcc)}\ B(XcY) \rightarrow S(XY) \qquad \frac{B(abcc)}{A(abcbc)}\ A(XbY) \rightarrow B(XY)$$

$$\frac{A(abcbc)}{S(aabcbc)}\ S(XaY) \rightarrow A(XY)$$

Literal Movement Grammars (5)

- The general definition of LMGs allows to have any combination of variables and terminals in the components of the left-hand side and the right-hand side of a clause.
- In particular, in the instantiated clauses, strings can be copied or deleted and we can combine strings into new strings.
- If all this is disallowed, we obtain LCFRSs.

Literal Movement Grammars (6)

Definition 4 (Linear Context-Free Rewriting Systems) *A LMG is*

- non-combinatorial *if for every clause $c \in P$, all the arguments in the right-hand side of c are single variables.*
- bottom-up (top-down) linear *if for every $c \in P$, no variable appears more than once in the left-hand (right-hand) side of c.*
- linear *if it is top-down and bottom-up linear.*
- bottom-up (top-down) non-erasing *if for every $c \in P$, each variable occurring in the right-hand (left-hand) side of c occurs also in its left-hand (right-hand) side.*
- non-erasing *if it is top-down and bottom-up non-erasing.*
- *a* Linear Context-Free Rewriting System (LCFRS) *if it is non-combinatorial, linear and non-erasing.*

Literal Movement Grammars (7)

Other interesting LMGs are parallel multiple context-free grammars (PMCFG) and simple LMGs:

Definition 5 (PMCFG) *An LMG is a* parallel multiple context-free grammar *(PMCFG) if it is non-combinatorial, top-down non-erasing and top-down linear.*

Ex.:

$$S(XXX) \to A(X) \quad A(aX) \to A(X)$$
$$A(bX) \to A(X) \quad A(\varepsilon) \to A(\varepsilon)$$

Literal Movement Grammars (8)

Proposition 1 *The set of string languages generated by PMCFGs properly contains the set of LCFRLs.*

Example of a PMCFG that generates a language that is not of constant growth:

PMCFG for $\{a^{2^n} \mid n \geq 0\}$:

$$S(a) \to \varepsilon \quad S(XX) \to S(X)$$

Literal Movement Grammars (9)

Definition 6 (Simple LMG)

An LMG is simple *if it is non-combinatorial, bottom-up non-erasing and bottom-up linear.*

In other words, every variable in a clause occurs exactly once in its left-hand side. Furthermore, the right-hand side components are single variables.

Simple LMG for $\{a^{2^n} \mid n \geq 0\}$:

$$S(XY) \to S(X)eq(X,Y) \quad S(a) \to \varepsilon$$
$$eq(aX, aY) \to eq(X,Y) \quad eq(a,a) \to \varepsilon$$

Literal Movement Grammars (10)

Simple LMG for $\{(a^m b^m)^n \mid m, n \geq 1\}$:

$$S(XY) \to T(X,Y) \quad T(X,\varepsilon) \to A(X)$$
$$T(X,YZ) \to A(X)eq(X,Y)T(X,Z)$$
$$A(aXb) \to A(X) \quad A(ab) \to \varepsilon$$
$$eq(aX,aY) \to eq(X,Y) \quad eq(bX,bY) \to eq(X,Y) \quad eq(\varepsilon,\varepsilon) \to \varepsilon$$

Literal Movement Grammars (11)

Simple LMGs generate the entire set of all polynomial languages:

Proposition 2 *The set of string languages generated by simple LMGs is exactly the class PTIME, i.e., the class of all polynomial languages [Groenink, 1996].*

PMCFGs are less powerful than simple LMGs. [Ljunglöf, 2005] extends PMCFG with intersection, which leads to a formalism equivalent to simple LMG.

Proposition 3

- *For every PMCFG G, there is a simple LMG G' such that $L(G) = L(G')$.*
- *There exists a simple LMG G such that there is no PMCFG G with $L(G) = L(G')$.*

Range Concatenation Grammars (1)

Now we keep the syntax of the clauses but instantiate variables with ranges with respect to a given string w. This leads to Range Concatenation Grammars (RCGs)
[Boullier, 1998a, Boullier, 1998b, Boullier, 1999, Boullier, 2000].

Example:

$S(X) \to M(X, X, X)$

$M(bX, Y, Z) \to M(X, Y, Z)$	$M(cX, Y, Z) \to M(X, Y, Z)$
$M(X, aY, Z) \to M(X, Y, Z)$	$M(X, cY, Z) \to M(X, Y, Z)$
$M(X, Y, aZ) \to M(X, Y, Z)$	$M(X, Y, bZ) \to M(X, Y, Z)$
$M(aX, bY, cZ) \to M(X, Y, Z)$	$M(\varepsilon, \varepsilon, \varepsilon) \to \varepsilon$

$L(G) = MIX = \{w \mid w \in \{a, b, c\}^*, |w|_a = |w|_b = |w|_c\}$

Range Concatenation Grammars (2)

Definition 7 (Clause instantiation) *Let $G = (N, T, V, P, S)$ be a RCG. For a given clause $c = A_0(\vec{\alpha_0}) \to A_1(\vec{\alpha_1}) \cdots A_m(\vec{\alpha_m})$ $(0 \leq m)$ and a string $w = t_1 \dots t_n$,*

1. *an* instantiation *of c with respect to w consists of a function $f : \{t' \mid t'$ is an occurrence of some $t \in T$ in the clause$\} \cup V \cup \{Eps_{i,j} \mid 1 \leq i \leq m, 1 \leq j \leq dim(A_i), \vec{\alpha_i}(j) = \varepsilon\} \to \{\langle i, j\rangle \mid i \leq j, i, j \in \mathbb{N}\}$ such that*
   a) *for all occurrences t' of a $t \in T$ in the clause, $f(t') := \langle i, i+1\rangle$ for some $i, 0 \leq i < n$ such that $t_i = t$,*
   b) *for all $X \in V$, $f(X) = \langle j, k\rangle$ for some $0 \leq j \leq k \leq n$,*
   c) *for all x, y adjacent in one of the elements of $\vec{\alpha_i}$ $(0 \leq i \leq m)$, there are l, j, r with $f(x) = \langle l, j\rangle, f(y) = \langle j, r\rangle$; we then define $f(xy) = \langle l, r\rangle$,*
   d) *for all*

$Eps \in \{Eps_{i,j} \mid 1 \leq i \leq m, 1 \leq j \leq dim(A_i), \vec{\alpha_i}(j) = \varepsilon\}$, there is a k, $0 \leq k \leq n$ with $f(Eps) = \langle k, k\rangle$; we then define for every ε-argument $\vec{\alpha_i}(j)$ that $f(\vec{\alpha_i}(j)) = f(Eps_{i,j})$,

2. *if f is an instantiation of c with respect to w, then $A_0(f(\vec{\alpha_0})) \to A_1(f(\vec{\alpha_1})) \cdots A_m(f(\vec{\alpha_m}))$ is an* instantiated clause *where $f(\langle x_1, \dots, x_k\rangle) = \langle f(x_1), \dots, f(x_k)\rangle$.*

Range Concatenation Grammars (3)

In each RCG derivation step, the left-hand side of an instantiated clause is replaced by its right-hand side.

In other words, the set of instantiated rules with respect to some given w is used as a CFG with start symbol $S(\langle\langle 0, |w|\rangle\rangle)$.

The *string language* of an RCG G is

$$L(G) = \{w \in T^* \mid S(\langle\langle 0, |w|\rangle\rangle) \stackrel{*}{\Rightarrow} \varepsilon \text{ with respect to } w\}.$$

Range Concatenation Grammars (4)

Ex.: $w = abc$, RCG for the MIX language.

Derivation:

$$\begin{array}{rcl}
S(\langle\langle 0,3\rangle\rangle) & \rightarrow & M(\langle\langle 0,3\rangle, \langle 0,3\rangle, \langle 0,3\rangle\rangle) \\
& \rightarrow & M(\langle\langle 0,3\rangle, \langle 1,3\rangle, \langle 0,3\rangle\rangle) \\
& \rightarrow & M(\langle\langle 0,3\rangle, \langle 1,3\rangle, \langle 1,3\rangle\rangle) \\
& \rightarrow & M(\langle\langle 0,3\rangle, \langle 1,3\rangle, \langle 2,3\rangle\rangle) \\
& \rightarrow & M(\langle\langle 1,3\rangle, \langle 2,3\rangle, \langle 3,3\rangle\rangle) \\
& \rightarrow & M(\langle\langle 2,3\rangle, \langle 2,3\rangle, \langle 3,3\rangle\rangle) \\
& \rightarrow & M(\langle\langle 3,3\rangle, \langle 2,3\rangle, \langle 3,3\rangle\rangle) \\
& \rightarrow & M(\langle\langle 3,3\rangle, \langle 3,3\rangle, \langle 3,3\rangle\rangle) \\
& \rightarrow & \varepsilon
\end{array}$$

Range Concatenation Grammars (5)

The definitions of combinatorial, linear and non-erasing are taken over from LMGs.

Definition 8 (Simple Range Concatenation Grammar) *An RCG is* simple *if it is non-combinatorial, linear and non-erasing.*

In the LCFRS/simple RCG case, it does not matter which string language definition we adopt, the result is the same.

Proposition 4 *LCFRS and simple RCG are equivalent.*

Range Concatenation Grammars (6)

[Boullier, 1998a] shows the following:

Proposition 5

1. *For any RCG, there is an equivalent non-combinatorial RCG.*
2. *For any non-combinatorial bottom-up erasing RCG, there is an equivalent non-combinatorial bottom-up non-erasing RCG.*
3. *For any non-combinatorial bottom-up non-erasing top-down erasing RCG, there is an equivalent non-combinatorial non-erasing RCG.*

In other words, the possibilities of combinatorial clauses and erasing clauses do not increase the generative capacity of the grammar. The crucial property for RCG's being more powerful than simple RCG is the possible non-linearity of the clauses.

Range Concatenation Grammars (7)

Proposition 6 *The set of string languages generated by RCGs is exactly the class PTIME of all polynomial languages ([Bertsch and Nederhof, 2001]).*

- The fact that every language generated by an RCG is polynomial is confirmed by the existence of polynomial parsing algorithms [Kallmeyer et al., 2009].
- The inclusion of all polynomial languages in the set of RCG string languages is shown in Appendix A of [Bertsch and Nederhof, 2001] by constructing an equivalent RCG for a given two-way alternating finite automaton with k heads. It is known that two-way alternating finite automata recognize exactly the class PTIME.

References

[Bertsch and Nederhof, 2001] Bertsch, E. and Nederhof, M.-J. (2001). On the complexity of some extensions of RCG parsing. In *Proceedings of the Seventh International Workshop on Parsing Technologies*, pages 66–77, Beijing, China.

[Boullier, 1998a] Boullier, P. (1998a). A generalization of mildly context-sensitive formalisms. In *Proceedings of the Fourth International Workshop on Tree Adjoining Grammars and Related Formalisms (TAG+4)*, pages 17–20, University of Pennsylvania, Philadelphia.

[Boullier, 1998b] Boullier, P. (1998b). A Proposal for a Natural Language Processing Syntactic Backbone. Technical Report 3342, INRIA.

[Boullier, 1999] Boullier, P. (1999). Chinese numbers, mix,

scrambling, and range concatenation grammars. In *Proceedings of the 9th Conference of the European Chapter of the Association for Computational Linguistics (EACL'99)*, pages 53–60, Bergen, Norway.

[Boullier, 2000] Boullier, P. (2000). Range Concatenation Grammars. In *Proceedings of the Sixth International Workshop on Parsing Technologies (IWPT2000)*, pages 53–64, Trento, Italy.

[Groenink, 1995] Groenink, A. V. (1995). Literal movement grammars. In *Proceedings of the 7th EACL Conference*.

[Groenink, 1996] Groenink, A. V. (1996). Mild context-sensitivity and tuple-based generalizations of context-free grammar. Report CS-R9634, Centrum voor Wiskunde en Informatica, Amsterdam.

[Groenink, 1997] Groenink, A. V. (1997). *Surface Without Structure. Word Order and Tractability in Natural Language Analysis*. PhD thesis, Utrecht University.

[Kallmeyer et al., 2009] Kallmeyer, L., Maier, W., and Parmentier, Y. (2009). An Earley Parsing Algorithm for Range Concatenation Grammars. In *Proceedings of ACL 2009*, Singapore.

[Kracht, 2003] Kracht, M. (2003). *The Mathematics of Language*. Number 63 in Studies in Generative Grammar. Mouton de Gruyter, Berlin.

[Ljunglöf, 2005] Ljunglöf, P. (2005). A polynomial time extension of parallel Multiple Context-Free Grammar. In *Logical Aspects of Computational Linguistics*, volume 3492 of *Lecture Notes in Computer Science*, pages 177–188. Springer, Berlin/Heidelberg.