Parsing Beyond Context-Free Grammars:

Linear Context-Free Rewriting Languages: Formal Properties

Laura Kallmeyer
Heinrich-Heine-Universität Düsseldorf

Sommersemester 2016

Overview

1. Pumping Lemma
   (a) Intuition
   (b) The Pumping Lemma
   (c) Applications
2. Closure Properties
   (a) Substitution
   (b) Union, Concatenation, Kleene closure
   (c) Intersection with Regular Languages

Pumping Lemma: Intuition (1)

LCFRS have a context-free backbone: the productions constitute a *generalized context-free grammar*. A derivation step consists of replacing a lhs of a production with its rhs.

Example (LCFRS for the double copy language):

$$S \to f_1[A] \qquad A \to f_2[A] \qquad A \to f_3[A] \qquad A \to f_4[\,] \qquad A \to f_5[\,]$$

$$f_1[\langle X, Y, Z\rangle] = \langle XYZ\rangle \qquad f_4[\,] = \langle a, a, a\rangle$$

$$f_2[\langle X, Y, Z\rangle] = \langle aX, aY, aZ\rangle \qquad f_5[\,] = \langle b, b, b\rangle$$

$$f_3[\langle X, Y, Z\rangle] = \langle bX, bY, bZ\rangle$$

Derivation in underlying generalized CFG:

$$S \Rightarrow f_1(A) \Rightarrow f_1(f_3(A)) \Rightarrow f_1(f_3(f_2(A))) \Rightarrow f_1(f_3(f_2(f_4())))$$

The term $f_1(f_3(f_2(f_4())))$ denotes $\langle baabaabaa\rangle$.

Pumping Lemma: Intuition (2)

- In such a derivation, the expansion of a non-terminal A does not depend on the context A occurs in.
- Consequently, as in the case of CFG, if we have a derivation

$$A \stackrel{+}{\Rightarrow} f_1(\dots f_2(\dots f_k(\dots A \dots)\dots)\dots)$$

then this part of the derivation can be iterated, i.e., we can also have

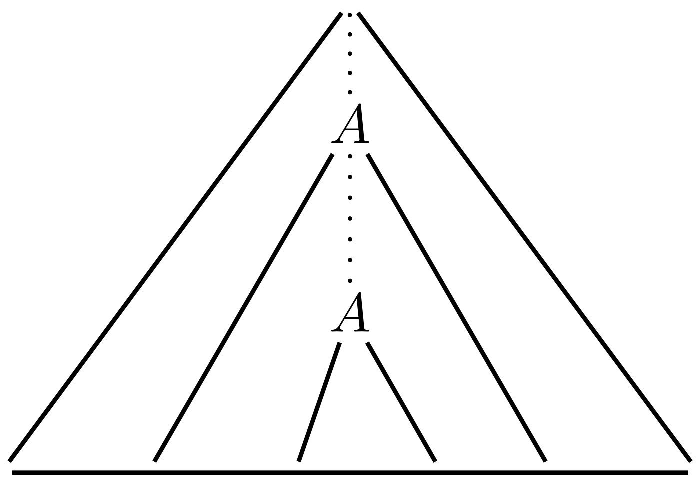
$$A \stackrel{+}{\Rightarrow} f_1(\dots f_2(\dots f_k(\dots f_1(\dots f_2(\dots f_k(\dots A \dots)\dots)\dots)\dots)\dots)\dots)$$

etc.

Pumping Lemma (1)

Question: What does this mean for the string language?

Assume that we have such an iteration, i.e., in the derivation tree, we have



with no other derivation $B \stackrel{+}{\Rightarrow} B$ in the subtree corresponding to $A \stackrel{+}{\Rightarrow} A$. The part between the two A nodes can be iterated.

Pumping Lemma (2)

- Let m be the fan-out (the arity) of A. Then the higher A spans an m-tuple of strings and the lower A spans a (smaller) m-tuple of strings that is part of the m-tuple of the higher A.

  Assume that $\langle w_1, \dots, w_m \rangle$ is the span of the lower A.

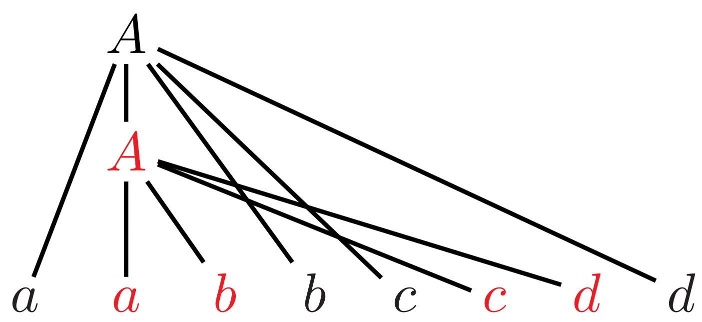
- There are different cases for how the components of the lower A are part of the span of the higher A. Either w_i is part of the ith component ($1 \leq i \leq m$) or there are components of the higher A that do not contain parts of the span of the lower A.

Pumping Lemma (3)

Case 1: w_i is part of the ith component of the higher A $(1 \leq i \leq m)$. Then the span of the higher A has the form $\langle v_1 w_1 u_1, \dots, v_m w_m u_m \rangle$.

Consequently (iteration), $\langle v_1^k w_1 u_1^k, \dots, v_m^k w_m u_m^k \rangle$ is also in the yield of A.

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



Iteration:

$a^n abb^n c^n cdd^n$

The iterated parts are present in the original string (in the tree with just two As on the path).

Pumping Lemma (4)

Case 2: The $w_1, \dots, w_m$ are part of only j components ($j < m$) of the span of the higher A. Then, when iterating, the components of the higher A go again into only j components, i.e., the $m - j$ components that do not contain any of the $w_1, \dots, w_m$ must be added to the other ones.

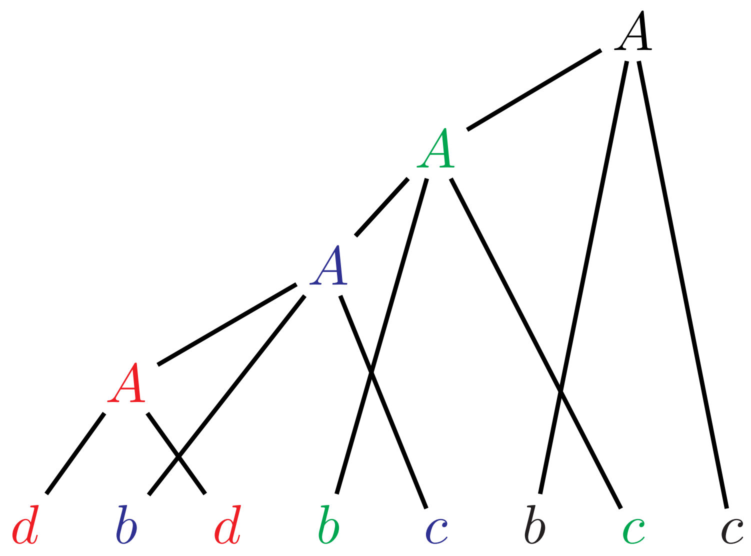
Consequently, in a component of the higher A, we either have the form $v_1 w_i v_2 w_{i+1} \dots v_{k-1} w_{i+k} v_k$ or a form u (without components from the lower A).

In the next iteration, the u will be added to one of the other components. This can be repeated and will lead to iterations of strings that are concatenations of some of the u and some of the v_i. These iterated strings are not necessarily present in the span of the higher A, before iteration.

Pumping Lemma (5)

Example:

$S(XYZU) \to A(XYZ, U), A(XbY, c) \to A(X, Y), A(d, d) \to \varepsilon$



Iteration pattern: $dbd(bc)^n c$

Here, the iterated parts are not present in the original string (in the tree with just two As on the path).

Pumping Lemma (6)

Along these lines, [Seki et al., 1991] show the following pumping lemma for k-MCFLs, the class of languages generated by k-MCFGs:

Proposition 1 (Pumping Lemma for k-MCFLs) *For any k-MCFL L, if L is an infinite set then there exist some $u_j \in T^*$ $(1 \leq j \leq k+1)$, $v_j, w_j, s_j \in T^*(1 \leq j \leq k)$ which satisfy the following conditions:*

1. $\Sigma_{j=1}^{k} |v_j s_j| > 0$, *and*
2. *for any* $i \geq 0$,

$$z_i = u_1 v_1^i w_1 s_1^i u_2 v_2^i w_2 s_2^i \dots u_k v_k^i w_k s_k^i u_{k+1} \in L$$

Pumping Lemma (7)

- Note that the pumping lemma is only *existential* in the sense that it does not say that within each string that is long enough we find pumpable substrings.
- It only says that there exist strings in the language that are of a limited length and that contain pumpable substrings.
- In contrast to this, the CFG pumping lemma is *universal*: within every string of sufficient length we find two pumpable substrings of a limited distance.

Pumping Lemma: Applications (1)

Proposition 2 *For every $k \geq 1$, the language $\{a_1^n a_2^n \dots a_{2k+1}^n \,|\, n \geq 0\}$ is not a k-MCFL.*

Proof: Assume that it is a k-MCFL. Then it satisfies the pumping lemma with $2k$ pumpable strings. At least one of these strings is not empty and none of them can contain different terminals. However, if at most $2k$ strings are pumped, we necessarily obtain strings that are not in the language. Contradiction.

Pumping Lemma: Applications (2)

For every $k \geq 1$, the language $\{a_1^n a_2^n \dots a_{2k+1}^n \mid n \geq 0\}$ is a $(k+1)$-MCFL.

It is generated by an MCFG/LCFRS with the following rules:

$$S(X_1 X_2 \dots X_k X_{k+1}) \to A(X_1, X_2, \dots, X_k, X_{k+1})$$

$$A(a_1 X_1 a_2, a_3 X_2 a_4, \dots, a_{2k-1} X_k a_{2k}, a_{2k+1} X_{k+1}) \to A(X_1, X_2, \dots, X_k, X_{k+1})$$

$$A(\varepsilon, \dots, \varepsilon) \to \varepsilon$$

Proposition 3 *k-MCFL is a proper subset of $(k+1)$-MCFL.*

Closure Properties

[Seki et al., 1991] show the following closure properties for k-MCFL:

Proposition 4 *For every $k \geq 1$, the class k-MCFL*

- *is closed under substitution;*
- *is closed under union, concatenation, Kleene closure, ε-free Kleene closure;*
- *is closed under intersection with regular languages.*

Closure Properties: Substitution

k-MCFL being closed under substitution means:

If L is a k-MCFL over the terminal alphabet T and f assigns a k-MCFL to every $t \in T$, then

$$f(L) = \{w_1 \dots w_n \mid \text{ there is a } t_1 \dots t_n \in L \text{ with } w_i \in f(t_i) \text{ for } 1 \leq i \leq n\}$$

is also a k-MCFL.

Idea of the construction of the new k-MCFG from the original one and the ones for the images of the terminals: take the original and replace every terminal a in a lhs with a new variable X_a and add $S_a(X_a)$ to the rhs where S_a the start symbol of the grammar of the image of a.

Closure Properties: Union and Concatenation

Let L_1, L_2 be languages generated by the k-MCFGs G_1, G_2 with start symbols S_1, S_2 respectively (and without loss of generality disjoint sets of non-terminals).

- The union, $L_1 \cup L_2$ is generated by the grammar with the rules from G_1 and G_2 and additional rules $S(X) \rightarrow S_1(X)$, $S(X) \rightarrow S_2(X)$ where S is a new start symbol.
- The concatenation $\{w_1w_2 \,|\, w_1 \in L_1, w_2 \in L_2\}$ is generated by the grammar with the rules from G_1 and G_2 and an additional rule $S(XY) \rightarrow S_1(X)S_2(Y)$ where S is a new start symbol.

Closure Properties: Kleene star

Let L be a language generated by the k-MCFG G.

- If we add the rules $S'(XY) \rightarrow S(X)S'(Y)$ and $S'(\varepsilon) \rightarrow \varepsilon$ where S' is a new start symbol, we generate the Kleene closure L^* of L.
- If we add the rules $S'(XY) \rightarrow S(X)S'(Y)$ and $S'(X) \rightarrow S(X)$ where S' is a new start symbol, we generate the ε-free Kleene closure L^+ of L.

Closure Properties: Intersection with regular lang. (1)

Construction idea: enrich the non-terminals A with lists of states $q_1, q'_1, \dots, q_{dim(A)}, q'_{dim(A)}$ where the path from q_i to q'_i is the path traversed while processing the ith component of A.

Example:

Take the copy language, generated by an MCFG with

$$S(XY) \rightarrow A(X,Y)$$

$$A(aX,aY) \rightarrow A(X,Y) \quad A(bX,bY) \rightarrow A(X,Y) \quad A(\varepsilon,\varepsilon) \rightarrow \varepsilon$$

Intersect with $a^*b^*a^*b^*$, generated by a DFA with $Q = F = \{q_0, q_1, q_2, q_3\}$, initial state q_0 and

$\delta(q_0, a) = q_0, \delta(q_0, b) = q_1, \delta(q_1, b) = q_1, \delta(q_1, a) = q_2,$

$\delta(q_2, a) = q_2, \delta(q_2, b) = q_3, \delta(q_3, b) = q_3.$

Closure Properties: Intersection with regular lang. (2)

Result:

a^* $S[q_0,q_0](XY) \to A[q_0,q_0,q_0,q_0](X,Y)$,
$A[q_0,q_0,q_0,q_0](aX,aY) \to A[q_0,q_0,q_0,q_0](X,Y)$,
$A[q_0,q_0,q_0,q_0](\varepsilon,\varepsilon) \to \varepsilon$

b^+ $S[q_0,q_1](XY) \to A[q_0,q_1,q_1,q_1](X,Y)$,
$A[q_0,q_1,q_1,q_1](bX,bY) \to A[q_1,q_1,q_1,q_1](X,Y)$,
$A[q_1,q_1,q_1,q_1](bX,bY) \to A[q_1,q_1,q_1,q_1](X,Y)$,
$A[q_1,q_1,q_1,q_1](\varepsilon,\varepsilon) \to \varepsilon$

$a^+b^+a^+b^+$ $S[q_0,q_3](XY) \to A[q_0,q_1,q_1,q_3](X,Y)$,
$A[q_0,q_1,q_1,q_3](aX,aY) \to A[q_0,q_1,q_2,q_3](X,Y)$,
$A[q_0,q_1,q_2,q_3](aX,aY) \to A[q_0,q_1,q_2,q_3](X,Y)$,
$A[q_0,q_1,q_2,q_3](bX,bY) \to A[q_1,q_1,q_3,q_3](X,Y)$,
$A[q_1,q_1,q_3,q_3](bX,bY) \to A[q_1,q_1,q_3,q_3](X,Y)$,
$A[q_1,q_1,q_3,q_3](\varepsilon,\varepsilon) \to \varepsilon$

Closure Properties: Intersection with regular lang. (3)

Proposition 5 *[Kallmeyer, 2010]*

$L = \{(a^m b^m)^n \mid m, n \geq 1\}$ *is not an MCFL.*

Proof: We assume that there is a fixed k such that there is a k-MCFG generating L.

We intersect L with the regular language $(a^+b^+)^{k+1}$, which yields $L' = \{(a^m b^m)^{k+1} \mid m \geq 1\}$. L' does not satisfy the pumping lemma for k-MCFL since the iterated parts in the pumping lemma must each consist of either as or bs (otherwise we would increase the number of substrings a^m and b^m when iterating). Furthermore, if we have at most $2k$ iterated parts, the iterations necessarily lead to words where the a^m and b^m parts no longer have all the same exponent. Consequently, L' and therefore also L are not k-MCFLs. Since this holds for any k, L is not an MCFL.

References

[Kallmeyer, 2010] Kallmeyer, L. (2010). *Parsing Beyond Context-Free Grammars*. Cognitive Technologies. Springer, Heidelberg.

[Seki et al., 1991] Seki, H., Matsumura, T., Fujii, M., and Kasami, T. (1991). On multiple context-free grammars. *Theoretical Computer Science*, 88(2):191–229.