Mildly Context-Sensitive Grammar Formalisms:

Formal Properties of Tree Adjoining Languages

Laura Kallmeyer
Heinrich-Heine-Universität Düsseldorf

Sommersemester 2011

Overview

1. Closure properties
2. Pumping lemma

Closure properties (1)

One of the reasons why the TAG formalism is appealing from a formal point of view is the fact that it has nice closure properties [Vijay-Shanker and Joshi, 1985, Vijay-Shanker, 1987].

Proposition 1 *TALs are closed under union.*

This can be easily shown as follows: Assume the two sets of non-terminals to be disjoint. Then build a large TAG putting the initial and auxiliary trees from the two grammars together.

Closure properties (2)

Proposition 2 *TALs are closed under concatenation.*

In order to show this, assume again the sets of non-terminals to be disjoint. Then

- build the unions of the initial and auxiliary trees,
- introduce a new start symbol S and
- add one initial tree with root label S and two daughters labeled with the start symbols of the original grammars.

Closure properties (3)

Proposition 3 *TALs are closed under Kleene closure.*

The idea of the proof is as follows: We add an initial tree with the empty word and an auxiliary tree that can be adjoined to the roots of initial trees with the start symbol and that has a new leaf with the start symbol.

Closure properties (4)

Proposition 4 *TALs are closed under substitution.*

In order to obtain the TAG that yields the language after substitution, we replace all terminals by start symbols of the corresponding TAGs.

As a corollary one obtains:

Proposition 5 *TALs are closed under arbitrary homomorphisms.*

Closure properties (5)

Proposition 6 *TALs are closed under intersection with regular languages.*

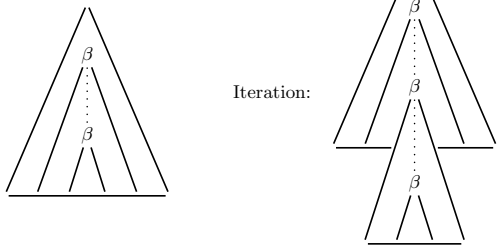
The proof in [Vijay-Shanker, 1987] uses extended push-down automata (EPDA), the automata that recognize TALs. We will introduce EPDAs later. Vijay-Shanker combines such an automaton with the finite state automaton for a regular language in order to construct a new EPDA that recognizes the intersection.

Pumping lemma (1)

- In CFLs, from a certain string length on, two parts of the string can be iterated ("pumped").
- The proof idea is the following: Context-free derivation trees from a certain maximal path length on have the property that a non-terminal occurs twice on this path. Then the part between the two occurrences can be iterated. This means that the strings to the left and right of this part are pumped.

The same kind of iteration is possible in TAG derivation trees since TAG derivation trees are context-free. This leads to a pumping lemma for TALs [Vijay-Shanker, 1987].

Pumping lemma (2)



Pumping lemma (3)

In other words,

- A derived auxiliary tree β' can be repeatedly adjoined into itself.
- Into the lowest β' (low in the sense of the derivation tree) another auxiliary tree β'' derived from β is adjoined.

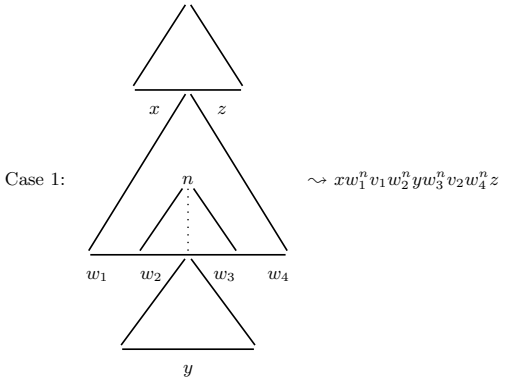
Pumping lemma (4)

What does that mean for the derived tree?

Let n be the node in β' to which β' can be adjoined and to which the final β'' is adjoined as well. There are three cases for the corresponding derived trees before adjoining the final β'':

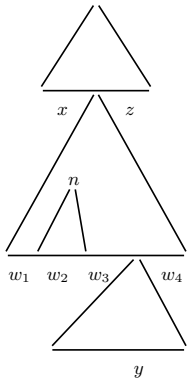
1. n is on the spine (i.e., on the path from the root to the foot node),
2. n is on the left of the spine, or
3. n is on the right of the spine.

Pumping lemma (5)



Pumping lemma (6)

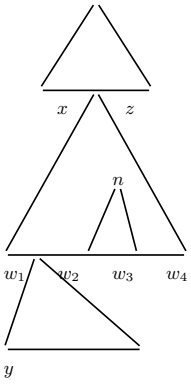
Case 2:



$\rightsquigarrow xw_1^{n+1}v_1w_2v_2w_3(w_2w_4w_3)^nyw_4z$

Pumping lemma (7)

Case 3:



$\rightsquigarrow xw_1y(w_2w_1w_3)^nw_2v_1w_3v_2w_4^{n+1}z$

Pumping lemma (8)

Proposition 7 (Pumping Lemma for TAL) *If L is a TAL, then there is a constant c such that if $w \in L$ and $|w| \geq c$, then there are x, y, z, v_1, v_2, w_1, w_2, w_3, $w_4 \in T^*$ such that*

- $|v_1v_2w_1w_2w_3w_4| \leq c$, $|w_1w_2w_3w_4| \geq 1$, *and*
- *one of the following three cases holds:*
  1. $w = xw_1v_1w_2yw_3v_2w_4z$ *and* $xw_1^nv_1w_2^nyw_3^nv_2w_4^nz$ *is in the string language for all* $n \geq 0$, *or*
  2. $w = xw_1v_1w_2v_2w_3yw_4z$ *and* $xw_1^{n+1}v_1w_2v_2w_3(w_2w_4w_3)^nyw_4z$ *is in the string language for all* $n \geq 0$, *or*
  3. $w = xw_1yw_2v_1w_3v_2w_4z$ *and* $xw_1y(w_2w_1w_3)^nw_2v_1w_3v_2w_4^{n+1}z$ *is in the string language for all* $n \geq 0$.

Pumping lemma (9)

As a corollary, the following weaker pumping lemma holds:

Proposition 8 (Weak Pumping Lemma for TAL)

If L is a TAL, then there is a constant c such that if $w \in L$ and $|w| \geq c$, then there are $x, y, z, v_1, v_2, w_1, w_2, w_3, w_4 \in T^*$ such that

- $|v_1v_2w_1w_2w_3w_4| \leq c$,
- $|w_1w_2w_3w_4| \geq 1$,
- $w = xv_1yv_2z$, *and*
- $xw_1^nv_1w_2^nyw_3^nv_2w_4^nz \in L(G)$ *for all* $n \geq 0$.

In this weaker version, the w_1, w_2, w_3, w_4 need not be substrings of the original word w.

Pumping lemma (10)

A pumping lemma can be used to show that certain languages are not in the class of the string languages satisfying the pumping proposition.

Proposition 9 *The double copy language* $L := \{www \mid w \in \{a, b\}^*\}$ *is not a TAL.*

Pumping lemma (11)

Proof: Assume that L is a TAL.

Then $L' := L \cap a^*b^*a^*b^*a^*b^* = \{a^n b^m a^n b^m a^n b^m \mid n, m \geq 0\}$ is a TAL as well. Assume that L' satisfies the weak pumping lemma with a constant c.

Consider the word $w = a^{c+1}b^{c+1}a^{c+1}b^{c+1}a^{c+1}b^{c+1}$.

None of the w_i, $1 \leq i \leq 4$ from the pumping lemma can contain both as and bs. Furthermore, at least three of them must contain the same letters and be inserted into the three different a^{c+1} respectively or into the three different b^{c+1}. This is a contradiction since then either $|v_1| \geq c + 1$ or $|v_2| \geq c + 1$.

Pumping lemma (12)

Another example of a language that can be shown not to be a TAL, using the pumping lemma, is $L_5 := \{a^n b^n c^n d^n e^n \mid n \geq 0\}$.

References

[Vijay-Shanker, 1987] Vijay-Shanker, K. (1987). *A Study of Tree Adjoining Grammars*. PhD thesis, University of Pennsylvania.

[Vijay-Shanker and Joshi, 1985] Vijay-Shanker, K. and Joshi, A. K. (1985). Some computational properties of Tree Adjoining Grammars. In *Proceedings of the 23rd Annual Meeting of the Association for Computational Linguistics*, pages 82–93.