

## 4 Formal properties of Tree Adjoining Languages

This section is concerned with formal properties of Tree Adjoining Languages (TAL), i.e., of the string sets generated by TAGs.

### 4.1 Closure properties

Vijay-Shanker and Joshi (1985); Vijay-Shanker (1987)

**Proposition 10** *TALs are closed under union.*

This can be easily shown: 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.

**Proposition 11** *TALs are closed under concatenation.*

Easy to show: Assume the sets of non-terminals to be disjoint. Assume both grammars to be with start symbols. Then build the unions of the initial and of the auxiliary trees, introduce 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.

**Proposition 12** *TALs are closed under Kleene closure.*

Proof idea: 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.

**Proposition 13** *TALs are closed under substitution.*

Idea: replace all terminals by start symbols of the corresponding TAGs.

As a corollary one obtains:

**Proposition 14** *TALs are closed under arbitrary homomorphisms.*

**Proposition 15** *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. 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.

TALs are closed under regular substitution (i.e., substitution with regular languages, follows from Prop. 13). With a theorem from Ginsburg (1966) we then get the following corollary:

**Proposition 16** *TALs are closed under inverse homomorphisms.*<sup>6</sup>

$\Rightarrow$  TALs form a substitution closed *Full Abstract Family of Languages (AFL)*.<sup>7</sup>

**Exercise 6** *Show that  $\{a^n b^n c^n a^m b^m c^m \mid n, m \geq 0\}$  is a TAL.*

*Hint: The language  $L_3 = \{a^n b^n c^n \mid n \geq 0\}$  is a TAL.*

**Exercise 7** *Show that  $\{a^i b^j a^i b^j \mid i, j \geq 0\}$  is a TAL.*

*Hint: The copy language is a TAL.*

---

<sup>6</sup>For a homomorphism  $f$ , the image of a language  $L$  under the inverse homomorphism  $f^{-1}$  is  $f^{-1}(L) := \{x \mid f(x) \in L\}$ .

<sup>7</sup>Full AFL = closed under intersection with regular languages, homomorphisms, inverse homomorphisms, union, concatenation and Kleene star.

## 4.2 Pumping lemma

Vijay-Shanker (1987)

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 left and the right of this part are pumped.

The same kind of iteration is possible in TAG derivation trees since TAG derivation trees are context-free (see Fig. 12).

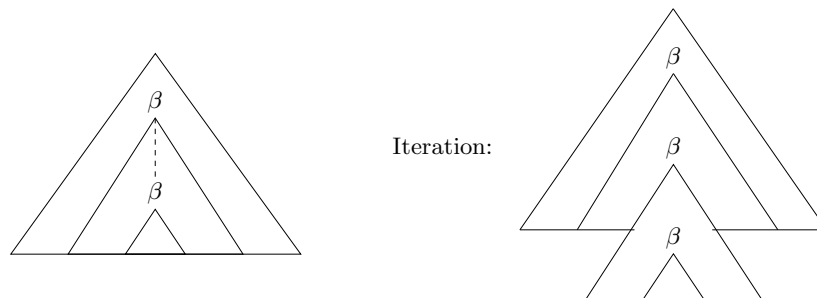


Figure 12: Iteration in a TAG derivation tree

A derived auxiliary tree  $\beta'$  can be repeatedly adjoined into itself. Into the lowest  $\beta'$  (low in the sense of the derivation tree) another auxiliary tree  $\beta''$  derived from  $\beta$  is adjoined. What does that mean for the derived tree? Let  $n$  be the node in  $\beta'$  to which  $\beta'$  can be adjoined and to which the final  $\beta''$  is adjoined as well. There are three cases (see Fig. 13 for the corresponding derived trees before adjoining the final  $\beta''$ ):

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

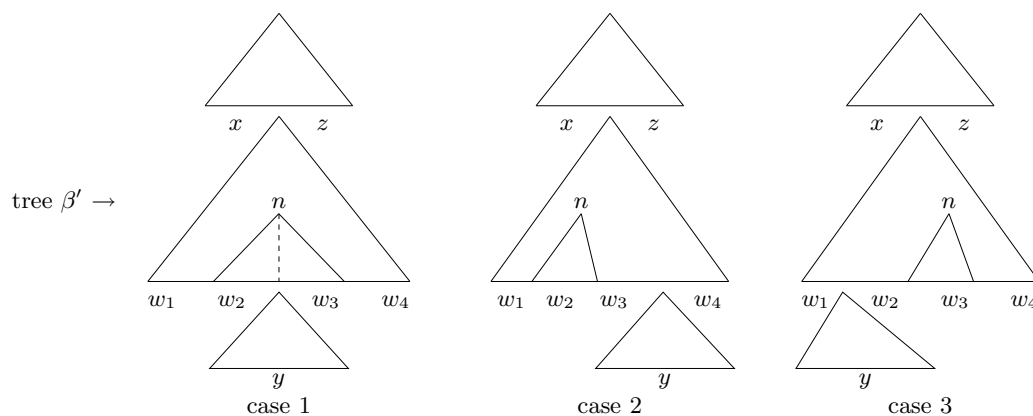


Figure 13: Different cases for the iteration in the derived tree

Assume that the final  $\beta''$  adds strings  $v_1$  and  $v_2$  on the left and the right of the foot node.

Case 1:  $\beta'$  adds the two strings  $w_1w_2$  and  $w_3w_4$  on the two sides of its foot node. Adjunction to  $n$  adds new strings between  $w_1$  and  $w_2$  and between  $w_3$  and  $w_4$  resp. Leads to strings

$$\begin{aligned} & xw_1v_1w_2yw_3v_2w_4z \text{ (no iteration of } \beta') \\ & xw_1w_1v_1w_2w_2yw_3w_3v_2w_4w_4z \text{ (one iteration)} \\ & xw_1w_1w_1v_1w_2w_2w_2yw_3w_3w_3v_2w_4w_4w_4z \\ & xw_1w_1w_1w_1v_1w_2w_2w_2w_2yw_3w_3w_3w_3v_2w_4w_4w_4w_4z \\ & \dots \\ \Rightarrow & xw_1^n v_1 w_2^n y w_3^n v_2 w_4^n z \text{ in the string language for all } n \geq 0. \end{aligned}$$

Case 2:  $\beta'$  adds the two strings  $w_1w_2w_3$  and  $w_4$  on the two sides of its foot node. Adjunction to  $n$  adds new strings between  $w_1$  and  $w_2$  and between  $w_2$  and  $w_3$  resp. Leads to strings

$$\begin{aligned} & xw_1v_1w_2v_2w_3yw_4z \text{ (no iteration)} \\ & xw_1w_1v_1w_2v_2w_3w_2w_4w_3yw_4z \text{ (one iteration of } \beta') \\ & xw_1w_1w_1v_1w_2v_2w_3w_2w_4w_3w_2w_4w_3yw_4z \\ & xw_1w_1w_1w_1v_1w_2v_2w_3w_2w_4w_3w_2w_4w_3w_2w_4w_3yw_4z \\ & \dots \\ \Rightarrow & xw_1^{n+1} v_1 w_2 v_2 w_3 (w_2 w_4 w_3)^n y w_4 z \text{ in the string language for all } n \geq 0. \end{aligned}$$

Case 3:  $\beta'$  adds the two strings  $w_1$  and  $w_2w_3w_4$  on the two sides of its foot node. Adjunction to  $n$  adds new strings between  $w_2$  and  $w_3$  and between  $w_3$  and  $w_4$  resp. Leads to strings

$$\begin{aligned} & xw_1yw_2v_1w_3v_2w_4z \text{ (no iteration)} \\ & xw_1yw_2w_1w_3w_2v_1w_3v_2w_4w_4z \text{ (one iteration)} \\ & xw_1yw_2w_1w_3w_2w_1w_3w_2v_1w_3v_2w_4w_4w_4z \\ & xw_1yw_2w_1w_3w_2w_1w_3w_2w_1w_3w_2v_1w_3v_2w_4w_4w_4w_4z \\ & \dots \\ \Rightarrow & xw_1y(w_2w_1w_3)^n w_2v_1w_3v_2w_4^{n+1}z \text{ in the string language for all } n \geq 0. \end{aligned}$$

**Proposition 17 (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:<sup>8</sup>
  1.  $w = xw_1v_1w_2yw_3v_2w_4z$  and  $xw_1^n v_1 w_2^n y w_3^n v_2 w_4^n z$  is in the string language for all  $n \geq 0$ , or
  2.  $w = xw_1v_1w_2v_2w_3yw_4z$  and  $xw_1^{n+1} v_1 w_2 v_2 w_3 (w_2 w_4 w_3)^n y w_4 z$  is in the string language for all  $n \geq 0$ , or
  3.  $w = xw_1yw_2v_1w_3v_2w_4z$  and  $xw_1y(w_2w_1w_3)^n w_2v_1w_3v_2w_4^{n+1}z$  is in the string language for all  $n \geq 0$ .

Proof: Let  $G$  be a TAG and let  $m$  be the maximum length of the yield of trees in the tree language of  $G$  where there is no  $\beta$  occurring twice on the same path in the derivation tree.  $c_1 := m + 1$ .

For any word with length  $\geq c$ , a  $\beta'$  as above can be iterated and one of the three cases above holds.

If inside  $\beta'$  and below the foot node of  $\beta'$  there are no other derived auxiliary trees adjoined into themselves (and one can always choose such a  $\beta'$ ), then there is a constant  $c_2$  such that  $|v_1v_2w_1w_2w_3w_4| \leq c_2$  in all three cases.

Furthermore, since the length of the word is  $> m$ ,  $\beta'$  can be chosen such that at least one of the four iterated parts  $w_1, w_2, w_3, w_4$  is not  $\epsilon$ .

---

<sup>8</sup>According to Vijay-Shanker (1987) the second and third case can be reduced to the first. But if this is true, it is at least not at all obvious: in the second and third case the parts that are iterated are not present in the original word  $x$ : in case 2,  $w_2w_4w_3$  is no substring of  $x$  and in case 3,  $w_2w_1w_3$  is no substring of  $x$ . This is why I prefer to stay with the three cases. It is true however that cases 2 and 3 are somehow weaker than case 1, i.e., a language not satisfying case 1 will probably not satisfy cases 2 or 3 either.

Let  $c := \max\{c_1, c_2\}$ . Then the pumping lemma holds for  $G$  with the constant  $c$ .

□

As a corollary, the following weaker pumping lemma holds:

**Lemma 1 (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*

- $|v_1 v_2 w_1 w_2 w_3 w_4| \leq c$ ,
- $|w_1 w_2 w_3 w_4| \geq 1$ ,
- $x = x v_1 y v_2 z$ , and
- $x w_1^n v_1 w_2^n y w_3^n v_2 w_4^n z \in L(G)$  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  $w$ .

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

**Lemma 2** *The double copy language  $L := \{www \mid w \in \{a, b\}^*\}$  is no TAL.*

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  $a$ 's and  $b$ 's. 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  $b^{c+1}$ . Contradiction since then either  $|v_1| \geq c + 1$  or  $|v_2| \geq c + 1$ .

□

**Exercise 8** *Show that  $L := \{a^{2^n} \mid n \geq 0\}$  is no TAL using the (weak) pumping lemma.*

**Exercise 9**  $L_4 := \{a^n b^n c^n d^n \mid n \geq 0\}$ ,  $L_5 := \{a^n b^n c^n d^n e^n \mid n \geq 0\}$

1. Give a TAG generating  $L_4$ .
2. Show that  $L_5$  is not a TAL using the weak pumping lemma.  
Hint: Consider the word  $w = a^{c+1} b^{c+1} c^{c+1} d^{c+1} e^{c+1}$  with  $c$  being the constant from the pumping lemma.

### 4.3 TAL and the Chomsky hierarchy

We have already seen that  $CFL \stackrel{\subset}{\neq} TAL$ .

In this section we will show that TAL is a true subset of indexed languages that are a true subset of context-sensitive languages.

Indexed grammars were introduced by Aho (1968). An indexed grammar looks like a CFG except that the nonterminals are equipped with stacks of indices.

**Definition 11 (Indexed grammar)** *An indexed grammar is a tuple  $\langle N, T, I, P, S \rangle$  where*

- $N, T$  and  $I$  are pairwise disjoint alphabets, the nonterminals, terminals and indices,
- $P$  is a finite set of productions that are either of the form

- $A \rightarrow \alpha$  or
- $A \rightarrow Bf$  or
- $Af \rightarrow \alpha$

with  $A, B \in N, f \in I, \alpha \in (N \cup T)^*$ ,

- $S \in N$  is the start symbol.

In a derived sentential form  $x$ , nonterminals can be equipped with stacks of indices, i.e.,  $x \in (NI^* \cup T)^*$ . The first kind of production works like context-free productions while copying the stack of  $A$  to all nonterminals in  $\alpha$ . The second kind of production adds a symbol to the stack of  $A$  while replacing  $A$  with  $B$ . The third kind of production deletes a symbol  $f$  from the stack of  $A$  and then works like the first kind of production.

Example: Indexed grammar for  $\{a^{2^n} \mid n \geq 0\}$  with  $N := \{S, A, B\}, I := \{f, g\}, T := \{a\}$  and productions  $P := \{S \rightarrow \epsilon, S \rightarrow Ag, A \rightarrow Af, A \rightarrow B, Bf \rightarrow BB, Bg \rightarrow aa\}$ .

Derivation for  $a^{2^4} = a^{16}$ :

$S \Rightarrow Ag$	production $S \rightarrow Ag$
$\Rightarrow Afg$	production $A \rightarrow Af$
$\xRightarrow{*} Afffg$	
$\Rightarrow Bfffg$	production $A \rightarrow B$
$\Rightarrow BffgBffg$	production $Bf \rightarrow BB$
$\xRightarrow{*} BfgBfgBfgBfg$	
$\xRightarrow{*} BgBgBgBgBgBgBgBg$	
$\xRightarrow{*} aaaaaaaaaaaaaaaaaa$	production $Bg \rightarrow aa$

An indexed grammar is called a *linear indexed grammar (LIG)* (Gazdar 1988; Vijay-Shanker 1987) if in a production  $A \rightarrow \alpha$  or  $Af \rightarrow \alpha$  the stack of  $A$  is copied only to one nonterminal in  $\alpha$ .

We write the productions in a LIG as follows:

- $A[\dots] \rightarrow X_1 \dots X_i[\dots] \dots X_n$  with  $X_j \in N \cup T$  for  $j \neq i, X_i \in N$ .
- $A[\dots] \rightarrow B[f \dots]$
- $A[f \dots] \rightarrow X_1 \dots X_i[\dots] \dots X_n$  with  $X_j \in N \cup T$  for  $j \neq i, X_i \in N$ .

For a given TAG, a weakly equivalent LIG can be constructed. Example: LIG for the copy language:

$\langle S, \alpha \rangle \rightarrow \epsilon$	
$\langle S, \alpha \rangle \rightarrow \langle S_1, \beta_a \rangle [\langle \alpha, 0 \rangle]$	$\langle S, \alpha \rangle \rightarrow \langle S_1, \beta_b \rangle [\langle \alpha, 0 \rangle]$
$\langle S_1, \beta_a \rangle [\dots] \rightarrow a \langle S_2, \beta_a \rangle [\dots]$	$\langle S_1, \beta_b \rangle [\dots] \rightarrow b \langle S_2, \beta_b \rangle [\dots]$
$\langle S_2, \beta_a \rangle [\dots] \rightarrow \langle S_3, \beta_a \rangle [\dots] a$	$\langle S_2, \beta_b \rangle [\dots] \rightarrow \langle S_3, \beta_b \rangle [\dots] b$
$\langle S_2, \beta_a \rangle [\dots] \rightarrow \langle S_1, \beta_a \rangle [\langle \beta_a, 2 \rangle \dots]$	$\langle S_2, \beta_a \rangle [\dots] \rightarrow \langle S_1, \beta_b \rangle [\langle \beta_a, 2 \rangle \dots]$
$\langle S_2, \beta_b \rangle [\dots] \rightarrow \langle S_1, \beta_a \rangle [\langle \beta_b, 2 \rangle \dots]$	$\langle S_2, \beta_b \rangle [\dots] \rightarrow \langle S_1, \beta_b \rangle [\langle \beta_b, 2 \rangle \dots]$
$\langle S_3, \beta_a \rangle [\langle \alpha, 0 \rangle \dots] \rightarrow \langle S, \alpha \rangle [\dots]$	$\langle S_3, \beta_b \rangle [\langle \alpha, 0 \rangle \dots] \rightarrow \langle S, \alpha \rangle [\dots]$
$\langle S_3, \beta_a \rangle [\langle \beta_a, 2 \rangle \dots] \rightarrow \langle S_2, \beta_a \rangle [\dots]$	$\langle S_3, \beta_b \rangle [\langle \beta_a, 2 \rangle \dots] \rightarrow \langle S_2, \beta_a \rangle [\dots]$
$\langle S_3, \beta_a \rangle [\langle \beta_b, 2 \rangle \dots] \rightarrow \langle S_2, \beta_b \rangle [\dots]$	$\langle S_3, \beta_b \rangle [\langle \beta_b, 2 \rangle \dots] \rightarrow \langle S_2, \beta_b \rangle [\dots]$

**Proposition 18** For each TAG there exists a weakly equivalent LIG.

Sketch of the construction: Assume that the TAG is without substitution nodes. The nonterminals are pairs  $\langle \gamma, p \uparrow \rangle$  and  $\langle \gamma, p \downarrow \rangle$  with  $\gamma$  being an elementary tree and  $p$  a node position in  $\gamma$ . The indices are all  $\bar{X}$  where  $X$  is a nonterminal.

Productions:

1. For each internal node or foot node without OA constraint at position  $p$  in a tree  $\gamma$ : There is a production

$$\langle \gamma, p \uparrow \rangle \rightarrow \langle \gamma, p \downarrow \rangle$$

2. For each subtree of height 1 of the elementary tree  $\gamma$  with root position  $p$  and daughter positions  $p_1, \dots, p_n$ :

If there is no foot node below the root, then there is a production

$$\langle \gamma, p \downarrow \rangle \rightarrow \langle \gamma, p_1 \uparrow \rangle \dots \langle \gamma, p_n \uparrow \rangle$$

If there is a foot node below the position  $p_i$ , then there is a production

$$\langle \gamma, p \downarrow \rangle[\dots] \rightarrow \langle \gamma, p_1 \uparrow \rangle \dots \langle \gamma, p_i \uparrow \rangle[\dots] \dots \langle \gamma, p_n \uparrow \rangle$$

3. For each node with position  $p$  in  $\gamma$  that allows adjunction of  $\beta$ , there is a production

$$\langle \gamma, p \uparrow \rangle[\dots] \rightarrow \langle \beta, 0 \uparrow \rangle[\overline{\langle \gamma, p \uparrow \rangle} \dots]$$

4. For each node  $\mu$  with position  $p$  in  $\gamma$  that allows adjunction of  $\beta$  with  $f$  being the position of the foot node in  $\beta$ : Then there is a production

$$\langle \beta, f \downarrow \rangle[\overline{\langle \gamma, p \uparrow \rangle} \dots] \rightarrow \langle \gamma, p \downarrow \rangle[\dots]$$

5. For each leaf with position  $p$  in  $\gamma$  and with a terminal label  $a$ , there is a production

$$\langle \gamma, p \uparrow \rangle \rightarrow a$$

The other direction, i.e.,  $LIL \subseteq TAL$  can be shown constructing an epda for a given LIL (see Vijay-Shanker 1987).

This gives us the following:

$$CFL \stackrel{\subset}{\neq} TAL = LIL \stackrel{\subset}{\neq} IL \stackrel{\subset}{\neq} CSL$$

( $TAL \stackrel{\subset}{\neq} IL$  can be seen with the language  $\{a^{2^n} \mid n \geq 0\}$  that is no TAL but an IL.)

**Exercise 10** Give an indexed grammar for  $L_A = \{a^n b^n c^n d^n \mid n \geq 0\}$ .