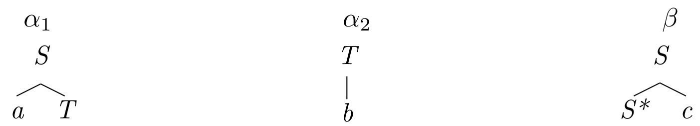
Parsing Beyond CFG
Homework 4: TAG Parsing, Abgabe 22.05.2013

Laura Kallmeyer, Patrick Hommers

SS 2013, Heinrich-Heine-Universität Düsseldorf

Question 1 (TAG CYK parsing)

Consider the TAG consisting of the two trees α and β:



Give the trace of the CYK parse (the version from the course slides) of $w = abc$, i.e., a list of all items that get generated. Explain for each item, by which operation it is obtained and from which antecedent items.

Solution:

	Item	Rule
1.	$[\alpha_1, 1_\top, 0, -, -, 1]$	lex-scan (a)
2.	$[\alpha_2, 1_\top, 1, -, -, 2]$	lex-scan (b)
3.	$[\beta, 2_\top, 2, -, -, 3]$	lex-scan (c)
4.	$[\beta, 1_\top, 0, 0, 2, 2]$	foot-predict
5.	$[\alpha_2, \epsilon_\bot, 1, -, -, 2]$	move-unary from 2.
6.	$[\alpha_2, \epsilon_\top, 1, -, -, 2]$	null-adjoin from 5.
7.	$[\alpha_1, 2_\top, 1, -, -, 2]$	substitute 6.
8.	$[\alpha_1, \epsilon_\bot, 0, -, -, 2]$	move binary from 1. and 7.
9.	$[\alpha_1, \epsilon_\top, 0, -, -, 2]$	null-adjoin from 8.
10.	$[\beta, \epsilon_\bot, 0, 0, 2, 3]$	move-binary from 3. and 4.
11.	$[\beta, \epsilon_\top, 0, 0, 2, 3]$	null-adjoin from 10.
12.	$[\alpha_1, \epsilon_\top, 0, -, -, 3]$	adjoin 11. in 8.

Question 2 *Assume the following definitions:*

In a tree γ, a node n_1 with address p_1 linearly precedes *a node n_2 with address p_2 (notation $n_1 \prec n_2$) iff there are prefixes pi and pj of p_1 and p_2 respectively ($p \in \mathbb{N}^*, i, j \in \mathbb{N}$) such that $i < j$.*

Let us call an auxiliary tree β a left auxiliary tree *iff there is no node in β that is linearly preceded by the foot node.*

Now define a left-auxiliary TAG *as a TAG where all auxiliary trees are left auxiliary trees.*

Obviously, in a left-auxiliary TAG*, the yield of an auxiliary tree comprises only one substring of the input string. (Not two, as is the case in general in TAG.) This makes parsing less complex.*

Modify the Earley algorithm under the assumption that we have a left-auxiliary TAG*. (Give the modified deduction rules.)*

Solution:

In our items we have only three indices, i, j, k, where i and k delimit the total span of the relevant part of the tree (these were i and l in the original algorithm). j gives the start position of the part below the foot node for left auxiliary trees.

The *Scan* and *Predict* rules remain more or less the same except for the reduced number of indices:

$$\textit{ScanTerm} \quad \frac{[\alpha, p, la, i, j, k, 0]}{[\alpha, p, ra, i, j, k+1, 0]} \quad \alpha(p) = w_{k+1}$$

Scan-ε $\dfrac{[\alpha, p, la, i, j, k, 0]}{[\alpha, p, ra, i, j, k, 0]}$ $\alpha(p) = \varepsilon$

PredictAdjoinable $\dfrac{[\alpha, p, la, i, j, k, 0]}{[\beta, 0, la, k, -, k, 0]}$ $\beta \in f_{SA}(\alpha, p)$

PredictNoAdj $\dfrac{[\alpha, p, la, i, j, k, 0]}{[\alpha, p, lb, k, -, k, 0]}$ $f_{OA}(\alpha, p) = 0$

PredictAdjoined $\dfrac{[\beta, p, lb, k, -, k, 0]}{[\delta, p', lb, k, -, k, 0]}$ $p = foot(\beta), \beta \in f_{SA}(\delta, p')$

For the *Complete* rule, we obtain the following:

CompleteFoot

$$\frac{[\alpha, p, rb, i, j, k, 1][\beta, p', lb, i, -, i, 0]}{[\beta, p', rb, i, i, k, 0]} \quad p' = foot(\beta), \beta \in f_{SA}(\alpha, p)$$

CompleteNode (remains the same)

$$\frac{[\beta, p, rb, i, j, k, sat?][\beta, p, la, h, -, i, 0]}{[\beta, p, ra, h, j, k, 0]} \quad \beta(p) \in N$$

For the *Adjoin* rule, we obtain the following:

Adjoin

$$\frac{[\beta, 0, ra, i, j, k, 0][\alpha, p, rb, j, l, k, 0]}{[\alpha, p, rb, i, l, k, 1]} \quad \beta \in f_{SA}(\alpha, p)$$

The *Move* rules and also the *Initialize* rule and the goal item remain the same, except for the reduced number of indices.