Schwach kontextsensitive Grammatikformalismen
Zwischenklausur

17.5.2011

Laura Kallmeyer

SS 2011, Heinrich-Heine-Universität Düsseldorf

Dauer: 90 Minuten.

Hilfmittel: Sämtliche Unterrichtsmaterialien und Notizen in nicht-elektronischer Form.

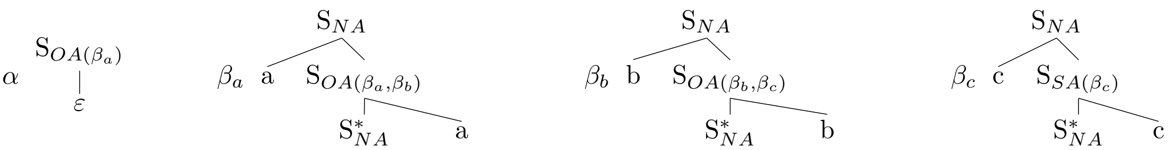
Aufgabe 1

Geben Sie für die Sprache

$$L = \{a^n b^m c^k a^n b^m c^k \mid n, m, k > 0\}$$

eine TAG an, die diese Sprache generiert.

Lösung:



10

Aufgabe 2

Zeigen Sie, dass die Sprache

$$L = \{w \mid w \in \{a, b, c, d, e\}^*, |w|_a = |w|_b = |w|_c = |w|_d = |w|_e\}$$

keine Tree Adjoining Language ist.

Hinweis: Nach Schnittbildung mit einer geeigneten regulären Sprache ergibt sich eine Sprache, von der wir schon gezeigt haben, dass sie das Pumping Lemma für TAL nicht erfüllt.

Lösung:

Annahme: L ist eine TAL. Dann muss auch

$$L' = L \cap a^*b^*c^*d^*e^* = \{a^n b^n c^n d^n e^n \mid n \geq 0\}$$

eine TAL sein und das Pumping Lemma mit einer Konstante k erfüllen. Wir haben schon gezeigt, dass dies nicht der Fall ist.

Somit sind L' und damit auch L keine TALs.

7

Aufgabe 3 *Geben Sie für folgende Sprachen das Parikh-Bild an.*

1. $L_1 = \{w \mid w \in \{a, b, c\}^*, |w|_a = |w|_b = |w|_c\}$
2. $L_2 = \{a^n b^m c^k a^n b^m c^k \mid n, m, k > 0\}$
3. $L_2 = \{a^n b^m a^n b^m \mid n > m > 0\}$

Lösung:

1. $\{n\langle 1,1,1\rangle \mid n \geq 0\}$ 3
2. $\{n\langle 2,0,0\rangle + m\langle 0,2,0\rangle + k\langle 0,0,2\rangle \mid n,m,k > 0\}$ 6
3. $\{n\langle 2,0\rangle + m\langle 0,2\rangle \mid n > m > 0\}$ 5

Aufgabe 4 *Betrachten Sie den folgenden EPDA* $M = \langle\{q_0,q_1,q_2\},\{a,b,c,d\},\{A,B,C,\#\},\delta,q_0,\emptyset,\#\rangle$ *mit Übergangsfunktion* δ*:*

$\delta(q_0,a,\#) = \{(q_0,\varepsilon,\#A,\varepsilon)\}$	$\delta(q_0,a,A) = \{(q_0,\varepsilon,AA,\varepsilon)\}$	$\delta(q_0,a,B) = \{(q_0,\varepsilon,BA,\varepsilon)\}$
$\delta(q_0,b,\#) = \{(q_0,\varepsilon,\#B,\varepsilon)\}$	$\delta(q_0,b,A) = \{(q_0,\varepsilon,AB,\varepsilon)\}$	$\delta(q_0,b,B) = \{(q_0,\varepsilon,BB,\varepsilon)\}$
$\delta(q_0,c,\#) = \{(q_0,\ddagger C,\#,\varepsilon)\}$	$\delta(q_0,c,A) = \{(q_0,\ddagger C,A,\varepsilon)\}$	$\delta(q_0,c,B) = \{(q_0,\ddagger C,B,\varepsilon)\}$
$\delta(q_0,d,\#) = \{(q_1,\varepsilon,\#,\varepsilon)\}$	$\delta(q_0,d,A) = \{(q_1,\varepsilon,A,\varepsilon)\}$	$\delta(q_0,d,B) = \{(q_1,\varepsilon,B,\varepsilon)\}$
$\delta(q_1,a,A) = \{(q_1,\varepsilon,\varepsilon,\varepsilon)\}$	$\delta(q_1,\varepsilon,B) = \{(q_1,\ddagger B,\varepsilon,\varepsilon)\}$	$\delta(q_1,\varepsilon,\#) = \{(q_2,\varepsilon,\varepsilon,\varepsilon)\}$
$\delta(q_2,b,B) = \{(q_2,\varepsilon,\varepsilon,\varepsilon)\}$	$\delta(q_2,c,C) = \{(q_2,\varepsilon,\varepsilon,\varepsilon)\}$	

Akzeptanz mit leerem Stack.

1. *Geben Sie die Konfigurationen (Tupel aus Zustand, Stack, schon verarbeitetem Teil der Eingabe und verbleibender Eingabe) an, die bei Verarbeitung von acabcdaabcc durchlaufen werden.*
2. *Welche Sprache wird von* M *als* $N(M)$ *akzeptiert? Begründen Sie Ihre Antwort, indem Sie erklären, welche Bedeutung die verschiedenen Zustände haben.*

Lösung:

1. Konfigurationen:

Zust.	Stack	erkannte Eingabe	Resteingabe	
q_0	$\ddagger\#$	ε	$acabcdaabcc$	
q_0	$\ddagger\#A$	a	$cabcdaabcc$	
q_0	$\ddagger C\ddagger\#A$	ac	$abcdaabcc$	
q_0	$\ddagger C\ddagger\#AA$	aca	$bcdaabcc$	
q_0	$\ddagger C\ddagger\#AAB$	$acab$	$cdaabcc$	
q_0	$\ddagger C\ddagger C\ddagger\#AAB$	$acabc$	$daabcc$	
q_1	$\ddagger C\ddagger C\ddagger\#AAB$	$acabcd$	$aabcc$	7
q_1	$\ddagger C\ddagger C\ddagger B\ddagger\#AA$	$acabcd$	$aabcc$	
q_1	$\ddagger C\ddagger C\ddagger B\ddagger\#A$	$acabcda$	$abcc$	
q_1	$\ddagger C\ddagger C\ddagger B\ddagger\#$	$acabcdaa$	bcc	
q_2	$\ddagger C\ddagger C\ddagger B$	$acabcdaa$	bcc	
q_2	$\ddagger C\ddagger C$	$acabcdaab$	cc	
q_2	$\ddagger C$	$acabcdaabc$	c	
q_2	ε	$acabcdaabcc$	ε	

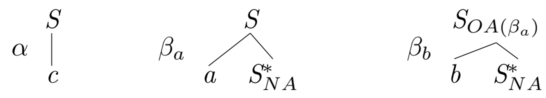
2. In q_0 wird die erste Worthälfte (bis zum d) gelesen, auf dem oberen Stack wird festgehalten, wieviel as und bs gesehen werden und die Anzahl der gesehenen cs wird in den Stacks darunter festgehalten. 2

   In q_1 werden so viel as eingelesen wie es Symbole A auf dem oberen Stck gibt (d.h., genauso viele wie es in der ersten Worthälfte gab). Alle Bs auf dem Stack werden unter den Stack geschoben. q_1 baut den oberen Stack ab (bis zu $\#$), dann wird nach q_2 gewechselt. 2

   In q_2 werden von dem verbleibenden Stack alle Bs und Cs bei Auftreten von entsprechenden Eingabesymbolen abgebaut. D.h., es muss zunächst so viel bs geben wie in der ersten Worthälfte und anschließend so viele cs wie in der ersten Worthälfte. 2

   $N(M) = \{w_1dw_2 \mid w_1 \in \{a,b,c\}^*, w_2 \in a^*b^*c^*, |w_1|_a = |w_2|_a, |w_1|_b = |w_2|_b, |w_1|_c = |w_2|_c\}$. 4

Aufgabe 5 *Betrachten Sie die TAG mit $N = \{S\}, T = \{a, b, c\}$, Startsymbol S und folgenden Elementarbäumen:*



($OA(\beta_a)$ bedeutet obligatorische Adjunktion von β_a.)

Betrachten Sie das Eingabewort $w = abc$.

1. *Vervollständigen Sie die folgende Liste von für diese Eingabe vom CYK Parser erzeugten Items. Dabei sollen nur die erfolgreichen Items aufgelistet werden. Bei jedem Item soll angegeben werden, mit welcher Regel es aus welchen Antezedensitems hergeleitet wurde (die Items werden über die Nummern in der ersten Spalte identifiziert).*

Id	*Item*	*Operation*	*Antezedens-Items*
1	$[\alpha, 1_\top, 2, --, --, 3]$	*Lex-scan*	–
2	$[\beta_a, 1_\top, 0, --, --, 1]$	*Lex-scan*	–
3	$[\beta_a, 2_\top, 1, 1, 3, 3]$	*Foot-predict*	–

2. *Woran erkennt der Parser, dass abc zu der von der TAG generierten Kettensprache gehört?*

Lösung:

1.

Id	Item	Operation	Antezedens-Items
4	$[\beta_b, 1_\top, 1, --, --, 2]$	Lex-scan	–
5	$[\beta_b, 2_\top, 2, 2, 3, 3]$	Foot-predict	–
6	$[\alpha, \varepsilon_\bot, 2, --, --, 3]$	Move-unary	1
7	$[\beta_a, \varepsilon_\bot, 0, 1, 3, 3]$	Move-binary	2,3
8	$[\beta_b, \varepsilon_\bot, 1, 2, 3, 3]$	Move-binary	4,5
9	$[\beta_a, \varepsilon_\top, 0, 1, 3, 3]$	Null-adjoin	7
10	$[\beta_b, \varepsilon_\top, 0, 2, 3, 3]$	Adjoin	8,9
6	$[\alpha, \varepsilon_\top, 0, --, --, 3]$	Adjoin	6,10

10

2. An dem goal item $[\alpha, \varepsilon_\top, 0, --, --, 3]$ in der Chart. 2

Maximale Anzahl Punkte: 60