

Human Language Technology – Praktikum 3
SYNTAX IN DER SPRACHVERARBEITUNG

Abgabe: 2. Mai 2013

TEIL I: FORMALE GRAMMATIKEN, FORMALE SPRACHEN & AUTOMATEN

Hinweise:

- Grammatiken seien definiert als $G=(N, \Sigma, P, S)$ mit N = Nichtterminale, Σ = Alphabet, P = Produktionsregeln, S = Startsymbol
- $\mathbb{N} = \mathbb{N} \setminus \{0\}$
 $\mathbb{N}^+ = \mathbb{N} \cup \{0\}$

1) Gegeben seien die beiden kontextfreien Grammatiken:

$G_1 = (\{S\}, \{a,b\}, \{S \rightarrow ababS, abab\}, S)$ und
 $G_2 = (\{S, T\}, \{a, b\}, \{S \rightarrow aS, S \rightarrow aT, T \rightarrow aT, T \rightarrow abS, T \rightarrow ab\}, S)$

- Geben Sie jeweils zwei Ausdrücke für die Sprachen an, die durch diese Grammatiken erzeugt werden!
- Überführen Sie G_1 in die Chomsky Normalform!
- Transformieren Sie G_2 in eine rechtsreguläre Grammatik!

2) Die Grammatik $G = (N, T, P, S)$ sei gegeben durch:

$N = \{S, X, Y, Z\}$
 $T = \{0, 1, 2\}$
 $P = \{S \rightarrow OSO, S \rightarrow OS1, S \rightarrow 1SO, S \rightarrow 1S1, S \rightarrow XYZ, S \rightarrow 2, XYZ \rightarrow 2\}$

- Geben Sie für das Wort 01201 zwei verschiedene Ableitungen an!
- Geben Sie eine Grammatik G' an, für die gilt: $L(G) = L(G')$ und $|P'| < |P|$.

3) Geben Sie zu den folgenden Sprachen über $\Sigma = \{a, b\}$ die entsprechende kontextfreie Grammatik, die diese Sprache erzeugt an!

- $L_a = \{w \in \Sigma^* \mid \text{in } w \text{ kommt } abab \text{ als Teilwort vor}\}$
- $L_b = \{w \in \Sigma^* \mid \text{jeder zweite Buchstabe in } w \text{ ist ein } a\}$

Formulieren Sie für die folgenden Teilaufgaben zusätzlich (neben der Grammatik) eine kurze umgangssprachliche Beschreibung (wie in a und b)!

- $L_e = \{w \in \Sigma^* \mid a^n b^{2n}, n \in \mathbb{N}\}$
- $L_f = \{w \in \Sigma^* \mid ab^n, n \in \mathbb{N}^+\}$

- 4) Gegeben sei folgende Sprache über dem Eingabealphabet $A = \{c, d\}$:

$$L_a = \{ w \in A^* \mid c^n d^m c^n; n, m \in \mathbb{N} \}$$

Sei $G_a = (\{S, B\}, \{c, d\}, S, R)$. Geben Sie eine möglichst kleine kontextfreie Regelmenge R an, die die oben eingeführte Sprache L_a generiert. Hinweis: R besteht aus fünf Regeln.

- 5) Geben Sie zu folgenden Grammatiken den höchsten Typ nach Chomsky an!

$$G = (N, \Sigma, P, A)$$

$$\Sigma = \{a, b\}$$

$$N = \{A, B\}$$

$$A = \{A\}$$

a)	b)	c)	d)	e)
$P = \{$				
$A \rightarrow B$	$A \rightarrow B$	$A \rightarrow bB$	$A \rightarrow bB$	$A \rightarrow Ba \mid Bb$
$B \rightarrow BB$	$B \rightarrow Ba$	$B \rightarrow aB$	$B \rightarrow Ba$	$Ba \rightarrow aa$
$B \rightarrow a$	$B \rightarrow b$	$B \rightarrow a$	$B \rightarrow a$	$Bb \rightarrow bb$
$\}$	$\}$	$\}$	$\}$	$\}$

- 6) Erstellen Sie einen nicht-deterministischen endlichen Automaten für 5c)
 7) Entwerfen Sie einen Kellerautomaten für 3c)

TEIL II: GRAMMATIKEN IN DER SPRACHVERARBEITUNG

Hinweis:

Mit „Grammatik-Testprogramm“ ist das auf der Lehrstuhl-Website erhältliche Tool gemeint, das es Ihnen ermöglicht, Grammatiken im SRGS ABNF bzw. XML Format mit Hilfe des Spracherkenners der Firma *Loquendo* auszuprobieren. Infos zu SRGS unter <http://www.w3.org/TR/speech-grammar/>.

- 8) Es sei folgende kontextfreie Grammatik gegeben:

$$G = \{N, T, P, S\}$$

$$N = \{N, V, W, S\}$$

$$T = \{mag, liebt, hasst, und, Maria, Josef, Jakob, Elisabeth\}$$

$$P = \left\{ \begin{array}{ll} S \rightarrow N V N W & V \rightarrow mag \\ N \rightarrow Maria & V \rightarrow liebt \\ N \rightarrow Josef & V \rightarrow hasst \\ N \rightarrow Jakob & W \rightarrow und S \\ N \rightarrow Elisabeth & W \rightarrow \varepsilon \end{array} \right\}$$

- a) Transformieren Sie die Grammatik in das SRGS ABNF-Format!
 b) Geben Sie zwei Sätze an, die mit dieser Grammatik erkannt werden können und testen Sie diese mit dem Grammatik-Testprogramm!

- c) Erweitern Sie die ABNF-Grammatik so, dass Sätze der Form „Maria liebt Josef und Jakob“ möglich sind und überführen Sie die Grammatik in das SRGS XML Format! Testen Sie wieder mit Hilfe des Testprogramms!

9) Es sei folgende SRGS-ABNF-Grammatik gegeben:

```
#ABNF 1.0 iso-8859-1;
language de-de;
mode voice;
tag-format <semantics/1.0>;
root $start;

public $start = $A<1->;
private $A = mein | boot | schippert | den | rhein | entlang |
fährt | auf | dem | main | das | rein | gelbe;
```

- a) Was fällt Ihnen bei der gegebenen Grammatik auf, d.h. welche Nachteile sehen Sie?
- b) Verändern Sie die gegebene Grammatik so, dass die folgenden Sätze korrekt erkannt werden und testen Sie wieder mit Hilfe des Grammatik Testprogramms:
- *Mein Boot schippert den Rhein entlang*
 - *Mein Boot fährt auf dem Main*
 - *Das rein gelbe Boot fährt auf dem Rhein entlang*
 - *Das rein gelbe Boot schippert den Main entlang*
- c) Ist es mit der von Ihnen veränderten Grammatik möglich, den Satz „Auf dem Main fährt mein Boot“ zu sagen? Wenn nein, warum nicht? Welche Erkenntnis erlangen Sie?

10) Semantische Rückgabewerte (SISR) erleichtern die spätere Verarbeitung im Programm. Erstellen Sie eine SRGS-ABNF-Grammatik, die die Variablen Abflugs- und Ankunftsort für Sätze der Form

„Ich möchte von <Ort> nach <Ort> fliegen.“

füllt. Folgende Orte sind vorhanden: Köln, Hamburg und Bad Segeberg. Die Bezeichnungen *Kölle*, *Hansestadt Hamburg* und *Segeberg* sollen auf die Originalnamen abgebildet werden.

Hinweis: Variablen werden über folgende Syntax definiert:

```
$regel = terminal {out.var_name="wert"};
```

Über `rules.latest()` kann auf das letzte Ergebnis des Spracherkenners zurückgegriffen werden, über `out` kann das Ergebnis manipuliert werden.

Testen Sie das Ergebnis wieder im Grammatik-Testprogramm!

11) Entwerfen Sie eine ABNF-Grammatik für folgende Sätze:

- *Ich würde heute sehr sehr gerne ins Kino gehen weil dort ein guter Film kommt und ich Lust darauf habe.*
- *Ich würde sehr sehr sehr gerne ins Kino gehen weil ich morgen Geburtstag habe.*
- *Ich möchte heute sehr gerne ins Kino gehen weil ich Lust darauf habe.*
- *Ich möchte morgen gerne ins Kino gehen weil ich morgen Geburtstag habe und Lust darauf habe und dort ein guter Film kommt.*
- *Ich würde gerne ins Kino gehen weil dort ein guter Film kommt.*
- *Ich möchte sehr sehr sehr sehr gerne ins Kino gehen weil ich Lust darauf habe und ich morgen Geburtstag habe.*
- ...

12) Wie muss die folgende Grammatik verändert werden, damit die Determinator-Nomen Kongruenz (hier: Genus) gewährleistet wird? Hinweis: Die Artikel sollen nicht redundant definiert werden.

```
[header omitted]
public $phrase = $NP $VP;
private $NP= $DET $N;
private $VP = $V | ($V $NP);
private $DET = der | die | das;
private $N = Informatiker | Profi | Programm | Quellcodezeilen;
private $V = schreibt | kompiliert
```

Wenn nun zusätzlich Numerus und Kasus betrachtet werden, kommt es zu extrem vielen neuen Regeln. Mit welcher Technik kann das Problem gelöst werden?