Fundamentele Informatica 3

najaar 2016

http://www.liacs.leidenuniv.nl/~vlietrvan1/fi3/

Rudy van Vliet
kamer 124 Snellius, tel. 071-527 5777
rvvliet(at)liacs(dot)nl

college 7, 18 oktober 2016

8. Recursively Enumerable Languages
8.3. More General Grammars
8.4. Context-Sensitive Languages and The Chomsky Hierarchy

Huiswerkopgave 1

Voor 0.4pt

Inleveren: donderdag 20 oktober 2016, 13:45 uur

A slide from lecture 6

Definition 8.10. Unrestricted grammars

An unrestricted grammar is a 4-tuple $G = (V, \Sigma, S, P)$, where V and Σ are disjoint sets of variables and terminals, respectively, S is an element of V called the start symbol, and P is a set of productions of the form

$$\alpha \to \beta$$

where $\alpha, \beta \in (V \cup \Sigma)^*$ and α contains at least one variable.

A slide from lecture 6

Theorem 8.13.
For every unrestricted grammar G, there is a Turing machine T with $L(T) = L(G)$.

Proof.

1. Move past input
2. Simulate derivation in G on the tape of a Turing machine
3. Equal

A slide from lecture 6

Definition 8.16. Context-Sensitive Grammars
A *context-sensitive grammar* (CSG) is an unrestricted grammar in which no production is length-decreasing.
In other words, every production is of the form $\alpha \to \beta$, where $|\beta| \geq |\alpha|$.

A language is a context-sensitive language (CSL) if it can be generated by a context-sensitive grammar.

A slide from lecture 6

Definition 8.18. Linear-Bounded Automata
A *linear-bounded automaton* (LBA) is a 5-tuple $M = (Q, \Sigma, \Gamma, q_0, \delta)$ that is identical to a nondeterministic Turing machine, with the following exception.

There are two extra tape symbols [and], assumed not to be elements of the tape alphabet Γ.
The initial configuration of M corresponding to input x is $q_0[x]$, with the symbol [in the leftmost square and the symbol] in the first square to the right of x.
During its computation, M is not permitted to replace either of these brackets or to move its tape head to the left of the [or to the right of the].

A slide from lecture 6

Theorem 8.19.
If $L \subseteq \Sigma^*$ is a context-sensitive language, then there is a linear-bounded automaton that accepts L.

Proof. . .

A slide from lecture 6

8.4. Context-Sensitive Languages and the Chomsky Hierarchy

reg. languages	FA	reg. grammar	reg. expression
determ. cf. languages	DPDA		
cf. languages	PDA	cf. grammar	
cs. languages	LBA	cs. grammar	
re. languages	TM	unrestr. grammar	

Theorem 8.14.
For every Turing machine T with input alphabet Σ,
there is an unrestricted grammar G
generating the language $L(T) \subseteq \Sigma^*$.

Proof.
1. Generate (every possible) input string for T.
2. Simulate computation of T for this input string as derivation in grammar.
3. If T reaches accept state, reconstruct original input string.

A slide from lecture 3

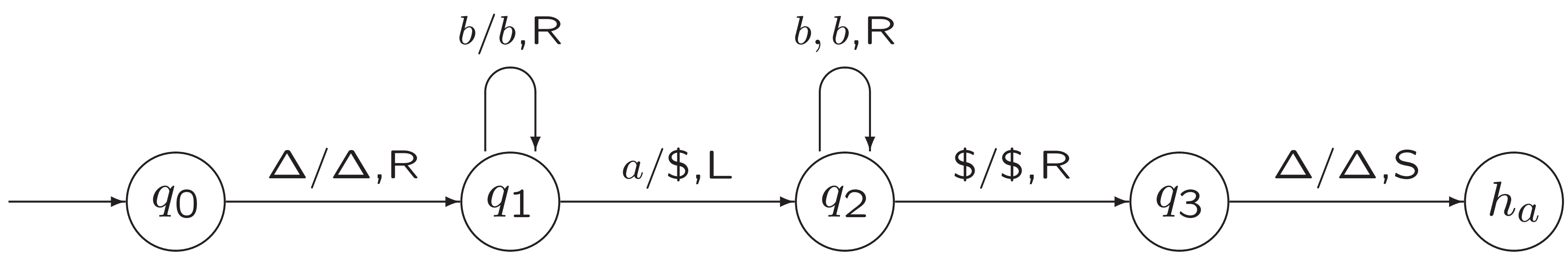
Notation:

description of tape contents: $x\underline{\sigma}y$ or $x\underline{y}$

configuration $xqy = xqy\Delta = xqy\Delta\Delta$

initial configuration corresponding to input x: $q_0\Delta x$

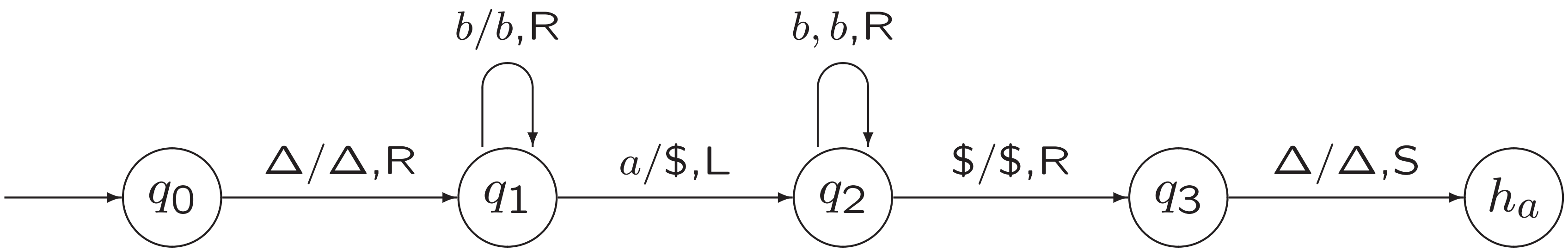
In the third edition of the book, a configuration is denoted as $(q, x\underline{y})$ or $(q, x\underline{\sigma}y)$ instead of xqy or $xq\sigma y$.
In one case, we still use this old notation.



b/b,R
b, b,R
Δ/Δ,R
a/$,L
$/$,R
Δ/Δ,S
q0
q1
q2
q3
ha

Theorem 8.14.
For every Turing machine T with input alphabet Σ,
there is an unrestricted grammar G
generating the language $L(T) \subseteq \Sigma^*$.

Proof.
1. Generate (every possible) input string for T (two copies),
with additional ($\Delta\Delta$)'s and state.
2. Simulate computation of T for this input string as derivation
in grammar (on second copy).
3. If T reaches accept state, reconstruct original input string.


b/b,R
b, b,R
Δ/Δ,R
a/$,L
$/$,R
Δ/Δ,S
q0
q1
q2
q3
ha

3. If T reaches accept state, reconstruct original input string...

Theorem 8.14.
For every Turing machine T with input alphabet Σ, there is an unrestricted grammar G generating the language $L(T) \subseteq \Sigma^*$.

Proof.
1. Generate (every possible) input string for T (two copies), with additional $(\Delta\Delta)$'s and state.
2. Simulate computation of T for this input string as derivation in grammar (on second copy).
3. If T reaches accept state, reconstruct original input string.

Ad 2. Move $\delta(p, a) = (q, b, R)$ of T
yields production $p(\sigma_1 a) \to (\sigma_1 b)q$

Ad 3. Propagate h_a all over the string
$h_a(\sigma_1\sigma_2) \to \sigma_1$, for $\sigma_1 \in \Sigma$
$h_a(\Delta\sigma_2) \to \Lambda$

Exercise 8.27.

Show that if L is any recursively enumerable language, then L can be generated by a grammar in which the left side of every production is a string of one or more variables.

Theorem 8.20. If $L \subseteq \Sigma^*$ is accepted by a linear-bounded automaton $M = (Q, \Sigma, \Gamma, q_0, \delta)$, then there is a context-sensitive grammar G generating $L - \{\Lambda\}$.

Proof. . .

Theorem 8.20. If $L \subseteq \Sigma^*$ is accepted by a linear-bounded automaton $M = (Q, \Sigma, \Gamma, q_0, \delta)$, then there is a context-sensitive grammar G generating $L - \{\Lambda\}$.

Proof. Much like proof of Theorem 8.14, except

- consider $h_a(\sigma_1\sigma_2)$ as a single symbol
- no additional $(\Delta\Delta)$'s needed
- incorporate [and] in leftmost/rightmost symbols of string

Exercise 8.31.

In the proof of Theorem 8.30, the CSG productions corresponding to an LBA move of the form $\delta(p, a) = (q, b, \mathsf{R})$ are given.

Give the productions corresponding to the move $\delta(p, a) = (q, b, \mathsf{L})$ and those corresponding to the move $\delta(p, a) = (q, b, \mathsf{S})$.

Exercise 8.32.

Suppose G is a context-sensitive grammar.
In other words, for every production $\alpha \to \beta$ of G, $|\beta| \geq |\alpha|$.

Show that there is a grammar G', with $L(G) = L(G')$, in which every production is of the form

$$\gamma A \zeta \to \gamma X \zeta$$

where A is a variable and γ, ζ, and X are strings of variables and/or terminals, with X not null.

Exercise 8.32.

$$
\begin{aligned}
S &\rightarrow bA \quad | \quad aAA \\
bA &\rightarrow Ab \\
Ab &\rightarrow ab \\
AA &\rightarrow aa
\end{aligned}
$$

$L(G) = \ldots$

Exercise 8.32.

$$\begin{array}{rcl} S & \to & X_bA \quad | \quad X_aAA \\ X_bA & \to & AX_b \\ AX_b & \to & X_aX_b \\ AA & \to & X_aX_a \\ X_a & \to & a \\ X_b & \to & b \end{array}$$

Exercise 8.32.

$$
\begin{array}{rcl}
S & \to & X_b A \quad | \quad X_a A A \\
X_b A & \to & A A \\
A A & \to & A X_b \\
A X_b & \to & X_a X_b \\
A A & \to & X_a X_a \\
X_a & \to & a \\
X_b & \to & b
\end{array}
$$

$L(G) = \ldots$

Exercise 8.32.

$$
\begin{aligned}
S &\rightarrow X_b A \quad | \quad X_a A A \\
X_b A &\rightarrow X_1 A \\
X_1 A &\rightarrow X_1 X_2 \\
X_1 X_2 &\rightarrow A X_2 \\
A X_2 &\rightarrow A X_b \\
A X_b &\rightarrow X_a X_b \\
A A &\rightarrow X_a X_a \\
X_a &\rightarrow a \\
X_b &\rightarrow b
\end{aligned}
$$

A slide from lecture 6

8.4. Context-Sensitive Languages and the Chomsky Hierarchy

reg. languages	FA	reg. grammar	reg. expression
determ. cf. languages	DPDA		
cf. languages	PDA	cf. grammar	
cs. languages	LBA	cs. grammar	
re. languages	TM	unrestr. grammar	

Chomsky hierarchy

3	reg. languages	FA	reg. grammar	reg. expression
2	cf. languages	PDA	cf. grammar	
1	cs. languages	LBA	cs. grammar	
0	re. languages	TM	unrestr. grammar	

What about recursive languages?

A slide from lecture 5

Theorem 8.2.
Every recursive language is recursively enumerable.

Proof. . .

A slide from lecture 6

Theorem 8.22. Every context-sensitive language L is recursive.

Proof. . .

Chomsky hierarchy

3	reg. languages	FA	reg. grammar	reg. expression
2	cf. languages	PDA	cf. grammar	
1	cs. languages	LBA	cs. grammar	
0	re. languages	TM	unrestr. grammar	

$$\mathcal{S}_3 \subseteq \mathcal{S}_2 \subseteq \mathcal{S}_1 \subseteq \mathcal{R} \subseteq \mathcal{S}_0$$

(modulo Λ)

Huiswerkopgave 2. . .