

Grammars

Bernhard Nebel and Christian Becker-Asano

Overview

- Sipser: the automata/machine approach
- Schoening: the grammar approach
- State links among the two approaches
- See more types of grammars:
 - * Regular grammars
 - * Context-free
 - * Context-sensitive
 - * Grammar
- Based on: Uwe Schoening's "Theoretische Informatik – Kurzgefasst", Spektrum.

Grammars

DEFINITION:

A **grammar** is a 4-tuple (V, Σ, R, S) , where

1. V is a finite set called the **variables**,
2. Σ is a finite set, disjoint from V , called the **terminals**
3. R is a finite set of **rules**, with each rule $u \rightarrow v$ having $u \in (V \cup \Sigma)^+$ and $v \in (V \cup \Sigma)^*$
4. $S \in V$ is the **start symbol**.

Most concepts carry over from CFGs, i.e. derivation, language accepted by grammar, ambiguity, leftmost derivation, ...

Natural language example:

```

<SENTENCE> → <NOUN-PHRASE><VERB-PHRASE>
<NOUN-PHRASE> → <CMPLX-NOUN>|<CMPLX-NOUN><PREP-PHRASE>
<VERB-PHRASE> → <CMPLX-VERB>|<CMPLX-VERB><PREP-PHRASE>
<PREP-PHRASE> → <PREP><CMPLX-NOUN>
<CMPLX-NOUN> → <ARTICLE><NOUN>
<CMPLX-VERB> → <VERB>|<VERB><NOUN-PHRASE>
<ARTICLE> → a | the
<NOUN> → boy | girl | flower
<VERB> → touches | likes | sees
<PREP> → with
  
```

a boy sees

the boy sees a flower

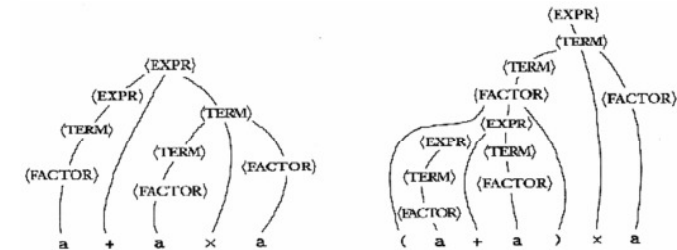
a girl with a flower likes the boy

<SENTENCE>	→ <NOUN-PHRASE><VERB-PHRASE>	<SENTENCE>	⇒ <NOUN-PHRASE><VERB-PHRASE>
<NOUN-PHRASE>	→ <CMLPX-NOUN> <CMLPX-NOUN><PREP-PHRASE>		⇒ <CMLPX-NOUN><VERB-PHRASE>
<VERB-PHRASE>	→ <CMLPX-VERB> <CMLPX-VERB><PREP-PHRASE>		⇒ <ARTICLE><NOUN><VERB-PHRASE>
<PREP-PHRASE>	→ <PREP><CMLPX-NOUN>		⇒ a <NOUN><VERB-PHRASE>
<CMLPX-NOUN>	→ <ARTICLE><NOUN>		⇒ a boy <VERB-PHRASE>
<CMLPX-VERB>	→ <VERB> <VERB><NOUN-PHRASE>		⇒ a boy <CMLPX-VERB>
<ARTICLE>	→ a the		⇒ a boy <VERB>
<NOUN>	→ boy girl flower		⇒ a boy sees
<VERB>	→ touches likes sees		
<PREP>	→ with		

Parsing

$G_3 = (V, \Sigma, R, \langle Expr \rangle)$
 $V = \{ \langle Expr \rangle, \langle Term \rangle, \langle Factor \rangle \}$
 $\Sigma = \{ a, +, \times, (,) \}$
 R is
 $\langle Expr \rangle \rightarrow \langle Expr \rangle + \langle Term \rangle | \langle Term \rangle$
 $\langle Term \rangle \rightarrow \langle Term \rangle \times \langle Factor \rangle | \langle Factor \rangle$
 $\langle Factor \rangle \rightarrow (\langle Expr \rangle) | a$

* Construct meaning (parse tree)



* Parse trees for the strings **a + a x a** and **(a + a) x a**

$S \Rightarrow aSBC$
 $\Rightarrow aaSBCBC$
 $\Rightarrow aaaBCBCBC$
 $\Rightarrow aaaBBCBCC$
 $\Rightarrow aaaBBBCCC$
 $\Rightarrow aaabBBCCC$
 $\Rightarrow aaabbBCCC$
 $\Rightarrow aaabbbCCC$
 $\Rightarrow aaabbbcCC$
 $\Rightarrow aaabbbccC$

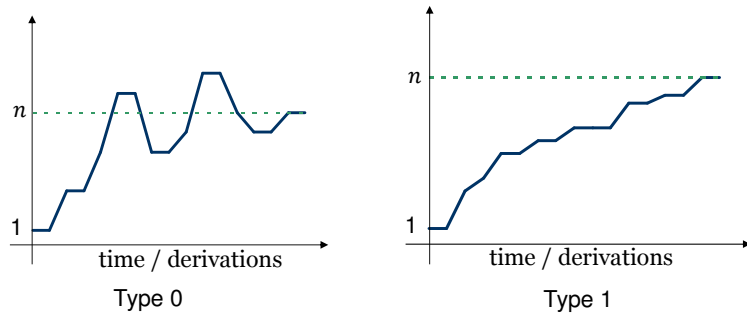
$L(G) = \{ a^n b^n c^n \mid n \geq 1 \} \Rightarrow aaabbbccc$

Chomsky Hierarchy

- Type 0 : every grammar; Turing recognisable language
- Type 1 : context-sensitive
for all rules $u \rightarrow v$ holds : $|u| \leq |v|$
- Type 2 : context free
for all rules $u \rightarrow v$ holds : u is a single variable
- Type 3 : regular
for all rules $u \rightarrow v$ holds : u is a single variable
and v is either a terminal or a terminal followed by a variable
- Exception for
 ϵ : the empty string
Type 1 : $S \rightarrow \epsilon$ and S start symbol that does not appear at the right hand side
Type 2 and 3 : $A \rightarrow \epsilon$ is allowed

Difference Type 0 and Type 1

- Type 0 : every grammar; Turing recognisable language
 - Type 1 : context-sensitive; for all rules $u \rightarrow v$ holds: $|u| \leq |v|$
- n is the length of your string, then:



Acceptance Problem for Grammars of Type 1, 2, or 3

$$A_{L(G)} = \{(G, w) \mid w \in L(G)\}$$

Theorem

$A_{L(G)}$ is decidable if G of Type 1,2 or 3

Proof

For $m, n \in \mathbb{N}$, define

$$T_m^n = \{w \in (V \cup \Sigma)^* \mid |w| \leq n \text{ and } w \text{ can be derived from } S \text{ in at most } m \text{ derivation steps}\}$$

$T_m^n, n \geq 1$ can be defined inductively

$$T_0^n = \{S\}$$

$$T_{m+1}^n = Der_n(T_m^n) \text{ where } Der_n(X) = X \cup \{w \in (V \cup \Sigma)^* \mid |w| \leq n \text{ and } w' \Rightarrow w \text{ for some } w' \in X\}$$

As there are only a finite number of words of length n , it must be that for some m

$$T_m^n = T_{m+1}^n = T_{m+2}^n = \dots$$

and this T_m^n must contain w if $|w| = n$ and $w \in L(G)$

Acceptance Problem for Grammars of Type 1, 2, or 3 (ctd.)

Algorithm

Input $(G, w); |w| = n$

$T := \{S\}$

Repeat

$T_1 := T;$

$T := Der_n(T_1);$

Until $w \in T$ or $T = T_1$

If $w \in T$ then output *yes*; otherwise output *no*

Compute e.g. T_4^4

for the following grammar:

$$S \rightarrow AB$$

$$A \rightarrow AB \mid a$$

$$B \rightarrow b$$

$$T_0^4 = \{S\}$$

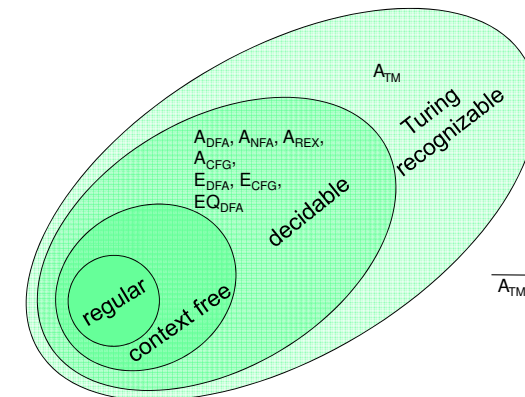
$$T_1^4 = \{S, AB\}$$

$$T_2^4 = \{S, AB, ABB, aB, Ab\}$$

$$T_3^4 = \{S, AB, ABB, aB, Ab, ABBB, aBB, ab, ABb, AbB\}$$

$$T_4^4 = \{S, AB, ABB, aB, Ab, ABBB, aBB, ab, ABb, AbB, aBBB, AbBB, ABbB, ABBb, Abb\}$$

Earlier



The relationship among languages

$Type3 \subset Type2 \subset Type1 \subset Decidable \subset Type0 \subset Languages$

Machines corresponding to languages

- Type 3, regular languages :
 - * Regular grammar, DFA, NFA, regular expression
- Type 2, context-free languages :
 - * Context free grammar, PDA
- Type 1, context-sensitive language
 - * Context sensitive grammar, LBA
- Type 0, Turing recognizable
 - * Grammar, Turing machine

Deterministic versus non-deterministic machines

- NFA and DFA are equivalent
- PDA and DPA are not equivalent
 - * DPA : deterministic subset of PDA
- The LBA-problem: Are LBA and DLBA equivalent?
 - * In short: LBA = DLBA?
 - * Still an open question!
- NTM and DTM are equivalent

Non-determ. automata	Deterministic automata	Equivalent?
NFA	DFA	Yes
PDA	DPDA	No
LBA	DLBA	?
TM	DTM	Yes

Closure properties

closed under ?	\cap	\cup	$-$	\times	*
Regular	yes	yes	yes	yes	yes
Context free	no	yes	no	yes	yes
Context sensitive	yes	yes	yes	yes	yes
Type 0	yes	yes	no	yes	yes

Decidability

<i>decidable ?</i>	<i>A</i>	<i>E</i>	<i>EQ</i>
regular	yes	yes	yes
context free	yes	yes	no
context sensitive	yes	no	no
turing recognizable	no	no	no