Grammars

Andreas Karwath and Malte Helmert

Overview

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

Grammars

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

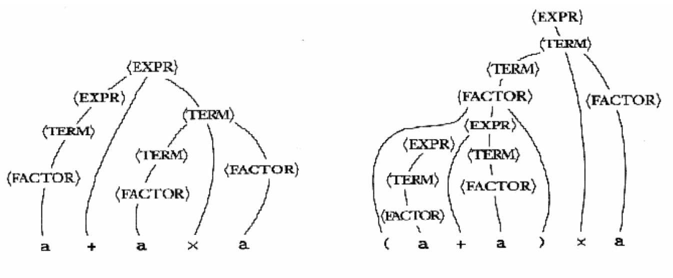
<SENTENCE>	→	<NOUN-PHRASE><VERB-PHRASE>
<NOUN-PHRASE>	→	<CMPLX-NOUN>|<CMPLX-NOUN><PREP-PHRASE>
<VERB-PHRASE>	→	<CMPLX-VERB>|<CMPLX-VERB><PREP-PHRASE>
<PREP-PHRASE>	→	<PREP><CMPLX-NOUN>
<CMPLX-NOUN>	→	<ARTICLE><NOUN>
<CMPLX-VERB>	→	<VERB>|<VERB><NOUN-PHRASE>
<ARTICLE>	→	a | the
<NOUN>	→	boy | girl | flower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

<SENTENCE>	→	<NOUN-PHRASE><VERB-PHRASE>
	→	<CMPLX-NOUN><VERB-PHRASE>
	→	<ARTICLE><NOUN><VERB-PHRASE>
	→	a <NOUN><VERB-PHRASE>
	→	a boy <VERB-PHRASE>
	→	a boy <CMPLX-VERB>
	→	a boy <VERB>
	→	a boy sees

Parsing

$G_3 = (V, \Sigma, R, <Expr>)$
$V = \{<Expr>, <Term>, <Factor>\}$
$\Sigma = \{a, +, \times, (,)\}$
R is
$<Expr> \to <Expr> + <Term> \mid <Term>$
$<Term> \to <Term> \times <Factor> \mid <Factor>$
$<Factor> \to (<Expr>) \mid a$

- Construct meaning (parse tree)



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

$$S \to aSBC$$
$$S \to aBC$$
$$CB \to BC$$
$$aB \to ab$$
$$bB \to bb$$
$$bC \to bc$$
$$cC \to cc$$

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

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

Chomsky Hierarchy

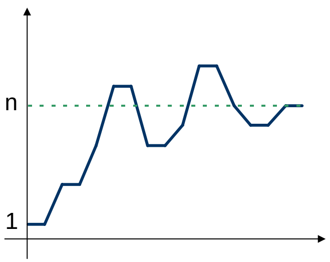
- Type 0 : every grammar; Turing recognisable language
- Type 1 : context-sensitive
  for all rules $u \to v$ holds : $|u| \leq |v|$
- Type 2 : context free
  for all rules $u \to v$ holds : u is a single variable
- Type 3 : regular
  for all rules $u \to v$ holds : u is a single variable
  and v is either a terminal or a terminal followed by a variable
- Exception for

ε: the empty string
Type 1 : $S \to \varepsilon$ and S start symbol that does not appear at the right hand side
Type 2 and 3 : $A \to \varepsilon$ is allowed

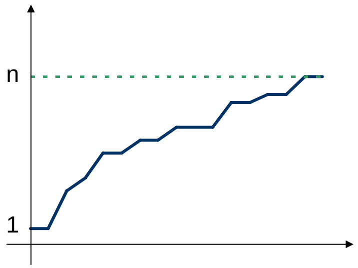
Difference Type 0 andType 11

Type 0 : every grammar; Turing recognisable language

Type 1 : context-sensitive for all rules $u \to v$ holds : $|u| \le |v|$



Type 0



Type 1

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 $m,n \in$ N,define

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

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

$T_0^n = \{S\}$

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

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

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

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 (cont.)

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:

$S \to AB$

$A \to AB \mid a$

$B \to b$

T_0^4 = {S}

T_1^4 = {S,AB}

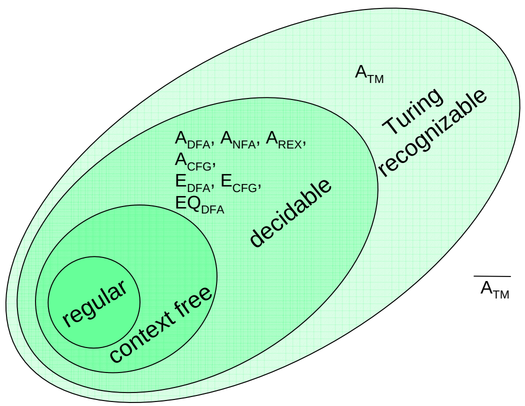
T_2^4 = {S,AB,ABB,aB,Ab}

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}

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

Earlier



The relationship amoung 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
- LBA and DLBA are not equivalent
  - DLBA deterministic subset of LBA
- NTM and DTM are equivalent

Closure properties

closed under ?	$\cap$	$\cup$	$\overline{}$	$\times$	$*$
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

decidable ?	*A*	*E*	*EQ*
regular	yes	yes	yes
context free	yes	yes	no
context sensitive	yes	no	no
turing recognizable	no	no	no