CMSC 141 Automata and Language Theory
Turing Machines

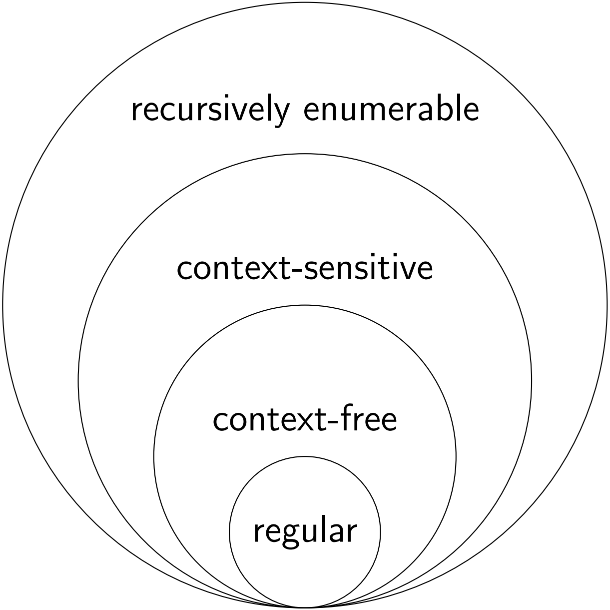
Mark Froilan B. Tandoc

November 7, 2014

Unrestricted Grammars

- Grammars for Turing-recognizable languages are called *Unrestricted Grammars*
- Every rule in the grammar has the form $\alpha \rightarrow \beta$, where α and β can be any sequence of terminals and variables.
- For example: *bXc* $\rightarrow$ *aBcDeF*
- When $|\alpha| \leq |\beta|$ in all the rules, we have a *context-sensitive grammar* (which corresponds to linear bounded automata, a weaker form of Turing machines)

Chomsky Hierarchy of Formal Grammars



Example

- Consider the language with equal number of a's, b's and c's
- $= \{abc, acb, bac, ..., aabbcc, aabcbc, ...\}$
- Exercise: Show that this language cannot be regular nor context-free

Example

Equal number of a's, b's and c's

$$
\begin{array}{lcl}
S & \to & ABCS \text{ (generate } (ABC)^+ \text{ first)} \\
AB & \to & BA \text{ (swapping)} \\
AC & \to & CA \\
BC & \to & CB \\
BA & \to & AB \\
CA & \to & AC \\
CB & \to & BC \\
A & \to & a \text{ (convert to terminals)} \\
B & \to & b \\
C & \to & c
\end{array}
$$

Exercise:

Revise the grammar to generate the language

$\{a^n b^n c^n : n > 0\}$

Example 2

- Consider the language
  $\{a^n : \text{n is a positive power of } 2\}$
- $= \{a^2, a^4, a^8, a^{16}, ...\}$
- Exercise: Show that this language cannot be regular nor context-free

Example 2

From Hopcroft & Ullman

$\{a^n$: n is a positive power of 2$\}$

$$
\begin{array}{lcl}
\mathrm{S} & \rightarrow & <\mathrm{Ca}> \ (<, > \text{ are endmarkers}) \\
\mathrm{Ca} & \rightarrow & \mathrm{aaC} \ (\text{C is a doubler}) \\
\mathrm{C}> & \rightarrow & \mathrm{D}> \ (\text{prepare to double again}) \\
\mathrm{C}> & \rightarrow & \mathrm{E} \ (\text{stop when enough}) \\
\mathrm{aD} & \rightarrow & \mathrm{Da} \\
<\mathrm{D} & \rightarrow & <\mathrm{C} \\
\mathrm{aE} & \rightarrow & \mathrm{Ea} \\
<\mathrm{E} & \rightarrow & \varepsilon
\end{array}
$$

Sample Linear Derivations

$$
\begin{array}{lcl|lcl}
\mathsf{S} & \rightarrow & <\mathsf{Ca}> \text{ ($<, >$ are endmarkers)} & \mathsf{aD} & \rightarrow & \mathsf{Da} \\
\mathsf{Ca} & \rightarrow & \mathsf{aaC} \text{ (C is a doubler)} & <\mathsf{D} & \rightarrow & <\mathsf{C} \\
\mathsf{C}> & \rightarrow & \mathsf{D}> \text{ (prepare to double again)} & \mathsf{aE} & \rightarrow & \mathsf{Ea} \\
\mathsf{C}> & \rightarrow & \mathsf{E} \text{ (stop when enough)} & <\mathsf{E} & \rightarrow & \varepsilon
\end{array}
$$

Derive: aa

$\mathsf{S} \Rightarrow_1 <\mathsf{Ca}> \Rightarrow_2 <\mathsf{aaC}> \Rightarrow_4 <\mathsf{aaE}$

$\Rightarrow_7 <\mathsf{aEa} \Rightarrow_7 <\mathsf{Eaa} \Rightarrow_8 \mathsf{aa}$

Sample Linear Derivations

S	$\rightarrow$	$<$Ca$>$ ($<, >$ are endmarkers)	aD	$\rightarrow$	Da
Ca	$\rightarrow$	aaC (C is a doubler)	$<$D	$\rightarrow$	$<$C
C$>$	$\rightarrow$	D$>$ (prepare to double again)	aE	$\rightarrow$	Ea
C$>$	$\rightarrow$	E (stop when enough)	$<$E	$\rightarrow$	ε

Derive: aaaa

$$\mathsf{S} \Rightarrow_1 \mathsf{<Ca>} \Rightarrow_2 \mathsf{<aaC>} \Rightarrow_3 \mathsf{<aaD>} \Rightarrow_5 \mathsf{<aDa>}$$
$$\Rightarrow_5 \mathsf{<Daa>} \Rightarrow_6 \mathsf{<Caa>} \Rightarrow_2 \mathsf{<aaCa>}$$
$$\Rightarrow_2 \mathsf{<aaaaC>} \Rightarrow_4 \mathsf{<aaaaE} \Rightarrow_7 \mathsf{<aaaEa}$$
$$\Rightarrow_7 \mathsf{<aaEaa} \Rightarrow_7 \mathsf{<aEaaa} \Rightarrow_7 \mathsf{<Eaaaa} \Rightarrow_8 \mathsf{aaaa}$$

References

- Previous slides on CMSC 141
- M. Sipser. Introduction to the Theory of Computation. Thomson, 2007.
- J.E. Hopcroft, R. Motwani and J.D. Ullman. Introduction to Automata Theory, Languages and Computation. 2nd ed, Addison-Wesley, 2001.
- E.A. Albacea. Automata, Formal Languages and Computations, UPLB Foundation, Inc. 2005
- JFLAP, www.jflap.org
- Various online LaTeX and Beamer tutorials