Unrestricted and Context-Sensitive Grammars

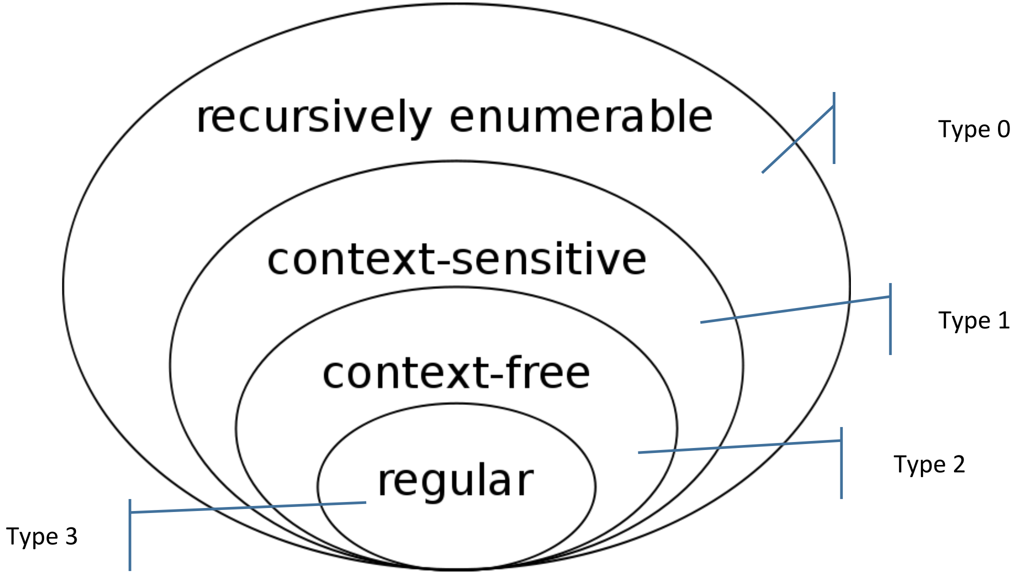
MIEIC, 2nd Year

João M. P. Cardoso
Email: jmpc@acm.org

U.PORTO
FEUP FACULDADE DE ENGENHARIA UNIVERSIDADE DO PORTO

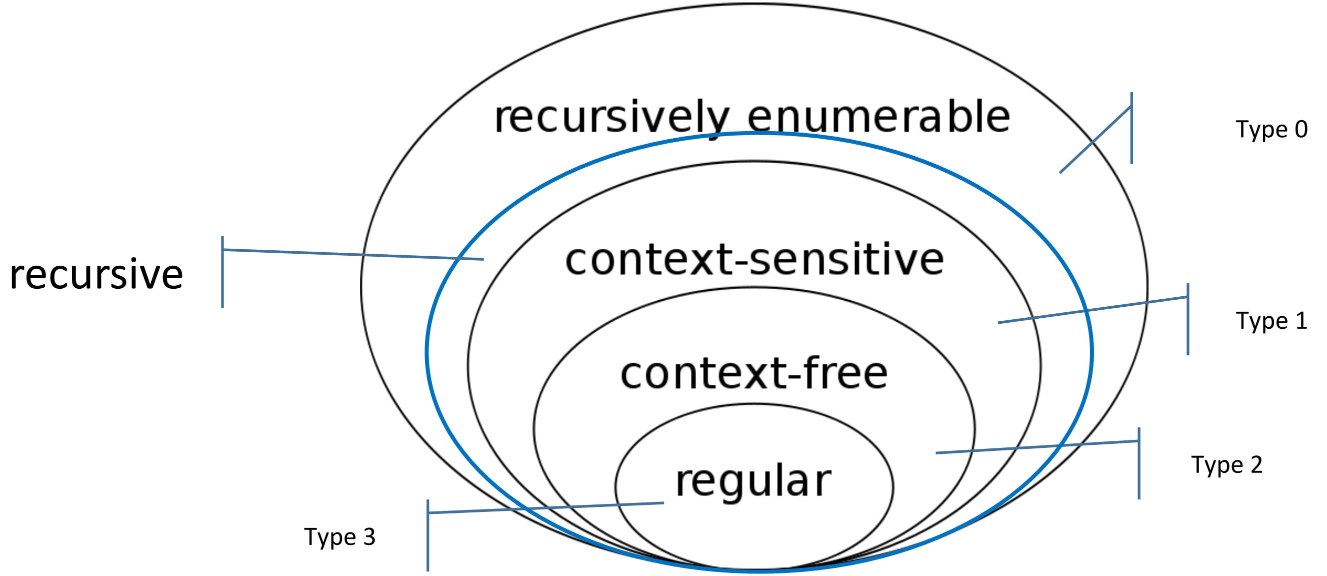
DEI DEPARTAMENTO DE ENGENHARIA INFORMÁTICA

Languages



Set inclusions described by the Chomsky hierarchy

Languages



recursively enumerable
Type 0
recursive
context-sensitive
Type 1
context-free
Type 2
regular
Type 3

Languages: hierarchy

Class	Grammars	Languages	Automaton
Type-0	Unrestricted	Recursively Enumerable (Turing-Recognizable or Turing-Acceptable)	Turing Machine
	Unrestricted	Recursive (Turing- Decidable)	Turing Machine
Type-1	Context-Sensitive	Context-Sensitive	Linear-Bounded
Type-2	Context-Free	Context-Free	Pushdown
Type-3	Regular	Regular	Finite

Recursive Languages

- Recursive languages are also called **decidable**
- **Turing-decidable language** is also a synonym of “recursive language”
- Require that the Turing machine halts in all cases

Recursively Enumerable Languages

- The class of decision problems for which a “yes” answer can be verified by a Turing machine in a finite amount of time
- If the answer is “no”, the machine might never halt
- *Recursively Enumerable Languages* are also called as Turing recognizable languages (or *Turing acceptable languages*)

Unrestricted Grammars

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
- P is a set of productions of the form

  $\alpha \rightarrow \beta$

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

- *Every **recursively enumerable language** can be implemented by an unrestricted grammar*

Example 1

$$L = \{a^{2^k} \mid k \in \mathcal{N}\}$$

- $S \rightarrow LaR$
- $L \rightarrow LD$
- $Da \rightarrow aaD$
- $DR \rightarrow R$
- $L \rightarrow \varepsilon$
- $R \rightarrow \varepsilon$

Sequence of derivations for aaaa:

$S \Rightarrow LaR \Rightarrow LDaR \Rightarrow LaaDR \Rightarrow LaaR \Rightarrow LDaaR \Rightarrow LaaDaR \Rightarrow LaaaaDR \Rightarrow LaaaaR \Rightarrow aaaaR \Rightarrow aaaa$

Context-Sensitive Grammars (CSGs)

Context-Sensitive Grammars (CSG)

- 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 implemented by a context-sensitive grammar
- CSGs can be implemented using a *Linear Bounded Automaton* (LBA)
  - restricted form of Turing machine

https://en.wikipedia.org/wiki/Linear_bounded_automaton

Example 2

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

Example 2

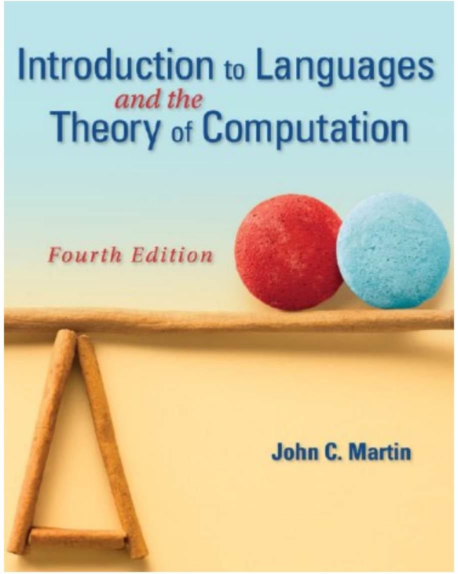
$$L = \{a^n b^n c^n \mid n \geq 1\}$$

- $S \rightarrow SABC \mid ABC$
- $BA \rightarrow AB$
- $CA \rightarrow AC$
- $CB \rightarrow BC$
- $A \rightarrow a$
- $aA \rightarrow aa$
- $aB \rightarrow ab$
- $bB \rightarrow bb$
- $bC \rightarrow bc$
- $cC \rightarrow cc$

Summary

- The Chomsky hierarchy

Type	Languages (Grammars)	Form of Productions in Grammar	Accepting Device
3	Regular	$A \to aB,\ A \to \Lambda$ $(A, B \in V,\ a \in \Sigma)$	Finite automaton
2	Context-free	$A \to \alpha$ $(A \in V,\ \alpha \in (V \cup \Sigma)^*)$	Pushdown automaton
1	Context-sensitive	$\alpha \to \beta$ $(\alpha, \beta \in (V \cup \Sigma)^*,\ \lvert\beta\rvert \geq \lvert\alpha\rvert,$ α contains a variable)	Linear-bounded automaton
0	Recursively enumerable (unrestricted)	$\alpha \to \beta$ $(\alpha, \beta \in (V \cup \Sigma)^*,$ α contains a variable)	Turing machine



Summary

- ***Recursively enumerable languages*** are the ones that can be accepted by Turing machines (also called Turing-acceptable languages)
- ***Recursive Languages*** are those that can be *decided* by Turing machines (also called Turing-decidable languages)

- A language L is ***recursively enumerable*** if there is a TM that accepts L, and L is ***recursive*** if there is a TM that decides L

Credits

- Most material from:
  - John C. Martin***, Introduction to Languages and the Theory of Computation***, McGraw-Hill Higher Education, 4th Ed. (2010).