Course 2
Introduction to Automata Theory (cont'd)

The structure and the content of the lecture is based on http://www.eecs.wsu.edu/~ananth/CptS317/Lectures/index.htm

Excursion: Previous lecture

Languages & Grammars

Or "**words**"

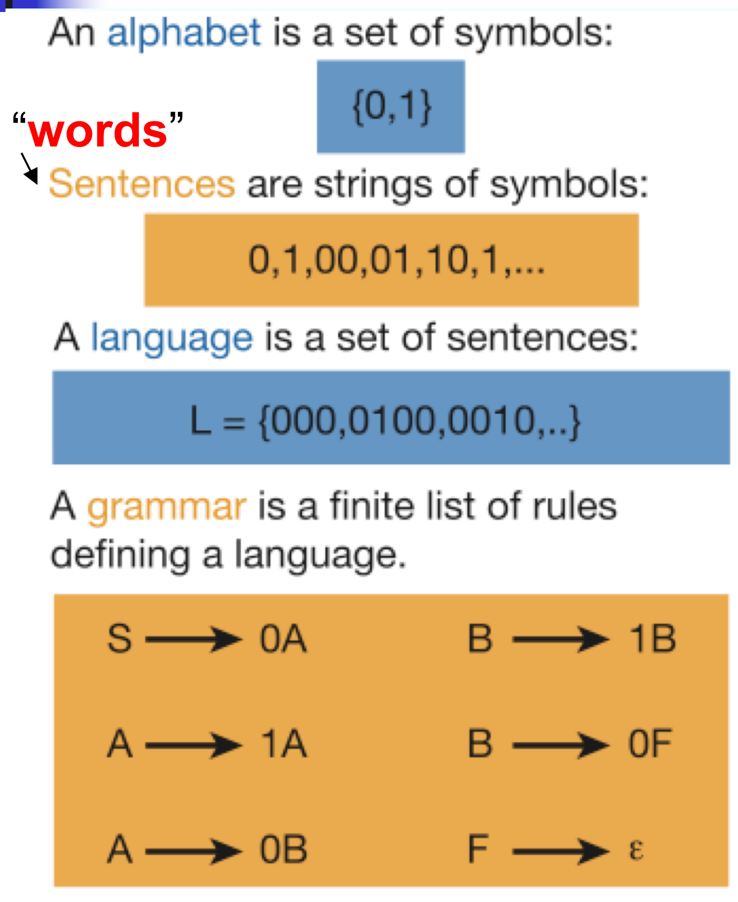
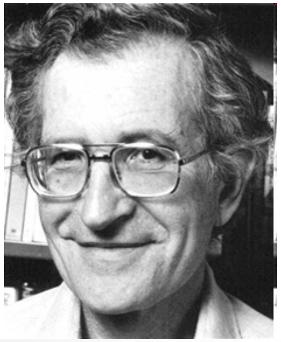


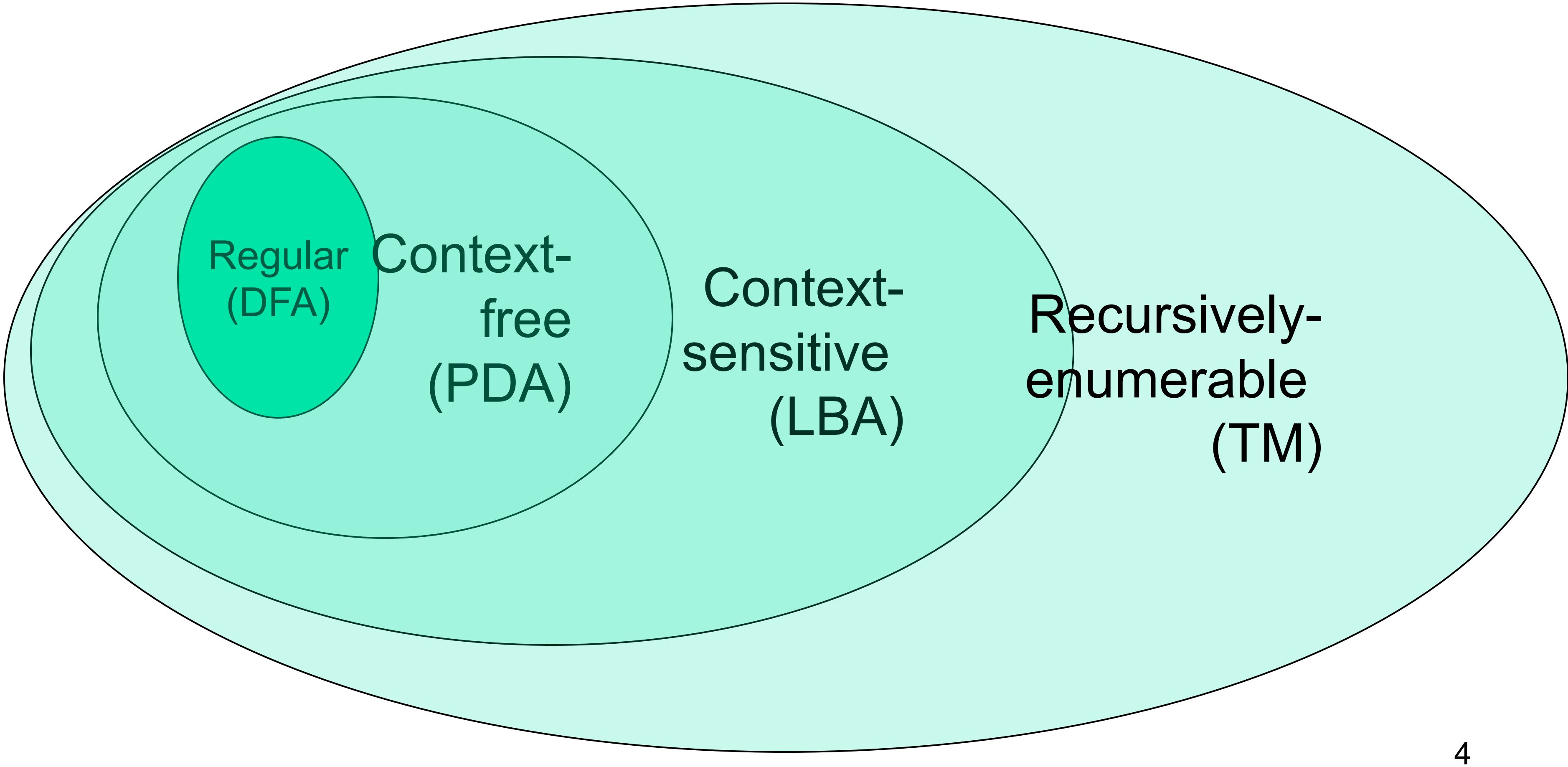
Image source: Nowak et al. Nature, vol 417, 2002

- <u>Languages</u>: "*A language is a collection of sentences of finite length all constructed from a finite alphabet of symbols*"
- <u>Grammars</u>: "*A grammar can be regarded as a device that enumerates the sentences of a language*" - nothing more, nothing less
- $G = (V_N, V_T, S, P)$
  - V_N – *list of non-terminal symb.*
  - V_T – *list of terminal symb.*
  - S – *start symb.*
  - P – list of production rules
  - $V_N \cap V_T = \emptyset$

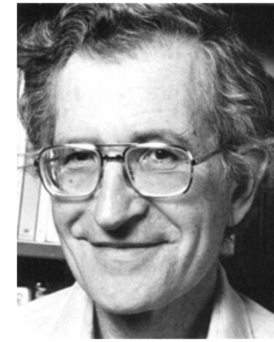
The Chomsky Hierachy

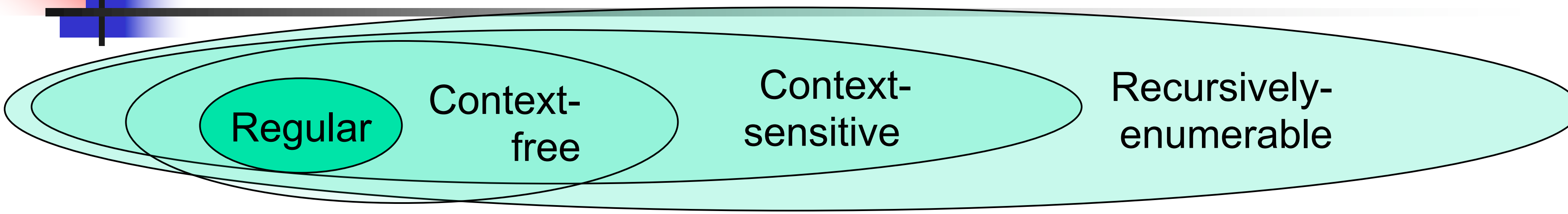


- A containment hierarchy of classes of formal languages



The Chomsky Hierarchy





Grammar	Languages	Automaton	Production Rules
Type-0	Recursively enumerable $\mathcal{L}_0$	Turing machine	$\alpha \rightarrow \beta$
Type-1	Context sensitive $\mathcal{L}_1$	Linear-bounded non-deterministic Turing machine	$\alpha A \beta \rightarrow \alpha \gamma \beta$
Type-2	Context-free $\mathcal{L}_2$	Non-deterministic push down automaton	$A \rightarrow \gamma$
Type-3	Regular $\mathcal{L}_3$	Finite state automaton	$A \rightarrow a$ and $A \rightarrow aB$

The Chomsky Hierarchy (cont'd)

The Chomsky Hierarchy (cont'd)

Classification using the structure of their rules:

- *Type-0 grammars*: there are no restriction on the rules;
- *Type-1 grammars/Context sensitive grammars*: the rules for this type have the next form:

$$uAv \to upv,\ u,p,v \in V_G^*,\ p \neq \lambda,\ \mathrm{A} \in V_N$$

or $A \to \lambda$ and in this case A does not belong to any right side of a rule.

Remark. The rules of the second form have sense only if A is the start symbol.

The Chomsky Hierarchy (cont'd)

Remarks

1. A grammar is *Type 1 monotonic* if it contains no rules in which the left-hand side consists of more symbols than the right-hand side. This forbids, for instance, the rule , $.NE \rightarrow and\ N$, where N, E are non-term. symb.; and is a terminal symb ($3 = |.NE| \geq |and\ N| = 2$).

The Chomsky Hierarchy (cont'd)

Remarks

- A grammar is *Type 1 context-sensitive* if all of its rules are context-sensitive. A rule is context-sensitive if actually only one (non-terminal) symbol in its left-hand side gets replaced by other symbols, while we find the others back undamaged and in the same order in the right-hand side.
- **Example**: $Name\ Comma\ Name\ End \rightarrow Name\ and\ Name\ End$ meaning that the rule $Comma \rightarrow and$ may be applied if the left context is $Name$ and the right context is $Name\ End$. The contexts themselves are not affected. The replacement must be at least one symbol long; this means that context-sensitive grammars are always monotonic.
- **Examples: see whiteboard**

The Chomsky Hierarchy (cont'd)

Classification using the structure of their rules:

- *Type-2 grammars/Context free grammars*: the rules for this type are of the form:

$$A \to p,\ p \in V_G^*,\ \mathrm{A} \in V_N$$

- *Type-3 grammars/*regular grammars: the rules for this type have one of the next two forms:

Cat. I rules	$A \to Bp$	or	$A \to pB$
Cat. II rules	$\mathrm{C} \to q$		$\mathrm{C} \to q$

$$\mathrm{A}, B, C \in V_N,\ p, q \in V_T^*$$

- **Examples: see whiteboard**

The Chomsky Hierarchy (cont'd)

Localization lemma for context-free languages (CFL) (or uvwxy theorem or pumping lemma for CFL)

Motivation for the lemma: almost anything could be expressed in a CF grammar.

Let G be a context free grammar and the derivation $x_1 \ldots x_m \overset{*}{\Rightarrow} p$, where $x_i \in V_G$, $p \in V_G^*$. Then there exists $p_1 \ldots p_m \in V_G^*$ such that $p = p_1 \ldots p_m$ and $x_j \overset{*}{\Rightarrow} p_j$.

Example: see whiteboard

The Chomsky Hierarchy (cont'd)

What do you observe on the right-hand side (RHS) of the production rules of a context-free grammar?

The Chomsky Hierarchy (cont'd)

- *It is convenient to have on the RHS of a derivation only terminal or nonterminal symbols!*
- *This can be achieved without changing the type of grammar.*

Lemma $A \rightarrow i$

Let system $G = (V_N, V_T, S, P)$ be a context-free grammar.

There exists an *equivalent* context free grammar G' with the property: if one rule contains terminals then the rule is of the form $A \rightarrow i$, $\mathrm{A} \in V_N$, $i \in V_T$.

The Chomsky Hierarchy (cont'd)

Lemma $A \to i$

Let system $G = (V_N, V_T, S, P)$ be a context free grammar.

There exists an *equivalent* context free grammar G' with the property: if one rule contains terminals then the rule is of the form $A \to i$, $\mathrm{A} \in V_N$, $i \in V_T$.

Proof. *Let* $G' = (V_N', V_T', S', P')$, where $V_N \subseteq V_N'$ *and* P' *contains all „convenient" rules from* P. *Let the following incoveninent rule:*

$$u \to v_1 i_1 v_2 i_2 \ldots i_n v_{n+1}, \qquad i_k \in V_T, \qquad v_k \in V_N^*$$

We add to P' the following rules:

$$u \to v_1 X_{i1} v_2 X_{i2} \ldots X_{in} v_{n+1}$$

$$X_{ik} \to i_k \qquad k = 1..n,\ X_{ik} \in {V_N}'$$

Key ideas in the transformation!

The Chomsky Hierarchy (cont'd)

What is the relationship between $\mathcal{L}_0$, $\mathcal{L}_1$, $\mathcal{L}_2$, $\mathcal{L}_3$?

Closure properties of Chomsky families

Definition. Let $\circ$ be a binary operation on a family of languages L. We say that the family L is closed on the operation $\circ$ if $L_1, L_2 \in L$ then $L_1, L_2 \in L$.

Let $G_1 = (N_1, T_1, S_1, P_1)$, $G_2 = (N_2, T_2, S_2, P_2)$.

Closure of Chomsky families under union

The families $\mathcal{L}_0, \mathcal{L}_1, \mathcal{L}_2, \mathcal{L}_3$ are closed under union.

Key idea in the proof

$$G_{\cup} = (N_1 \cup N_2 \cup \{S\}, T_1 \cup T_2, P_1 \cup P_2 \cup \{S \rightarrow S_1 | S_2\})$$

Examples: see whiteboard

Closure properties of Chomsky families

Closure of Chomsky families under product

The families $\mathcal{L}_0, \mathcal{L}_1, \mathcal{L}_2, \mathcal{L}_3$ are closed under product.

Key ideas in the proof

For $\mathcal{L}_0, \mathcal{L}_1, \mathcal{L}_2$

$$G_p = (N_1 \cup N_2 \cup \{S\}, T_1 \cup T_2, P_1 \cup P_2 \cup \{S \to S_1 S_2\})$$

For $\mathcal{L}_3$

$$G_p = (N_1 \cup N_2, T_1 \cup T_2, S_1, P_1{}' \cup P_2)$$

where $P_1{}'$ is obtained from P_1 by replacing the rules $A \to p$ with $A \to pS_2$

Examples: see whiteboard

Closure properties of Chomsky families

Closure of Chomsky families under Kleene closure

The families $\mathcal{L}_0, \mathcal{L}_1, \mathcal{L}_2, \mathcal{L}_3$ are closed under Kleene closure operation.

Key ideas in the proof

For $\mathcal{L}_0, \mathcal{L}_1$

$$G^* = (V_N \cup \{S^*, X\}, V_T, S^*, P \cup \{S^* \to \lambda|S|XS, Xi \to Si|XSi, \qquad i \in V_T\})$$

The new introduced rules are of type 1, so G^* does not modify the type of G.

For $\mathcal{L}_2$

$$G^* = (V_N \cup \{S^*\}, V_T, S^*, P \cup \{S^* \to S^*S|\lambda\})$$

For $\mathcal{L}_3$

$$G^* = (V_N \cup \{S^*\}, V_T, S^*, P \cup P' \cup \{S^* \to S|\lambda\})$$

where P' is obtained with category II rules, from P, namely if $A \to p \in P$ then $A \to pS \in P$.

Examples: see whiteboard

Closure properties of Chomsky families

Observation. Union, product and Kleene closure are called regular operations.

Hence, *the language families from the Chomsky classification are closed under regular operations*.

Summary

- Chomsky hierarchy
  - Closure properties of Chomsky families