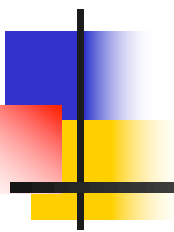


Course 2

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

An **alphabet** is a set of symbols:

$\{0,1\}$

Or “**words**”

↓
Sentences are strings of symbols:

0,1,00,01,10,1,...

A **language** is a set of sentences:

$L = \{000,0100,0010,.. \}$

A **grammar** is a finite list of rules defining a language.

$S \longrightarrow 0A$ $B \longrightarrow 1B$

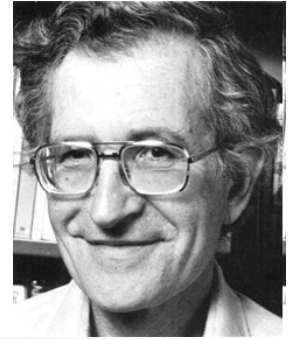
$A \longrightarrow 1A$ $B \longrightarrow 0F$

$A \longrightarrow 0B$ $F \longrightarrow \epsilon$

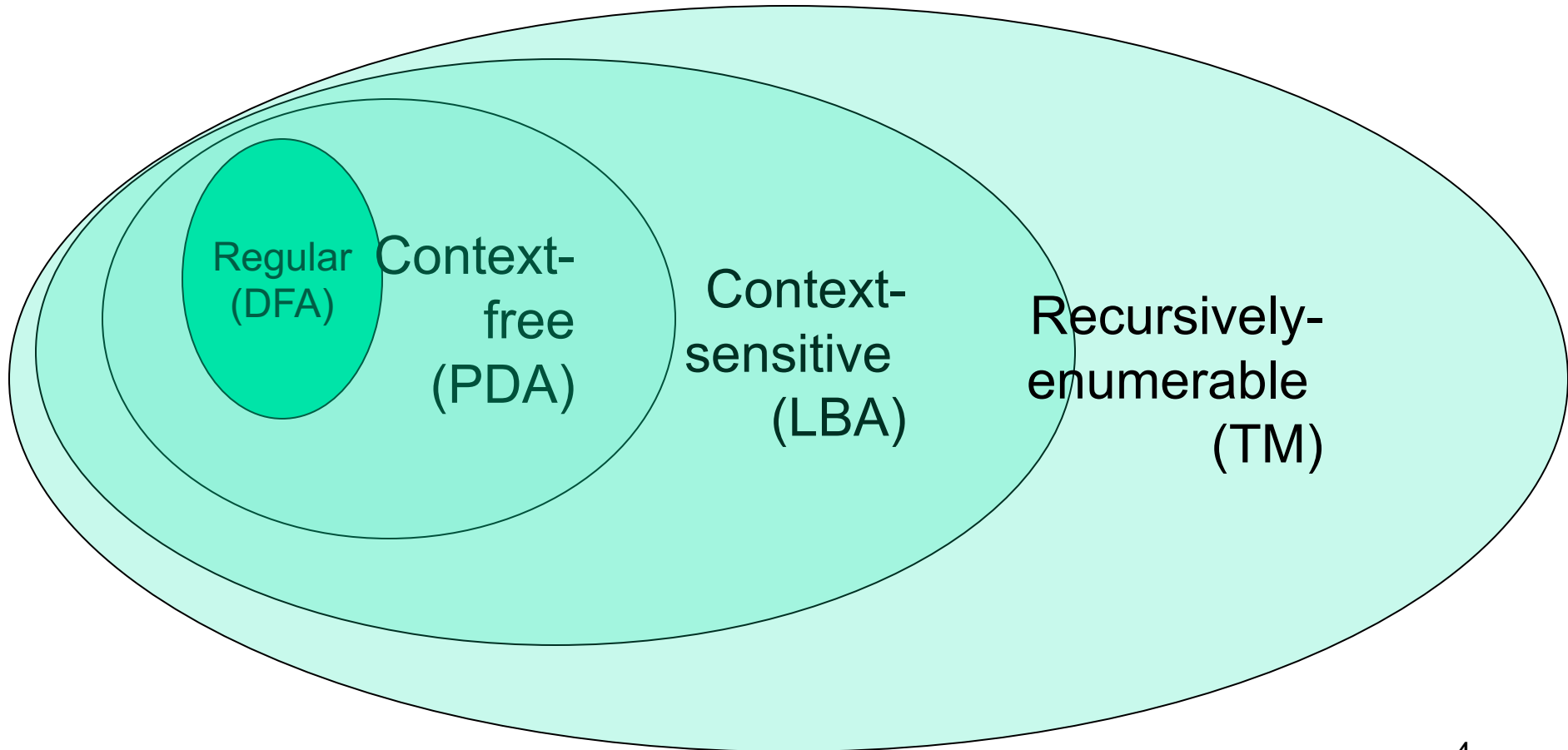
- Languages: “A language is a collection of sentences of finite length all constructed from a finite alphabet of symbols”
- Grammars: “A grammar can be regarded as a device that enumerates the sentences of a language” - nothing more, nothing less
- $G = (V_N, V_T, S, P)$
 - V_N – list of non-terminal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

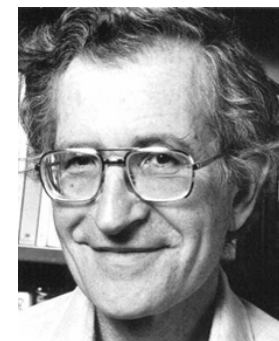


The Chomsky Hierarchy

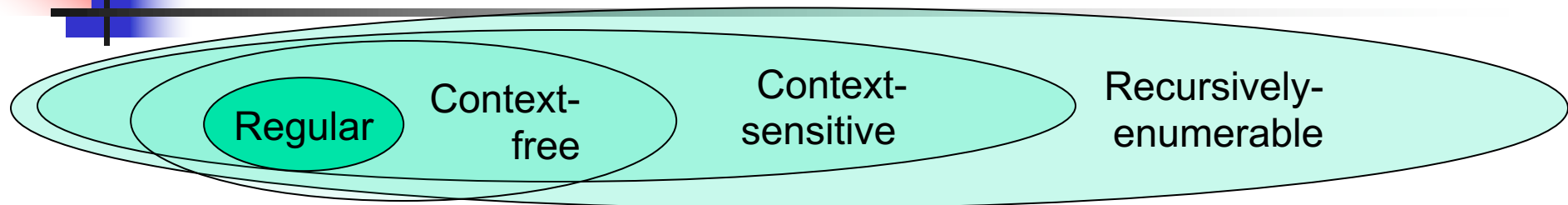


- A containment hierarchy of classes of formal languages





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:

$$uAv \rightarrow upv, u, p, v \in V_G^*, p \neq \lambda, A \in V_N$$

or $A \rightarrow \lambda$ 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 $\cdot NE \rightarrow \text{and } N$, where N, E are non-term. symb.; *and* is a terminal symb ($3 = |\cdot NE| \geq |\text{and } N| = 2$).



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:

$$A \rightarrow p, p \in V_G^*, A \in V_N$$

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

Cat. I rules $A \rightarrow Bp$

or

$A \rightarrow pB$

Cat. II rules $C \rightarrow q$

$C \rightarrow q$

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

- Rule $A \rightarrow \lambda$ is allowed if A does not belong to any right side of a rule.
- **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 $x_1 \dots x_m \xRightarrow{} p$, where $x_i \in V_G$, $p \in V_G^*$. Then there exists $p_1 \dots p_m \in V_G^*$ such that $p = p_1 \dots p_m$ and $x_j \Rightarrow p_j$.*

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 $A \rightarrow i$, $A \in V_N$, $i \in V_T$.

The Chomsky Hierarchy (cont'd)

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 $A \rightarrow i$, $A \in V_N$, $i \in V_T$.

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 *inconvenient* rule:

$$u \rightarrow v_1 i_1 v_2 i_2 \dots i_n v_{n+1}, \quad i_k \in V_T, \quad v_k \in V_N^*$$

We add to P' the following rules:

$$u \rightarrow v_1 X_{i_1} v_2 X_{i_2} \dots X_{i_n} v_{n+1}$$

$$X_{i_k} \rightarrow i_k \quad k = 1..n, \quad X_{i_k} \in V_N'$$

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 $L_1 \circ L_2 \in L$.

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

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

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 $A \rightarrow p$ with $A \rightarrow pS_2$

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

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

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

For \mathcal{L}_2

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

For \mathcal{L}_3

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

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

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