CSE322
The Chomsky Hierarchy

Lecture #16

Definitions

- **Language**: "A language is a collection of sentences of finite length all constructed from a finite alphabet of symbols."
- **Grammar**: "A grammar can be regarded as a device that enumerates the sentences of a language."
- A grammar of *L* can be regarded as a function whose range is exactly *L*

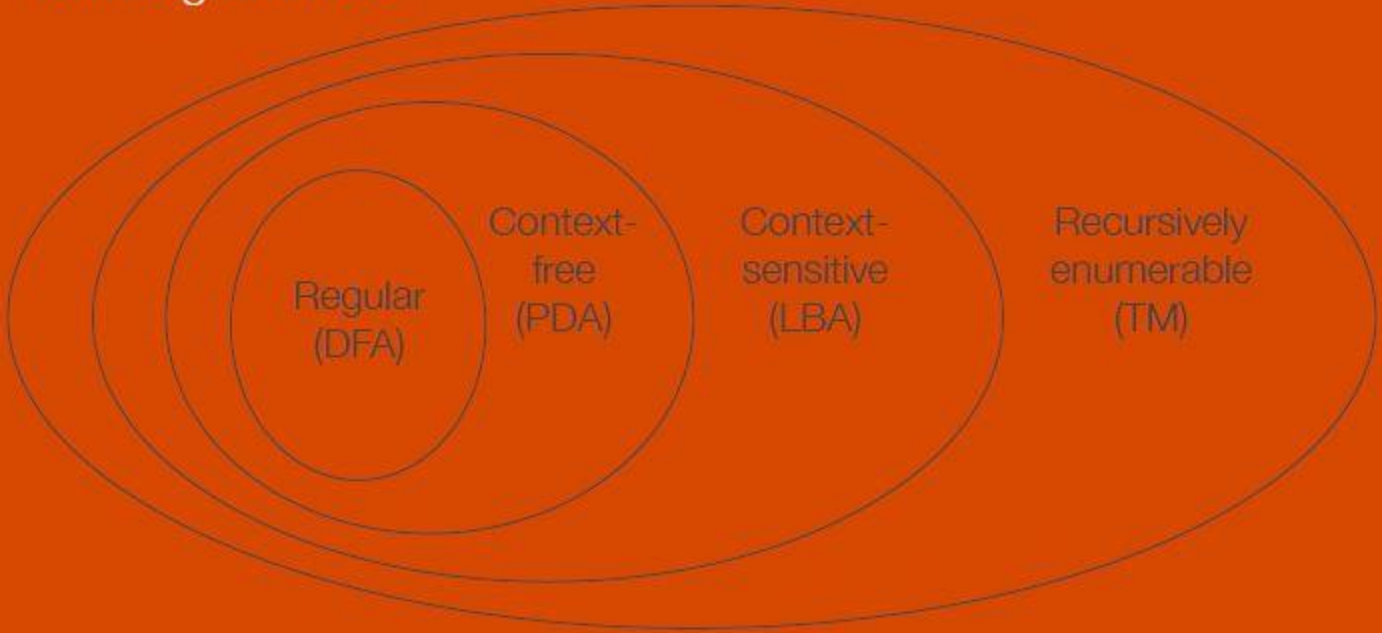
Noam Chomsky, *On Certain Formal Properties of Grammars*, Information and Control, Vol 2, 1959

Formal grammar

- A formal grammar is a quad-tuple $G = (N, \Sigma, P, S)$ where
  - N is a finite set of non-terminals
  - Σ is a finite set of terminals and is disjoint from N
  - P is a finite set of production rules of the form
    $w \in (N \cup \Sigma)* \rightarrow w \in (N \cup \Sigma)*$
  - $S \in N$ is the start symbol

The hierarchy

- A containment hierarchy (strictly nested sets) of classes of formal grammars



The hierarchy

Class	Grammars	Languages	Automaton
Type-0	Unrestricted	Recursively enumerable (Turing-recognizable)	Turing machine
Type-1	Context-sensitive	Context-sensitive	Linear-bounded
Type-2	Context-free	Context-free	Pushdown
Type-3	Regular	Regular	Finite

The hierarchy

Class	Grammars	Languages	Automaton
Type-0	Unrestricted	Recursively enumerable (Turing-recognizable)	Turing machine
	none	Recursive (Turing-decidable)	Decider
Type-1	Context-sensitive	Context-sensitive	Linear-bounded
Type-2	Context-free	Context-free	Pushdown
Type-3	Regular	Regular	Finite

Applications of Automata

- TM- Real Life Implementation ,Software Implementation
- LBA- Generic Programming, Parse Trees
- PDA-Online Tracking processing system,Top Down Parsing in LL Grammer
- FA-Finite State Programming,UML State Diagrams, Acceptors and Recoganizers, Lexical Analyzer

4.4 RECURSIVE AND RECURSIVELY ENUMERABLE SETS

The results given in this section will be used to prove $\mathscr{L}_{cs1} \subsetneq \mathscr{L}_0$ in Section 9.7. For defining recursive sets, we need the definition of a procedure and an algorithm.

A *procedure* for solving a problem is a finite sequence of instructions which can be mechanically carried out given any input.

An *algorithm* is a procedure that terminates after a finite number of steps for any input.

Definition 4.14 A set X is recursive if we have an algorithm to determine whether a given element belongs to X or not.

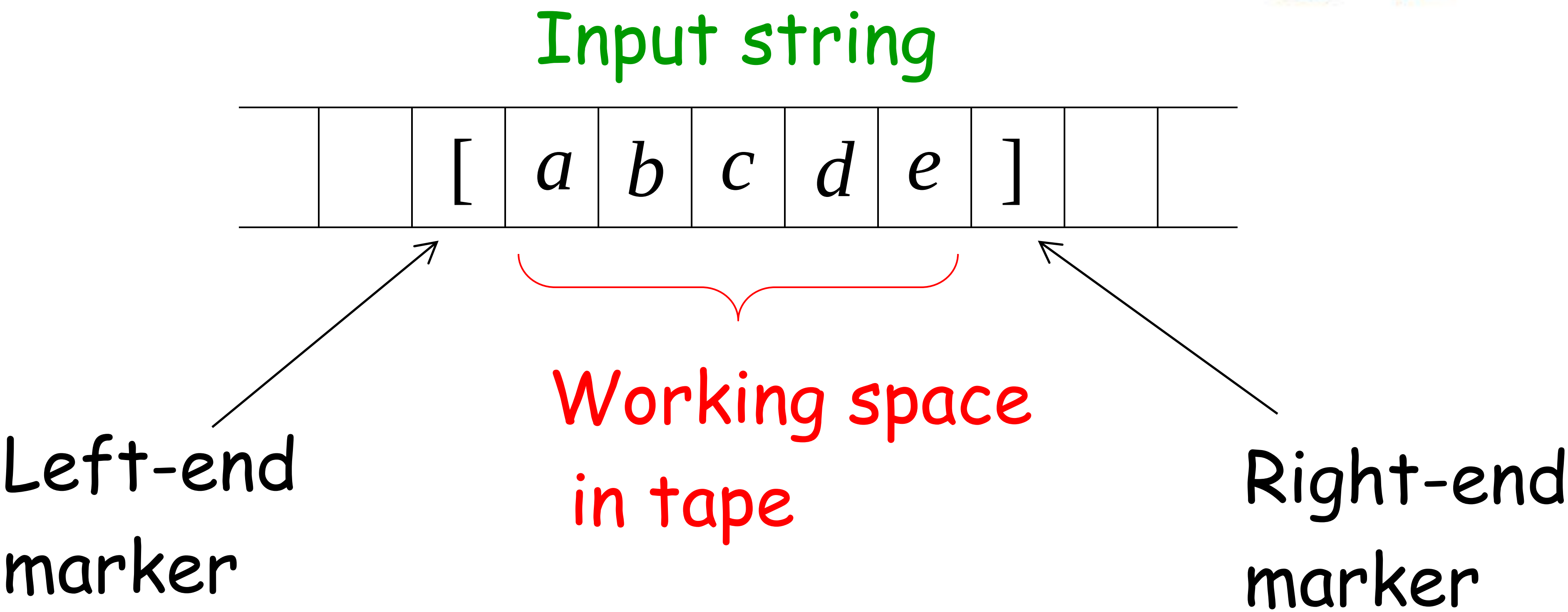
Definition 4.15 A recursively enumerable set is a set X for which we have a procedure to determine whether a given element belongs to X or not.

Linear-Bounded Automata:

Same as Turing Machines with one difference:

the input string tape space
is the only tape space allowed to use

Linear Bounded Automaton (LBA)



All computation is done between end markers

Open Problem:

NonDeterministic LBA's
have same power as
Deterministic LBA's ?

Example languages accepted by LBAs:

$$L = \{a^n b^n c^n\}$$

$$L = \{a^{n!}\}$$

LBA's have more power than PDA's
(pushdown automata)

LBA's have less power than Turing Machines

Unrestricted Grammars:

Productions

$$u \rightarrow v$$

String of variables and terminals

String of variables and terminals

Example unrestricted grammar:

$$S \rightarrow aBc$$

$$aB \rightarrow cA$$

$$Ac \rightarrow d$$

Theorem:

A language L is Turing-Acceptable if and only if L is generated by an unrestricted grammar

Context-Sensitive Grammars:

Productions

$$u \rightarrow v$$

String of variables and terminals

String of variables and terminals

and: $|u| \leq |v|$

The language $\{a^n b^n c^n\}$

is context-sensitive:

$$S \rightarrow abc \mid aAbc$$

$$Ab \rightarrow bA$$

$$Ac \rightarrow Bbcc$$

$$bB \rightarrow Bb$$

$$aB \rightarrow aa \mid aaA$$

Theorem:

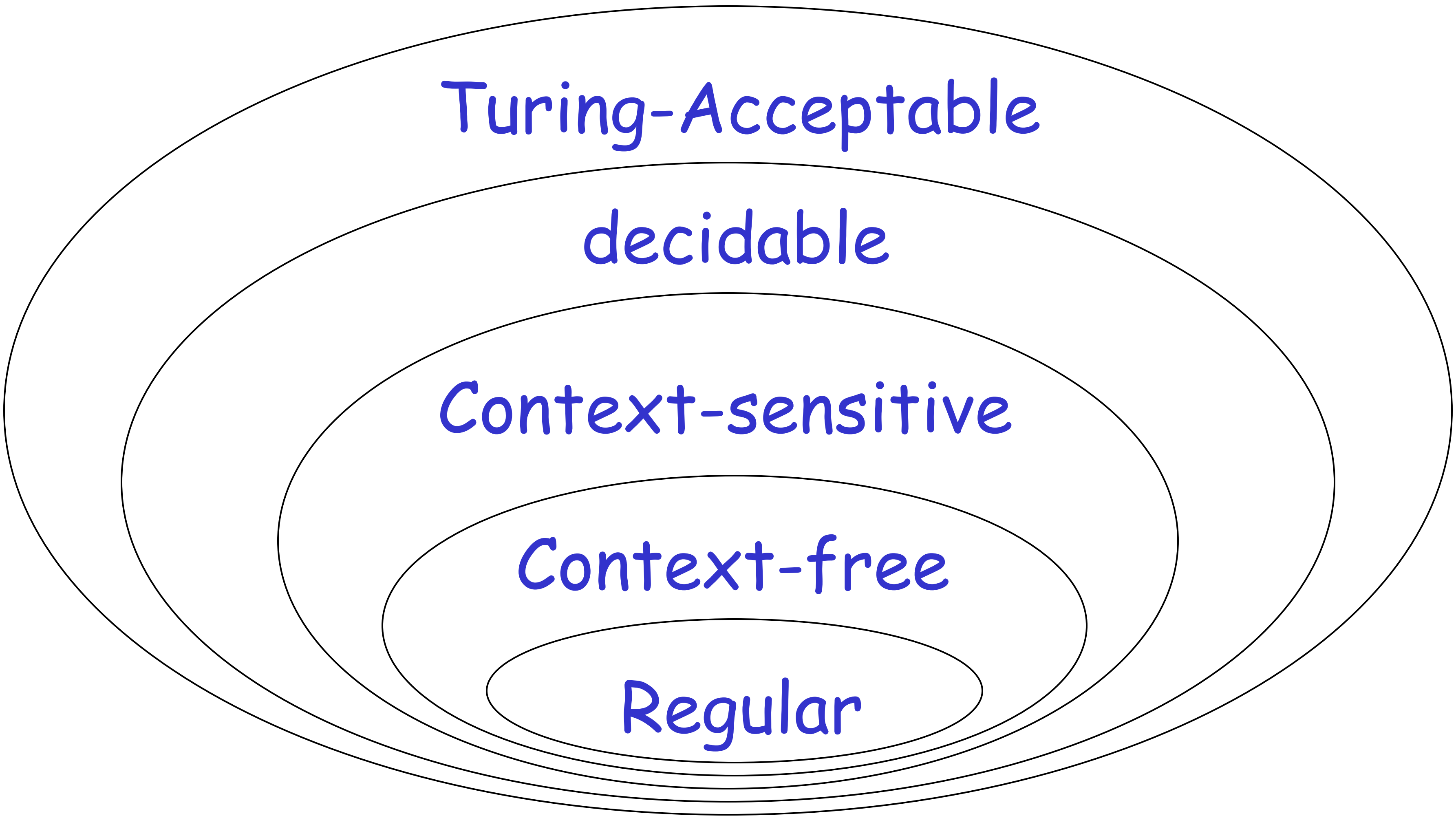
A language L is context sensistive
if and only if
it is accepted by a Linear-Bounded automaton

Observation:

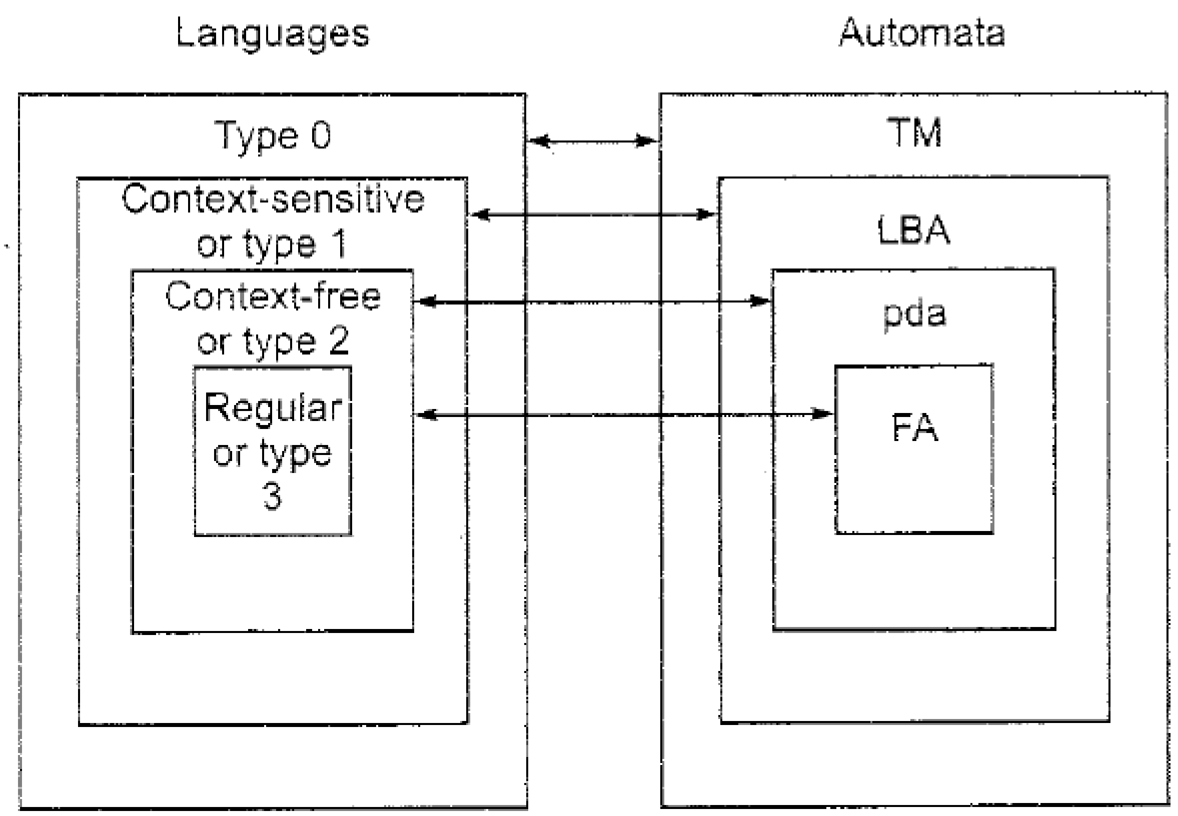
There is a language which is context-sensitive
but not decidable

The Chomsky Hierarchy

Non Turing-Acceptable



LANGUAGES AND AUTOMATON



Questions

EXAMPLE 4.2

If $G = (\{S\}, \{0, 1\}, \{S \rightarrow 0S1, S \rightarrow \Lambda\}, S)$, find $L(G)$.

EXAMPLE 4.3

If $G = (\{S\}, \{a\}, \{S \rightarrow SS\}, S)$, find the language generated by G.

EXAMPLE 4.4

Let $G = (\{S, C\}, \{a, b\}, P, S)$, where P consists of $S \rightarrow aCa$, $C \rightarrow aCa \mid b$. Find $L(G)$.

EXAMPLE 4.5

If G is $S \rightarrow aS \mid bS \mid a \mid b$, find $L(G)$.

EXAMPLE 4.7

Construct a grammar generating $L = \{wcw^T \mid w \in \{a, b\}^*\}$.

EXAMPLE 4.9

Find a grammar generating $\{a^j b^n c^n \mid n \geq 1, j \geq 0\}$.

EXAMPLE 4.8

EXAMPLE 4.10

Let $G = (\{S, A_1\}, \{0, 1, 2\}, P, S)$, where P consists of $S \rightarrow 0SA_12$, $S \rightarrow 012$, $2A_1 \rightarrow A_12$, $1A_1 \rightarrow 11$. Show that

$$L(G) = \{0^n 1^n 2^n \mid n \geq 1\}$$

Prove it as $a^n b^n c^n$

$$S \rightarrow aSBC \mid aBC,\ CB \rightarrow BC,\ aB \rightarrow ab,\ bB \rightarrow bb,\ bC \rightarrow bc,\ cC \rightarrow cc$$

$$S \Rightarrow aBC \Rightarrow abC \Rightarrow abc$$

Thus

EXAMPLE 4.14

If the grammar G is given by the productions $S \rightarrow aSa \mid bSb \mid aa \mid bb \mid \Lambda$, show that (i) $L(G)$ has no strings of odd length, (ii) any string in $L(G)$ is of length $2n$, $n \geq 0$, and (iii) the number of strings of length $2n$ is 2^n.

EXAMPLE 4.15

Let $G = (\{S, A_1, A_2\}, \{a, b\}, P, S)$, where P consists of

$$S \rightarrow aA_1A_2a,\ A_1 \rightarrow baA_1A_2b,\ A_2 \rightarrow A_1ab,\ aA_1 \rightarrow baa,\ bA_2b \rightarrow abab$$

Test whether $w = baabbabaaabbaba$

is in $L(G)$.

EXAMPLE 4.17

Find the highest type number which can be applied to the following productions:

(a) $S \to Aa, \quad A \to c \mid Ba. \quad B \to abc$

(b) $S \to ASB \mid d, \quad A \to aA$

(c) $S \to aS \mid ab$

EXAMPLE 4.19

Construct a context-free grammar generating

(a) $L_1 = \{a^n b^{2n} \mid n \geq 1\}$
(b) $L_2 = \{a^m b^n \mid m > n,\ m,\ n \geq 1\}$
(c) $L_3 = \{a^m b^n \mid m < n,\ m,\ n, \geq 1\}$

Left Linear Grammer vs Right Linear Grammer

Left Linear and Right Linear Grammar

Left Linear Grammar: In a grammar if all productions are in the form $A \rightarrow B\alpha$ or $A \rightarrow \alpha$, where $A, B \in V_N$ and $\alpha \in \Sigma^*$, then the grammar is called Left Linear Grammar.

Example - $A \rightarrow Aa \mid Bb \mid b$

Right Linear Grammar: In a grammar if all productions are in the form $A \rightarrow \alpha B$ or $A \rightarrow \alpha$, where $A, B \in V_N$ and $\alpha \in \Sigma^*$, then the grammar is called Right Linear Grammar.

Example - $A \rightarrow aA \mid bB \mid b$

Left Linear Grammer vs Right Linear Grammer

- Regular language works on right linear
- Whereas CFG and CSG can work on left linear