ITS64304 Theory of Computation

School of Computer Science
Taylor's University Lakeside Campus

Lecture 7: Chomsky Hierarchy

Dr Raja..

Source: Thomas Sudkamp. Languages and Machines: An Introduction to the Theory of Computer Science, Addison-Wesley, 3rd Edition, 2005

Learning outcomes

At the end of this topic students should be able to:

- Describe the relationship between different grammar types
- Analyze and compare the characteristics of different models of computation using Chomsky hierarchy*

* Course Learning Outcome 3

Grammar

- A set of rules for generating strings
- *V* is the set of non-terminal symbols e.g., *A, B, X, Y* , ...
- *T* is the set of terminal symbols e.g., *a, b, 1, 2, ...*
- A rule is of the form: $A \rightarrow aA$
- *Application of a rule* transforms one string to another

- Grammars generate languages and Automata accept languages

Grammar types

- Various restrictions on rules provide various grammar types

Grammar Type	**Restrictions**	**E.g.**
Unrestricted (0)		$AbC \longrightarrow AC$
Context-sensitive (1)	$\|L\| \leq \|R\|$	$AbC \;\; u \; AaC$
Context-free (2)	$\|L\| = 1$	$A \;\longrightarrow AaC$
Regular (3)	$\|L\| = 1$ and $R \in T$ **|**ε| TV ($R \in \{a, \varepsilon , A\}$)	$B \longrightarrow 3$ $B \longrightarrow \varepsilon$ $A \longrightarrow 1C$

Regular ⊂ Context free ⊂ Context Sensitive ⊂ Unrestricted

Chomsky Hierarchy

AUTOMATA	POWER	GRAMMARS
Turing Machines	💪💪💪💪	unrestricted grammars
Linear-bounded Automata	💪💪💪	context-sensitive grammars
PDA	💪💪	context-free grammars
NFA/DFA	💪	regular grammar regular expressions

Chomsky Normal Form

- A CFG is in **Chomsky normal form** if its rules are of the form:
  - $A \rightarrow BC$ or
  - $A \rightarrow a$ or
  - $S \rightarrow e$

  - S is the start symbol.
  - Neither B nor C may be the start symbol.
  - RHS of a rule in CNF must have length 2 (*e.g.*: $A \rightarrow BC$)

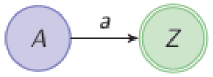
- Every grammar in CNF is context-free.

Chomsky Hierarchy

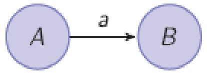
- Noam Chomsky developed a 4 level classification for grammars

- **Type 3** **Regular Grammars**
  - least powerful
  - can be accepted by DFA/NFA
  - can use subset of CFG, where all rules of the form:

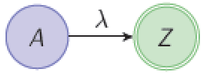
  $A \rightarrow a$



  $A \rightarrow aB$ (or $A \rightarrow Ba$)



  $A \rightarrow \lambda$



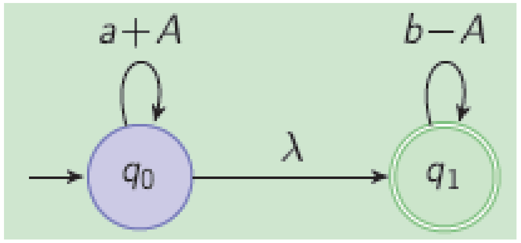
- Regular language is language expressed by a regular grammar
  - all regular languages are context-free
  - some context free languages are not regular (eg $a^n b^n$ | n > 0)

Chomsky Hierarchy

- **Type 2** - **Context Free Grammars**
  - accepted by PDA
  - all rules have a single non-terminal on the LHS

- Example: $a^i b^i$

  $S \rightarrow aSb \mid \lambda$



- '*aSb*' is equivalent to
  - push X on PDA stack when input '*a*'
  - pop X off PDA stack when input '*b*'

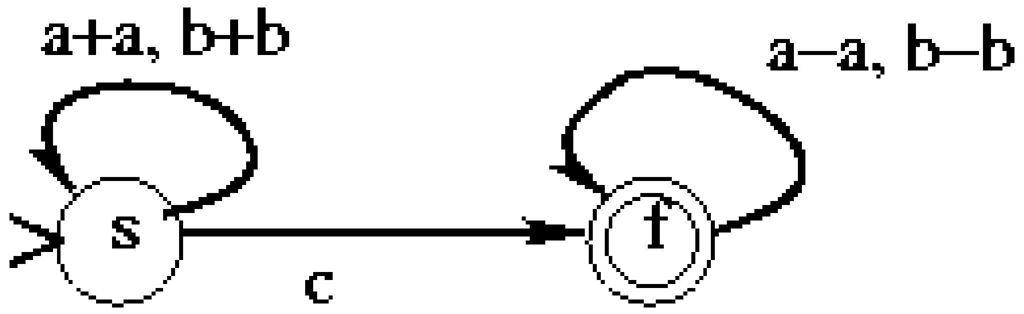
CFG and PDA

- Language accepted by PDA **iff** generated by a CFG
- e.g $a^i c^j d^j b^i$

  $S \rightarrow aSb \mid cSd \mid e$

- e.g palindrome $\{wcw^R \mid w \in \{a, b\}^* \}$

  $S \rightarrow aSa \mid bSb \mid c$



- Note: PDA is deterministic

Chomsky Hierarchy

- **<u>Type 1</u> - Context Sensitive Grammars**
  - More than one symbol permitted on LHS (but only 1 non-terminal)
    - e.g. $uAv \rightarrow uwv$ (means '*A*' can only be replaced if '*u*' is the prefix of *A* and *v* the suffix. The *context* of '*A*' matters!)
    - accepted by a restricted Turing machine (called a linear-bounded automata - the length of the tape is limited)
- Example:

$$S \rightarrow abc \mid aSBc$$
$$cB \rightarrow Bc$$
$$bB \rightarrow bb$$

recognizes $a^n b^n c^n$

Context Sensitive Grammars

$S \rightarrow abc \mid aSBc$

$cB \rightarrow Bc$

$bB \rightarrow bb$

- accept *aaabbbccc* ?

- $S \Rightarrow aSBc \Rightarrow aaSBcBc \Rightarrow aaabcBcBc \Rightarrow aaabBccBc \Rightarrow aaabbccBc \Rightarrow aaabbcBcc \Rightarrow aaabbBccc \Rightarrow aaabbbccc$

Chomsky Hierarchy

- **Type 0** **Unrestricted Grammars**
  - can be accepted only by a Turing machine
  - any type of rule structure possible as long as at least one non-terminal is on the LHS (e.g. $BA \rightarrow D$ is OK)

- Two types of language can result from unrestricted grammars
  - recursive
  - recursively enumerable

Unrestricted Grammars

- Recursive language:
  - Turing machine eventually stops if string *w* **is in** language.
  - Machine will also eventually stop (and reject) if *w* **is not in** language

- Recursively enumerable language:
  - Turing machine eventually stops if *w* **is in** language.
  - But machine *may go into an infinite loop if w* ***is not in*** *the language*

Chomsky Hierarchy summarized

Grammar	Languages	Automaton	Production Rule of the form (e.g.)
Type - 3	Regular	Finite State Automaton	$A \rightarrow a$ $A \rightarrow aB$ |$A \rightarrow Ba$) $A \rightarrow e$
Type – 2	Context-free	Non-deterministic Pushdown Automaton	$S \rightarrow aSb \mid \lambda$
Type - 1	Context-sensitive	Non-deterministic Turing Machine	$S \rightarrow abc \mid aSBc$ $cB \rightarrow Bc$ $bB \rightarrow bb$
Type – 0	Unrestricted	Turing Machine	$BA \rightarrow D$

Conclusion

- Grammars are rules for string generation/replacement
- All grammar classes are equivalent to some kind of automaton.
- PDAs and context-free grammars have the same power.