San José State University
Department of Computer Science

Ahmad Yazdankhah
ahmad.yazdankhah@sjsu.edu
www.cs.sjsu.edu/~yazdankhah

Grammars

(Part 3)

Lecture 22

Day 24/31

CS 154

Formal Languages and Computability

Fall 2019

Agenda of Day 24

- Quiz++ and Quiz 8 Papers Are Not Ready.
- Summary of Lecture 21
- Lecture 22: Teaching ...
  - Grammars (Part 3)

Summary of Lecture 21: We learned ...

Grammars

- Associated language to the grammar G is ...
  - ... the set of all strings generated by it.
  - ... denoted by L(G).
- Formal definition of grammar:

$$G = (V, T, S, P)$$

- Two grammars are equivalent if ...
  - ... both has the same associated language.

Types of Grammars

- A grammar G is linear if ...
  - ... the right hand side of every production rule has at most one variable.
- Right-linear grammar is ...
  - ... a linear grammar whose production rules are of the form:

  $A \rightarrow w \mid u\,B$

  Where $A, B \in V$ and $w, u \in T^*$
- Left-linear grammar is ...
  - ... a linear grammar whose production rules are of the form:

  $A \rightarrow w \mid B\,u$

  Where $A, B \in V$ and $w, u \in T^*$

Summary of Lecture 21: We learned ...

Types of Grammars (cont'd)

- A grammar is said to be regular if ...
  - ... it is either right-linear or left-linear.

Theorems

- Regular grammars produce regular languages.
- Regular languages have regular grammars.

- Regular Languages Representations

Any Question

Context-Free Grammars (CFG)

Context-Free Grammars (CFGs)

Definition

- A grammar G is said to be context-free (CFG) if all production rules are of the form:

- Note again that:
  In this course, LHS has always one variable.

$A \rightarrow v$

Where $A \in V$ and $v \in (V \cup T)^*$

Example 18

- Is the following grammar context-free?

  $S \rightarrow a\, S\, b \mid \lambda$

- Yes, it is.

CFGs Examples

Example 19

- Let the grammar G be:

  $S \rightarrow a\,S\,a \mid b\,S\,b \mid \lambda$

1. Is G context-free?
2. L(G) = ? // show it by a set-builder.

Solution

CFGs Examples

Example 20

- Let the grammar G be:

  $S \to S\,S \mid a\,S\,b \mid b\,S\,a \mid \lambda$

1. Is G context-free?
2. L(G) = ? // show it by a set-builder.

Solution

Context-Free Languages (CFL)

Definition

- A language L is said to be context-free (CFL) if there exists a context-free grammar G such that $L = L(G)$.

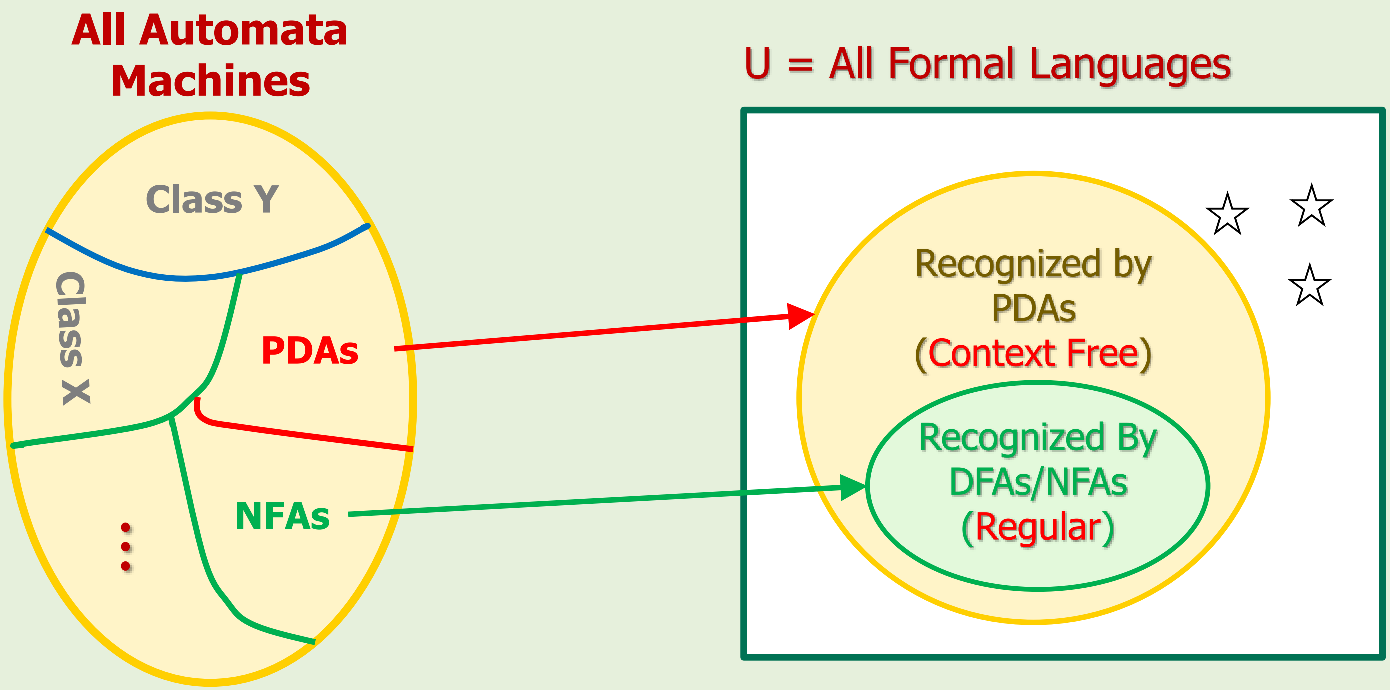
- Therefore, all of the following languages are context-free:
- $L = \{a^n b^n : n \geq 0\}$
- $L = \{ww^R : w \in \Sigma^*\}$
- $L = \{w : n_a(w) = n_b(w), w \in \{a, b\}^*\}$

- Note that a regular grammar is CFG but NOT vice versa!

PDAs and Languages Association

Flashback

- We mentioned CFLs when we introduced PDAs.
  - We were supposed to explain them later.



- PDAs can recognize CFLs.

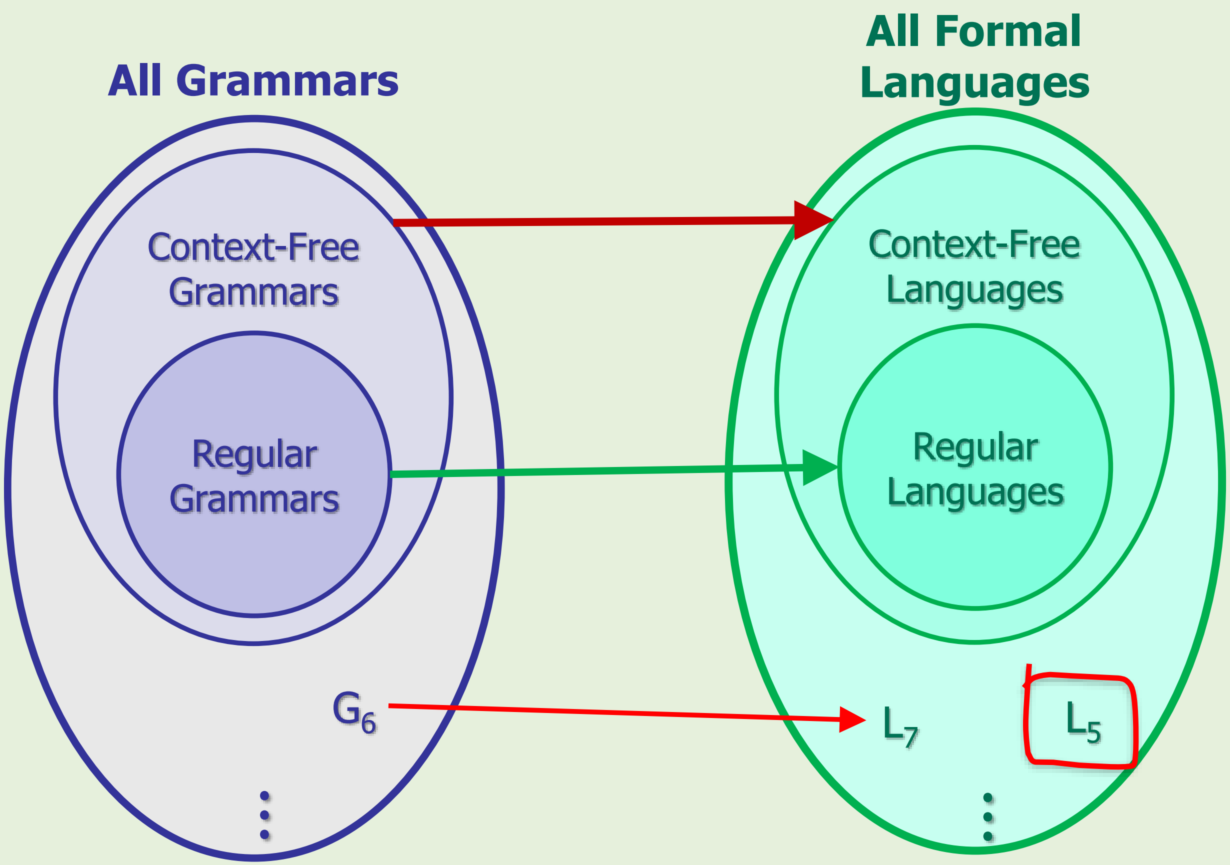
CFLs Representations

- So, now we have two ways for representing CFLs:
  - PDAs
  - CFGs

- Let's revisit the grammars and languages association.

Grammars and Languages Association

Revisited



Application of CFGs in Programming Languages

Example 21

- Consider $L_1 = \{a^n b^n : n \geq 0\}$ over $\Sigma = \{a, b\}$.
- Let's take a different look at this language.

- For example, consider this language:
- $L_2 = \{(^n\ )^n : n \geq 0\}$ over $\Sigma = \{(\ ,\ )\}$ //parentheses are just symbols!

1. What strings would this language contain?
2. What strings do not belong to this language?

- What is L_2 representing?

Grammars Hierarchy

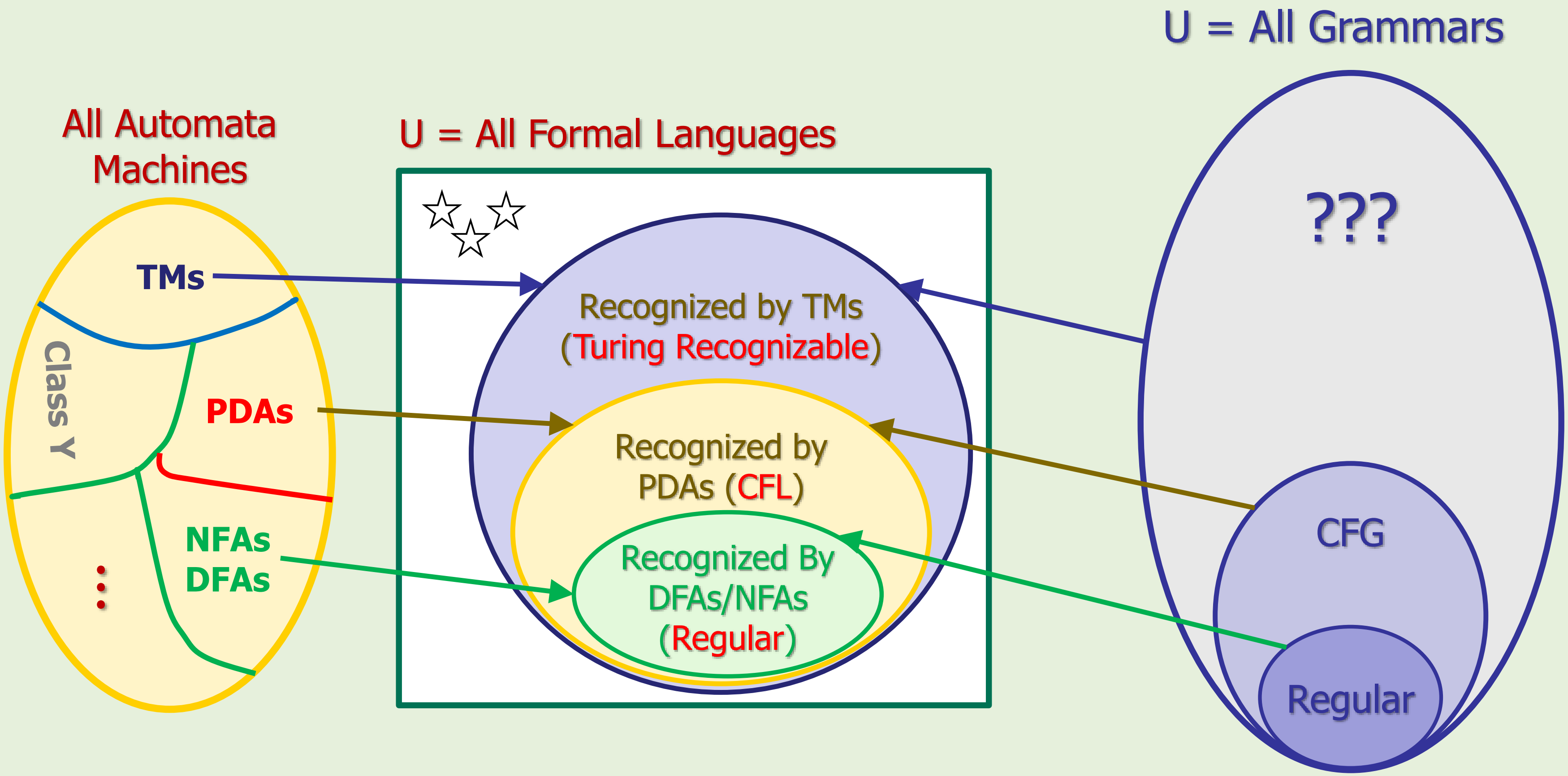
CFG for More Complex Languages

- Find a CFG for each of the following languages:
  1. $L = \{a^n b^n c^n : n \geq 0\}$ over $\Sigma = \{a, b, c\}$
  2. $L = \{ww : w \in \Sigma^*\}$ over $\Sigma = \{a, b\}$

Solution

- Struggling?!
- After some struggling, you realize that you cannot find any grammar for these languages! Why?
- Recall that we could not construct PDAs for these languages too!

Machines, Languages, and Grammars Association



- Is there any other grammar that can produce more complex languages like "Turing-recognizable"?

Context-Sensitive Grammar (CSG)

Definition

- A grammar G is said to be context-sensitive (CSG) if all production rules are of the form:

$$xAy \rightarrow xvy$$

Where $A \in V$ and $x, y, v \in (V \cup T)^*$ and $v \neq \lambda$

Example

- Context-sensitive example

$aSb \rightarrow aSSb$

$A \rightarrow bcA$

$B \rightarrow b$

Recursively Enumerable Grammar

Definition

- A grammar G is said to be Recursively enumerable (aka unrestricted) if all production rules are of the form:

$xAy \rightarrow z$

Where $A \in V$, $x, y, z \in (V \cup T)^*$

Example 22

$S \rightarrow bcA \mid aAbB \mid A$

$aAbB \rightarrow bcA \mid A \mid \lambda$

$A \rightarrow a \mid \lambda$

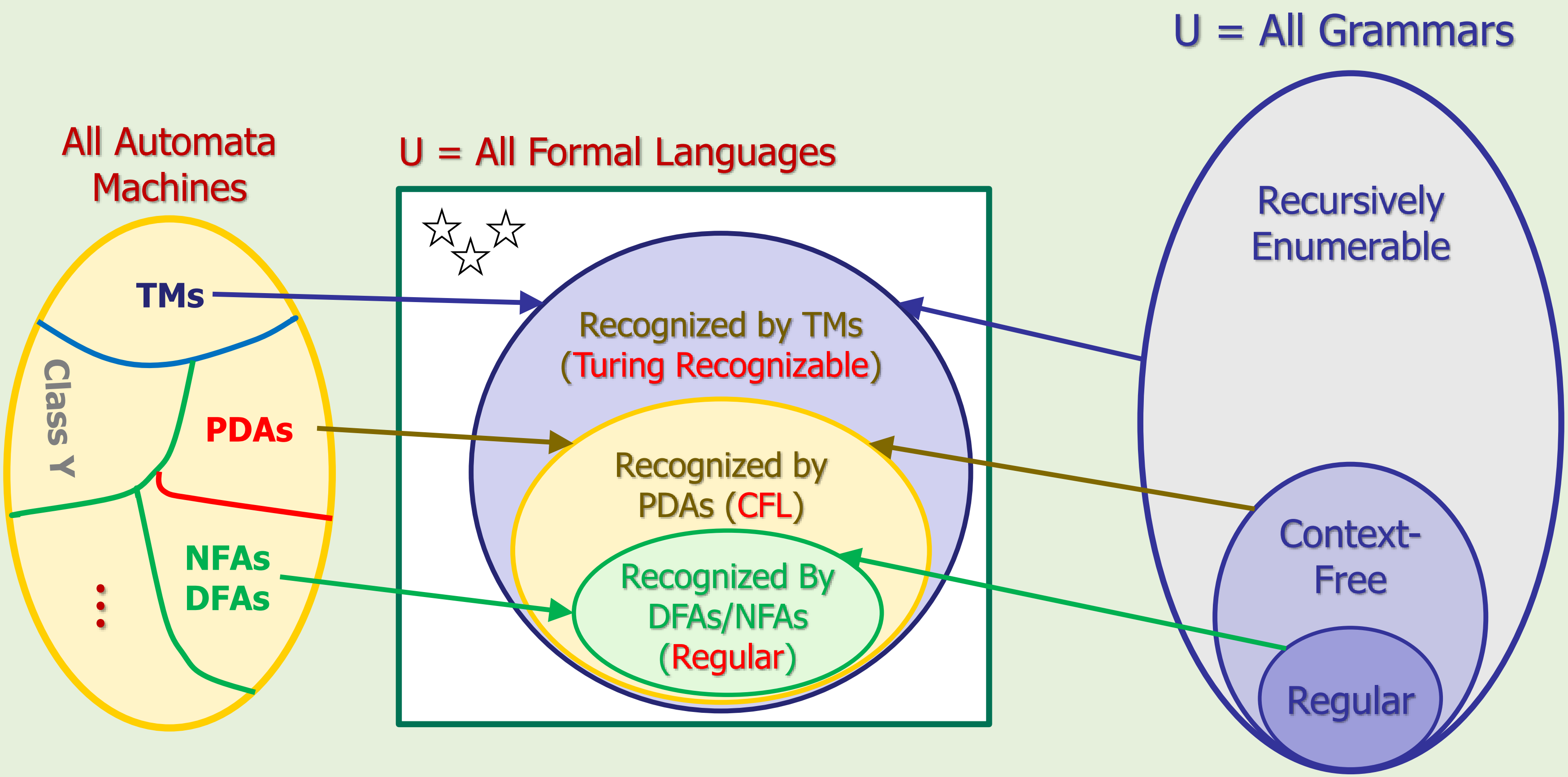
$bcA \rightarrow bbA \mid \lambda$

...

- This type of grammar can produce Turing-recognizable languages.

Machines, Languages, and Grammars Association

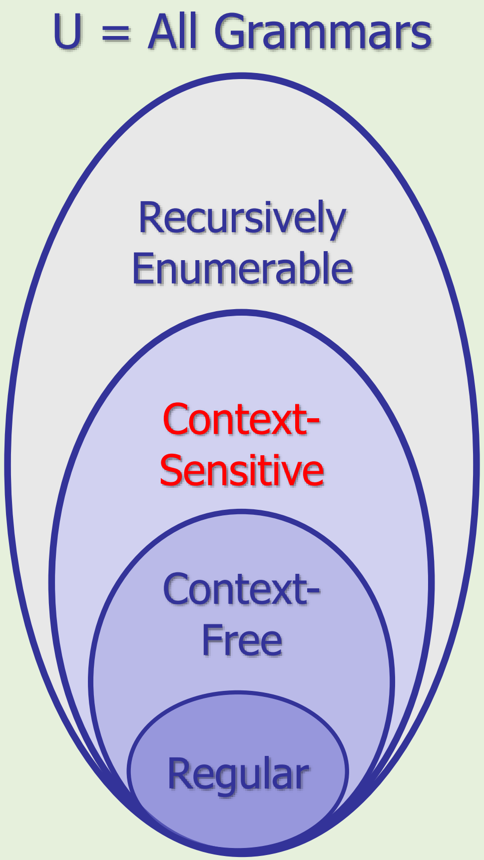
Revisited



- Note that recursively enumerable grammars include all formal grammars.

Grammars Hierarchy

- So, a normal question here would be:
- What kind of automata is associated with context-sensitive grammars?
- There is a class of automata but it is very hard and beyond the scope of this course.
- So, we won't cover that.
- Note that both context-sensitive and recursively enumerable grammars are beyond the scope of this course.
- We just need to know their definition and there won't be any homework or exercises.
- These categorizations was defined by Noam Chomsky (next slide).



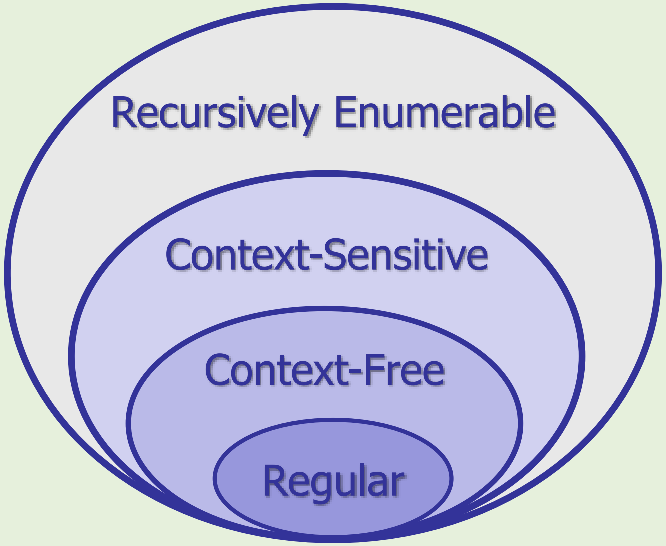
Chomsky's Hierarchy

- Avram Noam Chomsky, the American linguist, philosopher, and historian (1928 - ?), has categorized formal languages that is called "Chomsky's Hierarchy".

- He categorized formal grammars into 4 types as:

- Type 0: Recursively-enumerable
- Type 1: Context-sensitive
- Type 2: Context-free
- Type 3: Regular

U = All Grammars



Derivations Techniques

Derivations Techniques

- Consider a production rule that has two or more variables.

  $S \rightarrow a\ A\ B$

  $A \rightarrow \ldots$

  $B \rightarrow \ldots$

- To derive a string, we should substitute A and B with some other production rules.
- But in what order?
  - We can substitute them randomly.
  - Or we can pick a specific order. (e.g. left var first or right var first ...)
- Note that we are looking into this question from a software angle, not a human.

Derivations Techniques Example

Example 23

- Derive string "aab" from the following grammar:

  $S \rightarrow AB$
  $A \rightarrow aaA \mid \lambda$
  $B \rightarrow Bb \mid \lambda$

- **Approach 1**: Substitute the leftmost variables first

  $$S \overset{1}{\Rightarrow} AB \overset{2}{\Rightarrow} aaAB \overset{3}{\Rightarrow} aaB \overset{4}{\Rightarrow} aaBb \overset{5}{\Rightarrow} aab$$

- **Approach 2**: Substitute the rightmost variables first

  $$S \overset{1}{\Rightarrow} AB \overset{4}{\Rightarrow} ABb \overset{5}{\Rightarrow} Ab \overset{2}{\Rightarrow} aaAb \overset{3}{\Rightarrow} aab$$

- Both derivations yielded the same results.

Leftmost / Rightmost Derivations

Definition

- A derivation is said to be leftmost if in each step the leftmost variable in the sentential form is substituted.

Definition

- A derivation is said to be rightmost if in each step the rightmost variable in the sentential form is substituted.

- The default method would be "leftmost" if we don't mention specifically.

Homework

- Derive string "abbbb" from the following grammar:
  1. $S \rightarrow aAB$
  2. $A \rightarrow bBb$
  3. $B \rightarrow A \mid \lambda$

- Leftmost derivation:

- Rightmost derivation:

Parsing

Introduction

- Parsing is a very important topic in computer science.
- There are many theorems, algorithms, and
  a lot of researches about it.

- In this lecture, we give you only a big picture about it.
- So, consider this as a very short introduction about parsing.

- For more information, you need to take "Compiler Course".

Motivation

- Assume you have the following statement in your Java program:

```
if (x > 5) {
  y = y * 2 + 1;
}
```

- How does Java compiler know that this is a valid statement?
  - Note: valid = well-formed

- To answer this question, let's remove all whitespaces:

```
if(x>5){y=y*2+1;}
```

- This is just a string like other strings that we have seen so far.
- So, this string is well-formed if we can derive it from a grammar.

A Simplified Grammar for If-Statement

Example 24

Construct a grammar to produce if-statements like:

```
if(Condition){Statement}
```

Simplified Requirements

1. Condition: only one condition containing '>' or '<' or '=' symbols
   - e.g. "x<5", "5<x", "x<y", "y>2", "x=3", ...
2. Statements: only one Java assignment-statement
   - e.g. "**x=y*2+3;**", "**y=1+x;**", ...
3. Identifiers: only x or y.
4. Arithmetic operators: * , +

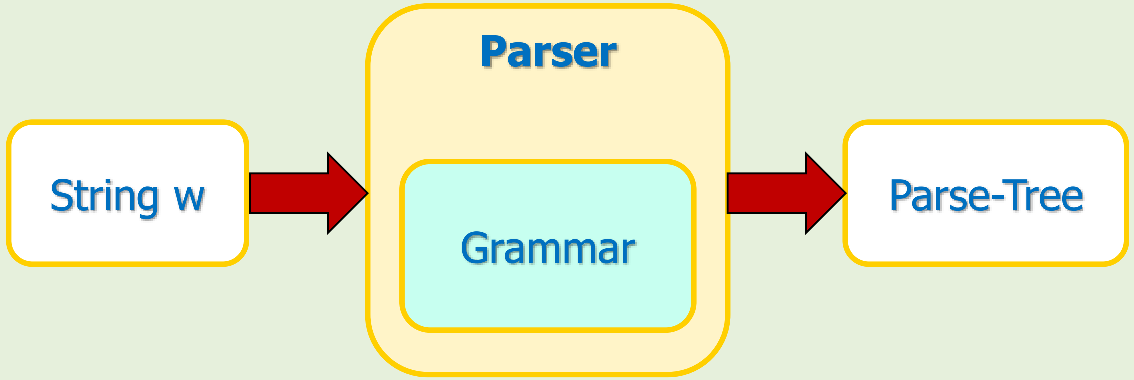
Solution

On the whiteboard!

Note
The provided grammar on the whiteboard is just for getting some idea.
It is neither efficient nor practical!

For Parsing, What Do We Need?

1. We need a grammar.
2. We need a program called "parser" to transform the strings to a data structure called "PARSE-TREE".



- Note that real compilers are more complicated than this.
- I'll show you the components of Java compiler later.

References

1. Linz, Peter, "An Introduction to Formal Languages and Automata, 5th ed.," Jones & Bartlett Learning, LLC, Canada, 2012
2. Michael Sipser, "Introduction to the Theory of Computation, 3rd ed.," CENGAGE Learning, United States, 2013
   ISBN-13: 978-1133187790
3. The ELLCC Embedded Compiler Collection, available at: http://ellcc.org/