CS 6041
Theory of Computation

Context-free language

Kun Suo

Computer Science, Kennesaw State University

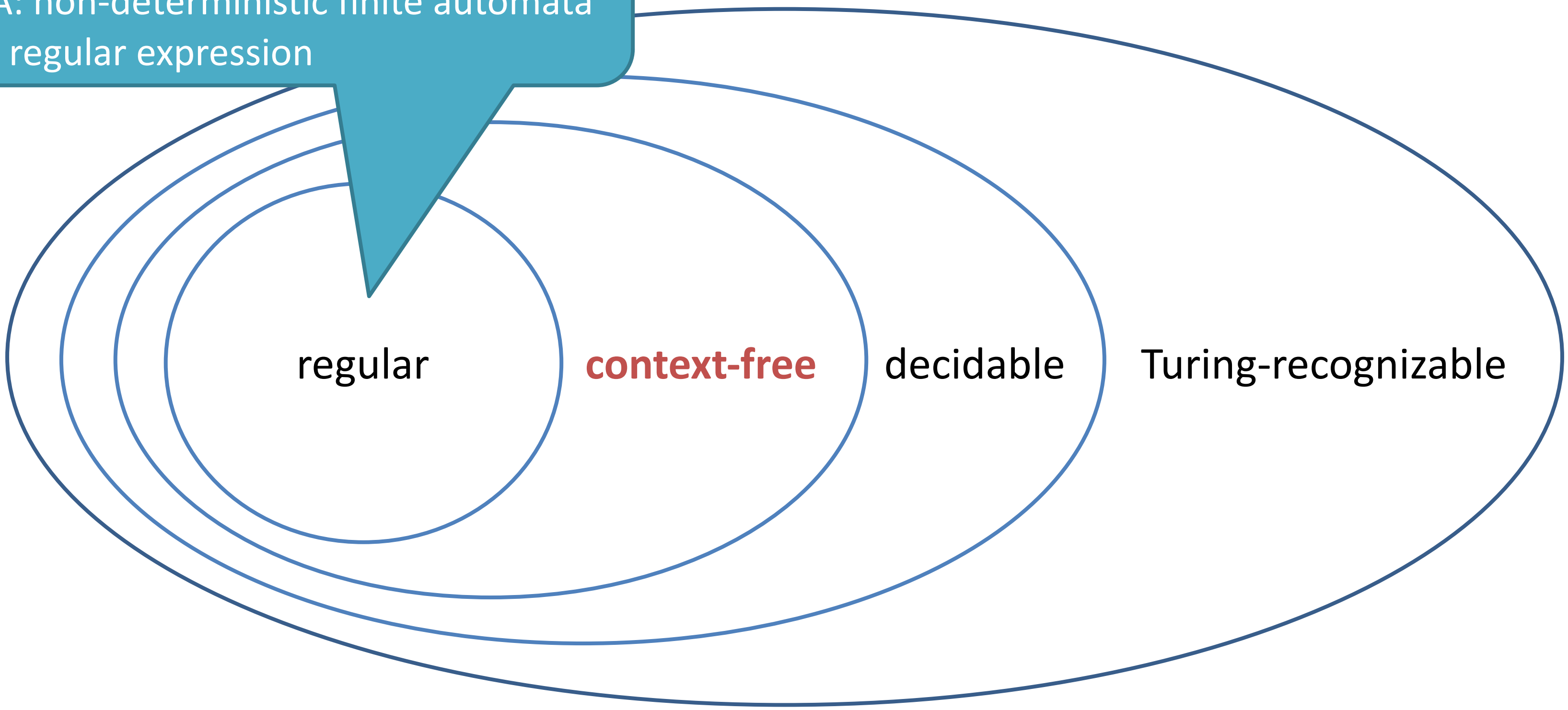
https://kevinsuo.github.io/

Outline

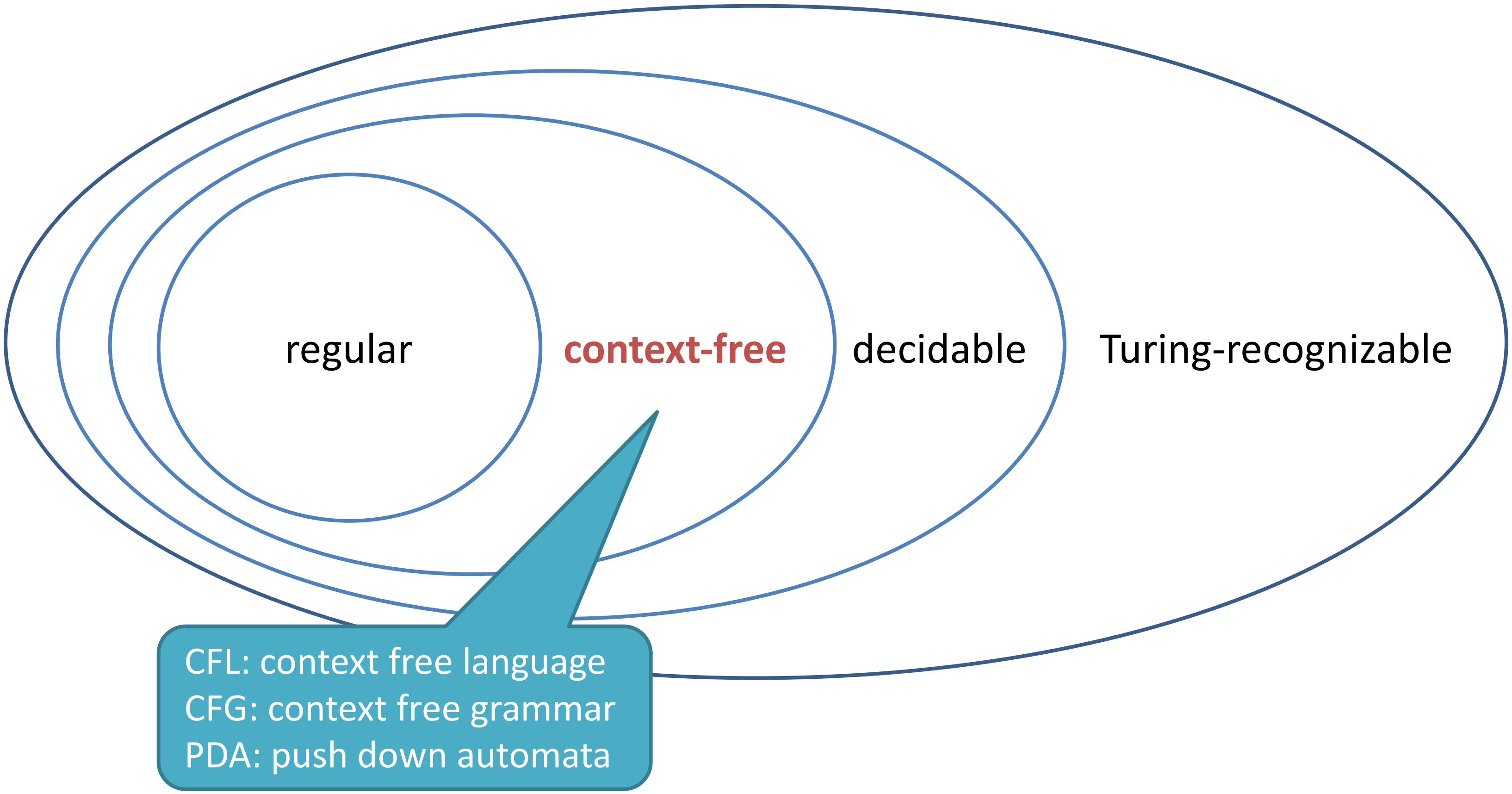
- Context-free language
  - Context-free language and grammar
  - Parse tree
  - Definition of CFG
- Design CFG
  - Example
  - Ambiguity
  - Leftmost derivation

Context-free language

Regular language
DFA: deterministic finite automata
NFA: non-deterministic finite automata
RE: regular expression



Context-free language



Context Free Grammar

- Example, G_1

3 substitution rules (productions)

$$A \rightarrow 0A1$$
$$A \rightarrow B$$
$$B \rightarrow \#$$

Context Free Grammar

- Example, G_1

Variable:
A, B

$A \rightarrow 0A1$

$A \rightarrow B$

$B \rightarrow \#$

Context Free Grammar

- Example, G_1

Start variable: A

$$A \rightarrow 0A1$$
$$A \rightarrow B$$
$$B \rightarrow \#$$

Context Free Grammar

- Example, G_1

$$A \rightarrow 0A1$$
$$A \rightarrow B$$
$$B \rightarrow \#$$

Terminals:
0, 1, #

Context Free Grammar

- Example, G_1

Variable:
A, B

Start variable:
A

3 substitution rules (productions)

$$A \rightarrow 0A1$$
$$A \rightarrow B$$
$$B \rightarrow \#$$

Terminals:
0, 1, #

$$A \Rightarrow 0A1 \Rightarrow 00A11 \Rightarrow 000A111 \Rightarrow 000B111 \Rightarrow 000\#111$$

Context Free Grammar

- The sequence of substitutions to obtain a string is called a ***derivation***

Grammar G_1: $A \rightarrow 0A1$

$A \rightarrow B$

$B \rightarrow \#$

Rule: $\rightarrow$

$$
\begin{aligned}
A &\Rightarrow 0A1 \\
&\Rightarrow 00A11 \\
&\Rightarrow 000A111 \\
&\Rightarrow 000B111 \\
&\Rightarrow 000\#111
\end{aligned}
$$

derivation : $\Rightarrow$

The language of G_1:

$L(G_1)=\{ 0^n\#1^n \mid n\geq 0 \}$

Abbreviating the CFGs

- Grammar G_1:

  $A \rightarrow 0A1$

  $A \rightarrow B$

  $B \rightarrow \#$

- Abbreviation of G_1 :

  G_1: $A \rightarrow 0A1 \mid B$

  $B \rightarrow \#$

Question

- context-free grammar G.

$R \rightarrow XRX \mid S$

$S \rightarrow aTb \mid bTa$

$T \rightarrow XTX \mid X \mid \varepsilon$

$X \rightarrow a \mid b$

1. What are the variables of G?	**R, X, S, T**
2. What are the terminals of G?	**a,b, ε**
3. Which is the start variable of G?	**R**

Question

- context-free grammar G.

$R \rightarrow XRX \mid S$

$S \rightarrow aTb \mid bTa$

$T \rightarrow XTX \mid X \mid \varepsilon$

$X \rightarrow a \mid b$

1. Give three strings in L(G). **ab, ba, aab**

2. Give three strings not in L(G). **a, b, ε**

Question

- context-free grammar G.

$R \rightarrow XRX \mid S$

$S \rightarrow aTb \mid bTa$

$T \rightarrow XTX \mid X \mid \varepsilon$

$X \rightarrow a \mid b$

1.T => aba	**False**
2.T=>* aba	**True**
3.T=>T	**False**
4.XXX =>* aba	**True**
5.X=>* aba	**False**

Question

- CFG: $S \to SS+ \mid SS* \mid a$
- How to generate string aa+a*

$$
\begin{aligned}
S &\Rightarrow SS* \\
&\Rightarrow SS+S* \\
&\Rightarrow aS+S* \\
&\Rightarrow aa+S* \\
&\Rightarrow aa+a*
\end{aligned}
$$

Question

- Describe what language it generates based on CFG: $S \rightarrow 0S1 \mid 01$

- {w| w has the same number of 0s and 1s, and the 0s are ahead of 1s}

Question

- Describe what language it generates based on CFG: $S \rightarrow aSbS \mid bSaS \mid \epsilon$

- {w| w has the same number of as and bs}

Parse tree

A

- Grammar G_1:

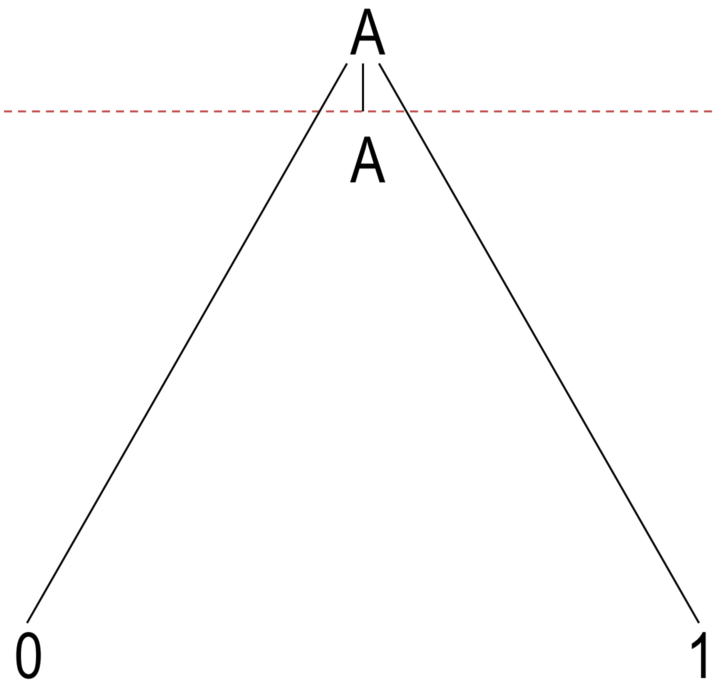
  $A \rightarrow 0A1$

  $A \rightarrow B$

  $B \rightarrow \#$

- Derivation: A
- Parse tree

Parse tree



- Grammar G_1:

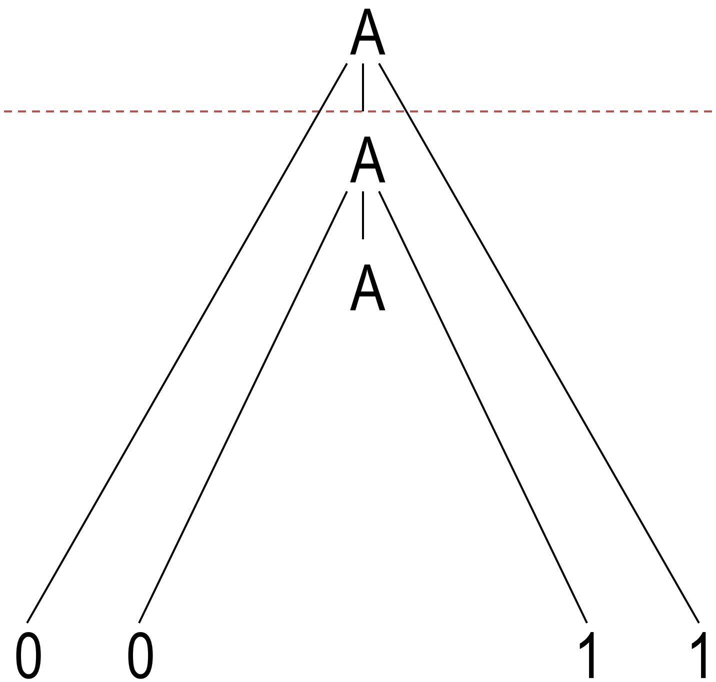
  $A \rightarrow 0A1$

  $A \rightarrow B$

  $B \rightarrow \#$

- Derivation: $A \Rightarrow 0A1$
- Parse tree

Parse tree



- Grammar G_1:

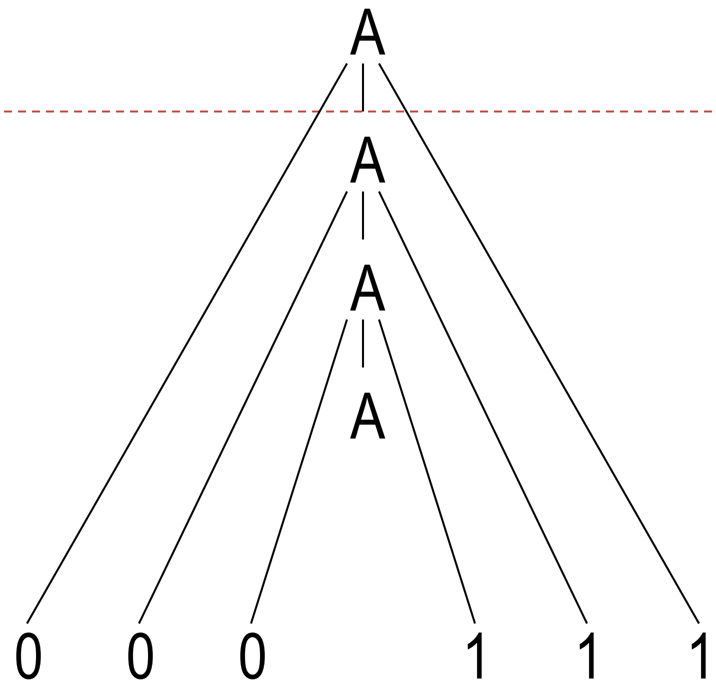
  $A \rightarrow 0A1$

  $A \rightarrow B$

  $B \rightarrow \#$

- Derivation: $A \Rightarrow 0A1 \Rightarrow 00A11$
- Parse tree

Parse tree



- Grammar G_1:

  $A \rightarrow 0A1$

  $A \rightarrow B$

  $B \rightarrow \#$

- Derivation: $A \Rightarrow 0A1 \Rightarrow 00A11$

  $\Rightarrow 000A111$

- Parse tree

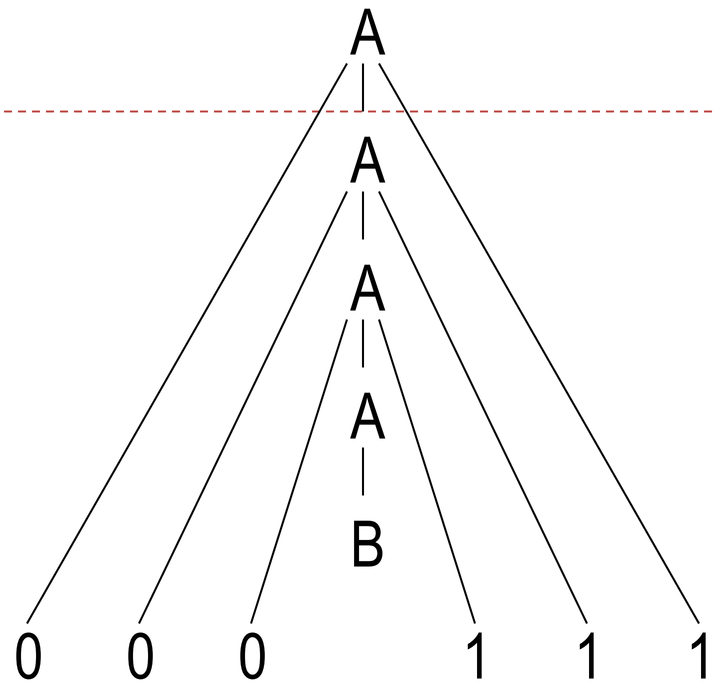
Parse tree

- Grammar G_1:

  $A \rightarrow 0A1$

  $A \rightarrow B$

  $B \rightarrow \#$

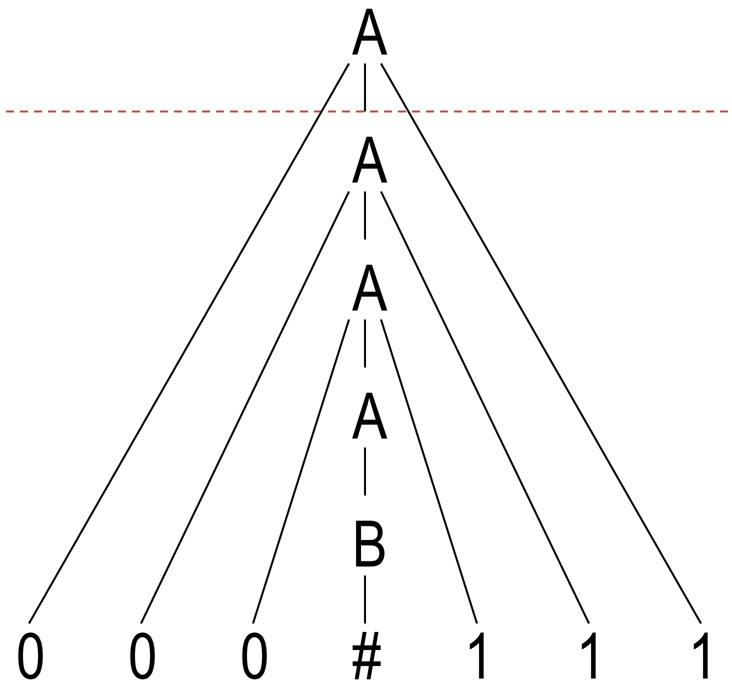


- Derivation: $A \Rightarrow 0A1 \Rightarrow 00A11$

  $\Rightarrow 000A111 \Rightarrow 000B111$

- Parse tree

Parse tree



- Grammar G_1:

  $A \rightarrow 0A1$

  $A \rightarrow B$

  $B \rightarrow \#$

- Derivation: $A \Rightarrow 0A1 \Rightarrow 00A11$

  $\Rightarrow 000A111 \Rightarrow 000B111 \Rightarrow 000\#111$

- Parse tree

The language of grammar

- Grammar G_1:

  $A \rightarrow 0A1$

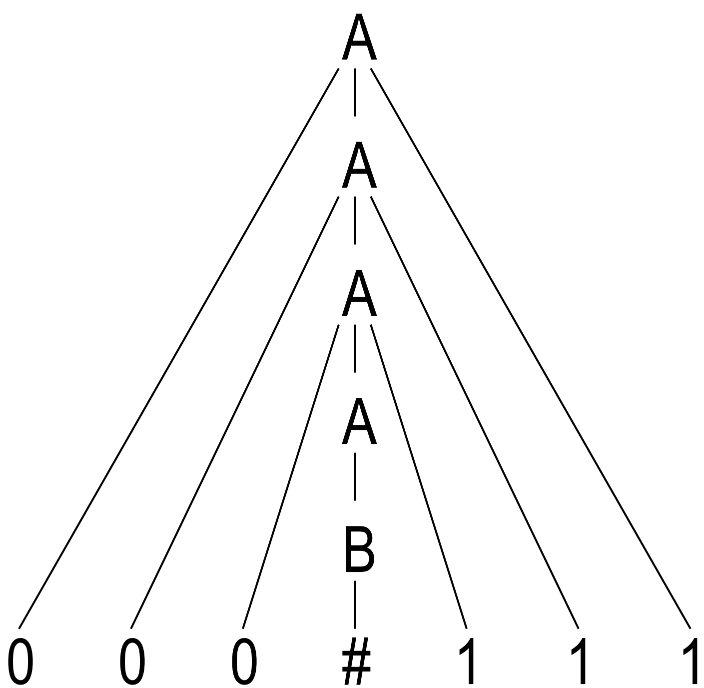
  $A \rightarrow B$

  $B \rightarrow \#$

- The language of G_1:

  $L(G_1)=\{\ 0^n\#1^n \mid n \geq 0\ \}$

- Context-free language
  - Languages generated by context-free grammars



000#111

Definition of context-free grammar

- Context-free grammar is a 4-tuple $G=(V,\Sigma,R,S)$,

  1) V: finite variable set

  2) Σ: finite terminal set

  3) R: finite rule set

     ($A \rightarrow w$, $w \in (V \cup \Sigma)^*$)

  4) $S \in V$: start variable

Example

- Grammar G_1:

  $A \rightarrow 0A1$

  $A \rightarrow B$

  $B \rightarrow \#$

- $G_1 = ($

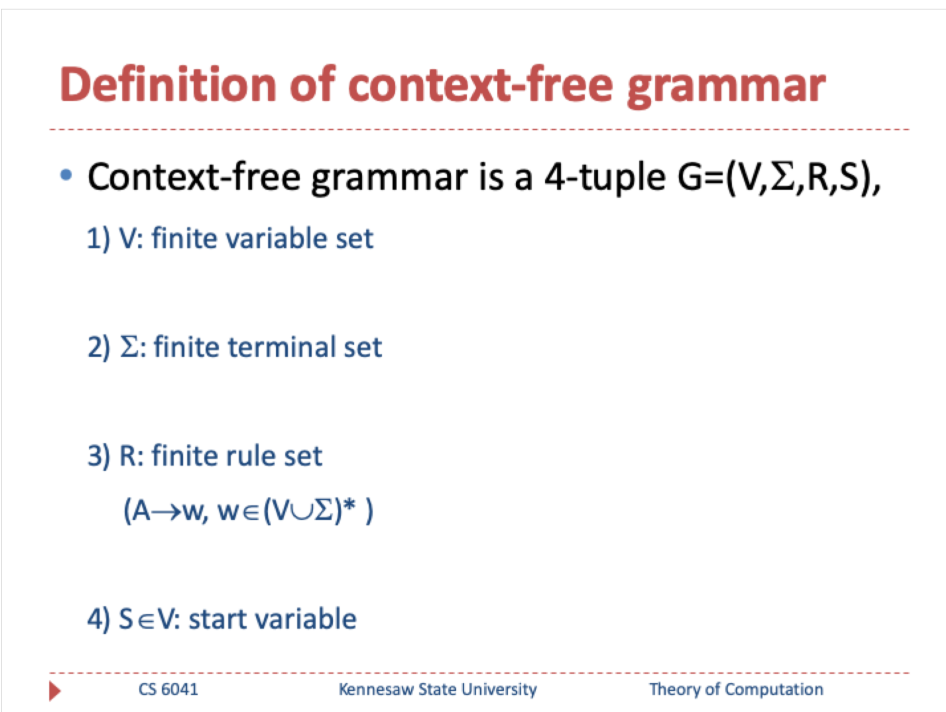
  $\{A,B\},$

  $\{0,1,\#\},$

  $\{A \rightarrow 0A1, A \rightarrow B, B \rightarrow \#\},$

  A

  $)$



Definition of context-free grammar

- Context-free grammar is a 4-tuple $G=(V,\Sigma,R,S)$,

  1) V: finite variable set

  2) Σ: finite terminal set

  3) R: finite rule set

     ($A \rightarrow w$, $w \in (V \cup \Sigma)^*$)

  4) $S \in V$: start variable

Example

- Grammar G_1:

  S -> S+S | S*S | a

- G_1 = (

  {S},

  {a,+,*},

  {S -> S+S | S*S | a},

  S

)

Definition of context-free grammar

- Context-free grammar is a 4-tuple G=(V,Σ,R,S),

  1) V: finite variable set

  2) Σ: finite terminal set

  3) R: finite rule set

     ($A \rightarrow w$, $w \in (V \cup \Sigma)^*$)

  4) $S \in V$: start variable

Definition of context-free grammar

- Yield
  - If $A \rightarrow w$ is a rule of the grammar, we say that uAv ***yields*** uwv
- Derive
  - u ***derives*** v ($u \Rightarrow v$), if $u \Rightarrow u_1 \Rightarrow u_2 \Rightarrow \ldots \Rightarrow u_k \Rightarrow v$
- The language of grammar
  - $L(G)=\{ w \in \Sigma^* \mid S \Rightarrow^* w \}$
- Context-free language (CFL)
  - The language of CFG

Question: how to derive it?

- $G_3=(\{S\},\{a,b\},R,S)$, R is

  $\{ S \rightarrow aSb \mid SS \mid \varepsilon \}$ $\qquad S \Rightarrow abab$?

S

$\Rightarrow SS$

$\Rightarrow aSbS$

$\Rightarrow abS$

$\Rightarrow abaSb$

$\Rightarrow abab$

Question: how to derive it?

- $G_3=(\{S\},\{a,b\},R,S)$, R is

$$\{ S \rightarrow aSb \mid SS \mid \varepsilon \}$$

$S \Rightarrow aaabbb$?

S

$\Rightarrow aSb$

$\Rightarrow aaSbb$

$\Rightarrow aaaSbbb$

$\Rightarrow aaabbb$

Question: how to derive it?

- G_3=({S},{a,b},R,S), R is

$$\{ S \rightarrow aSb \mid SS \mid \varepsilon \}$$

$S \Rightarrow aababb$?

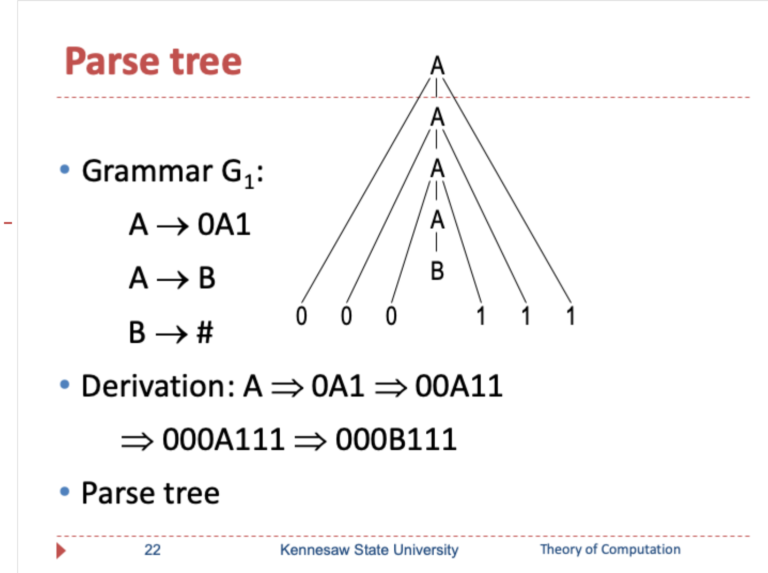
S

$\Rightarrow$ aSb

…. //follow by $S \Rightarrow abab$

$\Rightarrow$ aababb

Example of Parse tree



- $G_4=(V,\Sigma,R,E)$,

  $V=\{E, T, F\}$,

  $\Sigma=\{ a, +, \times, (,) \}$,

  $R=\{$

  $$E \to E + T \mid T$$

  $$T \to T \times F \mid F$$

  $$F \to (E) \mid a$$

  $\}$

Parse tree of a+a×a

- $G_4=(V,\Sigma,R,E)$,

  V={E, T, F},

  Σ={ a, +, ×, (,) },

  R={

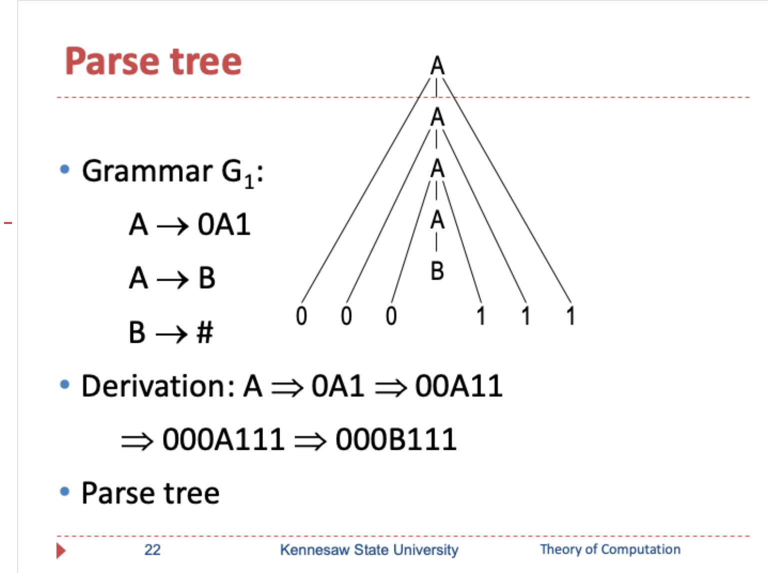
  $$E \to E + T \mid T$$

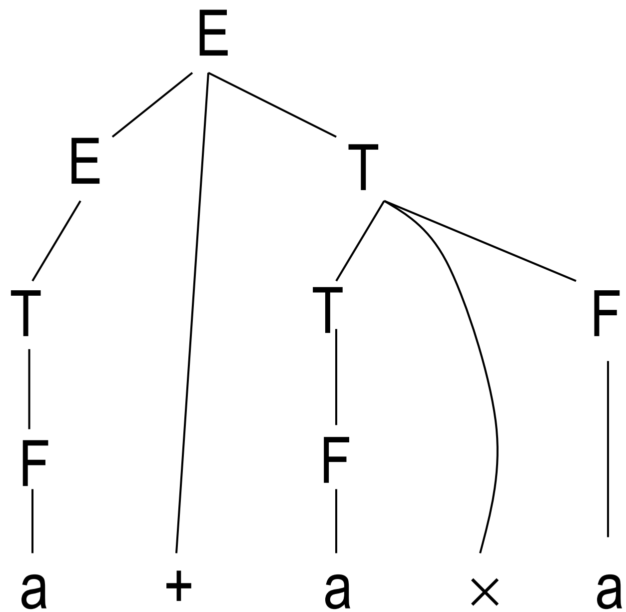
  $$T \to T \times F \mid F$$

  $$F \to (E) \mid a$$

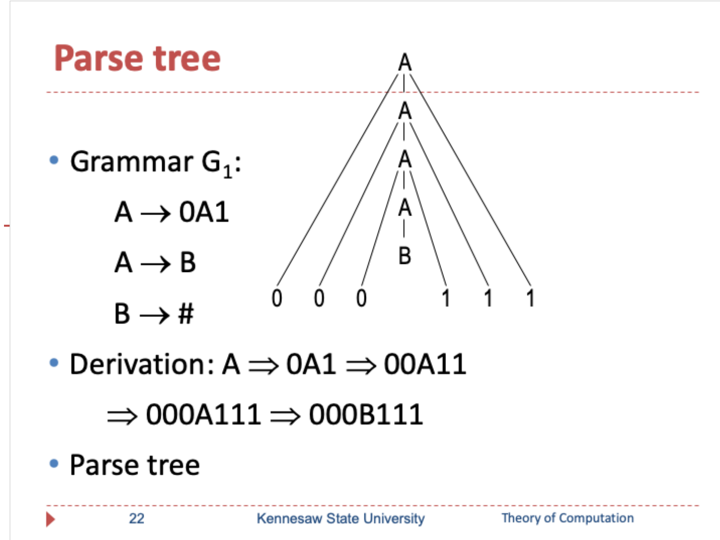
  }

  E





Parse tree of (a+a)×a


Parse tree

- Grammar G_1:
  - $A \rightarrow 0A1$
  - $A \rightarrow B$
  - $B \rightarrow \#$
- Derivation: $A \Rightarrow 0A1 \Rightarrow 00A11$
  $\Rightarrow 000A111 \Rightarrow 000B111$
- Parse tree

- $G_4=(V,\Sigma,R,E)$,

  $V=\{E, T, F\}$,

  $\Sigma=\{ a, +, \times, (,) \}$,

  $R=\{$

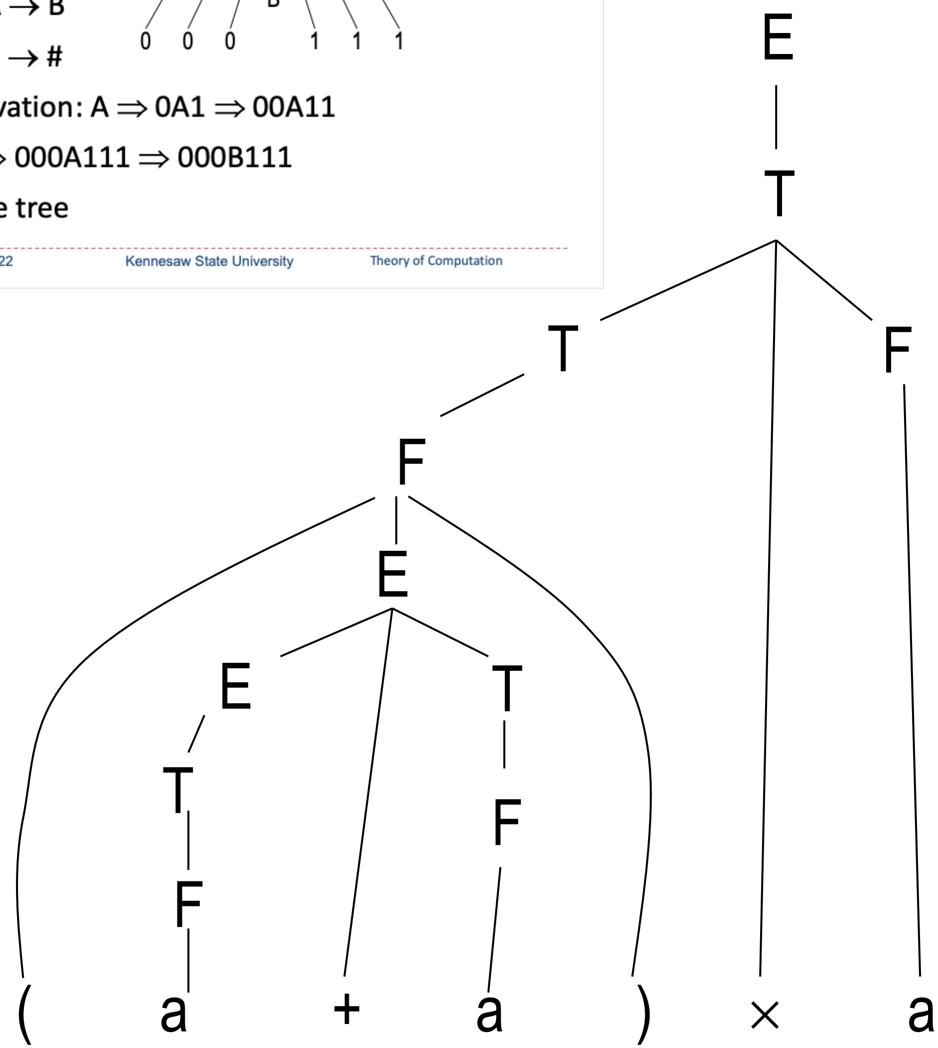
  $E \rightarrow E + T \mid T$

  $T \rightarrow T \times F \mid F$

  $F \rightarrow (E) \mid a$

  $\}$

  E



Outline

- Context-free language
  - Context-free language and grammar
  - Parse tree
  - Definition of CFG
- Design CFG
  - Example
  - Ambiguity
  - Leftmost derivation

Design context-free grammar

- Design CFG for $\{w \mid w=0^n1^n \text{ or } w=1^n0^n, n\geq 0\}$

Design context-free grammar

- Design CFG for $\{w|w=0^n1^n \text{ or } w=1^n0^n, n\geq 0\}$
  - Design CFG for $\{w|w=0^n1^n, n\geq 0\}$
  - Design CFG for $\{w|w=1^n0^n, n\geq 0\}$

Design context-free grammar

- Design CFG for $\{w|w=0^n1^n \text{ or } w=1^n0^n, n\geq 0\}$
  - Design CFG for $\{w|w=0^n1^n, n\geq 0\}$

Generating same number of 0 and 1
Generating 0 before 1

01 0011 000111 00..011..1

Design context-free grammar

- Design CFG for $\{w|w=0^n1^n \text{ or } w=1^n0^n, n\geq 0\}$
  - Design CFG for $\{w|w=0^n1^n, n\geq 0\}$

Generating same number of 0 and 1
Generating 0 before 1

01   0011   000111   00..011..1

Design context-free grammar

- Design CFG for $\{w|w=0^n1^n \text{ or } w=1^n0^n, n\geq 0\}$
  - Design CFG for $\{w|w=0^n1^n, n\geq 0\}$

Generating same number of 0 and 1
Generating 0 before 1

01 0011 000111 00..011..1

Design context-free grammar

- Design CFG for $\{w|w=0^n1^n \text{ or } w=1^n0^n, n\geq 0\}$
  - Design CFG for $\{w|w=0^n1^n, n\geq 0\}$

Generating same number of 0 and 1
Generating 0 before 1

01 0011 000111 00..01..11

$S \rightarrow 0S1$

Design context-free grammar

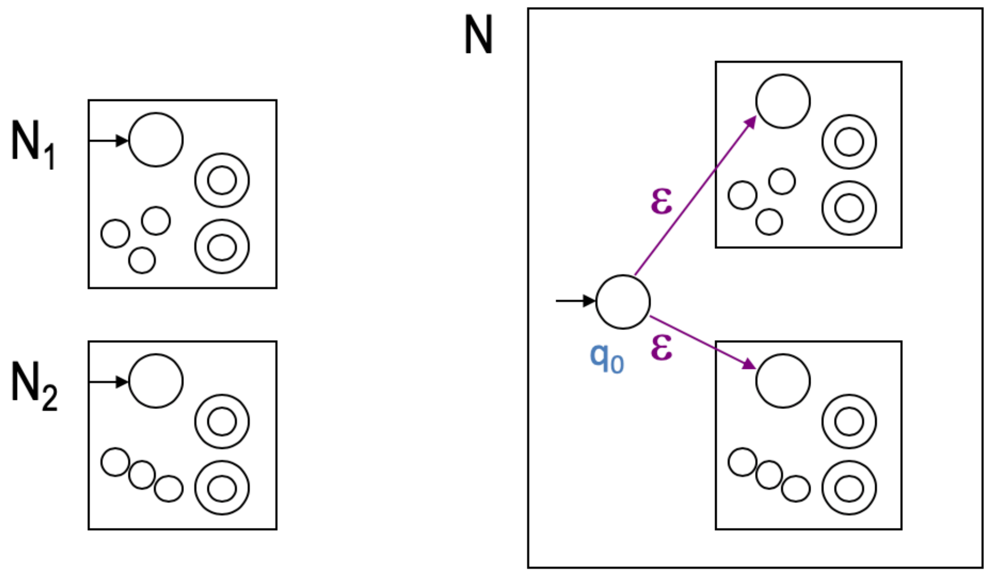
- Design CFG for $\{w|w=0^n1^n \text{ or } w=1^n0^n, n\geq 0\}$
  - Design CFG for $\{w|w=0^n1^n, n\geq 0\}$
    - $G_1=(\{S\},\{0,1\}, \{S\rightarrow 0S1, S\rightarrow \varepsilon\}, S)$
  - Design CFG for $\{w|w=1^n0^n, n\geq 0\}$
    - $G_2=(\{S\},\{0,1\}, \{S\rightarrow 1S0, S\rightarrow \varepsilon\}, S)$

Design context-free grammar

- Design CFG for $\{w|w=0^n1^n \text{ or } w=1^n0^n, n\geq 0\}$
  - Design CFG for $\{w|w=0^n1^n, n\geq 0\}$
    - $G_1=(\{S_1\},\{0,1\},\{S_1\rightarrow 0S_11, S_1\rightarrow\varepsilon\},S_1)$
  - Design CFG for $\{w|w=1^n0^n, n\geq 0\}$
    - $G_2= (\{S_2\},\{0,1\},\{S_2\rightarrow 1S_20, S_2\rightarrow\varepsilon\},S_2)$

Design context-free grammar

- Design CFG for $\{w \mid w=0^n1^n \text{ or } w=1^n0^n, n\geq 0\}$
  - Design CFG for $\{w \mid w=0^n1^n, n\geq 0\}$
    - $G_1=(\{S_1\},\{0,1\},\{S_1\rightarrow 0S_11, S_1\rightarrow \varepsilon\},S_1)$
  - Design CFG for $\{w \mid w=1^n0^n, n\geq 0\}$
    - $G_2= (\{S_2\},\{0,1\},\{S_2\rightarrow 1S_20, S_2\rightarrow \varepsilon\},S_2)$



Design context-free grammar

- Design CFG for $\{w \mid w=0^n1^n \text{ or } w=1^n0^n, n\geq 0\}$
  - Design CFG for $\{w \mid w=0^n1^n, n\geq 0\}$
    - $G_1=(\{S_1\},\{0,1\},\{S_1\rightarrow 0S_11, S_1\rightarrow \varepsilon\},S_1)$
  - Design CFG for $\{w \mid w=1^n0^n, n\geq 0\}$
    - $G_2= (\{S_2\},\{0,1\},\{S_2\rightarrow 1S_20, S_2\rightarrow \varepsilon\},S_2)$

  - $G=(\{S,S_1,S_2\},\{0,1\},$

    $\{ S\rightarrow S_1,\ S\rightarrow S_2,\ S_1\rightarrow 0S_11,\ S_1\rightarrow \varepsilon,\ S_2\rightarrow 1S_20,\ S_2\rightarrow \varepsilon\},$

    $S)$

Combine CFG into one

- General case:

  Add $S \rightarrow S_1 \mid S_2 \mid \ldots \mid S_k$

  - S is the new start variable
  - $S_1, S_2, \ldots, S_k$ are original start variables

- CFL is closure on the Union operation

Operation on languages

	RL: DFA/NFA/RE	CFL: CFG/PDA	TM
Union	close	close	?
Concatenation	close	?	?
Star	close	?	?
Complement	close	?	?
Boolean operation	close	?	?

Design CFG for languages

- Design CFG is much difficult than designing an automata for language

- Basic idea:

  1. divide CFL into small parts
  2. design CFG for each small part
  3. combine them together

Design CFG for languages

- Design CFG is much difficult than designing an automata for language

- Other ideas:

  1. Simulate the regular expressions
  2. Look for a pattern from example strings
  3. ...

Design CFG for languages

- $L=\{w \mid w \text{ has at least three 1s}\}$, $\Sigma = \{0,1\}$

$$\Sigma^* 1\, \Sigma^*\, 1\, \Sigma^*\, 1\, \Sigma^*$$

Design CFG for languages

- L={w| w has at least three 1s}, $\Sigma = \{0,1\}$

$\Sigma^* 1 \, \Sigma^* 1 \, \Sigma^* 1 \, \Sigma^*$

S → R1R1R1R

Design CFG for languages

- L={w| w has at least three 1s}, Σ = {0,1}

$\Sigma^* 1 \Sigma^* 1 \Sigma^* 1 \Sigma^*$

$S \rightarrow R1R1R1R$

$R \rightarrow 0R$

$R \rightarrow 1R$

$R \rightarrow \varepsilon$

Design CFG for languages

- L={w| w has odd length}, $\Sigma = \{0,1\}$

$$\Sigma(\Sigma\,\Sigma)^*$$

Design CFG for languages

- L={w| w has odd length}, Σ = {0,1}

$\Sigma(\Sigma\,\Sigma)^*$

$$\left.\begin{array}{l} S \rightarrow 0 \\ S \rightarrow 1 \end{array}\right\}$$

Design CFG for languages

- L={w| w has odd length}, Σ = {0,1}

$\Sigma(\Sigma\,\Sigma)^*$

S → 0

S → 1

S → S00

S → S01

S → S10

S → S11

Design CFG for languages

- L={w| w has odd length}, $\Sigma = \{0,1\}$

$\Sigma(\Sigma\ \Sigma)^*$

$S \rightarrow 0$

$S \rightarrow 1$

$S \rightarrow S00$

$S \rightarrow S01$

$S \rightarrow S10$

$S \rightarrow S11$

Design CFG for languages

- L={w| w has odd length and the middle symbol is 0}, $\Sigma = \{0,1\}$

0
000
001
100
101
00011
…

Design CFG for languages

- L={w| w has odd length and the middle symbol is 0}, $\Sigma = \{0,1\}$

$S \rightarrow 0$

$S \rightarrow 0S0$

$S \rightarrow 0S1$

$S \rightarrow 1S0$

$S \rightarrow 1S1$

0
000
001
100
101
00011
…

Design CFG for languages

- $L = \{0^n1^n \mid n \geq 0\}$. $\Sigma = \{0,1\}$

ε, 01, 0011, …

$S \rightarrow 0S1 \mid \varepsilon$

Design CFG for languages

- $L = \{0^n 1^{2n} \mid n \geq 0\}$. $\Sigma = \{0,1\}$

ε, 011, 001111, …

$S \rightarrow 0S11 \mid \varepsilon$

Design CFG for languages

- $L = \{00^*11^*\}$. $\Sigma = \{0,1\}$

01, 011, 0011, …

How to design 00^*

How to design 11^*

Design CFG for languages

- $L = \{00^*11^*\}$. $\Sigma = \{0,1\}$

How to design 00^*

$C \rightarrow 0$

$C \rightarrow 0C$

Design CFG for languages

- $L = \{00^*11^*\}$. $\Sigma = \{0,1\}$

How to design 11^*

$D \rightarrow 1$

$D \rightarrow 1D$

Design CFG for languages

- $L = \{00^*11^*\}$. $\Sigma = \{0,1\}$

How to design 00^*

$C \rightarrow 0$

$C \rightarrow 0C$

How to design 00^*11^*

$S \rightarrow CD$

$C \rightarrow 0C \mid 0$

$D \rightarrow 1D \mid 1$

How to design 11^*

$D \rightarrow 1$

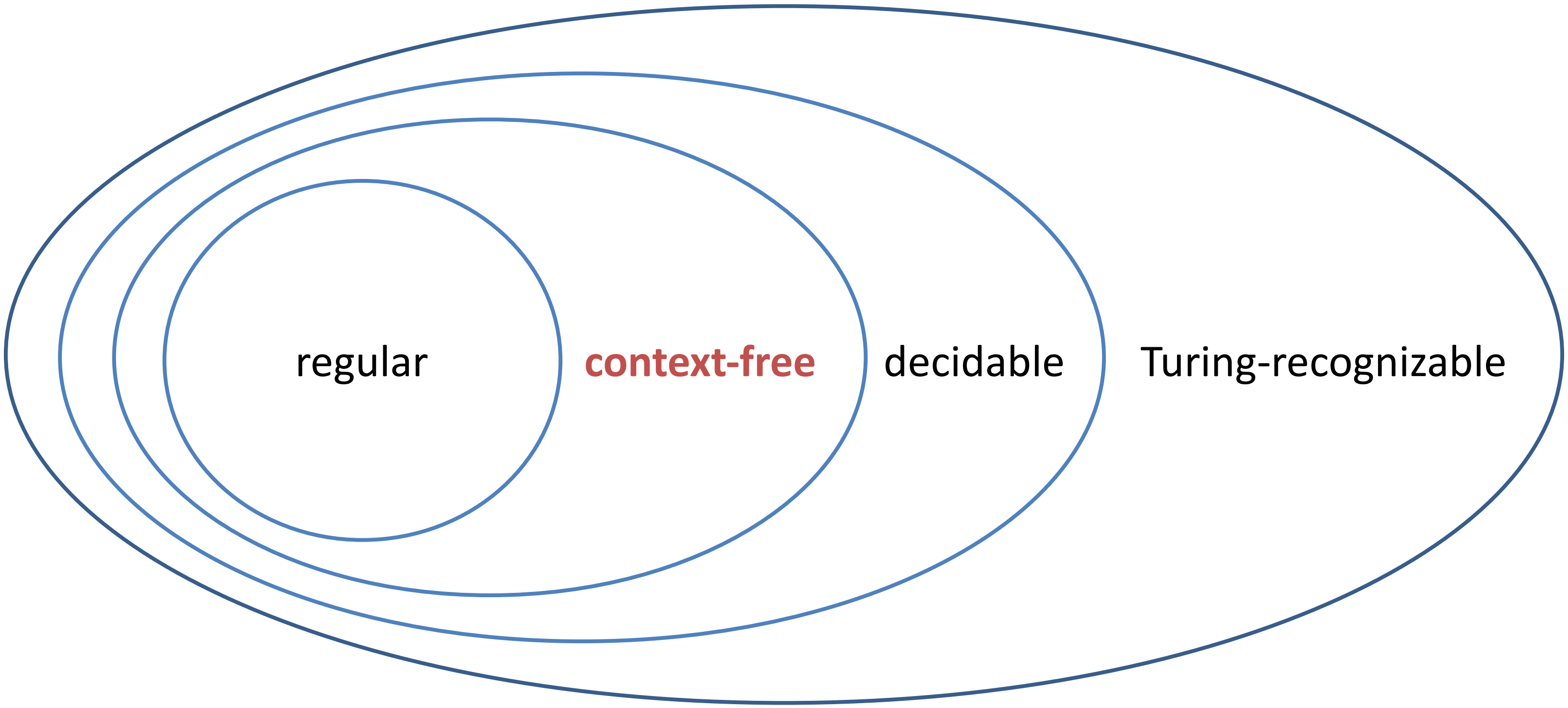
$D \rightarrow 1D$

Design CFG for languages

- L=∅

$S \rightarrow S$

Design CFG for regular languages



Design CFG for regular languages

- Transfer DFA into equivalent CFG

Design CFG for regular languages

- Transfer DFA into equivalent CFG
- Let DFA $M=(Q,\Sigma,\delta,q_0,F)$

  then CFG $G=(V,\Sigma,R,R_0)$

Design CFG for regular languages

- Transfer DFA into equivalent CFG
- Let DFA $M=(Q,\Sigma,\delta,q_0,F)$
  - $Q=\{q_0,q_1,\ldots,q_k\}$,

  then CFG $G=(V,\Sigma,R,R_0)$
  - $V=\{R_0,R_1,\ldots,R_k\}$,

Design CFG for regular languages

- Transfer DFA into equivalent CFG
- Let DFA $M=(Q,\Sigma,\delta,q_0,F)$
  - $Q=\{q_0,q_1,\ldots,q_k\}$,
  - $\delta(q_i,a)=q_j$,

  then CFG $G=(V,\Sigma,R,R_0)$
  - $V=\{R_0,R_1,\ldots,R_k\}$,
  - $R_i \rightarrow aR_j$,

Design CFG for regular languages

- Transfer DFA into equivalent CFG
- Let DFA M= (Q,Σ,δ,q_0,F)
  - $Q=\{q_0,q_1,\ldots,q_k\}$,
  - $\delta(q_i,a)=q_j$,
  - $q_i \in F$

  then CFG $G=(V,\Sigma,R,R_0)$
  - $V=\{R_0,R_1,\ldots,R_k\}$,
  - $R_i \rightarrow aR_j$,
  - $R_i \rightarrow \varepsilon$

Grammar G_1:

$A \rightarrow 0A1$

$A \rightarrow B$

$B \rightarrow \varepsilon$

Design CFG for regular languages

- True/False?

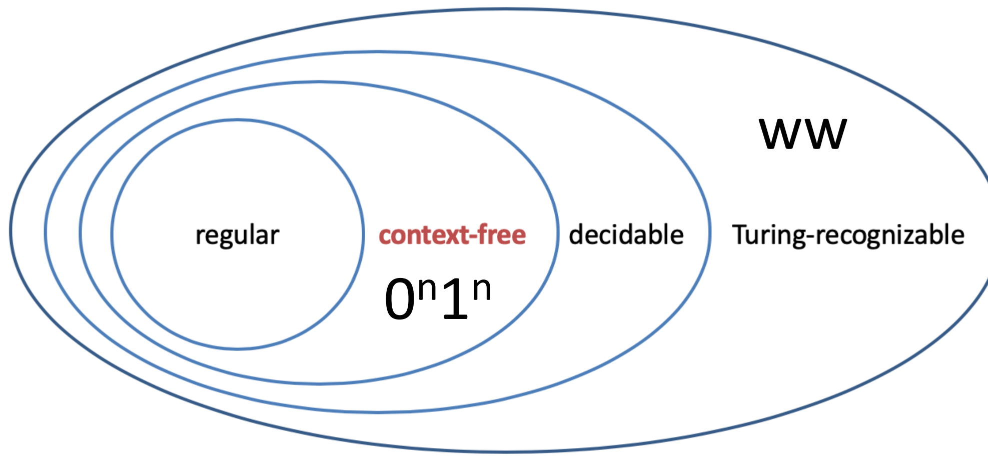
- Every Regular Language is Context-Free
  - T

- For each regular language L there exists a context-free grammar G, such that L = L(G)
  - T

More languages

- 0^n1^n
  - is not regular language, proved by pumping lemma
  - is a context-free language built by CFG

    $R \rightarrow 0R1$, $R \rightarrow \varepsilon$



- ww
  - is not regular language
  - Is not context-free language

Ambiguity

- If a grammar generates the ***same*** string in several ***different*** ways, we say that the string is derived *ambiguously* in that grammar.

- If a grammar generates some string ambiguously, we say that the grammar is ***ambiguous***.

- G_5: $E \to$

  $E+E \mid$

  $E \times E \mid$

  $(E) \mid a$

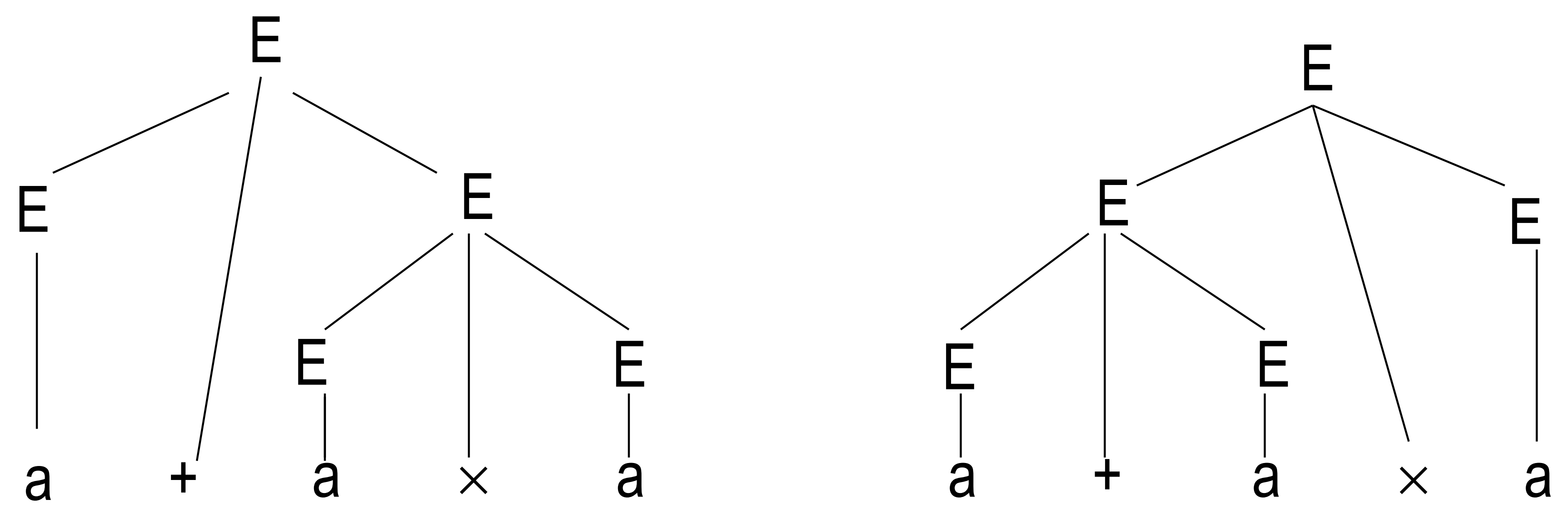
Ambiguity

- G_5: $E \rightarrow$

  $E+E$ |

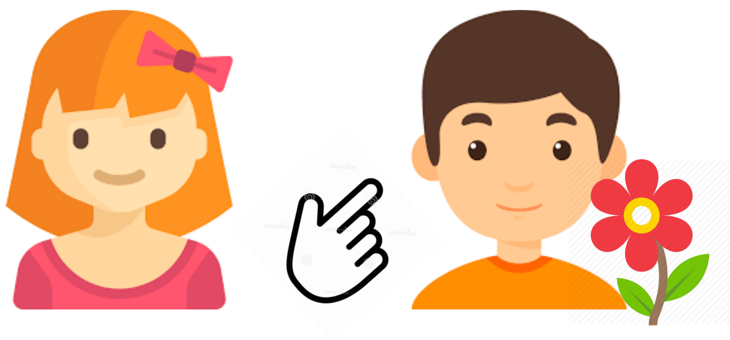
  $E\times E$ |

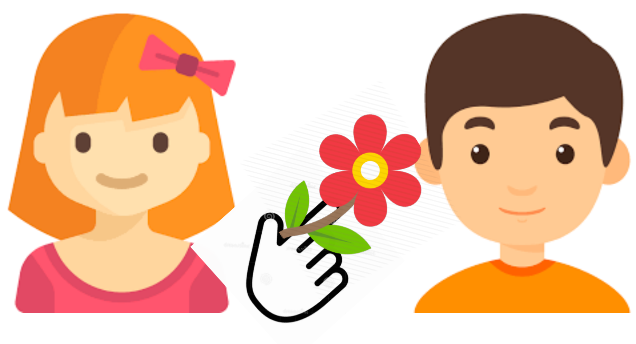
  (E) | a



Ambiguity in real life

- G_2:
- the_girl_touches_the_boy_with_flower





Leftmost derivation

- A derivation of a string w in a grammar G is a ***leftmost derivation*** if at every step the ***leftmost*** remaining variable is the one replaced

- $E \Rightarrow E+E$

  $\Rightarrow a+E$

  $\Rightarrow a+E\times E$

  $\Rightarrow a+a\times E \Rightarrow a+a\times a$

- G_5: $E \rightarrow$

  $E+E \mid$

  $E\times E \mid$

  $(E) \mid a$

Two different leftmost derivation

- E
  $\Rightarrow E+E$
  $\Rightarrow a+E$
  $\Rightarrow a+E\times E$
  $\Rightarrow a+a\times E$
  $\Rightarrow a+a\times a$

- E
  $\Rightarrow E\times E$
  $\Rightarrow E+E \times E$
  $\Rightarrow a+E\times E$
  $\Rightarrow a+a\times E$
  $\Rightarrow a+a\times a$

- G_5: $E \rightarrow$
  $E+E \mid$
  $E\times E \mid$
  $(E) \mid a$

Ambiguity

- A string w is derived ***ambiguously*** in context-free grammar G if it has two or more different ***leftmost derivations***.

- Grammar G is ***ambiguous*** if it generates some string ambiguously.

- Some context-free languages can be generated only by ambiguous grammars. (***inherently ambiguous***)

Inherently ambiguous example

- $\{ 0^i 1^j 2^k \mid i=j \text{ or } j=k \}$
  - $\{ 0^n 1^n 2^m \mid n,m \geq 0 \} \cup$

    $\{ 0^m 1^n 2^n \mid n,m \geq 0 \}$
  - $0^n 1^n 2^n$ can only be generated by ambiguous grammars (due to the language definition)
  - Human languages like English/French/Spanish/Chinese/Japanese/Hindi … are inherently ambiguous

Question

- $G = \{S \to SS, S \to ab, S \to ba, S \to \varepsilon\}$
- G produces all strings with equal number of a's and b's. True or false? Why?

Question

- $G = \{S \rightarrow SS, S \rightarrow ab, S \rightarrow ba, S \rightarrow \varepsilon\}$

- G produces all strings with equal number of a's and b's. True or false? Why?

It can't generate aabb string. So the statement is incorrect.

Question

- $G = \{S \rightarrow SS, S \rightarrow ab, S \rightarrow ba, S \rightarrow \varepsilon\}$

- Is G ambiguous? Why?

Ambiguity

- A string w is derived ***ambiguously*** in context-free grammar G if it has two or more different ***leftmost derivations***.
- Grammar G is ***ambiguous*** if it generates some string ambiguously.
- Some context-free languages can be generated only by ambiguous grammars. (***inherently ambiguous***)

CS 6041 Kennesaw State University Theory of Computation

Question

- G = {$S \to SS$, $S \to ab$, $S \to ba$, $S \to \varepsilon$}
- Is G ambiguous? Why?

There are different LMD's for string abab which can be

$S \Rightarrow \underline{S}S \Rightarrow \underline{S}SS \Rightarrow ab\underline{S}S \Rightarrow abab\underline{S} \Rightarrow abab$

$S \Rightarrow \underline{S}S \Rightarrow ab\underline{S} \Rightarrow abab$

So the grammar is ambiguous.

Question

- True or false?
- There exist CFLs such that all the CFGs generating them are ambiguous.
  - True. Inherently ambiguous.

Question

- True or false?

- An unambiguous CFG always has a unique parse tree for each string of the language generated by it.

    - True. As unambiguous CFG has a unique parse tree for each string of the language generated by it.

Conclusion

- Context-free language
  - Context-free language and grammar
  - Parse tree
  - Definition of CFG
- Design CFG
  - Example
  - Ambiguity
  - Leftmost derivation