CS:314 Fall 2024

Section **04**
Recitation **3**

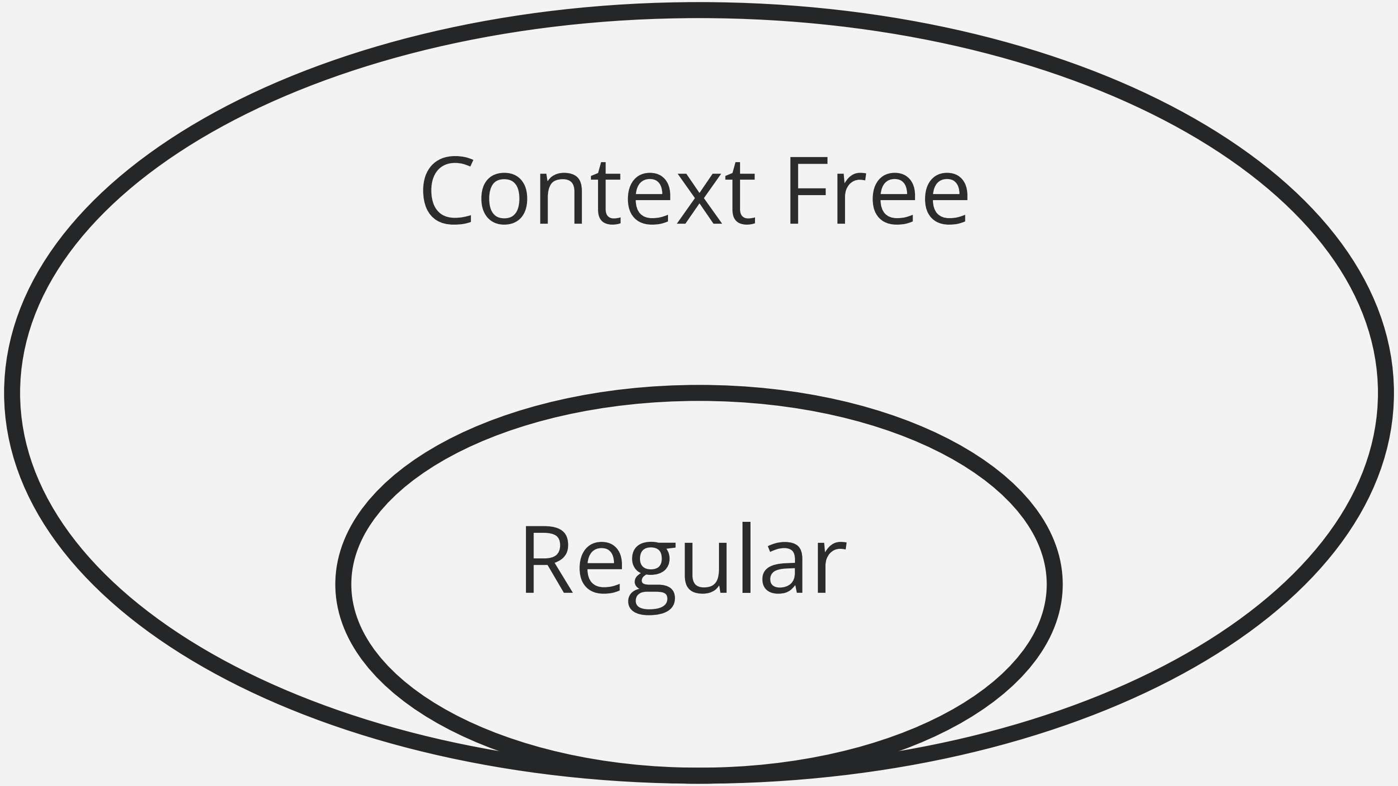
chris.tu@rutgers.edu
Office hours: 2-3pm Thursday CoRE 335

Context–free grammars

- A **context-free grammar** (CFG) G = <T, N, P, S> consists of:
    - A set T of terminal symbols (tokens). Kind of like values.
    - A set N of non-terminal symbols. Kind of like variables.
    - A set P of production (rewrite) rules.
    - A special start symbol S.
- **(Note: Often the start symbol S is the non-terminal symbol on the left hand side of the first rule.)**

- The language L(G) is the set of sentences of terminal symbols that can be derived from the start symbol S.

Context Free Subsumes Regex



Backus–Naur Form

::=

Backus–Naur form is used to describe syntax in programming or formal languages. We use this when writing CFG production rules.

It means the <symbol> on left must be replaced with the expression on the right.

Context-free grammars: Example 1

- Give a context-free grammar for the following language:

  {w | w contains at least three 1’s} with alphabet $\Sigma = \{0,1\}$

Context-free grammars: Example 1

- Give a context-free grammar for the following language:

  {w | w contains at least three 1's} with alphabet Σ = {0,1}

<**S**> ::= <**R**>1<**R**>1<**R**>1<**R**>

<**R**> ::= 0<**R**> | 1<**R**> | ε

Context-free grammars: Example 2

- Give a context-free grammar for the following language:

  {w | the length of w is odd and its middle symbol is 0}

  with alphabet $\Sigma = \{0,1\}$

Context-free grammars: Example 2

- Give a context-free grammar for the following language:

  {w | the length of w is odd and its middle symbol is 0}

  with alphabet Σ = {0,1}

<**S**> ::= 0 | 0<**S**>0 | 0<**S**>1 | 1<**S**>0 | 1<**S**>1

Context-free grammars: Example 3

- Give a context-free grammar for the following language:

  {w | w starts and ends with the same symbol} with alphabet $\Sigma = \{0,1\}$

Context-free grammars: Example 3

- Give a context-free grammar for the following language:

  {w | w starts and ends with the same symbol} with alphabet $\Sigma = \{0,1\}$

  <**S**> ::= 0 | 1 | 0<**R**>0 | 1<**R**>1

  <**R**> ::= 0<**R**> | 1<**R**> | ε

Context-free grammars: Example 4

Suppose we have the following CFG on {0,1}:

<S> ::= 0<F>0

<F> ::= 1<F> | ε

What strings are generated by this CFG?

Context-free grammars: Example 4

Suppose we have the following CFG on {0,1}:

<S> ::= 0<F>0

<F> ::= 1<F> | ε

What strings are generated by this CFG?

- 010, 0110, 01110…
- The language generated by G is $\{01^n0 \mid n \geq 0\}$.

Context-free grammars: Derivation

- Given the following CFG:

<**E**> ::= <**E**>+<**T**> | <**T**>

<**T**> ::= <**T**>*<**F**> | <**F**>

<**F**> ::= <**E**> | i

Here, the start symbol is <E>.

Give a leftmost and rightmost derivation for : i + i * i

Context-free grammars: Derivation

<E>

=>$_L$ <E> + <T>

=>$_L$ <T> + <T>

=>$_L$ <F> + <T>

=>$_L$ i + <T>

=>$_L$ i + <T> * <F>

=>$_L$ i + <F> * <F>

=>$_L$ i + i * <F>

=>$_L$ i + i * i

<E>

=>$_R$ <E> + <T>

=>$_R$ <E> + <T> * <F>

=>$_R$ <E> + <T> * i

=>$_R$ <E> + <F> * i

=>$_R$ <E> + i * i

=>$_R$ <T> + i * i

=>$_R$ <F> + i * i

=>$_R$ i + i * i

CFG:

<E> ::= <E>+<T> | <T>

<T> ::= <T>*<F> | <F>

<F> ::= <E> | i

Context-free grammars: Derivation

- **The corresponding parse tree for the left-most derivation:**

<E>

=>$_L$ <E> + <T>

=>$_L$ <T> + <T>

=> <F> + <T>

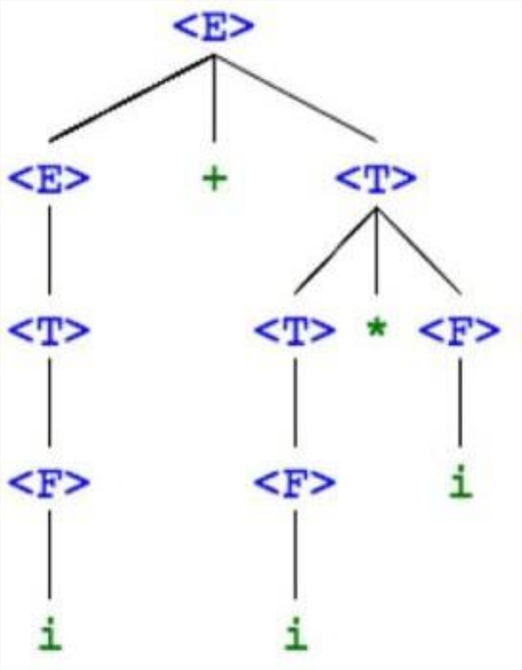
=>$_L$ i + <T>

=>$_L$ i + <T> * <F>

=>$_L$ i + <F> * <F>

=>$_L$ i + i * <F>

=>$_L$ i + i * i



Context-free grammars: Ambiguity

A grammar G is ambiguous **iff** there exists a $\mathbf{w \in L(G)}$ such that there are:

- two distinct parse trees for w, or
- two distinct leftmost derivations for w, or
- two distinct rightmost derivations for w.

Context-free grammars: Ambiguity

Given the following CFG G:

<E> ::= <E> <O> <E> | <E> | i

<O> ::= + | *

Suppose **w = i + i * i**.

We will show that G is ambiguous by showing that there are two distinct rightmost derivations.

Context-free grammars: Ambiguity

<E> ::= <E> <O> <E> | <E> | i

<O> ::= + | *

<E>
=>$_R$ <E> <O> <E>
=>$_R$ <E> <O> i
=>$_R$ <E> * i
=>$_R$ <E> <O> <E> * i
=>$_R$ <E> <O> i * i
=>$_R$ <E> + i * i
=>$_R$ i + i * i

<E>
=>$_R$ <E> <O> <E>
=>$_R$ <E> <O> <E> <O> <E>
=>$_R$ <E> <O> <E> <O> i
=>$_R$ <E> <O> <E> * i
=>$_R$ <E> <O> i * i
=>$_R$ <E> + i * i
=>$_R$ i + i * i

There are two distinct rightmost derivations for i + i * i, so the grammar is ambiguous.

Context-free grammars: Ambiguity

To deal with ambiguity, impose precedence.
the operation with higher precedence should be lower/deeper in the parsing tree

Context-free grammars: Ambiguity

To deal with ambiguity, impose precedence.

<S> ::= <E>

<E> ::= <E> + <E> | <T>

<T> ::= <T> * <T> | i

Take **w = i + i * i.**

<S>

=>$_R$<E>

=>$_R$<E> + <E>

=>$_R$<E> + <T>

=>$_R$<E> + <T> * <T>

=>$_R$<E> + <T> * i

=>$_R$<E> + i * i

=>$_R$<T> + i * i

=>$_R$i + i * i