GA.2110-003 Programming Languages - Fall 2022

Recitation Class 1

Elaine Li   Nisarg Patel

New York University

Course Information and Resources

Course web page (general info, syllabus, etc.)
https://cs.nyu.edu/wies/teaching/pl-fa22/

Brightspace (announcements, course-related discussions, grades)
https://brightspace.nyu.edu/d2l/home/222148

Github (class notes and code, homework submission)
https://github.com/nyu-pl-fa22

Important Dates

Class Meetings

Tue 4:55-6:55pm in Silv 408
Office hours: Thomas Wies, Wed 4-5pm (60FA 403)

Recitations

Fri 5:55-6:45pm (two parallel sessions: 60FA 150 and online)
Office hours: Elaine Li, Fri 1-2pm (60FA 418)
Office hours: Nisarg Patel, Mon 11am-12pm (60FA 418)

Graders

- Vaibhav Mavi
- Rajat Narlawar
- Adithya Viswanathan

Grader office hours will be announced soon.

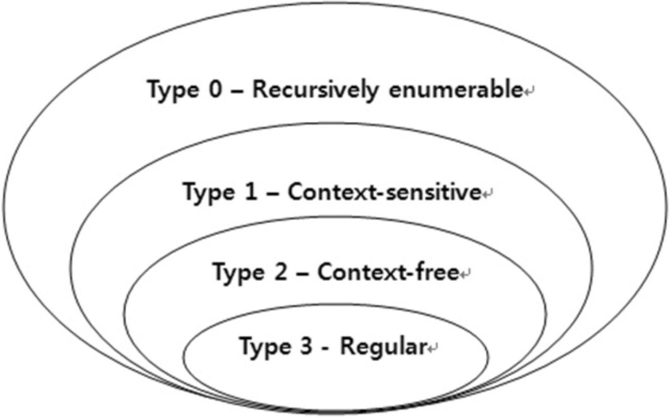
Midterm Exam

Tue Nov 1, 4:55-6:55pm in Silv 408

Final Exam

Tue Dec 20 (preliminary date)

Hierarchy of Languages



Grammars

A *grammar* G is a tuple (Σ, N, P, S), where:

- N is a set of *non-terminal* symbols
- $S \in N$ is a distinguished non-terminal: the *root* or *start* symbol
- Σ is a set of *terminal* symbols, also called the *alphabet*. We require Σ to be disjoint from N (i.e. $\Sigma \cap N = \emptyset$).
- P is a set of rewrite rules (productions) of the form:

$$\mathrm{ABC} \ldots \rightarrow \mathrm{XYZ} \ldots$$

  where $\mathrm{A}, \mathrm{B}, \mathrm{C}, \mathrm{X}, \mathrm{Y}, \mathrm{Z}$ are terminals and non-terminals.

Any sequence consisting of terminals and non-terminals is called a *word*.

The *language* defined by a grammar is the set of words containing *only* terminal symbols that can be generated by applying the rewriting rules starting from S.

Grammars Example

- $N = \{S\}$
- $S = S$
- $\Sigma = \{a, b\}$
- P consists of the following rules:
  - $S \rightarrow aSb$
  - $S \rightarrow \epsilon$

Some sample derivations:

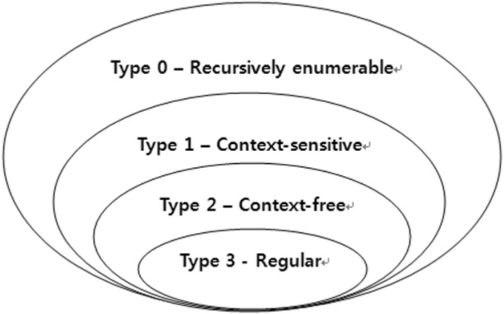
- $S \rightarrow \epsilon$
- $S \rightarrow aSb \rightarrow ab$
- $S \rightarrow aSb \rightarrow aaSbb \rightarrow aabb$

The Chomsky hierarchy

- Regular grammars (Type 3)
  - All productions must be of one of the following shapes:

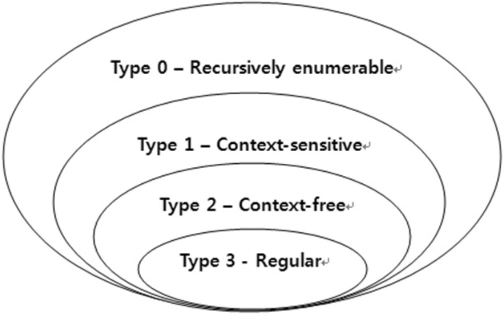
$$X ::= \epsilon \qquad X ::= a \qquad X ::= Y \qquad X ::= aY$$

  - Recognizable by finite state automaton
  - Used in *lexers*
- Context-free grammars (Type 2)
  - All productions have a single non-terminal on the left
  - Right side of productions can be any word
  - Recognizable by non-deterministic pushdown automaton
  - Used in *parsers*



The Chomsky hierarchy

- Context-sensitive grammars (Type 1)
  - Each production is of the form $\alpha A \beta \rightarrow \alpha \gamma \beta$,
  - A is a non-terminal, and α, β, γ are arbitrary words (α and β may be empty, but not γ)
  - Recognizable by linear bounded automaton
- Recursively-enumerable grammars (Type 0)
  - No restrictions
  - Recognizable by turing machine



Regular expressions

An alternate way of describing a regular language over an alphabet Σ is with *regular expressions*.
We say that a regular expression R denotes the language $[\![R]\!]$ (recall that a language is a set of words).
Regular expressions over alphabet Σ:

- $\emptyset$ denotes $\emptyset$
- ϵ denotes $\{\epsilon\}$
- a character x, where $x \in \Sigma$, denotes $\{x\}$
- (sequencing) a sequence of two regular expressions RS denotes $\{\alpha\beta \mid \alpha \in [\![R]\!], \beta \in [\![S]\!]\}$
- (alternation) $R \mid S$ denotes $[\![R]\!] \cup [\![S]\!]$
- (Kleene star) R^* denotes the set of words which are concatenations of zero or more words from $[\![R]\!]$
- parentheses are used for grouping
- $R^? \equiv \epsilon \mid R$
- $R^+ \equiv RR^*$

Regular expressions exercises

Given the alphabet $\{a, b, c\}$, give a regular expression for the language that contains the string aaa at some point.

Regular expressions exercises

Given the alphabet $\{a, b, c\}$, give a regular expression for the language that contains the string aaa at some point.

Answer : $(a \mid b \mid c)^* aaa(a \mid b \mid c)^*$

Regular expressions exercises

Given the alphabet $\{a, b, c\}$, give a regular expression for the language that must end with either the string a or the string bc.

Regular expressions exercises

Given the alphabet $\{a, b, c\}$, give a regular expression for the language that must end with either the string a or the string bc.

Answer : $(a \mid b \mid c)^*(a \mid bc)$

Regular expressions exercises

Given the alphabet $\{a, b, c\}$, give a regular expression for the language that does not contain the string ab.

Regular expressions exercises

Given the alphabet $\{a, b, c\}$, give a regular expression for the language that does not contain the string ab.

Answer : $(b \mid c)^*(ac(b \mid c)^* \mid a)^*$

Regular expressions questions

Closed under:

- union
- intersection
- complementation

Regular expressions questions

Closed under:

- union ✓ (using $|$ operator)
- intersection ✓ (through automata)
- complementation ✓ (through automata)

CFG exercises

Given the alphabet $\{a, b, c\}$, give a CFG for the language of Palindromes.

CFG exercises

Given the alphabet $\{a, b, c\}$, give a CFG for the language of Palindromes.

Answer : $P \rightarrow \; aPa \mid bPb \mid a \mid b \mid \epsilon$

CFG exercises

Given the alphabet $\{a, b, c\}$, give a CFG for the language $\{w \in \Sigma^* \mid w \text{ has equal number of } a\text{'s and } b\text{'s}\}$.

CFG exercises

Given the alphabet $\{a, b, c\}$, give a CFG for the language $\{w \in \Sigma^* \mid w \text{ has equal number of } a\text{'s and } b\text{'s}\}$.

Answer : $S \rightarrow \; aSbS \mid bSaS \mid \epsilon$

CFG (bonus) exercises

Translate earlier regular expressions to CFGs!

CFG questions

Give a language that can be expressed using a CFG but not regular expression.

CFG questions

Give a language that can be expressed using a CFG but not regular expression.

Answer: All the CFG languages we discussed today!

CFG questions

Give a language that can be expressed using a CFG but not regular expression.

Answer: All the CFG languages we discussed today!

Closed under:

- union ✓
- intersection X
- complementation X

CFG questions

Give a language that can be expressed using a CFG but not regular expression.

Answer: All the CFG languages we discussed today!

Closed under:

- union $\checkmark$
- intersection X
- complementation X

Languages $\{a^n b^n c^m \mid n, m \geq 0\}$ and $\{a^m b^n c^n \mid n, m \geq 0\}$ can be expressed as CFGs.

However, $\{a^n b^n c^m \mid n, m \geq 0\} \cap \{a^m b^n c^n \mid n, m \geq 0\} = \{a^n b^n c^n \mid n \geq 0\}$ cannot be expressed as a CFG.

CSG exercises

Give a CSG for the language $\{a^n b^n c^n \mid n \geq 0\}$.

CSG exercises

Give a CSG for the language $\{a^n b^n c^n \mid n \geq 0\}$.

Answer :

$$\begin{aligned} S &\to aTbc \mid abc \mid \epsilon \\ T &\to aTbU \mid abU \\ Ub &\to bU \\ Uc &\to cc \end{aligned}$$