An overview

SITE : http://www.sir.blois.univ-tours.fr/~mirian/

What about all these kinds of languages?

Chomsky-Schützenberger hierarchy

- Chomsky-Schützenberger hierarchy is a containment hierarchy of classes of formal grammars that generate formal languages

Type-0 grammars

- **Type-0 grammars** (unrestricted grammars) include all formal grammars.
- **They generate exactly all languages that can be recognized by a TM**. These languages are also known as the **recursively enumerable languages**.
- Note that this is different from the recursive languages which can be decided by an always-halting Turing machine.

Type-1 grammars

- **Type-1 grammars** (**context-sensitive grammars**) generate the **context-sensitive languages**.
- These grammars have rules of the form $\alpha A \beta \rightarrow \alpha \gamma \beta$ with A a nonterminal and α, β and γ strings of terminals and nonterminals.
- The strings α and β may be empty, but γ must be nonempty.
- The rule $S \rightarrow \epsilon$ is allowed if S does not appear on the right side of any rule.
- The languages described by these grammars are exactly all languages that can be recognized by a linear bounded automaton (a **nondeterministic Turing machine whose tape is bounded by a constant times the length of the input**).

Type-2 grammars

- Type-2 grammars (**context-free grammars**) generate the **context-free languages**
- These are defined by rules of the form $A \rightarrow \gamma$ with A a nonterminal and γ a string of terminals and nonterminals.
- These languages are exactly all languages that can be recognized by a **non-deterministic pushdown automaton**.
- Context free languages are the theoretical basis for the syntax of most programming languages

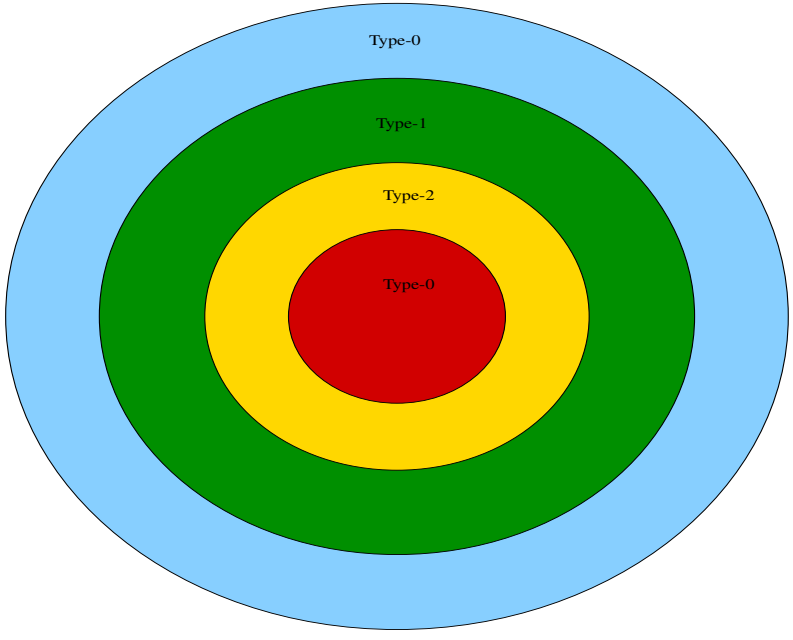
Type-3 grammars

- Type-3 grammars (regular grammars) generate the regular languages.
- Such a grammar restricts its rules to a single nonterminal on the left-hand side and a right-hand side consisting of a single terminal, possibly followed (or preceded, but not both in the same grammar) by a single nonterminal.
- The rule $S \rightarrow \epsilon$ is also here allowed if S does not appear on the right side of any rule.
- These languages are exactly all languages that can be decided by a **finite state automaton**.
- Additionally, this family of formal languages can be obtained by **regular expressions**
- Regular languages are commonly used to define search patterns and the lexical structure of programming languages.

Summary

Grammar	Language	Automaton	Production rule
Type-0	Recursively enumerable	Turing machine	$\alpha \rightarrow \beta$(no restrictions)
Type-1	Context-sensitive	Linear-bounded non-deterministic TM	$\alpha A \beta \rightarrow \alpha \gamma \beta$
Type-2	Context-free	Non-deterministic PDA	$A \rightarrow \gamma$
Type-3	Regular	Finite state automaton	$A \rightarrow a$ and $A \rightarrow aB$

Chomsky hierarchy



- The set of grammars corresponding to recursive languages is not a member of this hierarchy
- We have proper inclusions!

Differences among all these automata

- Basis: the FSA
- PDA is a FSA with a stack!
- TM is a FSA with a tape and the possibility of writting and moving over this tape

Important remarks about regular languages

Closure properties of regular languages

- The **union** of two regular languages is regular
- The **intersection** of two regular languages is regular
- The **complement** of a regular language is regular
- The **difference** of two regular languages is regular
- The **reversal** of a regular language is regular
- The **closure** of a regular language is regular
- The **concatenation** of regular languages is regular
- A **homomorphism** (substitution of strings of symbols) of a regular language is regular
- The **inverse homomorphism** of a regular language is regular

Important remarks about CFL

Closure properties of context-free languages

The CFL **are closed** under:

- **Union**
- **Concatenation**
- **Closure** (star)
- **Homomorphism**
- **Inverse homomorphism**
- **Reversal**

The CFL **are not closed** under:

- **Intersection**
- **Difference**