1. Automata and Languages

1.1. Regular Languages

Def. A (deterministic) **finite automaton** is a 5-tuple $(Q, \Sigma, \delta, q_o, F)$, where

1. Q is a finite set called the **states**,
2. Σ is a finite set called the **alphabet**,
3. $\delta : Q \times \Sigma \to Q$ is the **transition function**,
4. $q_o \in Q$ is the **start state**, and
5. $F \subseteq Q$ is the set of **accepted/final states**

Def. A language is called a **regular language** if some finite automaton recognizes it.
Ex. A language that has strings ending with 0; A language that has strings with substring 010.

The class of regular languages is closed under the following operations:

1. **Union**: $A \cup B = \{x \mid x \in A or x \in B\}$.
2. **Concatenation**: $A \circ B = \{xy \mid x, y \in A, B\}$.
3. **Star**: $A^* = \{x_1, x_2, \ldots, x_k \mid k \geq 0$ and each $x_i \in A\}$.

Determinism- When the machine is in a given state and reads the next input symbol, the next state is unique and already determined.
Nondeterminism- Several choices may exist for the next state at any point.

Def. A **nondeterministic finite automaton** is the same 5-tuple, except

$$\delta : Q \times \Sigma_\epsilon \to P(Q)$$

Theorem. Every NFA has an equivalent DFA.
Corollary. A language is regular if and only if some NFA recognizes it.

Def. R is a **regular expression** if R is

1. a for some a is the alphabet Σ,
2. ε,
3. ϕ,
4. $(R_1 \cup R_2)$, where R_1 and R_2 are regular expressions,
5. $(R_1 \circ R_2)$, where R_1 and R_2 are regular expressions, or
6. (R_1^*), where R_1 is a regular expression.

Theorem. A language is regular iff some regular expression describes it.
Note. Every regular language can be converted into an NFA.
Note. DFAs can be reduced to minimized DFAs.

Nonregular languages are those that cannot be recognized by DFAs.
Ex. $\{0^n 1^n | n \geq 0\}$.

Theorem. Pumping Lemma If A is a regular language, then $\exists$ a number p (the pumping length) where if any $s \in A$ s.t. $|a| \geq p$, then s can be divided into 3 pieces, $s = xyz$, s.t.

1. for each $i \geq 0, xy^i z \in A$,
2. $|y| > 0$, and
3. $|xy| \leq p$.

Pumping Lemma can help differentiate between regular and nonregular languages.

1.2. Context-free grammars

Def. A CFG is a 4-tuple (V, Σ, R, S), where

1. V is a finite set called the variables,
2. Σ is a finite set, disjoint from V , called the terminals,
3. R is a finite set of rules, with each rule being a variable and a string of variables and terminals, and
4. $S \in V$ is the start variable.

A left hand derivation exists for every string $s \in L(CFG)$.
The language of the grammar is $\{w \in \Sigma^* | S \Rightarrow^* w\}$.
Grammars can be unambiguous (i.e. each string has a unique LH derivation) or ambiguous (i.e. string has two or more LH derivations), or even inherently ambiguous.
Note. A context-free grammar is in Chomsky normal form if every rule is of the form

$$A \to BC$$
$$A \to a$$
$$S \to \varepsilon$$

Def. A (nondeterministic) **pushdown automaton** is a 6-tuple $(Q, \Sigma, \varsigma, \delta, q_0, F)$, where

1. Q is the set of states,
2. Σ is the input alphabet,
3. ς is the stack alphabet,
4. $\delta : Q \times \Sigma_\varepsilon \times \varsigma_\varepsilon \Rightarrow P(Q \times \varsigma_\varepsilon)$ is the transition function,
5. $q_0 \in Q$ is the start state, and
6. $F \subseteq Q$ is the set of accept states.

Theorem. A language A is a CFL iff $\exists$ a PDA P s.t. $L(P) = A$.
Ex. $\{0^i 1^j 2^k | i \neq j or j \neq k, i, j, k \geq 0\} \in CFL$.

Def. A **deterministic PDA** is the same 5-tuple, except

$$\delta : Q \times \Sigma_\varepsilon \times \varsigma_\varepsilon \Rightarrow Q \times \varsigma_\varepsilon \cup \{\phi\}.$$

s.t. $\forall q \in Q, a \in \Sigma, b \in \varsigma$ we have exactly one of the following to be non-empty:

$$\delta(q, a, b), \delta(q, a, \varepsilon), \delta(a, \varepsilon, b), \delta(\varepsilon, \varepsilon, \varepsilon)$$

Theorem. A language A is a DCFL iff $\exists$ a DPDA D s.t. $L(D) = A$.
Ex. $\{0^n 1^n | n \geq 0\} \in DCFL \setminus RL$.

Note. PDAs can either accept by final state or by emptying the stack.
Note. Every DPDA has an equivalent DPDA that always reads the entire input string.
Note. If $A = N(P)$,i.e. accepted by emptying the stack, then A is prefix-free.

Note. CFLs are closed under union, intersections and complement.
Note. DCFLs are closed under complement.

Def. A **DCFG** is a CFG such that every valid string has a forced handle. $\exists$ a similar pumping lemma for non-context-free languages where the string can be divided into five pieces instead, $s = uv^i xy^i z$.