CS 530: Theory of Computation

Based on Sipser (second edition): Notes on regular languages

Definition 1 (Alphabet) *A* ***alphabet*** *is a finite set of objects called symbols.*

Definition 2 (String) *A* ***string*** *is a finite list of symbols from an alphabet.*

Definition 3 (Length) *If w is a string over Σ, the* ***length*** *of w, written $|w|$, is the number of symbols that it contains.*

Definition 4 (Empty string) *The string of length zero is called the* ***empty string*** *and is written ϵ.*

Definition 5 (Reverse) *The* ***reverse*** *of w, written $w^{\mathcal{R}}$, is the string obtained by writing w in the opposite order(i.e., $w_n w_{n-1} \dots w_1$).*

Definition 6 (Substring) *String z is a* ***substring*** *of w if z appears consecutively within w.*

Definition 7 (Prefix, Suffix) *String z is a* ***prefix*** *of w if for all $1 \leq i \leq |z|$ we have that $z_i = w_i$. (z is a substring of w "starting" with w_1).*

String z is a ***suffix*** *of w if for all $0 \leq i \leq |z| - 1$ we have that $z_{|z|-i} = w_{|w|-i}$. (z is a substring of w "ending" with $w_{|w|}$).*

Definition 8 (Concatenation) ***Concatenation*** *is an operation that sticks two or more strings together, producing a string.*

Definition 9 (Lexicographic ordering) *The* ***lexicographic ordering*** *of strings is the same as the familiar dictionary ordering, except that shorter strings precede longer strings.*

Definition 10 (Language) *A* ***language*** *over an alphabet* Σ *is a set of strings with all their symbols from* Σ.

Definition 11 (Complement) *The* ***complement*** *of a language* A *over an alphabet* Σ *is denoted by* $\bar{A}$, *and is the set of all the strings over* Σ *that are not in* A.

Definition 12 (Definition 1.5, page 35) *A* ***finite automaton*** *(also called DFA) is a* 5*-tuple* $(Q, \Sigma, \delta, q_0, F)$, *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. $q_0 \in Q$ *is the* ***start state*** *and*
5. $F \subseteq Q$ *is the* ***set of accept states***.

Definition 13 (M **accepts** ω**)** *When machine* M *receives an input string* ω, *it processes that string and produces an output. The output is either* ***accept*** *or* ***reject***. *The processing begins in* M*'s start state. The automaton receives the symbols from the input string one by one from left to right. After reading each symbol,* M *moves from one state to another along the transition that has that symbol as its label. When it reads the last symbol,* M *produces its output. If the output is accept, then* M ***accepts*** ω.

More formal: Let M *be the DFA* $(Q, \Sigma, \delta, q_0, F)$ *and* w *be a string over the alphabet* Σ. *Then we say that* M ***reaches state*** q ***after processing*** $w = a_1 a_2 \cdots a_m$, *where each* a_i *is a member of* Σ, *if a sequence of states* $r_0, r_1, \ldots, r_m$ *exists in* Q *with the following three conditions:*

1. $r_0 = q_0$,
2. $r_{i+1} = \delta(r_i, a_{i+1})$, *and*
3. $r_m = q$.

We say that M ***accepts*** $w = a_1 a_2 \cdots a_m$, *if* M *reaches state* q *after processing* w *and* $q \in F$.

Definition 14 (M **recognizes language** A**)** *If* A *is the set of all strings that machine* M *accepts, we say that* A *is the* ***language of machine*** M *and write* $L(M) = A$. *We say that* M ***recognizes A***.

Definition 15 (Definition 1.16, page 40) *A language is called a* ***regular language*** *if some finite automaton recognizes it.*

Definition 16 (Definition 1.23, page 44) *Let* A *and* B *be languages. We define the regular operations* ***union***, ***concatenation***, *and* ***star*** *as follows.*

- ***Union***: $A \cup B = \{x | x \in A \text{ or } x \in B\}$.
- ***Concatenation***: $A \circ B = \{xy | x \in A \text{ and } y \in B\} = \{w \mid \exists x \in A\, \exists y \in B\ w = xy\}$.
- ***Star***: $A^* = \{x_1 x_2 ... x_k | k \geq 0 \text{ and each } x_i \in A\} = \{w \mid \exists k \geq 0\ \exists x_1 \in A\ \exists x_2 \in A \ldots \exists x_k \in A\ w = x_1 x_2 \cdots x_k\}$. *Note that when* $k = 0$, *we obtain that* $\epsilon \in A^*$ *(for any language* A*).*

Theorem 17 (Theorem 1.25, page 45) *The class of regular languages is closed under the union operation. In other words, if* A_1 *and* A_2 *are regular languages, so is* $A_1 \cup A_2$.

Theorem 18 (Theorem 1.26, page 47) *The class of regular languages is closed under the concatenation operation. In other words, if* A_1 *and* A_2 *are regular languages, so is* $A_1 \circ A_2$.

Definition 19 (Definition 1.37, page 53) *A **nondeterministic finite automaton** (also called NFA) is a* 5*-tuple* $(Q, \Sigma, \delta, q_0, F)$*, where*

1. Q *is a finite set of states,*
2. Σ *is a finite alphabet,*
3. $\delta : Q \times \Sigma_\epsilon \to \mathcal{P}(Q)$ *is the transition function,*
4. $q_0 \in Q$ *is the start state, and*
5. $F \subseteq Q$ *is the set of accept states.*

Here $\Sigma_\epsilon = \Sigma \cup \{\epsilon\}$*.*

Definition 20 (page 54) *Let* N *be the NFA* $(Q, \Sigma, \delta, q_0, F)$ *and* w *be a string over the alphabet* Σ*. Then we say that* N ***can reach state*** q ***after processing*** $w \in \Sigma^*$ *if there exist a way to write* $w = a_1 a_2 \cdots a_m$*, where each* a_i *is a member of* Σ_ϵ *and there exists a sequence of states* $r_0, r_1, \ldots, r_m$ *exists in* Q *with the following three conditions:*

1. $r_0 = q_0$*,*
2. $r_{i+1} \in \delta(r_i, a_{i+1})$*, and*
3. $r_m = q$*.*

Then we say that N ***accepts*** w *if* N *can reach state* q *after processing* w *and* $q \in F$*.*

Theorem 21 (Theorem 1.39, page 55) *Every nondeterministic finite automaton has an equivalent deterministic finite automaton.*

Corollary 22 (Corollary 1.40, page 56) *A language is regular if and only if some nondeterministic finite automaton recognizes it.*

Theorem 23 (Theorem 1.45, page 59) *The class of regular languages is closed under the union operation.*

Theorem 24 (Theorem 1.47, page 60) *The class of regular languages is closed under the concatenation operation.*

Theorem 25 (Theorem 1.49, page 62) *The class of regular languages is closed under the star operation.*

Theorem 26 (proved as part of the proof of Theorem 4.5, page 169) *The class of regular languages is closed under complement.*

Definition 27 (Definition 1.52, page 64) *Say that R is a **regular expression** if R is*

1. *a for some a in the alphabet Σ,*
2. *ϵ,*
3. *$\emptyset$,*
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 expressions.*

In item 1 and 2, the regular expressions a and ϵ represent the languages $\{a\}$ and $\{\epsilon\}$, respectively. In item 3, the regular expression $\emptyset$ represents the empty language. In items 4, 5, and 6, the expressions represent the languages obtained by taking the union or concatenation of the languages R_1 and R_2, or the star of the language R_1, respectively.

Theorem 28 (Theorem 1.54, page 66) *A language is regular if and only if some regular expression describes it.*

Lemma 29 (Lemma 1.55, page 67) *If a language is described by a regular expression, then it is regular.*

Lemma 30 (Lemma 1.60, page 69) *If a language is regular, then it is described by a regular expression.*

Definition 31 (Definition 1.64, page 73) *A* ***generalized nondeterministic finite automaton*** *(GNFA), $(Q, \Sigma, \delta, q_{start}, q_{accept})$, is a 5-tuple where*

1. *Q is the finite set of states,*
2. *Σ is the input alphabet,*
3. *$\delta : (Q - \{q_{accept}\}) \times (Q - \{q_{start}\}) \to R$ is the transition function,*
4. *q_{start} is the start state, and*
5. *q_{accept} is the accept state.*

A GNFA ***accepts*** *a string w in Σ^* if w can be written as $w_1 w_2 \cdots w_m$, where each w_i is a member of Σ^*, and a sequence of states $r_0, r_1, \ldots, r_m$ exists such that*

1. *$r_0 = q_{start}$,*
2. *$r_m = q_{accept}$, and*
3. *for each i, we have $w_i \in L(R_i)$, where $R_i = \delta(r_{i-1}, r_i)$.*

Claim 32 (Claim 1.65, page 74) *For any* ***GNFA*** *G, $CONVERT(G)$ is equivalent to G.*

Theorem 33 (Theorem 1.70, page 78) ***Pumping lemma*** *If A is a regular language, then there is a number p (the pumping length) where, if s is any string in A of length at least p, then s may be divided into three pieces, $s = xyz$, satisfying the following conditions:*

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

Recall the notation where $|s|$ represents the length of string s, y^i means that i copies of y are concatenated together, and y^0 equals ϵ.