Burt Rosenberg, June 26, 1996

0.0.1 Languages

Definition 1 *A alphabet is a finite set of symbols. A string or word is a finite sequence of symbols from the alphabet.*

For instance, $\Sigma = \{a, b, \ldots, z\}$ is an alphabet, and *abc* or *howdyall* are words.

Definition 2 *The length of a string σ from an alphabet Σ, denoted $|\sigma|$, is the number of characters in the word.*

For instance $|abc| = 3$.

Definition 3 *The operation of concatentation, denoted $\cdot$, is a non-commutative, associative operation defined on the set of strings of an alphabet, with identity element ϵ, the empty string.*

For instance, with $\sigma = abc$ and $\tau = de$ then $\sigma \cdot \tau = abcde$. For all σ, $\epsilon\sigma = \sigma\epsilon = \sigma$, according to the rules of an identity element. Hence the empty string is the unique string of length zero on the alphabet.

Definition 4 *Suppose σ is a word, written $\sigma = \tau\gamma\delta$, where τ, γ and δ are (possibly empty) words. Then τ is a prefix of σ, δ is a suffix of σ, and γ as well as τ and δ are substrings of σ. A prefix, suffix or substring is proper if it does not equal the string.*

Hence *pre* is a proper prefix of $prefix$, and fix is a proper suffix of $suffix$.

Definition 5 *A language is a subset of the set of all strings of an alphabet.*

The empty language (the empty set) and the language containing only the empty string $L = \{\epsilon\}$ are distinct. The set of all valid C programs is a language of the alphabet of letters, numbers and several special punctuation. The names of Snow White's 12 dwarves is a language over the set of letters.

Definition 6 *Let L and M be languages over a common alphabet. Then the language $L \cdot M$ is defined as:*

$$LM = \{\, \sigma \cdot \tau \,|\, \sigma \in L, \tau \in M \,\}.$$

Let L and M be languages. The union of languages $L \cup M$ is the set-theoretic union of L and M. The Kleene closure of a language L, denoted L^*, is defined as:

$$L^* = \bigcup_{i=0}^{\infty} L^i.$$

Hence, L as a language over alphabet Σ is a subset of Σ^*, said "sigma-star." Since $\{\epsilon\}$ is a language acting as the identity for concatentation:

$$\{\epsilon\} \cdot L = L \cdot \{\epsilon\} = L,$$

it is logical to define $L^0 = \{\epsilon\}$. So $\epsilon \in L^*$ always. Another useful definition is $L^+ = L \cdot L^*$, which might not include the empty string, depending on L. Note that any σ in Σ^* has finite length. The union is infinite, but each term of the union includes only finite length words.

The empty language $\emptyset$ acts as the zero element where concatentation is multiplication: concatentation is multiplication and union is addition:

$$\emptyset \cdot L = L \cdot \emptyset = \emptyset,$$

and union is addition:

$$\emptyset \cup L = L \cup \emptyset = L.$$

The left and right distributative laws hold:

$$L \cdot (M \cup N) = L \cdot M \cup L \cdot N, \;\; (M \cup N) \cdot L = M \cdot L \cup N \cdot L,$$

where L, M and N are subsets of Σ^*.

0.0.2 Regular Expressions and Finite Automata

Regular Expressions are simple formulas for languages. Only simple languages result from the specification by regular expressions, they are languages recognizable by a finite state automata. The following is a recursive definition for regular expressions.

Definition 7 *Let Σ be an alphabet.*

1. *ϵ is a regular expression, with language $\{\epsilon\}$.*
2. *For all $a \in \Sigma$, a is a regular expression, with language $\{a\}$.*
3. *If R and S are regular expressions, with languages L_R and L_S respectively, then*
   (a) *$(R)|(S)$ is a regular expression denoting $L_R \cup L_S$,*
   (b) *$(R) \cdot (S)$ is a regular expression denoting $L_R \cdot L_S$,*
   (c) *$(R)^*$ is a regular expression denoting $L_R{}^*$.*

Unix utilities generally have pattern matching abilities, such as wildcards, which is unified by the use of regular expressions. Grep, ex, vi, sed and awk all use the same conventions, and languages such as Perl and Lex borrow with certain exceptions.

.	Match any character
*	Match zero or more preceeding
+	Match one or more preceeding
?	Match zero or one preceeding
[ ]	Match one character from a set
∧	Match beginning of line
$	Match end of line.

Definition 8 *A nondeterministic finite automata is a finite set Q of states, of which $q_o \in Q$ is distinguished as starting and $F \subset Q$ is distinguished as final, where Σ is an alphabet, and transition relations $\delta \subset \Sigma \times Q \times Q$ and $\delta^\epsilon \subset Q \times Q$. A deterministic finite automata differs in having transition function $\delta : Q \times \Sigma \to Q$.*

Beginning in state q_o, the symbols in the input word are used up, one by one from left to right, to orchestrate state transitions in the FA. In the case of the nondeterministic FA, $(s, q, q') \in \delta$ means that from state q seeing symbol s the automata can transition into state q'. If $(q, q') \in= \delta^\epsilon$, the transition from q to q' can occur irregardless of the next input symbol, and without using up a symbol. Elements of δ^ϵ are called *epsilon transitions.* Beginning in state q_o, a word $\sigma \in \Sigma^*$ is accepted if there exists a transition on the letters in σ leaving the FA in a final state. Else the word is rejected. The set of all accepted words, a subset of Σ^*, is the FA's language.

Theorem 1 (Kleene Representation Theorem) *A language can be described by a regular expression if and only if the languages can be accepted by a nondeterministic finite automata. Any language accepted by a nondeterministic finite automata can be accepted by a deterministic finite automata.*

To write a nondeterministic FA as a deterministic FA, we use the following procedure. *(See the Dragon book, pages 92 and 93)*

Definition 9 *For s a state, the epsilon-closure of s, $\epsilon(s)$, is a subset of states constructed by:*

1. *$s \in \epsilon(s)$.*
2. *If $t \in \epsilon(s)$ and $(t, t') \in \delta^\epsilon$, then $t \in \epsilon(s)$.*

If $T \subset \Sigma$, then

$$\epsilon(T) = \bigcup_{t \in T} \epsilon(t).$$

The states of the DFA are D, each $d \in D$ is a subset of Q, the states of the NFA. Start with D containing only $\epsilon(q_o)$. Repeat the following until the situation stabilizes. For any $d \in D$, and $s \in \Sigma$, define

$$T = \{\, t \in Q \mid \exists t' \in d : (s, t', t) \in \delta \,\}$$

Add $d' = \epsilon(T)$ to D as a new state, and add transition from d to d' labeled s.

To write a regular expression as a nondeterministic finite automata, build the NFA inductively according to the induction defining the regular expression. We create schemas for NFA's accepting ϵ and single letters, a schema for taking N accepting L_N and M accepting L_M and creating NFA's accepting $L_N | L_M, L_N L_M$ and ${L_M}^*$, by gluing together machines, ϵ transitions and a small number of additional states.

The proof of the second half of the Kleene Representation Theorem is now outlined. Given a deterministic FA we need to produce a regular expression which denotes the FA's language. The regular expression is created by building inductively R_{ij}^k, the set of strings which take the FA from the i-th to j-th state yet never pass through any state of index higher than k. For $k = 0$, it is a subset of Σ, including ϵ in the case $i = j$. For larger k the induction equation is:

$$R_{ij}^k = R_{ik}^{k-1} (R_{kk}^{k-1})^* R_{kj}^{k-1} \cup R_{ij}^{k-1}.$$

(See Hopcroft and Ullman's white book page 33-34.)