Words! - Regular Languages with Josh, 6/29

Jason Schuchardt

6/29/19

1 Words!

Definition 1.1. An *alphabet* is any finite set, which we think of as being a set of letters.

For example, $\{a, b, c\}$ is an alphabet, as is $\{\square, \text{apple}\}$.

Definition 1.2. Given an alphabet, a *word* over that alphabet, A is a finite sequence with digits in A.

For example, with the alphabets above, we have the words

$$X = abba,$$

and

$$Y = \square\square\square\text{apple}.$$

Definition 1.3. For a word X, we can define the *length* of the word to be the length of its sequence of letters, and we denote the length of X by $|X|$.

In particular, there is a word of length 0, called the *null word*, which we denote by λ.

Then $|X| = 0$ if and only if $X = \lambda$.

We can also speak of operations on words.

One such operation is concatenation of words. Given two words $x = x_0x_1 \cdots x_n$ and $y = y_0y_1 \cdots y_n$, then the concatenation of x with y is

$$xy := x_0x_1 \cdots x_ny_0y_1 \cdots y_m.$$

For example, if $x = abb$, $y = ba$, $xy = abbba$.

Observations: It's not generally commutative.

Definition 1.4. Given a word x, and a nonnegative integer $n \in \mathbb{Z}_{\geq 0}$ define

$$x^n = \begin{cases} \lambda & n = 0 \\ x^{n-1}x & n > 0 \end{cases}$$

Proposition 1.1. *For words u and v, $uv = vu$ if and only if there exists a word z such that $u = z^p$, $v = z^q$ for some $p, q \in \mathbb{Z}_{\geq 0}$.*

Proof. ($\Longleftarrow$) Suppose there exists z such that $u = z^p$, $v = z^q$. Then $uv = vu$ if and only if $z^pz^q = z^{p+q} = z^qz^p$, so we are done.

($\Longrightarrow$) Now we want to prove $uv = vu$ implies there exists z such that $u = z^p$, $v = z^q$. We'll need a WOP proof to prove this for all words. We will apply WOP to $|uv|$.

Let

$$S = \{n \in \mathbb{Z}_+ : \exists_{u,v}\ |uv| = n,\ uv = vu, \text{ and there is no word } z \text{ satisfying the claim}\}.$$

Assume for contradiction that S is nonempty. Let k be the least element of S, and let u, v be the corresponding words, so $uv = vu$, $|uv| = k$, and there is no z satisfying the claim.

Assume without loss of generality that $|u| \geq |v|$. Then $u = vt$ for some word t, because $uv = vu$, so the first $|v|$ letters of v are the same as the first $|v|$ letters of uv, which are the same as the first $|v|$ letters of u.

Then we have

$$vtv = uv = vu = vvt,$$

so $tv = vt$. If $v = \lambda$, we have $u = u^1$, $v = u^0$, so $|v| > 0$, so $|t| < |u|$. Thus $|tv| < |uv|$, so $|tv| \notin S$.

Thus $t = z^p$, $v = z^q$, and thus $u = vt = z^{p+q}$. Contradiction. ■

2 Languages

Definition 2.1. For the alphabet A, define

$$A^n = \begin{cases} \{\lambda\} & n = 0 \\ \{ax : a \in A, x \in A^{n-1}\} & n > 0, \end{cases}$$

that is, A^n is the set of all words over A of length n.

For example, if $A = \{a, b\}$, then

$$A^2 = \{aa, ab, ba, bb\}$$

Definition 2.2. If A is an alphabet, then we define

$$A^* = \bigcup_{n=0}^{\infty} A^n,$$

so A^* is the set of all words over A of any length.

We also have

$$A^+ = \bigcup_{n=1}^{\infty} A^n,$$

which excludes λ.

Definition 2.3. A *language*, L, over the alphabet A is a subset of A^*. I.e., $L \subseteq A^*$.

For example, if $A = \{a, b, c\}$, we might have

$$L = \{cab, ab, b, c, \lambda\}$$

3 Deterministic Finite State Automata (DFAs)

Definition 3.1. A *deterministic finite state automaton* (DFA), $\mathcal{M}$, is a quintuple

$$\mathcal{M} = (A, Q, \delta, q_0, F),$$

where

1. A is an alphabet,
2. Q is a finite set of elements we call *states*, often written as $\{q_0, q_1, \ldots, q_n\}$,

3. $\delta : Q \times A \to Q$ is a function called the *transition function* of the DFA,
4. $q_0 \in Q$ is the *initial state*, or *start state* of the DFA, and
5. $F \subseteq Q$ is the set of *final states* or *accepting states* of the DFA.

Example 3.1. Example DFA:

1. $A = \{a, b\}$,
2. $Q = \{q_0, q_1\}$,
3. $\delta(q_0, a) = q_1$, $\delta(q_0, b) = q_0$, $\delta(q_1, a) = q_1$, and $\delta(q_1, b) = q_1$.
4. q_0 is the initial state,
5. and $F = \{q_1\}$.

We can draw the directed graph of $\mathcal{M}$ $G(\mathcal{M})$ by the following rules:

1. Each state $q \in Q$ is a vertex of the graph.
2. The initial state is preceded by an arrow.
3. The final states are circled.
4. There is an edge labelled a between vertices q_b, q_c if $\delta(q_b, a) = q_c$.

So that I can draw these quickly in LaTeX, I will omit self loops, i.e., the edges that don't change the state of the machine.

$$\longrightarrow q_0 \xrightarrow{a} q_1$$

Definition 3.2 (Informal)**.** We say a DFA $\mathcal{M}$ *accepts* a word over A, x, if the sequence of letters in x describes a path in $G(\mathcal{M})$ from q_0 to an accepting state.

In this particular case, we can see from the graph that $\mathcal{M}$ accepts a word if and only if the word contains at least one a. For example, it accepts the words

$$abb, bab, b^k a, a.$$

Definition 3.3 (Informal)**.** The set of all words over A, that $\mathcal{M}$ accepts is called the language recognized by $\mathcal{M}$, denoted by $L(\mathcal{M})$. A language $L \subseteq A^*$ is called *regular* if there exists a DFA, $\mathcal{M}$, such that $L = L(\mathcal{M})$.

Exercise 1. What language is recognized if we changed the machine to:

$$\begin{tikzcd} {} \arrow[r] & q_0 \arrow[r, bend left, "a"] & q_1 \arrow[l, bend left, "{a,b}"] \end{tikzcd}$$

4 Regular Languages

Definition 4.1. Given $\mathcal{M}$, define $\delta_* : Q \times A^* \to Q$ by

$$\delta^*(q, x) = \begin{cases} q & x = \lambda \\ \delta(\delta^*(q, y), a) & x = ya, \text{ where } a \in A, y \in A^* \end{cases}$$

For example, for our original machine $\mathcal{M}$,

$$\begin{aligned}\delta^*(q_0, ab) &= \delta(\delta^*(q_0, a), b) \\ &= \delta(\delta(\delta^*(q_0, \lambda), a), b) \\ &= \delta(\delta(q_0, a), b) \\ &= \delta(q_1, b) = q_1\end{aligned}$$

Now we can formally define the language accepted by a DFA.

Definition 4.2. The *language accepted by a DFA*, $\mathcal{M}$, $L(\mathcal{M})$ is defined as

$$\{x \in A^* : \delta^*(q_0, x) \in F\}.$$

Definition 4.3 (Same as before)**.** A language over A, $L \subseteq A^*$, is called *regular* if there exists some DFA, $\mathcal{M}$, such that $L = L(\mathcal{M})$.

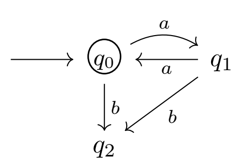
For example, over $A = \{a, b\}$, the set of words that contain at least one a is a regular language.

$\varnothing$ is a regular language, A^* is a regular language.

Exercise 2. Given an alphabet, $|A| = n$, and a set of states, $|Q| = m$, how many different DFAs can you make? How many distinct regular languages can you generate?

5 Definitely Fun Automata

Our language is $A = \{a, b\}$. Is $\{a^{2n} : n \in \mathbb{Z}_{\geq 0}\}$ regular? Well, we can draw a machine for it:

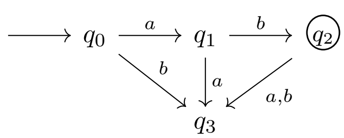


Question: Is every language in one symbol regular? No, $\{a^{n^2}\}$ is not regular.

How about the language:

$$\{ab\}$$

Yes. Consider:

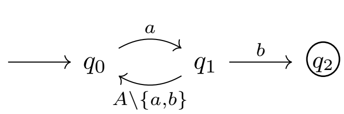


In fact, any finite language is regular.

Now we'll change our alphabet. $A = \{a, b, c_1, \ldots, c_n\}$.

Is $\{xaby : x, y \in A^*\}$ regular?

Yes. Consider

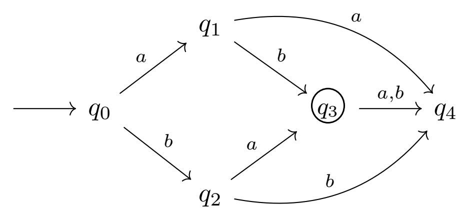


With the same language as before, what about

$$\{a^n b^n : n \in \mathbb{Z}_{\geq 0}\}?$$

This language is not regular. The problem is that we need to keep track of the number of as we've seen when we see b, but a DFA has a finite number of states, which means its memory is finite, so it can't keep track of that information. This is similar to the real world, since a real computer has a finite amount of memory.

What is the language accepted by the DFA:



The language accepted by this DFA is

$$\{x \in A^* : x \text{ has exactly one } a \text{ and one } b\}.$$