CMPS 260 (Theoretical Foundations of Computer Science)
The Pumping Theorem for Regular Languages

Here we state and prove the Pumping Theorem for regular languages, and then use it to prove the non-regularity of several languages.

First we state the following lemma, as it is used in the proof of the theorem.

Lemma: Let q be a state in a DFA and y be a string such that the path beginning at q and spelling out the string y ends at q. Then, for all $i \geq 0$, the path beginning at q and spelling out y^i ends at q.

Proof: omitted, as it would only complicate what is an obvious result!

Theorem: Let L be a regular language. Then there is a constant n (the value of which depends upon L) such that, for every string $w \in L$ of length n or greater, there exist strings x, y, and z satisfying $w = xyz$, $|xy| \leq n$, and $|y| > 0$ such that, for all $i \geq 0$, $xy^iz \in L$.

Proof: Assume that $L \subseteq \Sigma^*$ is regular, and let M be a DFA such that $L = L(M)$. Let n be the number of states in M, and let $w = a_1a_2 \cdots a_{|w|} \in L$, where $|w| \geq n$ and $a_k \in \Sigma$ for each k. For k satisfying $0 \leq k \leq |w|$, let $w_k = a_1a_2 \cdots a_k$ be the prefix of w of length k. Let q_k be the state at the end of the path from the initial state spelling out w_k. Then the sequence of states visited along the path is $q_0, q_1, q_2, \ldots, q_n, \ldots, q_{|w|}$. By the pigeonhole principle, there exist j and m satisfying $0 \leq j < m \leq n$ such that $q_j = q_m$. It follows that there are paths

(a) from the initial state to q_j spelling out $x = a_1a_2 \cdots a_j$,
(b) from q_j back to itself spelling out $y = a_{j+1}a_{j+2} \cdots a_m$, and
(c) from q_j to $q_{|w|}$ spelling out $z = a_{m+1}a_{m+2} \cdots a_{|w|}$.

Applying the above lemma to (b), we get

(d) for all $i \geq 0$, there is a path from q_j back to itself spelling out y^i.

From (a), (c), and (d), it follows that, for all $i \geq 0$, $xy^iz \in L$. **End of proof**

Notice that the Pumping Theorem describes a **necessary** condition for a language to be regular. That is, it says that for L to be regular, it must possess a particular property. (The theorem does not say that no non-regular language possesses that same property. Indeed, some non-regular languages **do** have that property.)

To state it a bit more formally, the Pumping Theorem is of the form

$$L \text{ is regular } \Rightarrow L \text{ satisfies } P$$

where P is the rather complicated condition stated in the theorem. Recall from Propositional Logic that an implication $A \Rightarrow B$ and its contrapositive $\neg B \Rightarrow \neg A$ (which we can also write as $\neg A \Leftarrow \neg B$) are equivalent. Thus, the Pumping Theorem is equivalent to its contrapositive

$$\neg(L \text{ is regular}) \Leftarrow \neg(L \text{ satisfies } P)$$

The Pumping Theorem tells us that to show that a language L is not regular, it suffices to show that L does not possess the property embodied by P. But P is a fairly complicated property! Exactly what would be required to show that a language does not possess that property? Well, if we were to state the Pumping Theorem in more formal terms (using the notation of predicate logic), it would look like this:

$$\begin{array}{l} L \text{ is regular } \Rightarrow \\ \quad (\exists n : n > 0 \wedge (\forall w : (w \in L \wedge |w| \geq n) \Rightarrow \\ \quad\quad (\exists x, y, z : w = xyz \wedge |y| > 0 \wedge |xy| \leq n \wedge (\forall i : i \geq 0 \Rightarrow xy^i z \in L)))) \end{array}$$

The contrapositive —the antecedant of which we find by several applications of DeMorgan's laws, namely $\neg(P \wedge Q) = \neg P \vee \neg Q$, $\neg(P \vee Q) = \neg P \wedge \neg Q$, $\neg(\forall x : Q) = (\exists x : \neg Q)$, and $\neg(\exists x : Q) = (\forall x : \neg Q)$— is as follows:

$$\begin{array}{l} L \text{ is not regular } \Leftarrow \\ \quad (\forall n : n > 0 \Rightarrow (\exists w : w \in L \wedge |w| \geq n \wedge \\ \quad\quad (\forall x, y, z : w = xyz \wedge |y| > 0 \wedge |xy| \leq n \Rightarrow (\exists i : i \geq 0 \wedge xy^i z \notin L)))) \end{array}$$

In words, this says that L is not regular if, for every positive integer n, there exists a string $w \in L$ of length at least n such that, for every x, y, and z satisfying $w = xyz$, $|y| > 0$, and $|xy| \leq n$, there is some nonnegative integer i for which $xy^i z \notin L$.

Thus, to prove that a language L is not regular, it suffices to

(1) Let $n > 0$ be arbitrary.

(2) Choose a string $w \in L$ of length at least n.

(3) Identify every possible way to choose strings x, y, and z satisfying $w = xyz$, $|y| > 0$, and $|xy| \leq n$, and partition them into cases (such that all the choices covered by any one case can be treated in a uniform manner).

(4) For each case arising from (3), find a value of i for which $xy^i z \notin L$.

Example 1: Show that $\{a^i b^j : i < j\}$ is not regular.
Solution: Let $n > 0$ be arbitrary, and choose $w = a^n b^{n+1}$. Every choice of x, y, and z satisfying the three conditions described in (3) above is such that $x = a^p$, $y = a^q$ and $z = a^{n-p-q} b^{n+1}$ for some $p \geq 0$ and $q > 0$. Choosing $i = 2$ we have

$$xy^2 z = a^p (a^q)^2 a^{n-p-q} b^{n+1} = a^p a^{2q} a^{n-p-q} b^{n+1} = a^{p+2q+n-p-q} b^{n+1} = a^{q+n} b^{n+1}$$

But $q + n \geq n + 1$ (due to the fact that $q > 0$), which means that $xy^2 z \notin L$.

Example 2: Show that $L = \{a^p : p \text{ is prime}\}$ is not regular.
Solution: Let $n > 0$ be arbitrary, and choose $w = a^p$, where p is the smallest prime number greater than or equal to n. (It is well known that there are infinitely many primes; hence p exists.) Every choice of x, y, and z satisfying the three conditions described in (3) above is

such that $x = a^q$, $y = a^r$ and $z = a^{p-q-r}$ for some $q \geq 0$ and $r > 0$. To complete the proof, we must find a value for i that makes xy^iz a non-member of L. We have

$$xy^iz = a^q a^{ri} a^{p-q-r} = a^{p+ri-r} = a^{p+r(i-1)}$$

Thus, our problem boils down to finding a value for i such that $p + r(i-1)$ is non-prime. Choose $i = p+1$. Then we get $p + r(i-1) = p(1+r)$, which is the product of two numbers greater than one and hence is not prime.

Example 3: Show that $L = \{(ab)^i b^i : i \geq 0\}$ is not regular.
Solution: Let $n > 0$ be arbitrary, and choose $w = (ab)^n b^n$. Considering every possible choice of x, y, and z satisfying the three conditions described in (3) above, we get the following cases:

Case 1: $y = (ab)^q$ for some $q > 0$. Then $x = (ab)^p$ for some $p \geq 0$ and $z = (ab)^{n-p-q}b^n$. Taking $i = 0$, we have

$$xy^0z = (ab)^p(ab)^{q\cdot 0}(ab)^{n-p-q}b^n = (ab)^{n-q}b^n$$

which is not in L because there are fewer occurrences of ab than of the following b's.

Case 2: $y = (ab)^q a$ for some $q \geq 0$. Then $x = (ab)^p$ for some $p \geq 0$ and $z = b(ab)^{n-p-q-1}b^n$. Taking $i = 0$, we have

$$xy^0z = (ab)^p((ab)^q a)^0 b(ab)^{n-p-q-1}b^n = (ab)^p b(ab)^{n-p-q-1}b^n$$

which is not in L because either it begins with b (in the case $p = 0$) or else it contains an occurrence of bb to the left of an occurrence of a (in the case $p > 0$). (It is clear from the definition of L that none of its members begins with b or contains an occurrence of bb to the left of an occurrence of a.)

Case 3: $y = b(ab)^q$ for some $q \geq 0$. Then $x = (ab)^p a$ for some $p \geq 0$ and $z = (ab)^{n-p-q-1}b^n$. Taking $i = 0$, we have

$$xy^0z = (ab)^p a(b(ab)^q)^0(ab)^{n-p-q-1}b^n = (ab)^p a(ab)^{n-p-q-1}b^n$$

which is not in L because it, unlike any member of L, it contains an occurrence of aa. (Note: To justify the claim that $(ab)^p a(ab)^{n-p-q-1}b^n$ contains an occurrence of aa, we show that $n-p-q-1 > 0$, or, equivalently, $n > p+q+1$. We have $|x| = |(ab)^p a| = 2p+1$ and $|y| = |b(ab)^q| = 2q+1$. Recall that we need to consider only those choices of x and y satisfying $|xy| \leq n$, which here translates into $(2p+1) + (2q+1) \leq n$. With a little algebra, this yields $n > 2p+2q+1$, which implies the desired result.)