Section 1.4 - Nonregular Languages

CSC 320

Nonregular Languages – the Pumping Lemma

Consider

$$L = \{a^n b^n | n \geq 0\}$$
$$= \{\varepsilon, ab, aabb, aaabbb, aaaabbbb \ldots\}$$

Intuitively: must remember *how many* a's we have seen to match with the number of b's:

$$aaaaaaaaaaaaaaaaaaabbbbbbbbbbbbbbbbbbb$$

But FA have only finite memory, and the center can be arbitrarily far from the start.

Pigeonhole Principle

To prove formally that there is no DFA that accepts L we need:

The Pigeonhole Principle: If A and B are finite sets and $|A| > |B|$ then there is no 1-1 function from A to B, i.e., if we assign each element of A (the "pigeons") to an element of B (the "pigeonholes") eventually we must put more than one pigeon in the same hole.

Proof that L is not Regular

The proof is by contradiction. Suppose L is regular. Then there is a DFA M such that $L = L(M)$.

- Let p = number of states in M.
- Given $a^l b^l$ for $l > p$, M must be in some state q more than once while the a's are scanned, by the pigeonhole principle.
- Partition $a^l b^l$ into x, y, and z, where y is the string of a's scanned between the two times state q is entered. Let $i = |y|$.

Observe: We can leave out y or repeat y any number of times and end up in the same state. But then for any $k \geq 0$, $a^{l+(k-1)i} b^l \in L(M)$!

E.g., $a^{l-i} b^l \in L(M)$.

The Pumping Lemma

Theorem: Let L be a regular language. Then there is a number $n > 0$ (the "pumping length" of L) such that for every string w in L such that $|w| \geq n$, we can break w into three strings, $w = xyz$, such that:

1. $|xy| \leq n$
2. $y \neq \varepsilon$
3. $xy^k z \in L$ for each $k \geq 0$.

Proof of the Pumping Lemma

- Let n be the number of states in the finite automaton $M = (Q, \Sigma, \delta, q_0, F)$ which accepts L. Let $w = w_1 w_2 \ldots w_l$ be a string of length $l \geq n$. Let $r_1 \ldots r_{l+1}$ be the sequence of states M enters into while processing w.

- By the Pigeonhole Principle, two of the states among the first $l + 1$ states are the same. Call the first r_s and the second r_t.

- – Let $x = w_1 \ldots w_{s-1}$, $y = w_s \ldots w_{t-1}$, $z = w_t \ldots w_l$.
  – We can easily verify each of the conditions of the lemma.

Proving a Language L is not Regular

The Pumping Lemma gives a condition that must be satisfied by every regular language. How can we use it to show a language is *not* regular?

Contrapositive: For any language L:

IF for every $n > 0$, there exists a string $w \in L$, $|w| \geq n$, such that for *any* decomposition of w into xyz with $|xy| \leq n$, there is some $k \geq 0$ such that $xy^kz \notin L$

THEN L is *not* regular

Example: $L = \{a^r b^s | r \geq s\}$

We are given $n > 0$.

We pick $w = a^n b^n \in L$.

We are given xyz with
the following properties:

1. $w = xyz$
2. $|xy| \leq n$
3. $y \neq \varepsilon$

We pick $k = 0$.

Now, since $|xy| \leq n$, it *must* be the case that $xy = a^j$ for some $j > 0$. Since $y \neq \varepsilon$, it must be the case that $y = a^i$ with $i > 0$. So $xy^k z = xy^0 z = a^{n-i} b^n \notin L$ since there are more b's than a's. So L is *not* regular. (Pumping down)

Using Closure Properties

Theorem: The class of languages accepted by finite automata is closed under

1. union;
2. concatenation;
3. star;
4. complementation;
5. intersection;
6. reversal;.

Example 2

$L = \{w \in \{a, b\}^* | w \text{ has an equal number of } a's \text{ and } b's\}$

If L is regular then $L \cap L(a^*b^*)$ is regular, since the regular languages are closed under intersection. But $L \cap L(a^*b^*) = \{a^n b^n \mid n \geq 0\}$, which we already showed is not regular, giving a contradiction.

Or Using the Pumping Lemma

How to pick a string:

Example 3

$L = \{ww \mid w \in \{0,1\}^*\}$

Example 4

$L = \{010^n1^n \,|\, n \geq 0\}$

More than one case for the decomposition.