Welcome to Cpts 317

Mon Feb 10

Topic of the Day
The Pumping Lemma

(aside: return of HW1 papers)
Regular Language

- Finite Automata
  - Formal definition of Computation
  - Designing F.A.
  - Regular Operations

- Non-determinism
  - Equivalence of NFAs & DFAs
  - Closure under regular operation

- Regular expressions
  - Formal definition of reg. expr.
  - Equivalence with F.A.

- Non-regular Language
  - The Pumping Lemma for Regular Language
Pumping Lemma

If $A$ is a regular language, then there is a number $p$ where if $s$ is any string in $A$ of length $|s| \geq p$, 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. $|xy| \leq p$

$p$ is called the pumping length.

It is typically $=$ number of states in DFA.
Note:

* When 5 is divided into xy2, either x or 2 may be E, but condition 2 says that y ≠ E.

* Condition 3 states that the pieces x and y together have length at most p.
Proof Idea

Let \( M = (Q, \Sigma, \delta, q_0, F) \) be a DFA that recognize \( A \).

We assign the pumping length \( p \) to be the number of states of \( M \).

We want to show that any string \( s \) in \( A \) of length at least \( p \) may be broken into three pieces \( x y z \), satisfying our three conditions.

Case 1: no strings in \( A \) are of length \( \geq p \).

Then, the Lemma is vacuously true; the three conditions hold for all strings of length \( \geq p \) if there aren't any such strings.
Case 2: If $s \in A$ has $|s| \geq p$:

Consider the sequence of states that $M$ goes through when computing with input $s$.

It starts with the start state $q_1$, say, then it goes to some state $q_5$, then say $q_{10}$, ..., until it reaches $q_{\text{accept}}$.

If we let $n = |s|$, then the sequence $q_1, q_5, ..., q_{\text{accept}}$ has length $n + 1$.

Because $n$ is at least $p$, we know that $n + 1 \geq p = \text{number of states of } M$. 
Therefore, the sequence must contain a repeated state.

This result is an example of the pigeonhole principle — if \( p \) pigeons are placed into fewer than \( p \) holes, some hole has to have more than one pigeon in it.

Illustration (97 is the one that repeats)

\[
S = S_1 \rightarrow S_2 \rightarrow S_3 \rightarrow S_4 \rightarrow S_5 \rightarrow S_6 \rightarrow \cdots \rightarrow S_n
\]

\[
\begin{array}{ccccccc}
9_1 & 9_5 & 9_2 & 9_7 & 9_1 & 9_7 & 9_{25} & \text{accept}
\end{array}
\]

We divide \( S \) into the three pieces \( x, y, \) and \( z \).

- \( x \): part of \( S \) appearing before 97
- \( y \): part between the two appearances of 97
- \( z \): remaining part of \( S \), after second occurrence of 97
Let us see why this division of $S$ satisfies the three conditions.

Suppose we run $M$ on input $xyyz$. $x$ takes $M$ from $q_1$ to $q_f$, then the first $y$ takes it from $q_f$ back to $q_1$, as does the second, and then $z$ takes it to $q_{accept}$. VOK

Similarly, it will accept $xzyz$ for $n > 0$. 

x: take M from $q_1$ to $q_f$
y: take M from $q_f$ back to $q_1$
z: take M from $q_1$ to $q_{accept}$
For the case $i = 0$, $xy^i z = xz$, which is accepted for similar reason. This establishes Condition 1.

Condition 2: We see that $1y1 > 0$, as it was the part of $S$ that occurred between two different occurrences of $77$.

Condition 3: We ensure that $77$ is the first repetition in the sequence. Then by the pigeonhole principle, the first $p + 1$ states in the sequence must contain a repetition. Therefore $|x y l| \leq p$. 
Proof

Let \( M = (Q, \Sigma, \delta, q_0, F) \) be a DFA recognizing \( A \) and \( p \) be the number of states of \( M \).

Let \( s = s_1 s_2 \ldots s_n \) be a string in \( A \) of length \( n \), where \( 1 \leq p \).

Let \( r_1, \ldots, r_n \) be the sequence of states that \( M \) enters when processing \( s \), so

\[
    r_i = \delta(r_{i-1}, s_i)
\]

for \( 1 \leq i \leq n \).

This sequence has length \( n+1 \), which is at least \( p+1 \).
Among the first $p+t$ elements in the sequence, two must be the same state, by the pigeonhole principle.

We can the fact of these $r_j$ and the second $r_e$.

Because $r_e$ occurs among the first $p+t$ places in the sequence starting at $r_1$, we have $k = p+1$.

Now we let

\[ x = s_1 \ldots s_{j-1} \]
\[ y = s_j \ldots s_{e-1} \]
\[ z = s_e \ldots s_n \]
As \( x \) takes \( M \) from \( r_i \) to \( r_j \),
\( y \) takes \( M \) from \( r_j \) to \( r_j \),
\( z \) takes \( M \) from \( r_j \) to \( r_{i+1} \),
which is an accept state, \( M \) must accept \( xy^i z \) for \( i \geq 0 \).

We know that \( i \neq k \), so \( |y| \geq 0 \);
and \( k \leq p+1 \), so \( |x y| \leq p \).

Thus we have satisfied all three conditions of the PL.
How to use the PL to prove that a language B is not regular:

(1) Assume that B is regular in order to obtain a contradiction.

(2) Use the PL to guarantee the existence of a pumping length p such that all strings of length p or greater in B can be pumped.

(3) Demonstrate that s cannot be pumped by considering all ways of dividing s into x, y, and z, and for each such division, finding a value i such that x y^i z ∈ B.