Non-Regular Languages

The finiteness of the state set of DFA's leads to ways to give examples of non-regular languages.

Suppose DFA M recognizes an infinite language L.

- If we look at long enough strings in L, we will find one that causes M to visit the same state q more than once.
- We can then repeat (or "pump") the segment that takes us from q back to q, to obtain an infinite family of related strings in L.

The Pumping Lemma

Pumping Lemma: *(Sipser, 1.70)* Suppose L is a regular language. Then there exists a number p such that, if w is any string in L with $|w| \geq p$, then w may be written $w = xyz$ such that the following all hold:

1. $xy^iz \in L$, for all $i \geq 0$
2. $|y| > 0$
3. $|xy| \leq p$.

$$\begin{aligned} &\forall L.\ L\ \texttt{regular}\ \rightarrow \\ &\qquad \exists p.\ \forall w.\ w \in L \wedge |w| \geq p \rightarrow \\ &\qquad\qquad \exists x\ y\ z.\ w = xyz \wedge (1) \wedge (2) \wedge (3) \end{aligned}$$

Proof of the Pumping Lemma

- Assume L is regular.
- Let M be a DFA that recognizes L.
- Let p be the number of states of M.
- Assume $w \in L$ is such that $|w| \geq p$.
- Choose states r_0, r_1, . . . , $r_{|w|}$ such that $r_0 = q_0$, $r_{|w|} \in F$, and $r_{i+1} = \delta(r_i, w_{i+1})$ for $0 \leq i \leq |w|$. Such states exist because M accepts w.
- By the Pigeonhole Principle, there exist j and k with $0 \leq j < k \leq p$ such that $r_j = r_k$.

- Let
  - $x = w_1 w_2 \ldots w_j$,
  - $y = w_{j+1} w_{j+2} \ldots w_k$, and
  - $z = w_{k+1} w_{k+2} \ldots w_n$;

  so that $w = xyz$. Since $j < k$ we have $|y| = k - j > 0$. Since $k \leq p$ we have $|xy| \leq p$.

- For any given $i \geq 0$, the sequence of states:

$$r_0, \ldots, r_j, \underbrace{r_{j+1}, \ldots, r_k,}_{i \text{ times}} r_{k+1}, \ldots, r_{|w|}$$

  shows that M accepts xy^iz, hence $xy^iz \in L$.

Applying the Pumping Lemma

Example: *(Sipser, 1.73)* The language $B = \{0^n1^n.\ n \geq 0\}$ is not regular.

Proof:

- Suppose (to obtain a contradiction) that B is regular.
- By the Pumping Lemma (using $\forall$-elimination):

$$\begin{array}{l} \exists p.\ \forall w.\ w \in B \wedge |w| \geq p \rightarrow \\ \qquad \exists x\ y\ z.\ w = xyz \wedge (1) \wedge (2) \wedge (3) \end{array}$$

  where (1), (2), (3) are as in the statement of the Pumping Lemma.

- Obtain p (using $\exists$-elimination) so that

$$\forall w.\ w \in B \wedge |w| \geq p \to (\exists x\ y\ z.\ w = xyz \wedge (1) \wedge (2) \wedge (3))$$

- Let w be the string 0^p1^p. Since $w \in B$ and $|w| = 2p \geq p$, we have (by $\forall$-elimination and implication elimination):

$$\exists x\ y\ z.\ w = xyz \wedge (1) \wedge (2) \wedge (3).$$

- Obtain (using $\exists$-elimination) x, y, and z so that
  - $w = xyz$
  - $xy^iz \in B$, for all $i \geq 0$
  - $|y| > 0$
  - $|xy| \leq p$

- Since $|xy| \leq p$, the string y must have the form 0^k for some k with $0 < k \leq p$. Then x must be 0^{p-k} and we must have $0^{p-k}1^p = xy^0z \in B$.
- But $0^{p-k}1^p$ cannot be in B, because $p - k \neq p$, so the assumption that B is regular leads to a contradiction. We conclude that B cannot be regular.

Note:

- The PL gives us p (we cannot choose it as we wish).
- The pumping length p depends on L.
- We get to choose w however we wish, as long as $|w| \geq p$ and $w \in L$.
- The PL gives us x, y, and z (we can't choose them).

Using Other Facts with the PL

Sometimes it is useful to apply other knowledge about regular languages before applying the PL.

Example: Let $L = \{w \in \{0,1,2\}^*. \#0(w) + \#1(w) = \#2(w)\}$. Then L is not regular.

Proof:

- Suppose L were regular.

- Then

$$L \cap 0^*1^*2^* = \{0^m 1^n 2^{m+n}.\ m \geq 0, n \geq 0\}.$$

  would also be regular.

- Let p be the pumping length given by the PL (for $L \cap 0^*1^*2^*$), and let $w = 0^p 1^p 2^{2p}$.

- Obtain, by the PL, strings x, y, z with $w = xyz$, $|y| > 0$, $|xy| \leq p$, and $xy^i z \in L \cap 0^*1^*2^*$ for all $i \geq 0$.

- Since $|xy| \leq p$, we must have $y = 0^k$ for some k with $0 < k \leq p$.

- Then $xy^0 z = xz = 0^{p-k} 1^p 2^{2p} \in L \cap 0^*1^*2^*$. This is a contradiction, because $(p-k) + p = 2p - k \neq 2p$.

- Since our initial assumption that L is regular leads to a contradiction, we conclude that L is not regular.

A Non-Regular Unary Language

Example: *(Sipser, 1.75)* The language $D = \{1^{n^2}.\ n \geq 0\}$ is not regular.

Proof: Suppose D were regular. Let p be the pumping length, given by the PL. Let $w = 1^{p^2}$. By the PL, we may obtain x, y, z such that $w = xyz$, $|y| > 0$, $|xy| \leq p$, and $xy^iz \in D$ for all $i \geq 0$. In particular xyz and $xy^2z \in D$.

Now, since $|xy| \leq p$, we must have $|y| \leq p$. Also $|xyz| = p^2$, so $|xy^2z| = p^2 + |y| \leq p^2 + p$. However, there is no perfect square between p^2 and $(p+1)^2 = p^2 + 2p + 1$, so $p^2 + p$ cannot be a perfect square. This contradicts $xy^2z \in D$, and we conclude that D cannot be regular.

A Generalization

Def. An *arithmetic progression* is a sequence of numbers such that the difference between consecutive terms is constant.

Prop. If $L \subseteq \{1\}^*$ is regular, then the set of lengths of strings in L includes an arithmetic progression.

Proof Idea: The lengths of strings $|xy^kz|$ given by a particular application of the Pumping Lemma is such an arithmetic progression.

The gaps between the lengths of strings in L cannot become arbitrarily large.