COMP30026 Models of Computation

Lecture 12: The Pumping Lemma for Regular Languages

Mak Nazecic-Andrlon and William Umboh

Semester 2, 2024

Limitations of Finite Automata

Cannot look ahead.

Fixed number of bits of memory.

How many bits to recognise this, without lookahead?

$$\{0^n 1^n \mid n \geq 0\} = \{\epsilon, 01, 0011, 000111, \ldots\}$$

Exercise: Is the language $L_1 = \{0^n 1^n \mid 0 \leq n \leq 999999999\}$ regular?

What about $L_2 = \left\{ w \;\middle|\; \begin{array}{l} w \text{ has an equal number of occurrences} \\ \text{of the substrings 01 and 10} \end{array} \right\}$?

The Pumping Lemma for Regular Languages

Lemma

If A is a regular language over Σ, then there is some integer p such that, for all $s \in A$ of length at least p, there exist $x, y, z \in \Sigma^*$ such that $s = xyz$ and

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

The standard tool for proving languages non-regular.

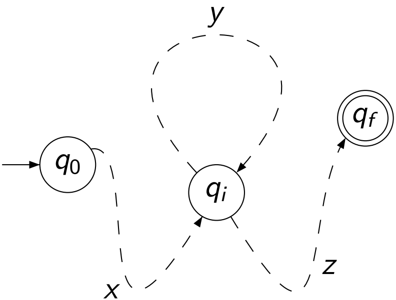
Loosely, if we have a DFA M for a regular language A and a sufficiently long string $s \in A$, then M must traverse a loop to accept s. So A must contain infinitely many strings exhibiting repetition of some substring in s.

Intuition for the Pumping Lemma

Pigeonhole principle: If you put p pigeons into fewer than p holes, some hole has more than one pigeon.

If a DFA has p states, and you run it on a string longer than p symbols, it must enter some state twice.

Therefore it passes through a cycle in the graph!



Tools for the Proof

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

Definition

Let $\hat{\delta} : Q \times \Sigma^* \to Q$ such that for all $q \in Q$, $a \in \Sigma$ and $s \in \Sigma^*$,

$$\hat{\delta}(q, \epsilon) = q,$$
$$\hat{\delta}(q, as) = \hat{\delta}(\delta(q, a), s).$$

Lemma

M accepts a string s if and only if $\hat{\delta}(q_0, s) \in F$.

Lemma

For all $q \in Q$ and $x, y \in \Sigma^*$,

$$\hat{\delta}(q, xy) = \hat{\delta}(\hat{\delta}(q, x), y).$$