Pumping Lemma

Why we need it. It turns out that not everything can be recognized by finite automata. The usual proof of this statement uses an auxiliary result known as Pumping Lemma.

What is a lemma. In mathematics, there are three types of statements:

- a *theorem* is an interesting difficult-to-prove statement;
- a *proposition* is an interesting easier-to-prove statement; and
- a *lemma* is a statement that, by itself, is not very interesting, but which is useful in proving other results.

Formulation of the Pumping Lemma. For every regular language L, there exists a natural number p such that every word from L whose length $\text{len}(w)$ is at least p can be represented as a concatenation $w = xyz$, where:

- y is non-empty;
- the length $\text{len}(xy)$ does not exceed p, and
- for every natural number i, the word $xy^iz \stackrel{\text{def}}{=} xy\ldots yz$, in which y is repeated i times, also belongs to the language L.

Notes.

- While y has to be non-empty, both x and z can be empty.
- A regular language can be empty, it can consists of a single word. In this case, the pumping lemma is still true. All this lemma says is that:
  - *if* a word w from this language has length p or larger,
  - *then* this word can be represented as a concatenation xyz.

  If the language has no words of length $\geq p$, then the if-then statement is still true. In general:
  - if the condition A is false,
  - then the conclusion "if A then B" is true.

  For example, it is true that for every natural number n:

  – if n is divisible by 4,
  – then n is an even number.

  For $n = 5$ the if-condition is false, but this does not make the whole if-then-statement false.

- This argument applies to every finite language, i.e., to every language that consists of finitely many words. By the way, every finite language is regular – because:

  – every word can be represented as a concatenation of letters, and
  – every finite language can be represented as a union of 1-word languages.

  For example, the language consisting of three words $\{Mom, Dad, uncle\}$ can be represented as

$$Mom \cup Dad \cup uncle.$$

  Here is a related trick question: is the set of all the names of all aliens from other planets who currently live on the Earth and pretend to be human regular? Yes, because it is finite. It may be empty, it may be not, but it is finite.

Why is it called pumping. First, we have the part x, then we repeat y several times – similarly to how when we pump air into the bicycle tires, we repeat the same pushing action several times.

How we can make the description shorter. We can shorten the description if we use mathematical abbreviations:

- $\forall$ for "for every";
- $\exists$ for "there exists";
- $A \to B$ for "if A then B", and
- & for "and".

By using these notations, the above text takes the following form:

$$\forall\,\text{reg.}L\,\exists p\,\forall w \in L\,(\text{len}(w) \ge p \to$$

$$\exists x, y, z\,(w = xyz \,\&\, \text{len}(y) > 0 \,\&\, \text{len}(xy) \le p \,\&\, \forall i\,(xy^i z \in L))).$$

Proof. By definition, a regular language is a language which is accepted by an automaton. Let p be the number of states in this automaton.

Suppose that we have a word w whose length L is at least p. Then, we have:

- the starting state,
- the state after reading the first symbol,
- the state after reading the second symbol,
- $\ldots$,
- the state after reading the p-th symbol.

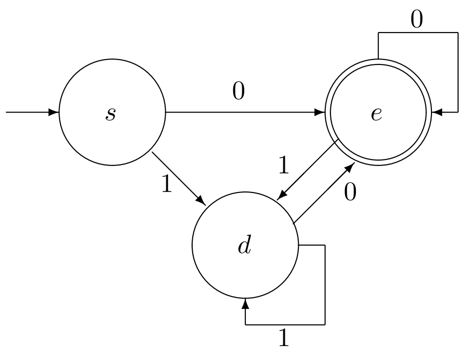
In the process if reading p symbols, the automaton passes through $1 + p$ states.

However, the automaton has only p different states. Thus, within the first p symbols, at least one state is repeated.

So, we can define x, y, and z as follows: we find the first repeating state; then:

- x is everything before the first occurrence of the repeating state;
- y is everything in between the first and the second occurrences of the repeating state; and
- z is everything after the second occurrence of the repeating state.

Example. Let us consider the automata for recognizing even unsigned binary integers – i.e., binary sequences that end in 0:



Let us trace how the word 11010 will be accepted by this automaton:

- we start at the state s;
- then, we read 1 and move to the state d;
- then, we read another 1 and stay in d;
- then, we read 0 and go to the state e, etc.

This tracing can be described graphically as follows:

$$\begin{array}{ccccccccccc} | & 1 & | & 1 & | & 0 & | & 1 & | & 0 & | \\ s & & d & & d & & e & & d & & e \end{array}$$

This automaton has $p = 3$ states. The first repeating state is the state d:

$$\begin{array}{ccccccccccc} | & 1 & | & 1 & | & 0 & | & 1 & | & 0 & | \\ s & & \underline{d} & & \underline{d} & & e & & d & & e \end{array}$$

Here:

- x is everything before the first occurrence of the repeating state d, i.e., $x = 1$;
- y is everything between the two occurrences of the repeating state, i.e., $y = 1$; and
- z is everything after the second occurrence of the repeating state, i.e. $z = 010$.

Here indeed, $w = xyz$.

When we read the word y in the state d, we go back to the state d. So, if we repeat it again, we will again go back to d – i.e., the word $xyyz$ will still be accepted:

$$\begin{array}{ccccccccccccc} | & 1 & | & 1 & | & 1 & | & 0 & | & 1 & | & 0 & | \\ s & & \underline{d} & & \underline{d} & & \underline{d} & & e & & d & & e \end{array}$$

Same thing will happen if we repeat y three time, four times, etc.

Practice. Try this on some other automaton.

Important: we talk about repeating *states*, not repeating *symbols*.

Pigeonhole Principle. In the proof of the Pumping Lemma, we used the fact that if the automaton passes through $p+1$ states, and there are only p different states, then at least two of the passing states must coincide.

In general, this statement is known as the *Pigeonhole Principle:* if we place all pigeons into holes, and there are more pigeons than holes, then some hole will contain at least two pigeons.