

COMP2022

Models of Computation

Non-regularity

Sasha Rubin

August 19, 2024



To show that a language is regular, it is sufficient to find a DFA, NFA, or Regular Expression for it.

But how do we show a language is **not regular**?

One must show that there is no DFA that recognises it.

A conversation with Kleene

Student: I can't show $L = \{a^n b^n : n \geq 1\}$ is regular. I've tried NFAs, DFAs, and regular expressions. I actually suspect it is not regular.

Kleene: Let's try prove that there is no DFA for L .

S: But how do we prove something doesn't exist?

K: Well, we can let M be any DFA, without saying exactly which one it is, and show that M doesn't recognise L (sort of like in algebra when you reason that $x^2 < 0$ has no real solution by reasoning about x).

S: Ok. But we need to know *something* about M , otherwise we can't reason about it...

K: Well, we know it only has finitely many states, say k many; and when the DFA consumes a string it ends in just one of these states.

S: That doesn't seem like a lot of information... ;(

K: Amazingly, it is enough. Here is the idea:

We will show that there are two strings x and y that go to the same state in M , and another string z such that $xz \in L$ and $yz \notin L$.

S: Oh! If we can do this then $L(M) \neq L$ because M accepts yz which is not in L !

S: Ok, so I need to find these three strings x, y and z . Well... we want xz to be in the language, yz not in the language... so let's try $x = a^{10}$, and $z = b^{10}$, and $y = a^{20}$.

K: You probably shouldn't be using fixed numbers...

S: Right right right. We're gonna want $x = a^n$ and $z = b^n$ for some n . And we're gonna want $y = a^m$ for some $m \neq n$. That would give us $xz = a^n b^n \in L$ and $yz = a^m b^n \notin L$.

K: Right. Now we just need to reason why we can choose n, m so that a^n and a^m go to the same state... We haven't yet used that there are only k many states...

S: Hmmm....

K: Look at all the strings you have to choose from a^1, a^2, a^3, \dots
Each string goes to some state, say q_1, q_2, q_3, \dots

S: ... so at some point we're gonna run out of states and some state will repeat!

K: This is called the **pigeonhole principle** in Discrete Math... It says "If we put objects into containers, but we have more objects than containers, then some container must hold more than one object".

S: So if we think of every state as a container that holds all the strings that go to it, then since we only have k many states, at least one state must contain two of the strings from a^1, a^2, \dots, a^{k+1} .

K: Done!

To summarise: we can conclude that $L(M) \neq L$ if we can

find strings x and y that go to the same state in M , and another string z such that $xz \in L$ and $yz \notin L$.

And, to show this we can:

find strings x_1, x_2, x_3, \dots so that for every $i \neq j$ we can find a string z (which may depend on i, j), such that $x_i z \in L$ and $x_j z \notin L$.

$L = \{a^n b^n : n \geq 0\}$ is not regular

find strings x_1, x_2, x_3, \dots so that for every $i \neq j$ we can find a string z (which may depend on i, j), such that $x_i z \in L$ and $x_j z \notin L$.

Proof.

1. Let $x_i = a^i$ for $i \in \mathbb{N}$
2. For $i \neq j$, let $z = b^i$.
3. Then $x_i z = a^i b^i \in L$.
4. And $x_j z = a^j b^i \notin L$.



$L = \{ww : w \in \{a,b\}^*\}$ is not regular

Proof.

1. Let $x_i = a^i b$ for $i \in \mathbb{N}$.
2. Given i, j , say $i < j$, let $z = a^i b$.
3. Then $x_i z = a^i b a^i b$ which is in L (take $w = a^i b$).
4. And $x_j z = a^j b a^i b$ which is not in L since the left half of this string only contains a s while the right contains two b s (since $i < j$).



Another technique

Once we know that a language is not regular, we can deduce that any language that is a Boolean combination of regular languages is regular.

- The union of two regular languages is regular.
- The intersection of two regular languages is regular.
- The complement of a regular language is regular.

We can use this to show that a language L is not regular using **proof of a negation** (see Tutorial on Assumed Knowledge).

Proof of a negation:

We want to prove that a statement P is false.

1. We assume that P is true.
2. We arrive at an impossible situation (incorrect conclusion).
3. Conclude the statement P must in fact be false.

We want to prove that a particular language L is not regular.

1. Assume L is regular.
2. Express some other language L' which we know is not regular (because we've proved this already) as a Boolean combination of regular languages (including L).
3. This is an impossible situation (the 'contradiction').
4. Conclude L cannot be regular.

Showing a language is not regular

Let L_{equal} be the strings over $\{a, b\}$ with the same number of a s as b s.

Note that $\{a^n b^n : n \geq 0\} = L_{\text{equal}} \cap L(a^* b^*)$.

So L_{equal} cannot be regular, because if it were, also $\{a^n b^n : n \geq 0\}$ would be regular, which is impossible.

Showing a language is not regular

Let L_{diff} consists of strings over $\{a, b\}$ with a different number of a s as b s.

Note that $L_{\text{equal}} = \{a, b\}^* \setminus L_{\text{diff}}$.

So L_{diff} cannot be regular, because if it were, also were regular then also L_{equal} would be regular, which is impossible.

Summary

- We used the pigeonhole principle to give a direct proof that certain languages are not regular.
 - A variation is called the *pumping lemma*, see Sipser.
- We used the closure of the regular languages under Boolean operations to deduce that certain languages are not regular.

Where next?

So, to recognise more languages that are not regular we need a more powerful model of computation. Next we will study **context-free grammars** which are a different style of model than machine models, and can describe more than just the regular languages.

Extra slides

$L = \{w \in \{a, b\}^* : |w| \text{ is a power of } 2\}$ is not regular

Proof.

1. Let x_i be any string of length 2^i , for $i \in \mathbb{N}$
2. Given i, j , say $i < j$, let z be any string of length 2^i .
3. Then $x_i z \in L$ since $|x_i z| = 2^i + 2^i = 2^{i+1}$.
4. And $x_j z \notin L$ since

$$2^j < |x_j z| = 2^j + 2^i < 2^j + 2^j = 2^{j+1}$$

so $|x_j z|$ is not a power of 2 (because there is no power of 2 between 2^j and 2^{j+1}).



Good to know

Let L be a language over Σ .

We say that two strings $x, y \in \Sigma^*$ are **distinguished by L** if there is some string $z \in \Sigma^*$ such that one of the strings xz and yz is in L and the other is not in L .

Our proof that L is not regular amounts to finding infinitely many strings x_1, x_2, x_3, \dots that are pairwise distinguished by L (i.e., such that any two of them are distinguished by L).