6 Nonregular Languages

In this section we see how to prove that a language is not regular.

BIG IDEA: You *can* prove a negative.

6.1 A Motivating Example

Notation a^n means a string of n consecutive a's. For example, a^1 = "a", a^2 = "aa" and a^3 = "aaa". It is easy to design a finite-state machine that solves language

$$L_1 = \{a^m b^n \mid m > 0 \text{ and } n > 0\}.$$

A string s is in L_1 if and only if s consists of some positive number of a's followed by a positive number of b's. But suppose that

$$L_2 = \{a^n b^n \mid n > 0\}.$$

Notice that a string s is in L_2 if and only if s consists of some positive number of a's followed by *the same number* of b's. L_2 = {"ab", "$aabb$", "$aaabbb$", ...}.

Suppose that you want to design a finite state machine M where $L(M) = L_2$. What information does M need to remember? What if M reads a string of n a's and the next symbol is a b? M must remember n. If it doesn't, then how will M be able to check whether there are exactly n b's?

So it seems that M must have a state remembering that it has read exactly 1 a, another state remembering that it has read exactly 2 a's, another remembering that it has read exactly 3 a's, etc., without any limit. But that requires infinitely many states!

Can we conclude that L_2 is not regular? Be careful! Many incorrect "proofs" have been proposed that follow the rough outline: "I can only see one way to solve this problem. That way does not work. Therefore, this problem is unsolvable." That is nonsense. What if you missed an idea? We need a more careful proof.

6.2 A Proof Technique

The above idea about why L_2 is not regular is sound, but it needs to be presented more carefully. This section illustrates a way to show that a language is not regular (if it really isn't regular), using language L_2 as an example.

Theorem 6.1. L_2 is not regular.

Proof.

1. The proof is by contradiction. Suppose that L_2 is regular. We need to derive a contradiction by proving that **F** is true.

Know:	L_2 is regular.
Goal:	**F**.

2. Our knowlede uses term *regular*. By definition, L_2 is regular if an only if there is a DFA M where $L(M) = L_2$.

Know:	There exists a DFA M where $L(M) = L_2$.
Goal:	**F**.

3. When you know that there exists something with a particular property, ask someone else to give you such a thing. So let's ask for M, and suppose the start state of M is q_0.

Known variables:	M, q_0
Know:	$L(M) = L_2$.
Know:	q_0 is the start state of M.
Goal:	**F**.

4. This kind of proof involves a clever idea, and here it is. Our intuitive reasoning above looked at the state that M reaches after reading each of a^1, a^2, a^3, etc. So let's think about those states. In fact, since we have M in hand, we can do an experiment where we run M on each of a^1, a^2, a^3, a^4, etc. For each one, let's write the state that M reaches. That gives a table that *might* start out looking like this.

Input x	State $q_0 : x$ reached
"a"	2
"aa"	6
"aaa"	3
"$aaaa$"	9
...	...

But M only has finitely many states. By the pigeonhole principle, as you expand the table for longer and longer strings of a's, there must come a point where a state is repeated. Suppose that strings a^i and a^k take M to the same state q, where $i < k$.

Input x	State $q_0 : x$ reached
...	...
a^i	q
...	...
a^k	q
...	...

The experiment shows that $q_0 : a^i = q_0 : a^k = q$.

Known variables:	M, q_0, q, i, k
Know:	$L(M) = L_2$.
Know:	q_0 is the start state of M.
Know:	$q_0 : a^i = q$.
Know:	$q_0 : a^k = q$.
Know:	$i < k$.
Goal:	**F**.

5. Now comes a second clever trick. We have seen that M forgets the difference between a^i and a^k, since the only thing M can remember is the state that it is in. What if i b's come next? On input $a^i b^i$, M should answer yes. But on input $a^k b^i$, M should answer no.

Define $q' = q : b^i$. Recalling that $q = q_0 : a^i$ and $q = q_0 : a^k$,

$$
\begin{aligned}
q' &= q : b^i \\
&= (q_0 : a^i) : b^i \\
&= q_0 : a^i b^i \qquad \text{(by Theorem 5.5)} \\
q' &= q : b^i \\
&= (q_0 : a^k) : b^i \\
&= q_0 : a^k b^i \qquad \text{(by Theorem 5.5)}
\end{aligned}
$$

So M reaches the same state q' on input $a^i b^i$ as on input $a^k b^i$.

Suppose that q' is an accepting state. Then M correctly accepts $a^i b^i$ but incorrectly accepts $a^k b^i$.

Suppose that q' is a rejecting state. Then M correctly rejects $a^k b^i$ but incorrectly rejects $a^i b^i$.

No matter what, M does not correctly solve language L_2.

Known variables:	M
Know:	$L(M) = L_2$.
Know:	$L(M) \neq L_2$.
Goal:	**F**.

6. That gives us the contradiction: $(L(M) = L_2) \wedge (L(M) \neq L_2)$ is equivalent to **F**.

$\diamondsuit$

The above proof is actually quite constructive. Suppose that Archibald says he can produce a DFA M that solves L_2. Ask Archibald to give you M. Perform the above experiment. You find a string on which M gets the wrong answer. Sending that string to Archibald provides him with an irrefutable reason to believe that he was mistaken, and that M does not solve L_2.

6.3 Another Example

Suppose

$$L_3 = \{a^n \mid n \text{ is a perfect square}\}.$$

Theorem 6.2. L_3 is not regular.

Proof.

1. The proof is by contradiction. Suppose that L_3 is regular. We need to derive a contradiction by proving that **F** is true.

Know:	L_3 is regular.
Goal:	**F**.

2. By definition, L_3 is regular if an only if there is a DFA M where $L(M) = L_3$.

Know:	There exists a DFA M where $L(M) = L_3$.
Goal:	**F**.

3. Ask someone else to give you a DFA M where $L(M) = L_3$. Suppose the start state of M is q_0.

Known variables:	M, q_0
Know:	$L(M) = L_3$.
Know:	q_0 is the start state of M.
Goal:	**F**.

4. To employ the first clever idea, we need to find an infinite sequence of strings to try M on. The requirement is that M cannot afford to forget the difference between any two of those infinitely many strings; it needs to stop in a different state for each of them. Finding that sequence is the part of this kind of proof that requires the most thought.

   A sequence of strings that does the job is a^1, a^4, a^9, a^{16}, etc.; that is, run M on sequences of a's of lengths 1^2, 2^2, 3^2, 4^2, etc. We have M in hand, and we can do an experiment where we run M on each of those strings. The table *might* start out looking like this.

Input x	**State** $q_0 : x$ **reached**
a^{1^2}	8
a^{2^2}	1
a^{3^2}	14
a^{4^2}	6
...	...

Since M only has finitely many states, but there are infinitely many strings in the sequence, the right-hand column must eventually contain a repetition. Suppose that inputs a^{i^2} and a^{k^2} stop on the same state, q.

Input x	**State** $q_0 : x$ **reached**
...	...
a^{i^2}	q
...	...
a^{k^2}	q
...	...

Known variables:	M, q_0, q, i, k
Know:	$L(M) = L_2$.
Know:	q_0 is the start state of M.
Know:	$q_0 : a^{i^2} = q$.
Know:	$q_0 : a^{k^2} = q$.
Know:	$i < k$.
Goal:	$\mathbf{F}$.

5. For the second clever trick, we must show that the first clever trick was chosen correctly. We have seen that M forgets the difference between a^{i^2} and a^{k^2}, since the only thing M can remember is the state that it is in. Our goal is to find *one* string r where M should accept $a^{i^2}r$ but

M should reject $a^{k^2}r$. A string r that does the job is $r = a^{2i+1}$. Notice that

$$\begin{aligned} a^{i^2}r &= a^{i^2}a^{2i+1} \\ &= a^{i^2+2i+1} \\ &= a^{(i+1)^2} \end{aligned}$$

So $a^{i^2}r \in L_3$. But

$$\begin{aligned} a^{k^2}r &= a^{k^2}a^{2i+1} \\ &= a^{k^2+2i+1} \end{aligned}$$

But $i < k$, so

$$\begin{aligned} k^2 &< k^2 + 2i + 1 \\ &< k^2 + 2k + 1 \\ &= (k+1)^2 \end{aligned}$$

Since there are no perfect squares between k^2 and $(k+1)^2$, $k^2 + 2i + 1$ cannot be a perfect square. That means $a^{k^2}r \notin L_3$.

Recall that M stops in the same state, q, on input a^{i^2} as on input a^{k^2}. Therefore, it stops on the same state $q' = q : r$ on input $a^{i^2}r$ as on input $a^{k^2}r$.

If q' is an accepting state, then M correctly accepts $a^{i^2}r$ but incorrectly accepts $a^{k^2}r$.

If q' is an rejecting state, then M correctly rejects $a^{k^2}r$ but incorrectly rejects $a^{i^2}r$.

No matter what, there is an input on which M gives the wrong answer. So $L(M) \neq L_3$.

Known variables:	M
Know:	$L(M) = L_3$.
Know:	$L(M) \neq L_3$.
Goal:	**F**.

6. That gives us the contradiction: $(L(M) = L_3) \wedge (L(M) \neq L_3)$ is equivalent to **F**.

$\diamondsuit$

6.4 Yet Another Example

Suppose

$$L_4 = \{ww \mid w \in \{a, b\}^*\}.$$

Strings in L_4 include "aa", "$abab$", "$aabbbaabbb$" and "$bbaabbaa$", among infinitely many others.

Theorem 6.3. L_4 is not regular.

Proof.

1. As before, the proof is by contradiction. Suppose that L_4 is regular. We need to derive a contradiction by proving that **F** is true.

Know:	L_4 is regular.
Goal:	**F**.

2. By definition, L_4 is regular if an only if there is a DFA M where $L(M) = L_4$.

Know:	There exists a DFA M where $L(M) = L_4$.
Goal:	**F**.

3. Ask someone else to provide us with a DFA M where $L(M) = L_4$. Suppose the start state of M is q_0.

Known variables:	M, q_0
Know:	$L(M) = L_4$.
Know:	q_0 is the start state of M.
Goal:	**F**.

4. We need to find an infinite sequence of strings to try M on, where M cannot afford to forget the difference between any two of those strings. A sequence that works is a^1b, a^2b, a^3b, etc. Let's try running M on those strings and write down the state that M reaches for each of them. The experiment *might* yield the following.

Input x	**State** $q_0 : x$ **reached**
"ab"	1
"aab"	2
"$aaab$"	3
"$aaaab$"	4
...	...

But M only has finitely many states. As you expand the table for longer and longer strings, there must come a point where a state is repeated. Suppose that strings a^ib and a^kb take M to the same state q.

Input x	**State** $q_0 : x$ **reached**
...	...
a^ib	q
...	...
a^kb	q
...	...

The experiment shows that $q_0 : a^ib = q_0 : a^kb = q$.

Known variables:	M, q_0, q, i, k
Know:	$L(M) = L_4$.
Know:	q_0 is the start state of M.
Know:	$q_0 : a^ib = q$.
Know:	$q_0 : a^kb = q$.
Know:	$i < k$.
Goal:	**F**.

5. M forgets the difference between a^ib and a^kb, since the only thing M can remember is the state that it is in. What if string $r = a^ib$ comes next? On input a^iba^ib, M should answer yes, since $a^iba^ib = ww$ where $w = a^ib$. But on input a^kba^ib, M should answer no, since there does not exist any string w where $a^kba^ib = ww$. But M reaches the same state $q' = q\mathbin{:}r$ on input a^iba^ib as on input a^kba^ib. If q' is an accepting state, then M incorrectly accepts a^kba^ib. If q' is a rejecting state, then M incorrectly rejects a^iba^ib. So M does not solve L_4.

Known variables:	M
Know:	$L(M) = L_4$.
Know:	$L(M) \neq L_4$.
Goal:	**F**.

6. That gives us the contradiction.

♢

I have done these proofs in with a lot of detail. Here is the the previous proof done in a more typical way.

Theorem 6.4. L_4 is not regular.

Proof. Suppose L_4 is regular. Let M by a DFA that solves L_4.

Imagine doing an experiment where you run M on strings a^1b, a^2b, a^3b, and so on. Because M has finitely many states, eventually values i and k must be found, with $i < k$, where a^ib and a^kb take M to the same state q.

Now imagine running M on inputs a^ibr and a^kbr where $r = a^ib$. Since M reaches the same state on inputs a^ib and a^kb, M must also reach the same state on inputs a^ibr and a^kbr. So M accepts both a^ibr and a^kbr or it rejects both. But that means that M does not solve L_4, since $a^iba^ib \in L$ and $a^kba^ib \notin L$. That is a contradiction.

♢

6.5 A Common Mistake

Suppose $L_5 = \{a^na^n \mid n > 0\}$. Let's try to prove the following.

Claim. L_5 is not regular.

"Proof."

1. The proof is by contradiction. Suppose that L_5 is regular. We need to derive a contradiction by proving that **F** is true.

Know:	L_5 is regular.
Goal:	**F**.

2. By definition, L_5 is regular if and only if there is a DFA M where $L(M) = L_5$.

Know:	There exists a DFA M where $L(M) = L_5$.
Goal:	**F**.

3. Ask someone else to give such you a DFA M where $L(M) = L_5$, and suppose the start state of M is q_0.

Known variables:	M, q_0
Know:	$L(M) = L_5$.
Know:	q_0 is the start state of M.
Goal:	**F**.

4. Do an experiment using M. Run M on strings a^1, a^2, a^3, etc. and record the state reached for each string. Continue until a state q has been written twice, which must happen because M has finitely many states.

Input x	**State** $q_0 : x$ **reached**
...	...
a^i	q
...	...
a^k	q
...	...

Known variables:	M, q_0, q, i, k
Know:	$L(M) = L_2$
Know:	q_0 is the start state of M
Know:	$q_0 : a^i = q$
Know:	$q_0 : a^k = q$
Know:	$i < k$
Goal:	**F**

5. Now we need to find a string r so that $a^i r \in L_5$ but $a^k r \notin L_5$. Choose $r = a^i$.

   Notice that $a^i r = a^i a^i$ and that is in L_5 from the definition of L_5.

   Notice that $a^k r = a^k a^i$. But that does not have the form $a^n a^n$ so $a^k r \notin L_5$.

   As before, that leads to a contradiction.

$\diamondsuit$

But that "proof" cannot be correct. $\{a^n a^n \mid n > 0\} = \{a^{2n} \mid n > 0\}$. So L_5 is the set of all strings of a's whose length is even, and that is a regular language. Where did the proof go wrong?

The incorrect proof states that $a^k a^i$ does not have the form $a^n a^n$. Suppose $k = 4$ and $i = 2$. Then $a^k a^i = a^4 a^2 = a^6 = a^3 a^3$. In fact, as long as $i + k$ is even, $a^k a^i$ *does* have the form $a^n a^n$, where $n = (i + k)/2$.

Can you insist that $i + k$ is odd? Clearly not. The claim is false. It is pointless to try to modify the proof since the claim is false.

6.6 Be Careful Not to Be Sloppy

Having seen a few proofs like the above, all using similar ideas, it is easy to get the idea that it is not necessary to write out all of the details, and instead to skip directly to step 5. But step 4 says what the experiment is; that is, what is the infinite sequence of strings to run M on? If you don't say what the experiment is, you will find yourself making inconsistent statements about that experiment.

There is an easy way to avoid that. Don't skip the details. Write them down and check that what you have written is sensible. Look at an example. (We found that the above incorrect proof was not right by looking at the example $i = 2$ and $k = 4$.)

You don't need to write out tables of known things and goals, as in our very detailed proofs. Use the typical (shorter) proof style. But don't expect a person who reads your proof to fill in important details, such as the nature of the experiment, or why one string is in $L(M)$ while the other is not.

prev **next**