COMS W3261, Section 1, Spring 2024
Lecture : Lower bounds review

Instructor : Josh Alman
Notes by : William Pires

We will consider that the alphabet is $\Sigma = \{a, b\}$.

Definition 1. A string $w \in \Sigma^*$ is "*nice*" if it has even length and at least as many bs in the first half as in the second half.

Let $L := \{w \in \Sigma^* \mid w \text{ is nice}\}$. We will show in 3 different ways that L isn't regular.

Theorem 1. L isn't regular.

1 Via Pumping Lemma

Recall the pumping lemma :

Theorem 2 (Pumping Lemma)**.** Let L be a regular language over Σ. If L is regular, then there exists a positive integer p such that for any $s \in L$ with $|s| \geq p$, there are $x, y, z \in \Sigma^*$ such that :

(1) $s = xyz$

(2) $|y| > 0$

(3) $|xy| \leq p$

(4) For all $i \geq 0$, $xy^iz \in L$

Assume for contradiction that L is regular and let p be the pumping length [1] and consider $s = a^pbba^p$. [2]

[1] In any pumping lemma proof DO NOT choose what p is. For instance if you show a contradiction by assuming $p = 4$, then you show there's no DFA with 4 states for the language. But since we want to show no DFA exists, you need to let p be arbitrary.

[2] The most tricky part of pumping lemma proofs is picking the correct string s to pump. The string s must be in L, and usually it should "barely be in L", so that changing things makes it not be in L.

By the pumping lemma, there are x, y, z with $a^p bba^p = xyz$ and condition $(2), (3), (4)$.

What can y be ? [3] We must have $y = a^m$ for some $0 < m \leq p$. Why ? By (3), $|xy| \leq p$, so xy consist of only as, so y is made of as only. We must have $m > 0$ by (2).

By (4) setting $i = 3$, we have $xy^3z = a^{p+2m}bba^p \in L$. xy^3z has length $p+2m+2+p = 2p+2m+2$ so the first half is the first $p+m+1$ symbols. Since $m \geq 1$, we have $p+m+1 \leq p+2m$. So, the first half of the string is in the a^{p+2m} part (it contains only a's) while the second half must contain the two bs. So $xy^3z \notin L$. This contradicts the pumping lemma, hence L isn't regular.

2 Via Streaming

Recall the following :

Theorem 3. A language L is regular if and only if it has a constant space streaming algorithm.

So, we can prove that $L := \{w \in \Sigma^* \mid w \text{ is nice}\}$ isn't regular by showing a streaming lower bound for it. What's the best upper bound for L ? Surely it has a $O(n)$ space algorithm. It turns out this is optimal, no algorithm uses $o(n)$ space, but this really hard to prove. However, it's enough to prove a $\Omega(\log(n))$ space lower bound to show that L isn't regular. Recall the following:

Definition 2 (Length n-distinguishing)**.** Let L be a language over Σ^*. A length n-distinguishing set $S_n \subseteq \Sigma^*$ is such that for any distinct $x, y \in S_n$ we have :

- Exactly one of xz or yz is in L.
- $|xz| \leq n$ and $|yz| \leq n$.

Theorem 4. If a language L has a length-n distinguishing set S_n, then any streaming algorithm for L must use $\Omega(\log(|S_n|))$ space for inputs of length $\leq n$.

Let

$$S_n := \{b^m \mid 0 \leq m < \frac{n}{5}\}$$

[3] It's critical to consider all the possible cases for what y could be. Do not make assumptions besides what $(2), (3)$ tell you. For example saying $x = \epsilon, y = a$ would be wrong. And make sure you don't forget any case.

We claim this is a length-n distinguishing set for L. Consider distinct $x, y \in S_n$. We have $x = b^k$, $y = b^\ell$, wlog we have $0 \leq k < \ell \leq \frac{n}{5}$.

Pick $z = a^{n-2\ell}b^\ell$ [4]. Then

$$xz = b^k a^{n-2\ell} b^\ell \text{ and } yz = b^\ell a^{n-2\ell} b^\ell$$

Because $\ell < n/5$, it's easy to check that for both these of strings the middle of the string is in the $a^{n-2\ell}$ part. So, xz has k bs on the left, ℓ bs on the right, so $xz \notin L$. However, yz has ℓ bs on the left, and ℓ bs on the right, so $yz \in L$.

We clearly have $|xz|, |yz| \leq n$.

Since S_n is a distinguishing set of size $\frac{n}{5}$ any streaming algorithm for L needs space $\Omega(\log(|S_n|)) = \Omega(\log(n/5)) = \Omega(\log(n))$. Thus, L isn't regular.

3 Via Communication

In homework 3, you prove the following theorem.

Theorem 5. If $L \subseteq \Sigma^*$ is a regular language, then there is $O(1)$ cost protocol for $f_L : \Sigma^* \times \Sigma^* \to \{0, 1\}$ which is defined as

$$f(x, y) = \begin{cases} 1 \text{ if } xy \in L \\ 0 \text{ otherwise} \end{cases}$$

So to show that L isn't regular, we can show $\mathsf{cc}(f_L)(n) = \omega(1)$[5].

Note that the other direction isn't true. Some languages have $\mathsf{cc}(f_L) = O(1)$ but L isn't regular. So this approach doesn't work for all regular languages. We will need the following definition. Again, you don't have to use this definition in your proofs. You can use the same proof as Josh did for EQUALITY, which also involves coming up with a set S, and then using the pigeonhole principle arguing that for two pairs in the set Alice and Bob send the same transcript. But using this definition would probably be easier.

Definition 3. A fooling set $S_n \subseteq \Sigma \in \Sigma^n \times \Sigma^n$ is a set such that such there is $b \in \{0, 1\}$:

- For all $f(x, y) \in S_n$ we have $f(x, y) = b$.
  That is, everything in the set is mapped to the same output b.

[4] If n is odd, pick $z = a^{n-2\ell-1}b^\ell$ instead and the proof would still work.
[5] If you're confused by the ω notation, this just means showing that $\mathsf{cc}(f_L)(n)$ can't be constant.

- For all distinct $(x_1, y_2), (x_2, y_2) \in S_n$ then $f(x_1, y_2) \neq b$ or $f(x_2, y_1) \neq b$ (or both).

  That is, if you pick two distinct pairs and swap their first elements, for one of the resulting pairs, it's not mapped to b.

Finally, we have the following :

Theorem 6. If f has a fooling set S_n, then any protocol P that computes f must have $\mathsf{cost}(P)(n) = \Omega(\log(|S_n|))$. This means $\mathsf{cc}(f)(n) = \Omega(\log(|S_n|))$.

So it's enough to give a size a fooling set S_n for f_L of size cn for some constant c, this would imply $\mathsf{cc}(f_L)(n) = \Omega(\log(n))$, and thus that L isn't regular.