

How to Learn by Counting; Finite/Infinite; Pumping Lemma

JT

October 4, 2007

Contents

1	For orientation	1
2	Counting arguments	2
2.1	Size of infinite sets	3
2.2	Counting argument # 1: different orders of infinity	7
2.3	Languages with infinite alphabets	9
3	The Pumping Lemma	10
3.1	The Lemma itself	10
3.2	Using the Lemma to Prove Languages Nonregular	12

1. For orientation

The university has various distribution requirements for graduation. These are based on an old-fashioned idea that there are certain ideas, experiences and skills that are so important to a student's intellectual development that without them, an education is incomplete. For example, a student should have the experience of learning a foreign language, and have read and reflected on a range of classic works of literature, and have attained at least a rudimentary grasp of at least some of the basic physical and biological facts about the world around us; without these things and more an education is impoverished and one - sided. I'd add to this list the experience of studying and dissecting some real mathematical theorems. It isn't easy to say precisely what makes an argument a real theorem rather than a simple exercise: the point is that a real theorem has a kind of complexity and depth in its solution. Even if it is easy to understand, it requires profound ideas for its proof, and these ideas are independently worth knowing.

These notes are aimed at proving a simple example of a real theorem, called the Pumping Lemma. In fact, the ideas we use to prove the pumping

lemma are even more surprising and interesting than the lemma itself. The main lessons I want you to take from these notes are these: i) There's more regularity and order inherent in the idea of infinity than you would have guessed beforehand ii) Good old familiar *counting* is more complicated and powerful than you might have guessed beforehand.

The Pumping Lemma is also a handy result when we are trying to prove a language *isn't* regular. Given that the regular languages are exactly the languages accepted by FSA's we have two simple strategies for proving that a set of strings is a regular language: produce a regular expression for the set, or design an FSA that accepts it. But we typically need a bit more finesse if we want to prove that a language isn't regular.

2. Counting arguments

One kind of argument that is so familiar in everyday life that we hardly notice it involves counting. If we count five people in the party and we have only four hamburgers then we know that at least one person won't get a whole hamburger. In a room with 367 people, we can be absolutely sure that at least one pair of people in the room have the same birthday. We can be sure that some A is not a B if there are 12 A's and 10 B's. Many simple facts about abstract languages can be worked out just by more general versions of this method. Say that I have a language containing just 5 strings, and I want to talk about 15 objects. I know just by counting that some object won't have a name.

A particular counting principle that is used frequently is called the *pigeonhole principle*: If you have $n+1$ pigeons and you are assigning them to n pigeonholes, then at least one pigeonhole will get more than one pigeon. (The observation in the last paragraph that given 367 people, at least two will share a birthday, is a simple example of the pigeonhole principle at work.) This is an obvious fact, but it turns out to be unexpectedly crucial.

Sometimes, with counting arguments, we can assign precise numbers, as in the case of 367 people in the room, versus 366 days in a leap year. In other cases, we may count in a more general way: We know that at least one A isn't a B if we have m A's and n B's and we know that $m < n$. In other cases, the counting is even more approximate: if we can attach a precise sense to "not very many" and "lots", then we can establish that at least one A isn't a B if we can establish that there are "lots" of A's but "not very many" B's. The simplest examples involve finite and infinite collections: we know that some A is not a B if we can show that there are infinitely many A's but only

finitely many B's.¹ Also, we can be sure that some A is not a B if there are infinitely many B's but the set of A's is - so to speak - even *more* infinite. To understand this last point, I'll have to explain what is meant by "more infinite".

2.1. Size of infinite sets

Many ideas that clump together when we deal with simple everyday concepts come apart when we consider infinite sets. So, for example, there are several different senses of "size" when we're speaking of collections, so the first job is to be clear about what we mean. When are two collections "the same size" or "different sizes" in the relevant sense? When is one set "bigger" than another? One natural candidate is given by the subset relation: we might naturally say that A is bigger than B if $B \subset A$.² Here we'll be considering the size of sets in a *different* sense, defined by the possibilities of pairing up members of sets with one-to-one functions.

The core idea is simple enough: you can tell if you have (say) more knives than forks if you pair them up, setting each knife beside exactly one fork. If you can match them up exactly, with no forks or knives left over, then you have the same number of knives as forks. If you can't set up such a one-to-one pairing, then there is more in the set with a few things left over. To capture this idea, we can define:

Z and Y are *the same size* (or: *have the same number of elements*) if there is a one-to-one function mapping Z to Y so that every element of Y is the partner of some element of Z.

Z is *bigger than* Y (or: *has more elements than* Y) if i) there is a one-to-one function mapping some *subset* of Z to Y so that every element of Y is the partner of some element of Z, but ii) there is no one-to-one function mapping *all of* Z to Y so that every element of Y is the partner of some element of Z. (Recall that when every member of the range Y has some element of the domain Z paired to it by f , we say f is an *onto* function and that it maps Y *onto* Z.) This fits with the way we would use these words with everyday

¹Reflecting this, mathematicians often say "Almost all..." when they mean "Except for at most finitely many exceptions". In cases where uncountable sets are under discussion, "almost all" can mean "Except for countably many exceptions". ("Countable" and "Uncountable" are defined in the next section.)

²We used this sense of "size" when speaking of an inductively defined set as the "smallest" set containing some given set and closed under given operations. In that case a set S counts as the "smallest" set satisfying some conditions if i) S satisfies the conditions and ii) if S' is any other set that satisfies the conditions then $S \subseteq S'$. In these notes we will be using "smaller" and "bigger" to mean something different.

finite sets: if you can set every fork beside a unique knife, but you can't pair up every knife beside a unique fork without having some knives left over, then you have more knives than forks.

A nice feature of these definitions is that they apply to infinite sets as well as finite sets like the knives and forks around a table. With this understanding of the relevant concepts, any infinite set has more members than any finite set, which is what we would want. It also turns out that many infinite sets have "the same size" in the one-to-one function sense, even if they have different sizes in some other senses. An easy example is that the even numbers have the same size as the natural numbers \mathbb{N} . Let f be the function $f(x) = 2x$. This pairs up the even and natural numbers like this:

$0 \leftrightarrow 0$
 $1 \leftrightarrow 2$
 $2 \leftrightarrow 4$
 $3 \leftrightarrow 6$
 $4 \leftrightarrow 8$
 $\vdots \leftrightarrow \vdots$

Since $\{x/x \text{ is even}\} \subset \mathbb{N}$, this gives us an example of two sets which are different sizes in one sense, but the same size in the sense we are currently using. Some terminology will be useful: if a set S is finite or the same size as the natural numbers \mathbb{N} , we say that S is *countable*. A set that isn't countable, we'll call *uncountable*.

It might be a little surprising to learn that the rational numbers (standardly written as \mathbb{Q} , for "quotients") are the same size as the natural numbers \mathbb{N} . It's easy to show this. Every rational number can be written as a fraction $\frac{m}{n}$. To avoid duplicates like $\frac{1}{2}$ and $\frac{2}{4}$, say that we consider only the fraction in which the numerator and denominator are in lowest terms. Then we can define: $f(\frac{m}{n}) = 2^m \cdot 3^n$. This correlates every member of \mathbb{Q} with a unique member of \mathbb{N} . The image of f is a proper subset of \mathbb{N} - it contains just natural numbers that are only evenly divisible by 2 or 3 - but it is easy to use f to define the function f' onto the whole set \mathbb{N} : $f'(\frac{m}{n}) =$ the number k such that $f(\frac{m}{n})$ is the k 'th number in the image of f .

On reflection, it might seem not so surprising after all that \mathbb{Q} and \mathbb{N} have the same size. After all, they are infinite. Infinite is as big as you can get, right? You can't get bigger than infinity, right? So you might want to say: shouldn't *all* infinite sets be the same size? Quite the contrary - as I mentioned above, some infinite sets are bigger than others. So, for example, the set of real numbers (\mathbb{R}) is bigger than \mathbb{N} and even the set of real numbers

between zero and one is bigger than \mathbb{N} . There is a famous argument (famous, at least, among people who regard arguments as things that can be famous) called the “diagonal argument” that proves \mathbb{R} is bigger than \mathbb{N} . I won’t look at the argument here, but I should say that it is short and charming, easy for a beginner to understand, and worth looking up. A Google search on “real numbers” and “Cantor’s diagonal argument” will turn up many pages devoted to it. (Be careful, though, since the diagonal argument is a magnet for cranks and loons. It is the abstract realm’s answer to the stories about UFO landings in New Mexico.)

Here I’ll consider a variation on the diagonal argument, to show that the power set of \mathbb{N} is bigger than \mathbb{N} . (Recall that the power set $\mathcal{P}(X)$ of a set X is the set of subsets of X . In symbols, $\mathcal{P}(X) = \{Y/Y \subseteq X\}$.)

Claim: There is **no** 1 - 1, onto function f mapping \mathbb{N} onto $\mathcal{P}(\mathbb{N})$.

Proof of the claim:

Here is how this proof works: you *assume* that there is such a function f and you derive a contradiction from that assumption. Thus you can conclude the original assumption is untrue. (This is known as a proof by contradiction, or, to use the classical phrase, *Reductio ad absurdum*).

Assumption (To be reduced to a contradiction): Say that there is a one-one, onto function f , such that $f : \mathbb{N} \rightarrow \mathcal{P}(\mathbb{N})$. Every set in $\mathcal{P}(\mathbb{N})$ is therefore assigned some number in \mathbb{N} . To choose an arbitrary example for illustration, the first few entries of such a list might be:

$0 \leftrightarrow \emptyset$
 $1 \leftrightarrow \{1, 3\}$
 $2 \leftrightarrow \{2, 3, 5\}$
 $3 \leftrightarrow \{x/x \text{ is even}\}$
 $4 \leftrightarrow \{x/x \text{ is odd}\}$
 $\vdots \leftrightarrow \vdots$

Since each number is assigned a subset of \mathbb{N} , we can separate \mathbb{N} into two categories: those numbers n that **are** members of $f(n)$, and those numbers n that are **not** members of $f(n)$. Let’s look more closely at the set of numbers n that are **not** in the corresponding $f(n)$. That is, let’s look at the set we’ll

call D (for “diagonal set”):³

$$D = \{n/n \in \mathbb{N} \& n \notin f(n)\}$$

D is a set of numbers; that is, $D \subseteq \mathbb{N}$. In other words, D is a member of $\mathcal{P}(\mathbb{N})$. So D must be paired with some number k by f . (That is: $f(k) = D$ for some $k \in \mathbb{N}$.)

Let \tilde{k} be the number such that $f(\tilde{k}) = D = \{n/n \in \mathbb{N} \& n \notin f(n)\}$. We get the contradiction we are looking for if we ask the obvious question:

Is \tilde{k} an element of $\{n/n \in \mathbb{N} \& n \notin f(n)\}$? That is: is $\tilde{k} \in f(\tilde{k})$?

If we say yes, then we have to say no, and if we say no, then we have to say yes! That is:

Possibility 1: Say that $\tilde{k} \in f(\tilde{k})$.

(In other words: $\tilde{k} \in \{n/n \in \mathbb{N} \& n \notin f(n)\}$)

That means, it is **not** true that $\tilde{k} \notin f(\tilde{k})$

But then, by the *definition* of the set $\{n/n \in \mathbb{N} \& n \notin f(n)\}$, we know that:

$$\tilde{k} \notin \{n/n \in \mathbb{N} \& n \notin f(n)\}$$

But $f(\tilde{k}) = \{n/n \in \mathbb{N} \& n \notin f(n)\}$, so:

$\tilde{k} \notin f(\tilde{k})$, which contradicts what we assumed.

Possibility 2: On the other hand, say that $\tilde{k} \notin f(\tilde{k})$.

(In other words: $\tilde{k} \notin \{n/n \in \mathbb{N} \& n \notin f(n)\}$)

But then, by the *definition* of the set $\{n/n \in \mathbb{N} \& n \notin f(n)\}$, we know that:

$$\tilde{k} \in \{n/n \in \mathbb{N} \& n \notin f(n)\}$$

³It isn't obvious in this example why “diagonal set” is a good name, but I'm sticking with it because many textbooks use it. If you look up the proof that there are more real numbers than natural numbers, the reference to a diagonal will be more easily understood.

But $f(\tilde{k}) = \{n / n \in \mathbb{N} \& n \notin f(n)\}$, so:

$\tilde{k} \in f(\tilde{k})$, which contradicts what we assumed.

So it is impossible for \tilde{k} to be either a member or not a member of $\{n / n \in \mathbb{N} \& n \notin f(n)\}$

This is the contradiction we've been looking for. We got this contradiction by assuming that we could find some 1-1, onto function f from \mathbb{N} to $\mathcal{P}(\mathbb{N})$. We can therefore conclude that **there is no such function**. Since it's easy to find a 1-1, onto function from a subset of $\mathcal{P}(\mathbb{N})$ onto \mathbb{N} , we can conclude that in the relevant sense, $\mathcal{P}(\mathbb{N})$ is bigger ("more infinite") than \mathbb{N} .

An aside about generality:

To be properly aware of the logical structure of an argument, you need to always ask yourself "Where does this particular fact get used?" Perhaps the argument is more general than you realize. If you think that way about the argument we've just looked at, you'll ask "Where do we use the fact that the set we are dealing with is \mathbb{N} ?" In fact, we don't use that fact anywhere: this argument gives a general proof, for any set X (finite or infinite), that $\mathcal{P}(X)$ is bigger than X .

2.2. Counting argument # 1: different orders of infinity

We can use the above idea to prove that there are languages over finite alphabets that are **not** recognized by FSA's. We do this by counting: The set of possible languages is uncountable (it is as infinite as $\mathcal{P}(\mathbb{N})$), while the set of languages recognized by FSA's is countable (i.e. it is only the size of \mathbb{N}).

Here is how this works. For simplicity, let's consider just languages over the alphabet $\{a, b\}$. It is easy to show that the set of strings in the language over $\{a, b\}$ has the same size as \mathbb{N} . For example, we can associate 'a' with 1 and 'b' with 2, and then associate a string of a's and b's with the corresponding string of 1's and 2's. So for example the string 'abbabb' would be associated with the number 122122, and the string 'aaaab' would be associated with 11112. I could give a complete definition of this association, but it would be wordy, and I expect you get the idea fine without it. This gives us a mapping of the strings in $\{a, b\}^*$ onto a subset of \mathbb{N} . We can turn that into a mapping onto \mathbb{N} : $f^*(\sigma) = n$, where the number associated with σ is the n 'th number associated with the strings over $\{a, b\}$.

Remember that f^* is what we are calling the function that maps the strings over $\{a, b\}$ onto \mathbb{N} .

Now that we have counted the strings, that gives us a tally of the languages too: Since a language in an alphabet is any subset of the set of all strings in that alphabet, there will be as many languages over $\{a, b\}$ as there are subsets of the set of strings over $\{a, b\}$. In fact, we can specify a one-to-one, onto function \tilde{f} from the subsets of the set of strings over $\{a, b\}$ onto the subsets of \mathbb{N} , defining it in the obvious way - every set of strings is paired up with the set of numbers that f^* assigns the strings:

For X a set of strings over $\{a, b\}$, $\tilde{f}(X) = \{f^*(\sigma) / \sigma \text{ is a string in } X\}$

The next step is to show that the set of languages recognized by FSA's has the same size as \mathbb{N} . This can be done in more or less the same way we showed that the set of strings over $\{a, b\}$ has the same size as \mathbb{N} . Instead of counting the languages recognized by FSA's, we just need to count the FSA's.⁴ Note that each FSA has a description in terms of states, transitions, etc. that we can write as a string of letters in a finite alphabet. As above, we can assign each *description* a unique number, and then use the order of those numbers to fix a mapping from the set of FSA's onto \mathbb{N} . There are many ways to do this, but the details tend to get complicated, so at this point I'll just ask you to take my word for it: The set of FSA's is countable.⁵

Taking stock: we have shown that the set of languages over $\{a, b\}$ is the same size as $\mathcal{P}(\mathbb{N})$, and the set of FSA's (and hence, of languages recognized by FSA's) has the same size as \mathbb{N} . By simple (infinite) counting, then, we know there is *at least one* language that is not recognized by an FSA.

This is a paradigm of a kind of argument logicians call *nonconstructive*. It shows that *there is* some non-regular language, but it doesn't show you how to produce an example. We don't yet have a technique to tell, when presented with a language, if that language is regular or not. In lots of cases we can show that a language *is* regular, if we can figure out a regular expression that denotes it or an FSA that recognizes it, but it's harder to show a language to be non-regular.

⁴Since each FSA recognizes exactly one language there are at least as many FSA's as there are languages recognized by FSA's.

⁵To get the strategy for coding up the description of a machine as a natural number, find any advance logic textbook and look up "Gödel numbering". Or Google "Godel numbering" ("Gödel numbering" or "Goedel numbering" can also work) and check the first few pages you find. One of them will probably have a description of what to do.

To consider just one example: we are told in the text that the language DUPLICATES is non-regular. (DUPLICATES consists of all strings xx , where x is a string over the alphabet $\{a, b\}$). But how can we show this? The next topic is a tool that can sometimes allow us to prove that specific languages are not regular (not recognized by FSA's).

2.3. Languages with infinite alphabets

In the last subsection we proved and used the fact that given a finite alphabet, there are only countably many finite strings over that alphabet. In fact, even if the alphabet is *infinite*, but countable, the number of strings over the alphabet will be countable. We don't need this fact here, but it will be useful in later chapters, and it is important enough that we should digress for a moment to prove it. Say we have an infinite alphabet that has the same size as \mathbb{N} : $\{a_1, a_2, a_3, \dots, a_n, \dots\}$

The trick we used above - replacing numbers with digits - will work for languages with alphabets of ten letters or less. We'll need to do something different if there are more than ten letters. We can use the following fact about natural numbers, which is one of the fundamental properties of numbers used in modern systems of codes:

Prime Decomposition of Natural Numbers: Every natural number can be factored as a product of prime numbers in exactly one way.

So for example, $6 = 2 \cdot 3$, $72 = 2^3 \cdot 3^2$, $980 = 2^2 \cdot 5 \cdot 7^2$. The general pattern is: for every $n \in \mathbb{N}$, there are distinct prime numbers p_1, p_2, \dots, p_m and natural numbers $\alpha_1, \alpha_2, \dots, \alpha_m$ such that $n = p_1^{\alpha_1} \cdot p_2^{\alpha_2} \cdot \dots \cdot p_m^{\alpha_m}$, and this decomposition into prime numbers is *unique*.

This allows us to define a 1-1, onto function from the strings in our alphabet to the natural numbers as follows. First assign each letter in the alphabet a number. Since the letters in our alphabet have subscripts, let's make it easy and assign each letter its subscript:

$$f(a_i) = i$$

Then we assign strings numbers by exponents in the prime decomposition according to the rule:

$$a_{x_1} a_{x_2} \dots a_{x_n} \leftrightarrow 2^{x_1} \cdot 3^{x_2} \cdot 5^{x_3} \dots \cdot p_n^{x_n} \text{ (where } p_n \text{ is the } n\text{th prime number).}$$

This pairs a unique number to each string in $\{a_1, a_2, a_3, \dots, a_n, \dots\}^*$.

3. The Pumping Lemma

3.1. The Lemma itself

This result is nice for many reasons, including that the proof displays the importance of counting, our present theme. Also, once we have this result in hand, we'll have a way of showing that specific languages are not regular/not recognized by an FSA. Here is a statement of the theorem, called *The Pumping Lemma*:

Say that \mathcal{L} is a language accepted by a deterministic FSA, and that \mathcal{L} is *infinite*. Show that there are strings x , y and z , with y not the empty string, such that $x \underbrace{y \dots y}_n z$ is a member of \mathcal{L} , for every $n \geq 0$. That is, for every $n \geq 0$, \mathcal{L} contains a string consisting of x concatenated with n instances of y and then concatenated with z .

Here is an example to illustrate. Let the alphabet be $\{a, b, c\}$ and consider the set of strings $\{\sigma / \sigma \text{ has an even number of a's}\}$. This set is accepted by an FSA and it is infinite. So the pumping lemma applies, and we can expect to find an infinite collection of strings of the form $x \underbrace{y \dots y}_n z$ (with y not the empty string) in $\{\sigma / \sigma \text{ has an even number of a's}\}$. There are many possible choices. For example, say we set $y = aa$, $x = \epsilon$, and $z = \epsilon$. The set $\{x \underbrace{y \dots y}_n z / n \in \mathbb{N}, y = aa, x = \epsilon, \text{ and } z = \epsilon\}$ will be the set of strings of the form $\underbrace{aa \dots aa}_{2n \text{ a's}}$ for some n . This is obviously an infinite set of strings, contained in the set of strings with an even number of a's. Other choices of x, y, z will also work: for example, if $y = aaaa$, $x = bb$, and $z = bb$ we get a different infinite set of strings of the form $x \underbrace{y \dots y}_n z$, also contained in the set of strings with an even number of a's.

Proof:

Outline: You need three separate mini-arguments:

a) Using a counting argument, show that since the number of states in F is finite (say it is k), and the number of strings in \mathbb{L} less than any given length l is finite, and so the number of strings in \mathbb{L} of length greater than l is infinite, there must be a string $\sigma = \sigma_1 \dots \sigma_{k+1}$ with more letters than there are states in F .

b) observe by another counting argument (pigeonhole principle) that in the computation reading σ , some state (say it's q_i) occurs at least twice.

c) Note that this means that the computation reading σ must "loop around" from q_i , possibly through other states, and then back to q_i . If the string read in the loop occurs multiple times in succession, F must read it "around the loop" several times in succession. So let the string that is read along this loop be the "y" in the statement of the theorem.

Actually the outline is not that far from the rigorous argument; you don't need much more detail. Now let's spell it out.

a) The language \mathbb{L} is infinite, by hypothesis. The alphabet of \mathbb{L} is finite, and there are at most finitely many combinations of letters of a finite alphabet of length less than $k+1$.

(To show that and there are at most finitely many words in \mathbb{L} of length less than $k+1$, just count them. Say that the alphabet of \mathbb{L} has m letters. There is one string of length 0, m strings of length 1, $m \cdot m$ strings of length 2, \dots , $\underbrace{m \cdot m \cdot \dots \cdot m}_{k+1 \text{ } m's}$ strings of length $k+1$. Add these up and you have a finite number.)

Since the language \mathbb{L} is infinite, and the number of strings in \mathbb{L} with fewer than $k+1$ letters is finite, there must be at least one string σ in \mathbb{L} with at least $k+1$ letters. (Of course, there will in fact be *infinitely* many strings in \mathbb{L} with at least $k+1$ letters, but we only need one.) Note that in the initial state, F hasn't read any letters, so F needs to pass through $k+2$ states to read a string of length $k+1$.

b) As just noted, since F has only k states, this means that to read a string of length $k+1$, at least one state has to be passed through at least twice.

c) So let's say that q_i is such a state passed through twice when reading σ . Say that the first time F is in q_i when reading σ is when it reads the letter u_r , and when F reads u_r in q_i it goes into q_{α_1} . (That is, $\langle q_i, u_r, q_{\alpha_1} \rangle$ is a transition of F.) F will then pass through some more states and return to q_i . (Of course, F could pass through *zero* states before returning to q_i . That is, it could loop directly from q_i to q_i .)

Subsequently say that the substring $u_r \cdots u_{r+n}$ is read by F, passing through states $q_{\alpha_1}, q_{\alpha_2} \dots q_{\alpha_n}$, where $q_{\alpha_n} = q_i$. That is, F loops around and reenters q_i , in the course of reading σ . Now we observe that if, at this point in the computation, F reads $u_r \cdots u_{r+n}$ *again* it will loop around precisely

the same states yet one more time. So:

Let x be the part of σ before u_r . That is, $x = u_1u_2\dots u_{r-1}$

Let y be the part of σ beginning with u_r and ending with u_{r+n} .

Let z be the part of σ beginning with u_{r+n+1} and ending with u_{k+1} .

We can see that F will accept every string consisting of x followed by some concatenation of instances of y followed by z , since it accepts xyz . It reads x and goes into state q_i , then it reads y and goes back into q_i . If, in q_i , it reads the string y again it will loop around and come back to q_i again. It will do this as many times as there are instances of y to read. If, after all this looping, the machine then returns to q_i and this time it reads z , it will reach a terminal state, since it reaches a terminal state when it reads xyz . This completes the proof.

This version of the lemma is relatively simple. If - during our counting of states and letters - we had been more exacting and creative in our book-keeping we could have extracted extra conditions on the length of xyz , and those extra conditions can be useful. However, even this minimal version of the Pumping Lemma allows us to prove some languages to be non-regular. Here is an example of the lemma in action.

3.2. Using the Lemma to Prove Languages Nonregular

Here is an example of a language that isn't regular. The alphabet is $\{a, b\}$, and $\mathcal{L} = \{\underbrace{aa\dots a}_{n \text{ a's}} \underbrace{bb\dots b}_{n \text{ b's}} / n \in \mathbb{N}\} = \{\epsilon, ab, aabb, aaabbb, aaaabbbb, \dots\}$. That is, \mathcal{L} consists of exactly the strings that begin with some number n of a 's followed by exactly n b 's.

Assume (for a contradiction) that \mathcal{L} is regular. Since \mathcal{L} is obviously infinite, the Pumping Lemma applies, and there is a string $xyz \in \mathcal{L}$, with $y \neq \epsilon$, such that for every n , $x\underbrace{y\dots y}_{n \text{ times}}z \in \mathcal{L}$. Say we have such a string $xyz \in \mathcal{L}$. Then $xyyz \in \mathcal{L}$ too. There are three possibilities: i) y consists of just a 's ii) y consists just of b 's iii) y consists of some a 's followed by some b 's.

If i), then since xyz has the same number of a 's and b 's, and y is nothing but a 's, then $xyyz$ has a different number of a 's and b 's. So $xyyz \notin \mathcal{L}$. Contradiction.

If ii), then since xyz has the same number of a 's and b 's, and y is nothing but b 's, then $xyyz$ has a different number of a 's and b 's. So $xyyz \notin \mathcal{L}$. Contradiction.

If iii) then since y has both a's and b's in it, the string yy has some b's (in the first instance of y) that come before some a's (in the second instance of y). Hence some b's come before some of the a's in $xyyz$, so by the definition of \mathcal{L} , So $xyyz \notin \mathcal{L}$. Contradiction.

Each of the three possibilities leads to a contradiction. We obtained this contradiction by assuming that \mathcal{L} is regular, so we conclude that \mathcal{L} isn't regular.