A question ~~but~~ that I did <u>not</u> put on the exam (b/c it's impossible)

Take a language L and let repeat(L) be the language ~~constitu~~ consisting of strings which are some string in L repeated (exactly twice)

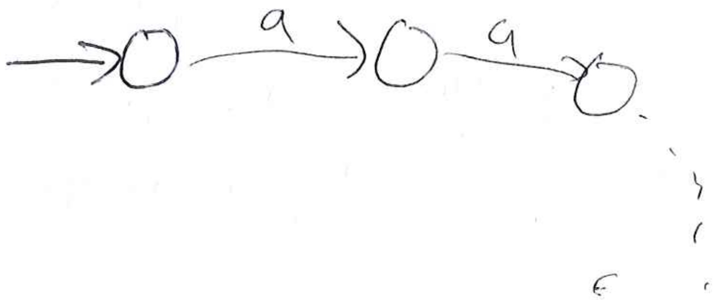
(If abaaab is in L,

abaaababaaab is in repeat(L).)

Tell me, how, given a DFA for L, I can modify it to create a DFA for repeat(L).

It turns out this is impossible.

Think about how this string is processed by the DFA.



By the time the DFA has "read" the first m a's, it has gone through m+1 states. Since there are only m states, it must have gone through some state twice.

Initial state + 1 state for every letter read.

So, for $w = \underbrace{a \cdots a}_{m} b \underbrace{a \cdots a}_{m} b$, the path to a final state looks like

Next 30(?) minutes - a proof this is impossible.

In fact, it's impossible for $L=\Sigma^*$.

So repeat(L) = $\{ww \mid w \in \{a,b\}^*\}$

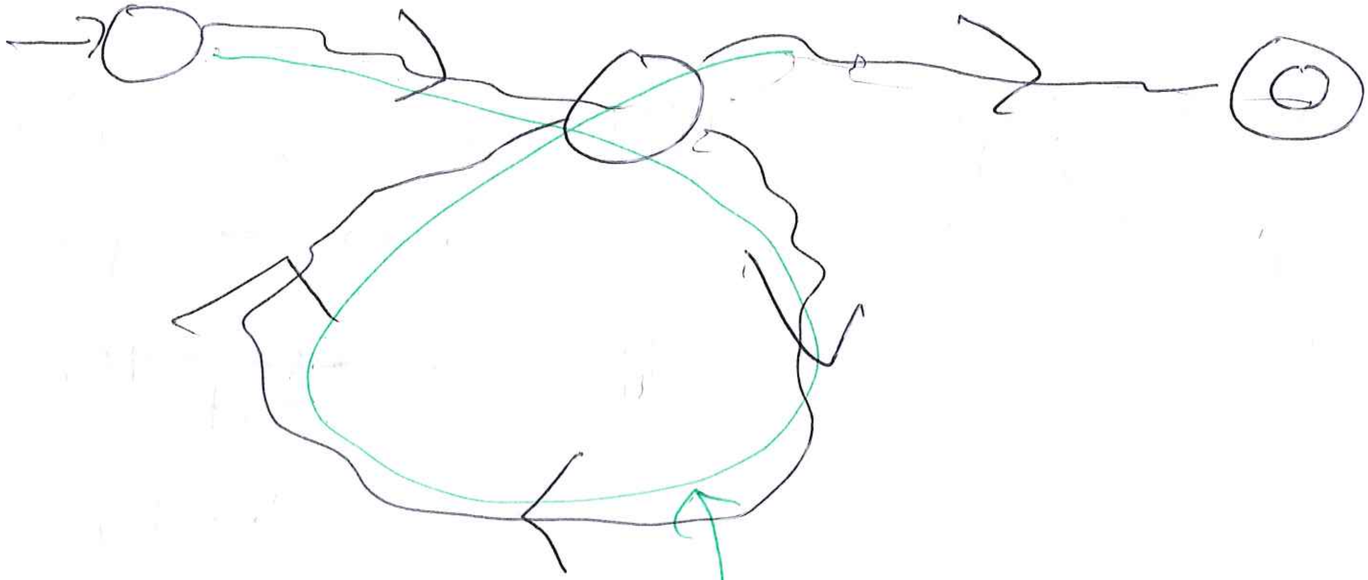
$=$ set of strings that ~~too~~ are some string repeated twice.

Assume for the purpose of a proof by contradiction there is a such a DFA M.

This DFA has some finite number of states m.

Consider the string $w = \underbrace{a \cdots a}_{m} b \underbrace{a \cdots a}_{m} b$

w is in repeat(L).



first m a's

(There could be more loops, but I'm forced to have a loop in the first m a's)

I can repeat (or delete) the loop and still end up at a final state.

So, if we ~~read~~ go through the loop twice, we will process

$\underbrace{a \ldots a}_{m+k \text{ times}}\, b\, \underbrace{a \ldots a}_{m \text{ times}}\, b$ and end at a final state.

($k \geq 1$ is the length of the loop)

This is NOT some ~~thin~~ string repeated twice. So ~~this~~ M accepts ~~son~~ a string that isn't of the form ww.

Conclusion: It is impossible to build a DFA that only accepts

$$\text{repeat}(\{a,b\}^*) = \{ww \mid w \in \{a,b\}^*\}.$$

Another example of a (provably) non-regular language:

$$L = \{\underbrace{a^n b^n}_{\substack{\| \\ \underbrace{a\cdots a}_{n \text{ times}}\underbrace{b\cdots b}_{n \text{ times}}}} \mid n \geq 0 \text{ some integers}\}$$

$$= \{\lambda, ab, aabb, aaabbb, aaaabbbb, \ldots\}$$

Why is this not possible?

Suppose there was a DFA M for L

M has some number m of states.

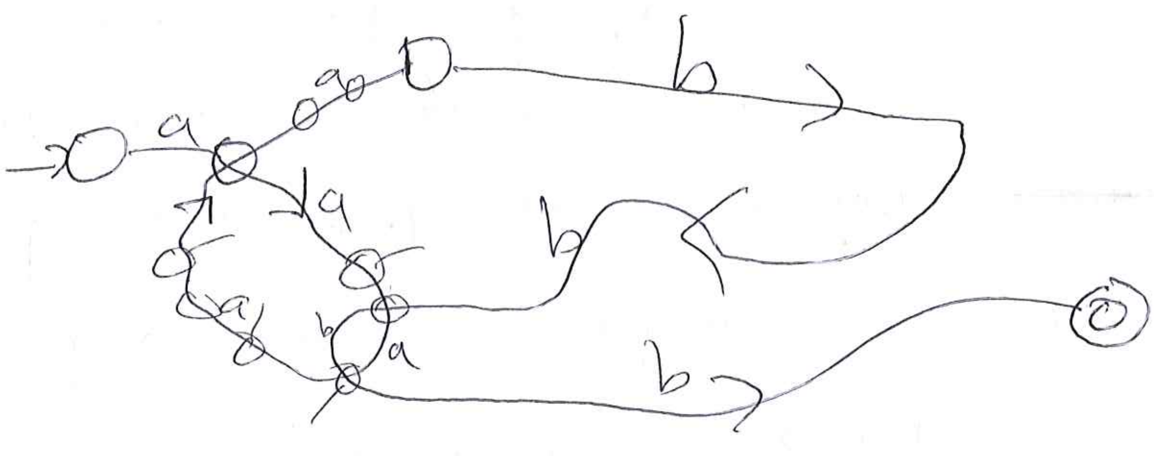
Whatever m is, we can construct the string

$$a^m b^m \in L,$$

so $a^m b^m$ is accepted.

Somewhere in the first m letters of the path we take for $a^m b^m$, there has to be a loop.

Our DFA looks like



If we repeat the loop then we have a path reading

$a^{m+k} b^m$ (for some k, which is the length of the loop) and ending up at a final state.

So this means M accepts

$$a^{m+k} b^m,$$

which is <u>not</u> in L

So it's not possible to build M that accepts only the strings in L.

These proofs are similar, so we should be able to abstract out the principle behind them to use in other similar problems.

<u>Pumping Lemma</u>: Let L be a regular language. Then there exists a~~n~~ ~~integ~~ positive integer m so that, for every string w with $|w| \geq m$, we can break w up as $w = xyz$, where $|xy| \leq m$, $y \neq \lambda$, so that $x\boxed{y^i}z \in L$ for all i.

repeat y i times.

We usually use the pumping lemma to prove a language L is <u>not</u> regular by showing that it does <u>not</u> have the required property.

Revisit the first 2 proofs.

For $L = \text{repeat}(\{a,b\}^*)$:

Assume L is regular. ~~Then there exists some m, and~~ Given m, we can choose $w = a^m b a^m b \in L$. By the pumping lemma $w = xyz$, $|xy| \le m$, $y \ne \lambda$, with $xy^i z \in L$. But, since $|xy| \le m$, $xy = a^l$ for some l, so $y = a^k$, $k \ge 1$. This means $xy^2 z = a^{m+k} b a^m b \notin L$, a contradiction. So L can't be regular.