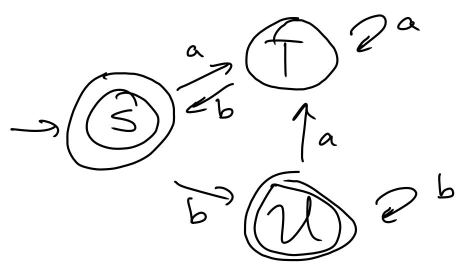
What about the union of regular languages?

We again use the product construction, but choose accepting states differently.

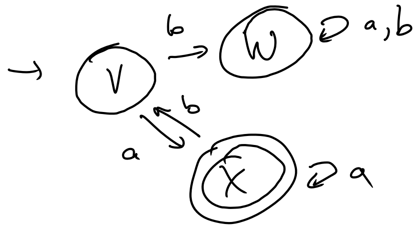
we need to accept anything accepted by the 1st machine OR by 2nd machine.

Ex)

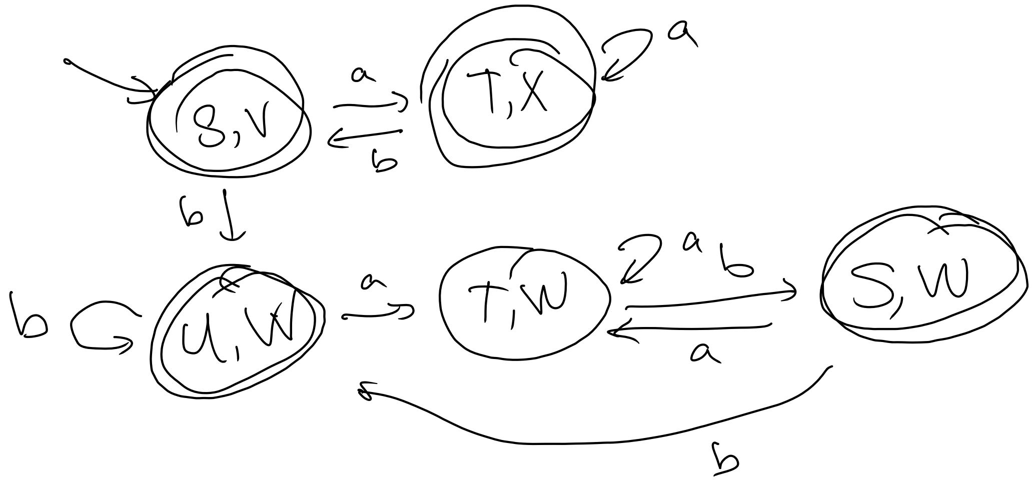
M_1:



M_2:



Make a DFA which accepts $L(M_1) \cup L(M_2)$.



Formally: if $M_1 = (Q, \Sigma, \delta_1, q_0, F_1)$ $\quad F_1 \cup F_2$

& $M_2 = (R, \Sigma, \delta_2, r_0, F_2)$

Then we use: $\quad \cancel{F_1 \times F_2}$

$$(Q \times R, \Sigma, \delta, (q_0, r_0), \underline{(F_1 \times R) \cup (Q \times F_2)})$$

want: first thing in F_1, or 2^{nd} thin in F_2

$$\underbrace{(F_1 \times R)}_{\text{1st thing in } F_1 \text{, 2nd is anything}} \cup \underbrace{(Q \times F_2)}_{\text{1st thing is anything, 2nd in } F_2}$$

1st thing in F_1
2nd is anything

OR

1st thing is anything,
2nd in F_2

<u>Thm</u> If L_1 & L_2 are regular, then $L_1 \cup L_2$ is regular.

what about $L_1 - L_2$? on HW

DFA's can't compute every possible language.

To show some L is non-regular, we need to prove no possible DFA can accept L.

(hard!)

Ex

$$L = \{ a^n b^n \mid n \in \mathbb{N} \}$$

all a's followed by the same # of b's.

Thm L is not a regular language

Pf by contradiction: Assume we have a DFA

$$M = (Q, \Sigma, \delta, q_0, F)$$

with $L(M) = L$

Consider $\delta^*(q_0, a^i)$ for various i,
these can't all be different because Q is finite,

so we must have

$$\delta^*(q_0, a^i) = \delta^*(q_0, a^j)$$

for some $i \neq j$.

Then we'll have

$$\delta^*(q_0, a^i b^i) = \delta^*(q_0, a^j b^i)$$

this is in L not in L

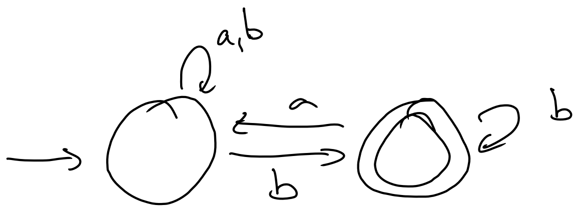
So this machine: if it accepts $a^i b^i$,
it must also accept $a^i b^j$

So $L(M)$ cannot be $\{a^n b^n \mid n \in \mathbb{N}\}$

Sharon

Nondeterministic Finite Automaton (NFA)

The NFA can act nondeterministically,
the same string can result in different
final states.



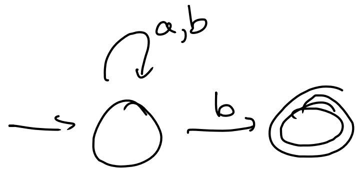
ab is accepted
a^n is rejected.

2 outgoing b's

So if we do ab, we can end up
in either state.

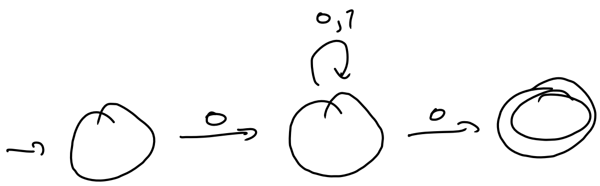
A string is accepted if any of possible final states are accepting.

An NFA can have any number (even 0) of outgoing arrows for each letter.



accepts anything ending with b.

$\{0x0 \mid x \in \{0,1\}^*\}$



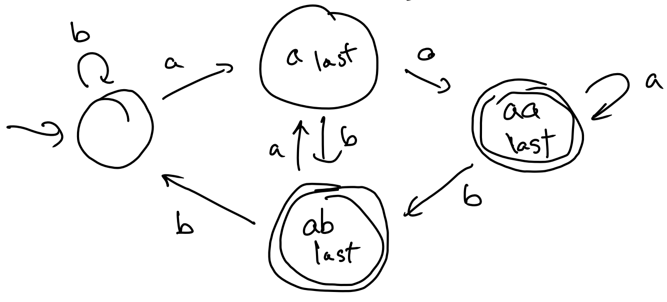
0101101110

An NFA is often simpler than a DFA for the same language:

on $\{a,b\}$ $L = \{x \in \{a,b\}^* \mid 2^{nd} \text{ to last letter is } a\}$

$x \in \Sigma^*$

DFA



$\Sigma = \{a,b\}$

<u>NFA</u>