HW 3 due today 9pm
GPS 4 due Mon 21st 9pm
HW 4 due Tue 22nd 9pm

Midterm 1 Mon Sep 28th 7pm - 9pm
Conflict Midterm 1 Tue Sep 29th
3 practice exams available later

Language transformations that preserve regularity.

Given a function f that takes languages to languages.

Want to prove that if L is regular, then $f(L)$ is regular.

Two methods:

1) Give an algorithm to turn a regular expression R for some L into a regular expression R' s.t. $L(R') = f(L(R))$.

2) Give an algorithm/method to turn a DFA M into an NFA M' s.t.

$L(M') = f(L(M))$

Bitwise complement:

$$w^c := \begin{cases} \epsilon & \text{if } w = \epsilon \\ (1-a)\cdot x^c & \text{if } w = a\cdot x \end{cases}$$

i.e. w^c is w with all bits flipped

$(w^c)^c = w$

$L^c = \{w^c \mid w \in L\} = \{w \mid w^c \in L\}$

Thm: For any regular language L, L^c is also regular.

Proof: Let $M=(Q, \delta, s, A)$ be an arbitrary DFA that accepts L.

We construct an NFA $M^c=(Q^c, \delta^c, s^c, A^c)$.

$Q^c = Q$

$\delta^c(q,a) = \{\delta(q, 1-a)\}$ ← its an NFA

(while reading w^c, simulate M reading w)

$s^c = s$

$A^c = A$

String reversal: $w^R := \begin{cases} \epsilon & \text{if } w = \epsilon \\ x^R \bullet a & \text{if } w = a \cdot x \end{cases}$

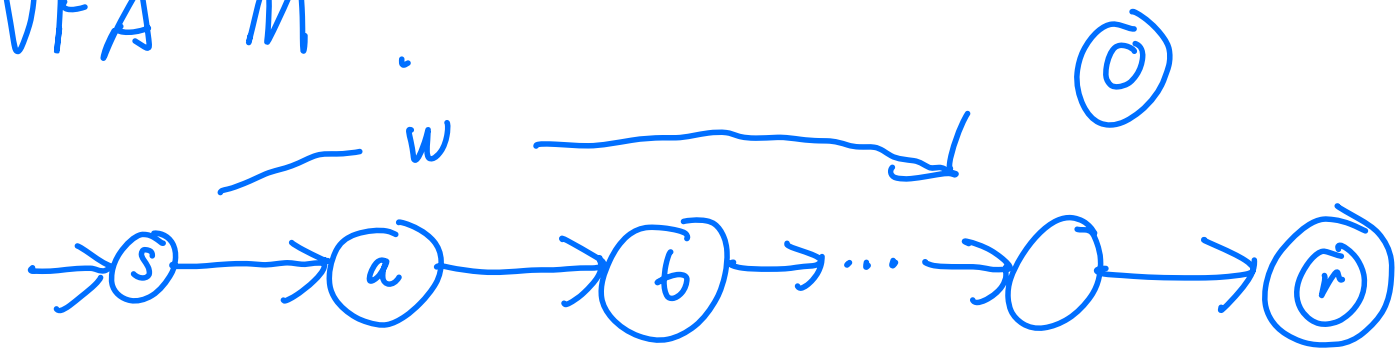
$(w^R)^R = w$

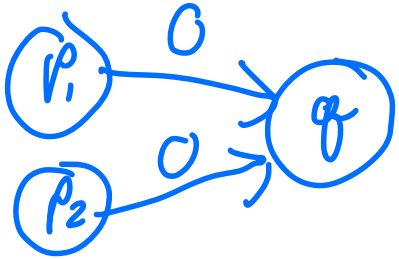
$L^R := \{ w^R \mid w \in L \} = \{ w \mid w^R \in L \}$

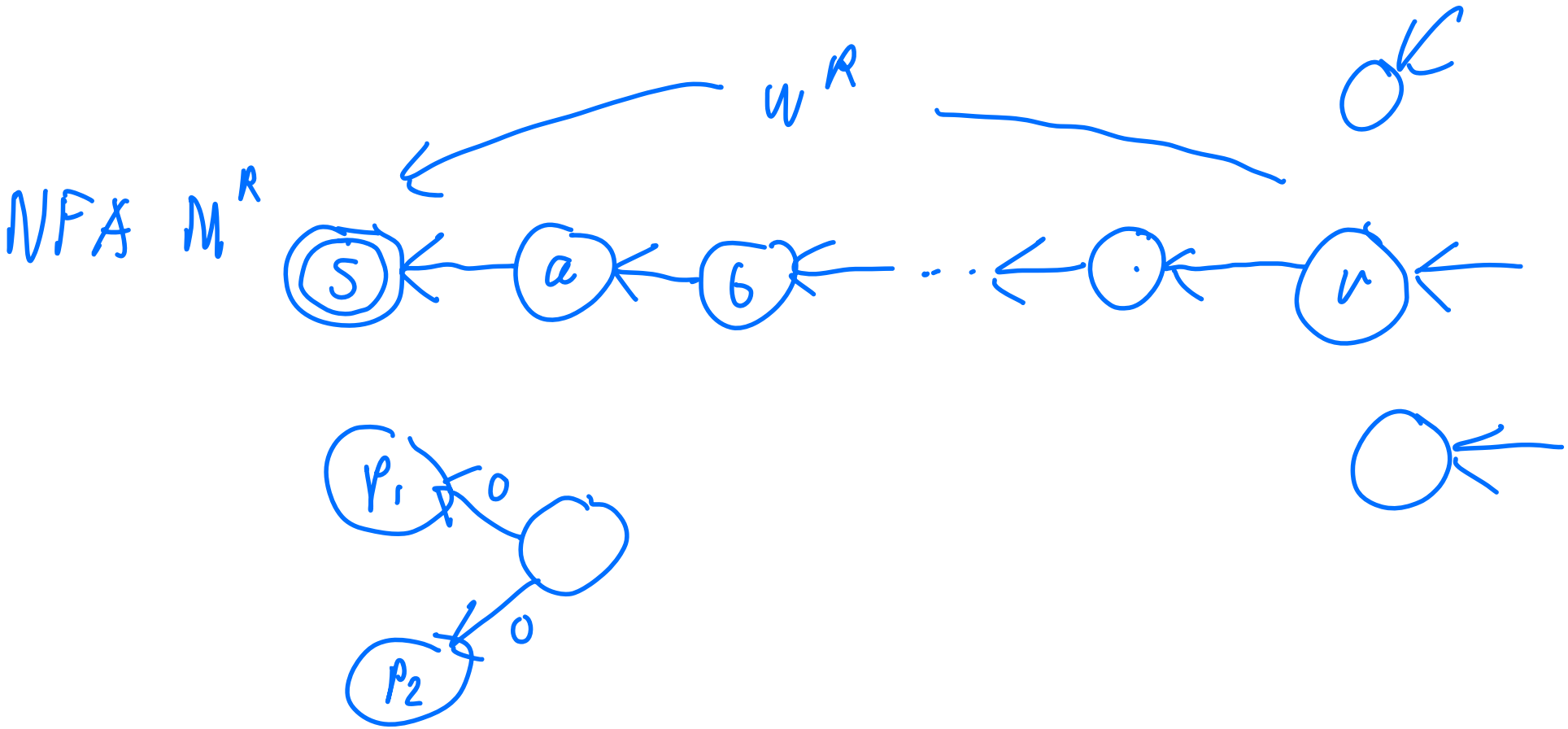
Let $M = (Q, \delta, s, A)$ be a DFA.

Build NFA M^R.

DFA M:







simulate M going backwards!

Proof: Let $M = (Q, \delta, s, A)$ be an arbitrary DFA for L.

We construct an NFA with multiple start states $M^R = (Q^R, \delta^R, S^R, A^R)$ that accepts L^R.

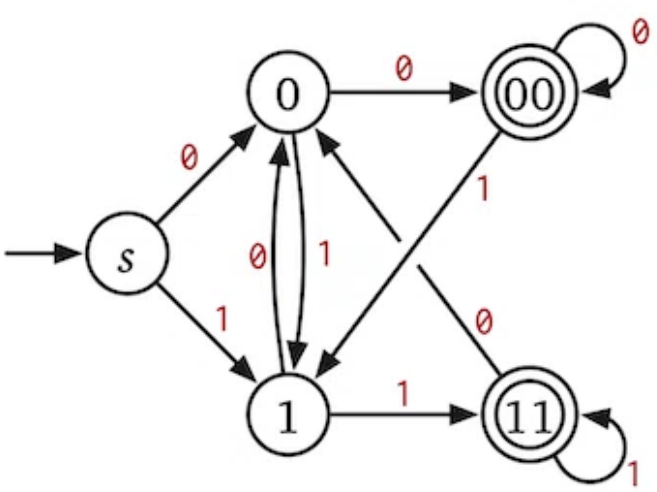
$Q^R = Q$

(M^R has w^R take something in S^R to $q \in Q^R$ iff M has w take q to an accept state)

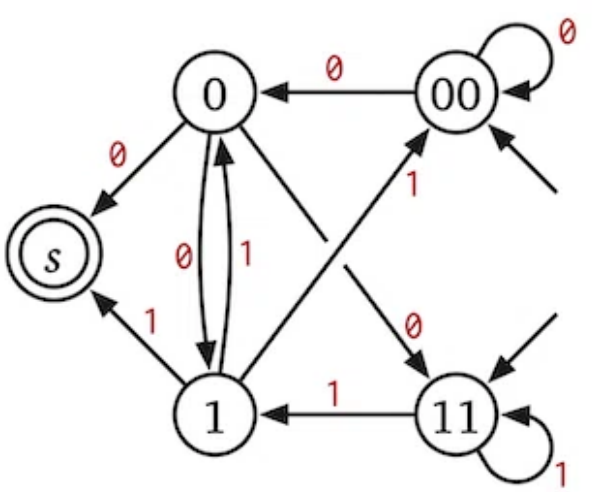
$S^R = A$

$A^R = \{s\}$

$$\delta^R(q,a) = \{ p \mid \delta(p,a) = q \} \qquad \forall q \in Q, a \in \Sigma$$



DFA M



NFA M^R

First half of palindromes:

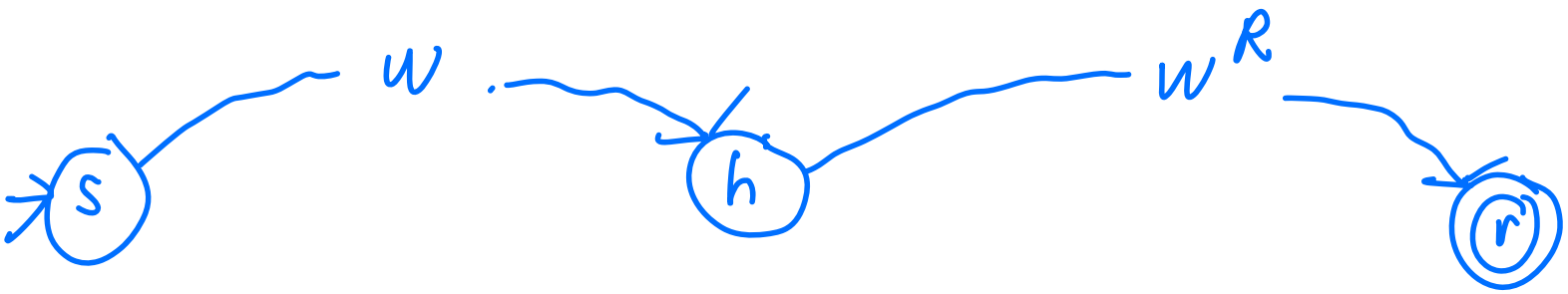
For any language L, let

$$palin(L) := \{ w \mid w \cdot w^R \in L \}$$

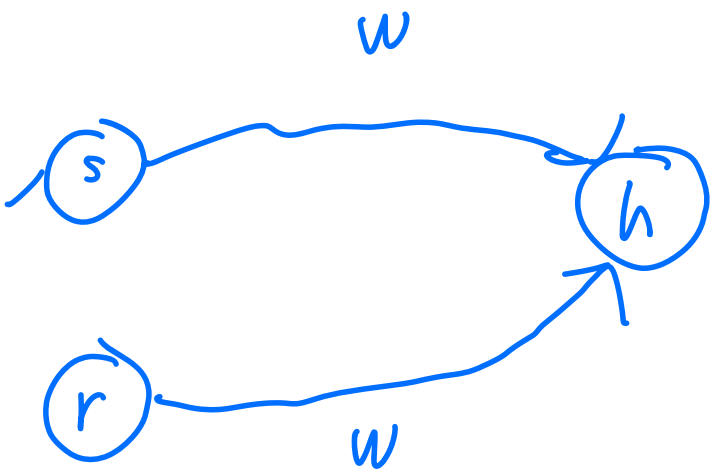
(first half of even length palindromes in L)

(not even length palindromes in L)

Thm: For any regular language L,
$palin(L)$ is regular.



DFA M



NFA M'

Need to simulate Forward + Backward!

Proof: Let $M = (Q, \delta, s, A)$ be an arbitrary DFA for language L.

We construct an NFA with multiple start states $M' = (Q', \delta', S', A')$.

$Q' = Q \times Q$

(M' has w take something in S to (p, q) iff M has w take s to p and it has w^R take q to an accept state $r \in A$)

$S' = \{(s, r) \mid r \in A\}$

$A' = \{(h, h) \mid h \in Q\}$

$$\delta'((p,r),a) = \{(\delta(p,a),q) \mid \delta(q,a) = r\}$$

first sim. goes forward

second sim. goes backward

For a language L, let

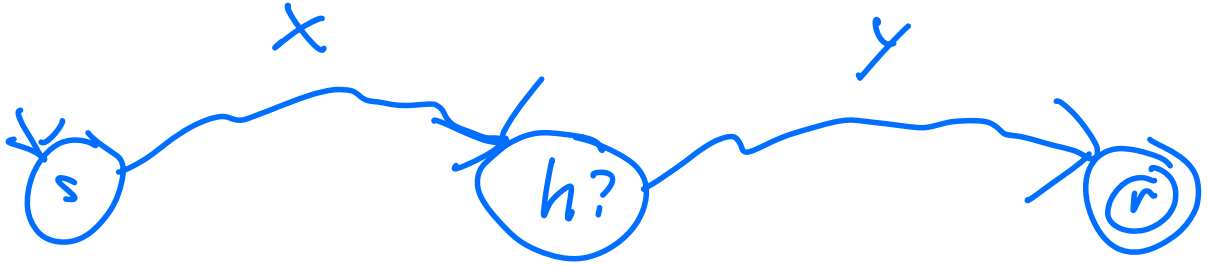
$$\text{comp-suffix}(L) = \{xy^c \mid xy \in L\}$$

Ex: $L = \{1001\}$, comp suffix $(L) =$

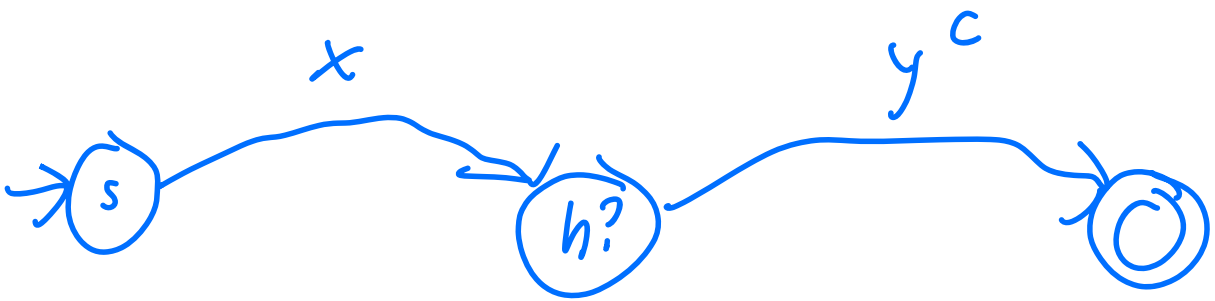
$\{1001, 1000, 1010, 1110,$

$0110\}$

Thm: For any regular L, comp-suffix (L) is regular.



DFA M

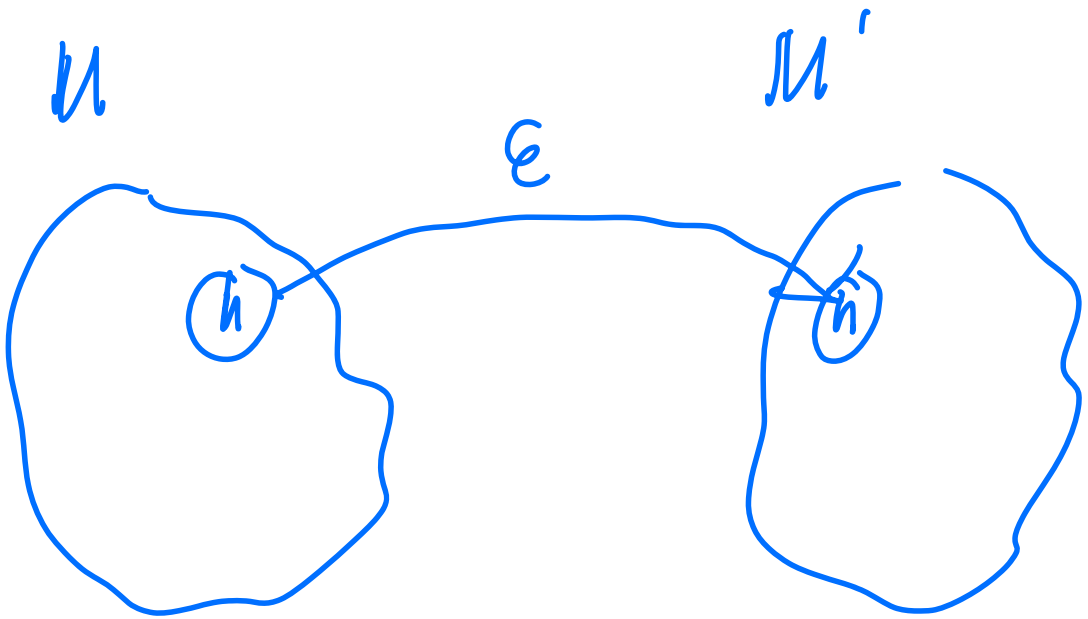


NFA M'

So two copies of M.

How to handle switching?

Use an ϵ-transition!



Proof: Let $M = (Q, \delta, s, A)$ be a DFA.

Construct NFA with ϵ-transitions

$M' = (Q', \delta', s', A')$.

$Q' = Q \times \{same, comp\}$

$(q, same)$: M can get to q with x

$(q, comp)$: M can get to some h with x
& then from h to q with y.

(M' reads xy^c)

$s' = (s, same)$

$A' = \{(r, comp) \mid r \in A\}$

$$\delta'((q, same), a) = \{(\delta(q, a), same)\}$$

$$\delta'((q, same), \epsilon) = \{(q, comp)\}$$

$$\delta'((q, comp), a) = \{(\delta(q, 1-a), comp\}$$

$$\delta'((q, comp), \epsilon) = \emptyset$$

L^R:

Let S be a reg. expression. Make S^R.

If $S = \emptyset$, use $\emptyset$.

If $S = w$, use w^R.

If $S = A + B$, use $A^R + B^R$

If $S = AB$, use $B^R A^R$

If $S = A^*$, use $(A^R)^*$

Second proof that L regular $\Rightarrow L^R$ regular.

This technique is generally <u>less</u> useful.