Intro to Theory of Computation

CS 464

LECTURE 5

Last time:
- Closure properties.
- Equivalence of NFAs, DFAs and regular expressions

Today:
- Conversion from NFAs to regular expressions
- Proving that a language is not regular: pumping lemma

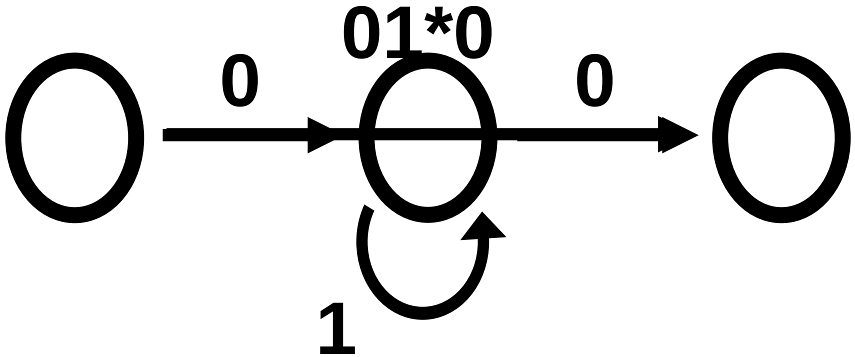
Sofya Raskhodnikova

NFAs to regular expressions

Theorem. **Every NFA has an equivalent regular expression.**

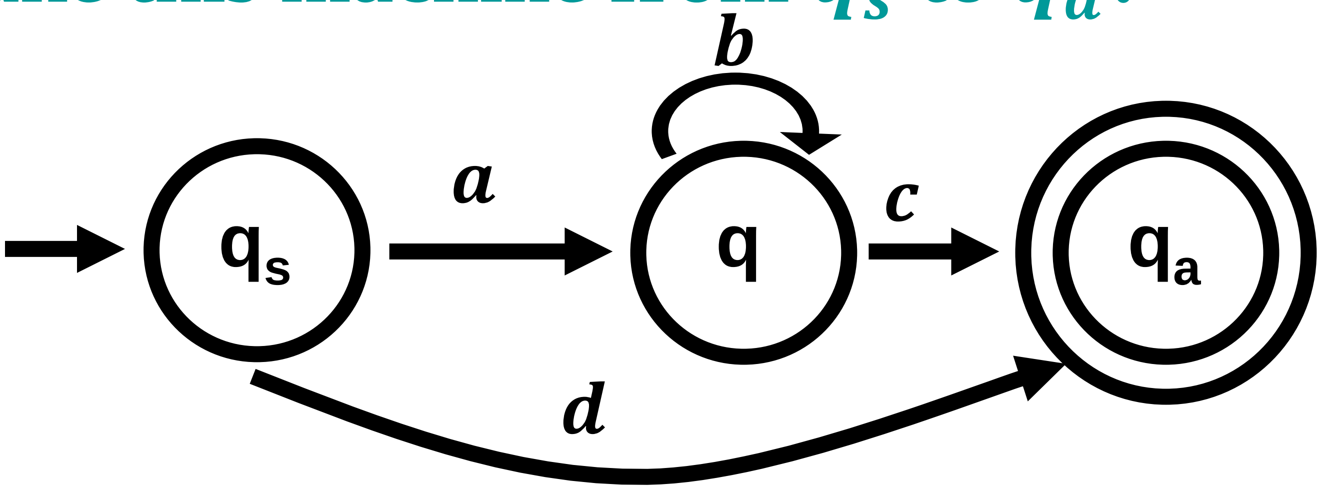
Proof idea:

Transform NFA to a regular expression by removing states and relabeling the arrows with regular expressions.



I-clicker problem (frequency: AC)

What is the regular expression that generates all strings that take this machine from q_s to q_a?



A. $ab^*c \cup d$

B. $ab^*c \cap d$

C. $abc \cup d$

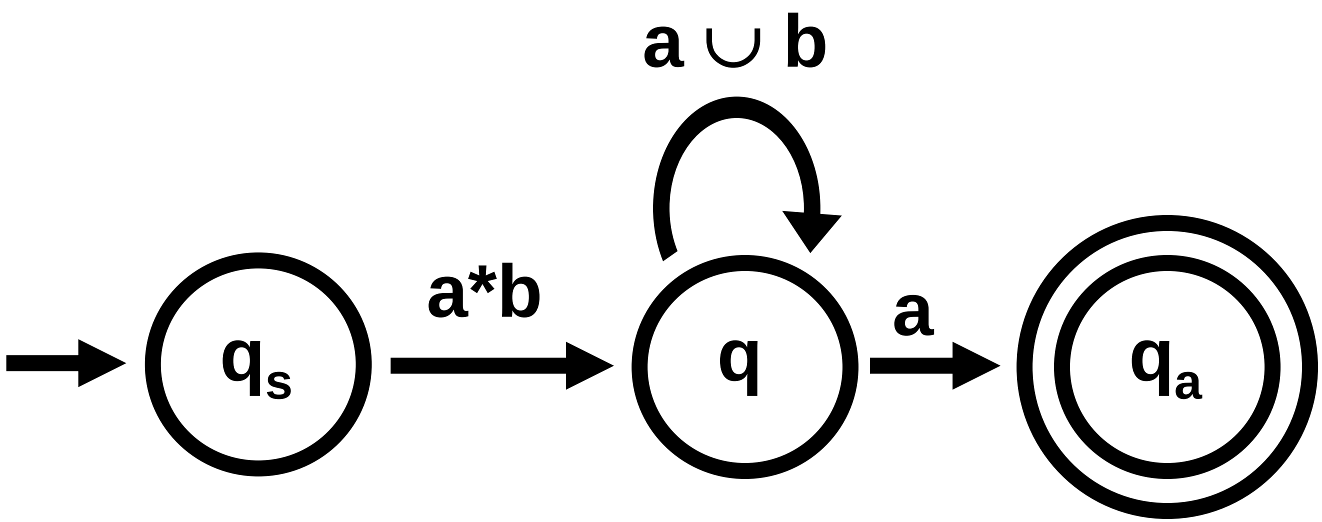
D. $(ab^*cd)^*$

E. None of the above.

Generalized NFAs

- **Each transition is labeled with a regular expression**
- Unique and distinct start and accept states
- No transitions **to** the start state
- No transitions **from** the accept state

Generalized NFAs



G accepts w if it finds $q_0 q_1 \ldots q_k$, $w_1 \ldots w_k$:

w_i is generated by $R(q_i, q_{i+1})$

$w = w_1 w_2 \ldots w_k$

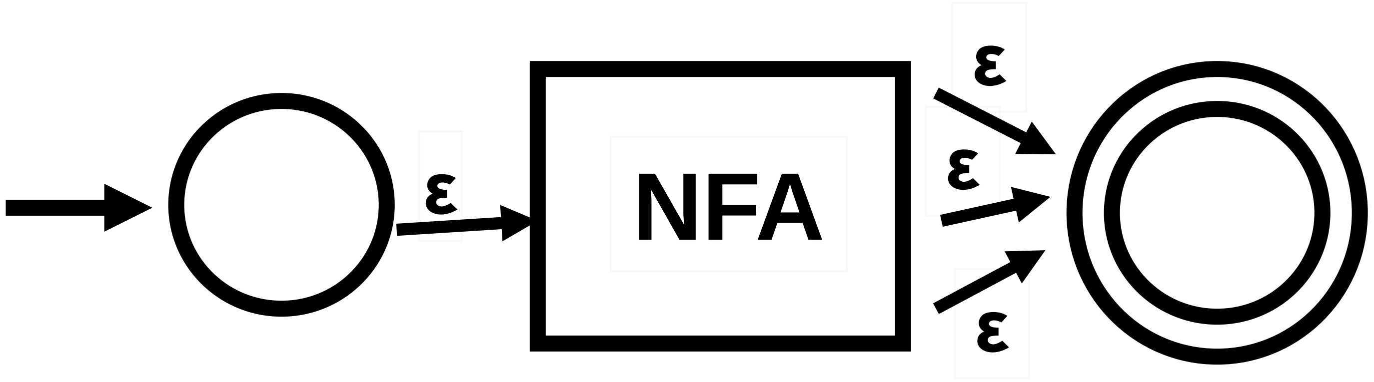
$q_0 = q_s$, $q_k = q_a$

$R(q_s, q) = a^*b$

$R(q_a, q) = \emptyset$

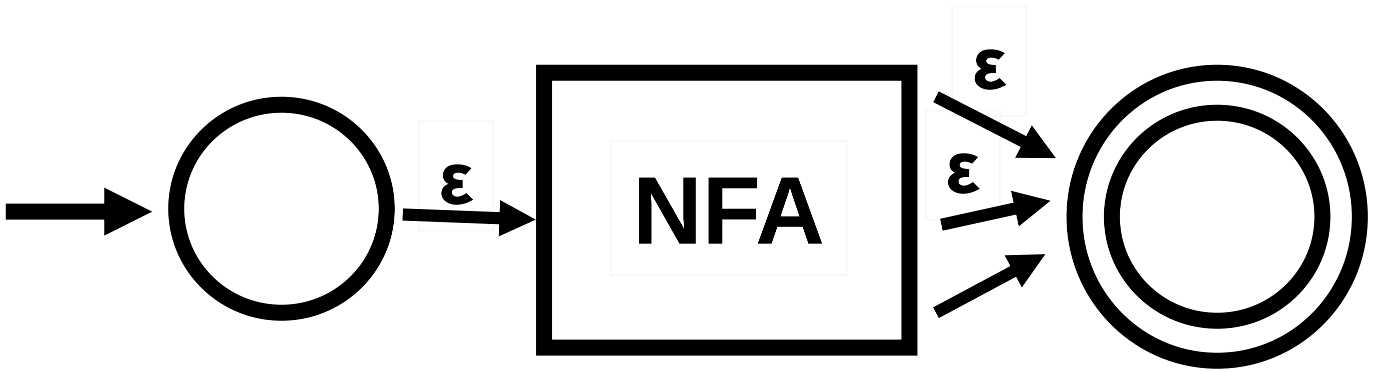
$R(q, q_s) = \emptyset$

NFA to GNFA



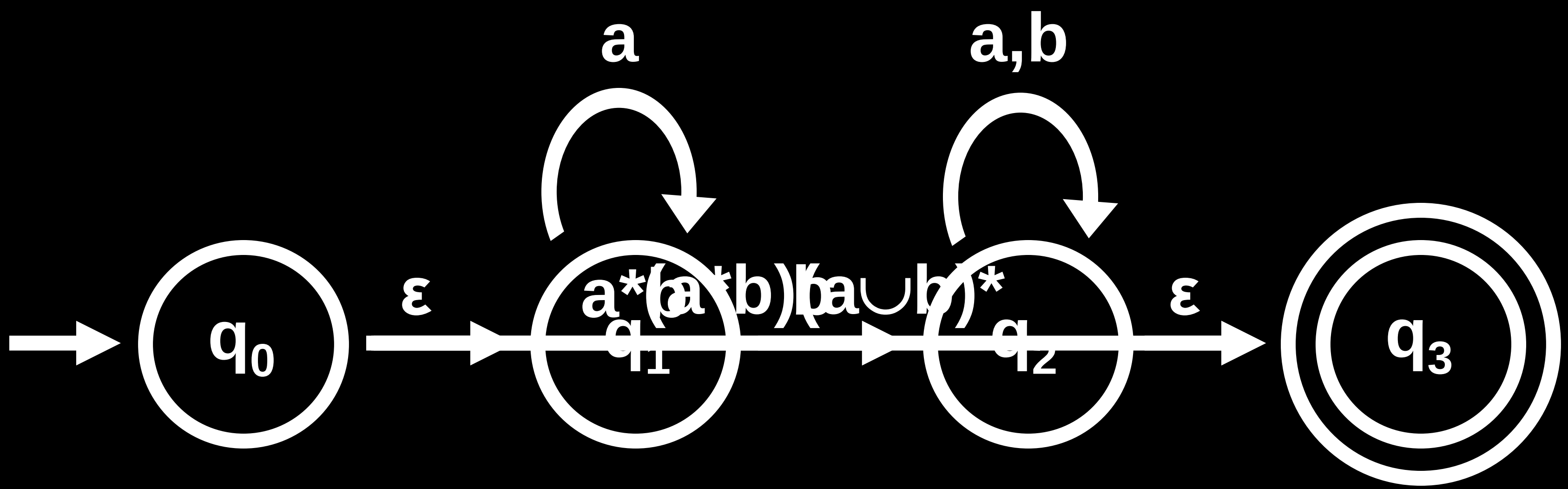
Add a new start state with no incoming arrows.
Make a unique accept state with no outgoing arrows.

GNF to regular expression

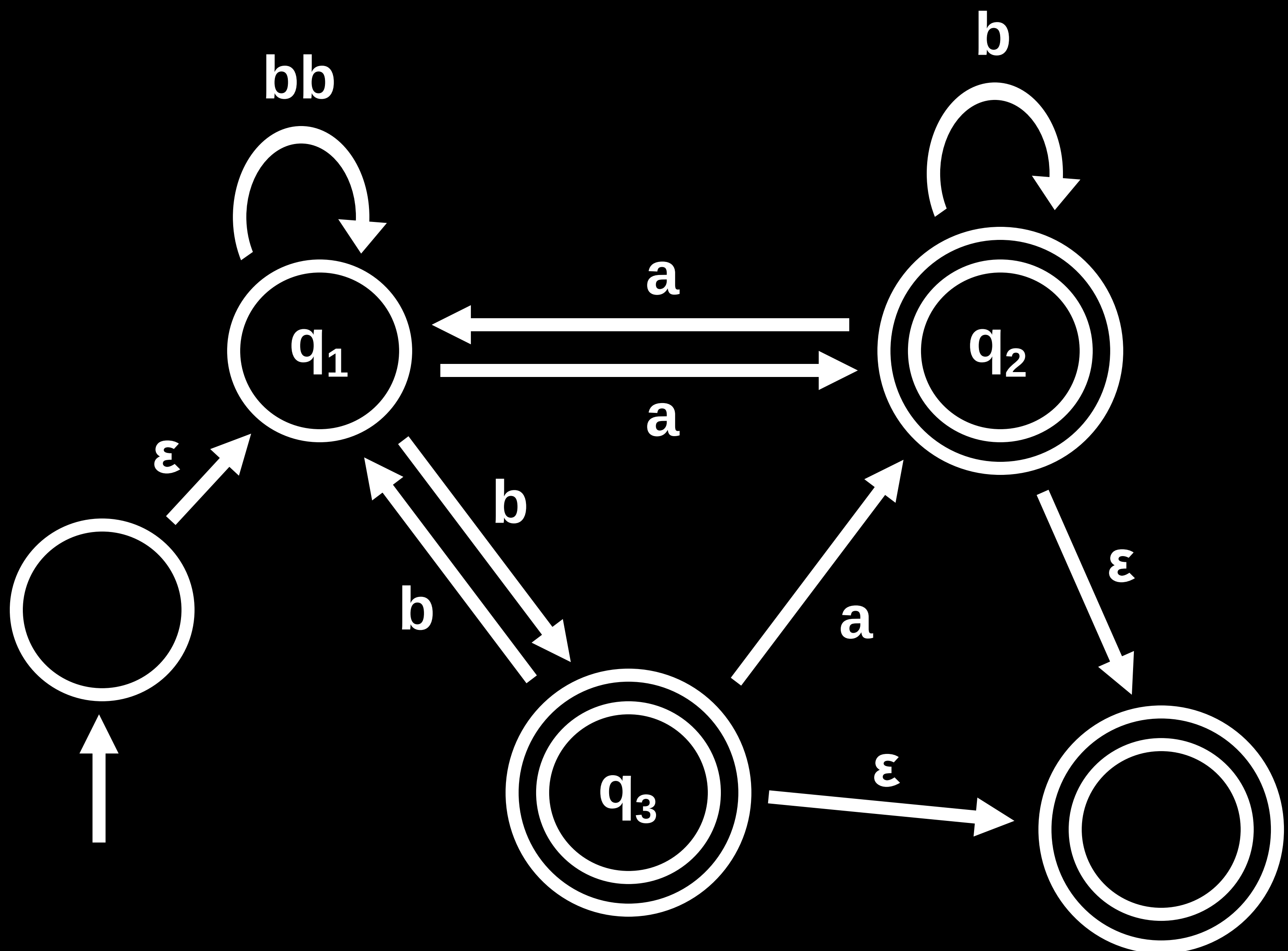


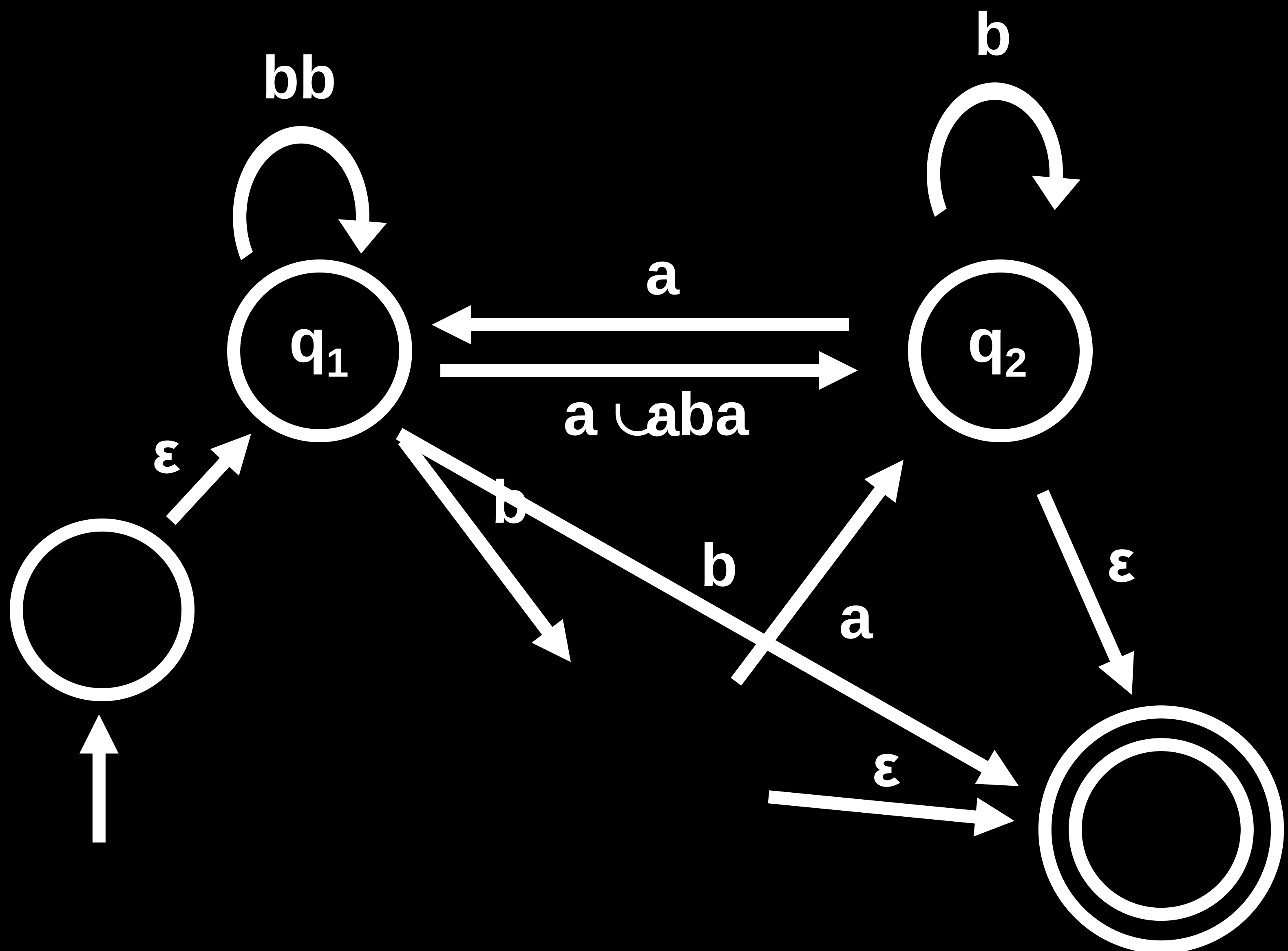
While machine has more than 2 states:

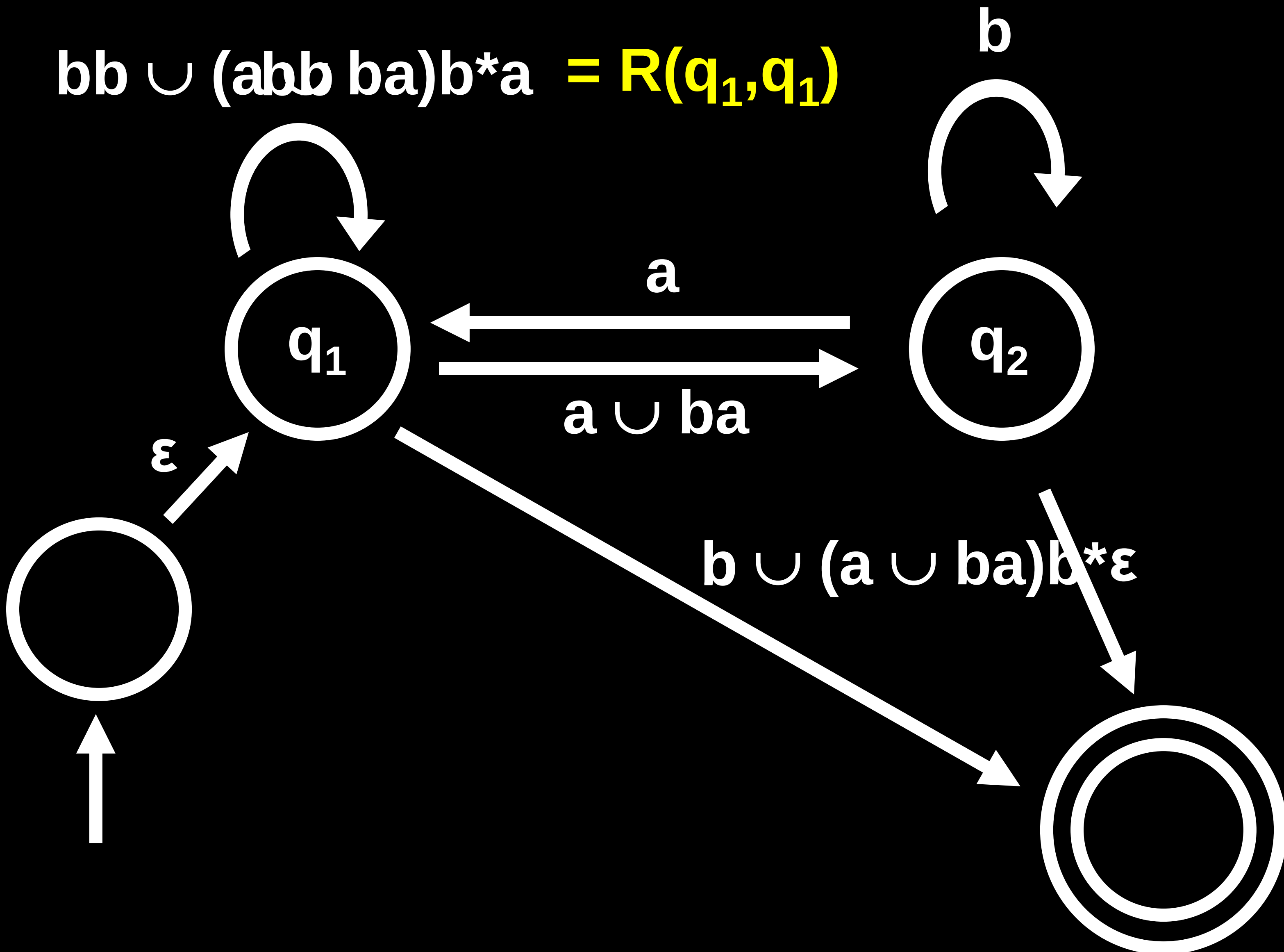
> **Pick an internal state, rip it out and relabel the arrows with regular expressions to account for the missing state.**

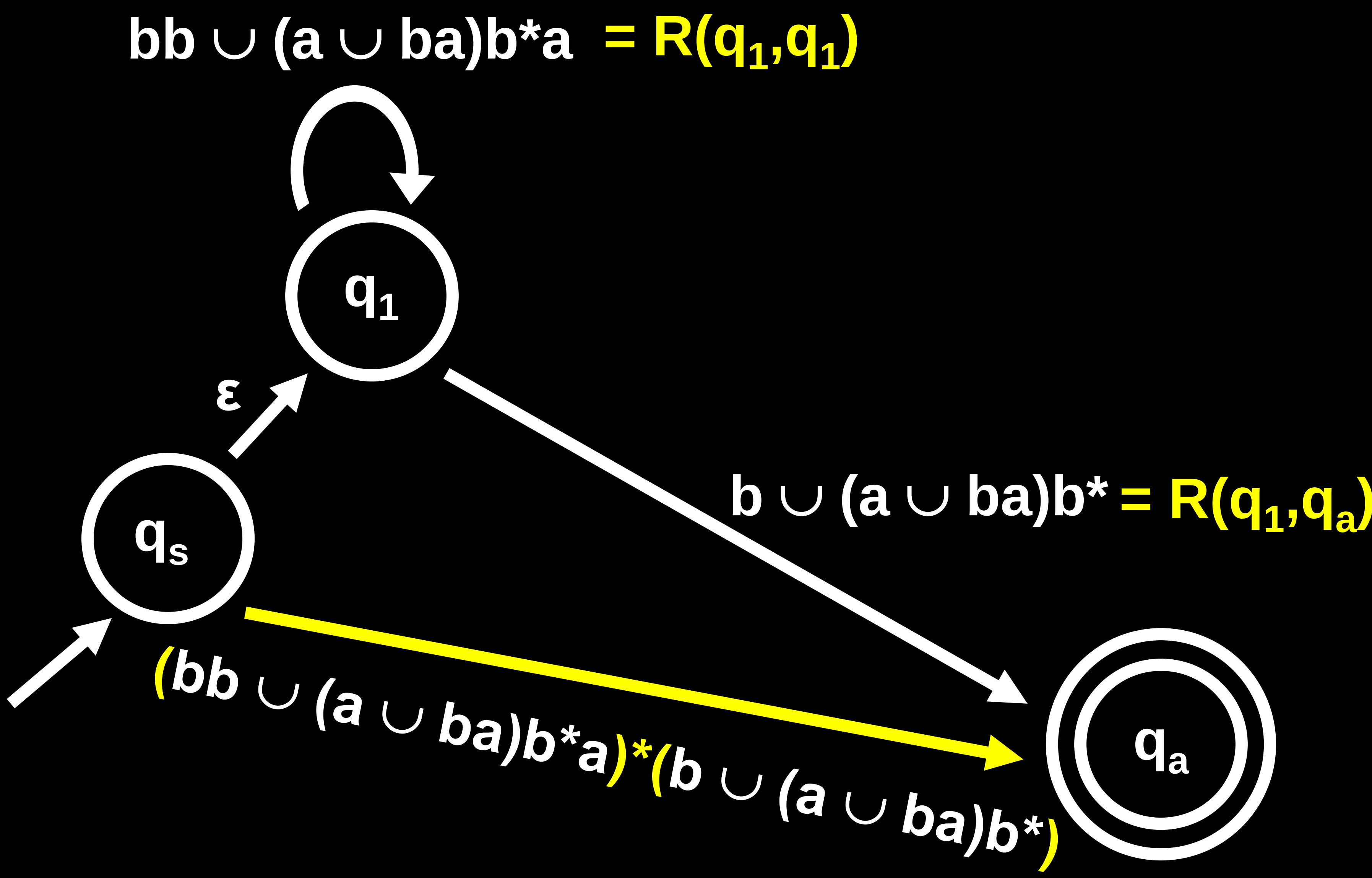


$R(q_0,q_3) = (a^*b)(a \cup b)^*$


bb
b
a
q1
q2
a
ε
b
ε
b
a
q3
ε

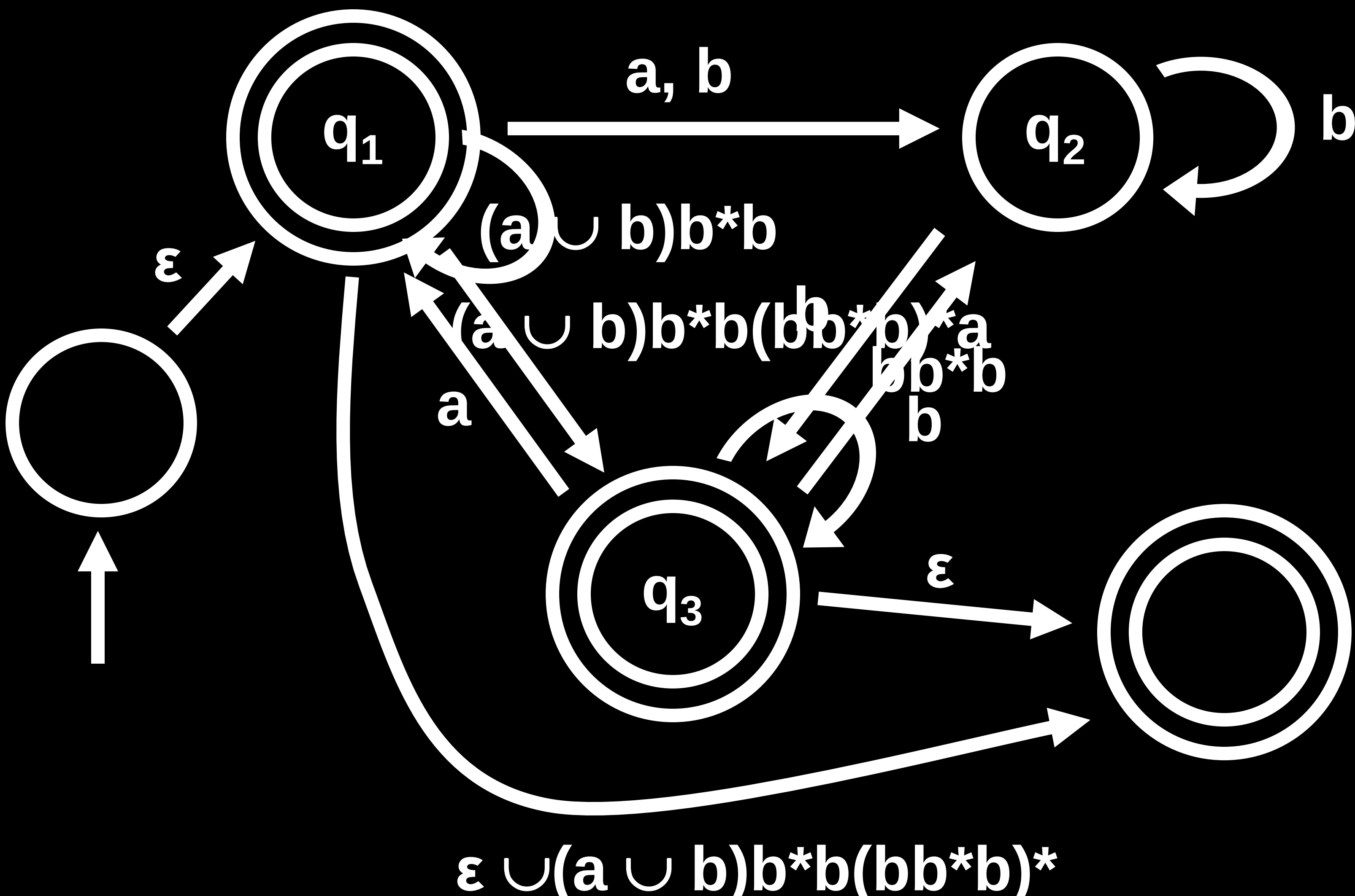

b
bb
a
q1
q2
a ∪ aba
ε
b
b
ε
a
ε


bb ∪ (abb ba)b*a = R(q1,q1)
b
q1
a
q2
a ∪ ba
ε
b ∪ (a ∪ ba)b*ε

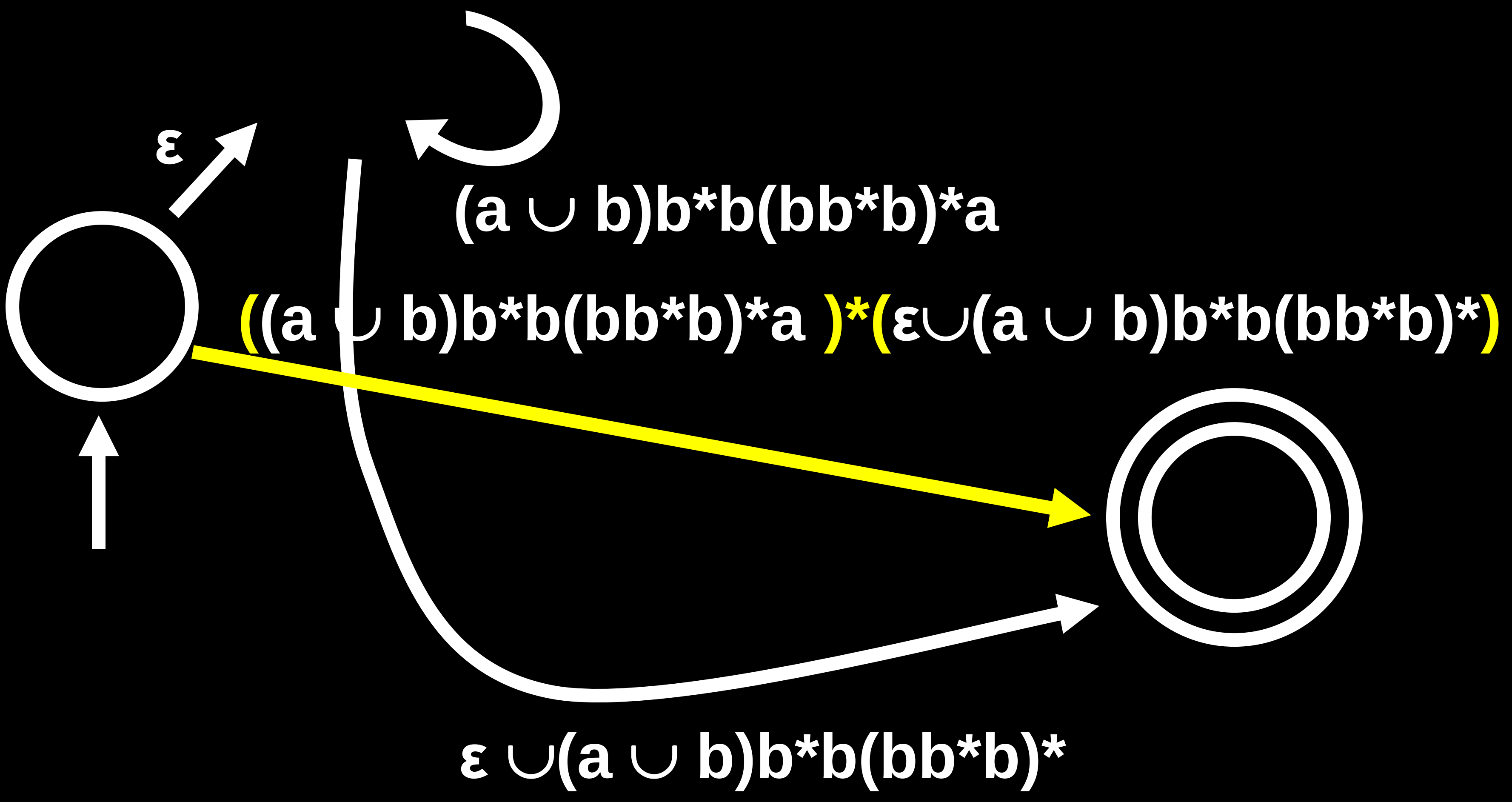


bb ∪ (a ∪ ba)b*a = R(q1,q1)
q1
ε
qs
b ∪ (a ∪ ba)b* = R(q1,qa)
(bb ∪ (a ∪ ba)b*a)*(b ∪ (a ∪ ba)b*)
qa

Convert the NFA to a regular expression



Convert the NFA to a regular expression



NFA to regular expression

Add q_{start} and q_{accept} to create GNFA G.
Run CONVERT(G)

CONVERT(G):
If #states ≥ 2
return the expression on the arrow going from q_{start} to q_{accept}

NFA to regular expression

Add q_{start} and q_{accept} to create GNFA G.
Run CONVERT(G)

CONVERT(G):
If #states > 2

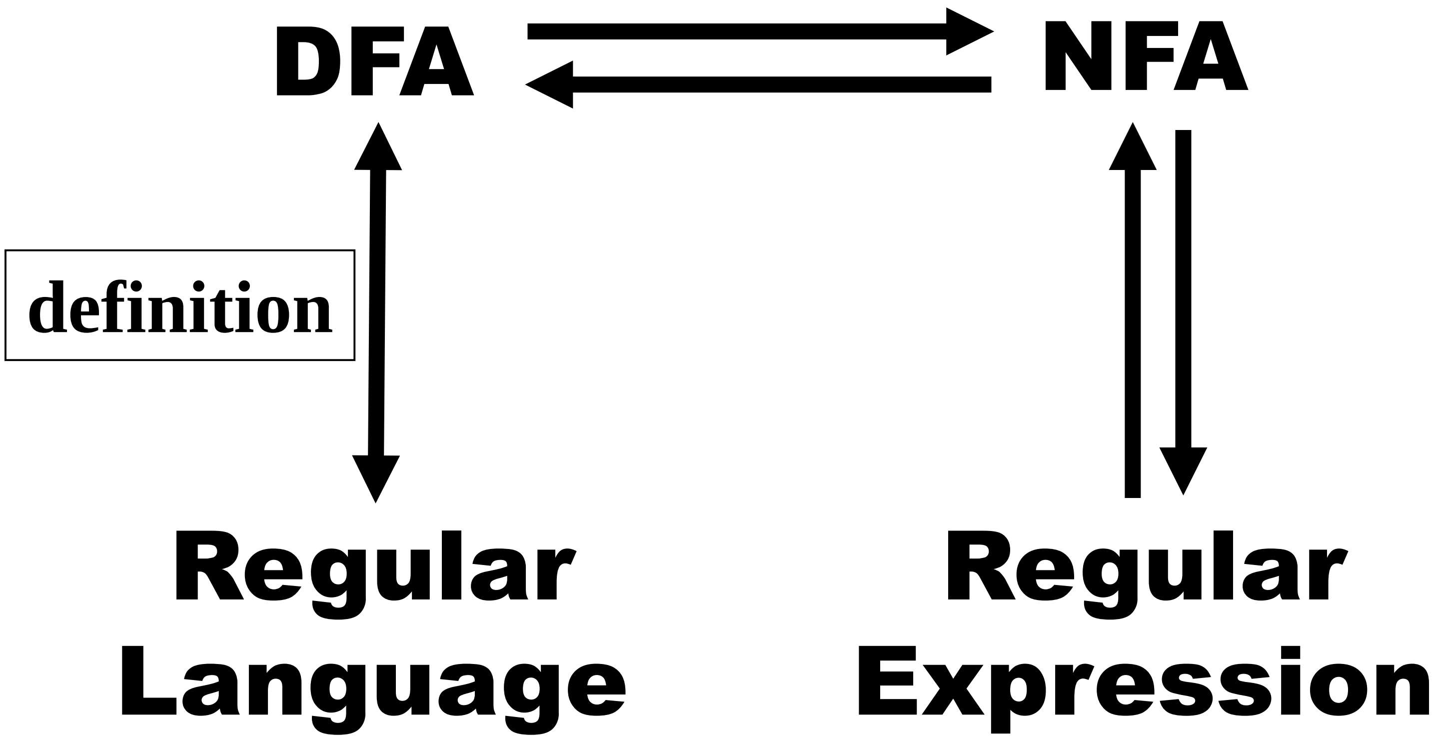
Build G′ from G:

select $q_{rip} \in Q$ different from q_{start} and q_{accept}
define $Q' = Q - \{q_{rip}\}$
define R′ as:

$$R'(q_i,q_j) = R(q_i,q_{rip})R(q_{rip},q_{rip})^*R(q_{rip},q_j) \cup R(q_i,q_j)$$

return CONVERT(G′)

Conversion procedures



REGULAR OR NOT?

Design an NFA for the language:

$$\{0^n1^n \mid 0 < n \leq 2\}$$

$$\{0^n1^n \mid 0 < n \leq k\}$$

$$\{0^n1^n \mid n > 0\}?$$

(For R a regexp, R^2 means RR, and R^n means $\overbrace{RR\ldots R}^{n}$)

SOME LANGUAGES **ARE NOT** REGULAR!

B = $\{0^n 1^n \mid n \geq 0\}$ is NOT regular!

Proof (by contradiction)

Let M be a k-state DFA that recognizes B.

Consider the path M takes on $0^k 1^k$:

$$q_0 q_1 q_2 \ldots \quad q_i q_{i+1} \quad q_j \quad q_k \quad \ldots \quad q_{2k} \in F$$

0000...00..0..011111...11

There must be $i < j \leq k$ such that $q_i = q_j$

M accepts $0^{k-(j-i)} 1^k \notin B$!

So M does not recognize the language B.

REGULAR OR NOT?

C = { w | w has equal number of 1s and 0s}

NOT REGULAR

D = { w | w has equal number of occurrences of 01 and 10}

$$(0\Sigma^*0) \cup (1\Sigma^*1) \cup 1 \cup 0 \cup \varepsilon$$

THE PUMPING LEMMA

Let L be a regular language with $|L| = \infty$

Then there exists a length p such that

if $w \in L$ and $|w| \geq p$ then

w can be split into three parts w=xyz where:

1. $|y| > 0$
2. $|xy| \leq p$
3. $xy^iz \in L$ for all $i \geq 0$

THE PUMPING LEMMA

if $w \in L$ and $|w| \geq p$
then $w = xyz$, where:

1. $|y| > 0$
2. $|xy| \leq p$
3. $xy^iz \in L$ for all $i \geq 0$

Example:

Let $L = 0^*1^*$; $p = 1$
$w = 011$
$x = \varepsilon$
$y = 0$
$z = 11$

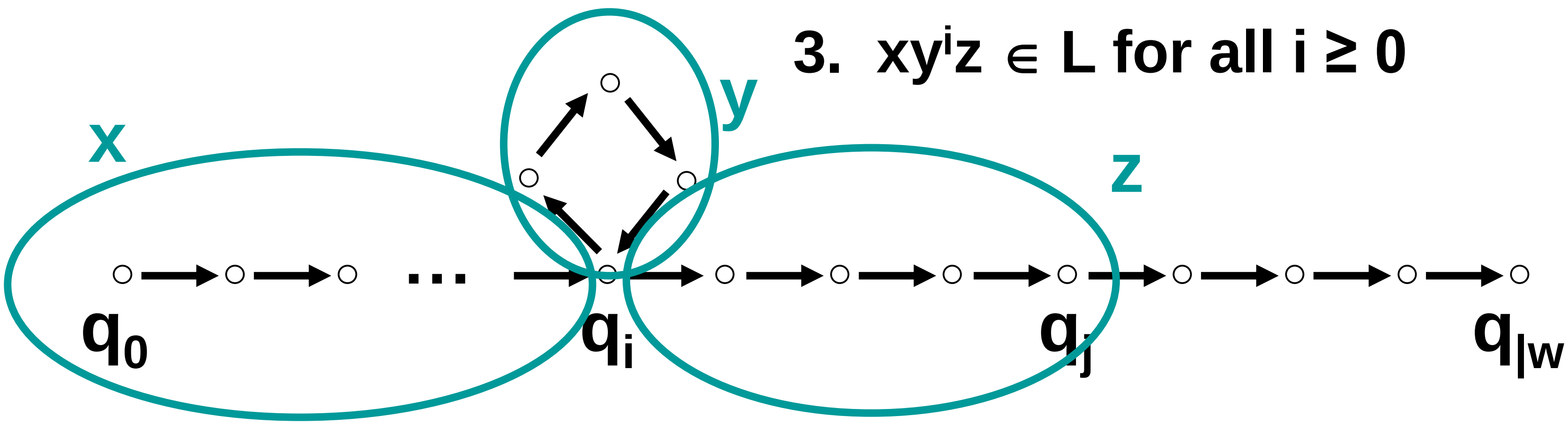
Let $L = (0 \cup 1)2^*$; $p = 2$
$w = 12$
$x = 1$
$y = 2$
$z = \varepsilon$

Proof of the pumping lemma

Let M be a DFA that recognizes L.
Let p be the number of states in M.
Assume $w \in L$ is such that $|w| \geq p$.

We show w = xyz

1. $|y| > 0$
2. $|xy| \leq p$
3. $xy^iz \in L$ for all $i \geq 0$



There must be j > i such that $q_i = q_j$.

CS 464

USING THE **PUMPING LEMMA**

Use the pumping lemma to prove that
B = $\{0^n1^n \mid n \geq 0\}$ is not regular

Hint: Assume B is regular, and try pumping w = 0^P1^P

If B is regular, w can be split into w = xyz, where

1. **$|y| > 0$**
2. **$|xy| \leq p$**
3. **$xy^iz \in B$ for all $i \geq 0$**

y is all 0s: xyyz has more 0s than 1s

Contradiction!

CS 464

GENERAL STRATEGY

Proof by contradiction: assume L is regular.

Then there is **a pumping length p.**

Find a string w $\in$ L with $|w| \geq p$.

Show that no matter how you choose xyz, w cannot be pumped!

Conclude that L is not regular.

CS 464

USING THE **PUMPING LEMMA**

PALINDROMES = $\{ ww^R \mid w \in \{0,1\}^* \}$ is not regular.

Proof: Assume … pumping length p

Find a w ∈ PALINDROMES longer than p

$0^P1^P1^P0^P$

Show that w cannot be pumped:

p 2p p

w= 00…0011…1100…00

y must be in this part

xyyz = 00…00011…1100…00

> p 2p p

CS 464

USING THE **PUMPING LEMMA**

PALINDROMES = { ww^R | w ∈ {0,1}* } is not regular.

Proof: Assume … pumping length p

Find a w ∈ PALINDROMES longer than p

$0^P1^P1^P0^P$

Show that w cannot be pumped:

If w = xyz with |xy| ≤ p then

y = 0^J for some J>0.

Then $xyyz$ = $0^{P+J}1^{2P}0^P$ ∉ PALINDROMES

Contradiction!

PUMPING **DOWN**

Prove $C = \{ 0^i 1^j \mid i > j \geq 0\}$ is not regular.

Proof: **Assume … pumping length p**

Find a w ∈ C longer than p

$0^{P+1}1^P$

Show that w cannot be pumped:

$$w = \overbrace{00\ldots00}^{p+1}\overbrace{11\ldots11}^{p}$$

y must be in this part (the 0s)

$$xyyyz = \underbrace{00\ldots000}_{> p+1}011\ldots11 \qquad xz = \underbrace{0\ldots00}_{\leq p}11\ldots11$$

PUMPING **DOWN**

Prove $C = \{ 0^i 1^j \mid i > j \geq 0 \}$ is not regular.

Proof: **Assume … pumping length p**

Find a w ∈ C longer than p

$0^{P+1}1^P$

If w = xyz with $|xy| \geq$ p then

$y = 0^J$ for some $J \geq 1$.

Then $xy^0z = xz = 0^{P+1-J}1^P \notin C$

Contradiction!

CS 464

Pumping lemma as a game

1. YOU pick the language L to be proved nonregular.
2. ADVERSARY picks p, but doesn't reveal to YOU what p is; YOU must devise a play for all possible p's.
3. YOU pick w, which should depend on p and which must be of length at least p.
4. ADVERSARY divides w into x, y, z, obeying conditions stipulated in the pumping lemma: $|y| > 0$ and $|xy| \leq n$. Again, ADVERSARY does not tell YOU what x, y, z are, although they must obey the constraints.
5. YOU win by picking i, which may be a function of p, x, y, z, such that $xy^i z$ is not in L.