TDDD14/TDDD85

Slides for Lecture 4, 2017

Slides originally for TDDD65 by Gustav Nordh

Some differences to Kozen:

- Closure properties for regular languages uses $\varepsilon$-NFA constructions instead of DFA constructions.
- Patterns are not used.
- Conversion of DFA to regular expression uses the GNFA method, instead of Kozens method.

# Closure properties of regular languages

The natural numbers $\mathbb{N} = \{0, 1, 2, 3, \ldots\}$ are closed under multiplication in the sense that for any natural numbers $x$ and $y$, $x \cdot y$ is again a natural number

The natural numbers are not closed under subtraction ($3 - 5 = -2$ which is not a natural number)

**Definition**

We say that a class of languages $\mathcal{C}$ is closed under an operation *op* if applying *op* to any languages from $\mathcal{C}$ results in a language in $\mathcal{C}$

# Closure properties of regular languages

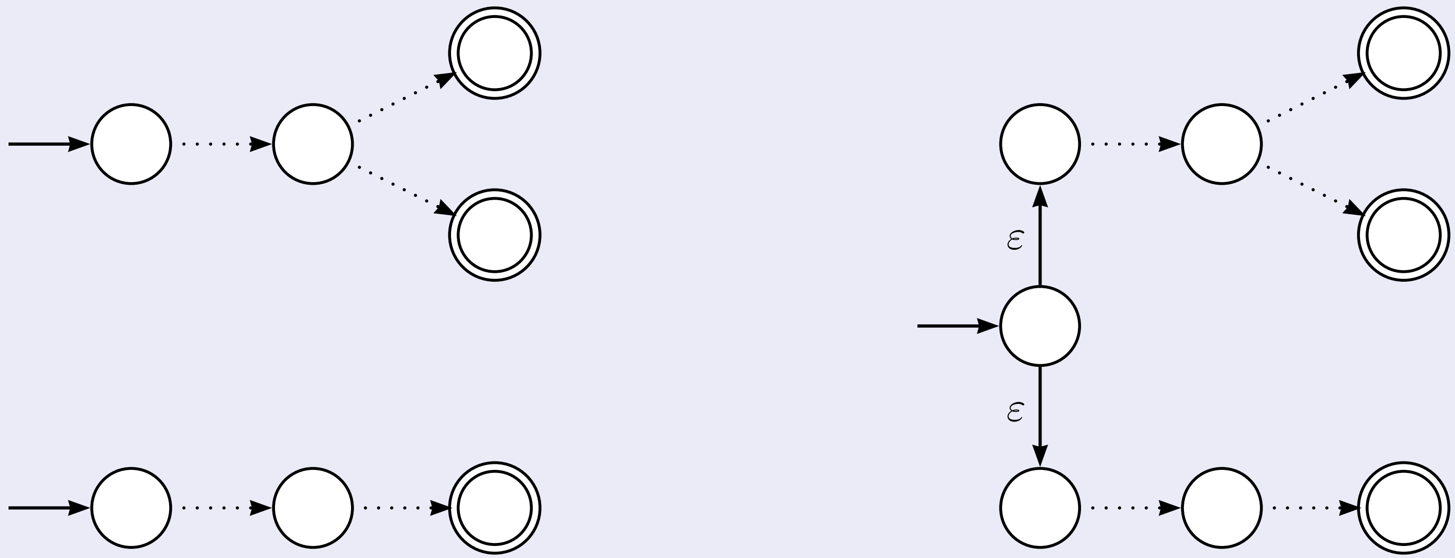
Understanding under which operations a class of languages $\mathcal{C}$ is closed is important!

# Closure properties of regular languages

**Theorem**

*The class of regular languages is closed under union (if $L_1$ and $L_2$ are regular languages, then so is $L_1 \cup L_2$)*

**Proof.**



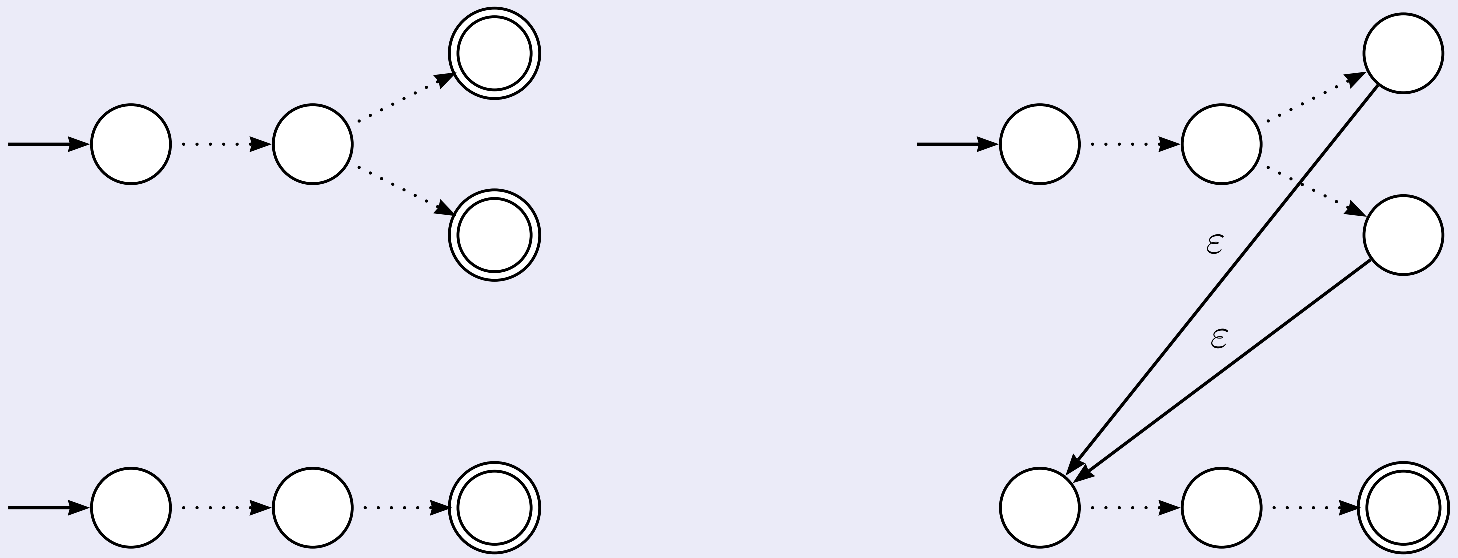

□

# Closure properties of regular languages

**Theorem**

*The class of regular languages is closed under concatenation (if $L_1$ and $L_2$ are regular languages, then so is $L_1L_2$)*

**Proof.**



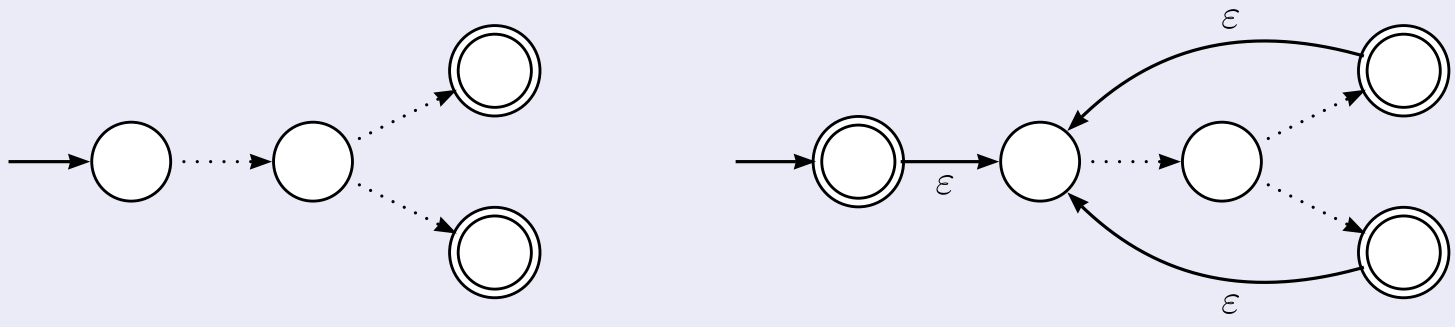

□

# Closure properties of regular languages

**Theorem**

*The class of regular languages is closed under star (if $L_1$ is a regular language, then so is $L_1^*$)*

**Proof.**




□

# Regular expressions

# Definition of regular expressions

## Definition

$L(R)$ denotes the language described by the regular expression $R$.

$R$ is a regular expression if $R$ is

1. $a$ for $a \in \Sigma$, $L(a) = \{a\}$
2. $\varepsilon$, $L(\varepsilon) = \{\varepsilon\}$
3. $\emptyset$, $L(\emptyset) = \emptyset$
4. $R_1 + R_2$ where $R_1$ and $R_2$ are regular expressions, $L(R_1 + R_2) = L(R_1) \cup L(R_2)$
5. $R_1R_2$ where $R_1$ and $R_2$ are regular expressions, $L(R_1R_2) = L(R_1)L(R_2)$
6. $R_1^*$ where $R_1$ is a regular expression, $L(R_1^*) = L(R_1)^*$

$^*$ has higher precedence than concatenation and $+$,
concatenation has higher precedence than $+$

# Examples of regular expressions

## Example

- $(0+1)^*0$ binary strings ending with 0
- $(0+1)^*00(0+1)^*$ binary strings with at least two consecutive 0's
- $(0+1+2+3+4+5+6+7+8+9)^*1234(0+1+2+3+4+5+6+7+8+9)^*$

# Equivalence with finite automata

**Theorem**

*A language is regular if and only if some regular expression describes it*

# Equivalence with finite automata

**Lemma**

*If a language is described by a regular expression then it is recognized by a NFA*

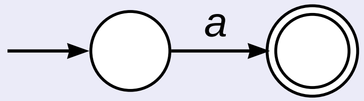
# Equivalence with finite automata

**Lemma**

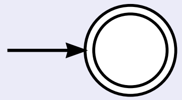
*If a language is described by a regular expression then it is recognized by a NFA*

**Proof.**

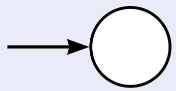
- $R = a$ for $a \in \Sigma$, $L(R) = \{a\}$




- $R = \varepsilon$, $L(R) = \{\varepsilon\}$



- $R = \emptyset$, $L(R) = \emptyset$



$\square$

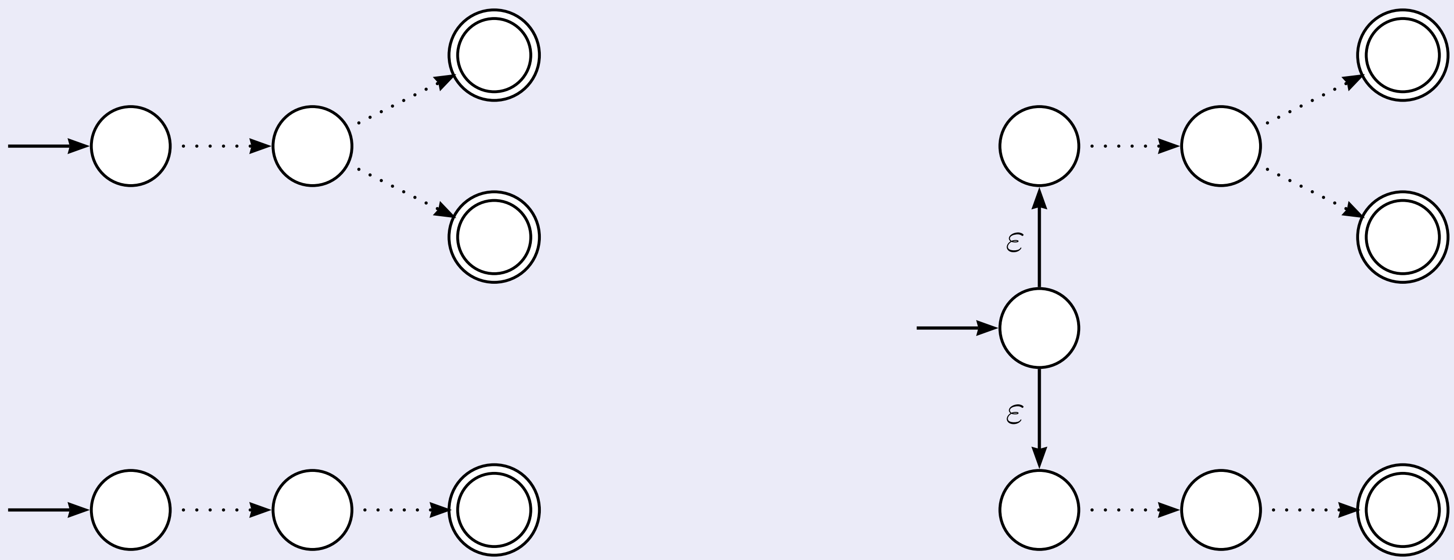
# Equivalence with finite automata

## Lemma

*If a language is described by a regular expression then it is recognized by a NFA*

## Proof.

- $R = R_1 + R_2$, $L(R) = L(R_1) \cup L(R_2)$




□

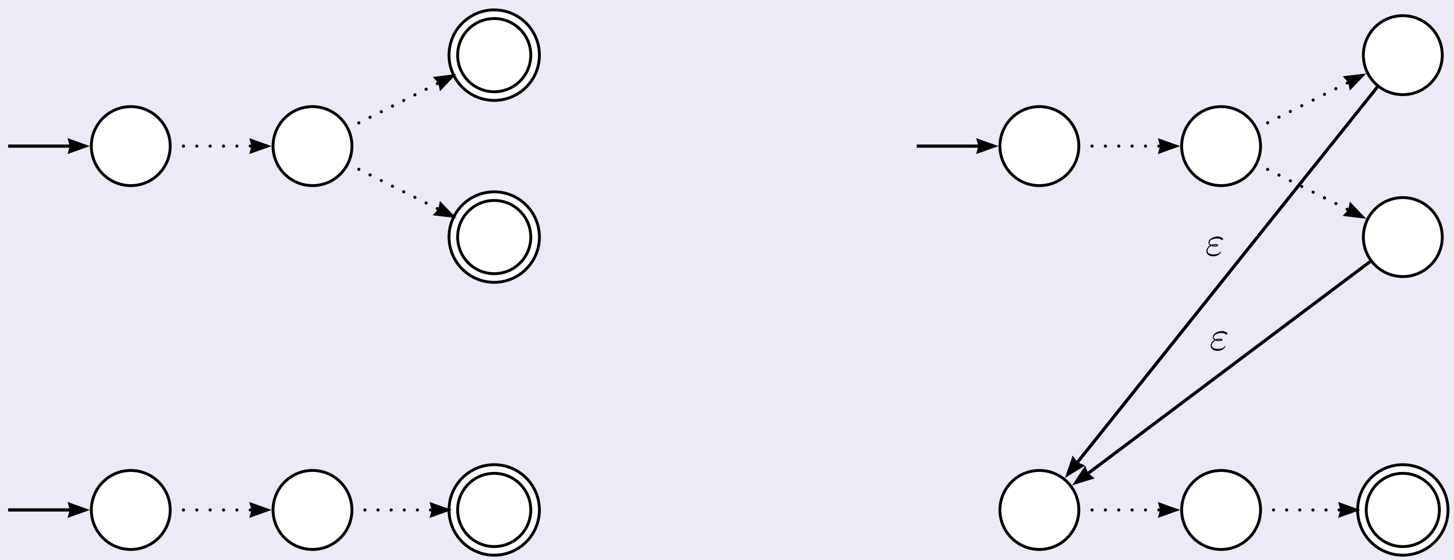
# Equivalence with finite automata

**Lemma**

*If a language is described by a regular expression then it is recognized by a NFA*

**Proof.**

- $R = R_1R_2$, $L(R) = L(R_1)L(R_2)$




□

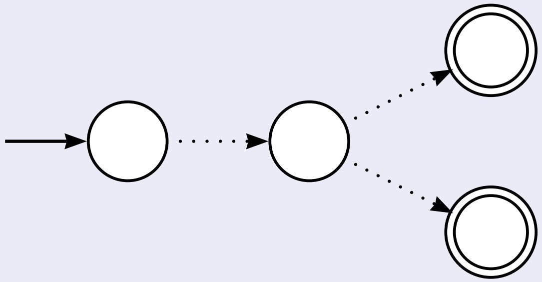
# Equivalence with finite automata

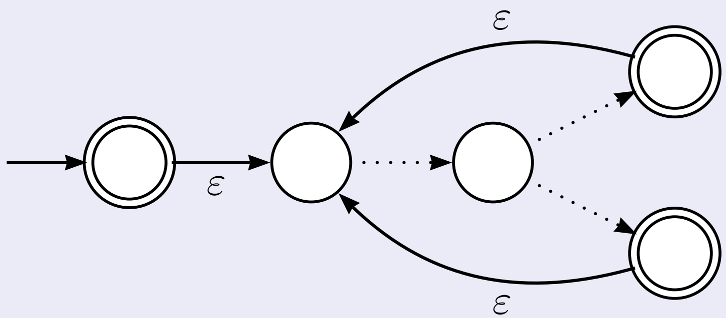
## Lemma

*If a language is described by a regular expression then it is recognized by a NFA*

## Proof.

- $R = R_1^*$, $L(R) = L(R_1)^*$





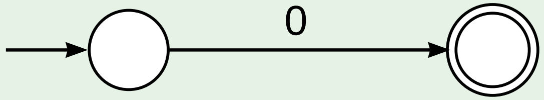

□

# Equivalence with finite automata: Example

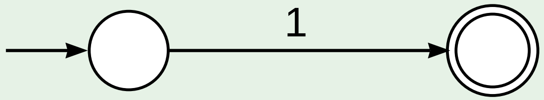
## Example

$(0+1)^*$ to NFA

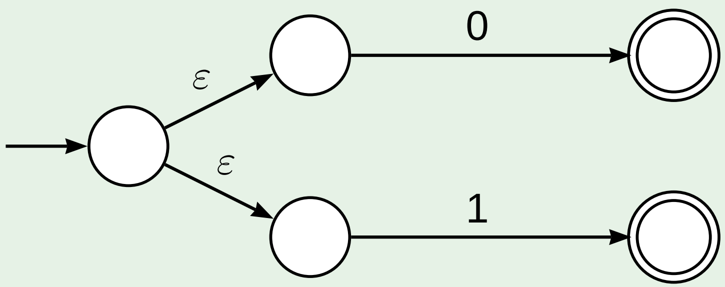
- $0$



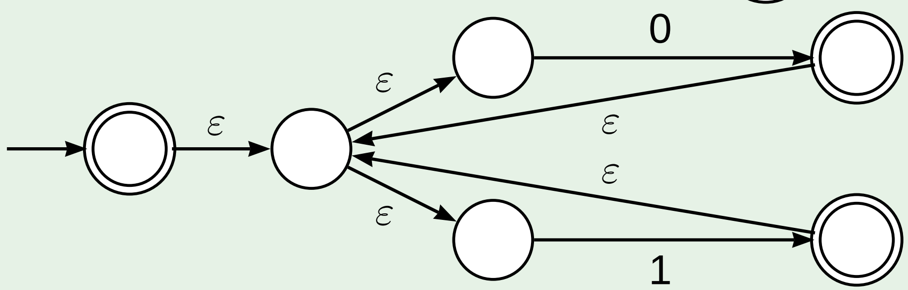

- $1$



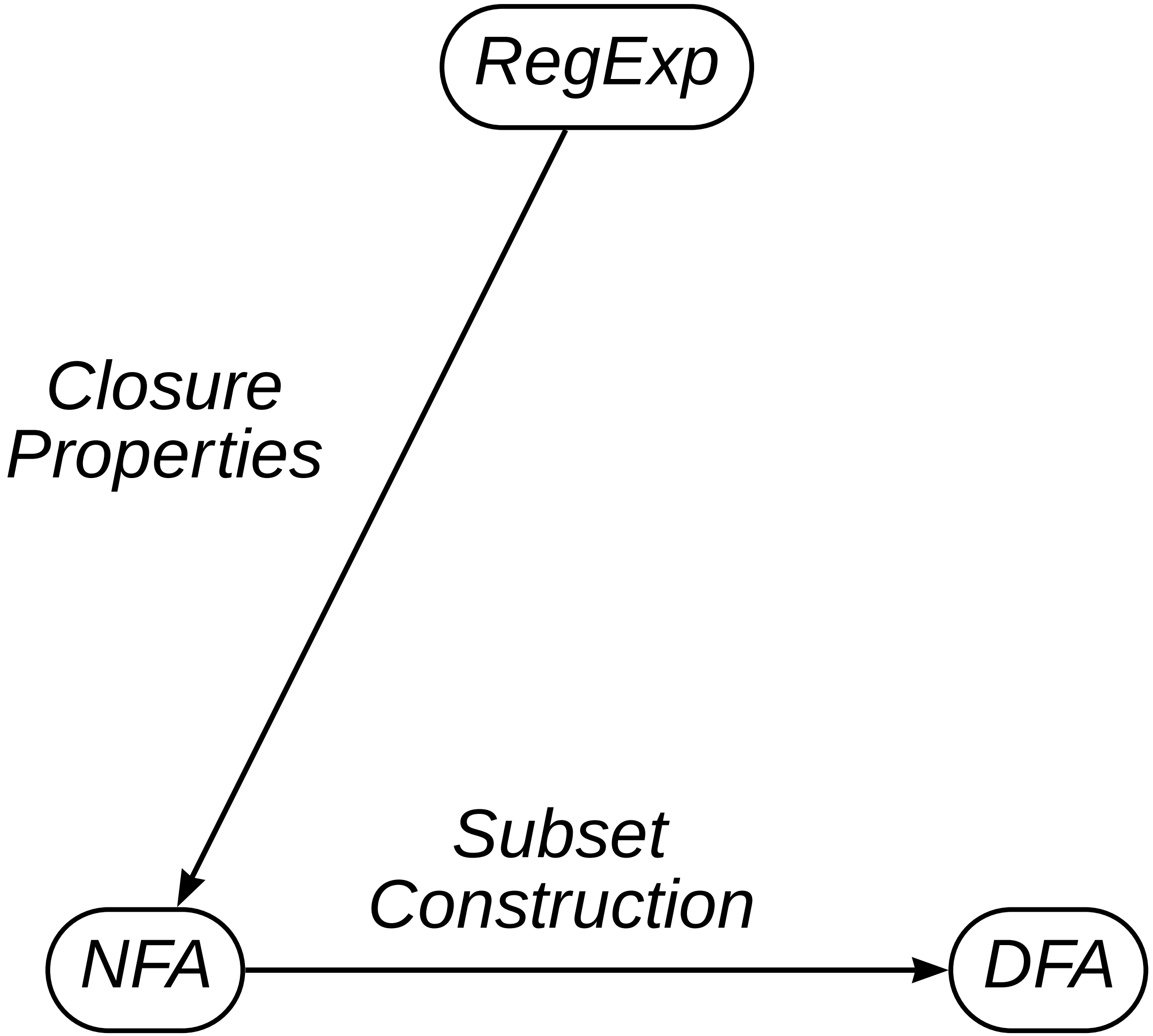

- $0+1$



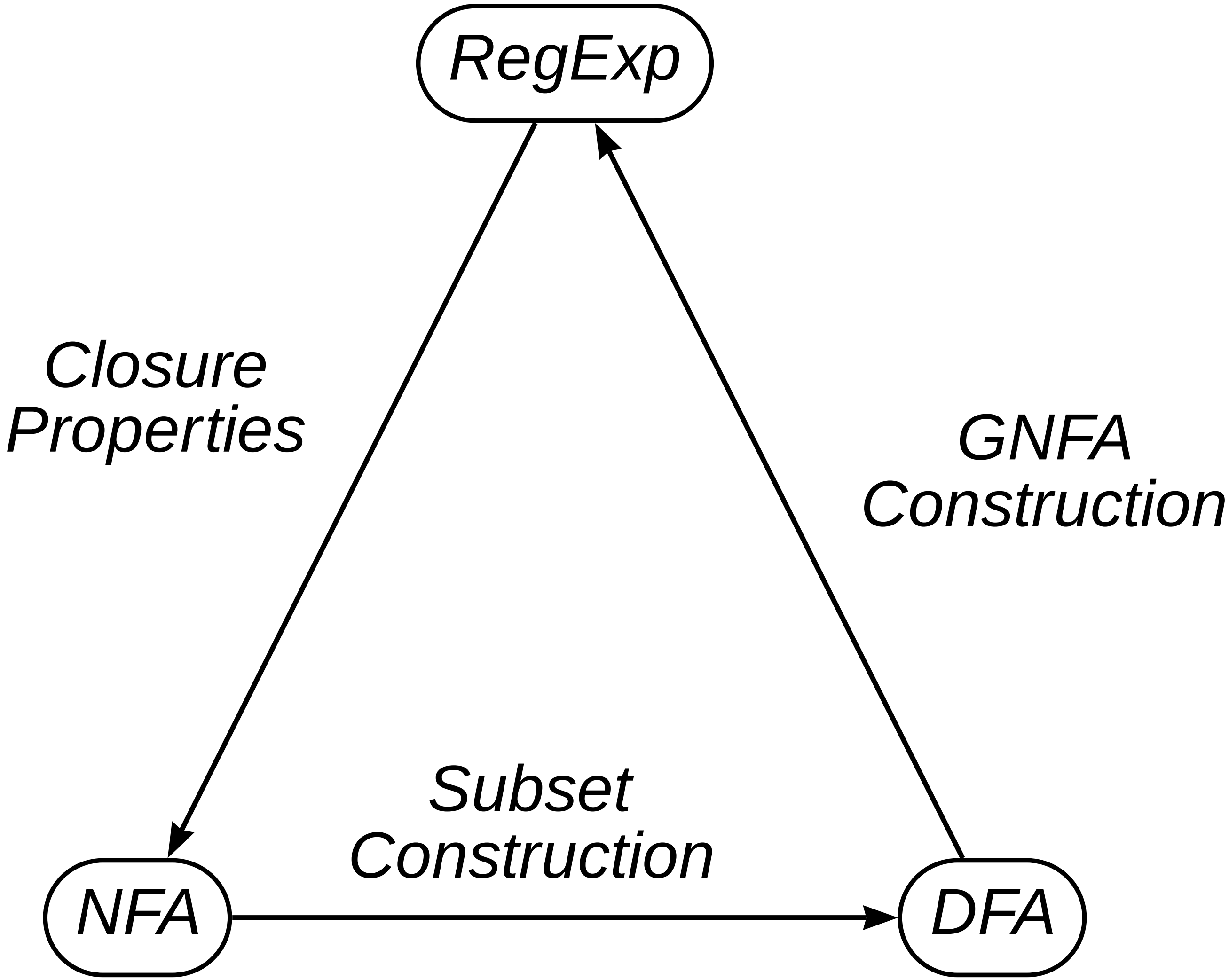

- $(0+1)^*$



# Equivalence with finite automata



# Equivalence with finite automata

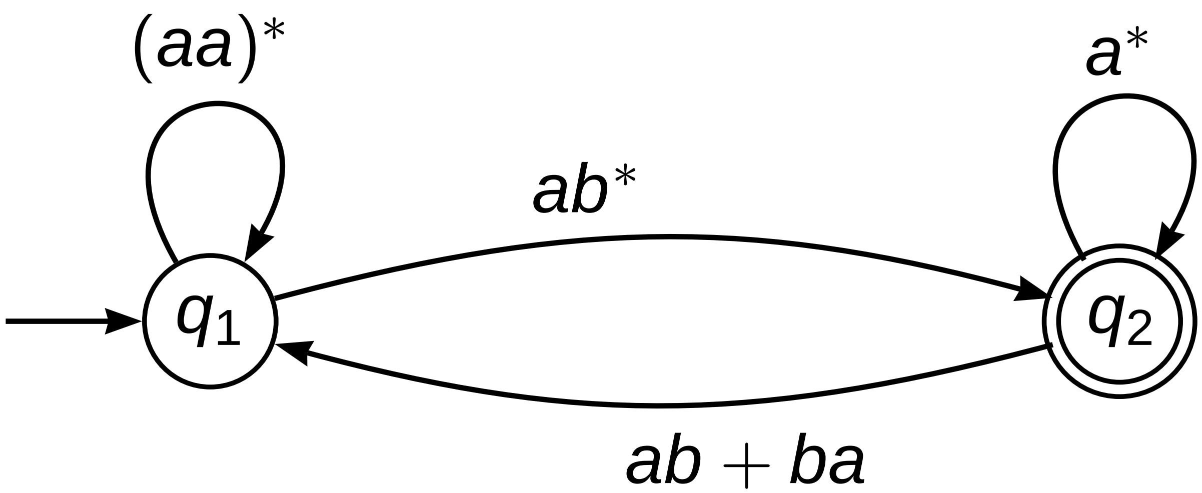


# Equivalence with finite automata

**Lemma**

*If a language is recognized by a DFA then it is described by a regular expression*

Idea: Use a generalized NFA (GNFA) where the transition arrows can be labeled by regular expressions



# Equivalence with finite automata

**Lemma**

*If a language is recognized by a DFA then it is described by a regular expression*

Idea: Use a generalized NFA (GNFA) where the transition arrows can be labeled by regular expressions

Given a DFA

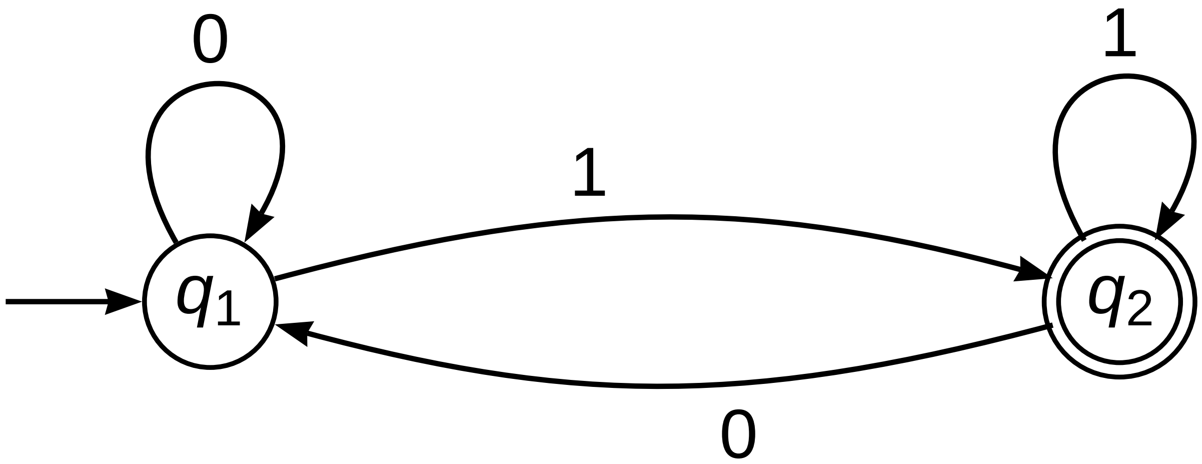
- Add a new start state with an $\varepsilon$ transition to the old start state
- Add a new accept state with $\varepsilon$ transitions from all old accept states
- Replace transitions of the form $a, b, c$ by $a + b + c$

# Equivalence with finite automata

**Lemma**

*If a language is recognized by a DFA then it is described by a regular expression*

- Add a new start state with an $\varepsilon$ transition to the old start state
- Add a new accept state with $\varepsilon$ transitions from all old accept states
- Replace transitions of the form $a, b, c$ by $a + b + c$

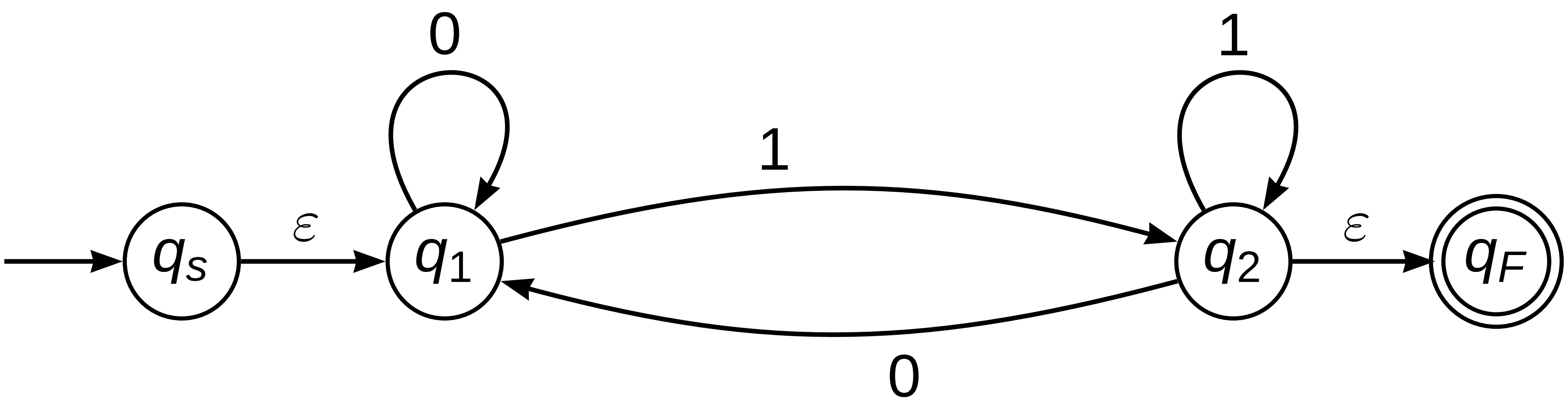


# Equivalence with finite automata

**Lemma**

*If a language is recognized by a DFA then it is described by a regular expression*

- Add a new start state with an $\varepsilon$ transition to the old start state
- Add a new accept state with $\varepsilon$ transitions from all old accept states
- Replace transitions of the form $a, b, c$ by $a + b + c$



# Equivalence with finite automata

**Lemma**

*If a language is recognized by a DFA then it is described by a regular expression*

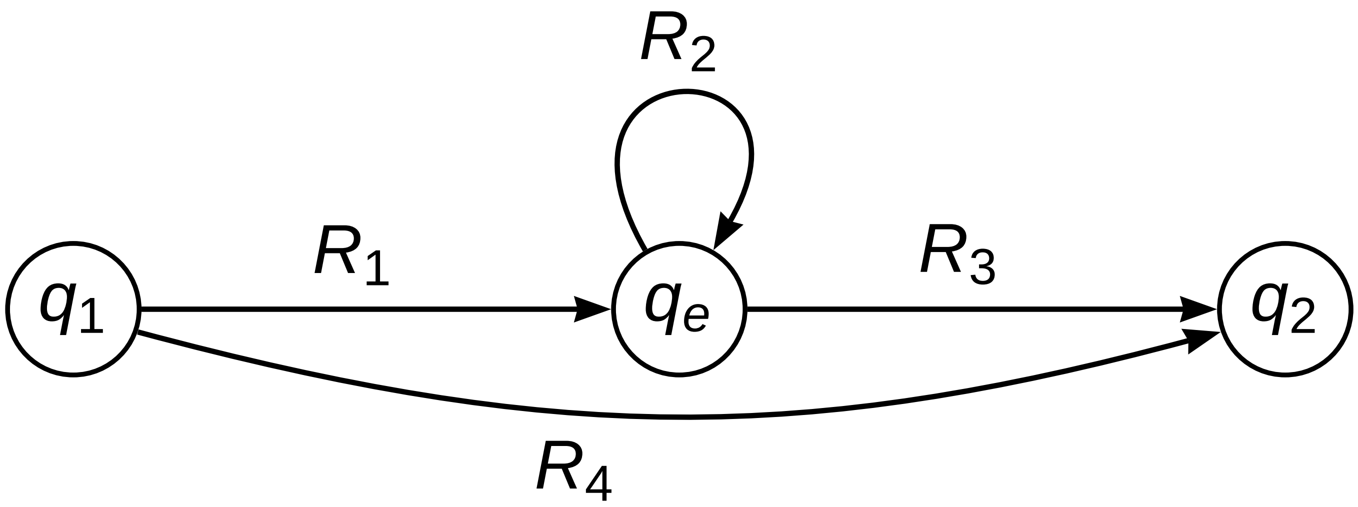
- Add a new start state with an $\varepsilon$ transition to the old start state
- Add a new accept state with $\varepsilon$ transitions from all old accept states
- Replace transitions of the form $a, b, c$ by $a + b + c$
- Eliminate a state different from the start and accept state (reducing the number of states by 1)

# Equivalence with finite automata

**Lemma**

*If a language is recognized by a DFA then it is described by a regular expression*

- Eliminate a state different from the start and accept state (reducing the number of states by 1)

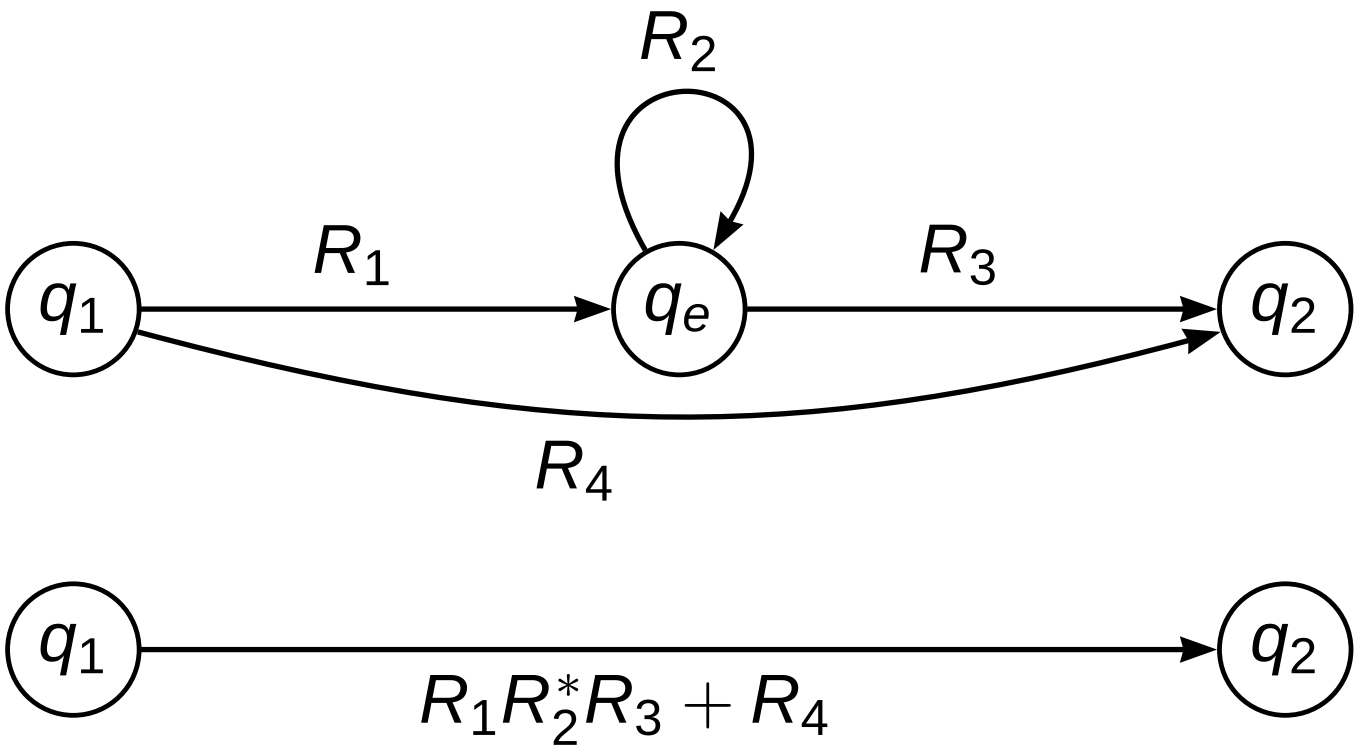


# Equivalence with finite automata

**Lemma**

*If a language is recognized by a DFA then it is described by a regular expression*

- Eliminate a state different from the start and accept state (reducing the number of states by 1)

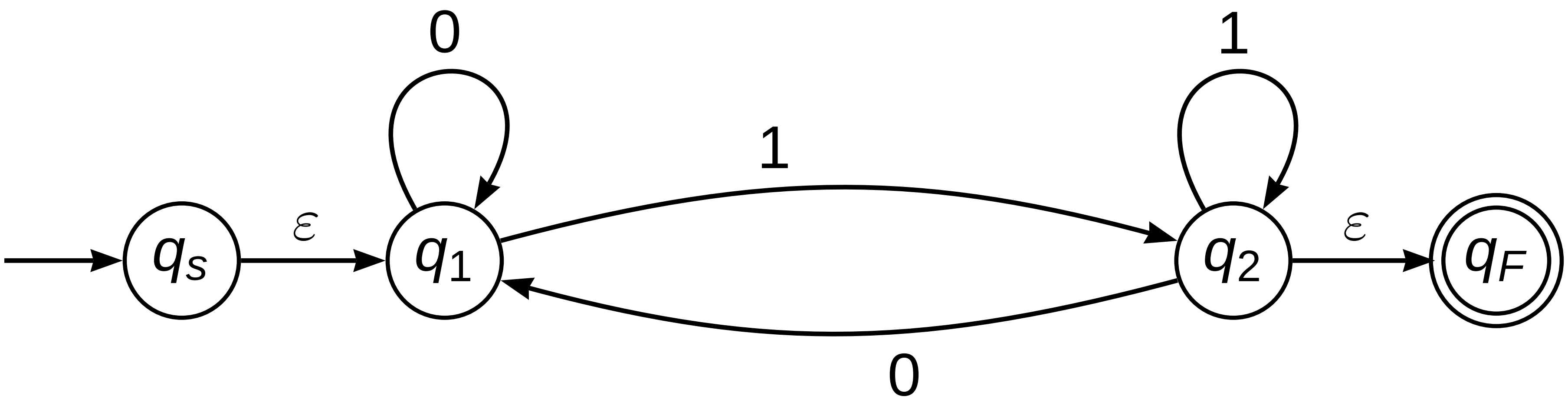


# Equivalence with finite automata

**Lemma**

*If a language is recognized by a DFA then it is described by a regular expression*

- Eliminate a state different from the start and accept state (reducing the number of states by 1)

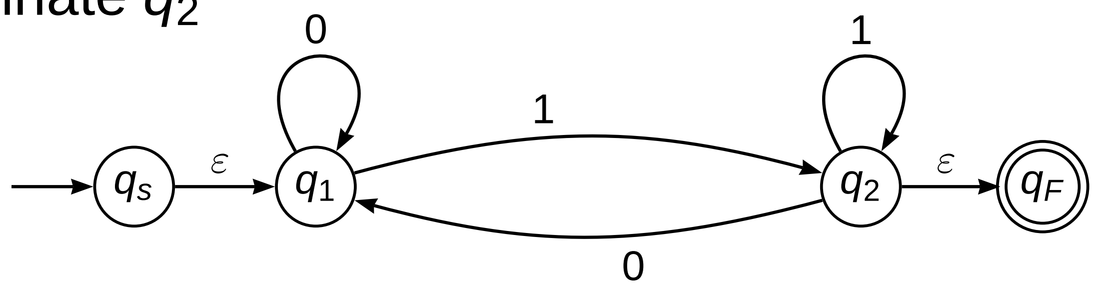


# Equivalence with finite automata

**Lemma**

*If a language is recognized by a DFA then it is described by a regular expression*

- Eliminate $q_2$

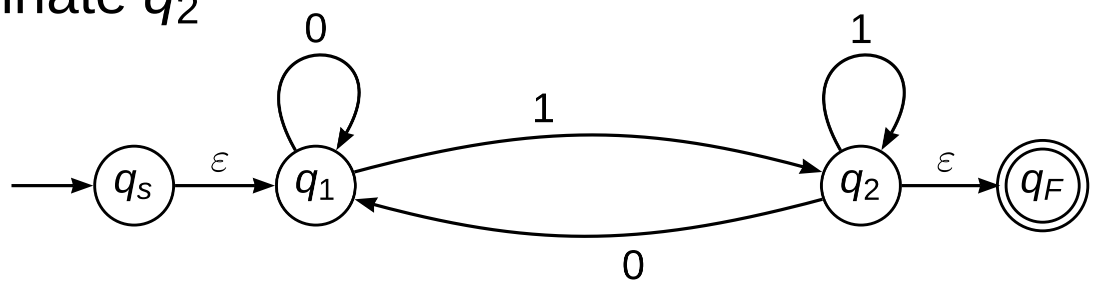


# Equivalence with finite automata

**Lemma**

*If a language is recognized by a DFA then it is described by a regular expression*

- Eliminate $q_2$



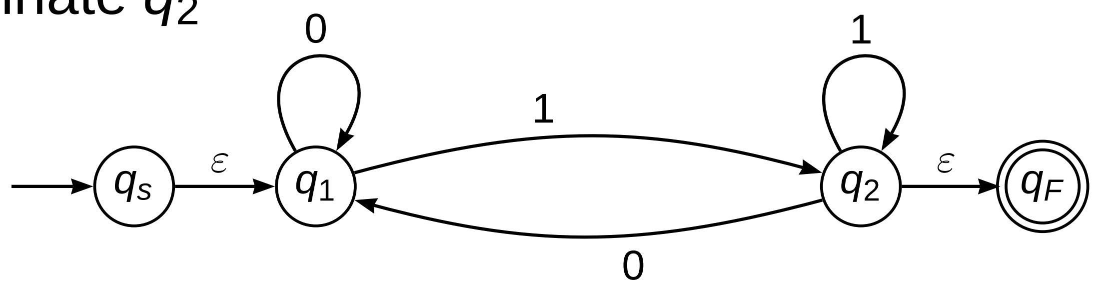

Using the rule $R_1 R_2^* R_3 + R_4$ the new transition from $q_1$ to $q_F$ is labeled $11^*\varepsilon + \emptyset$

# Equivalence with finite automata

**Lemma**

*If a language is recognized by a DFA then it is described by a regular expression*

- Eliminate $q_2$




Using the rule $R_1R_2^*R_3 + R_4$ the new transition from $q_1$ to $q_F$ is labeled $11^*\varepsilon + \emptyset$

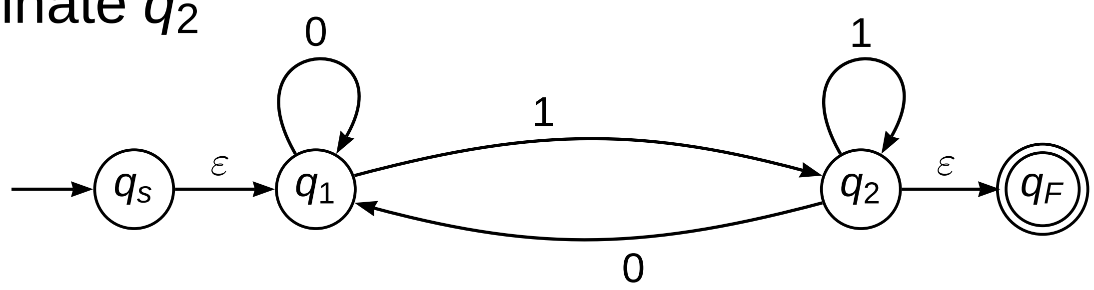
Using the rule $R_1R_2^*R_3 + R_4$ the new transition from $q_1$ to $q_1$ is labeled $11^*0 + 0$

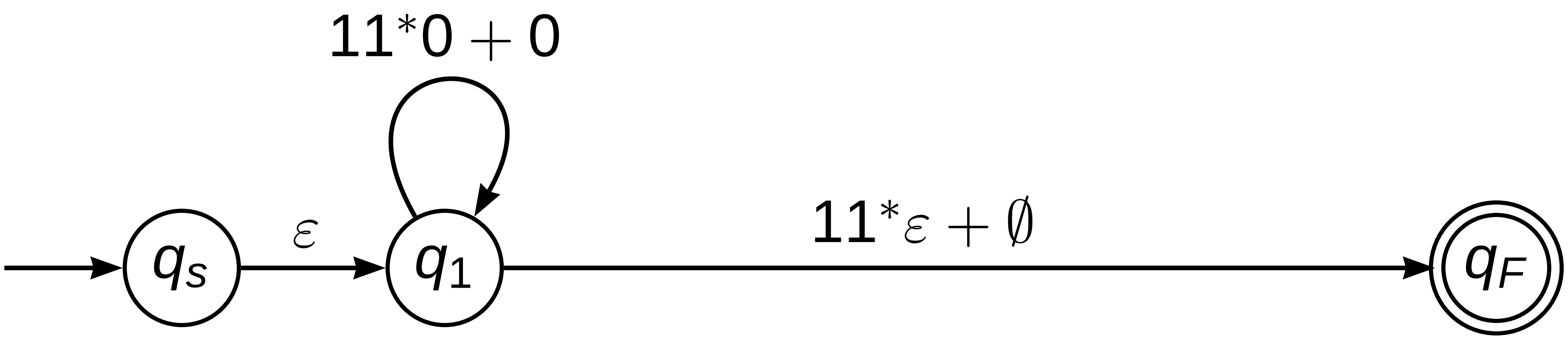
## Equivalence with finite automata

**Lemma**

*If a language is recognized by a DFA then it is described by a regular expression*

- Eliminate $q_2$



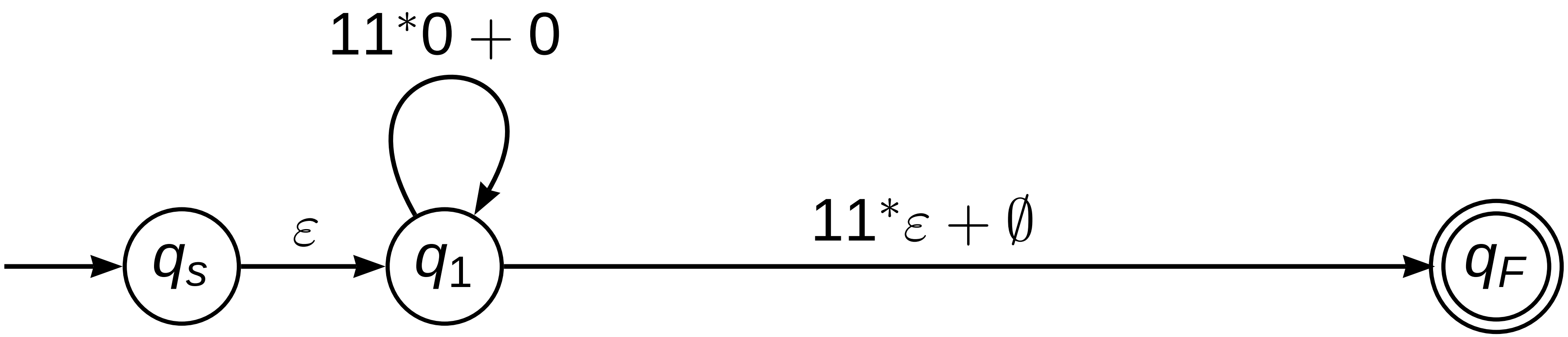



# Equivalence with finite automata

**Lemma**

*If a language is recognized by a DFA then it is described by a regular expression*

- Eliminate $q_1$

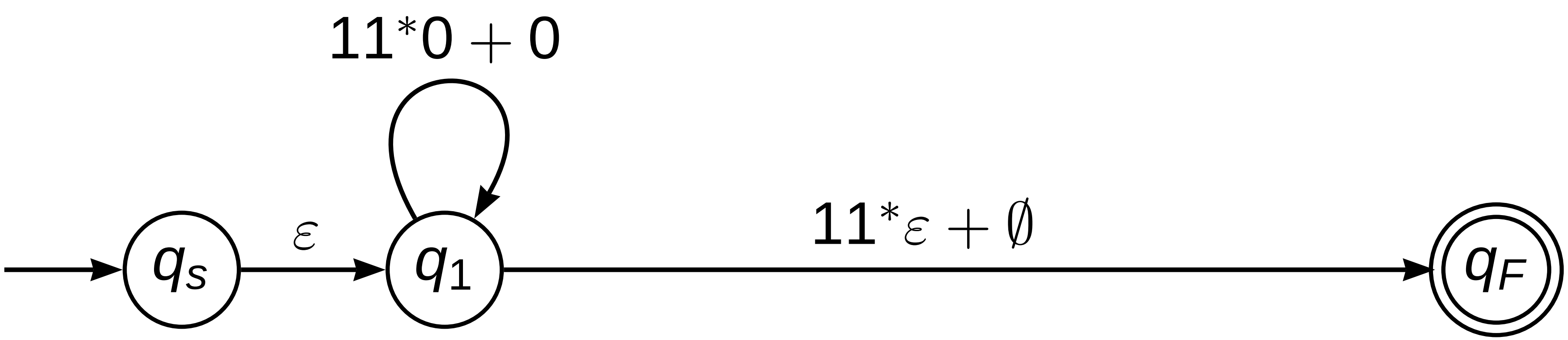


# Equivalence with finite automata

**Lemma**

*If a language is recognized by a DFA then it is described by a regular expression*

- Eliminate $q_1$



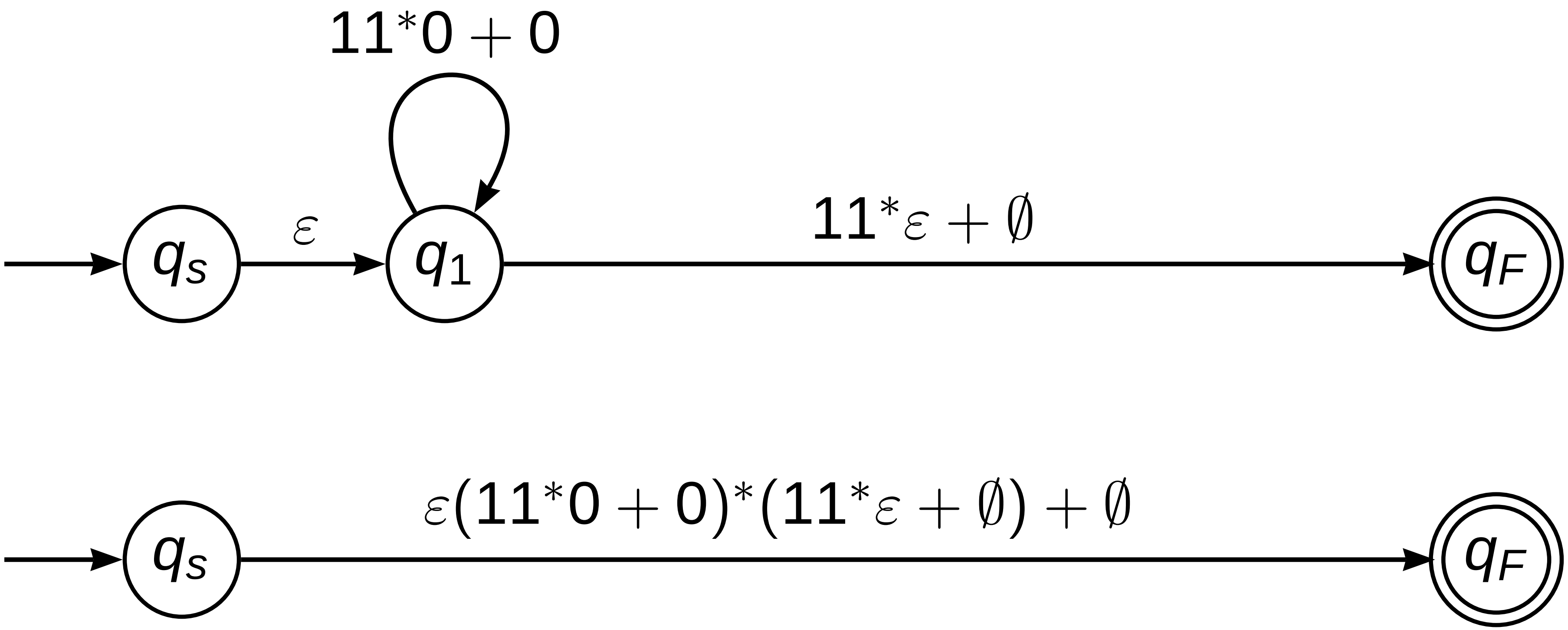

Using the rule $R_1R_2^*R_3+R_4$ the new transition from $q_s$ to $q_F$ is labeled $\varepsilon(11^*0+0)^*(11^*\varepsilon+\emptyset)+\emptyset$

# Equivalence with finite automata

**Lemma**

*If a language is recognized by a DFA then it is described by a regular expression*

- Eliminate $q_1$



# Equivalence with finite automata