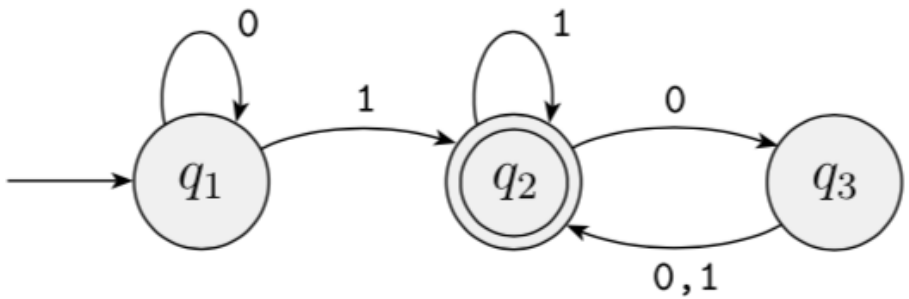
Administrivia

- Exercises, 90 minutes every week, starting this week:
  - 2x Thursday 14:15-15:45
  - 2x Thursday 16:15-17:45
  - 1x Friday 14:15-15:45
- Exercise sheet #1 will be discussed in these meetings
- Optional Exam #1: Wednesday October 29, 9:00-10:00
- Reading: Chapters 0 and 1
- Last time: DFAs, NFAs, regular languages
- Today: NFAs, properties of regular languages, regular expressions

Recall Finite Automata

A finite automaton is a 5-tuple $(Q, \Sigma, \delta, q_s, F)$, where

- Q is a finite set called the states,
- Σ is a finite set called the alphabet,
- δ is the transition function,
- $q_s \in Q$ is the start state,
- $F \subseteq Q$ is the set of accept states.



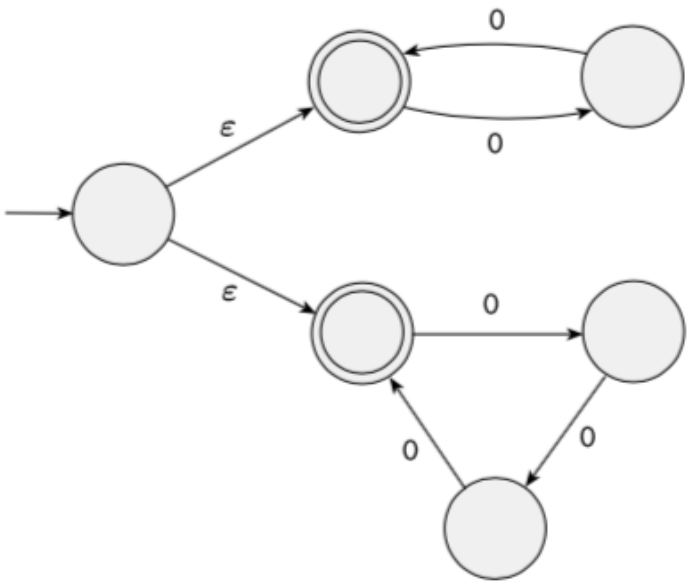
Definition

A language is called a **regular language** if some finite automaton recognizes it.

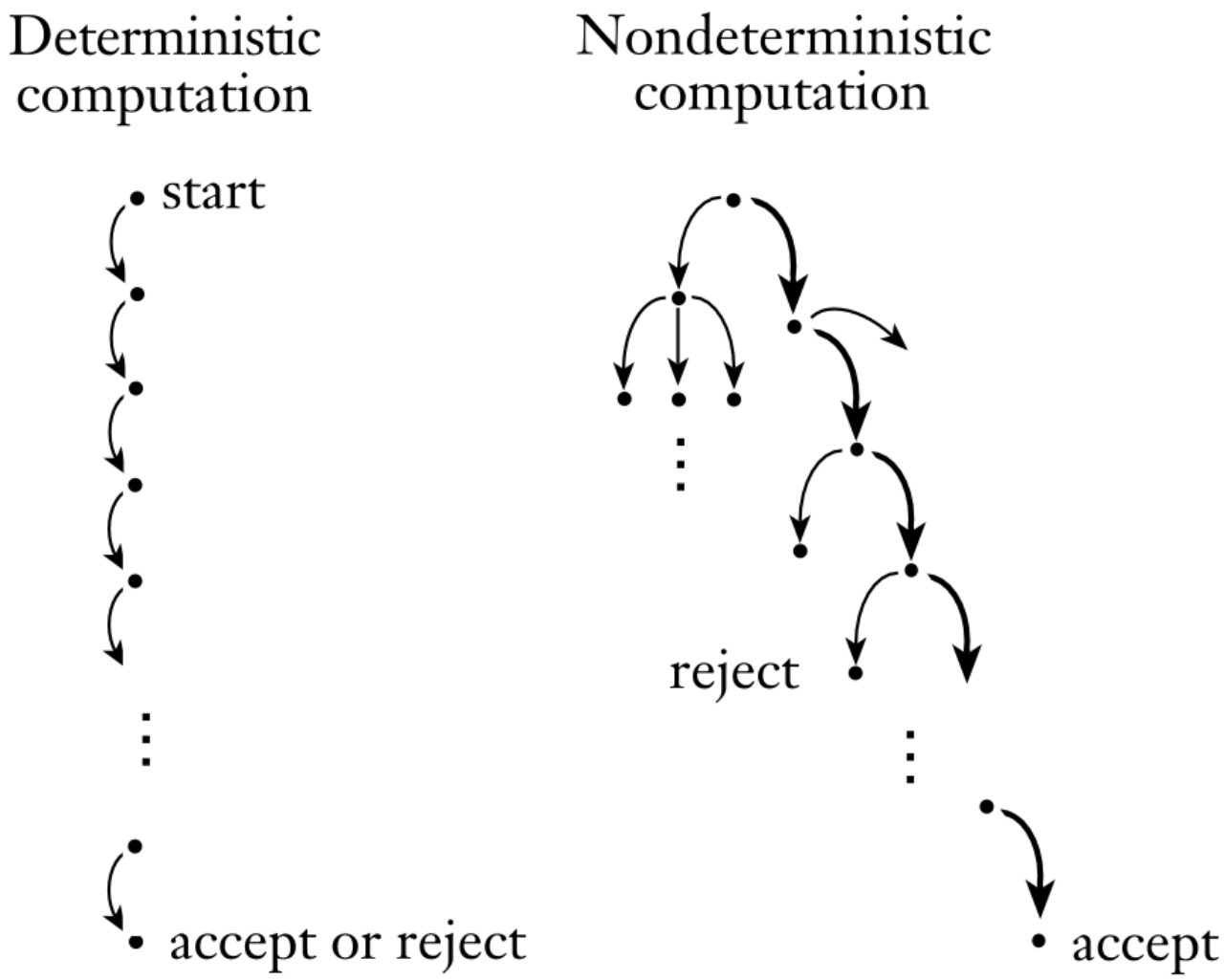
Nondeterministic Finite Automata

A non-deterministic finite automaton is a 5-tuple $(Q, \Sigma, \delta, q_s, F)$, where

- Q is a finite set called the states,
- Σ is a finite set called the alphabet,
- $\delta : Q \times \Sigma_\epsilon \to P(Q)$ is the transition function,
- $q_s \in Q$ is the start state,
- $F \subseteq Q$ is the set of accept states.



NFA Computation



An NFA accepts if any of its (possibly exponentially many) computation paths ends in an accept state.

NFAs

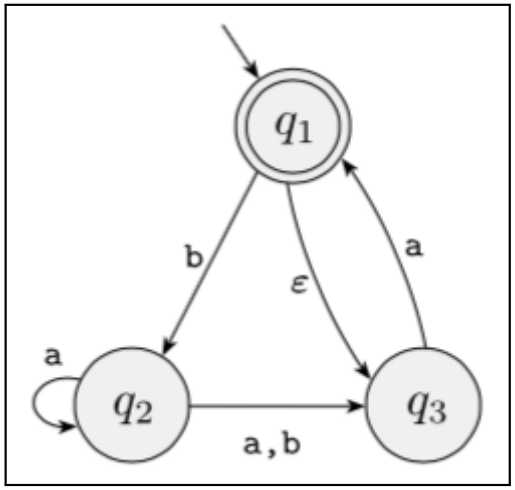
Theorem

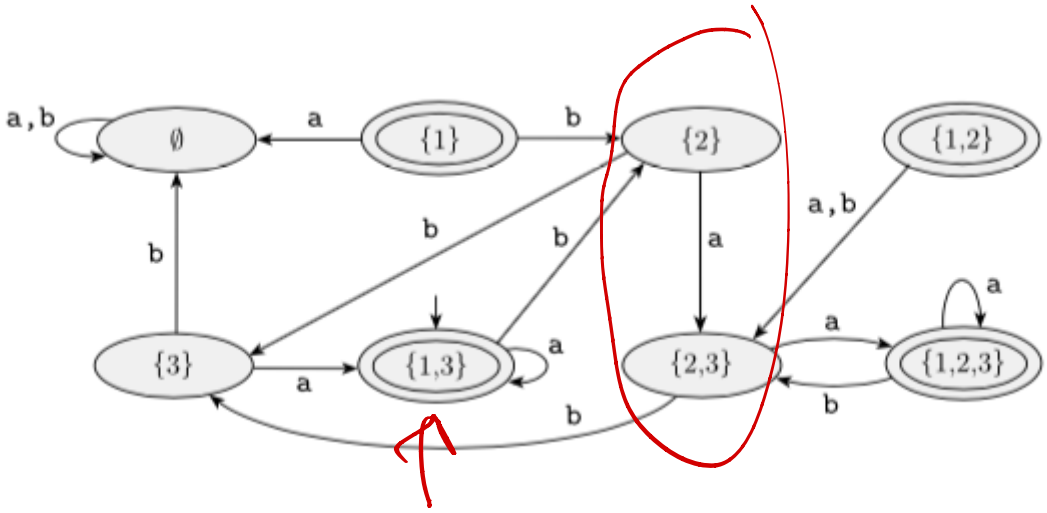
A language is regular if and only if some NFA recognizes it.

Proof.

$\Rightarrow$ Let L be a regular language. Then by definition there exists a DFA for L. Since a DFA is a special case of an NFA (an NFA without multiple transitions on the same symbol), we are done.
$\Leftarrow$ Let N be an NFA. We will show how to create an equivalent DFA M. Then $L(N) = L(M)$ and by definition, this language is regular. $\square$

NFA to DFA Example





To create the DFA M that recognizes the same language as the given NFA N, we make M *simulate* the behavior of N.

Note that the number of states in the DFA could be exponentially larger than the number of states in the NFA.

Equivalence of NFAs and DFAs (cont.)

Theorem

Every NFA has an equivalent DFA

Proof.

Let $N = (Q, \Sigma, \delta, q_s, F)$ be an NFA that recognizes language A.
We will construct DFA $M = (Q', \Sigma, \delta', q'_s, F')$ that recognizes A.

- $Q' = P(Q)$
- for $R \in Q'$ and $a \in \Sigma$ let $\delta'(R, a) = \{q \in Q | q \in E(\delta(r, a))$ for some $r \in R\}$
- $q'_S = E(q_s)$
- $F' = \{R \in Q' | R$ contains an accept state of $N\}$

where $E(R) = \{q | q$ can be reached from R by traveling along 0 or more ϵ arrows. $\square$

Closure under Union

Theorem

The class of regular languages is closed under the union operation.

Proof.

We proved this with DFAs; now we redo it with NFAs.

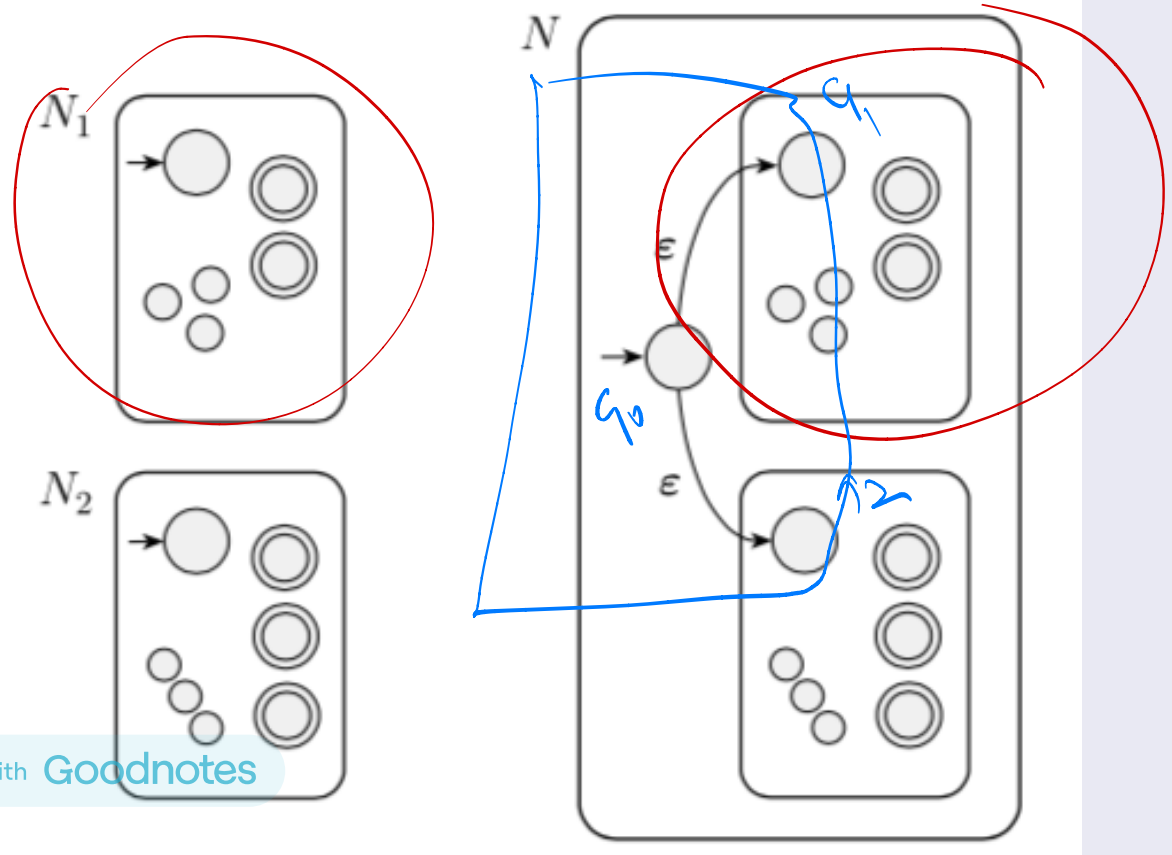
Closure under Union

Theorem

The class of regular languages is closed under the union operation.

Proof.

We proved this with DFAs; now we redo it with NFAs.



Closure under Union

Theorem

The class of regular languages is closed under the union operation.

Proof.

Let $N_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$ recognize A_1, and
$N_2 = (Q_2, \Sigma, \delta_2, q_2, F_2)$ recognize A_2.
Construct $N = (Q, \Sigma, \delta, q_0, F)$ to recognize $A_1 \cup A_2$.

- $Q = \{q_0\} \cup Q_1 \cup Q_2$.
- start state q_0
- $F = F_1 \cup F_2$.

$$\delta(q, a) = \begin{cases} \delta_1(q, a) : q \in Q_1 \\ \delta_2(q, a) : q \in Q_2 \\ \{q_1, q_2\} : q = q_0, a = \epsilon \\ \emptyset : q = q_0, a \neq \epsilon \end{cases}$$

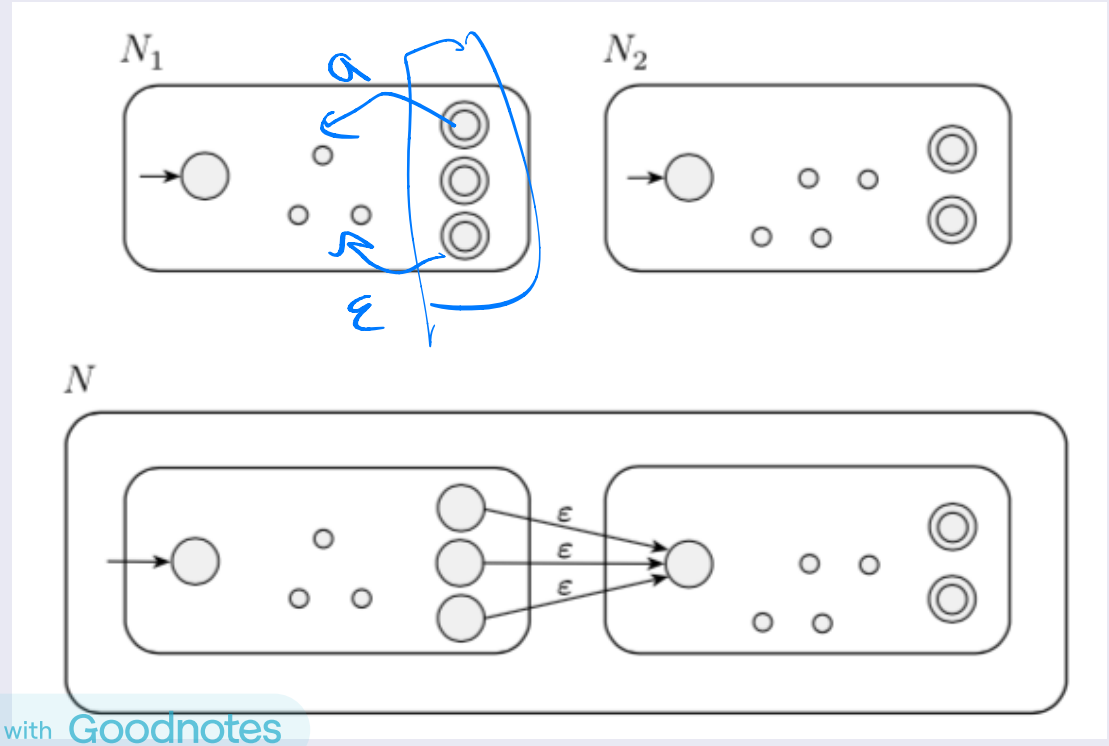
Closure under Concatenation

Recall concatenation: $A_1 \circ A_2 = \{xy | x \in A_1 \text{ and } y \in A_2\}$.

Theorem

The class of regular languages is closed under the concatenation operation.

Proof.



□

Closure under Concatenation

Theorem

The class of regular languages is closed under the concatenation operation.

Proof.

Let $N_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$ recognize A_1, and
$N_2 = (Q_2, \Sigma, \delta_2, q_2, F_2)$ recognize A_2.
Construct $N = (Q, \Sigma, \delta, q_1, F_2)$ to recognize $A_1 \circ A_2$.

- $Q = Q_1 \cup Q_2$.
- start state q_1
- accept states F_2

$$\delta(q, a) = \begin{cases} \delta_1(q, a) : q \in Q_1, q \notin F_1 \\ \delta_1(q, a) : q \in F_1, a \neq \epsilon \\ \delta_1(q, a) \cup \{q_2\} : q \in F_1, a = \epsilon \\ \delta_2(q, a) : q \in Q_2 \end{cases}$$

□

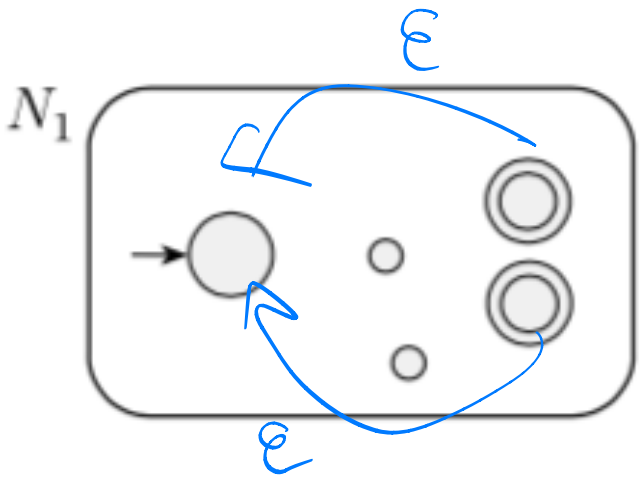
Closure under Star

Recall star: $A^* = \{x_1 x_2 \ldots x_k | k \geq 0 \text{ and each } x_i \in A\}$.

Theorem

The class of regular languages is closed under the star operation.

Proof.



almost correct
- recognizes $A^* - \{\varepsilon\}$
- we also want ε

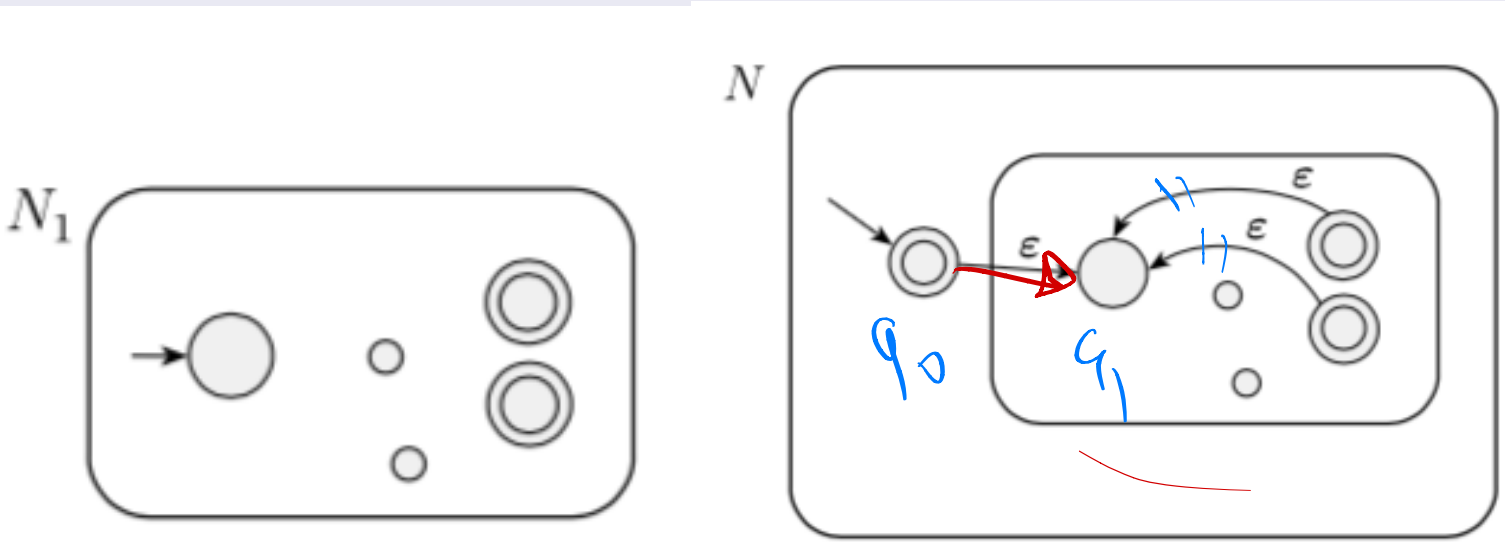
Closure under Star

Recall star: $A^* = \{x_1 x_2 \ldots x_k | k \geq 0 \text{ and each } x_i \in A\}$.

Theorem

The class of regular languages is closed under the star operation.

Proof.



□

Closure under Star

Theorem

The class of regular languages is closed under the star operation.

Proof.

Let $N_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$ recognize A_1.
Construct $N = (Q, \Sigma, \delta, q_0, F)$ to recognize A_1^*.

- $Q = \{q_0\} \cup Q_1$
- start state q_0
- $F = \{q_0\} \cup F_1$.

$$\delta(q, a) = \begin{cases} \delta_1(q, a) : q \in Q_1, q \notin F_1 \\ \delta_1(q, a) : q \in F_1, a \neq \epsilon \\ \delta_1(q, a) \cup \{q_1\} : q \in F_1, a = \epsilon \\ \{q_1\} : q = q_0, a = \epsilon \end{cases}$$

DFAs and NFAs Again

- We have proven that DFAs and NFAs are equivalent in power
- DFAs and NFAs recognize the same class of languages: the regular languages
- We used NFAs to prove some closure properites of regular languages
- Some of these proofs can be easily replicated with DFAs but others are not so easy (e.g., Kleene star, concatenation)

Regular Expressions: Examples

Let $\Sigma = \{0, 1\}$

- $0^*10^* = \{w | w \text{ contains a single 1}\}$
- $\Sigma^*1\Sigma^* = \{w | w \text{ has at least one 1}\}$
- $\Sigma^*001\Sigma^* = \{w | w \text{ contains the string 001 as a substring}\}$
- $(\Sigma\Sigma)^* = \{w | w \text{ is a string of even length}\}$
- $(\Sigma\Sigma\Sigma)^* = \{w | w \text{ has length that is a multiple of 3}\}$
- $01 \cup 10 = \{01, 10\}$
- $0\Sigma^*0 \cup 1\Sigma^*1 \cup 0 \cup 1 = \{w | w \text{ starts and ends with the same symbol}\}$
- $00^*1^* = 0^+1^* = \{w | w \text{ starts with one or more 0s and ends in zero or more 1s}\}$
- $(0 \cup \epsilon)1^* = \{01^* \cup 1^*\}$
- $(0 \cup \epsilon)(1 \cup \epsilon) = \{\epsilon, 0, 1, 01\}$
- $1^*\emptyset = \emptyset$
- $\emptyset^* = \{\epsilon\}$

Regular Expressions

R is a regular expression if R is

1. a for some a in the alphabet Σ,
2. ϵ,
3. $\emptyset$,
4. $(R_1 \cup R_2)$, where R_1 and R_2 are regular expressions,
5. $(R_1 \circ R_2)$, where R_1 and R_2 are regular expressions, or
6. (R_1^*), where R_1 is a regular expression.

RE Identities

- $R \cup \emptyset = R$
- $R \circ \epsilon = R$
- $R \cup \epsilon$ may NOT equal R – why?
  if $R = 0$, then $L(R) = \{0\}$, but $L(R \cup \epsilon) = \{0, \epsilon\}$
- $R \circ \emptyset$ may not equal R – why?
  if $R = 0$, then $L(R) = \{0\}$, but $L(R \circ \emptyset) = \emptyset$

Regular expressions are useful tools in the design of compilers

- Programming language tokens (such as variable names and constants) are usually described by REs
- a numerical constant that may include a fractional part and/or a sign may be described as a member of the language $(+ \cup - \cup \epsilon)(D^+ \cup D^+.D^* \cup D^*.D^+)$, where $D = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$ is the alphabet of decimal digits
- examples of valid strings are 42, 3.14159, +7., and -.01

RE $\iff$ FAs

Theorem

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

Proof.

$\Leftarrow$ Consider some regular expression R that describes language A. We show how to convert R into an NFA recognizing A.

1. $R = a$ for some $a \in \Sigma$. Then $L(R) = \{a\}$, and the following NFA recognizes $L(R)$

RE $\iff$ FAs

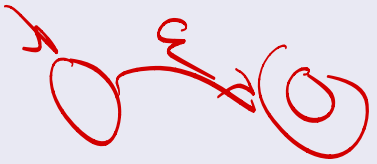
Theorem

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

Proof.

$\Leftarrow$ Consider some regular expression R that describes language A. We show how to convert R into an NFA recognizing A.

1. $R = a$ for some $a \in \Sigma$. Then $L(R) = \{a\}$, and the following NFA recognizes $L(R)$
2. $R = \epsilon$. Then $L(R) = \{\epsilon\}$, and the following NFA recognizes $L(R)$



RE $\iff$ FAs

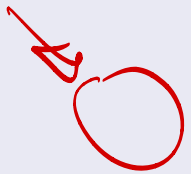
Theorem

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

Proof.

$\Leftarrow$ Consider some regular expression R that describes language A. We show how to convert R into an NFA recognizing A.

1. $R = a$ for some $a \in \Sigma$. Then $L(R) = \{a\}$, and the following NFA recognizes $L(R)$
2. $R = \epsilon$. Then $L(R) = \{\epsilon\}$, and the following NFA recognizes $L(R)$
3. $R = \emptyset$. Then $L(R) = \emptyset$, and the following NFA recognizes $L(R)$



RE $\iff$ FAs

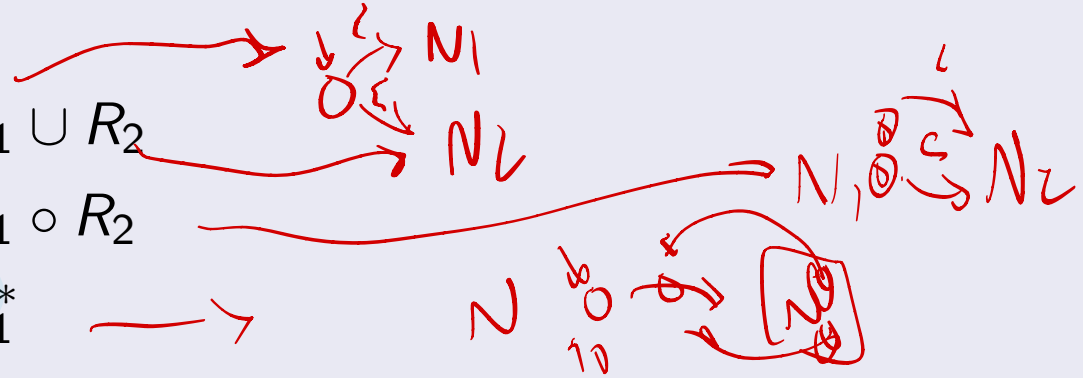
Theorem

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

Proof.

$\Leftarrow$ Consider some regular expression R that describes language A. We show how to convert R into an NFA recognizing A.

1. $R = a$ for some $a \in \Sigma$. Then $L(R) = \{a\}$, and the following NFA recognizes $L(R)$
2. $R = \epsilon$. Then $L(R) = \{\epsilon\}$, and the following NFA recognizes $L(R)$
3. $R = \emptyset$. Then $L(R) = \emptyset$, and the following NFA recognizes $L(R)$
4. $R = R_1 \cup R_2$
5. $R = R_1 \circ R_2$
6. $R = R_1^*$



Example of RE to NFA conversion

Let $R = (ab \cup a)^*$

Example of RE to NFA conversion

Let $R = (ab \cup a)^*$