

# Finite Automata Theory and Formal Languages

TMV027/DIT321– LP4 2015

Lecture 6  
Ana Bove

April 20th 2015

## Overview of today's lecture:

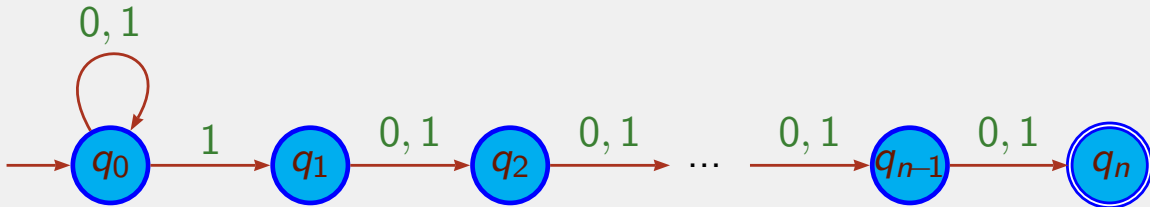
- More on NFA;
- NFA with  $\epsilon$ -Transitions;
- Equivalence between DFA and  $\epsilon$ -NFA;

## Recap: Non-deterministic Finite Automata

- Defined by a 5-tuple  $(Q, \Sigma, \delta, q_0, F)$ ;
- Why “non-deterministic”?;
- $\delta : Q \times \Sigma \rightarrow \mathcal{P}ow(Q)$ ;
- Easier to define for some problems;
- Accept set of words  $x$  such that  $\hat{\delta}(q_0, x) \cap F \neq \emptyset$ ;
- Given a NFA  $N$  we can apply the subset construction and get a DFA  $D$  ...
- ... such that  $\mathcal{L}(N) = \mathcal{L}(D)$ ;
- Hence, also accept the so called regular language.

## A Bad Case for the Subset Construction

**Proposition:** Any DFA recognising the same language as the NFA below has at least  $2^n$  states:



This NFA recognises strings over  $\{0, 1\}$  such that the  $n$ th symbol from the end is a 1.

**Proof:** Let  $\mathcal{L}_n = \{x1u \mid x \in \Sigma^*, u \in \Sigma^{n-1}\}$  and  $D = (Q, \Sigma, \delta, q_0, F)$  a DFA.

We want to show that if  $|Q| < 2^n$  then  $\mathcal{L}(D) \neq \mathcal{L}_n$ .

## A Bad Case for the Subset Construction (Cont.)

**Lemma:** If  $\Sigma = \{0, 1\}$  and  $|Q| < 2^n$  then there exists  $x, y \in \Sigma^*$  and  $u, v \in \Sigma^{n-1}$  such that  $\hat{\delta}(q_0, x0u) = \hat{\delta}(q_0, y1v)$ .

**Proof:** Let us define a function  $h : \Sigma^n \rightarrow Q$  such that  $h(z) = \hat{\delta}(q_0, z)$ .

$h$  cannot be *injective* because  $|Q| < 2^n = |\Sigma^n|$ .

Hence, we have  $a_1 \dots a_n \neq b_1 \dots b_n$  such that

$$h(a_1 \dots a_n) = \hat{\delta}(q_0, a_1 \dots a_n) = \hat{\delta}(q_0, b_1 \dots b_n) = h(b_1 \dots b_n)$$

Let us assume that  $a_i = 0$  and  $b_i = 1$ .

Let  $x = a_1 \dots a_{i-1}$ ,  $y = b_1 \dots b_{i-1}$ ,  $u = a_{i+1} \dots a_n 0^{i-1}$ ,  $v = b_{i+1} \dots b_n 0^{i-1}$ .

Hence (recall that for a DFA,  $\hat{\delta}(q, zw) = \hat{\delta}(\hat{\delta}(q, z), w)$ ):

$$\begin{aligned} \hat{\delta}(q_0, x0u) &= \hat{\delta}(q_0, a_1 \dots a_n 0^{i-1}) = \hat{\delta}(\hat{\delta}(q_0, a_1 \dots a_n), 0^{i-1}) = \\ &= \hat{\delta}(\hat{\delta}(q_0, b_1 \dots b_n), 0^{i-1}) = \hat{\delta}(q_0, b_1 \dots b_n 0^{i-1}) = \hat{\delta}(q_0, y1v) \end{aligned}$$

## A Bad Case for the Subset Construction (Cont.)

**Lemma:** If  $|Q| < 2^n$  then  $\mathcal{L}(D) \neq \mathcal{L}_n$ .

**Proof:** Assume  $\mathcal{L}(D) = \mathcal{L}_n$ .

Let  $x, y \in \Sigma^*$  and  $u, v \in \Sigma^{n-1}$  as in previous lemma.

Then,  $y1v \in \mathcal{L}(D)$  but  $x0u \notin \mathcal{L}(D)$ ,

That is,  $\hat{\delta}(q_0, y1v) \in F$  but  $\hat{\delta}(q_0, x0u) \notin F$ .

However, this contradicts the previous lemma that says that  $\hat{\delta}(q_0, x0u) = \hat{\delta}(q_0, y1v)$ .

## Product Construction for NFA

**Definition:** Given 2 NFA  $N_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$  and  $N_2 = (Q_2, \Sigma, \delta_2, q_2, F_2)$  over the same alphabet  $\Sigma$ , we define the product  $N_1 \times N_2 = (Q, \Sigma, \delta, q_0, F)$  as follows:

- $Q = Q_1 \times Q_2$ ;
- $\delta((p_1, p_2), a) = \delta_1(p_1, a) \times \delta_2(p_2, a)$ ;
- $q_0 = (q_1, q_2)$ ;
- $F = F_1 \times F_2$ .

**Lemma:**  $(t_1, t_2) \in \hat{\delta}((p_1, p_2), x)$  iff  $t_1 \in \hat{\delta}_1(p_1, x)$  and  $t_2 \in \hat{\delta}_2(p_2, x)$ .

**Proof:** By induction on  $x$ .

**Proposition:**  $\mathcal{L}(N_1 \times N_2) = \mathcal{L}(N_1) \cap \mathcal{L}(N_2)$ .

## Complement for NFA

**OBS:** Given NFA  $N = (Q, \Sigma, \delta, q, F)$  and  $N' = (Q, \Sigma, \delta, q, Q - F)$  we do *not* have in general that  $\mathcal{L}(N') = \Sigma^* - \mathcal{L}(N)$ .

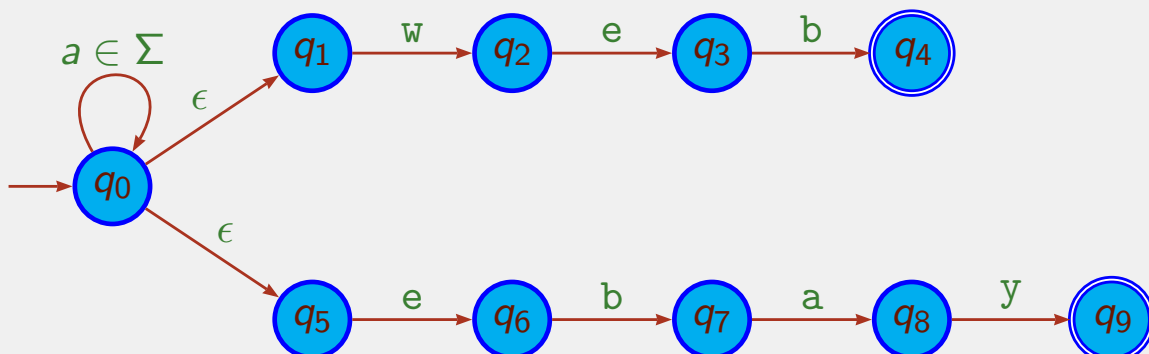
**Example:** Let  $\Sigma = \{a\}$  and  $N$  and  $N'$  as follows:



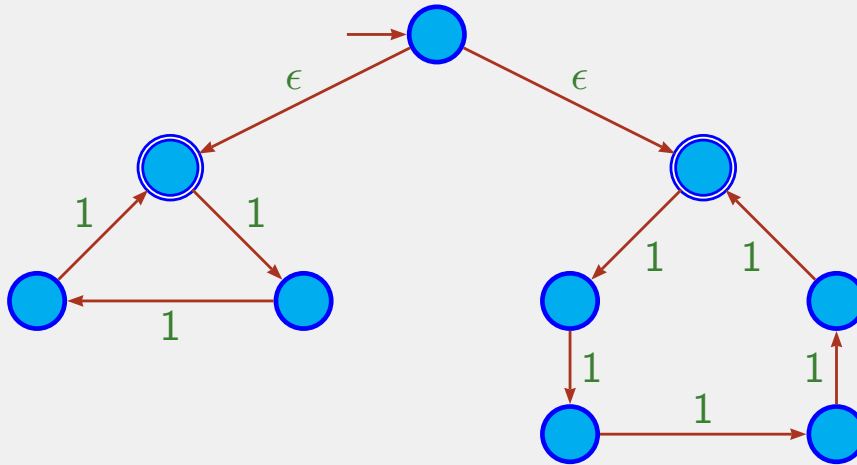
## NFA with $\epsilon$ -Transitions

Another useful extension of automata that does not add more power is the possibility to allow  $\epsilon$ -transitions, that is, transitions from one state to another *without* reading any input symbol.

**Example:** The following  $\epsilon$ -NFA searches for the keyword web and ebay:



**Example:** Let  $\Sigma = \{1\}$ .



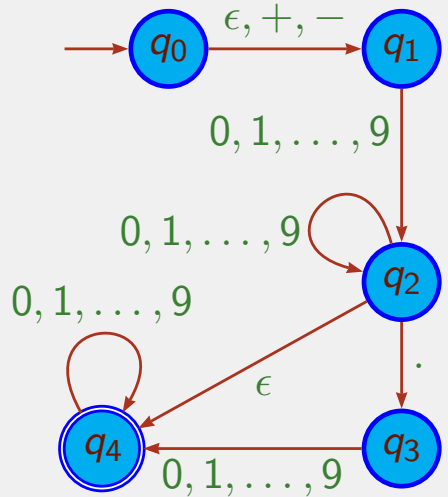
## NFA with $\epsilon$ -Transitions

**Definition:** A *NFA with  $\epsilon$ -transitions* ( $\epsilon$ -NFA) is a 5-tuple  $(Q, \Sigma, \delta, q_0, F)$  consisting of:

- 1 A finite set  $Q$  of *states*;
- 2 A finite set  $\Sigma$  of *symbols* (alphabet);
- 3 A “partial” transition function  $\delta : Q \times (\Sigma \cup \{\epsilon\}) \rightarrow \mathcal{P}ow(Q)$ ;
- 4 A *start state*  $q_0 \in Q$ ;
- 5 A set  $F \subseteq Q$  of *final or accepting states*.

## $\epsilon$ -NFA Accepting Decimal Numbers

**Exercise:** Define a NFA accepting number with an optional  $+/-$  symbol and an optional decimal part.



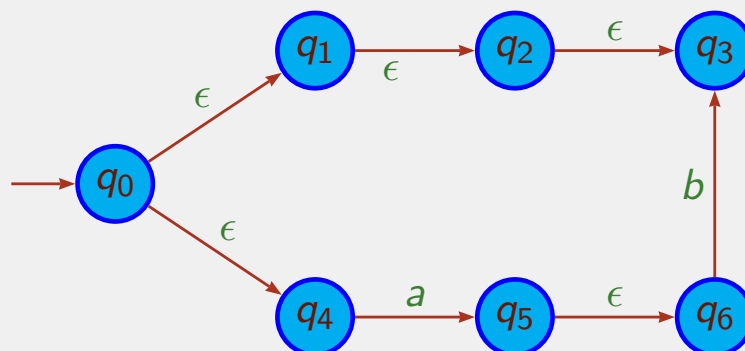
	$+,-$	$\cdot$	$0, 1, \dots, 9$	$\epsilon$
$\rightarrow q_0$	$\{q_1\}$	$\emptyset$	$\emptyset$	$\{q_1\}$
$q_1$	$\emptyset$	$\emptyset$	$\{q_2\}$	$\emptyset$
$q_2$	$\emptyset$	$\{q_3\}$	$\{q_2\}$	$\{q_4\}$
$q_3$	$\emptyset$	$\emptyset$	$\{q_4\}$	$\emptyset$
$*q_4$	$\emptyset$	$\emptyset$	$\{q_4\}$	$\emptyset$

The uses of  $\epsilon$ -transitions represent the *optional* symbol  $+/-$  and the *optional* decimal part.

## $\epsilon$ -Closures

Informally, the  $\epsilon$ -closure of a state  $q$  is the set of states we can reach by doing nothing or by *only* following paths labelled with  $\epsilon$ .

**Example:** For the automaton



the  $\epsilon$ -closure of  $q_0$  is  $\{q_0, q_1, q_2, q_3, q_4\}$ .

## $\epsilon$ -Closures

**Definition:** Formally, we define the  $\epsilon$ -closure of a set of states as follows:

- If  $q \in S$  then  $q \in \text{ECLOSE}(S)$ ;
- If  $q \in \text{ECLOSE}(S)$  and  $p \in \delta(q, \epsilon)$  then  $p \in \text{ECLOSE}(S)$ .

**Note:** Alternative formulation

$$\frac{q \in S}{q \in \text{ECLOSE}(S)} \qquad \frac{q \in \text{ECLOSE}(S) \quad p \in \delta(q, \epsilon)}{p \in \text{ECLOSE}(S)}$$

**Definition:** We say that  $S$  is  $\epsilon$ -closed iff  $S = \text{ECLOSE}(S)$ .

## Remarks: $\epsilon$ -Closures

- Intuitively,  $p \in \text{ECLOSE}(S)$  iff there exists  $q \in S$  and a sequence of  $\epsilon$ -transitions such that



- The  $\epsilon$ -closure of a single state  $q$  can be computed as  $\text{ECLOSE}(\{q\})$ ;
- $\text{ECLOSE}(\emptyset) = \emptyset$ ;
- $S$  is  $\epsilon$ -closed iff  $q \in S$  and  $p \in \delta(q, \epsilon)$  implies  $p \in S$ ;
- We can prove that  $\text{ECLOSE}(S)$  is the *smallest* subset of  $Q$  containing  $S$  which is  $\epsilon$ -closed.

**Exercise:** Implement the  $\epsilon$ -closure!

## Extending the Transition Function to Strings

**Definition:** Given an  $\epsilon$ -NFA  $E = (Q, \Sigma, \delta, q_0, F)$  we define

$$\hat{\delta} : Q \times \Sigma^* \rightarrow \mathcal{P}ow(Q)$$

$$\hat{\delta}(q, \epsilon) = \text{ECLOSE}(\{q\})$$

$$\hat{\delta}(q, ax) = \bigcup_{p \in \Delta(\text{ECLOSE}(\{q\}), a)} \hat{\delta}(p, x)$$

$$\text{where } \Delta(S, a) = \bigcup_{p \in S} \delta(p, a)$$

**Remark:** By definition,  $\hat{\delta}(q, a) = \text{ECLOSE}(\Delta(\text{ECLOSE}(\{q\}), a))$ .

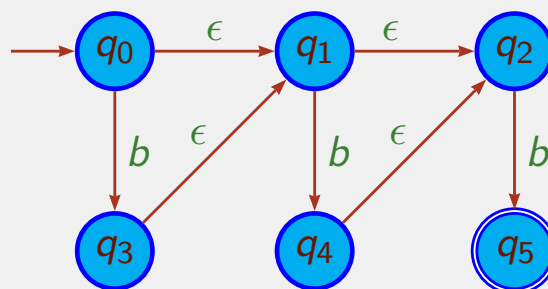
**Remark:** We can prove by induction on  $x$  that all sets  $\hat{\delta}(q, x)$  are  $\epsilon$ -closed.

This result uses that the union of  $\epsilon$ -closed sets is also a  $\epsilon$ -closed set.

## Language Accepted by a $\epsilon$ -NFA

**Definition:** The *language* accepted by the  $\epsilon$ -NFA  $(Q, \Sigma, \delta, q_0, F)$  is the set  $\mathcal{L} = \{x \in \Sigma^* \mid \hat{\delta}(q_0, x) \cap F \neq \emptyset\}$ .

**Example:** Let  $\Sigma = \{b\}$ .



The automaton accepts the language  $\{b, bb, bbb\}$ .

**Note:** Yet again, we could write a program that simulates a  $\epsilon$ -NFA and let the program tell us whether a certain string is accepted or not.

**Exercise:** Do it!



## Eliminating $\epsilon$ -Transitions

**Definition:** Given an  $\epsilon$ -NFA  $E = (Q_E, \Sigma, \delta_E, q_E, F_E)$  we define a DFA  $D = (Q_D, \Sigma, \delta_D, q_D, F_D)$  as follows:

- $Q_D = \{\text{ECLOSE}(S) \mid S \in \mathcal{P}ow(Q_E)\}$ ;
- $\delta_D(S, a) = \text{ECLOSE}(\Delta(S, a))$  with  $\Delta(S, a) = \cup_{p \in S} \delta(p, a)$ ;
- $q_D = \text{ECLOSE}(\{q_E\})$ ;
- $F_D = \{S \in Q_D \mid S \cap F_E \neq \emptyset\}$ .

**Note:** This construction is similar to the subset construction but now we need to  $\epsilon$ -close after each step.

**Exercise:** Implement this construction!

## Eliminating $\epsilon$ -Transitions

Let  $E$  be an  $\epsilon$ -NFA and  $D$  the corresponding DFA after eliminating  $\epsilon$ -transitions.

**Lemma:**  $\forall x \in \Sigma^*. \hat{\delta}_E(q_E, x) = \hat{\delta}_D(q_D, x)$ .

**Proof:** By induction on  $x$ .

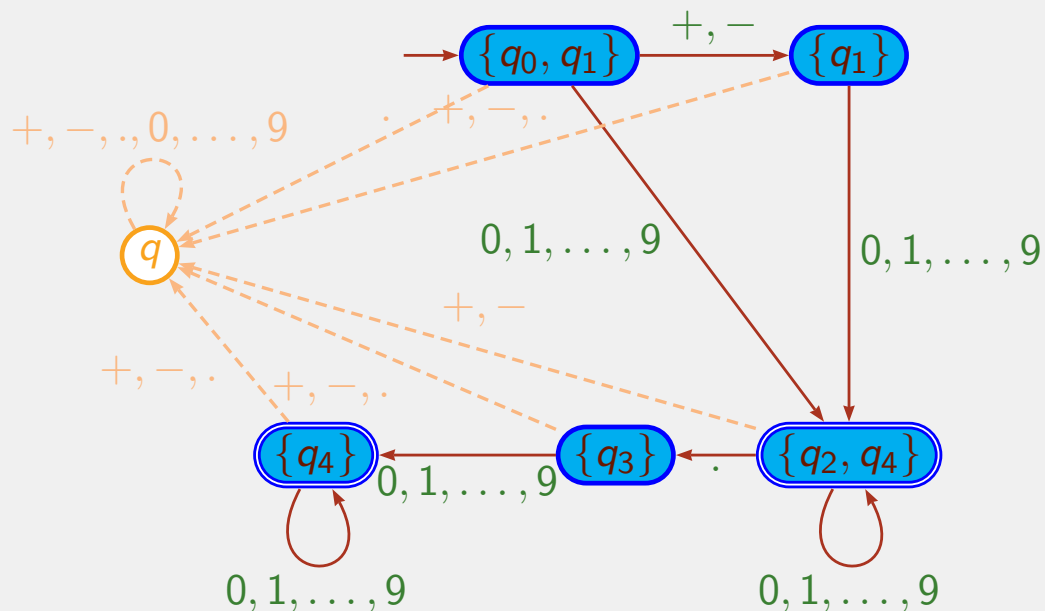
**Proposition:**  $\mathcal{L}(E) = \mathcal{L}(D)$ .

**Proof:**  $x \in \mathcal{L}(E)$  iff  $\hat{\delta}_E(q_E, x) \cap F_E \neq \emptyset$  iff  $\hat{\delta}_E(q_E, x) \in F_D$  iff (by previous lemma)  $\hat{\delta}_D(q_D, x) \in F_D$  iff  $x \in \mathcal{L}(D)$ .

## Example: Eliminating $\epsilon$ -Transitions

Let us eliminate the  $\epsilon$ -transitions in  $\epsilon$ -NFA that recognises numbers in slide 10.

We obtain the following DFA:



## Finite Automata and Regular Languages

We have shown that DFA, NFA and  $\epsilon$ -NFA are equivalent in the sense that we can transform one to the other.

Hence, a language is *regular* iff there exists a finite automaton (DFA, NFA or  $\epsilon$ -NFA) that accepts the language.

## Overview of Next Lecture (in HB3)

Sections 3.1, 3.4, 3.2.2:

- Regular expressions.
- Algebraic laws for regular expressions;
- Equivalence between FA and RE: from FA to RE.