eecs2001

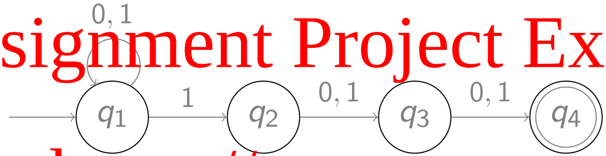
Introduction to the Theory of Computation

Lecture 11

Ruth Urner

Febuary 13, 2023

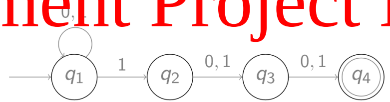
Example: the NFA N_2



For the machine N_2 we get

$$L(N_2) = \{\mathbf{w} \mid \mathbf{w} \text{ contains } 1 \text{ at the third position from the end}\}$$

Example: the NFA N_2



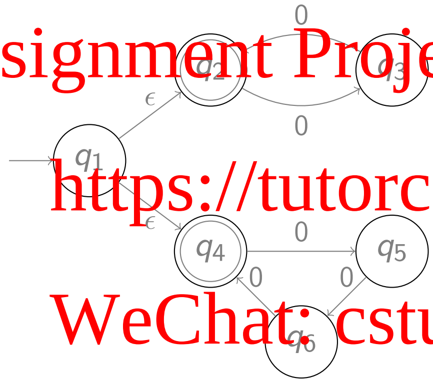
For the machine N_2 we get

$L(N_2) = \{ w \in \{0,1\}^* \mid w$ has a 1 at the third position from the end $\}$

$= \{ w = w_1 w_2 \ldots w_n \in \Sigma^* \mid w_{n-2} = 1 \}$

Example: the NFA N_3

Let's consider a unary alphabet $\Sigma = \{0\}$.

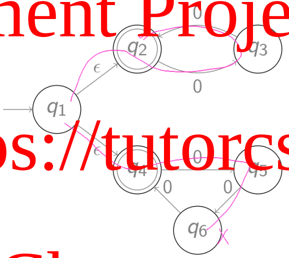


For the machine N_3 we get

$$L(N_3) = \{\mathbf{w} \mid \text{The number of 0 symbols in } \mathbf{w} \text{ is divisible by 2 or by 3}\}$$

Example: the NFA N_2

Let's consider a unary alphabet $\Sigma = \{0\}$.



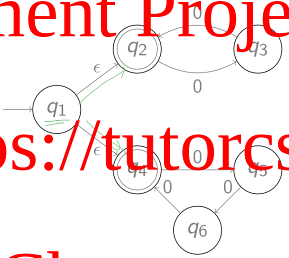
00000 X

00 ✓

$L(N_2) = \{ w \in \{0\}^* \mid$ the length of w is divisible by 2 or by 3 $\}$

Example:

Let's consider a unary alphabet $\Sigma = \{0\}$.



00000 X

00 ✓

is the same word as $\epsilon 00$

$L(N) = \{w \in \{0\}^* \mid$ the length of w is divisible by 2 or by 3$\}$

$= \{w \in \{0\}^* \mid$ length of w even$\} \cup \{w \in \{0\}^* \mid$ length of w divisible by 3$\}$

Definition

We call two finite state machines equivalent if they recognize the same language.

Theorem

For every NFA, there exists an equivalent DFA.

Corrolary

A language L is regular if and only if there exists an NFA that recognizes it.

Equival… f NFA … DFA

Definition

We call two finite state machines **equivalent** if they recognize the same language. = automata

Theorem

For every NFA, there exists an equivalent DFA.

For every NFA N, there exist a DFA M s.t. L(N) = L(M).

Corollary

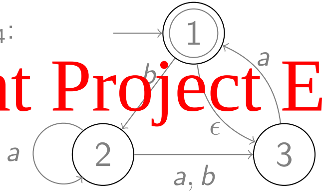
A language L is regular, if and only if there exists an NFA that recognizes L.

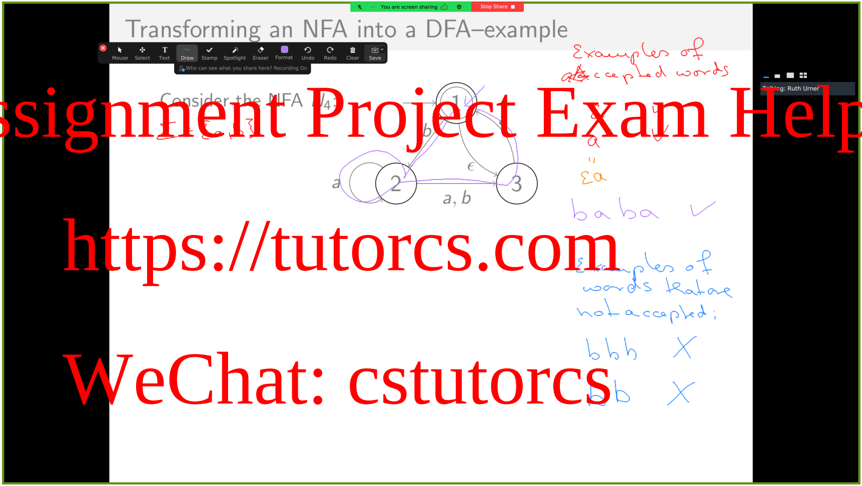
Transforming an NFA into a DFA–example

To prove the theorem from the previous slide, we need to show that for every NFA N, there exists a DFA M with $L(M) = L(N)$.

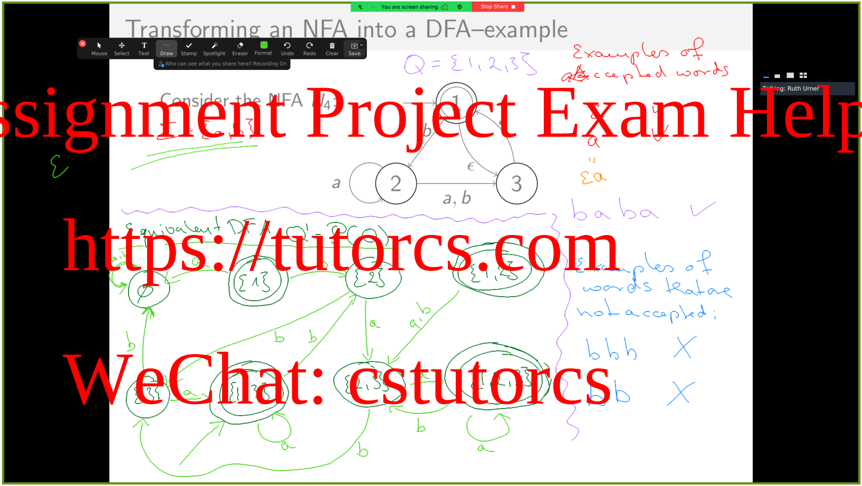
Before we see the general proof for this, we go through one concrete example. We transform NFA N_4 into an equivalent DFA.

Consider the NFA N_4:

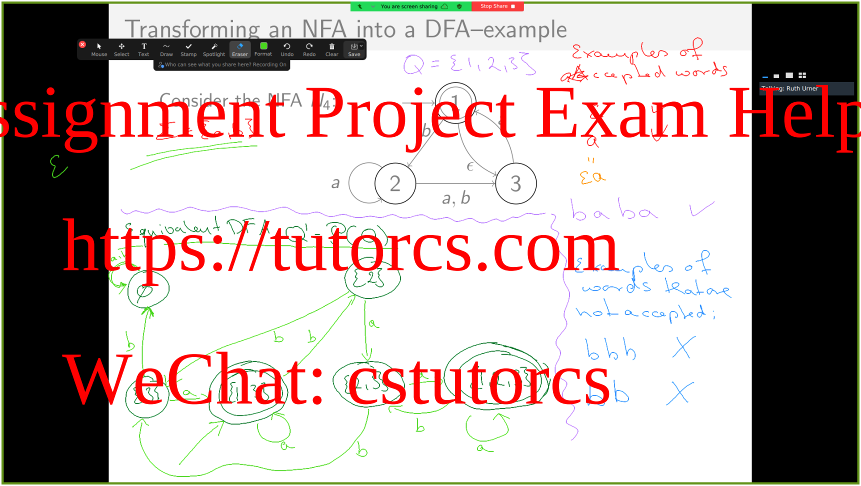




Transforming an NFA into a DFA–example
Consider the NFA N_4:
1
2
3
a
b
a
ϵ
a, b
Examples of accepted words
a
"
εa
baba
Examples of words that are not accepted;
bbb
bb



Transforming an NFA into a DFA–example
Consider the NFA N_4:
$Q = \{1, 2, 3\}$
Examples of accepted words
baba
Examples of words that are not accepted:
bbb
Equivalent DFA
Talking: Ruth Urner


Transforming an NFA into a DFA–example
Consider the NFA N_4:
Q = {1, 2, 3}
Examples of accepted words
baba ✓
Examples of words that are not accepted:
bbb ✗
Equivalent DFA
a, b
ε

Proof

Let $N = (Q, \Sigma, \delta, q_0, F)$ be some NFA.

We need to construct a DFA $M = (Q', \Sigma, \delta', q_0', F')$ with $L(M) = L(N)$.

Assuming that N does not contain any ϵ-transitions, we construct M as follows:

1. $Q' = \mathcal{P}(Q)$
2. For $R \in Q'$ and $a \in \Sigma$, we let

$$\delta'(R, a) = \{q \in Q \mid q \in \delta(r, a) \text{ for some } r \in R\}$$

3. $q_0' = \{q_0\}$
4. $F' = \{R \in Q' \mid \text{ there exists an } r \in F \text{ with } r \in R\}$.

This completes the construction for NFAs that don't contain ϵ-transitions.

General proof of Theorem

Proof

Let $N = (Q, \Sigma, \delta, q_0, F)$ be some NFA.

We need to construct a DFA $M = (Q', \Sigma, \delta', q_0', F')$ with $L(M) = L(N)$.

Assuming that N does not contain any ϵ-transitions, we construct M as follows:

$\in Q' = \mathcal{P}(Q)$

1. $Q' = \mathcal{P}(Q)$
2. For $R \in Q'$ and $a \in \Sigma$ we let

$$\delta'(R, a) = \{q \in Q \mid q \in \delta(r, a) \text{ for some } r \in R\}$$

3. $q_0' = \{q_0\}$
4. $F' = \{R \in Q' \mid \text{there exists an } r \in R \text{ with } r \in F\}$.

This completes the construction for NFAs that don't contain ϵ-transitions.

Proof continued

If N does contain ϵ-transitions, we extend the above construction as follows:

For a state $R \in Q'$, we let $E(R)$ denote the set of states that can be reached via 0 or more ϵ-transitions from some state $r \in R$ (in N).

Then we modify the transition function of M to be

$$\delta'(R, a) = \{q \in Q \mid q \in E(\delta(r, a)) \text{ for some } r \in R\}$$

and the start state of M to be

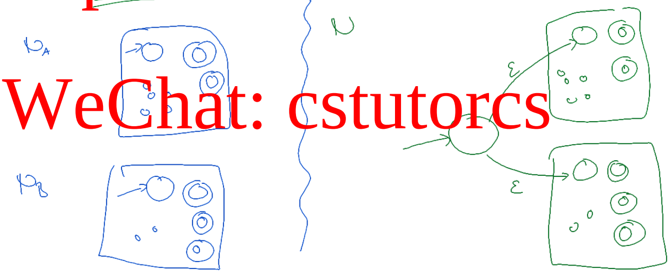
$$q_0' = E(q_0).$$

Theorem

Regular languages are closed under unions.

"Proof" by illustration

Given A and B and NFAs N_A and N_B with $L(N_A) = A$ and $L(N_B) = B$. We need to construct an NFA N that recognizes $A \cup B$.



Regular languages are closed under regular operations

Theorem

Regular languages are closed under unions.

Proof (formal version)

We need to show that if two languages A and B are regular, then so is $A \cup B$. If A and B are regular, then there exist NFAs $N_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$ and $N_2 = (Q_2, \Sigma, \delta_2, q_2, F_2)$ such that $L(N_1) = A$ and $L(N_2) = B$. We need to show that there exists an NFA M with $L(M) = A \cup B$. The picture on the previous slide illustrates a construction for $M = (Q, \Sigma, \delta, q_0, F)$. Formally, we set:

1. $Q = Q_1 \cup Q_2 \cup \{q_0\}$, where q_0 is a new state, that is $q_0 \notin Q_1 \cup Q_2$ and this state q_0 will then also be the start state of M.

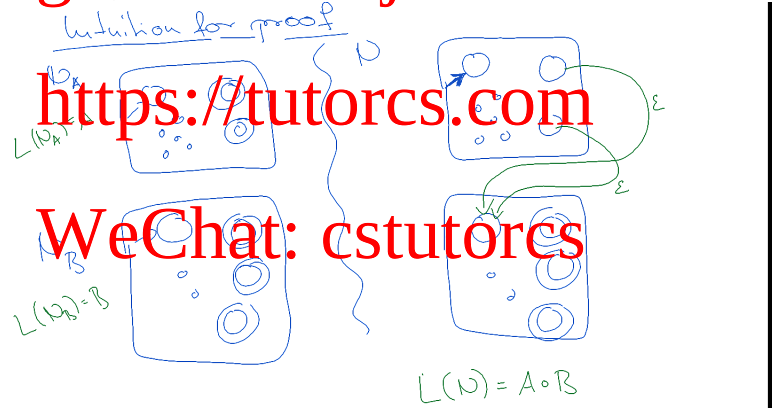
2. $\delta(q, a) = \begin{cases} \{q_1, q_2\} & \text{if} \quad q = q_0 \quad \text{and} \quad a = \epsilon \\ \delta_i(q, a) & \text{if} \quad q \in Q_i \\ \emptyset & \text{else} \end{cases}$

3. $F = F_1 \cup F_2$.

Theorem

Regular languages are closed under concatenations.

“Proof” by Illustration



Theorem

Regular languages are closed under concatenations.

Proof (formal version)

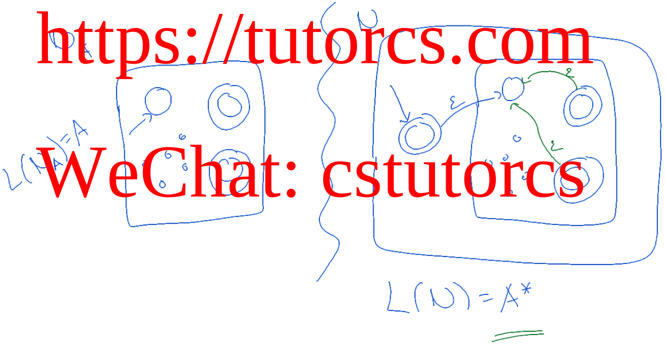
We need to show that if two languages A and B are regular, then so is $A \circ B$. If A and B are regular, then there exist NFAs $N_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$ and $N_2 = (Q_2, \Sigma, \delta_2, q_2, F_2)$ such that $L(N_1) = A$ and $L(N_2) = B$. We need to show that there exists an NFA M with $L(M) = A \circ B$. The picture on the previous slide illustrates a construction for $M = (Q, \Sigma, \delta, q_0, F)$. Formally, we set:

1. $Q = Q_1 \cup Q_2$
2. $q_0 = q_1$
3. $\delta(q, a) = \begin{cases} \delta_1(q, a) \cup \{q_2\} & \text{if} \quad q \in F_1 \text{ and } a = \epsilon \\ \delta_1(q, a) & \text{if} \quad q \in Q_1 \text{ and } (q \notin F_1 \text{ or } a \neq \epsilon) \\ \delta_2(q, a) & \text{if} \quad q \in Q_2 \end{cases}$
4. $F = F_2$.

Regular languages are closed under regular operations

Theorem
Regular languages are closed under the star-operation.

"Proof" by illustration



Theorem

Regular languages are closed under the star-operation.

Proof (formal version)

We need to show that if language A is regular, then so is A^*. If A is regular, then there exist an NFA $N_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$ such that $L(N_1) = A$. We need to show that there exists an NFA M with $L(M) = A^*$. The picture on the previous slide illustrates a construction for $M = (Q, \Sigma, \delta, q_0, F)$. We set:

1. $Q = Q_1 \cup \{q_0\}$, where q_0 is a new state, that is $q_0 \notin Q_1$ and this state q_0 will then also be the start state of M.
2. $\delta(q, a) = \begin{cases} \{q_1\} & \text{if} \quad q = q_0 \quad \text{and} \quad a = \epsilon \\ \delta_1(q, a) \cup \{q_1\} & \text{if} \quad q \in F_1 \text{ and } a = \epsilon \\ \delta_1(q, a) & \text{if} \quad q \in Q_1 \text{ and } (q \notin F_1 \text{ or } a \neq \epsilon) \end{cases}$
3. $F = F_1 \cup \{q_0\}$.

Regular Expressions

Reading: ITC Section 1.3

A regular expression is a compact way of defining a set of words. It is a sequence of symbols that represent a language over some alphabet Σ.

Inductive definition of sets – general pattern

An inductive definition of a set consists of

1. **A universe set** U
2. **A core set** $C \subseteq U$
3. **A finite set** $O = \{o_1, o_2, \ldots o_n\}$ **of operations from** $o_i : U^{r_i} \to U$ **for some arities** $r_i \in \mathbb{N}$

We define $\mathcal{I}(U, C, O)$ as the set of elements that we obtain by starting with the core set and putting all those elements of U into $\mathcal{I}(U, C, O)$ that one can reach by successively applying the operations in O.

Let Σ be some alphabet. We define the set $\mathcal{R}$ of regular expressions over Σ inductively by setting $\mathcal{R}_\Sigma = \mathcal{I}(U, C, O)$, where

1. The universe U is the set of all strings over $\Sigma \cup \{(,)\} \cup \circ, {}^*, \epsilon, \emptyset\}$
2. The core set C is the set of al symbols in Σ and $\epsilon, \emptyset$ and two additional symbols: $C = \Sigma \cup \{\epsilon, \emptyset\}$.

3. Three operations:
   - $o_\cup(R_1, R_2) = (R_1 \cup R_2)$,
   - $o_\circ(R_1, R_2) = (R_1 \circ R_2)$,
   - $o_*(R) = (R^*)$.

Regular expression–inductive definition

Let Σ be some alphabet. We define the set $\mathcal{R}$ of regular expressions over Σ inductively by setting $\mathcal{R}_\Sigma = \mathcal{I}(U, C, O)$, where

1. The universe U is the set of all strings over $\Sigma \cup \{(,), \cup, \circ, ^*, \epsilon, \emptyset\}$.
2. The core set C is the set of al symbols in Σ and $\epsilon, \emptyset$ and two additional symbols: $C = \Sigma \cup \{\epsilon, \emptyset\}$.
3. Three operations:
   - $o_\cup(R_1, R_2) = (R_1 \cup R_2)$,
   - $o_\circ(R_1, R_2) = (R_1 \circ R_2)$,
   - $o_*(R) = (R^*)$.

Examples: $\Sigma = \{0, 1\}$

0 ✓

$(1 \cup 0)$ ✓

$(1 \circ$ ✗

ε ✓

$(\varepsilon \circ 1)$ ✓

(1^*)

members of universe

$)(\circ\emptyset$

these two are not regular expressions

Exercise: Prove (by induction) that the number of "(" is always equal to the number of ")" in a regular expression

The language of a regular expression

Each regular expression $R \in \mathcal{R}_\Sigma$ over some alphabet Σ represents a language over Σ. We define the interpretation $L(R)$ of a regular expression R according to the inductive definition:

Members of the core-set:

- The expression a for $a \in \Sigma$ represents the language $\{a\}$, that is $L(a) = \{a\}$.
- The expression ϵ represents the language $\{\epsilon\}$, that is $L(\epsilon) = \{\epsilon\}$.
- The expression $\emptyset$ represents the language $\emptyset$,that is $L(\emptyset) = \emptyset$.

Result of operation: For regular expressions R_1, R_2 and R, we define:

- $L((R_1 \cup R_2)) \ = \ L(R_1) \cup L(R_2)$
- $L((R_1 \circ R_2)) \ = \ L(R_1) \circ L(R_2)$
- $L((R^*)) \ = \ L(R)^*$

We call $L(R)$ the language of R.

Assignment Project Exam Help

https://tutorcs.com

WeChat: cstutorcs

The language of a regular expression

Each regular expression $R \in \mathcal{R}_\Sigma$ over some alphabet Σ represents a language over Σ. We define the interpretation $L(R)$ of a regular expression R according to the inductive definition:

Members of the core-set:

- The expression a for $a \in \Sigma$ represents the language $\{a\}$, that is $L(a) = \{a\}$.
- The expression ϵ represents the language $\{\epsilon\}$, that is $L(\epsilon) = \{\epsilon\}$.
- The expression $\emptyset$ represents the language $\emptyset$, that is $L(\emptyset) = \emptyset$.

Result of operation: For regular expressions R_1, R_2 and R, we define:

- $L((R_1 \cup R_2)) = L(R_1) \cup L(R_2)$

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

$0 \in \mathcal{R}_\Sigma$

$L(0) = \{0\}$

$0, \epsilon \in \mathcal{R}_\Sigma$

$(0 \cup \epsilon) \in \mathcal{R}_\Sigma$

$L((0 \cup \epsilon))$
$= \{0, \epsilon\}$

$(0 \cup \emptyset) \in \mathcal{R}_\Sigma$

$L((0 \cup \emptyset))$
$= \{0\}$

Regular expression–additional notation

For convenience and readability, we use the following notational conventions:

1. For an alphabet Σ, we use Σ as a regular expression representing all words of length 1 over Σ. And then Σ^* is a regular expression for the set of all words over Σ.

2. We often omit brackets. The order of precedence then is: $*$, $\circ$, $\cup$.
3. The $\circ$-symbol is typically omitted: we use $R_1 R_2$ as shorthand for $R_1 \circ R_2$.
4. We let R^+ be shorthand for RR^*.

5. We let R^k be the k times repeated concatenation of R with itself: $R^k = R \circ R \circ R \circ \ldots \circ R$.

Regular expressions–examples

We let $\Sigma = \{0, 1\}$.

Now we can interpret the following regular expressions over Σ:

- 0^*10^*
- $\Sigma^*1\Sigma^*$

- $\Sigma^*001\Sigma^*$
- $1^*(01^+)^*$

- $(\Sigma\Sigma)^*$
- $(\Sigma\Sigma\Sigma)^*$

Regular expressions–examples

precedence: $* \circ \cup$

We let $\Sigma = \{0, 1\}$.

Now we can interpret the following regular expressions over Σ:

- $0^*10^* = ((0^*) \circ 1 \circ (0^*))$

$= \{w \in \Sigma^* \mid w \text{ contains exactly one } 1\}$

$1 \in L(0^*10^*)$

$0010 \in L(0^*1\,0^*)$

$10000 \in L(0^*10^*)$

$L((0^*))$ = {set of all words that have only letter 0}

ε is included

$L((0^*)) = \{\varepsilon, 0, 00, 000, \ldots\}$

$L(1) = \{1\}$

Regular expressions–examples

We let $\Sigma = \{0, 1\}$.

Now we can interpret the following regular expressions over Σ.

- 0^*10^*
- $\Sigma^*1\Sigma^* = (\Sigma^*)1(\Sigma^*) = (\Sigma^*) \circ 1 \circ (\Sigma^*)$

$L(\Sigma^*1\Sigma^*) = \{w \in \Sigma^* \mid w \text{ contains at least one } 1\}$

$1 \in L(\Sigma^*1\Sigma^*)$

$\underbrace{0101}_{\in\Sigma^*}1\underbrace{010}_{\in\Sigma^*} \in L(\Sigma^*1\Sigma^*)$

Regular expressions–examples

We let $\Sigma = \{0, 1\}$.

Now we can interpret the following regular expressions over Σ:

- 0^*10^*
- $\Sigma^*1\Sigma^*$
- $\Sigma^*001\Sigma^*$

$L(\Sigma^*001\Sigma^*) = \{w \in \Sigma^* \mid w$ contains 001 as a substring$\}$

Regular expressions–examples

We let $\Sigma = \{0, 1\}$.

Now we can interpret the following regular expressions over Σ:

- 0^*10^*
- $\Sigma^*1\Sigma^*$
- $\Sigma^*001\Sigma^*$
- $1^*(01^+)^*$

$L(1^*(0(1^+))^*) = \{w \in \Sigma^* \mid \text{every } 0 \text{ is followed by a } 1\}$

$1\,1\,1\,0\,1\,0\,1\,1\,1\,0\,1\,1$

$\epsilon\ 1^* \quad \epsilon\ 1^+ \quad 1^+ \quad 1^+$