eecs2001

Introduction to the Theory of Computation

Lecture 9

Ruth Urner

Febuary 6, 2023

Regular Languages

Reading: ITC Section 1.1

Definition

A language L over some alphabet Σ is called a regular language if there exists a finite automaton M such that $L = L(M)$, that is, if there exists a finite automaton that recognizes it.

Regular language

over some alphabet Σ

Definition

A language L is called a regular language if there exists a finite automaton M such that $L = L(M)$, that is, if there exists a finite automaton that recognizes it.

Examples of regular languages

$\{w \in \{0,1\}^* \mid w$ contains 001 as a substring$\}$

$\{w \in \{0,1\}^* \mid$ [illegible] $\}$

Regular operations on languages

Definition:

Let A and B be languages. Then we define the following operations that each form a new language:

- **Union:** $A \cup B = \{\mathbf{w} \mid \mathbf{w} \in A \text{ or } \mathbf{w} \in B\}$
- **Concatenation:** $A \circ B = \{\mathbf{wv} \mid \mathbf{w} \in A \text{ and } \mathbf{v} \in B\}$.
- **Star:** $A^* = \{\mathbf{w}_1\mathbf{w}_2 \ldots \mathbf{w}_k \mid k \geq 0 \text{ and } \mathbf{w}_i \in A\ \forall\, i \in \{0, 1, \ldots, k\}\}$

Regular o[perations]

In particular for an alphabet Σ, Σ^* is the set of all finite words over Σ.

$A \times B \ni (01, 110)$

Ex: $A = \{01, 001, 0001\}$
$B = \{11, 110\}$

not 0111

Definition:

Let A and B be languages. Then we define the following operations that each form a new language:

- **Union:** $A \cup B = \{\mathbf{w} \mid \mathbf{w} \in A \text{ or } \mathbf{w} \in B\}$.

Ex: $A \cup B = \{01, 001, 0001, 11, 110\}$

- **Concatenation:** $A \circ B = \{\mathbf{wv} \mid \mathbf{w} \in A \text{ and } \mathbf{v} \in B\}$.

Ex: $A \circ B = \{011, 0011, 00011, 0111, 00111, 000111, 01110, 001110, 0001110, 0110, 00110, 000110\}$

- **Star:**

$A^* = \{\mathbf{w}_1\mathbf{w}_2\ldots\mathbf{w}_k \mid k \geq 0 \text{ and } \mathbf{w}_i \in A\ \forall\, i \in \{0, 1, \ldots, k\}\}$

Ex: $A^* = \{\varepsilon, 01, 001, 0001, 0101, 01001, 010001, 001001, 0010001, 0001001, 010101, 0100101, \ldots\}$

infinite set of words...

Regular operations

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

$A = \{01, 001, 0001\}$

Ex: $B = \{1, 110\}$

$A \times B \ni (01, 110)$

Definition:
Let A and B be languages. Then we define the following operations that each form a new language:

- **Union:** $A \cup B = \{\mathbf{w} \mid \mathbf{w} \in A \text{ or } \mathbf{w} \in B\}$.

  Ex: $A \cup B = \{01, 001, 0001, 1, 110\}$

- **Concatenation:** $A \circ B = \{\mathbf{wv} \mid \mathbf{w} \in A \text{ and } \mathbf{v} \in B\}$.

  Ex: $A \circ B = \{011, 0011, 00011, 01110, 001110, 0001110, 0110, 00110, 000110\}$

- **Star:**

  $A^* = \{\mathbf{w}_1\mathbf{w}_2 \dots \mathbf{w}_k \mid k \geq 0 \text{ and } \mathbf{w}_i \in A \ \forall i \in \{0, 1, \dots, k\}\}$

  Ex: $A^* = \{\varepsilon, 01, 001, 0001, 0101, 01001, 010001, 001001, 0010001, 0001001, 010101, 0100101, \dots\}$

  infinite set of words...

In particular for an alphabet Σ, Σ^* is the set of all finite words over Σ.

$B \notin A^*$

$C = \{1, 11\}$

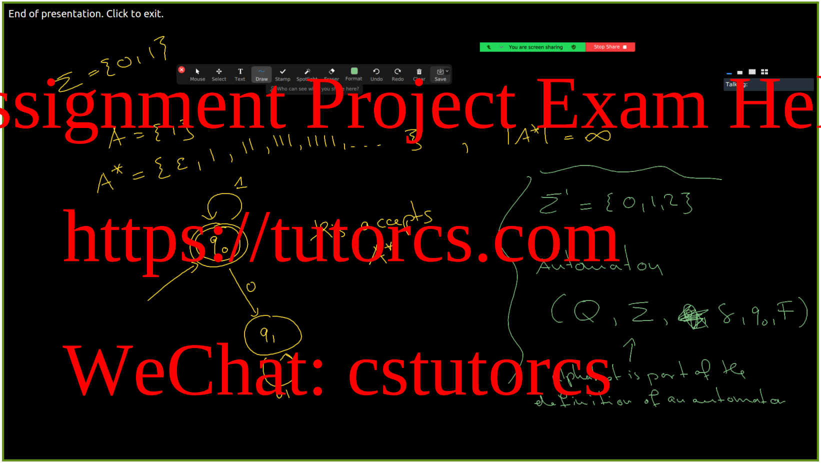
$B \subsetneq C^*$

$A \not\subseteq C^*$

$D = \{0, 1, 00, 11\}$

$B \subsetneq D^*$

$A \not\subseteq D^*$


End of presentation. Click to exit.
Σ = {0,1}
A = {1}
A* = {ε, 1, 11, 111, 1111, ...}
|A*| = ∞
Σ' = {0,1,2}
Automaton
(Q, Σ, δ, q0, F)
alphabet is part of the
definition of an automaton

Regular operations reproduce regular languages

Theorem

The set of all regular languages is closed under the three regular operations.

That is: if A and B are regular languages, then A^*, $A \cup B$ and $A \circ B$ are also regular languages.

Problem 1.4 from ITC

Here we design an automaton that recognizes the intersection of two languages for which we have simple automata recognizing them.

We will now use a similar "product" for the general construction of an automaton that recognizes the union of two regular languages.

Problem 1.4 from ITC

$\Sigma = \{a, b\}$

Here we design an automaton that recognizes the intersection of two languages for which we have simple automata recognizing them.

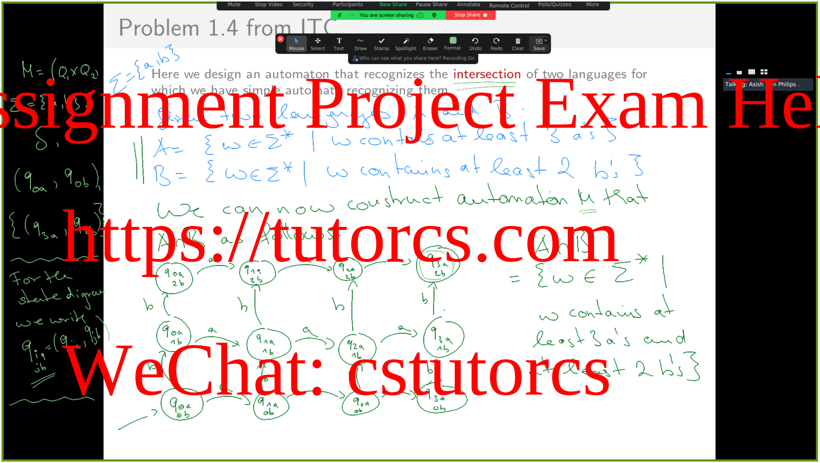
Given two languages A and B.

$A = \{ w \in \Sigma^* \mid w \text{ contains at least 3 a's} \}$

$B = \{ w \in \Sigma^* \mid w \text{ contains at least 2 b's} \}$

We define automaton M_1 & with $L(M_1) = A$

and automaton M_2 with $L(M_2) = B$


Problem 1.4 from ITC
Here we design an automaton that recognizes the intersection of two languages for which we have simple automata recognizing them.
Given two languages A and B
A = {w ∈ Σ* | w contains at least 3 a's}
B = {w ∈ Σ* | w contains at least 2 b's}
We can now construct automaton M that recognizes A∩B as follows
A∩B = {w ∈ Σ* | w contains at least 3 a's and at least 2 b's}
For the state diagram we write

Modification of ~~Problem~~ Pro Problem 1.4 from ITC

$\Sigma = \{a, b\}$

Here we design an automaton that recognizes the ~~intersection~~ union of two languages for which we have simple automata recognizing them.

Given two languages A and B

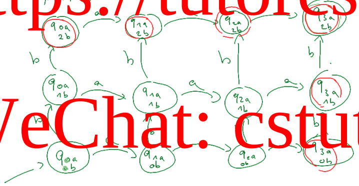
$A = \{ w \in \Sigma^* \mid w$ contains at least 3 a's $\}$

$B = \{ w \in \Sigma^* \mid w$ contains at least 2 b's $\}$

We can now construct automaton M that recognizes $A \cup B$ as follows

$M = (Q_1 \times Q_2,$
$\Sigma = \{a, b\},$
$\delta,$
$(q_{0a}, q_{0b}),$
$\{ \ldots \})$

For the state diagram we write $q_{ij} = (q_{ia}, q_{jb})$



$A \cup B = \{ w \in \Sigma^* \mid$ w contains at least 3 a's or at least 2 b's $\}$

← 6 accept states!

Regular languages closed under unions

We start by proving that if A and B are regular languages over some alphabet Σ, then so is $A \cup B$.

Proof:

To prove that $A \cup B$ is regular, we need to show that there exists an automaton

$$M = (Q, \Sigma, \delta, q_0, F)$$

with

$$L(M) = A \cup B$$

Since A and B are regular, we know there exist automata $M_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$ and $M_2 = (Q_2, \Sigma, \delta_2, q_2, F_2)$ with

$$L(M_1) = A \text{ and } L(M_2) = B$$

Regular languages closed under unions

We start by proving that if A and B are regular languages over some alphabet Σ, then so is $A \cup B$.

general Σ

Given M_1 and M_2 the goal is to construct an automaton that recognizes $A \cup B$ using the components of M_1 and M_2

Goal for the proof

To show that $A \cup B$ is regular, we need show that there exists an automaton that recognizes $A \cup B$.

We know that A and B are regular languages, and therefore there exist automata $M_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$ and $M_2(Q_2, \Sigma, \delta_2, q_2, F_2)$ such that $A = L(M_1)$ and $B = L(M_2)$.

Proof continued.

We now define the required automaton M as follows:

1. $Q = Q_1 \times Q_2$
2. We use the same alphabet Σ
3. $\delta((s_1, s_2), a) = (\delta_1(s_1, a), \delta_2(s_2, a))$
4. $q_0 = (q_1, q_2)$
5. $F = \{(s_1, s_2) \in Q \mid s_1 \in F_1 \text{ or } s_2 \in F_2\}$

Regular languages closed

$M_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$

$M_2 = (Q_2, \Sigma, \delta_2, q_2, F_2)$

$L(M_1) = A$

$L(M_2) = B$

Proof continued:

We now define the required automaton M as follows:

1. $Q = Q_1 \times Q_2$
2. We use the same alphabet Σ
3. $\delta((s_1, s_2), a) = (\delta_1(s_1, a), \delta_2(s_2, a))$
4. $q_0 = (q_1, q_2)$
5. $F = \{(s_1, s_2) \in Q \mid s_1 \in F_1 \text{ or } s_2 \in F_2\}$

Regular languages closed under unions

Proof continued:

We now define the required automaton M as follows:

1. $Q = \underbrace{Q_1 \times Q_2}$
2. We use the same alphabet Σ
3. $\delta((\overset{\in Q_1}{s_1}, \overset{\in Q_2}{s_2}), \underset{\in \Sigma}{a}) = (\delta_1(s_1, a), \delta_2(s_2, a))$
4. $q_0 = (q_1, q_2)$
5. $F = \{(s_1, s_2) \in Q \mid s_1 \in F_1 \text{ or } s_2 \in F_2\}$

Non-deterministic finite automata

Reading: ITC Section 1.2

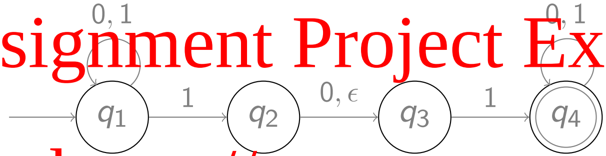
DFA versus NFAs

To show that regular languages are also closed under the concatenation and star operations, it is convenient to employ the concept of a Non-deterministic Finite Automaton (NFA for short).

In the automata we have seen so far, every state has exactly one outgoing edge for every symbol. That is, for every state, the “reaction” to any input symbol is uniquely determined. Such an automaton is also called a Deterministic Finite Automaton (DFA for short).

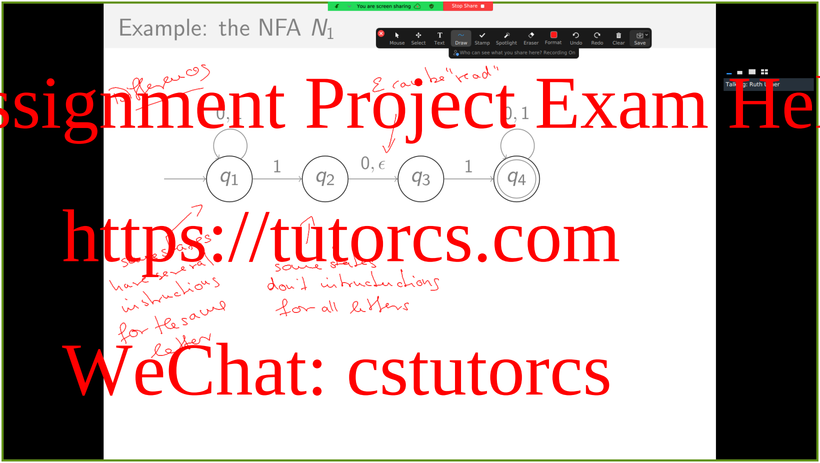
We will generalize this to also allow non-unique (think randomized or parallelized) computations. NFAs are a model for such computations.

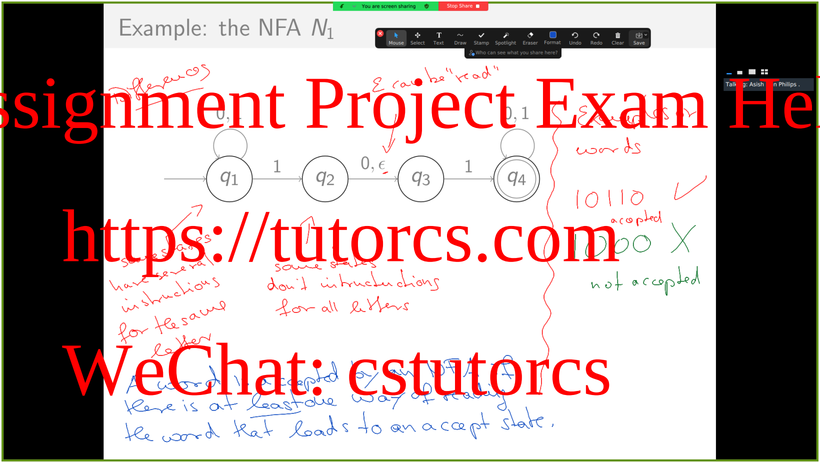
Example: the NFA N_1



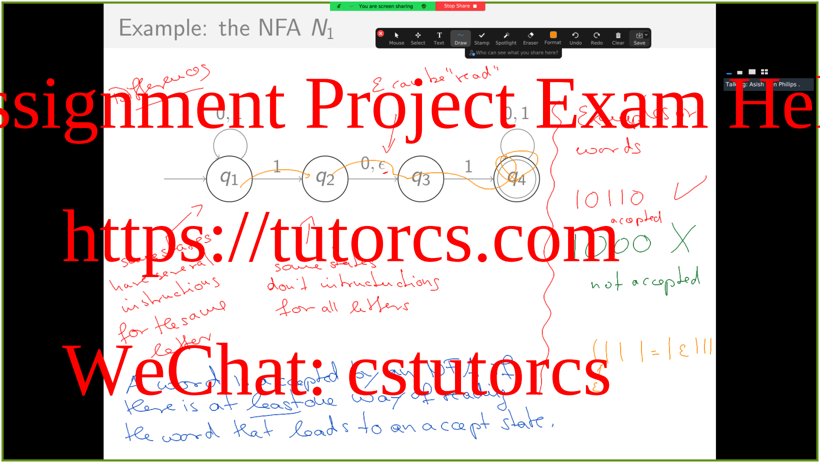
For the machine N_1 we get

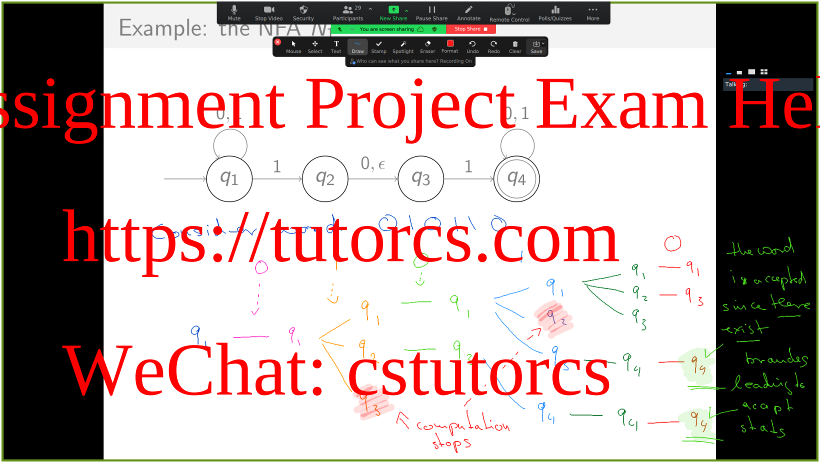
$$L(N_1) = \{\mathbf{w} \mid \mathbf{w} \text{ contains 11 or 101 as a substring}\}$$


You are screen sharing
Stop Share
Example: the NFA N_1
Mouse Select Text Draw Stamp Spotlight Eraser Format Undo Redo Clear Save
Who can see what you share here? Recording On
Differences
ε can be "read"
0, 1
1
0, ε
1
0, 1
q_1
q_2
q_3
q_4
some states have several instructions for the same letter
some states don't instructions for all letters



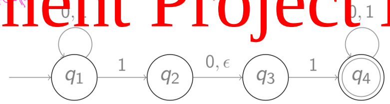
Example: the NFA N_1
q_1 q_2 q_3 q_4
0, 1
1
0, ε
1
0, 1
Differences
ε can be "read"
Examples of words
10110 ✓ accepted
1000 X not accepted
some states have several instructions for the same letter
some states don't instructions for all letters
A word is accepted by an NFA if there is at least one way of reading the word that leads to an accept state.


Example: the NFA N_1
differences
ε can be "read"
0, 1
0, 1
q_1
1
q_2
0, ε
q_3
1
q_4
words
10110
accepted
not accepted
some states
have several
instructions
for the same
letter
some states
don't instructions
for all letters
there is at least one way of reading
the word that leads to an accept state.


Example: the NFA
0, 1
0, 1
q1
1
q2
0, ϵ
q3
1
q4
the word is accepted since there exist branches leading to accept states
computation stops

Example: the NFA N_1

10001
not accepted



Reminder: For alphabet Σ, the set of all words over Σ is denoted by Σ^*.

For the machine N_1 we get

$L(N_1) = \{ w \in \{0,1\}^* \mid w \text{ contains } 101 \text{ or } 11 \text{ as a substring} \}$

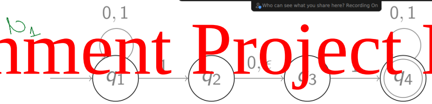
For an alphabet Σ, we define: $\Sigma_\epsilon = \Sigma \cup \{\epsilon\}$.

Definition

A non-deterministic finite automaton is a 5-tuple $(Q, \Sigma, \delta, q_0, F)$, where

1. Q is a finite set called the set of states,
2. Σ is a finite set called the alphabet,
3. $\delta : Q \times \Sigma_\epsilon \to \mathcal{P}(Q)$ is the transition function,
4. $q_0 \in Q$ is the start state, and
5. $F \subseteq Q$ is the set of accept states.

Formal description of the NFA N_1



Formal description of $N_1 = (Q, \Sigma, \delta, q_0, F)$

$Q = \{q_1, q_2, q_3, q_4\}$, $\Sigma = \{0, 1\}$, $q_0 = q_1$, $F = \{q_4\}$.

$\Sigma_\varepsilon = \{0, 1, \varepsilon\}$

δ	0	1	ε
q_1	$\{q_1\}$	$\{q_1, q_2\}$	$\emptyset$
q_2	$\{q_3\}$	$\emptyset$	$\{q_3\}$
q_3	$\emptyset$	$\{q_4\}$	$\emptyset$
q_4	$\{q_4\}$	$\{q_4\}$	$\emptyset$

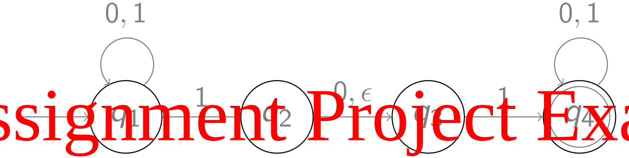
$|\varepsilon|$

$||$

$|\varepsilon\,\varepsilon|$

$|\varepsilon\varepsilon\varepsilon|\varepsilon$

these are the same word $||$



For the machine N_1 we get

$$L(N_1) = \{\mathbf{w} \mid \mathbf{w} \text{ contains 11 or 101 as a substring}\}$$

Formal definition of acceptance for NFAs

For a string $\mathbf{w} = w_1 w_2 \ldots w_n$ where $w_i \in \Sigma$ for some alphabet Σ, we let strings $\mathbf{z}_0 w_1 \mathbf{z}_1 w_2 \mathbf{z}_3 \ldots \mathbf{z}_{n-1} w_n \mathbf{z}_n$ over Σ_ϵ where each $\mathbf{z}_i$ is a sequence of 0 or more symbols ϵ represent the same word $\mathbf{w}$ as a string over Σ.

Definition

Let $N = (Q, \Sigma, \delta, q_0, F)$ a non-deterministic finite automaton (NFA) and let $\mathbf{w}$ be a string over Σ. We say that N accepts $\mathbf{w}$ if we can write $\mathbf{w} = y_1 y_2 \ldots y_m$ with $y_i \in \Sigma_\epsilon$ and there exists a sequence $s_0 s_1 s_2 \ldots s_m$ of states such that

1. $s_0 = q_0$,
2. $s_{i+1} \in \delta(s_i, y_{i+1})$ for $i = 0, 1, \ldots, m$, and
3. $s_m \in F$.

We then define the language $L(N)$ recognized by NFA N in the same way as for DFAs, namely as the set of all words that are accepted by N.

Formal definition of acceptance for NFAs

|| = ω

|ε|ε

ε|ε|

all the same word will

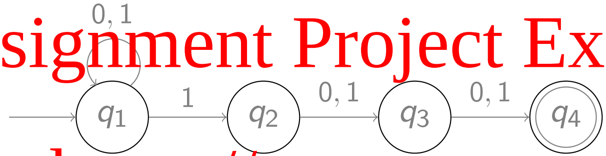
For a string $\mathbf{w} = w_1 w_2 \ldots w_n$ where $w_i \in \Sigma$ for some alphabet Σ, we let strings $\mathbf{z}_0 w_1 \mathbf{z}_1 w_2 \mathbf{z}_2 \ldots \mathbf{z}_{n-1} w_n \mathbf{z}_n$ over Σ_ϵ where each $\mathbf{z}_i$ is a sequence of 0 or more symbols ϵ represent the same word $\mathbf{w}$ as a string over Σ.

Definition

Let $N = (Q, \Sigma, \delta, q_0, F)$ a non-deterministic finite automaton (NFA) and let $\mathbf{w}$ be a string over Σ. We say that N accepts $\mathbf{w}$ if we can write $\mathbf{w} = y_1 y_2 \ldots y_m$ with $y_i \in \Sigma_\epsilon$ and there exist a sequence $s_0 s_1 s_2 \ldots s_m$ of states such that

1. $s_0 = q_0$,
2. $s_{i+1} \in \delta(s_i, y_{i+1})$ for $i = 0, 1, \ldots, m$, and
3. $s_m \in F$.

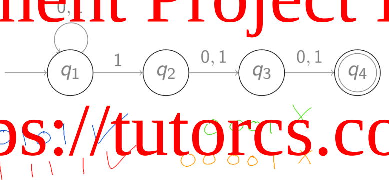
Example: the NFA N_2



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 $\}$

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