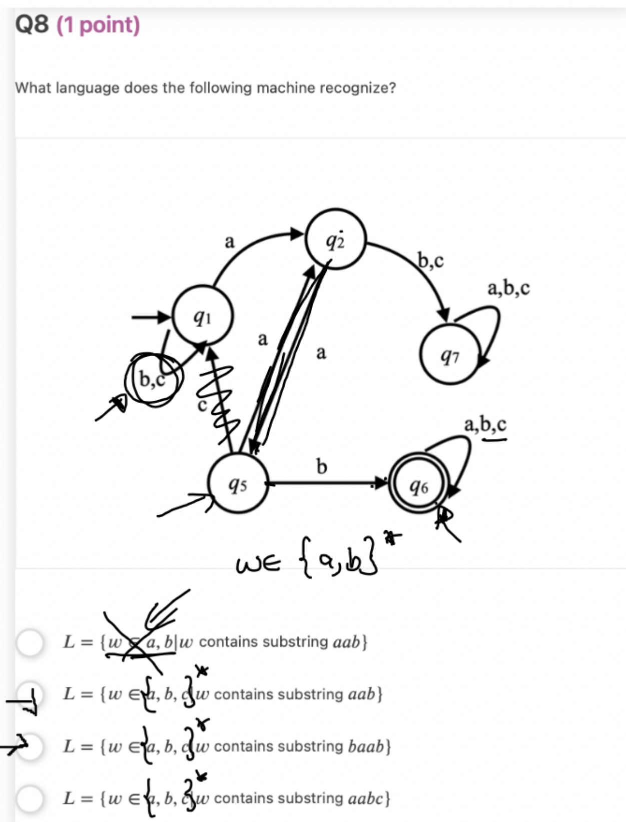
CSC 320 - Lecture 03

#computation #DFA #regular #languages #closure

Quiz 01 - Question 08

Question 08



Note. The first error in the question was that the set notation was erroneous. However, the rest of the question was defined correctly.

- $L = \{w \in \{a, b\}^* | w \text{ contains substring } aab\}$
- $L = \{w \in \{a, b, c\}^* | w \text{ contains substring } aab\}$
- $L = \{w \in \{a, b, c\}^* | w \text{ contains substring } baab\}$
- $L = \{w \in \{a, b, c\}^* | w \text{ contains substring } aabc\}$

And none of the options were correct as there are answers that contain those substrings that do not finish in the accept state. But the first option wasn't correct as the set needs to contain all options in the language.

Ex. The string `aaab` ends up in q_7 but contains the substring `aab`.

Formal Definition of Computation

Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA and let $w = w_1 w_2 \ldots w_n$ be a string over Σ. Then M **accepts** w if there is a sequence of states $r_0 r_1 r_2 \ldots r_n$ in Q such that...

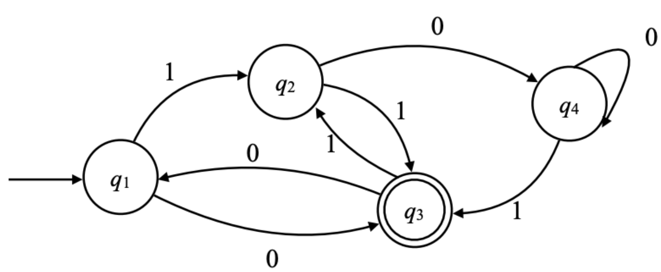
- $r_0 = q_0$
- $\delta(r_i, w_{i+1}) = r_{i+1}$
- $r_n \in F$

Then M recognizes language L if $L = L(M) = \{w \in \Sigma^* | M \text{ accepts } w\}$.

Given the sequence of state of M when asked for evidence tells us that it is recognized.

A language L is called a **regular language** if there exists a ~~deterministic~~ finite automaton that recognizes L (it doesn't need to be deterministic).

Example



Sequence of States for $W = 001101$

The sequence is $q_1 q_3 q_1 q_2 q_3 q_1 q_2$. And since q_2 is not a final state w is not accepted.

Sequence of States for $W = 0011011$

The sequence is $q_1 q_3 q_1 q_2 q_3 q_1 q_2 q_3$. And since q_3 is a final state w is accepted.

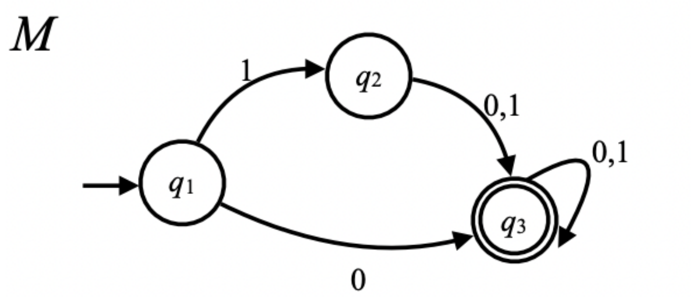
Regular Languages

The language of an automaton M, that is the language that automaton M recognizes, $L(M)$**, is exactly the set of all strings that** M **accepts** (NO MORE NO LESS)!

A language L is called a **regular language** if there exists a deterministic finite automaton that recognizes L (we note that there are languages that are not recognized by a DFA).

The set of languages that are recognized by the set of DFAs, the **regular languages**, is a **subset of the set of all languages**.

Language of DFA M



Let $L(M) = \{w | w \in \Sigma^*$ of length at least 1 where: if w starts with symbol 1 then w is of length at least 2$\}$.

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

$(\{q_1, q_2, q_3, q_4\}, \{0, 1\}, \delta, q_1, \{q_3\})$ with $\delta : Q \times \Sigma \to Q$ defined by...

Transition Table:

δ	0	1
q_1	q_3	q_2
q_2	q_3	q_3
q_3	q_3	q_3

$L(M)$ is a regular language.

Closure Properties for Sets

A **set** is **closed** under some operation if applying that operation to elements of the set returns another element of the set.

If a certain operation applied to **any language** in a certain **class of languages** (eg., the class of regular languages) produces a result that is also in that same class, then the language class (eg., the class of regular languages) is **closed under this operation**.

Closure Properties (1)

- Definition: Closed
- Operations: Union, Intersection, Concatenation

We will show that regular languages are closed under union, intersection, and concatenation.

We will show: given a regular languages L_1 and L_2 then...

- $L_1 \cup L_2$ is a regular language
- $L_1 \cap L_2$ is a regular language
- $L_1 L_2$ is a regular language

The Class of Regular Languages is Closed Under the Union Operation

Theorem. If L_1 and L_2 are regular languages over alphabet Σ then $L_1 \cup L_2$ is a regular language. (In other words: The class of regular languages is closed under the union operation).

Proof. Since L_1 and L_2 are regular languages there exist deterministic finite automata M_1 and M_2 with $L_1 = L(M_1)$ and $L_2 = L(M_2)$.

Idea. Construct DFA M that accepts exactly the strings accepted by M_1 and the strings accepted by M_2. (M accepts every string accepted by M_1 **AND** every string accepted by M_2 and nothing else).

We do this by simulation both machines concurrently, and accepting if (at least) one of them accepts.

Let $M_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$ (where $Q_1 = \{t_1, t_2, \ldots, t_n\}$) and $M_2 = (Q_2, \Sigma, \delta_2, q_2, F_2)$ (where $Q_2 = \{s_1, s_2, \ldots, s_n\}$) be DFAs. For each state $r_1 \in Q_1$ and every state $r_2 \in Q_2$ we create a state of M. Thus, we construct DFA $M = (Q, \Sigma, \delta, q_0, F)$ as follows...

- $Q = \{(r_1, r_2) | r_1 \in Q_1, r_2 \in Q_2\}$. (Start state of M: $q_0 = (q_1, q_2)$).
- For each $(r_1, r_2) \in Q$ and each $a \in \Sigma$: $\delta((r_1, r_2), a) = (\delta_1(r_1, a), \delta_2(r_2, a))$
- $q_0 = (q_1, q_2)$
- $F = \{(r_1, r_2) | r_1 \in F_1 \text{ OR } r_2 \in F_2\}$.

M recognizes $L_1 \cup L_2$

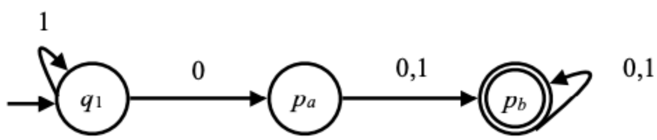
Example

Let:

- $\Sigma = \{0,1\}$
- $L_1 =$ is the set of all strings that, after a possible prefix of 1s, consist of at least one 0 followed by at least one symbol.
- $L_2 =$ is the set of all strings of length at exactly 1.

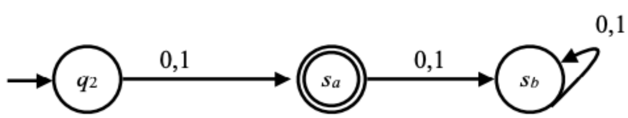
$L * M_1 = L_1$

M_1



$L * M_2 = L_2$

M_2



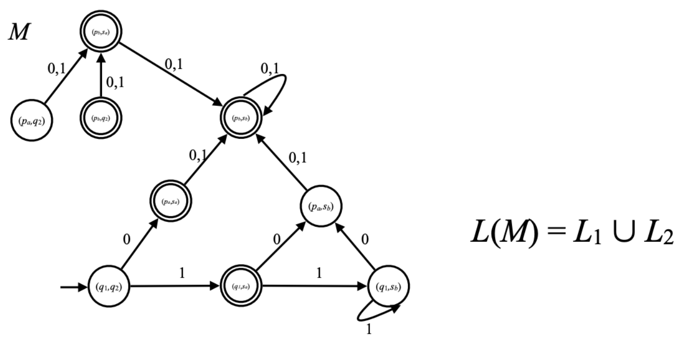
Q	**0**	**1**
(q_1, q_2)	(p_a, s_a)	(q_1, s_a)
(q_1, s_a)	(p_a, s_b)	(q_1, s_b)
(q_1, s_b)	(p_a, s_b)	(q_1, s_b)
(p_a, q_2)	(p_b, s_a)	(p_b, s_a)
(p_a, s_a)	(p_b, s_b)	(p_b, s_b)
(p_a, s_b)	(p_b, s_b)	(p_b, s_b)
(p_b, q_2)	(p_b, s_a)	(p_b, s_a)
(p_b, s_a)	(p_b, s_b)	(p_b, s_b)
(p_b, s_b)	(p_b, s_b)	(p_b, s_b)

Start: (q_1, q_2)

Q	**0**	**1**

Q	**0**	**1**
	(p_a, s_a)	(q_1, s_a)
(q_1, s_a)		
	(p_b, s_a)	(p_b, s_a)
(p_a, s_a)	(p_b, s_b)	(p_b, s_b)
	(p_b, s_b)	(p_b, s_b)
(p_b, q_2)	(p_b, s_a)	(p_b, s_a)
(p_b, s_a)	(p_b, s_b)	(p_b, s_b)
(p_b, s_b)	(p_b, s_b)	(p_b, s_b)

$M = L$



Previous Lecture

[Lecture02]

Next Lecture

[Lecture04]