03 Feb 2025 - Theory of Computation - Week 05

→ Solve exercises

<u>deciding emptiness</u>

Given an automaton M, is $L(M) = \phi$?

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

if $F = \phi$ then $L(M) = \phi$

* no path from start state to final state.

→ see FSM as a directed graph

→ run BFS from start node, if atleast one $f \in F$ is in the path, $L(M) \neq \phi$

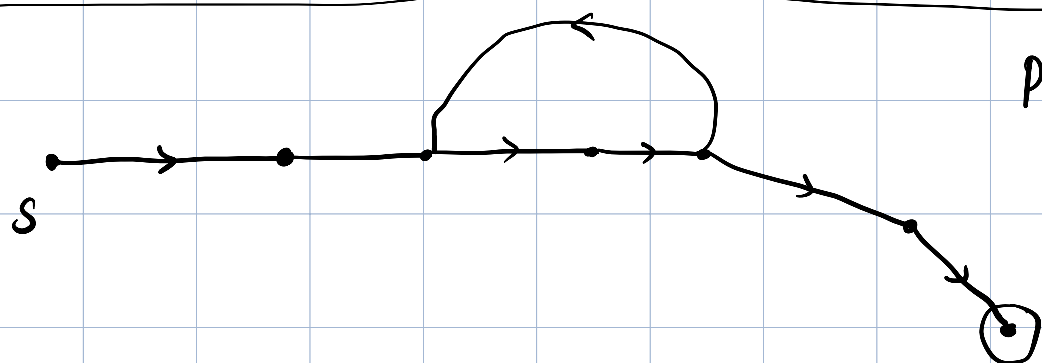
→ Given M, is the set $L(M)$ finite or infinite?

Fact: $L(M) \neq \phi$ iff $\exists$ a string x of length at most $k-1$ such that $x \in L(M)$, where k is # of states in M

(exam problem)

$\rightleftarrows$

Pumping lemma



$|x| \geq k$; $x \in L(M)$

$\Rightarrow L(M)$ is infinite (by Pumping Lemma)

Fact 2: $L(M)$ is infinite iff there exists a string of length $k \leq |x| < 2k$ s.t. $x \in L(M)$

$(2) \Rightarrow (1)$: Pumping lemma (pegionhole principle)

$(1) \Rightarrow (2)$

If $L(M)$ is infinite $\rightsquigarrow$ there cannot be an upper bound on the length of strings.

Choose $x \in L(M)$ s.t. it has minimal length $\geq k$

$|x| = j$

$\underbrace{j \geq k}$

no string in b/w

If $|x| < 2k$: we are done.

If not : $|x| \geq 2k$

$x = u\,v\,w$ $\qquad |v| \leq k$

$\forall\ i \geq 0$

$u(v)^i w \in L(M)$

For $i = 0$

$$k \underbrace{\leq}_{|v| = k} \underbrace{|u\,w|}_{\text{will be recognized}} \underbrace{<}_{v \neq \varepsilon} |x|$$

$\Rightarrow\Leftarrow$

Contradiction to x being the shortest string of length $\geq k$ that is recognized by M.

This gives us an algorithm to check if an automaton accepts a language of infinite length.

might take long, but will finish in finite time

→ not efficient, but works

??

How to check if two machines recognize the same language? Use above techniques

Efficient automata

Relations: Recap

Fix X : $R \subseteq X \times X$

$$(a, b) \in R \equiv (a \; R \; b) \equiv (a \sim b) \equiv (a \equiv b)$$

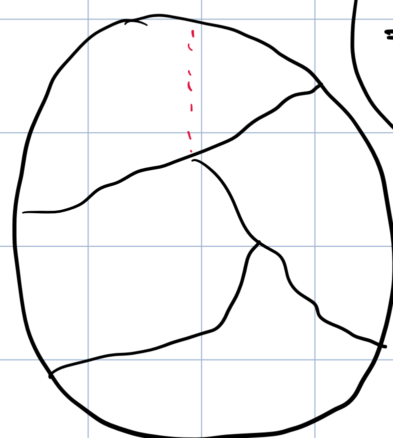
→ reflexive : $(a, a) \in R \quad \forall a \in X$

→ symmetric : $(a, b) \in R \Rightarrow (b, a) \in R$

→ transitive : $(a, b) \in R$ and $(b, c) \in R \Rightarrow (a, c) \in R$

→ equivalence relation : reflexive, symmetric, transitive.

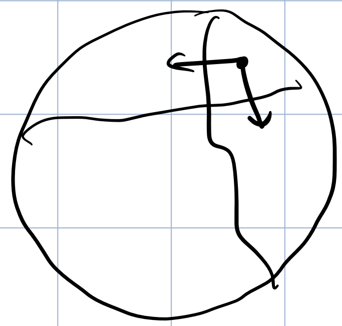
$$[a] = \{ x \in X \mid (x, a) \in R \}$$



partition
⇒ totally overlap
or totally disjoint

for $b \neq c$, either $[b] = [c]$ or $[b] \cap [c] = \phi$

- proof by contradiction



04 Feb 2025

Assume that $[b] \neq [c]$ ◦ proof

Suppose $\exists\ z \in [b] \cap [c]$

Given a DFA $M = (Q, \Sigma, \delta, s, F)$, define the following relation R_M on Σ^*

for any $(x, y) \in \Sigma^* \times \Sigma^*$

$$(x, y) \in R_M \iff \hat{\delta}(s, x) = \hat{\delta}(s, y)$$

1) reflexive ✓
2) symmetric ✓
3) transitive ✓

$\Big\}$ R_M is an equivalence relation

→ No. of equivalence classes = no. of states in M.

Note: M is such that every state in M is reachable from s.

Why is this needed?

Properties of R_M

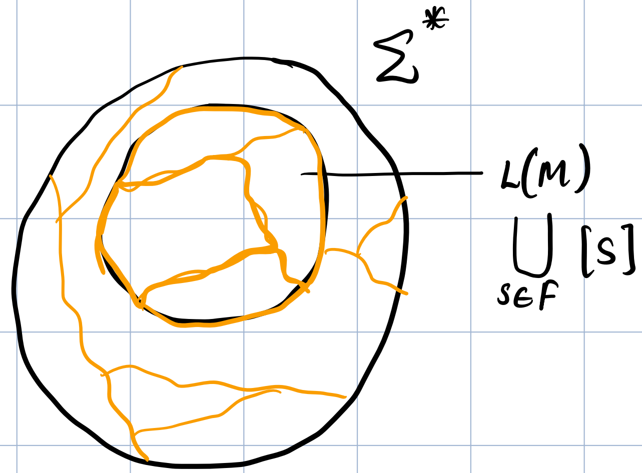
① Finite # of equivalence classes — Finite index

② $\forall x, y \in \Sigma^*$ and $\forall a \in \Sigma$ — Right congruence

$$(x,y) \in R_M \implies (xa, ya) \in R_M$$

③ R_M refines $L(M)$ represents the relation

$(x, y) \in R_M \Rightarrow$



A refines B

equivalence classes w.r.t A

equivalence classes w.r.t B

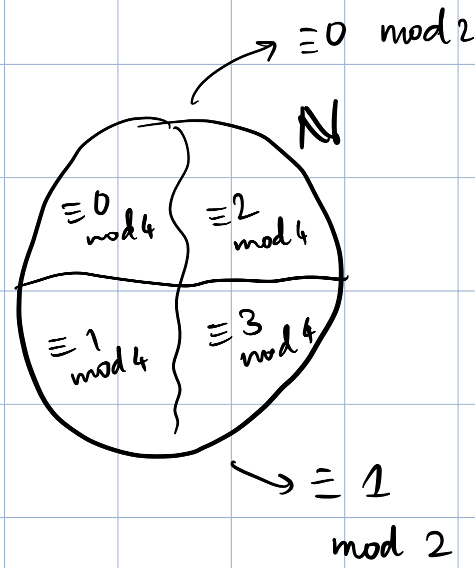
$C_i^A \subseteq C_j^B$

$$\text{relation}(L(M)) = \{ (x, y) \mid \text{both } x, y \in L(M) \}$$

mod 2 $\quad \mathbb{N} \times \mathbb{N}$

$(x, y) \in \text{mod } 2$ iff

$x \equiv y \mod 2$



mod 4 $\quad \mathbb{N} \times \mathbb{N}$

$(x, y) \in \text{mod } 4$ iff

$x \equiv y \mod 4$

Myhill - Nerode relation on L(M)

- → finite index
- → right congruence
- → refinement

- → need not be regular
- → any language
- → automaton is not required

06 Feb 2025

relation $(L(M)) \rightarrow 2$ equivalence classes

$$rel(L) = \{ (x, y) \in \Sigma^* \times \Sigma^* \mid x \in L \Leftrightarrow y \in L \}$$

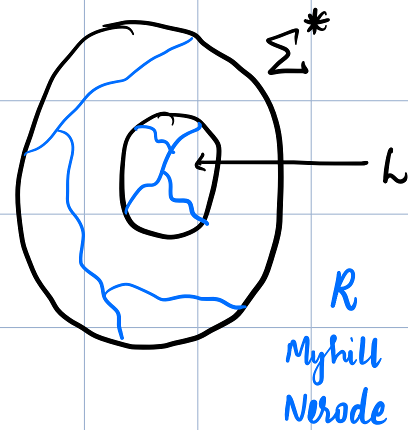
$\rightarrow L$

$\rightarrow \Sigma^* - L$

$$index(rel(L)) = 2$$

$L \neq \Sigma^*$

$$index(R) \geq index(rel(L))$$



Canonical Relation for L (R_L)

$\forall\ x, y \in \Sigma^*$

$(x,y) \in R_L \iff \forall\ z \in \Sigma^*$

$(xz \in L \iff yz \in L)$

cannot happen:

$\exists\ z \in \Sigma^*$

$xz \in L$ and $yz \notin L$

or vice-versa

* reflexive ✓
* symmetrix ✓
* transitive ✓

satisfies:

(i) right congruence

(ii) ??

(iii) R_L refines $rel(L)$

$$z = \varepsilon$$

$$(x \in L \iff y \in L)$$

Compilers

$\downarrow$

→ check syntax

→ use regular language

→ automata that accepts a program.

$\downarrow$

→ build efficient one

Claim: If L is regular, then the following holds:

1) R_L is of finite index.

2) Every Myhill - Nerode relation R for L refines R_L

2) $\Rightarrow$ 1) $\rightsquigarrow$ Myhill Nerode $\Rightarrow$ finite index

refines R_L $\Rightarrow$ $\text{index}(R) \geq \text{index}(R_L)$

$\therefore$ $\text{index}(R_L)$ is finite.

2) $\Rightarrow$ size of the minimal automaton accepting L is $\underbrace{\text{atleast}}_{\text{in fact, equal}}$ $\text{index}(R_L)$.

$Q := \{ [x] \mid x \in \Sigma^* \}$

$s := [\varepsilon]$

$F := \{ [x] \mid [x] \in L \}$

$[x] :=$ equivalence class of x w.r.t R_L

$$\delta([x], a) = [xa]$$

$$[x] = [y] \longrightarrow [xa] = [ya]$$

If $\delta([x], a) \neq \delta([y], a)$

$$\Downarrow$$

$$[x] \neq [y]$$

if $[x] = [y]$

$$\Rightarrow \underbrace{\delta([x], a) = \delta([y], a)}_{\text{by def}^{n}\ (z = \varepsilon)}$$

δ: Function
one shouldn't map to 2 things

$\hat{\delta}([\varepsilon], y) = [y] \in F$

y is accepted $\Longleftrightarrow [y] \in F$

$\Longleftrightarrow y \in L$

finite index claim $\longrightarrow$ finite index only when L is regular

$\{a^n b^n \mid n \geq 0\}$

$[a^{k_1}] \neq [a^{k_2}] \quad \forall \quad k_1 \neq k_2$