CS310 Automata Theory – 2016-2017

Nutan Limaye

Indian Institute of Technology, Bombay
nutan@cse.iitb.ac.in

Lecture 10: Finite state automata
January 23, 2017

Last class

The statement of the pumping lemma and the contrapositive.

Applications of the pumping lemma.

Equivalence relation on words.

Equivalence relation on words defined using DFAs.

Pumping lemma

A recipe for proving that a given language is non-regular.

Lemma (Pumping Lemma)

If L is a regular language, then $\exists p \in \mathbb{N}$ such that for any strings x, y, z with $x \cdot y \cdot z \in L$ and $|y| \geq p$,

1. *there exist strings u, v, w, s.t. y can be written as $y = u \cdot v \cdot w$,*
2. $\forall i \geq 0 \; x \cdot u \cdot v^i \cdot w \cdot z \in L$,
3. $|v| > 0$.

To prove that a given language L is not regular, the contrapositive of the above statement is useful.

Contrapositive of the pumping lemma

Lemma

We say that a language L satisfies **Property-NR** *if the following conditions hold:*

$$\forall p \geq 0,$$

$$\exists x, y, z \text{ such that } x \cdot y \cdot z \in L \text{ and } |y| \geq p,$$

$$\forall u, v, y \text{ such that } |v| > 0,\ y = u \cdot v \cdot w,$$

$$\exists i\ x \cdot u \cdot v^i \cdot w \cdot z \notin L.$$

If L satisfies Property-NR then L is not regular.

Relations on Σ

Let R be an equivalence relation on the set Σ^*, i.e. $R \subseteq \Sigma^* \times \Sigma^*$ such that

REFLEXIVE $\forall x \in \Sigma^* \; R(x,x)$ holds.

SYMMETRIC $\forall x,y \in \Sigma^* \; R(x,y) = R(y,x)$ hold.

TRANSITIVE $\forall x,y,z \in \Sigma^*$ if $R(x,y), R(y,z)$ hold then $R(x,z)$ also holds.

Relation of Σ^*

Let L be a regular language recognized by a DFA $A = (Q, \Sigma, \delta, q_0, F)$.

We say that $\forall x, y \in \Sigma^*$

$$x \equiv_A y \quad \text{iff} \quad \underbrace{\tilde{\delta}(q_0, x)}_{\substack{\text{state}\\ \text{reached}\\ \text{on } x\\ \text{from } q_0}} \quad = \quad \underbrace{\tilde{\delta}(q_0, y)}_{\substack{\text{state}\\ \text{reached}\\ \text{on } y\\ \text{from } q_0}}$$

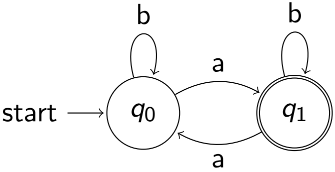
Assume that the auomaton is complete.

Observe that $\equiv_A$ is an equivalence relation.

Example

Example of an equivalence relation.

Consider the following automaton, say A.



$aab \equiv_A ababab a$.

$aabaaa \equiv_A a$.

The words with even number of a's form one equivalence class.

The words with odd number of a's form the other equivalence class.

There are no other equivalence classes.

Properties of equivalence relation on Σ^*

Definition (right congruence)

An equivalence relation $\equiv_A$ defined on Σ^* is said to be **a right congruence** if $\forall x, y \in \Sigma^*$ and $\forall a \in \Sigma$, $x \equiv y \implies x \cdot a \equiv y \cdot a$.

Definition (Refinement)

An equivalence relation $\equiv$ is said to **refine** a language L, if $x \equiv y$ then $(x \in L \iff y \in L)$.

Definition (Finite index)

An equivalence relation is said to have **finite index** if the number of equivalence classes defined by $\equiv$ is finite.

Lemma

For a DFA A, the equivalence relation $\equiv_A$ defined as before is

is a right congruence, refines $L(A)$, has finite index.

Properties of $\equiv_A$

Lemma

For a DFA A, the equivalence relation $\equiv_A$ defined as before is is a right congruence, refines $L(A)$, has finite index.

Proof.

right congruence

$$\begin{aligned}\tilde{\delta}(q_0, x \cdot a) &= \delta(\tilde{\delta}(q_0, x), a) \\ &= \delta(\tilde{\delta}(q_0, y), a) \because x \equiv_A y \\ &= \tilde{\delta}(q_0, y \cdot a)\end{aligned}$$

finite index

For $q \in Q$,

$[q] := \{w \in \Sigma^* \mid \tilde{\delta}(q_0, w) = q\}$

$\#$ equivalence classes $\leq |Q|$.

refinement

If $x \equiv_A y$

then $\tilde{\delta}(q_0, x) = \tilde{\delta}(q_0, y)$

$\therefore x, y$ both accepted or both rejected.

$\square$

Myhill-Nerode relation

Definition

An equivalence relation $\equiv$ on Σ^* is said to be **a Myhill-Nerode relation** for a language L if

- it is a right congruence
- refining L
- and has a finite index.

Lemma (Regular language $\implies$ Myhill-Nerode relation)

For any regular language there is a Myhill-Nerode relation.

What about the converse?

Non-regular languages

Let $L_{a,b} = \{a^n b^n \mid n \geq 0\}$.

Consider any relation $\equiv$ on $\{a, b\}^*$.

Assume that it is a right congruence and refines L.

Now we will show that it does not have finite index.

For $n \neq m$, can $a^n \equiv a^m$? NO!

$\because a^n b^n \in L$ but $a^m b^n \notin L$.

Let $FACTORIAL = \{a^{n!} \mid n \geq 0\}$.

Consider any relation $\equiv$ on $\{a\}^*$.

Assume that it is a right congruence and refines L.

Now we will show that it does not have finite index.

Can $a^{n!} \equiv a^{n+1!}$? NO!

$\because a^{n!} \cdot a^{n \cdot n!} \in L$ but $a^{n+1!} \cdot a^{n \cdot n!} \notin L$.

Converse also holds

Lemma

Let $L \subseteq \Sigma^*$. If there is a Myhill-Nerode relation for L then L is regular.

Proof idea

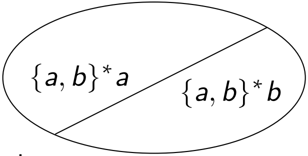
Using the relation, construct a finite state automaton.

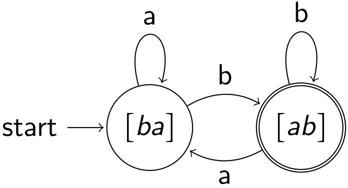
Let each equivalence class of the relation be a state of the automaton.

Define transitions naturally.

Converse: Myhill-Nerode relation $\implies$ regulararity

Example





$L = \{$words ending with $b\}$

Converse also holds

Lemma

Let $L \subseteq \Sigma^*$. If there is a Myhill-Nerode relation for L then L is regular.

Proof.

Construction

Let $\equiv$ be a Myhill-Nerode relation.

Let $[x] = \{y \mid y \equiv x\}$.

Let $A_\equiv = (Q, \Sigma, \delta, q_0, F)$ be defined as follows:

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

$q_0 = [\epsilon]$, $F = \{[x] \mid x \in L\}$,

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

Correctness: DIY. $\square$

Decision problems on regular languages

Acceptance problem (for fixed Σ)

Given: DFA A, input string $w \in \Sigma^*$
Output: "yes" iff A accepts w.

Construct a graph from an automaton:

> Let $Q = \{q_0, \ldots, q_{m-1}\}$, q_0 be the start state,
> $F \subseteq Q$ be the set of final states.
>
> Create a layered graph $G_{A,n}$, where $|w| = n$, as follows:
>
> Make $n+1$ copies of Q: $Q_0, Q_1, \ldots, Q_n$, where $Q_i = \{q_{i,0}, \ldots, q_{i,m-1}\}$.
>
> Add edge $(q_{i,u}, q_{i+1,v})$ with label $a \in \Sigma$
> if $\delta(q_u, a) = q_v$.

Lemma

There is a path from $q_{0,0}$ to $q_{n,u}$ labelled by a string w in $G_{A,|w|}$ if and only if $\tilde{\delta}(q_0, w) = q_u$ in A.

Decision problems on regular languages

Nonemptiness problem (for fixed Σ)

Given: DFA A

Output: "yes" iff $\exists w : A$ accepts w.

Lemma

If a DFA $A = (Q, \Sigma, \delta, q_0, F)$ accepts some string then it accepts a string of length $\leq |Q|$.

Minimization problem

Minimization problem (for fixed Σ)

Given:	DFA A
Output:	DFA B s.t. $L(A) = L(B)$ and B has the smallest number of states possible for recognizing $L(A)$

Definition

Let $A = (Q, \Sigma, \delta, q_0, F)$. We call states p, q indistinguishable if $\forall w \in \Sigma^*$, $\tilde{\delta}(p, w) \Leftrightarrow \tilde{\delta}(q, w)$.

Minimization algorithm.

Identify indistinguishable states.

Collapse them.

Minimization problem

Minimization problem (for fixed Σ)

Given: DFA A

Output: DFA B s.t. $L(A) = L(B)$ and B has the smallest number of states possible for recognizing $L(A)$

Example

	0	1	2	3	4	5
a	1	2	3	4	5	0

(Red color indicates final states.)

0					
–	1				
–	–	2			
–	–	–	3		
–	–	–	–	4	
–	–	–	–	–	5

Minimization problem

Minimization problem (for fixed Σ)

Given: DFA A

Output: DFA B s.t. $L(A) = L(B)$ and B has the smallest number of states possible for recognizing $L(A)$

Example

	0	1	2	3	4	5
a	1	2	3	4	5	0

(Red color indicates final states.)

0					
–	1				
–	–	2			
–	–	–	3		
–	–	–	–	4	
–	–	–	–	–	5

0					
✓	1				
–	✓	2			
–	✓	–	3		
✓	–	✓	✓	4	
–	✓	–	–	✓	5

Minimization problem

Minimization problem (for fixed Σ)

Given: DFA A

Output: DFA B s.t. $L(A) = L(B)$ and B has the smallest number of states possible for recognizing $L(A)$

Example

	0	1	2	3	4	5
a	1	2	3	4	5	0

(Red color indicates final states.)

0					
–	1				
–	–	2			
–	–	–	3		
–	–	–	–	4	
–	–	–	–	–	5

0					
✓	1				
✓	✓	2			
–	✓	–	3		
✓	–	✓	✓	4	
–	✓	–	–	✓	5

Minimization problem

Minimization problem (for fixed Σ)

Given: DFA A

Output: DFA B s.t. $L(A) = L(B)$ and B has the smallest number of states possible for recognizing $L(A)$

Example

	0	1	2	3	4	5
a	1	2	3	4	5	0

(Red color indicates final states.)

0					
–	1				
–	–	2			
–	–	–	3		
–	–	–	–	4	
–	–	–	–	–	5

0					
✓	1				
✓	✓	2			
–	✓	–	3		
✓	–	✓	✓	4	
✓	✓	–	–	✓	5

Minimization problem

Minimization problem (for fixed Σ)

Given: DFA A

Output: DFA B s.t. $L(A) = L(B)$ and B has the smallest number of states possible for recognizing $L(A)$

Example

	0	1	2	3	4	5
a	1	2	3	4	5	0

(Red color indicates final states.)

0					
–	1				
–	–	2			
–	–	–	3		
–	–	–	–	4	
–	–	–	–	–	5

0					
✓	1				
✓	✓	2			
–	✓	✓	3		
✓	–	✓	✓	4	
✓	✓	–	–	✓	5

Minimization problem

Minimization problem (for fixed Σ)

Given: DFA A

Output: DFA B s.t. $L(A) = L(B)$ and B has the smallest number of states possible for recognizing $L(A)$

Example

	0	1	2	3	4	5
a	1	2	3	4	5	0

(Red color indicates final states.)

0					
–	1				
–	–	2			
–	–	–	3		
–	–	–	–	4	
–	–	–	–	–	5

0					
✓	1				
✓	✓	2			
–	✓	✓	3		
✓	–	✓	✓	4	
✓	✓	–	✓	✓	5

Minimization problem

Minimization problem (for fixed Σ)

Given: DFA A

Output: DFA B s.t. $L(A) = L(B)$ and B has the smallest number of states possible for recognizing $L(A)$

Example

	0	1	2	3	4	5
a	1	3	4	5	5	5
b	2	4	3	5	5	5

(Red color indicates final states.)