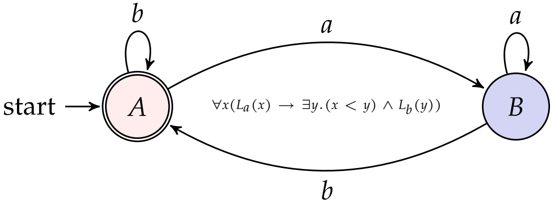
CS 208: Automata Theory and Logic

Lecture 5: Pumping Lemma and Myhill-Nerode Theorem

Ashutosh Trivedi



Department of Computer Science and Engineering,
Indian Institute of Technology Bombay.

Closure Properties of Regular Languages

Operations that preserve regularity of languages:

- union, intersection, complement, difference

Closure Properties of Regular Languages

Operations that preserve regularity of languages:

- union, intersection, complement, difference
- concatenation and Kleene closure (star)

Closure Properties of Regular Languages

Operations that preserve regularity of languages:

- union, intersection, complement, difference
- concatenation and Kleene closure (star)
- Reversal
  - reversal $\overline{w}$ of a string w is defined as:

$$\overline{w} = \begin{cases} \varepsilon & \text{if } w = \varepsilon \\ a\overline{x} & \text{if } w = xa \text{ where } x \in \Sigma^* \text{ and } a \in \Sigma \end{cases}$$

  - $\overline{L} = \{\overline{w} \ : \ w \in L\}$.
  - Swap initial and accepting states, and reverse the transitions, i.e. $\overline{\delta}(s, a) = s'$ iff $\delta(s', a) = s$.
  - Proof of correctness is via structural induction over regular expressions

Closure Properties of Regular Languages

Operations that preserve regularity of languages:

- union, intersection, complement, difference
- concatenation and Kleene closure (star)
- Reversal
  - reversal $\overline{w}$ of a string w is defined as:

$$\overline{w} = \begin{cases} \varepsilon & \text{if } w = \varepsilon \\ a\overline{x} & \text{if } w = xa \text{ where } x \in \Sigma^* \text{ and } a \in \Sigma \end{cases}$$

  - $\overline{L} = \{\overline{w} \ : \ w \in L\}$.
  - Swap initial and accepting states, and reverse the transitions, i.e. $\overline{\delta}(s, a) = s'$ iff $\delta(s', a) = s$.
  - Proof of correctness is via structural induction over regular expressions
- Homomorphism and inverse-homomorphism
  - String homomorphism is a function $h : \Sigma \to \Gamma^*$
  - Extended string homomorphism $\hat{h} : \Sigma^* \to \Gamma^*$
  - For $L \in \Sigma^*$ we define $h(L) \subseteq \Gamma^*$ as $h(L) = \{\hat{h}(w) \ : \ w \in L\}$.
  - For $L \in \Gamma^*$ we define $h^{-1}(L) \subseteq \Sigma^*$ as $h^{-1}(L) = \{w \ : \ \hat{h}(w) \in L\}$.

Closure under Homomorphism

Example: Let $h(0) = ab$ and $h(1) = \varepsilon$ and $L = 10^*1$ then $h(L) = (ab)^*$.

Closure under Homomorphism

Example: Let $h(0) = ab$ and $h(1) = \varepsilon$ and $L = 10^*1$ then $h(L) = (ab)^*$.

Theorem (Closure under Homomorphism)

For a homomorphism $h : \Sigma \to \Gamma^*$ if $L \subseteq \Sigma^*$ is regular then so is $h(L) \subseteq \Gamma^*$.

Closure under Homomorphism

Example: Let $h(0) = ab$ and $h(1) = \varepsilon$ and $L = 10^*1$ then $h(L) = (ab)^*$.

Theorem (Closure under Homomorphism)

For a homomorphism $h : \Sigma \to \Gamma^*$ if $L \subseteq \Sigma^*$ is regular then so is $h(L) \subseteq \Gamma^*$.

Proof.

- Consider the regular expression $E(L)$ characterizing L,
- Replace the alphabets a in $E(L)$ by string $h(a)$
- It is easy to see (by structural induction) that the corresponding expression is also a regular expression.

□

Closure under Homomorphism

Example: Let $h(0) = ab$ and $h(1) = \varepsilon$ and $L = 10^*1$ then $h(L) = (ab)^*$.

Theorem (Closure under Homomorphism)

For a homomorphism $h : \Sigma \to \Gamma^*$ if $L \subseteq \Sigma^*$ is regular then so is $h(L) \subseteq \Gamma^*$.

Proof.

- Consider the regular expression $E(L)$ characterizing L,
- Replace the alphabets a in $E(L)$ by string $h(a)$
- It is easy to see (by structural induction) that the corresponding expression is also a regular expression.

□

Corollary

Regular languages are closed under projections (dropping of certain alphabets).

Closure under Homomorphism

Example: Let $h(0) = ab$ and $h(1) = \varepsilon$ and $L = 10^*1$ then $h(L) = (ab)^*$.

Theorem (Closure under Homomorphism)

For a homomorphism $h : \Sigma \to \Gamma^*$ if $L \subseteq \Sigma^*$ is regular then so is $h(L) \subseteq \Gamma^*$.

Proof.

- Consider the regular expression $E(L)$ characterizing L,
- Replace the alphabets a in $E(L)$ by string $h(a)$
- It is easy to see (by structural induction) that the corresponding expression is also a regular expression.

□

Corollary

Regular languages are closed under projections (dropping of certain alphabets).

Theorem (Closure under Substitution)

For a substitution $h : \Sigma \to REGEX(\Gamma)$ if $L \subseteq \Sigma^*$ is regular then so is $h(L) \subseteq \Gamma^*$.

Closure under Inverse-Homomorphism

Example: Let $h(0) = ab$ and $h(1) = \varepsilon$ and $L = (ab)^*$ then $h^{-1}(L) = (0+1)*$.

Closure under Inverse-Homomorphism

Example: Let $h(0) = ab$ and $h(1) = \varepsilon$ and $L = (ab)^*$ then $h^{-1}(L) = (0 + 1)*$.

Theorem (Closure under Homomorphism)

For a homomorphism $h : \Sigma \to \Gamma^*$ if $L \subseteq \Gamma^*$ is regular then so is $h^{-1}(L) \subseteq \Sigma^*$.

Closure under Inverse-Homomorphism

Example: Let $h(0) = ab$ and $h(1) = \varepsilon$ and $L = (ab)^*$ then $h^{-1}(L) = (0+1)*$.

Theorem (Closure under Homomorphism)

For a homomorphism $h : \Sigma \to \Gamma^*$ if $L \subseteq \Gamma^*$ is regular then so is $h^{-1}(L) \subseteq \Sigma^*$.

Proof.

- Consider the DFA $\mathcal{A}(L) = (S, \Sigma, \delta, s_0, F)$ characterizing L,
- The DFA corresponding to $h^{-1}(L)$ is $(S, \Gamma, \gamma, s_0, F)$ such that

$$\gamma(s, a) = \hat{\delta}(s, h(a)).$$

- Proof via induction on string size that $\hat{\gamma}(s, w) = \hat{\delta}(s, h(w))$.

□

Pumping Lemma

Myhill-Nerode Theorem

Some languages are not regular!

Let's do mental computations again.

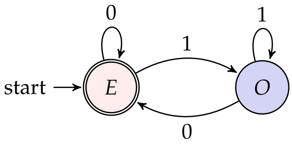
- The language $\{0^n 1^n \ : \ n \geq 0\}$
- The set of strings having an equal number of 0's and 1's
- The set of strings with an equal number of occurrences of 01 and 10.
- The language $\{ww \ : \ w \in \{0,1\}^*\}$
- The language $\{w\overline{w} \ : \ w \in \{0,1\}^*\}$
- The language $\{0^i 1^j \ : \ i > j\}$
- The language $\{0^i 1^j \ : \ i \leq j\}$
- The language of palindromes of $\{0,1\}$

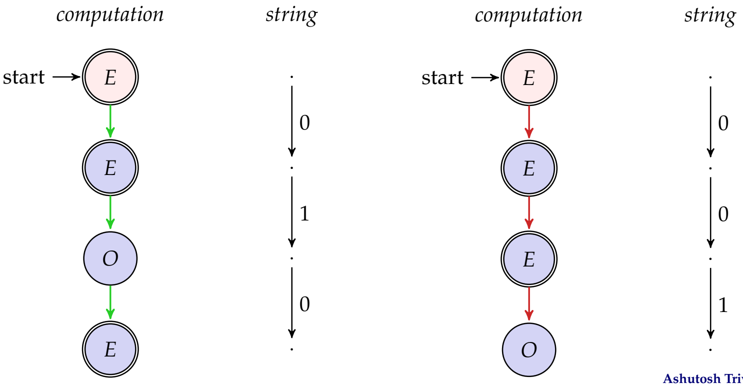
Some languages are not regular!

Let's do mental computations again.

- The language $\{0^n 1^n \ : \ n \geq 0\}$
- The set of strings having an equal number of 0's and 1's
- The set of strings with an equal number of occurrences of 01 and 10.
- The language $\{ww \ : \ w \in \{0,1\}^*\}$
- The language $\{w\overline{w} \ : \ w \in \{0,1\}^*\}$
- The language $\{0^i 1^j \ : \ i > j\}$
- The language $\{0^i 1^j \ : \ i \leq j\}$
- The language of palindromes of $\{0,1\}$

A simple observation about DFA





A simple observation about DFA

Image source: Wikipedia

- Let $A = (S, \Sigma, \delta, s_0, F)$ be a DFA.
- For every string $w \in \Sigma^*$ of the length greater than or equal to the number of states of A, i.e. $|w| \geq |S|$, we have that
- the unique computation of A on w re-visits at least one state.

Pumping Lemma

Theorem (Pumping Lemma for Regular Languages)

If L is a regular language, then
there exists a constant (pumping length) p such that
for every string $w \in L$ s.t. $|w| \geq p$
there exists a division of w in strings x, y, and z s.t. $w = xyz$ such that

1. $|y| > 0$,
2. $|xy| \leq p$, *and*
3. *for all* $i \geq 0$ *we have that* $xy^iz \in L$.

Pumping Lemma

Theorem (Pumping Lemma for Regular Languages)

If L is a regular language, then
there exists a constant (pumping length) p such that
for every string $w \in L$ s.t. $|w| \geq p$
there exists a division of w in strings x, y, and z s.t. $w = xyz$ such that

1. $|y| > 0$,
2. $|xy| \leq p$, *and*
3. *for all* $i \geq 0$ *we have that* $xy^iz \in L$.

- Let A be the DFA accepting L and p be the set of states in A.
- Let $w = (a_1a_2 \dots a_k) \in L$ be any string of length $\geq p$.
- Let $s_0a_1s_1a_2s_2 \dots a_ks_k$ be the run of w on A.
- Let i be the index of first state that the run revisits and let j be the index of second occurrence of that state, i.e. $s_i = s_j$,
- Let $x = a_1a_2 \dots a_{i-1}$ and $y = a_ia_{i+1} \dots a_{j-1}$, and $z = a_ja_{j+1} \dots a_k$.
- notice that $|y| > 0$ and $|xy| \leq n$
- Also, notice that for all $i \geq 0$ the string xy^iz is also in L.

Applying Pumping Lemma

How to show that a language L is non-regular.

1. Assume that L is regular and get contradiction with pumping lemma.
2. Let n be the pumping length.
3. (Cleverly) find a representative string w of L of size greater or equal to n.
4. Try out all ways to break the string into xyz triplet satisfying that $|y| > 0$ and $|xy| \leq n$. If the step 3 was clever enough, there will be finitely many cases to consider.
5. For every triplet show that for some i the string xy^iz is not in L, and hence it yields contradiction with pumping lemma.

Examples: 1.73, 1.74, 1.75, and 1.77.

Pumping Lemma

Myhill-Nerode Theorem

Equivalence and Minimization of DFA

Minimization of a DFA:

- Two states q, q' are equivalent, $q \equiv q'$, if for all strings w we have that $\hat{\delta}(q, w) \in F$ if and only if $\hat{\delta}(q', w) \in F$.

Equivalence and Minimization of DFA

Minimization of a DFA:

- Two states q, q' are equivalent, $q \equiv q'$, if for all strings w we have that $\hat{\delta}(q, w) \in F$ if and only if $\hat{\delta}(q', w) \in F$.
- It is easy to see that $\equiv$ is an equivalence relation and thus it partitions the set of all states into equivalence classes.
- States in the same class can be merged without changing the language of the DFA.
- Quotient Construction: To minimize a DFA find all classes of equivalent states and merge them.
- Given such an equivalence relation, $\equiv$, formalize this quotient construction and prove its correctness.

Equivalence and Minimization of DFA

How to find equivalent states:

- Notice that an accepting state q is distinguishable from a non-accepting state q' as $\hat{\delta}(q, \varepsilon) \in F$ while $\hat{\delta}(q', \varepsilon) \notin F$.

Equivalence and Minimization of DFA

How to find equivalent states:

- Notice that an accepting state q is distinguishable from a non-accepting state q' as $\hat{\delta}(q, \varepsilon) \in F$ while $\hat{\delta}(q', \varepsilon) \notin F$.
- We can mark such state pairs distinguishable.

Equivalence and Minimization of DFA

How to find equivalent states:

- Notice that an accepting state q is distinguishable from a non-accepting state q' as $\hat{\delta}(q, \varepsilon) \in F$ while $\hat{\delta}(q', \varepsilon) \notin F$.
- We can mark such state pairs distinguishable.
- Then iteratively keep on marking states distinguishable if in one step after reading a same alphabet they respectively reach to two distinguishable states.

Equivalence and Minimization of DFA

How to find equivalent states:

- Notice that an accepting state q is distinguishable from a non-accepting state q' as $\hat{\delta}(q, \varepsilon) \in F$ while $\hat{\delta}(q', \varepsilon) \notin F$.
- We can mark such state pairs distinguishable.
- Then iteratively keep on marking states distinguishable if in one step after reading a same alphabet they respectively reach to two distinguishable states.
- If in a step no new distinguishable state is marked then the process terminates.
- This process suggests an algorithm that is known as table filling algorithm.

Myhill-Nerode Theorem

- Let L be a language
- Two strings x and y are distinguishable in L if there exists z such that exactly one of xz and yz in L.
- We define a relation R_L (Myhill-Nerode relation) such that strings x, y we have that $(x, y) \in R_L$ is if x and y are not distinguishable in L.
- It is easy to see that R_A is an equivalence relation and thus it partitions the set of all strings into equivalence classes.

Myhill-Nerode Theorem

- Let L be a language
- Two strings x and y are distinguishable in L if there exists z such that exactly one of xz and yz in L.
- We define a relation R_L (Myhill-Nerode relation) such that strings x, y we have that $(x, y) \in R_L$ is if x and y are not distinguishable in L.
- It is easy to see that R_A is an equivalence relation and thus it partitions the set of all strings into equivalence classes.

Theorem (Myhill-Nerode Theorem)

A language L is regular if and only if R_L has a finite number of equivalence classes. Moreover, the number of states is the smallest DFA recognizing L is equal to the number of equivalence classes of R_L.

Myhill-Nerode Theorem

- Let L be a language
- Two strings x and y are distinguishable in L if there exists z such that exactly one of xz and yz in L.
- We define a relation R_L (Myhill-Nerode relation) such that strings x, y we have that $(x, y) \in R_L$ is if x and y are not distinguishable in L.
- It is easy to see that R_A is an equivalence relation and thus it partitions the set of all strings into equivalence classes.

Theorem (Myhill-Nerode Theorem)

A language L is regular if and only if R_L has a finite number of equivalence classes. Moreover, the number of states is the smallest DFA recognizing L is equal to the number of equivalence classes of R_L.

Corollary

There exists a unique minimal DFA for every regular language.

Myhill-Nerode Theorem

Theorem (Myhill-Nerode Theorem)

A language L is regular if and only if R_L has a finite number of equivalence classes. Moreover, the number of states is the smallest DFA recognizing L is equal to the number of equivalence classes of R_L.

Proof.

The "Only if" direction:

- Let L be regular and DFA $A = (S, \Sigma, \delta, s_0, F)$ accepts this languages.
- The indistinguishability relation R_L is defined using states of $A(L)$: two strings are indistinguishable if $\hat{\delta}(s_0, x) = \hat{\delta}(s_0, y)$.
- Notice that this relation has finitely many partitions (number of states of A and strings in one class are indistinguishable.

□

Myhill-Nerode Theorem

Theorem (Myhill-Nerode Theorem)

A language L is regular if and only if R_L has a finite number of equivalence classes. Moreover, the number of states is the smallest DFA recognizing L is equal to the number of equivalence classes of R_L.

Proof.

The "if" direction:

- Let R_L be the indistinguishability relation with finitely many equivalence classes.
- Let each class represent a state of a DFA, where starting state is the class containing ε, and the set final states is the set of equivalence classes containing strings in L.
- For two equivalence classes c and c' we have that $\delta(c, a) = c'$ if for some arbitrary string w in c we have that $wa \in c'$. By definition of Myhill-Nerode relation transition function is well-defined.

□