Characterization of Regular languages

Somitra Sanadhya

IIT-Ropar, Punjab
somitra@iitrpr.ac.in

Feb 2020

Recall

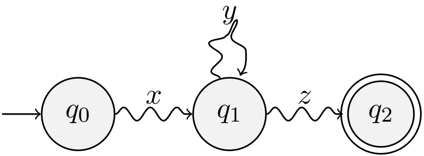
- Expressive power of the following automata/expressions are the same:
  DFA, NFA, NFA with ε transitions, Regex.
- Use depends on convenience in the given situation. For example, thinking about some languages may be easy with ε-NFA but implementation requires a DFA.
- How do we know if a given language in Regular? Just because one can't construct a DFA does not mean that there does not exist a DFA.

Motivation

- Given an automata M, how can you know if M does not accept any string ($L(M) = \phi$)?
-
- Given an automata M, how can you know if M accepts infinite strings?
-

M accepting infinite strings

- Let the DFA have n states.
- Consider a string w such that $|w| \geq n$. Then $w = xyz$ such that:



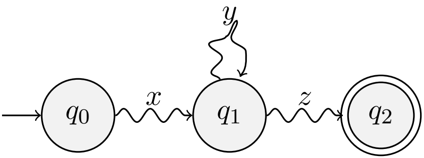
(Note: Zigzag line to denote strings, rather than alphabets)

- Why?

Improving the test

- The previous condition is not an efficient test.
- The number of strings of length $\geq n$ is infinite.
- Can we limit the string length?
- Prove that there exists a string $w \in L(M)$ such that $n \leq |w| \leq (2n-1)$.
- Hint: Note that there can be more than 1 loops. Take y to be first loop on the path. Then $|xy| \leq n$, and $|y| \geq 1$.

M accepting infinite strings



- Let y denote the first loop on the path.
- Then $|xy| \leq n$, $|y| \geq 1$, $|xyz| \geq n$.
- All strings of the form xy^iz will be in $L(M)$!

Pumping Lemma for regular languages

Pumping lemma

For every regular language L, $\exists n \in \mathbb{N}$ such that $\forall w \in L$, such that $|w| \geq n$, we can write $w = xyz$ such that

- $|xy| \leq n$
- $|y| \geq 1$

then $xy^i z \in L$ for all $i \geq 0$.

Some comments

- Look at the alternating quantifiers:

(1) $\forall$ regular languages L

(2) $\exists$ a number $n \geq 1$ such that

(3) $\forall$ strings w of length $\geq n$

(4) $\exists$ strings x, y, z such that $w = xyz$ and $|y| \geq 1$, such that

(5) $\forall i \geq 0$, $xy^iw \in L$.

Some comments

- It helps to think in terms of a two party interactive protocol.
- Prover: L is regular, Verifier: Testing the claim with skepticism.
- Prover gives an n. The verifier supplies a string $w \in L$ such that $|w| \geq n$. (Intuitively, the verifier tries to come up with the most challenging w for the prover.)
- Prover produces a decomposition of w into xyz and gives it to the verifier.
- Verifier attempts to find an i such that $xy^iz \notin L$. If she can't find such an i then the prover has won.

More comments

- We proved that all regular languages satisfy the Pumping lemma.
- But it does not imply the converse.
- That is, the pumping lemma is a necessary but not sufficient condition for regular languages.

hw Find a language which satisfies the conditions of the pumping lemma but is not regular.

Example

- Prove that $L = \{w | \ w$ has equal number of 0's and 1's$\}$ is not regular.
- Take n to be the "pumping length".
- Take $w = 0^n 1^n$. Clearly, $w \in L$.
- No matter how you divide w into xyz, both x and y consist of only 0's.
- But now $xy^i z$ can't be in L (for say, $i = 5$) because it has more 0's than 1's.
- Therefore L is not regular.
- (Note that the choice of w is crucial. Not every choice may work. Refer to the interactive protocol comment again.)

Implication

- An html page has tags of the kind <a> . . . .
- A C program has statements within { ... }
- And both of the above examples can be nested repeatedly.
- No regex can parse html tags or C programs !

Finding a necessary and sufficient condition for regular languages

Trivia: We call such a property as a "characteristic" of the object under study.

- Consider a language L which is not regular.
- What causes it to be non-regular?
- Que: Can a finite language be non-regular?

Non-regular language

- Such a langauge is clearly infinite.
- But the number of states is only finite.
- A state in an automata can remember only "some details" about how it reached there. (Back-traversal is not possible).
- Thus, a finite automata represents machines which have a finite memory.
- A non-regular language must require infinite memory !

Example

- Consider the language $L = \{0^n 1^n \ \text{ where } n \in \mathbb{N}\}$.
- Intuitively, you need to memorize how many 0's have you seen so far, before 1's start coming in.
- Therefore, L should be non-regular.
- But how do we formally show that the language is not regular (without using the pumping lemma)?

Example : continued

- Consider the language $L = \{0^n 1^n \text{ where } n \in \mathbb{N}\}$.
- We will prove the non-regularity of L by contradiction.
- Let there be a DFA M with n states which accepts L.
- Consider $S = \{0, 00, 000, \dots\}$.
- While consuming different strings from S, the automata reaches some intermediate states.
- Pick $(n+1)$ different strings from S.

Myhill-Nerode theorem

Defn 1 Let L be a language, and x, y be two distinct strings. The strings are called "distinguishable with respect to L" if and only if $\exists$ a string w (possibly empty) such that $xw \in L$ but $yw \notin L$.

Theorem Let L be a language. If there exists an infinite set S such that any two distinct strings $x, y \in S$ are distinguishable with respect to L, then L is non regular.

Proof ...

See the details in the additional notes on the course website.