

## CS 39

### Theory of Computation

## Simple computers

- The basic question: What is a computer?
- We'll start by looking at very simple computers.
  - Input: anything
  - Output: YES or NO  
(ACCEPT or REJECT)
- Each such computer *recognizes a language*.

## Languages

- What is a “language”?
  - [Merriam-Webster] “a systematic means of communicating ideas or feelings by the use of conventionalized signs...”
  - Examples: English, Spanish, Mandarin, Swahili, ...
  - More examples: Pascal, Scheme, C, Perl, ...
- For our purposes, a language is a set of strings.
  - English = {“Hello.”, “Come here.”, “Programming is fun.”, “In God we trust.”, “Crows are green.”, ...}
  - Perl = {p : p is a syntactically correct Perl program}
  - L = {“a”, “b”, “aa”, “bb”, “ab”, “ba”}

## Precise definitions

- A language is a set of *strings*.
- A string is a sequence of characters/symbols.
  - Note: sequence (ordered) versus set (unordered)
  - When writing a string, omit commas.
    - “abc” instead of (a,b,c).
- A character/symbol is an element of an *alphabet*.
- An alphabet is any *nonempty finite set*.
  - Alphabets usually denoted by Greek caps:  $\Sigma$  or  $\Gamma$ .

## Terminology

- We speak of a string or language *over* a certain alphabet.
  - Thus, “abracadabra” is a string over the alphabet  $\{a,b,c,\dots,y,z\}$ .
  - “καλημερα” is a string over the alphabet  $\{\alpha,\beta,\gamma,\dots,\psi,\omega\}$ .
  - $\{a, b, aa, bb, ab, ba\}$  is a language over the alphabet  $\{a,b\}$ . It’s also a language over the alphabet  $\{a,\dots,z\}$ .

## Operations on strings

- Let  $w, x$  be strings over alphabet  $\Sigma$ .
- $|w|$  = length of  $w$  = number of chars in  $w$ .
- $wx = w \circ x$  = concatenation:  $w$  followed by  $x$ .
- $w^R$  = reverse of  $w$ .
- Special:  $\varepsilon$  = empty string

## Operations on strings

- Let  $w, x$  be strings over alphabet  $\Sigma$ .
  - E.g.,  $w = abc, x = caeab$
- $|w|$  = length of  $w$  = number of chars in  $w$ .
  - $|w| = 3, |x| = 5$
- $wx = w \circ x$  = concatenation:  $w$  followed by  $x$ .
  - $wx = abccaeb, xw = caeababc$
- $w^R$  = reverse of  $w$ .
  - $w^R = cba, x^R = baeac$
- Special:  $\varepsilon$  = empty string
  - $|\varepsilon| = 0, \varepsilon w = w\varepsilon = w$ .

## Prefixes and suffixes

- Any “initial segment” of a string  $w$  is called a *prefix* of  $w$ .
- Let us make this precise and mathematical: “initial segment” is imprecise, though intuitive.
- **Definition:** The string  $x$  is a prefix of the string  $w$  iff  $w = xy$  for some string  $y$ .
- Even more precisely: ... iff  $\exists y (w = xy)$ .
- How do you think we should define suffixes?

## Practice

- Is  $c$  a prefix of  $cacba$ ?
  - Yes, if  $x = c$ , take  $y = acba$  to get  $cacba = xy$ .
- Is  $cacba$  a prefix of  $cacba$ ?
  - Yes, if  $x = cacba$ , take  $y = \epsilon$  to get  $cacba = xy$ .
- Is  $\epsilon$  a prefix of  $cacba$ ?
  - Yes, if  $x = \epsilon$ , take  $y = cacba$  to get  $cacba = xy$ .
  - Same reasoning shows that  $\epsilon$  is a prefix of *any* string.
- Is  $\epsilon$  a prefix of  $\epsilon$ ?
  - Sure!

## More practice

- If  $x$  is a prefix of  $w$ , is  $x^{\mathcal{R}}$  a prefix of  $w^{\mathcal{R}}$ ?
  - No; take  $w = abcdef$ ,  $x = ab$
  - Then  $w^{\mathcal{R}} = fedcba$ ,  $x^{\mathcal{R}} = ba$ ; not a prefix! But...
  - It seems  $x^{\mathcal{R}}$  is a *suffix* of  $w^{\mathcal{R}}$ . Can you prove this?
- Can you express  $(xy)^{\mathcal{R}}$  in terms of  $x^{\mathcal{R}}$  and  $y^{\mathcal{R}}$ ?
  - $(xy)^{\mathcal{R}} = y^{\mathcal{R}}x^{\mathcal{R}}$
- If  $x$  is a string what do  $x^2$ ,  $x^3$ , etc. denote?
  - $x^2 = xx$ ,  $x^3 = xxx$ , etc. Thus,  $(abc)^2 = abcabc$ .