Algorithms & Models of Computation CS/ECE 374, Fall 2017

# **Strings and Languages**

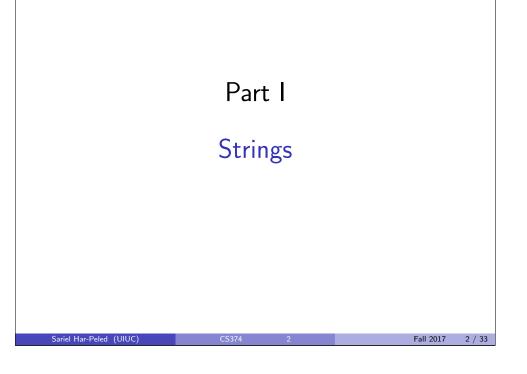
Lecture 1b Tuesday, August 29, 2017

### String Definitions

Sariel Har-Peled (UIUC

### Definition

- An alphabet is a finite set of symbols. For example  $\Sigma = \{0, 1\}, \Sigma = \{a, b, c, \dots, z\},$ 
  - $\Sigma = \{ \langle \text{moveforward} \rangle, \langle \text{moveback} \rangle \}$  are alphabets.
- (a)  $\epsilon$  is the empty string.
- The length of a string w (denoted by |w|) is the number of symbols in w. For example, |101| = 3,  $|\epsilon| = 0$
- So For integer  $n \ge 0$ , Σ<sup>n</sup> is set of all strings over Σ of length n. Σ\* is th set of all strings over Σ.



### Formally

Formally strings are defined recursively/inductively:

- $\epsilon$  is a string of length **0**
- ax is a string if  $a \in \Sigma$  and x is a string. The length of ax is 1 + |x|

The above definition helps prove statements rigorously via induction.

• Alternative recursive definiton useful in some proofs: xa is a string if  $a \in \Sigma$  and x is a string. The length of xa is 1 + |x|

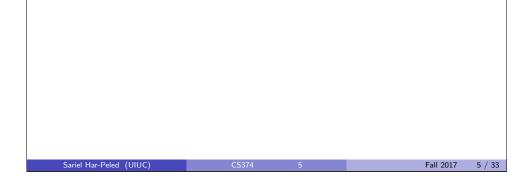
#### Convention

- $a, b, c, \ldots$  denote elements of  $\Sigma$
- $w, x, y, z, \ldots$  denote strings
- A, B, C, ... denote sets of strings

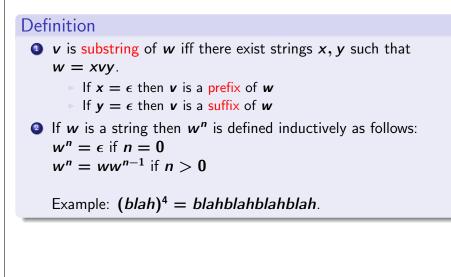
Fall 2017

### Much ado about nothing

- $\epsilon$  is a string containing no symbols. It is not a set
- {e} is a set containing one string: the empty string. It is a set, not a string.
- $\emptyset$  is the empty set. It contains no strings.
- {Ø} is a set containing one element, which itself is a set that contains no elements.



# Substrings, prefix, suffix, exponents



# Concatenation and properties

- If x and y are strings then xy denotes their concatenation. Formally we define concatenation recursively based on definition of strings:
  - xy = y if  $x = \epsilon$
  - xy = a(wy) if x = aw

Sometimes xy is written as  $x \bullet y$  to explicitly note that  $\bullet$  is a binary operator that takes two strings and produces another string.

- concatenation is associative: (uv)w = u(vw) and hence we write uvw
- **not** commutative: uv not necessarily equal to vu
- identity element:  $\epsilon u = u\epsilon = u$

# Set Concatenation

### Definition

Given two sets A and B of strings (over some common alphabet  $\Sigma$ ) the concatenation of A and B is defined as:

$$AB = \{xy \mid x \in A, y \in B\}$$

Example:  $A = \{fido, rover, spot\}, B = \{fluffy, tabby\}$  then  $AB = \{fidofluffy, fidotabby, roverfluffy, \ldots\}$ .

CS374

Fall 2017

# $\boldsymbol{\Sigma}^*$ and languages

### Definition

- Σ<sup>n</sup> is the set of all strings of length n. Defined inductively as follows:
  - $\Sigma^{n} = \{\epsilon\} \text{ if } n = 0$  $\Sigma^{n} = \Sigma \Sigma^{n-1} \text{ if } n > 0$
- $\ \, {\bf } {\bf$
- **3**  $\Sigma^+ = \bigcup_{n \ge 1} \Sigma^n$  is the set of non-empty strings.

#### Definition

A language L is a set of strings over  $\Sigma$ . In other words  $L \subseteq \Sigma^*$ .

# Canonical order and countability of strings

### Definition

Sariel Har-Peled (UIUC

An set A is countably infinite if there is a bijection f between the natural numbers and A.

Alternatively: A is countably infinite if A is an infinite set and there enumeration of elements of A

#### Theorem

 $\boldsymbol{\Sigma}^*$  is countably infinite for every finite  $\boldsymbol{\Sigma}$ .

Enumerate strings in order of increasing length and for each given length enumerate strings in dictionary order (based on some fixed ordering of  $\Sigma$ ).

Example:  $\{0,1\}^* = \{\epsilon, 0, 1, 00, 01, 10, 11, 000, 001, 010, \ldots\}$ .  $\{a, b, c\}^* = \{\epsilon, a, b, c, aa, ab, ac, ba, bb, bc, \ldots\}$ 



Fall 2017

### Exercise

Answer the following questions taking  $\Sigma = \{0, 1\}$ .

- What is  $\Sigma^0$ ?
- **2** How many elements are there in  $\Sigma^3$ ?
- **3** How many elements are there in  $\Sigma^n$ ?
- What is the length of the longest string in Σ? Does Σ\* have strings of infinite length?
- So If |u| = 2 and |v| = 3 then what is  $|u \cdot v|$ ?
- **(**) Let u be an arbitrary string  $\Sigma^*$ . What is  $\epsilon u$ ? What is  $u\epsilon$ ?
- Is uv = vu for every  $u, v \in \Sigma^*$ ?
- **3** Is (uv)w = u(vw) for every  $u, v, w \in \Sigma^*$ ?

### Exercise

Sariel Har-Peled (11110

Question: Is  $\Sigma^* \times \Sigma^* = \{(x, y) \mid x, y \in \Sigma^*\}$  countably infinite?

Question: Is  $\Sigma^* \times \Sigma^* \times \Sigma^* = \{(x, y, z) \mid x, y, x \in \Sigma^*\}$  countably infinite?

5374 12

Fall 2017

10 / 33

### Inductive proofs on strings

Inductive proofs on strings and related problems follow inductive definitions.

#### Definition

The reverse  $w^R$  of a string w is defined as follows:

• 
$$w^R = \epsilon$$
 if  $w = \epsilon$ 

•  $w^R = x^R a$  if w = ax for some  $a \in \Sigma$  and string x

#### Theorem

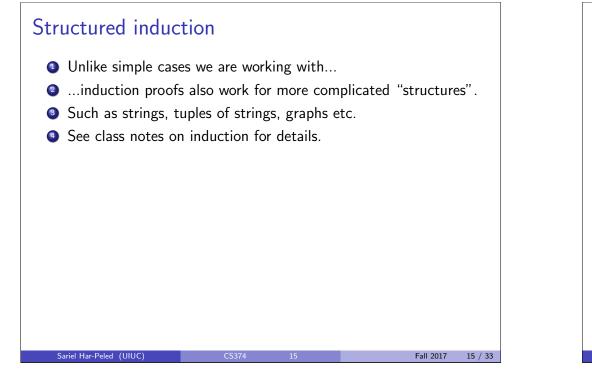
Prove that for any strings  $u, v \in \Sigma^*$ ,  $(uv)^R = v^R u^R$ .

Example: 
$$(dog \bullet cat)^R = (cat)^R \bullet (dog)^R = tacgod$$
.

Sariel Har-Peled (UIUC)

Fall 2017

13 / 3



## Principle of mathematical induction

Induction is a way to prove statements of the form  $\forall n \ge 0, P(n)$ where P(n) is a statement that holds for integer n.

Example: Prove that  $\sum_{i=0}^{n} i = n(n+1)/2$  for all n.

Induction template:

- Base case: Prove P(0)
- Induction hypothesis: Let k > 0 be an arbitrary integer. Assume that P(n) holds for any  $k \le n$ .
- Induction Step: Prove that P(n) holds, for n = k + 1.

# Proving the theorem

#### Theorem

Sariel Har-Peled (UIUC

Prove that for any strings  $u, v \in \Sigma^*$ ,  $(uv)^R = v^R u^R$ .

Proof: by induction. On what?? |uv| = |u| + |v|? |u|? |v|?

What does it mean to say "induction on |u|"?

Fall 2017

14 / 33

# By induction on **u**

#### Theorem

Prove that for any strings  $u, v \in \Sigma^*$ ,  $(uv)^R = v^R u^R$ .

Proof by induction on |u| means that we are proving the following. **Base case:** Let u be an arbitrary stirng of length 0.  $u = \epsilon$  since there is only one such string. Then

 $(uv)^R = (\epsilon v)^R = v^R = v^R \epsilon = v^R \epsilon^R = v^R u^R$ 

**Induction hypothesis:**  $\forall n \geq 0$ , for any string u of length n (for all strings  $v \in \Sigma^*$ ,  $(uv)^R = v^R u^R$ ).

Note that we did not assume anything about  $\nu$ , hence the statement holds for all  $\nu \in \Sigma^*$ .

Sariel Har-Peled (UIUC)

Fall 2017

17 / 33

### Induction on v

#### Theorem

Prove that for any strings  $u, v \in \Sigma^*$ ,  $(uv)^R = v^R u^R$ .

Proof by induction on |v| means that we are proving the following. Induction hypothesis:  $\forall n \ge 0$ , for any string v of length n (for all strings  $u \in \Sigma^*$ ,  $(uv)^R = v^R u^R$ ).

**Base case:** Let v be an arbitrary stirng of length **0**.  $v = \epsilon$  since there is only one such string. Then

$$(uv)^{R} = (u\epsilon)^{R} = u^{R} = \epsilon u^{R} = \epsilon^{R} u^{R} = v^{R} u^{R}$$

### Inductive step

- Let u be an arbitrary string of length n > 0. Assume inductive hypothesis holds for all strings w of length < n.</li>
- Since |u| = n > 0 we have u = ay for some string y with |y| < n and  $a \in \Sigma$ .
- Then

(uv) <sup>F</sup>	= = = =	$((ay)v)^{R}$ $(a(yv))^{R}$ $(yv)^{R}a^{R}$ $(v^{R}y^{R})a^{R}$ $v^{R}(y^{R}a^{R})$ $v^{R}(ay)^{R}$ $v^{R}(ay)^{R}$			
Sariel Har-Peled (UIUC)		CS374	18	Fall 2017	18 / 33

### Inductive step

- Let v be an arbitrary string of length n > 0. Assume inductive hypothesis holds for all strings w of length < n.
- Since |v| = n > 0 we have v = ay for some string y with |y| < n and  $a \in \Sigma$ .
- Then

$$(uv)^{R} = (u(ay))^{R}$$
  
=  $((ua)y)^{R}$   
=  $y^{R}(ua)^{R}$   
= ??

Cannot simplify  $(ua)^R$  using inductive hypothesi. Can simplify if we extend base case to include n = 0 and n = 1. However, n = 1 itself requires induction on |u|!

19

C 527

# Induction on $|\mathbf{u}| + |\mathbf{v}|$

#### Theorem

Prove that for any strings  $u, v \in \Sigma^*$ ,  $(uv)^R = v^R u^R$ .

Proof by induction on |u| + |v| means that we are proving the following. Induction hypothesis:  $\forall n > 0$ , for any  $u, v \in \Sigma^*$  with

induction hypothesis:  $\forall n \ge 0$ , for any  $u, v \in \Sigma^*$  with  $|u| + |v| \le n$ ,  $(uv)^R = v^R u^R$ .

**Base case:** n = 0. Let u, v be an arbitrary stirngs such that |u| + |v| = 0. Implies  $u, v = \epsilon$ .

**Inductive stepe:** n > 0. Let u, v be arbitrary strings such that |u| + |v| = n.

#### Sariel Har-Peled (UIUC)

Fall 2017

21 / 33

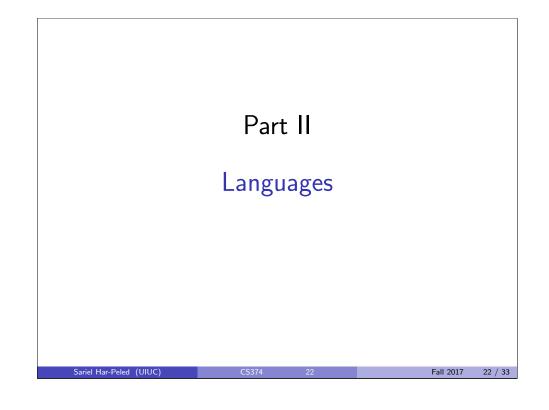
### Languages

#### Definition

A language *L* is a set of strings over  $\Sigma$ . In other words  $L \subseteq \Sigma^*$ .

Standard set operations apply to languages.

- For languages A, B the concatenation of A, B is  $AB = \{xy \mid x \in A, y \in B\}.$
- For languages A, B, their union is  $A \cup B$ , intersection is  $A \cap B$ , and difference is  $A \setminus B$  (also written as A B).
- For language  $A \subseteq \Sigma^*$  the complement of A is  $\overline{A} = \Sigma^* \setminus A$ .



### Exponentiation, Kleene star etc

#### Definition

For a language  $L \subseteq \Sigma^*$  and  $n \in \mathbb{N}$ , define  $L^n$  inductively as follows.

$$L^{n} = \begin{cases} \{\epsilon\} & \text{if } n = 0\\ L \bullet (L^{n-1}) & \text{if } n > 0 \end{cases}$$

And define 
$$L^* = \bigcup_{n>0} L^n$$
, and  $L^+ = \bigcup_{n>1} L^n$ 

24

### Exercise

### Problem

Answer the following questions taking  $A, B \subseteq \{0, 1\}^*$ .

- $Is \epsilon = \{\epsilon\}? \ Is \emptyset = \{\epsilon\}?$
- **2** What is  $\emptyset \bullet A$ ? What is  $A \bullet \emptyset$ ?
- **3** What is  $\{\epsilon\} \bullet A$ ? And  $A \bullet \{\epsilon\}$ ?
- If |A| = 2 and |B| = 3, what is  $|A \cdot B|$ ?

### Languages and Computation

What are we interested in computing? Mostly functions.

**Informal definition:** An algorithm  $\mathcal{A}$  computes a function  $f: \Sigma^* \to \Sigma^*$  if for all  $w \in \Sigma^*$  the algorithm  $\mathcal{A}$  on input w terminates in a finite number of steps and outputs f(w).

Examples of functions:

Sariel Har-Peled (IIIIIC

- Numerical functions: length, addition, multiplication, division etc
- Given graph G and s, t find shortest paths from s to t
- Given program *M* check if *M* halts on empty input
- Posts Correspondence problem

### Exercise

#### Problem

Consider languages over  $\Sigma = \{0, 1\}$ .

- What is  $\emptyset^0$ ?
- If |L| = 2, then what is  $|L^4|$ ?
- 3 What is  $\emptyset^*$ ,  $\{\epsilon\}^*$ ,  $\epsilon^*$ ?
- For what **L** is **L**\* finite?
- **(3)** What is  $\emptyset^+$ ,  $\{\epsilon\}^+$ ,  $\epsilon^+$ ?

Sariel Har-Peled (UIUC)	CS374	26	Fall

### Languages and Computation

#### Definition

A function f over  $\Sigma^*$  is a boolean if  $f: \Sigma^* \to \{0, 1\}$ .

**Observation:** There is a bijection between boolean functions and languages.

- Given boolean function  $f: \Sigma^* \to \{0, 1\}$  define language  $L_f = \{w \in \Sigma^* \mid f(w) = 1\}$
- Given language  $L \subseteq \Sigma^*$  define boolean function  $f : \Sigma^* \to \{0, 1\}$  as follows: f(w) = 1 if  $w \in L$  and f(w) = 0 otherwise.

27

Fall 2017

25 / 33

CS374

26 / 33

2017

### Language recognition problem

#### Definition

For a language  $L \subseteq \Sigma^*$  the language recognition problem associate with L is the following: given  $w \in \Sigma^*$ , is  $w \in L$ ?

- Equivalent to the problem of "computing" the function  $f_L$ .
- Language recognition is same as boolean function computation
- How difficult is a function f to compute? How difficult is the recognizing L<sub>f</sub>?

Why two different views? Helpful in understanding different aspects?

# Cantor's diagonalization argument

### Theorem (Cantor)

 $\mathbb{P}(\mathbb{N})$  is not countably infinite.

- Suppose  $\mathbb{P}(\mathbb{N})$  is countable infinite. Let  $S_1, S_2, \ldots$ , be an enumeration of all subsets of numbers.
- Let D be the following diagonal subset of numbers.

 $D = \{i \mid i \notin S_i\}$ 

- Since D is a set of numbers, by assumption,  $D = S_j$  for some j.
- Question: Is  $j \in D$ ?

### How many languages are there?

#### Recall:

#### Definition

An set A is countably infinite if there is a bijection f between the natural numbers and A.

#### Theorem

 $\Sigma^*$  is countably infinite for every finite  $\Sigma$ .

The set of all languages is  $\mathbb{P}(\Sigma^*)$  the power set of  $\Sigma^*$ 

#### Theorem (Cantor)

 $\mathbb{P}(\Sigma^*)$  is not countably infinite for any finite  $\Sigma$ .

# Consequences for Computation

- How many *C* programs are there? The set of *C* programs is countably infinite since each of them can be represented as a string over a finite alphabet.
- How many languages are there? Uncountably many!
- Hence some (in fact almost all!) languages/boolean functions do not have any *C* program to recognize them.

#### **Questions:**

- Maybe interesting languages/functions have *C* programs and hence computable. Only uninteresting langues uncomputable?
- Why should *C* programs be the definition of computability?
- Ok, there are difficult problems/languages. what lanauges are computable and which have efficient algorithms?

3

Fall 2017 31 / 33

Fall 2017

32

Fall 2017

30 / 33

# Easy languages

### Definition

A language  $L \subseteq \Sigma^*$  is finite if |L| = n for some integer n.

**Exercise:** Prove the following.

Theorem

The set of all finite languages is countably infinite.

Sariel Har-Peled (UIUC)

33

Fall 2017 33 / 33

