

CS240

# Language Theory And Automata

Fall 2011

## Introduction

- You are about to embark on the study of a fascinating and important subject: the **theory of computation**
  - Fundamental mathematical properties of computer hardware, software, and certain applications of these
  - Seek to determine
    - What can and cannot be computed
    - How quickly something can be computed
    - How much memory is needed to compute something
    - Which type of computational model can be used

*Obvious connections with engineering practice, but also purely philosophical aspects*

## Why study this stuff?

- Theoretical computer science has many fascinating big ideas, but also many small and sometimes dull details
  - The more you learn, the more interesting it becomes
  - **Our goal:** be exposed to the exciting aspects of computer theory without getting bogged down in the drudgery

## Theory is relevant to practice

- Provides conceptual tools that practitioners use in computer engineering
  - Design a new programming language for a special application - need **grammars!**
  - String searching and pattern matching - use **finite automata** and **regular expressions!**
  - Deal with a problem that seems to require more computer time than you can afford - consider **NP-completeness!**

## Theory shows the more elegant side of computers

- Usually think of computers as complicated machines
- Best computer designs and applications are conceived with elegance in mind
- A theoretical course can heighten your aesthetic sense and help you build more elegant systems

## Theory expands your mind

- Computer technology changes quickly
- Specific technical knowledge becomes outdated in a few years
- Studying theory trains you in the abilities to think, express yourself clearly and precisely, solve problems, know when you have not solved a problem
  - These abilities have lasting value
- Studying theory enables you to understand the underlying models of all computation, not just technical details that may change

## Automata, Computability, and Complexity

- Linked by the question: ***What are the fundamental capabilities and limitations of computers?***
- Each area interprets the question differently
  - **Complexity:** is a given problem easy or hard?
  - **Computability:** is a given problem solvable or unsolvable?
    - E.g., no computer can solve the problem of whether a mathematical statement is true or false
  - **Automata theory:** definitions and properties of mathematical models of computation

## Automata Theory

- We start with **automata theory**
  - Theories of computability and complexity require a precise definition of a computer
  - Automata theory allows practice with formal definitions of computation as it introduces concepts relevant to other non-theoretical areas of computer science

## Course Overview

- Study categories of languages and machines
  - **Regular languages** and **Finite automata**
  - **Content-free languages** and **Push-down automata**
  - **Unrestricted languages** and **Turing machines**
- Study solvability and efficiency
  - **The Halting problem**
  - **NP-completeness**


## Mathematical Background


- Set theory
- Relations and functions
- Recursive definitions
- Strings and languages
- Proofs

## Sets

- Group of objects represented as a unit
- May contain any type of object: numbers, symbols, other sets, ...
  - Set membership:  $\in$
  - Non-membership:  $\notin$
  - Subset:  $\subseteq$ 
    - Proper subset
  - Empty set:  $\emptyset$
  - Infinite set contains infinitely many elements
    - E.g., set of integers  $\{\dots-2, -1, 0, 1, 2, \dots\}$

## Sets

Determine the union ( $\cup$ ), intersection ( $\cap$ ), difference ( $-$ ), complement, and power set  $P(X) = 2^X$  of any given sets. 


Prove the equality of sets  $X$  and  $Y$ : show every element of  $X$  is an element of  $Y$  and vice versa (i.e., show that  $X \subseteq Y$  and  $Y \subseteq X$  to show  $X = Y$ .) 


If  $X$  is a set, the notations  $|X|$  and  $\text{card}(X)$  both refer to the number of elements in  $X$ .

## Cartesian Product and Relations

**Cartesian product** builds a set consisting of ordered pairs of elements from two or more existing sets.

$$X \times Y = \{[x, y] \mid x \in X \text{ and } y \in Y\}$$

A **binary relation** on sets  $X$  and  $Y$  is a subset of  $X \times Y$ . 

An **n-ary relation** on sets  $X_1, X_2, \dots, X_n$  is a subset of  $X_1 \times X_2 \times \dots \times X_n$ . 

## Functions

A **function** from a set  $X$  to a set  $Y$  is a mapping of elements of  $X$  to elements of  $Y$  such that each element of  $X$  maps to exactly one element of  $Y$ .

$$f : X \rightarrow Y$$

$X$  is the **domain** of  $f$ .

The **range** of  $f : X \rightarrow Y$  is the set  $\{y \in Y \mid y = f(x) \text{ for some } x \in X\}$ .

A **total function**  $f$  from  $X$  to  $Y$  is a binary relation on  $X \times Y$  such that

- i. For each  $x \in X$  there is a  $y \in Y$  such that  $[x, y] \in f$ .
- ii. If  $[x, y] \in f$  and  $[x, z] \in f$ , then  $y = z$ .


## Total Functions

$f : X \rightarrow Y$  is **one-to-one** if each element of  $X$  maps to a distinct element in the range.

$f : X \rightarrow Y$  is **onto** if the range is the entire set  $Y$ .

## Countable and Uncountable Sets

We divide sets into classes:

- **Countably infinite (denumerable):** has same number of elements as  $\mathcal{N}$ .
- **Countable:** finite or countably infinite.
- **Uncountable:** has more elements than  $\mathcal{N}$ . 

**To show a set  $A$  is countably infinite, we need to show a one-to-one function exists from  $\mathcal{N}$  to  $A$ .**

## Uncountable Sets

**Prove a set  $X$  is uncountable by showing that it is impossible to sequentially list its members**

**Cantor's diagonalization argument:** used to show that a set is uncountable. Proof by contradiction.

**Step 1:** Assume the set is countable and therefore its members can be exhaustively listed. This listing must contain all members of the set.

**Step 2:** Produce a member of the set that cannot occur anywhere in the listing, showing that such a listing cannot exist and therefore the set is uncountable.

## Recursive Definitions

Recursion provides a method for generating elements of a set by specifying:

- **basis elements** explicitly
- a finite set of **operators** which are used to construct the remaining elements of the set from the basis elements.

**Generation using a finite but unbounded number of operations is a fundamental property of recursive definitions.**

## Recursive Definitions

### Example:

A recursive definition of  $\mathcal{N}$ , the set of natural numbers using the successor function,  $s(n) = n+1$ :

1. Basis:  $0 \in \mathcal{N}$ .
2. Recursive step: If  $n \in \mathcal{N}$ , then  $s(n) \in \mathcal{N}$
3.  $n \in \mathcal{N}$  only if it can be obtained from 0 by a finite number of applications of the recursive step.

Elements are:  $0, s(0), s(s(0)), s(s(s(0))), \dots$   
 $0, 1, 2, 3, \dots$

## Languages

- **Alphabet ( $\Sigma$ )** : finite set of symbols, e.g.,
  - $\Sigma = \{0, 1\}$  (binary alphabet)
  - ASCII
- **String** : finite sequence of symbols chosen from some alphabet, e.g., 01101 or *abracadabra*.
- **Language** : set of strings chosen from some alphabet

## Examples of Languages

- The set of all binary strings consisting of some number of 0's followed by an equal number of 1's; that is,  $\epsilon$ ; 01; 0011; 000111; ...
- Java (the set of compilable Java programs).
- English.

## Strings

- **Empty string** :  $\epsilon$
- **Length** :
  - $|abcde|=5$
  - $|uv|=|u|+|v|$
- **Reverse** :  $w^R$ 
  - If  $w=abc$ ,  $w^R=cba$
- **Substring** : any string of consecutive characters in some  $w$
- **Prefix and suffix** : if  $w=vu$ ,  $v$  and  $u$  are a prefix and a suffix of  $w$ , respectively

**Convention:** Use lower case letters from the beginning of the alphabet for symbols, lower case letters from end of alphabet for strings

- **Concatenation** : Given strings  $x$  and  $y$ , concatenation is  $xy$ 
  - E.g.,  $x=abc$ ,  $y=def$ ,  $xy=abcdef$
- **Powers**
  - $\Sigma^k$  = set of all strings from alphabet  $\Sigma$  with length  $k$
  - $\Sigma^0 = \{\epsilon\}$
  - $\Sigma^*$  = set of all strings from alphabet  $\Sigma$
  - $\Sigma^+ = \Sigma^* - \{\epsilon\}$

## More About Languages

- Can concatenate languages
  - $L_1L_2 = \{xy \mid x \in L_1, y \in L_2\}$
  - $L^n = L$  concatenated with itself  $n$  times
    - $L^0 = \{\epsilon\}$ ;  $L^1 = L$
- **Star-closure**
  - $L^* = L^0 \cup L^1 \cup L^2 \cup \dots$  (note:  $L^+ = L^* - L^0$ )
- Languages can be finite or infinite
  - $L = \{a, aba, bba\}$
  - $L = \{a^n \mid n > 0\}$

### Subtle point

A language may be infinite, but there is some finite set of symbols of which all its strings are composed

## Proofs

- The only way to prove the truth or falsity of a mathematical statement is with a proof
- Finding proofs isn't always easy!
- **Helpful strategies:**
  - Carefully read the statement you want to prove
  - Rewrite the statement in your own words
  - Break the statement down and consider each part separately
  - Reduction to definitions

## Parts of proofs

- Parts of a multi-part statement are not always obvious
- Frequent multi-part statement:
  - **$P$  if and only if  $Q$**  ( $P$  iff  $Q$  or  $P \Leftrightarrow Q$ )
  - Two-part statement:
    - If  $P$  is true,  $Q$  is true ( $P \Rightarrow Q$ ) *forward direction*
    - If  $Q$  is true,  $P$  is true ( $P \Leftarrow Q$ ) *reverse direction*
  - To prove this kind of statement, must prove both directions

## Proving the Equivalence of Sets

- Many important facts in language theory are of the form that two sets of strings, described in two different ways, are really the same set. To prove sets  $S$  and  $T$  are the same, prove:
  - $x$  is in  $S$  if and only if  $x$  is in  $T$ . That is:
    - Assume  $x$  is in  $S$ ; prove  $x$  is in  $T$ .
    - Assume  $x$  is in  $T$ ; prove  $x$  is in  $S$ .

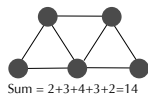
## More strategy for doing proofs

- Get an intuitive “gut” feeling of why the statement should be true
- **Experiment with examples**
  - E.g., if statement says all objects of a certain type have a particular property, pick a few objects of that type and observe that they actually have that property
  - Then, try to find an object that fails to have the property (**counterexample**)
    - If the statement actually is true, you won't be able to find a counterexample
    - **Seeing where you run into trouble when you try to find a counterexample can help understand why the statement is true**

## Example

**Prove:** For every graph  $G$ , the sum of the degrees of all the nodes is an even number

- Step 1: Pick a few graphs and observe the statement in action



- Step 2: Try to find a counterexample



Sum =  $1+1=2$   
 Sum =  $1+2+1=4$   
 Sum =  $1+3+1+1=6$   
**Sum always increases by 2!**

## Still stuck?

- Attempt to prove a special case of the statement

### Example

Trying to prove some property is true for every  $k > 0$

- Try to prove for  $k = 1$
- If you succeed, try for  $k = 2$ , etc. until you understand the more general case
- If not, try another special case

## Reduction to Definitions

- Convert all terms in the hypothesis to their definitions

- Example

Use the following two definitions:

- A set  $S$  is **finite** if there exists an integer  $n$  such that  $S$  has exactly  $n$  elements. We write  $|S| = n$ , where  $|S|$  is used to denote the number of elements of  $S$ . If the set  $S$  is not finite, we say it is **infinite**. Intuitively, an infinite set is a set that contains more than any integer elements.
- If  $S$  and  $T$  are both subsets of some set  $U$ , then  $T$  is the **complement** of  $S$  (with respect to  $U$ ) if  $S \cup T = U$  and  $S \cap T = \emptyset$ . That is, each element of  $U$  is in exactly one of  $S$  and  $T$ ; put another way,  $T$  consists of exactly those elements of  $U$  that are not in  $S$ .

## Using reduction to definitions

### Claim

- Let  $S$  be a finite subset of some infinite set  $U$ . Let  $T$  be the complement of  $S$  with respect to  $U$ . Then  $T$  is infinite.

### Proof

- Intuitively, the theorem says that if you have an infinite supply of something ( $U$ ), and you take away a finite amount ( $S$ ), then you still have an infinite amount left ( $T$ ).
- Restate facts of the theorem using definitions 1 and 2:

Original Statement	New Statement
$S$ is finite	There is an integer $n$ such that $ S  = n$
$U$ is infinite	For no integer $p$ is $ U  = p$
$T$ is the complement of $S$	$S \cup T = U$ and $S \cap T = \emptyset$

## Proof by Contradiction

- Assume the conclusion is false
- Use that assumption plus parts of the hypothesis to prove the opposite of one of the given statements of the hypothesis
- Impossible for all parts of the hypothesis to be true and the conclusion to be false
- Only possibility is for conclusion to be true when hypothesis is true

## Proof for our example

- Contradiction of conclusion is  **$T$  is finite**
- Assume  $T$  is finite,  $S$  is also finite, so using definitions we have
  - There is an integer  $m$  such that  $|T| = m$
  - There is an integer  $n$  such that  $|S| = n$
- We also know that the elements of  $U$  are exactly the elements of  $S$  and  $T$  by  $S \cup T = U$
- Thus, there must be  $n + m$  elements of  $U$ . Since  $n + m$  is an integer, and we have shown that  $|U| = n + m$ , it follows that  $U$  is finite (because that is the definition of finite). But the statement that  $U$  is finite contradicts the given statement that  $U$  is infinite.
- Thus use the contradiction of the conclusion to prove the contradiction of one of the given statements, and we may conclude that the theorem is true.

## Writing up the proof

- A well-written proof is a sequence of statements, wherein each one follows by simple reasoning from previous statements
- Carefully writing a proof :
  - Enables reader to understand
  - Assures you the proof is free from errors

## A Few Tips

- **Be patient**
  - Finding proofs takes time. Don't worry if you don't see how to do it right away.
- **Come back to it**
  - Let the unconscious, intuitive part of your brain have a chance to work
- **Be neat**
  - Use simple, clear pictures and text. Neatness helps both you and the person who has to read your proof (me!) understand it
- **Be concise**
  - Brevity helps express high-level ideas without getting lost in details. Use good mathematical notation. But be sure to include enough of your reasoning to make the argument clear.

## Types of Proof

- **Proof by construction**
  - Theorem states that a particular type of object exists
  - Prove by demonstrating how to construct such an object
- **Proof by contradiction**
  - Assume the theorem is false and show this leads to an obviously false consequence
  - Do this in everyday life:
    - Upon seeing someone who has just come in from outdoors and is completely dry, we know it is not raining. “Proof” is that *if it were raining* (the assumption that the statement is false), *the person who just came in would be wet* (the obviously false consequence). Therefore it must not be raining.
- **Proof by Induction**
  - Truth of a number of statements inferred from the truth of a few specific instances.

## Proof by Construction

- **Example**
  - Prove every pair of integers has a greatest common divisor by showing an algorithm to find the gcd
  - Show not only that the gcd exists, but also a method to determine the gcd for every pair of integers.
  - *This is a stronger claim than we started with, but in some cases stronger claims are easier to prove.*

## Proof by Contradiction

### Example

- Claim:  $\sqrt{2}$  is irrational.
- Prove by assuming  $\sqrt{2}$  is rational. In that case, it is the quotient of two integers,  $i$  and  $j$ . So we have
 
$$\sqrt{2} = \frac{i}{j}$$
- If  $i$  and  $j$  have any common factors, we reduce them by those factors. So now we have
 
$$\sqrt{2} = \frac{k}{n}$$
 where  $k$  and  $n$  have no common factors
 
$$2n^2 = k^2$$
- Since 2 is a factor of  $k^2$ ,  $k^2$  must be even and so  $k$  is even. Since  $k$  is even, we can rewrite it as  $2m$  for some integer  $m$ . Substituting  $2m$  for  $k$  we get:
 
$$2n^2 = (2m)^2$$

$$2n^2 = 4m^2$$

$$n^2 = 2m^2$$
- So  $n^2$  is even and thus  $n$  is even. Now both  $k$  and  $n$  are even and so have 2 as a common factor. But we had reduced them until they had no common factors. The assumption that  $\sqrt{2}$  is rational has led to a contradiction. Therefore  $\sqrt{2}$  cannot be rational.

## Proof by Induction

- Advanced method to show that all elements of an **infinite set** have a specified property
- **Two parts:**
  - **Each part is an individual proof on its own**
  - **Induction step** : Prove that for each  $i \geq 1$ , if  $\mathcal{P}(i)$  is true, then so is  $\mathcal{P}(i+1)$
  - **Basis**: Proves that  $\mathcal{P}(1)$  is true
- When both parts are proven the desired result follows, that  $\mathcal{P}(i)$  is true for each  $i$ .
  - **WHY?**
    - We know that  $\mathcal{P}(1)$  is true because the basis alone proves it.
    - We know that  $\mathcal{P}(2)$  is true because the induction step proves that, if  $\mathcal{P}(1)$  is true then  $\mathcal{P}(2)$  is true, and we already know  $\mathcal{P}(1)$  is true.
    - We know that  $\mathcal{P}(3)$  is true because the induction step proves that, if  $\mathcal{P}(2)$  is true then  $\mathcal{P}(3)$  is true, and we already know  $\mathcal{P}(2)$  is true.
    - and so on....

## Example

- A binary tree with  $n$  leaves has  $2n - 1$  nodes
  - Formally,  $S(T)$ : if  $T$  is a binary tree with  $n$  leaves, then  $T$  has  $2n - 1$  nodes.
  - **Basis:** If  $T$  has 1 leaf, it is a one-node tree.  $2(1) - 1 = 1$ .
  - **Induction:** Assume  $S(U)$  for trees with fewer nodes than  $T$ . In particular, assume for the subtrees of  $T$ .
    - $T$  must be a root plus two subtrees  $U$  and  $V$ .
    - If  $U$  and  $V$  have  $u$  and  $v$  leaves, respectively, and  $T$  has  $t$  leaves, then  $u + v = t$ .
    - By the inductive hypothesis,  $U$  and  $V$  have  $2u - 1$  and  $2v - 1$  nodes, respectively.
    - Then  $T$  has  $1 + (2u - 1) + (2v - 1)$  nodes
    - $= 2(u + v) - 1$
    - $= 2t - 1$ , proving the inductive step.

## Example

- The sum of the first  $n$  odd integers is  $n^2$ .
  - Check plausibility:
 

$(n = 1) 1$	$= 1 = 1^2$
$(n = 2) 1 + 3$	$= 4 = 2^2$
$(n = 3) 1 + 3 + 5$	$= 9 = 3^2$
$(n = 4) 1 + 3 + 5 + 7 = 16 = 4^2$ , etc.	
  - The claim appears to be true, so we should prove it. Let  $Odd_i = 2(i - 1) + 1$  denote the  $i^{\text{th}}$  odd integer. Then:

$$\text{For all } n \geq 1, \sum_{i=1}^n Odd_i = n^2$$

- Formal proof by induction on  $n$ :
  - **Basis:** Take 1 as the base case,  $1 = 1^2$ .
  - **Induction:** Prove that

$$\text{For all } n \geq 1, \sum_{i=1}^n Odd_i = n^2 \text{ implies } \sum_{i=1}^{n+1} Odd_i = (n+1)^2$$

## Example con't

- Observe that the sum of the first  $n + 1$  odd integers is the sum of the first  $n$  of them plus the  $n + 1^{\text{st}}$ , so:

$$\begin{aligned} \sum_{i=1}^{n+1} Odd_i &= \sum_{i=1}^n Odd_i + Odd_{n+1} \\ &= n^2 + Odd_{n+1} \text{ (using the induction hypothesis)} \\ &= n^2 + 2n + 1 \text{ (since } Odd_{n+1} \text{ is } 2(n + 1 - 1) + 1 = 2n + 1) \\ &= (n + 1)^2 \end{aligned}$$

- This proves the induction hypothesis.

APPENDIX  
APPENDIX  
APPENDIX



- Only-If

- An induction on  $|w|$ . Assume  $w$  is GB; prove it is SB.
- Basis:  $w = \epsilon$ . Clearly  $w$  obeys the conditions for being SB.
- Induction: Assume "GB implies SB" for strings shorter than  $w$ , and assume  $w$  is not  $\epsilon$ .

- Case 1:

- $w$  is GB because of rule 1(b); i.e.,  $w = (x)$  and  $x$  is GB.
  - by the IH,  $x$  is SB.
  - Since  $x$  has equal numbers of '('s and ')'s, so does  $(x)$ .
  - Since  $x$  has no prefix with more '('s than ')'s, so does  $(x)$ .

- Case 2:

- $w$  is not  $\epsilon$  and is GB because of rule (c); i.e.,  $w = xy$ , and  $x$  and  $y$  are GB.
  - By the IH,  $x$  and  $y$  are SB.
    - (Aside) Trickier than it looks: we have to argue that neither  $x$  nor  $y$  could be  $\epsilon$ , because if one were, the other would be  $w$ , and this rule application could not be the one that first shows  $w$  to be GB.
  - $xy$  has equal numbers of '('s and ')'s because  $x$  and  $y$  both do.
  - If  $w$  had a prefix with more ')'s than '('s, that prefix would either be a prefix of  $x$  (contradicting the fact that  $x$  has no such prefix) or it would be  $x$  followed by a prefix of  $y$  (contradicting the fact that  $y$  also has no such prefix).

*Above is an example of **proof by contradiction**. We assumed our conclusion about  $w$  was false and showed it would imply something that we know is false.*