Administrivia

1 Staff, and Office Hours

Instructional Staff

- *Instructor:*
  - Mahesh Viswanathan (`vmahesh`)
- *Teaching Assistants:*
  - Santosh Prabhu (`prabhum2`)
  - Matt Wala (`wala1`)
  - Chao Xu (`chaoxu3`)
- *Office Hours:* See course webpage

---

2 Resources

Electronic Bulletin Boards

- *Webpage:* `courses.engr.illinois.edu/cs373`
- *Newsgroup:* We will use Piazza. Sign up at `piazza.com/illinois/fall2013/cs373`. Piazza discussion page is `piazza.com/illinois/fall2013/cs373/home`

Resources for class material

- *Prerequisites:* All material in CS 173, and CS 225
- *Lecture Notes:* Available on the web-page
- *Additional References*
  - Introduction to the Theory of Computation: Michael Sipser
  - Introduction to Automata Theory, Languages, and Computation: Hopcroft, and Ullman
  - Introduction to Automata Theory, Languages, and Computation: Hopcroft, Motwani, and Ullman
  - Elements of the Theory of Computation: Lewis, and Papadimitriou

---

3 Grading Scheme

Grading Policy: Overview
Total Grade and Weight

- *Homeworks:* 20%
- *Quizzes:* 10%
- *Midterms:* 40% (2×20)
- *Finals:* 30%

Homeworks

- One homework every week: Assigned on Thursday and due the following Thursday (midnight in homework drop boxes)
- *No late homeworks.* Lowest two homework scores will be dropped.
- Homeworks may be solved in groups of size at most 3 except homework 1.
- Homework 1 will be solved online.
- For the other homeworks, read Homework Guidelines on course website.

Quizzes

- The day before every class on Moodle.
- About 25 to 26 in total.
- We will drop the 5 lowest scores.

Examinations

- First Midterm: October 3, 7pm to 8:30pm
- Second Midterm: October 31, 7pm to 8:30pm
- Final Exam: December 18, 7pm to 10pm
- Midterms will only test material since the previous exam
- Final Exam will test *all* the course material

Course Overview

4 Computation

Objectives

Understand the nature of computation in a manner that is independent of our understanding of physical laws (or of the laws themselves)

- Its a fundamental scientific question
- Provides the foundation for the science of computationally solving problems

---

Problems through the Computational Lens

Mathematical problems look fundamentally different when viewed through the computational lens

- Not all problems equally easy to solve — some will take longer or use more memory, *no matter how clever you are*
- Not all problems can be solved!
- The “complexity” of the problem influences the nature of the solution
  - May explore alternate notions of “solving” like approximate solutions, “probabilistically correct” solutions, partial solutions, etc.

---

5 Overview

Course Overview

The three main computational models/problem classes in the course

Computational Model	Applications
Finite State Machines/ Regular Expressions	text processing, lexical analysis, protocol verification
Pushdown Automata/ Context-free Grammars	compiler parsing, software modeling, natural language processing
Turing machines	undecidability, computational complexity, cryptography

---

6 Skills

Skills

- Comprehend mathematical definitions
- Write mathematical definitions
- Comprehend mathematical proofs
- Write mathematical proofs

Mathematics Background

7 Sets, Functions, and Relations

Sets

Sets
A *set* is a (unordered) collection of objects *without repetition*. The objects in the set are called *elements/members*. Sets can be described formally

- By listing the elements inside braces, e.g. $\{3, 7, 10\}$
- Using the set builder notation, like $\{w \mid p(w)\}$ where $p(\cdot)$ is a predicate. For example, $\{n \in \mathbb{N} \mid n \bmod 2 = 0\}$ is the set of all even natural numbers.

We will denote: the set of natural numbers by $\mathbb{N}$ ($0 \in \mathbb{N}$); the empty set $\emptyset$.

A set A is *finite* if it has finitely many elements. A is an *infinite set* if it is not finite. For example $\mathbb{N}$ is an infinite set. The *cardinality* of a set A is the number of elements in A, and we denote that by $|A|$.

A is a *subset* of B (denoted $A \subseteq B$) if every element of A is also an element of B. A is a *proper subset* of B (denoted $A \subsetneq B$) if $A \subseteq B$ and $A \neq B$.

Operations on Sets
Given sets A and B subsets of a universe U, we can define the following operations

union $A \cup B = \{w \in U \mid w \in A \text{ or } w \in B\}$

intersection $A \cap B = \{w \in U \mid w \in A \text{ and } w \in B\}$

difference $A \setminus B = \{w \in U \mid w \in A \text{ and } w \notin B\}$

complement $\overline{A} = \{w \in U \mid w \notin A\}$

powerset $\mathcal{P}(A) = \{K \subseteq U \mid K \subseteq A\}$

Sequences and Tuples

- A *sequence* is a ordered list of elements. For example, the sequence 7,2,3,3 is different than 2,7,3,3 and 7,2,3. Sequences maybe finite or infinite.
- A *tuple* is a finite sequence. A k-tuple has k elements. A *pair* is a 2-tuple.
- For sets A, B, the *Cartesian product* of A and B, denoted $A \times B$, is the set of all pairs where the first element belongs to A and the second element belongs to B.
- For sets $A_1, \ldots A_k$, the set $A_1 \times A_2 \times \cdots \times A_k$ is the collection of all k-tuples where the ith element is a member of A_i.

Functions and Relations

Functions
A *function* $f : A \to B$ maps each element of A to some element of B; A is said to be the *domain* of f and B is the *co-domain.* The *range* of f is the set $\{b \in B \mid \exists a \in A.\ f(a) = b\}$. A function $f : A \to B$ is said to be *onto* if the range of f is B. f is *1-to-1* iff $f(x) = f(y)$ implies that $x = y$. If f is 1-to-1 and onto then it is said to be *bijective.*

When the domain of function f is a set of the form $A_1 \times A_2 \times \cdots \times A_k$ then it called a k-ary function.

Relations
A *k-ary relation* on A is a $R \subseteq A \times A \times A$, i.e., it is a set of k-tuples all of whose elements are members of A. A 2-ary relation is called *binary relation.* A binary relation $R \subseteq A \times A$ is

- *reflexive* if for every $a \in A$, $(a, a) \in R$,
- *symmetric* if for every $a, b \in A$, $(a, b) \in R$ implies $(b, a) \in R$.
- *transitive* if for every $a, b, c \in A$, $(a, b) \in R$ and $(b, c) \in R$ implies $(a, c) \in R$.
- *equivalence* if R is reflexive, symmetric, and transitive.

7.1 Alphabets, Strings and Languages

Alphabet

Definition 1. An *alphabet* is any finite, non-empty set of symbols. We will usually denote it by Σ.

Example 2. Examples of alphabets include $\{0, 1\}$ (binary alphabet); $\{a, b, \ldots, z\}$ (English alphabet); the set of all ASCII characters; {moveforward, moveback, rotate90}.

Strings

Definition 3. A *string* or *word* over alphabet Σ is a (finite) sequence of symbols in Σ. Examples are '0101001', 'string', '$\langle$moveback$\rangle\langle$rotate90$\rangle$'

- ϵ is the *empty string*.
- The *length* of string u (denoted by $|u|$) is the number of symbols in u. Example, $|\epsilon| = 0$, $|011010| = 6$.
- *Concatenation:* uv is the string that has a copy of u followed by a copy of v. Example, if $u = $ 'cat' and $v = $ 'nap' then $uv = $ '$catnap'$. If $v = \epsilon$ the $uv = vu = u$.
- u is a prefix of v if there is a string w such that $v = uw$. Example 'cat' is a prefix of '$catnap'$.

Languages

Definition 4.
- For alphabet Σ, Σ^* is the set of all strings over Σ. Σ^n is the set of all strings of length n.
- A *language* over Σ is a set $L \subseteq \Sigma^*$. For example $L = \{1, 01, 11, 001\}$ is a language over $\{0, 1\}$.

---

8 Proofs

8.1 Induction Proofs

Induction Principle

- Infinite sequence of statements $S_0, S_1, \ldots$
- *Goal:* Prove $\forall i.\ S_i$ is true
- Prove S_0 is true *[Base Case]*
- For an arbitrary i, assuming S_j is true for all $j < i$ *[Induction Hypothesis]*, establishes S_i to be true *[Induction Step]*.
- Conclude $\forall i.\ S_i$ is true.

Why does induction work?

- Assume S_0 is true (Base case holds), and for any i, assuming S_j is true for all $j < i$, we can conclude S_i is true (Induction step holds).

- Suppose (for contradiction) S_i does not hold for some i.
- Let k be the smallest i such that S_i does not hold. Existence of such a smallest k is a consequence of a property of natural numbers that any non-empty set of natural numbers has a smallest element in it *(Well-ordering principle)*.
- That means for all $j < k$, S_j holds.
- Then by the induction step, S_k holds! Contradiction, establishing that S_i holds for all i.

Example

Proposition 5. *Prove that the sum of the first k odd numbers is kth square. That is, for all k,* $\sum_{i=1}^{k}(2i-1) = k^2$.

Proof. The result can be proved by induction on k.

Base Case Consider the case when $k = 1$. Then $\sum_{i=1}^{k}(2i-1) = 2.1 - 1 = 1 = 1^2$. This proves the base case.

Ind. Hyp. Assume that for all $k < k_0$, $\sum_{i=1}^{k}(2i-1) = k^2$.

Ind. Step Consider $k = k_0$. Then we have,

$$\begin{aligned}
\textstyle\sum_{i=1}^{k_0}(2i-1) &= \textstyle\sum_{i=1}^{k_0-1}(2i-1) + (2k_0-1) \\
&= (k_0-1)^2 + (2k_0-1) && \text{by ind. hyp.} \\
&= (k_0^2 - 2k_0 + 1) + (2k_0 - 1) \\
&= k_0^2
\end{aligned}$$

Thus, the induction step is established.

□
