CS311: Computational Theory

Lecture 1: Introduction – Ch 0

Dr. Manal Helal, Spring 2014.

http://moodle.manalhelal.com

Course Description

This course introduces the fundamental mathematical models of computation. The course presents both inherent capabilities and limitations of these computational models as well as their relationships with formal languages. Topics to be covered include: Finite automata and regular languages, Deterministic and nondeterministic computations, Context-free grammars, languages, and pushdown automata, Turing machines, recursive and recursively enumerable sets, undesirability, introduction to computability and complexity theory.

Course learning objectives

1. *Computability: Understand the capabilities and limitation of computational models.*
2. *Hardness: Classify problems according to their computational difficulty.*
3. *Solvability: Distinguish solvable problem from unsolvable ones*

Course Outline

1. *Introduction - Ch 0*
2. *Regular Languages - Ch 1*
3. *Context-Free Languages - Ch 2*
4. *The Church–Turing Thesis - Ch 3*
5. *Decidability - Ch 4*
6. *Reducibility - Ch 5*
7. *Midterm Exam*
8. *Recursion, Decidability, Reducability, Information Theory, Ch 6*
9. *Time Complexity - Ch7*
10. *Space Complexity - Ch8*
11. *Intractability - Ch9*
12. *Case Studies Presentations*
13. *Advanced Topics - Ch10*
14. *Spare/Revision*
15. *Spare/Revision*

Grading Scheme

20% - Midterm Exam – Week 7

15% - Lecture & Lab Quizzes

5% Section Submissions

10% - Assignments

10% - Case Study– Week 12

40% - Final Exam – will be announced

Grading scale

A+ = 95%, ∝) A = [90%, 95%)
A- = [85%, 90%) B+ = [80%, 85%)
B = [75%, 80%) B- = [70%, 75%)
C= [65%, 70%) C = [60%, 65%)
C-= [55%, 60%) D = [50%, 55%)

Textbook & References

Michael Sipser, Introduction to the Theory of Computation, 2nd or 3rd edition, Course technology, 2005 or 2013.

References

John E. Hopcroft, Rajeev Motwani, and Jeffrey D. Ullman, Introduction to automata theory, languages, and computation, 3rd Edition, Addison-Wesley, 2006.
Suprakash Datta's slides (http://www.eecs.yorku.ca/course_archive/2012-13/F/2001/)

Rules

All communications through: http://moodle.manalhelal.com/course/view.php?id=7, please subscribe today!

Attendance

Attendance is a must in all CCIT courses. A consecutive 3 absences will result in course forced withdrawal. 2 sections/labs count for 1 lecture. For example, a student absent for 2 lectures & 3 labs will be withdrawn. Medical and other excuses should be submitted to the department.

Submissions

Assignments and all graded activities are given codes, such as: ass1, ass2, proj1, exer1, … etc, and announced allowed submission file extensions, and due dates. All submissions should be done electronically using moodle website. Files submitted should be named "code_StudentID.ext", where code is the graded activity code, StudentID is your numerical AASTMT student ID, and ext is the announced allowed extension for each graded activity. If assignment 1 is coded as "ass1" and the allowed file extension is pdf, and your ID is 111238090, then the submitted file name should be: "ass1_111238090.pdf". Due dates are final and there is a 10% reduction in the earned grade for each late day after the due date. After 5 days of the due date, no submissions are accepted, and model answer will be published on the website.

Academic Honesty

First academic honesty violation will result in a disciplinary action ranging from zero mark in the graded activity and up to forced course withdrawal or forced academy dismissal, as regulated by AASTMT policies. This includes copied assignments/projects/papers, exam cheating of all types, inadequate individual participation in teamwork – more on course Description Document, and College and Academy Hand-Books.

Studying Plan & Teaching Method

- *Every lecture is followed with exercise problems to be attempted in the Section and uploaded on moodle. 5% of the total course marks are for these section submissions. The section exercises will help deepen your understanding and students are expected to do their best to attempt them independently.*
- *There are 4 assignments that will vary from analysis, design and programming requirements.*
- *Asking questions are encouraged in this preference order:*
  - *In Moodle to have the question and the answer available to everyone in written form to get back to while studying.*
  - *In office hours,*
  - *then finally in lecture and section times to avoid lengthy interruptions and delay in course contents.*
  - *Please don't accumulate material without full understanding and use the lecturer and the TA as much as you can to do your best.*
- *Understanding theoretical concepts in lectures, attempting and submitting all section problems, doing all assignments, and engaging in a good case study, are the methods to study for the lecture/lab quizzes, midterm and final exams.*

Lecture Learning Objectives

1. *Introduce Automata, Computability, and Complexity*
2. *Understand the Mathematical Notions and Terminology*

Course Objectives - 1

Reasoning about computation:

- *Different computation models*
  - *Finite Automata*
  - *Pushdown Automata*
  - *Turing Machines*
- *What these models can and cannot do*

Course Objectives - 2

- *What does it mean to say "there does not exist an algorithm for this problem"?*
  - *Reason about the hardness of problems*
  - *Eventually, build up a hierarchy of problems based on their hardness.*

Course Objectives - 3

- *We are concerned with solvability, NOT efficiency.*
- *CS 312 (Computer Algorithms) efficiency issues.*

Reasoning about Computation

- *Computational problems may be*
  - *Solvable, quickly*
  - *Solvable in principle, but takes an infeasible amount of time (e.g. thousands of years on the fastest computers available)*
  - *(provably) not solvable*

Theory of Computation: parts

- *Computational problems may be Solvable, quickly*
- *Solvable in principle, but takes an infeasible amount of time (e.g. thousands of years on the fastest computers available)*
- *(provably) not solvable*

Reasoning about Computation - 2

- *Need formal reasoning to make credible conclusions*
- *Mathematics is the language developed for formal reasoning*
- *As far as possible, we want our reasoning to be intuitive*

Next:

- *Ch. 0:Set notation and languages*
  - *Sets and sequences*
  - *Tuples*
  - *Functions and relations*
  - *Graphs*
  - *Boolean logic: ∨ ∧ ¬ ⇔⇒*
- *Review of proof techniques*
  - *Construction, Contradiction, Induction*

Topics you should know:

- *Elementary set theory*
- *Elementary logic*
- *Functions*
- *Graphs*

Set Theory Review

- *Definition*
- *Notation:* $A=\{x \mid x \in N, x \bmod 3 = 1\}$

  $N = \{1,2,3,\ldots\}$
- *Union:* $A \cup B$
- *Intersection:* $A \cap B$
- *Complement:* $\bar{A}$
- *Cardinality:* $|A|$
- *Cartesian Product:*

  $A \times B = \{ (x,y) \mid x \in A \text{ and } y \in B\}$

Any order, and repetition is alright in sets, but not in sequences or tuples

Some Examples

- $L_{<6} = \{ x \mid x \in N , x<6 \}$
- $L_{prime} = \{x \mid x \in N, x \text{ is prime}\}$
- $L_{<6} \cap L_{prime} = \{2,3,5\}$
- $\sum = \{0,1\}$
- $\sum \times \sum = \{(0,0), (0,1), (1,0), (1,1)\}$
- Formal: $A \cap B = \{ x \mid x \in A \text{ and } x \in B\}$

Power Set

- *"Set of all subsets"*
- *Formal: $P(A) = \{ S \mid S \subseteq A \}$*
- *Example: $A = \{x,y\}$*
- *$P(A) = \{ \{\} , \{x\} , \{y\} , \{x,y\} \}$*
- *Power set*
- *Note the different sizes: for finite sets*
- *$|P(A)| = 2^{|A|}$*
- *$|A \times A| = |A|^2$*

Functions: Review

- $f:A \rightarrow C$
- $f:A \times B \rightarrow C$

$f : D \rightarrow R$

Where D is Domain of input values, and R is Range of Output.

Examples:

- $f:N \rightarrow N, f(x)=2x$
- $f:N \times N \rightarrow N, f(x,y)=x+y$
- $f:A \times B \rightarrow A, A=\{a,b\}, B=\{0,1\}$

	0	1
a	a	b
b	b	a

Functions: An Alternate View

- *Functions as lists of pairs or k-tuples*
- *E.g. f(x) = 2x*

{(1,2), (2,4), (3,6),....}

- *For the function in the table:*

{(a,0,a),(a,1,b),(b,0,b),(b,1,a)}

	0	1
a	a	b
b	b	a

Example 1

- $f(n) \rightarrow (n+1) \bmod 5$

N	F(n)
0	1
1	2
2	3
3	4
4	0

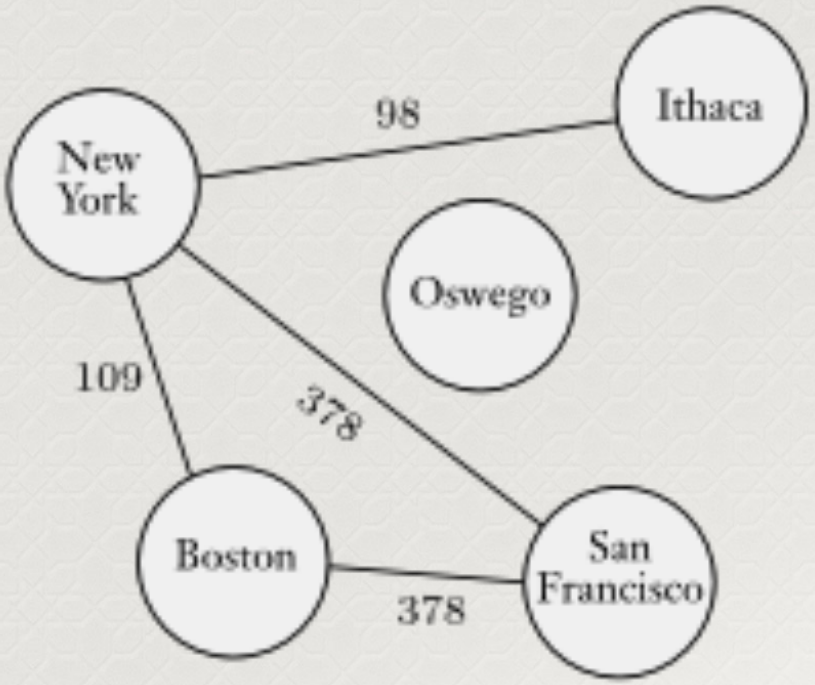
Example 2

- $g(i, j) \rightarrow (i+j) \bmod 4$

g	0	1	2	3
0	0	1	2	3
1	1	2	3	0
2	2	3	0	1
3	3	0	1	2

Graphs: review

- *Nodes, edges, weights*
- *Undirected, directed*
- *Cycles, trees*
- *Connected*



Next: Terminology

- *Alphabets*
- *Strings*
- *Languages*
- *Problems, decision problems*

Alphabets

- *An alphabet is a finite non-empty set.*
- *An alphabet is generally denoted by the symbols Σ, Γ.*
- *Elements of Σ, called symbols, are often denoted by lowercase letters, e.g., a,b,x,y,..*

Strings (or words)

- *Defined over an alphabet Σ*
- *Is a finite sequence of symbols from Σ*
- *Length of string w (|w|) – length of sequence*
- *ε– the empty string is the unique string with zero length.*
- *Concatenation of w1 and w2 – copy of w1 followed by copy of w2*
- *x^k =xxxxx...x(k times)*
- *w^R- reversed string; e.g. if w =abcd then w^R = dcba.*
- *Lexicographic ordering (dictionary order): is a generalization of the way the alphabetical order of words is based on the alphabetical order of their component letters.*

Languages

- *A language over Σ is a set of strings over Σ*
- *Σ^* is the set of all strings over Σ*
- *A language L over Σ is a subset of Σ^* ($L \subseteq \Sigma^*$)*
- *Typical examples:*
  - *$\Sigma=\{0,1\}$, the possible words over Σ are the finite bit strings.*
  - *$L = \{ x \mid x$ is a bit string with two zeros $\}$*
  - *$L=\{a^n b^n \mid n \in N\}$*
  - *$L = \{1^n \mid n$ is prime$\}$*

Concatenation of Languages

- *Concatenation of two langauges: $A \bullet B = \{ xy \mid x \in A \text{ and } y \in B \}$*
- *Caveat: Do not confuse the concatenation of languages with the Cartesian product of sets.*
- *For example, let $A = \{0,00\}$ then*
- *$A \bullet A = \{ 00, 000, 0000 \}$ with $|A \bullet A| = 3$,*
- *$A \times A = \{ (0,0), (0,00), (00,0), (00,00) \}$ with $|A \times A| = 4$*

Problems and Languages

- *Problem: defined using input and output*
- *– compute the shortest path in a graph*
- *– sorting a list of numbers*
- *– finding the mean of a set of numbers.*
- *Decision Problem: output is yes/no (or 1/0)*
- *Language: set of all inputs where output is yes*

Historical Perspective

- *Many models of computation from different fields*
- *– Mathematical logic*
- *– Linguistics*
- *– Theory of Computation*

Formal language theory

Logic: Review

- *Boolean logic:* $\vee \wedge \neg$
- *Quantifiers:* $\forall, \exists$
- *statement: Suppose* $x \in N, y \in N$. *Then* $\forall x \exists y$ *such that* $y > x$
- *for any integer, there exists a larger integer*
- $a \Rightarrow b$ *(is logically equivalent to)* $\neg a \vee b$
- $a \Leftrightarrow b$ *is logically equivalent to* $(a \Rightarrow b) \wedge (b \Rightarrow a)$

Logic: Review - 2

- *Contrapositive and converse:*
  - *the converse of $a \Rightarrow b$ is $b \Rightarrow a$*
  - *the contrapositive of $a \Rightarrow b$ is $\neg b \Rightarrow \neg a$*
  - *Any statement is logically equivalent to its contrapositive, but not to its converse.*

Example 1

Statement	If two angles are congruent, then they have the same measure.
Converse	If two angles have the same measure, then they are congruent.
Inverse	If two angles are not congruent, then they do not have the same measure.
Contrapositive	If two angles do not have the same measure, then they are not congruent.

Example 2

Statement	If a quadrilateral is a rectangle, then it has two pairs of parallel sides.
Converse	If a quadrilateral has two pairs of parallel sides, then it is a rectangle. (FALSE!)
Inverse	If a quadrilateral is not a rectangle, then it does not have two pairs of parallel sides. (FALSE!)
Contrapositive	If a quadrilateral does not have two pairs of parallel sides, then it is not a rectangle.

Logic: Review - 3

- *Negation of statements*
- *¬(∀x ∃y y > x) "=" ∃x ∀y y ≤ x*

(LHS: negation of "for any integer, there exists a larger integer", RHS: there exists a largest integer)

- *TRY: ¬ (a ⇒ b) = ?*

Logic: Review - 4

- *Understand quantifiers*
- *$\forall x \exists y\ P(y, x)$ is not the same as*

$\exists y \forall x\ P(y, x)$

- *Consider P(y,x): $x \leq y$.*
  - *$\forall x \exists y\ x \leq y$ is TRUE over N (set y = x + 1)*
  - *$\exists x \forall y\ x \leq y$ is FALSE over N (there is no largest number in N)*

Input/Output vs. Decision Problems

- *Input/output problem: "find the mean of n integers"*
- *Decision Problem: output is either yes or no*
  - *"Is the mean of the n numbers equal to k ?"*
  - *You can solve the decision problem if and only if you can solve the input/output problem.*

Example – Code Reachability

- *Code Reachability Problem:*
- *– Input: Java computer code*
- *– Output: Lines of unreachable code.*
- *Code Reachability Decision Problem:*
- *– Input: Java computer code and line number*
- *– Output: Yes, if the line is reachable for some input, no otherwise.*
- *Code Reachability Language:*
- *– Set of strings that denote Java code and reachable line.*

Example – String Length

- *Decision Problem:*
- *– Input: String w*
- *– Output: Yes, if |w| is even*
- *Language:*
- *– Set of all strings of even length.*

Relationship to Functions

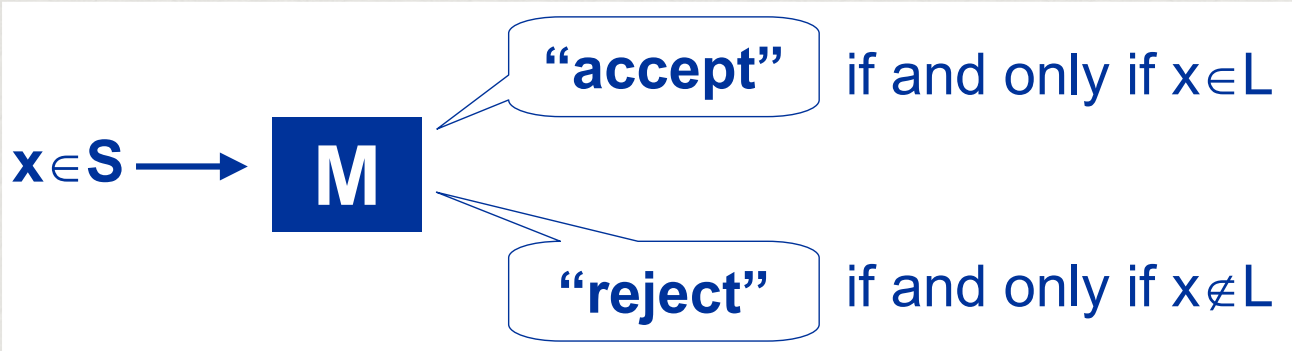
- *Use the set of k-tuples view of functions from before.*
- *A function is a set of k-tuples (words) and therefore a language.*
- *Shortest paths in graphs – the set of shortest paths is a set of paths (words) and therefore a language.*

Recognizing Languages

- *Automata/Machines accept languages.*
- *Also called “recognizing languages”.*
- *The power of a computing model is related to, and described by, the languages it accepts/recognizes.*
- *Tool for studying different models*

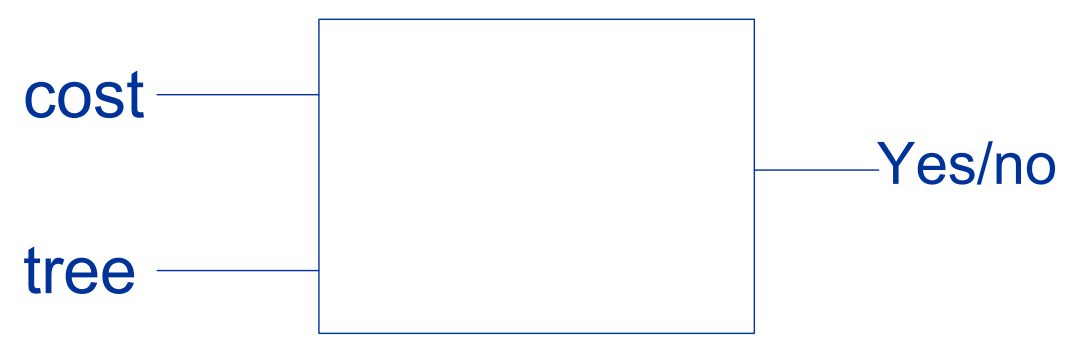
Recognizing Languages - 2

- *Let L be a language ⊆ S*
- *a machine M recognizes L if*



Recognizing Languages - 3

- *Minimal spanning tree problem solver:*



Recognizing Languages - 4

- *Tools from language theory*
- *Expressibility of languages*
- *Fascinating relationship between the complexity of problems and power of languages*

Proofs

- *What is a proof?*
- *Does a proof need mathematical symbols?*
- *What makes a proof incorrect?*
- *How does one come up with a proof?*

Proof Techniques (Sipser 0.4)

- *Proof by cases.*
- *Proof by contrapositive*
- *Proof by contradiction*
- *Proof by construction*
- *Proof by induction*
- *Others .....*

Proof by Cases

- *If n is an integer, then n(n+1)/2 is an integer*
- *Case 1: n is even.*

or n = 2a, for some integer a

So n(n+1)/2 = 2a(n+1)/2 = a*(n+1), which is an integer.*

- *Case 2: n is odd.*

*n+1 is even, or n+1 = 2a, for an integer a So n(n+1)/2 = n*2a/2 = n*a,*

which is an integer.

Proof by Contrapositive

- *If x^2 is even, then x is even*
- *Proof 1 (DIRECT): $x^2 = x*x = 2a$*

So 2 divides x.

- *Proof 2: prove the contrapositive!*

if x is odd, then x^2 is odd.

$x = 2b+1$. So $x^2 = 4b^2 + 4b + 1$ (odd)

Proof by Contradiction

- *$\sqrt{2}$ is irrational*
- *Suppose $\sqrt{2}$ is rational. Then $\sqrt{2} = p/q$, such that p, q have no common factors. Squaring : $2 = p^2 / q^2$*
- *and transposing : $p^2 = 2q^2$ (even number)*

So, p is even Or $p = 2x$ for some integer x

So $4x^2 = 2q^2$ or $q^2 = 2x^2$

So, q is even

So, p,q are both even – they have a common factor of 2.
CONTRADICTION.

- *So $\sqrt{2}$ is NOT rational. Q.E.D.*

"quod erat demonstrandum"
"which was to be demonstrated

Proof by construction

- *There exists irrational b,c, such that b^c is rational*
- *Consider $\sqrt{2}^{\sqrt{2}}$. Two cases are possible:*
- *Case1:$\sqrt{2}^{\sqrt{2}}$ is rational–DONE($b=c=\sqrt{2}$).*
- *Case 2: $\sqrt{2}^{\sqrt{2}}$ is irrational – Let $b = \sqrt{2}^{\sqrt{2}}$, $c = \sqrt{2}$. Then $b^c = (\sqrt{2}^{\sqrt{2}})^{\sqrt{2}} = (\sqrt{2})^{\sqrt{2}\times\sqrt{2}} = (\sqrt{2})^2 = 2$*

Debug this “proof”

- *For each positive real number a, there exists a real number x such that $x^2 > a$*
- *Proof: We know that $2a > a$ So$(2a)^2 = 4a^2 > a$*
- *So use $x = 2a$.*

Proof by Induction

- *Format:*
  - *Inductive hypothesis,*
  - *Base case,*
  - *Inductive step.*

Proof by Induction

- *Prove: For any $n \in N$, n^3-n is divisible by 3.*
- *IH: P(n): For any $n \in N$, $f(n)=n^3-n$ is divisible by 3.*
- *Base case: P(1) is true, because f(1)=0.*
- *Inductive step:*
  - *Assume P(n) is true. Show P(n+1) is true.*
  - *Observe that $f(n+1) - f(n) = ((n+1)^3-(n+1))-(n^3-n) = 3(n^2 + n)$. So $f(n+1) - f(n)$ is divisible by 3.*
  - *Since P(n) is true, f(n) is divisible by 3. So f(n+1) is divisible by 3.*
  - *Therefore, P(n+1) is true.*
- *Exercise: give a direct proof.*

Next: Finite Automata

Ch. 1: Deterministic finite automata (DFA)

We will study languages recognized by finite automata.