Computability Theory **Fall 2020**

Lecture 1: Introduction and Preliminaries

Lecturer: Renjie Yang

1.1 Introduction

Some historical milestones:

- Euclidean geometry: compass and straightedge
- The middle ages: arithmetic calculations
- Around 825 AD: al-Khowârizmi, *Hiŝab al-jabr w'al-muqâ-balah*
- 1642: Calculating machine by Blaise Pascal
- 19th century: "Difference Engine" and "Analytic Engine" by Charles Babbage
- 1879: Frege, *Begriffsschrift*
- 1900: David Hilbert's Diophantine problem
- 1902: Russell showed that Frege's formal system was inconsistent
- 1930: Turing, Gödel, Herbrand, Church
- 1931: Gödel's incompleteness theorems
- 1944: "Automatic sequence controlled calculator" by IBM and Harvard

1.2 The set theoretical view of math

The modern understanding of mathematics is that all mathematical objects can be defined in terms of the single notion of a "set."

If A is a set and x is some other mathematical object (possibly another set), the relation "x is an element of A" is written $x \in A$.

If A and B are sets, A is a subset of B, written $A \subseteq B$, if every element of A is an element of B.

A and B are equal, i.e. the same set, if $A \subseteq B$ and $B \subseteq A$.

If A and B are sets, $A \cup B$ denotes their union, i.e. the set of things that are in either one, and $A \cap B$ denotes their intersection, i.e. the set of things that are in both.

If A is any set, $P(A)$, "the power set of A," denotes the set of all subsets of A.

The empty set, i.e. the set with no elements, is denoted $\emptyset$;.

$\mathbb{N}$, $\mathbb{Q}$, $\mathbb{R}$ denote the sets of natural numbers, rationals, and real numbers respectively.

If P is a property, then $\{x \in A \mid P(x)\}$ denotes the set of all elements of A satisfying P. $\{x|x \notin x\}$ leads to Russell's paradox.

An ordered pair $\langle a, b\rangle$ can be defined as $\{\{a\}, \{a, b\}\}$, If A and B are sets, $A \times B$ is the set of all ordered pairs $\langle a, b\rangle$ consisting of an element $a \in A$ and an element $b \in B$. Iterating this gives us notions of ordered triple, quadruple, and so on.

A binary relation R on A and B is a subset of $A \times B$. A function f from A to B is a binary relation R_f on A and B such that

- For every $a \in A$, there is a $b \in B$ such that $R_f(a, b)$
- For every $a \in A$, $b \in B$, and $b' \in B$, if $R_f(a, b)$ and $R_f(a, b')$ then $b = b'$

If $f : A \to B$, A is called the domain of f, and B is called the codomain or range.

Definition 1.1 *Suppose f is a function from A to B.*

- *f is injective (or one-one) if whenever x and x' are in A and $x \neq x'$, then $f(x) \neq f(x')$*
- *f is surjective (or onto) if for every y in B there is an x in A such that $f(x) = y$.*
- *f is bijective (or a one-to-one correspondence) if it is injective and surjective.*

Definition 1.2 *Suppose f is a function from A to B, and g is a function from B to C. Then the composition of g and f, denoted $g \circ f$, is the function from A to C satisfying*

$$g \circ f = g(f(x))$$

for every x in C.

Definition 1.3 *A partial function f from A to B is a binary relation Rf on A and B such that for every x in A there is at most one y in B such that $R_f(x, y)$.*

An ordinary function from A to B is sometimes called a total function.

1.3 Graph

Definition 1.4 *An undirected graph G is a pair $(V; E)$, where*

- *V is a finite non-empty set called the set of vertices (or nodes),*
- *E is a set called the set of edges, and every element of E is of the form $\{u, v\}$ for distinct $u, v \in V$.*

Definition 1.5 *Let $G = (V, E)$ be a graph, and $e = \{u, v\} \in E$ be an edge in the graph. In this case, we say that u and v are neighbors or adjacent. We also say that u and v are incident to e. For $v \in V$, we define the neighborhood of v, denoted $N(v)$, as the set of all neighbors of v, i.e. $N(v) = \{u \mid \{v, u\} \in E\}$; The size of the neighborhood, $|N(v)|$, is called the degree of v, and is denoted by deg(v).*

Definition 1.6 *Let $G = (V, E)$ be a graph. A path of length k in G is a sequence of distinct vertices*

$$v_0, v_1, \ldots, v_k$$

such that $v_{i-1}, v_i \in E$ for all $i \in \{1, 2, \ldots, k\}$. In this case, we say that the path is from vertex v_0 to vertex v_k. A cycle of length k (also known as a k-cycle) in G is a sequence of vertices

$$v_0, v_1, \ldots, v_{k-1}, v_0$$

such that $v_0, v_1, \ldots, v_{k-1}, v_0$ is a path, and $\{v_0, \ldots, v_k\} \in E$. A graph that contains no cycles is called acyclic.

Definition 1.7 *Let $G = (V, E)$ be a graph. We say that two vertices in G are connected if there is a path between those two vertices. We say that G is connected if every pair of vertices in G is connected. A subset $S \subseteq V$ is called a connected component of G if G restricted to S, i.e. the graph $G' = (S, E' = \{\{u, v\} \in E \mid u, v \in S\})$ is a connected graph, and S is disconnected from the rest of the graph (i.e. $\{u, v\} \notin E$ when $u \in S$ and $v \notin S$).*

Definition 1.8 *A graph satisfying two of the following three properties is called a tree:*

1. *connected*

2. *$m = n - 1$ (n is the number of vertices; m is the number of edges)*

3. *acyclic*

A vertex of degree 1 in a tree is called a leaf. And a vertex of degree more than 1 is called an internal node.

Definition 1.9 *A directed graph G is a pair (V, A), where*

- *V is a finite set called the set of vertices (or nodes),*

- *A is a finite set called the set of directed edges (or arcs), and every element of A is a tuple $\langle u, v \rangle$ for $u, v \in V$. If $\langle u, v \rangle \in A$, we say that there is a directed edge from u to v.*

Definition 1.10 *Let $G = (V, A)$ be a directed graph. For $u \in V$, we define the neighborhood of u, $N(u)$, as the set $\{v \in V \mid \langle v, u \rangle \in A\}$. The out-degree of u, denoted $deg_{out}(u)$, is $|N(u)|$. The in-degree of u, denoted $deg_{in}(u)$, is the size of the set $v \in V \mid \langle v, u \rangle \in A$. A vertex with out-degree 0 is called a sink. A vertex with in-degree 0 is called a source.*

1.4 Language

Definition 1.11 *An alphabet is a non-empty, finite set, and is usually denoted by Σ. The elements of Σ are called symbols or characters.*

Definition 1.12 *Given an alphabet Σ, a string (or word) over Σ is a finite sequence of symbols, written as $a_1 a_2 a_3 \ldots a_k$, where each $a_i \in \Sigma$. The string with no symbols is called the empty string and is denoted by ϵ.*

Definition 1.13 *The length of a string w, denoted $|w|$, is the the number of symbols in w. If w has an infinite number of symbols, then the length is undefined.*

Definition 1.14 *Let Σ be an alphabet. We denote by Σ^* the set of all strings over Σ consisting of finitely many symbols:*

$$\Sigma^* = \{a_1 a_2 \ldots a_n \mid n \in N, a_i \in \Sigma\}$$

Definition 1.15 *If u and v are two strings in Σ^*, the concatenation of u and v, denoted by uv or $u \cdot v$, is the string obtained by joining together u and v.*

Definition 1.16 *For a word $u \in \Sigma^*$ and $n \in \mathbb{N}$, the n'th power of u, denoted by u^n, is the word obtained by concatenating u with itself n times.*

Definition 1.17 *We say that a string u is a substring of string w if $w = xuy$ for some strings x and y.*

Definition 1.18 *Any (possibly infinite) subset $L \subseteq \Sigma^*$ is called a language over the alphabet Σ.*

Definition 1.19 *Given two languages $L_1, L_2 \subseteq \Sigma^*$, we define their concatenation, denoted $L_1 L_2$ or $L_1 \cdot L_2$, as the language*

$$L_1 L_2 = \{uv \in \Sigma^* \mid u \in L_1, v \in L_2\}$$

Example The concatenation of languages $\{\epsilon, 1\}$ and $\{0, 1\}$ is the language $\{0, 01, 10, 101\}$.

Definition 1.20 *Given a language $L \subseteq \Sigma^*$ and $n \in \mathbb{N}$, the n'th power of L, denoted L^n, is the language obtained by concatenating L with itself n times, that is*

$$L^n = \underbrace{L \cdot L \cdot L \cdots L}_{n \text{ times}}$$

Equivalently,

$$L^n = \{u_1 u_2 \cdots u_n \in \Sigma^* \mid u_i \in L \text{ for all } i \in \{1, 2, \ldots, n\}\}$$

Example $(\{\epsilon, 1\})^3$ is the language $\{\epsilon, 1, 11, 111\}$

Example The 0th power of any language L is the language $\{\epsilon\}$

Definition 1.21 *Given a language $L \subseteq \Sigma^*$, define the star of L, denoted L^*, as the language*

$$L^* = \bigcup_{n \in \mathbb{N}} L^n$$

Equivalently,

$$L^* = \{u_1 u_2 \cdots u_n \in \Sigma^* \mid n \in \mathbb{N}, u_i \in L \text{ for all } i \in \{1, 2, \ldots, n\}\}$$

Example If $L = \{00\}$, then L^* is the language consisting of all words containing an even number of 0's and no other symbol.

1.5 Encoding

Definition 1.22 *Let A be a set, and let Σ be a alphabet. An encoding of the elements of A, using Σ, is an injective function $Enc : A \to \Sigma^*$. We denote the encoding of a $a \in A$ by $\langle a \rangle$. If $w \in \Sigma^*$ is such that there is some $a \in A$ with $w = \langle a \rangle$, then we say w is a valid encoding of an element in A. A set that can be encoded is called encodable.*

Example Every natural number has a base-2 representation (which is also known as the binary representation). This representation corresponds to an encoding of $\mathbb{N}$ using the alphabet $\Sigma = \{0, 1\}$. For example, four is encoded as 100 and twelve is encoded as 1100.

Example Suppose we want to encode the set $A = \mathbb{N} \times \mathbb{N}$ using the alphabet $\Sigma = \{0, 1, 2\}$. One way to accomplish this is to make use of a binary encoding $Enc' : \mathbb{N} \to \{0, 1\}^*$ of the natural numbers. With Enc' in hand, we can define $Enc = \mathbb{N} \times \mathbb{N} \to \{0, 1, \#\}^*$ as follows. For $(x, y) \in \mathbb{N} \times \mathbb{N}, Enc(x, y) = Enc'(x)\#Enc'(y)$. Here the symbol # acts as a separator between the two numbers.

Example Let A be the set of all undirected graphs. Every graph $G = (V, E)$ can be represented by its $|V|$ by $|V|$ adjacency matrix. In this matrix, every row corresponds to a vertex of the graph, and similarly, every column corresponds to a vertex of the graph. The (i, j)'th entry contains a 1 if $\{i, j\}$ is an edge, and contains a 0 otherwise.

Example Let A be the set of all functions in the programming language Python. Whenever we type up a Python function in a code editor, we are creating a string representation/encoding of the function, where the alphabet is all the Unicode symbols. For example, consider a Python function named absValue, which we can write as

```
def absValue(N):
    if (N < 0): return -N
    else: return N
```

By writing out the function, we have already encoded it. More specifically, habsValuei is the string:

absValue(N):\n if (N < 0): return -N \n else: return N

1.6 Types of proof

a) Proof by construction
   Theorem: "x exists; there is at least an x that satisfy $P(x)$"
   Proof: Show how to build an x

b) Proof by contradiction
   Theorem: "S is true."
   Proof: Assume S is false and derive the truth of something known to be false.

c) Proof by mathematical induction
   Theorem: "P is true for all integers ≥ 0."
   Proof: base case: show $P(0)$ is true
   inductive step: assume $P(i)$ is true, show that $P(i + 1)$ is also true

conclude that P is true for all $i \geq 0$

d) Proof by structural induction
Theorem: "P is true for all the elements of C that is recursively defined."
Proof: base case: show P is true for all the minimal structures of C
inductive step: assume P is true for the immediate substructures of a certain structure c, show that P is also true for c
conclude that P is true for all the structures in C

Example
Bese case: show P is true for the root of the tree;
Inductive step: assume P is true for all the ancestors of node x, show that P is also true for x;
Conclude that P is true for all nodes of the tree.

1.7 Cardinality

Definition 1.23 *Two sets A and B are equipollent (or equinumerous), written $A \approx B$, if there is a bijection from A to B.*

Definition 1.24 *A set A is finite if it is equinumerous with the set $\{1, \ldots, n\}$, for some natural number n. A is countably infinite if it is equinumerous with $\mathbb{N}$. A is countable if it is finite or countably infinite.*

Example The set of prime numbers is countably infinite: let $f(x)$ be the x'th prime number.

Proposition 1.25 *A set A is countable if and only if there is a surjective function from $\mathbb{N}$ to A.*

Proof: Suppose A is countable. If A is countably infinite, then there is a bijective function from $\mathbb{N}$ to A. Otherwise, A is finite, and there is a bijective function f from $\{1, \ldots, n\}$ to A. Extend f to a surjective function f' from $\mathbb{N}$ to A by defining

$$f'(x) = \begin{cases} f(x), & \text{if } x \in \{1, \ldots, n\} \\ \\ f(1), & \text{otherwise} \end{cases}$$

Conversely, suppose $f : \mathbb{N} \to A$ is a surjective function. If A is finite, we're done. Otherwise, let $g(0)$ be $f(0)$, and for each natural number i, let $g(i+1)$ be $f(k)$, where k is the smallest number such that $f(k)$ is not in the set $g(0), g(1), \ldots, g(i)$. Then g is a bijection from $\mathbb{N}$ to A. ■

Example If A and B are countable then so is $A \cup B$.

Example $\mathbb{N} \times \mathbb{N}$ is countable. Use the "dovetailing" technique:

$$J(\langle x, y \rangle) = x + \frac{(x+y)(x+y+1)}{2}$$

Example $\mathbb{Q}$ is countable. The function f from $\mathbb{N} \times \mathbb{N}$ to the nonnegative rational numbers

$$f(\langle x,y\rangle) = \begin{cases} \dfrac{x}{y}, \text{ if } y \neq 0 \\ \\ 0, \text{ otherwise} \end{cases}$$

is surjective, showing that the set of nonnegative rational numbers is countable.

Theorem 1.26 *The set of real numbers is not countable.*

Proof: Let us show that the real interval [0, 1] is not countable. Suppose $f : \mathbb{N} \to [0,1]$ is any function; it suffices to construct a real number that is not in the range of f. Note that every real number $f(i)$ can be written as a decimal of the form

$$0.a_{i,0}a_{i,1}a_{i,2}\ldots$$

Now define a new number $0.b_0b_1b_2\ldots$ by making each bi different from $a_{i,i}$. Specifically, set b_i to be 3 if $a_{i,i}$ is any number other than 3, and 7 otherwise. Then the number $0.b_0b_1b_2\ldots$ is not in the range of $f(i)$, because it differs from $f(i)$ at the i'th digit. ■

1.8 Computational problems

Definition 1.27 *Let Σ be an alphabet. Any function $f : \Sigma^* \to \Sigma^*$ is called a computational problem over the alphabet Σ.*

Example Consider the function $g : \mathbb{N} \to \mathbb{N}$ such that $g(x) = x + y$. We can view g as a computational problem over an alphabet Σ once we fix an encoding of the domain $\mathbb{N} \times \mathbb{N}$ using Σ. Take $\Sigma = \{0, 1, \#\}$. Let *Enc* be the ternary encoding of $\mathbb{N} \times \mathbb{N}$, and *Enc*$'$ be the binary encoding of $\mathbb{N}$. We now define the computational problem f corresponding to g. If $w \in \Sigma^*$ is a word that corresponds to a valid encoding of a pair of numbers (x, y), then define $f(w)$ to be $Enc'(x + y)$. If $w \in \Sigma^*$ is not a word that corresponds to a valid encoding of a pair of numbers (x, y), define $f(w)$ to be $\#$.

Definition 1.28 *Let Σ be an alphabet. Any function $f : \Sigma^* \to \{0, 1\}$ is called a decision problem over the alphabet Σ. The codomain of the function is not important as long as it has two elements. Other common choices for the codomain are $\{No, Yes\}$, $\{False, True\}$ and $\{Reject, Accept\}$.*

Example Consider the function $g : \mathbb{N} \to \{False, True\}$ such that $g(x) = True$ if and only if x is a prime number. We can view g as a decision problem over an alphabet Σ once we fix an encoding of the domain $\mathbb{N}$ using Σ. Take $\Sigma = \{0, 1\}$. Let *Enc* be the binary encoding of $\mathbb{N}$. We now define the decision problem f corresponding to g. If $w \in \Sigma^*$ is a word that corresponds to an encoding of a prime number, then define $f(w)$ to be $True$. Otherwise, define $f(w)$ to be False.

There is a one-to-one correspondence between decision problems and languages. Let $f : \Sigma^* \to \{0, 1\}$ be some decision problem. Now define $L \subseteq \Sigma^*$ to be the set of all words in Σ^* that f maps to 1. This L is the language corresponding to the decision problem f. Similarly, if you take any language L , we can define the corresponding decision problem $f : \Sigma^* \to \{0, 1\}$ as $f(w) = 1$ if and only if $w \in L$. We consider the set of languages and the set of decision problems to be the same set of objects.