

Math 114C: Computability Theory

UCLA

Darsh Verma

Winter 2026

Hello and welcome! As the title suggests, these are my lecture notes on Computability Theory. Our professor is **Tyler Arant**. The textbook that we are using is **Computability by NJ Cutland**.

The goal of these lecture notes is to write **understandable** math. Some dude said, "If you can't explain it to a six year old, then you don't understand it yourself". The hope is that anyone coming across these notes (like you!) will be able to at least take away the gist of these concepts. Email me at darsh [at] ucla [dot] edu if you find any errors!

Huge shoutout to <https://zitong.me/notes/rings-notes.pdf> who inspired me to attend class and lock in.

Contents

1 Lecture 1: Jan 5	3
1.1 Introduction	3
1.2 Setup	3
2 Lecture 2: Jan 7	5
2.1 Partial Functions	5
2.2 Definition by Substitution	5
3 Lecture 3: Jan 9	6

List of Definitions

Definition (Natural Numbers)	3
Definition (Tuples)	3
Definition (Relations)	3
Definition (Partial Function)	5
Definition (Total Function)	5

List of Theorems

1 Lecture 1: Jan 5

1.1 Introduction

What does it really mean when we say we can “compute” a quantity? We have some general notion of what an algorithm is:

- The list of instructions is finite
- The procedure is never vague
- No intuition from the user is required; they just follow the instructions
- “Scratch work” may be needed (RAM)
- The algorithm will always terminate in finitely many steps

We will assume an ideal computer, so we don’t care if it has the latest M4 chip or the Intel i239829 chip. This motivates the fact that we need precise definitions for what “computable” and “decidable” are.

It turns out that “yes” is easy to show. You just describe an algorithm, and prove that it works.

On the other hand, “no” is not easy to show. For example, you can be thinking about a problem for centuries, and not have come up with any algorithm to solve it, but that doesn’t necessarily mean that there does not exist such an algorithm. Proving that such an algorithm doesn’t exist is something we will do later in the course.

Example 1.1. There is no computable way to solve Diophantine equations (integer solutions to multivariable polynomial equations).

$$x^2 + y^2 = z^2$$

These are the **Pythagorean triplets**!

1.2 Setup

Definition (Natural Numbers).

$$\mathbb{N} := \{0, 1, 2, \dots\}$$

Definition (Tuples). Let n is a positive natural number and let X be a set, then

$$X^n := \{(x_0, x_1, \dots, x_{n-1}) : x_0, \dots, x_{n-1} \in X\}$$

Definition (Relations). An n-ary relation on \mathbb{N} is a subset $R \subseteq \mathbb{N}$. This was slightly confusing so the examples in the notes helped.

Example 1.2. The divisibility relation $R \subseteq \mathbb{N}^2$ can be defined as $R(x, y)$ iff x divides y . We can also use set notation.

$$R := \{(x, y) \in \mathbb{N}^2 : x \text{ divides } y\}$$

$R(4, 8)$ holds but $R(8, 4)$ doesn't hold.

2 Lecture 2: Jan 7

2.1 Partial Functions

Definition (Partial Function). Let X, Y be sets. A **partial function** $f : X \rightarrow Y$ is a function f whose domain D is a subset $D \subseteq X$ and whose codomain is Y .

Some other notation

$f(x) \downarrow \implies x$ is in the domain of f

$f(x) \uparrow \implies x$ is not in the domain of f

Example 2.1. $f(x, y) = \frac{x}{y}$ is a partial function $f : \mathbb{R}^2 \rightarrow \mathbb{R}$. $f(x, y) \downarrow \iff y \neq 0$

Example 2.2. $f(x) = \sum_{n=0}^{\infty} x^n$ is partial function $f : \mathbb{R} \rightarrow \mathbb{R}$. $f(x) \downarrow \iff |x| < 1$

In this course, we will almost exclusively be concerned with n -ary partial functions on \mathbb{N} , $f : \mathbb{N}^n \rightarrow \mathbb{N}$

Definition (Total Function). When $f(\vec{x}) \downarrow \forall \vec{x} \in \mathbb{N}^n$, we say f is a **total function**. According to our definition, total functions are a special type of partial function. Specifically, when we say "let f be a partial function", we are not ruling out that f is total.

When are two partial functions equal? Same notion as regular functions.

Definition schemes for partial functions 3 important ones:

- definition by substitution
- defintion by minimization
- definition by primitive recursion

2.2 Defintion by Substitution

Roughly speaking, when $f(\vec{x}) = h(g_0(\vec{x}), \dots, g_{k-1}(\vec{x}))$ we say f is defined via substitution from h, g_0, \dots, g_{k-1} .

Example 2.3. Let $h(x_0, x_1, x_2)$ be 3-ary. Define $f(x_0, x_1) := h(x_1, x_0, x_1)$. There is a substitution definition using h with **projection functions**

3 Lecture 3: Jan 9

if $f(y) = y!$ and we define $m_0 = 1$, $h(t, y) = t \cdot (y + 1)$, then