# Part III
# Introduction to Computational Complexity

Lectured by Timothy Gowers

Artur Avameri

## Contents

# 0 Introduction

A good book for the course is the first few chapters of *Computational Complexity: A modern approach* by Arora and Barack.

## 0.1 Computational problems

A problem in complexity theory is somewhat different from a mathematical problem – it is more like a class of problems.

**Example 0.1.** Given: A graph $G$ with $n$ vertices and $x, y \in V(G)$. Problem: Is there a path from $x$ to $y$? This is a decision problem (yes/no answer).

A problem has a variable input and an output. If the output belongs to $\{0,1\}$, i.e. $\{\text{no, yes}\}$, then the problem is called a **decision problem**. Write $\{0,1\}^*$ for the set of $\bigcup_{n=1}^{\infty}\{0,1\}^n$. Then a decision problem can be encoded as a **Boolean function**, that is, a function $f : \{0,1\}^* \to \{0,1\}$.

The set $\{x \in \{0,1\}^* \mid f(x) = 1\}$ is called the **language** defined by $f$.

## 0.2 Turing machines

A **Turing machine** formalizes the notion of an algorithm. There are many ways to formalize it – we (handwavily) describe a few.

A $k$–**tape Turing machine** consists of several ingredients. The first is a collection of $k$ **tapes**, where a tape is an infinite sequence of **cells**. There is also a finite set $A$ called the **alphabet**, and each cell contains an element of $A$. There is also a **head** which is in a **state** (an element of a finite set $S$ of states) and in a **position** in each state. $S$ contains two special states, $S_{\text{init}}$ and $S_{\text{halt}}$.

A state is a function that takes as input an element of $A^k \times S$ and outputs an element of $A^k \times S \times \{L, N, R\}^k$. If $S$ is this "transition function", then the machine rewrites $(a_1, \dots, a_k)$ according to the $A^k$ component of the image, changes the state of the head according to the $S$ component, and shifts each tape according to the $\{L, N, R\}^k$ component.

One tape is designated as the input tape and is never changed, another is the output tape. All tapes other than the input tape start full of zeroes. If the machine reaches the state $S_{\text{halt}}$, it stops. If the input is $x$ and the output is $y$, we say that the machine computed $y$ given input $x$.

**Variants** assume that $A = \{0,1\}$; that $k = 1$ (with a different convention about input–output tapes); that tapes are two–sided, etc.

# 1 Some complexity classes

The complexity class $P$ consists of all Boolean functions (i.e. problems) $f : \{0,1\}^* \to \{0,1\}$ such that there exists a Turing machine $T$ and a polynomial $p$ such that for every $x \in \{0,1\}^*$, $T$ computes $f(x)$ in at most $p(|x|)$ steps, where $|x| = m$ if $x \in \{0,1\}^m$.

**Example 1.1.** The problem st-CONN (input: directed graph $G$ and two vertices $s, t$; output: 1 if there is a directed path from $s$ to $t$) belongs to $P$.

**Example 1.2.** Input: positive integers $m, n$ and output: $mn$.