P vs NP
for the rigorous mathematician

Dominic van der Zypen

1. Introduction

I have long had an iffy feeling about the formulation of the **P** vs **NP** problem. I sort-of know Turing machines, I have a vague concept of what is "a class of problems", and I have heard many times that **NP** is the class of problems for which a solution can be checked in polynomial time for correctness.

All my ifs-and-buts add to a quite shaky view - and I decided to get to the bottom of things and provide a rigorous and (hopefully) mathematically appealing definition of Turing machines, languages, and the like.

2. Configurations

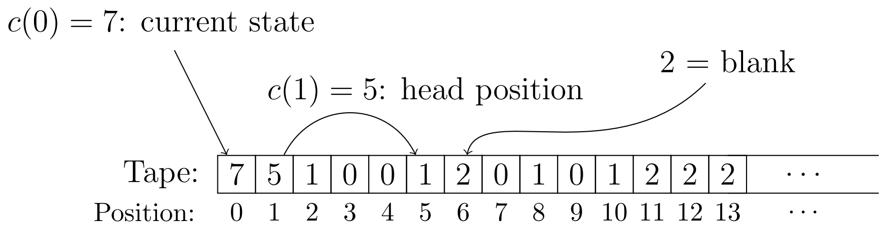
Let ω denote the first infinite ordinal (which can be thought of as $\mathbb{N}$, the set of non-negative integers). Recall that each $n \in \omega$ is an *ordinal*, that is, $0 := \emptyset$ and for $n > 0$, its members are the numbers $0, \ldots, n-1$, so $n = \{0, \ldots, n-1\}$. We write ω^ω for the collection of all maps $f : \omega \to \omega$. Members of ω^ω are also called *integer sequences*.

Our first central concept, the **configuration**, is the mathematical model of the notion of the **tape** of the Turing machine, together with the **writing head** and the **internal state** of the machine.

Definition 2.1. Let $n \geq 2$ be an integer. Then an *n-configuration* is a function (also called an *integer sequence*) $c \in \omega^\omega$ with the following properties:

(1) $c(0) \in n = \{0, \ldots, n-1\}$,
(2) $c(1) \geq 2$,
(3) $c(k) \in \{0, 1, 2\}$ for all $k \in \omega \setminus \{0, 1\}$, and
(4) c is eventually constant with value 2. (This means that there is $N \in \omega$ such that $c(k) = 2$ for all $k \in \omega$ with $k \geq N$.)

Next, we illustrate and explain the meanings of the entries of c in detail:



- $c(0) \in n$ represents the *state* of the possible n states $\{0, \ldots, n-1\}$ of the configuration. States 0 and 1 (stored in the first cell, $c(0)$) have a special meaning: $0 =$ *reject*, $1 =$ *accept*.
- $c(1) \geq 2$ represents the position of the *read/write head* of the configuration.

- We interpret 2 as being the *blank* symbol. The *value* symbols are 0 and 1. The blank symbol 2 can be used to separate "input strings" consisting of 0,1. Every configuration is eventually blank (=2).

The collection of n-configurations is denoted by $\text{Config}_(n)$.

3. Turing machines

Definition 3.1. A *Turing machine* is a tuple (n, δ) where $n \in \omega, n \geq 2$ and $\delta : n \times \{0,1,2\} \to n \times \{0,1,2\} \times \{-1,1\}$ is a function with following property:

$$\text{if } q \in \{0,1\}^1 \text{ and } b \in \{0,1,2\}, \text{ then } \delta(q,b) = (q,b,-1).$$

The interpretation of this is that δ gets constant whenever an *accept* or *reject* state has been reached.

If $\delta(q,b) = (q', b', s)$ for $q, q' \in Q, b, b' \in \{0,1,2\}$ and $s \in \{-1,1\}$ we write

- $\texttt{nextstate}(q,b) = q'$,
- $\texttt{output}(q,b) = b'$, and
- $\texttt{step}(q,b) = s \in \{-1,1\}$.

The function δ is called the *transition function* and can be looked at as the *clockwork* of the Turing machine.

4. Synthesis: combining configurations and Turing machines

We need the following *convention*: if $n \in \omega$, define the function $\texttt{pred} : \omega \to \omega$ by $0 \mapsto 0$ and $n \mapsto n-1$ for $n \in \omega \setminus \{0\}$. By slight abuse of notation, we write $n-1$ instead of $\texttt{pred}(n)$ for all $n \in \omega$.

Definition 4.1. Let $T = (n, \delta)$ be a Turing machine, T induces a *configuration map*

$$\mathfrak{C}_T : \text{Config}(n) \to \text{Config}(n)$$

in the following way. Let $c \in \text{Config}(n)$, then we define $\mathfrak{C}_T(c) : \omega \to \omega$ by

- $\mathfrak{C}_T(c)(0) = \texttt{nextstate}(c(0), c(c(1)))$ [2]
- $\mathfrak{C}_T(c)(1) = c(1) + \texttt{step}(c(0), c(c(1)))$, [3]
- $\mathfrak{C}_T(c)(c(1)) = \texttt{output}(c(0), c(c(1)))$, and [4]
- $\mathfrak{C}_T(c)(x) = c(x)$ for all $x \in \omega \setminus \{0, 1, c(1)\}$.

So we have $\mathfrak{C}_T(c) \in \omega^\omega$, and it easy to check that $\mathfrak{C}_T(c) \in \text{Config}(n)$.

---

[1] recall that 0,1 are special states with the meanings $\{$*reject, accept*$\}$, respectively.

[2] $c(0)$ is the current state in the set of possible states $\{0, \dots, n-1\}$, and $c(1)$ is the position of the read/write head, and finally $c(c(1))$ is the **value** of the cell at the head.

[3] $\texttt{step}(c(0), c(c(1))) \in \{-1,1\}$, and $0 - 1 = 0$ in our convention.

[4] so the output created by the transition function δ gets inserted at the head position $c(1)$.