Table of Contents

Chapter 3

Universal Machines

CERTAIN TURING MACHINES have the remarkable property that they can emulate, or imitate, all others.

3.1 Universal Turing machines

We have defined a code $\langle T \rangle$ for Turing machines in the last Chapter (Subsection 2.4.3.

Definition 3.1. *We say that the Turing machine U is* universal *if*

$$U\left(\langle T \rangle p\right) = T(p)$$

for every Turing machine T and every string $p \in \mathbb{S}$.

Informally, a universal Turing machine can 'emulate' any Turing machine.

There is another definition of a universal machine which is sometimes seen: the machine U is said to be universal if given any Turing machine T we can find a string $s = s(T)$ such that

$$U(sp) = T(p)$$

for all $p \in \mathbb{S}$. Evidently a universal machine according to our definition is universal in this sense; and since we are only concerned with the *existence* of a universal machine it does not matter for our purposes which definition is taken.

Theorem 3.1. *There exists a universal Turing machine.*

We shall outline the construction of a universal machine in Chapter **??**. The construction—which is rather complicated—can be divided into 4 parts:

1. We instroduct a variant of Turing machines, using stacks instead of a tape.

2. We show that '2 stacks suffice', with these stacks replacing the left and right halves of the tape.

3. We show that $n \geq 2$ stacks is equivalent to 2 stacks, in the sense that given an n-stack machine S_n we can always find a 2-stack machine S_2 such that
$$S_2(p) = S_n(p)$$
for all inputs p.

4. We show that a universal machine can be implemented as a 4-stack machine. 2 of the stacks being used to store the rules of the machine being emulated.

But for the moment we merely point out that the Church-Turing thesis suggests that such a machine must exist; for it is evident that given the rules defining a machine T, and an input string p, the 'human computer' can determine the state of the machine and the content of the tape at each moment $t = 0, 1, 2, \ldots$, applying the appropriate rule to determine the state and tape-content at the subsequent moment.

There are many universal machines. Indeed, a universal machine U can emulate itself. So the machine V defined by

$$V(p) = U(\langle U \rangle p)$$

is also universal.

It is an interesting question to ask if there is a *best* universal machine in some sense of 'best'. One might ask for the universal machine with the minimal number of states; for there are only a finite number of Turing machines with $\leq n$ states, since there are only a finite number of rules

$$(q_i, b) \mapsto (a, q_o)$$

with $0 \leq q_i, q_o \leq n$. This question has some relevance for us, since we shall define algorithmic entropy (shortly) *with respect to a given universal machine*. It will follow from the fact that two universal machines U, V can emulate each other that the entropies with respect to U and V cannot differ by more than a constant $C = C(U, V)$. But it would be nice to have a concept of *absolute* algorithmic entropy.

Summary

The universal machine U performs exactly like the machine T, provided the program is prefixed by the code $\langle T \rangle$ for the machine:

$$U\left(\langle T \rangle p\right) = T(p).$$