B.2 Polynomial Size Families of Circuits

A circuit has a fixed number of input nodes. Therefore it can process inputs of a fixed length only, in contrast with a TURING machine. However to assess the efficiency of an algorithm in general we have to estimate the increase of cost for increasing input sizes.

To this end we consider families $(C_n)_{n\in\mathbb{N}}$ of circuits with an increasing number of deterministic input nodes. Then the cost of a computation may be expressed as a function of the length of the input.

More exactly we define: A **family of probabilistic circuits (FPC)** is a family $C = (C_n)_{n\in\mathbb{N}}$,

$$C_n\colon \mathbb{F}_2^{r(n)} \times \mathbb{F}_2^{k(n)} \longrightarrow \mathbb{F}_2^{s(n)}, \tag{1}$$

where the circuit C_n has $r(n)$ deterministic input nodes, and $k(n)$ probabilistic ones. Of course, if all $k(n) = 0$, we speak of a family of deterministic circuits.

A **polynomial size family of probabilistic circuits (PPC)** is an FPC $C = (C_n)_{n\in\mathbb{N}}$, such that $\#C_n \leq \alpha(n)$ for all $n \in \mathbb{N}$ with a polynomial $\alpha \in \mathbb{N}[X]$ (non-negative integer coefficients). In particular the number of input nodes of all kinds, as well as the number $s(n)$ of output nodes, is polynomially bounded. (We don't require that the functions r, k, s themselves are polynomials.)

Even in the deterministic case this model of computation might be able to compute more functions than the common model of TURING machines (and it is in fact), since it allows to choose a different algorithm for each input length. For this reason we also speak of a "non-uniform computational model". On first sight this feature seems not so pleasant. Nevertheless it is particularly realistic for cryptanalysis: Depending on the input size n the cryptanalyst may choose a suitable algorithm.

If a TURING machine computation in polynomial time is possible, then for the same problem there is a PPC also. The reverse statement is *not* true, although we only know "artificial" counterexamples.

Should any NP-complete problem be computable by a PPC, then so would be all the other ones. Virtually nobody believes in this possibility.

Non-uniform complexity may be modelled by TURING machines also, simply admitting a different TURING machine for each input length. Moreover we could also define probabilistic TURING machines. After all preferring the SHANNON model of circuits over TURING machines is a matter of taste.

A computational problem is called **hard** if there is no PPC that solves it with a distinguished advantage; we'll make this definition more precise in the next sections. The "hard number theoretic problems" from Chapter 5, such as prime decomposition, are conjectured to be hard in this sense.

We know already that the basic operations on integers are computable by (even deterministic) PPC's. And therefore so are all algorithms on integers that are efficiently computable in the naive sense, using only "polynomially many" elementary arithmetic operations.