8.7 Derandomisation

God does not play dice with the universe.

Albert Einstein

Throughout this chapter, we introduced the power of adding randomisation into an algorithm. In this section, we examine a connection between randomised and deterministic algorithms. In particular, we examine how to arrive at deterministic algorithms from a randomised algorithm. This concept is called *derandomisation*. Why is this type of question even useful?

As it turns out, the concept of *randomisation* has a few practical and philosophical difficulties; one philosophical difficulty that arises can be best described with Einstein's quote in the epigraph: do *truly* random events even exist in the first place? Of course, there are occurrences that seem random but we can often describe these occurrences using some expression – however unwieldy they may be. Consider a random number generator, for example. Their outcomes may seem to follow no specific patterns. However, the underlying mechanism of these generators follow a kind of algorithm that mimic the behaviour of randomness. In a sense, they are not *truly random* since the outcomes can be predicted accurately by examining the behaviour of the algorithm.

To best demonstrate the concept of derandomisation, we will explore this idea with an example. Consider the following problem.

Problem (Polynomial testing)**.** Given two polynomials $Q(x_1, \dots, x_n)$ and $R(x_1, \dots, x_n)$ over n variables with coefficients in some field $\mathbb{F}$, decide whether $Q \equiv R$; that is, decide whether the two polynomials are *equivalent*.

For example, if $Q(x_1, x_2) = (1 + x_1)(1 - x_2)$ and $R(x_1, x_2) = 1 + x_1 - x_2 - x_1x_2$, then the algorithm should return YES. On the other hand, if $Q(x_1, x_2) = (1+x_1)(1-x_2)$ and $R(x_1, x_2) = 1+x_1+x_2+x_1x_2$, then the algorithm should return NO. Note that we can transform this problem into a simpler problem of determining whether a given polynomial is identically the zero polynomial by letting $P \equiv Q - R$. The problem is then equivalent to deciding whether $P \equiv 0$. For the rest of the example, we will just consider the problem of determining whether a polynomial is identically the zero polynomial stated below.

Problem (Polynomial identity testing)**.** Given a polynomial $P(x_1, \dots, x_n)$ over n variables with coefficients in some field $\mathbb{F}$, decide whether $P \equiv 0$; that is, decide whether the polynomial is *equivalent* to the zero polynomial.

A very easy way to approach the problem is to expand out P so that it is simply a sum of monomials; for example, we can expand $P(x_1, x_2) = (1 + x_1)(1 - x_2) - 1 - x_1 + x_2 + x_1x_2$ into $Q(x_1, x_2) = 1 + x_1 - x_2 - x_1x_2 - 1 - x_1 + x_2 + x_1x_2 = 0$. However, this has exponential-time complexity at worst case. A circuit representing the polynomial $P(x_1, \dots, x_n) = \prod_{i=1}^{n}(1+x_i)$ has length $O(n)$ but expands into $O(2^n)$ monomials.

Luckily, there is a simple and elegant probabilistic algorithm that tests for membership. The key insight is that, while naively expanding out all of the terms gives rise to an exponential-time computation, we can evaluate polynomials more efficiently at any given point $(a_1, \dots, a_n)$. This leads naturally to the *Schwartz-Zippel Lemma*.

8.7.1 Schwartz-Zippel Lemma

To motivate the lemma, we first observe that, if a polynomial $P(x_1, \dots, x_n)$ is *identically* the zero polynomial, then testing P on any point $(a_1, \dots, a_n)$ should give $P(a_1, \dots, a_n) = 0$. There are two cases that may occur.

1. If $P(a_1, \dots, a_n) \neq 0$, then clearly P cannot be the zero polynomial. So certainly, we should return NO.

2. If $P(a_1, \ldots, a_n) = 0$, then either: P *is* identical to the zero polynomial, or $(a_1, \ldots a_n)$ just so happens to be a root of the polynomial and P is not identical to the zero polynomial.

Our goal is to find a suitable bound on the probability that evaluating P on an arbitrary point $(a_1, \ldots, a_n)$ is equal to zero. This is the *Schwartz-Zippel Lemma*.

Lemma (Schwartz-Zippel Lemma)**.** Let $P(x_1, \ldots, x_n)$ be a non-zero polynomial with degree d, and let S denote a finite set of elements. If $a_1, \ldots, a_n$ is chosen independently and uniformly at random from S, then

$$\mathbb{P}[P(a_1, \ldots, a_n) = 0] \leq \frac{d}{|S|}.$$

In other words, the probability that $(a_1, \ldots, a_n)$ just so happens to be a root of P with $P \not\equiv 0$ is bounded by at most $d/|S|$.

Proof. We prove the lemma by induction on n.

- When $n = 1$, we arrive at the univariate polynomial case. A polynomial $P(x)$ of degree d has at most d distinct roots over S; therefore, the probability that a_1 is a root of P is at most $d/|S|$.
- Suppose that the statement is true for polynomials over $n-1$ variables. Note that we can express P as a polynomial over the variables $x_2, \ldots, x_n$ as follows:

  $$P(x_1, \ldots, x_n) = \sum_{i=0}^{d} x_1^i P_i(x_2, \ldots, x_n),$$

  where P_i is a polynomial over $n-1$ variables of degree $d-i$. Now, since P is a non-zero polynomial, at least one such P_i is non-zero. Let i be the largest index such that P_i is non-zero. We now arbitrarily pick some $a_2, \ldots, a_n$ from S. By our inductive hypothesis, we have that

  $$\mathbb{P}[P_i(a_2, \ldots, a_n) = 0] \leq \frac{d-i}{|S|}$$

  whenever $P_i(a_2, \ldots, a_m) = 0$. On the other hand, if $P_i(a_2, \ldots, a_m) \neq 0$, [complete the proof].

  Hence, the total probability is given by

  $$\mathbb{P}[P(a_1, \ldots, a_n) = 0] \leq \frac{d-i}{|S|} + \frac{i}{|S|} = \frac{d}{|S|},$$

  which completes the induction proof.

□

This means that, as we increase the size of the field that we're working over, the probability that a randomly chosen vector $(a_1, \ldots, a_n)$ is simply a root of a non-zero polynomial goes to zero. In particular, if we take the set S to have size twice the degree of our polynomial P, then indeed we can bound the probability of a false-positive by $1/2$. We can keep reducing this probability to any value by repeating the trials.

Theorem. If the *polynomial identity testing* problem is in P, then either the *permanent matrix* problem is not

8.7.2 Pseudorandom Generators and the Nisan-Wigderson Construction