SHOR’S ALGORITHM

Although any integer number has a unique decomposition into a product of primes, finding the prime factors is believed to be a hard problem. In fact, the security of our online transactions rests on the assumption that factoring integers with a thousand or more digits is practically impossible. This assumption was challenged in 1995 when Peter Shor proposed a polynomial-time quantum algorithm for the factoring problem. Shor’s algorithm is arguably the most dramatic example of how the paradigm of quantum computing changed our perception of which problems should be considered tractable. In this section we briefly summarize some basic facts about factoring, highlight the main ingredients of Shor’s algorithm, and illustrate how it works by using a toy factoring problem.

Suppose our task is to factor an integer N with d decimal digits. The brute force algorithm goes through all primes p up to root(N) and checks whether p divides N . In the worst case, this would take time roughly root(N), which is exponential in the number of digits d . A more efficient algorithm, known as the quadratic sieve, attempts to construct integers a,b such that root(a^2-b^2) is a multiple of N. Once such a,b are found, one checks whether a± b have common factors with N.The quadratic sieve method has asymptotic runtime exponential in root(d). The most efficient classical factoring algorithm, known as general number field sieve, achieves an asymptotic runtime exponential in cube root(d).

Let us now show that a quantum computer can efficiently simulate the period-finding machine. As in the case of the Deutsch-Jozsa algorithm, we shall exploit quantum parallelism and constructive interference to determine whether a complicated function has a certain global property that cannot be learned by evaluating the function only at a few points. However, instead of detecting the property of being a balanced function, we seek to detect and measure the periodicity of the modular exponentiation function. The fact that interference makes it easier to measure periodicity should not come as a big surprise. After all, physicists routinely use scattering of electromagnetic waves and interference measurements to determine periodicity of physical objects such as crystal lattices. Likewise, Shor’s algorithm exploits interference to measure periodicity of arithmetic objects.

In order to use the phase estimation algorithm, we need to construct a quantum circuit implementing the modular multiplication operation. By analogy with classical algorithms that can link standard library functions, a quantum algorithm is allowed to call classical subroutines; for example, a subroutine for computing the modular multiplication. Importantly, before such classical subroutines are incorporated into a quantum circuit, they must be transformed into a *reversible form.* More precisely, a quantum algorithm can call a classical subroutine only if it is compiled into a sequence of reversible logical gates such as CNOT or Toffoli gates (in particular, the number of input and output wires in each gate must be the same). The subroutine is allowed to use a scratch memory similar to local variables used by the standard library functions. However, once the subroutine is completed, the scratch memory must be totally clean (say, all zeros). The reason is that a quantum algorithm operates on coherent superpositions of different classical states. Leaving information about the inputs or the outputs in the scratch memory could potentially destroy quantum coherence and prevent the algorithm from seeing interference between different states. Since the notion of reversible classical circuits plays an important role in Shor’s algorithm and many other quantum algorithms, below we briefly discuss methods for constructing such circuits.

