Shor's Algorithm & Fermat Numbers

Gabriel Maxwell

May 5 2022

1 Period Finding for Classical Factoring

Given any N, let $a \in \{\mathbb{N} | \, 1 < a < N\}$
Let $N = pq$, where $p, q \in \mathbb{N}$
If $a^r = 1 \, mod(N)$, then r is the period of $a \, mod(N)$
Theorem: Given a, N, r we have $a^{r/2} \pm 1 = P, Q$, such that $gcd(P, N) = p$, $gcd(Q, N) = q$
The proof of which can be found in any Number Theory textbook or Appendix 4 of Nielsen and Chuang.

Example:
Take $N = 57$ and $a = 13$
$13^{18} = 1 \, mod(57)$
$13^9 \pm 1 = 10604499374, 10604499372$
$gcd(57, 10604499372) = 3$
$gcd(57, 10604499374) = 19$
$19 * 3 = 57$

Thus, if you can find the period, you can easily calculate the factors.

2 Shor's Algorithm

The general algorithm factorizes any number through period finding. There are two main components to Shor's Algorithm and that is the phase estimation and the inverse fourier transform. The algorithm quantum mechanically finds the period of the modular exponentiation function, then classically determines the factors. For a b-bit number the algorithm requires $3b$ qubits in the first register.

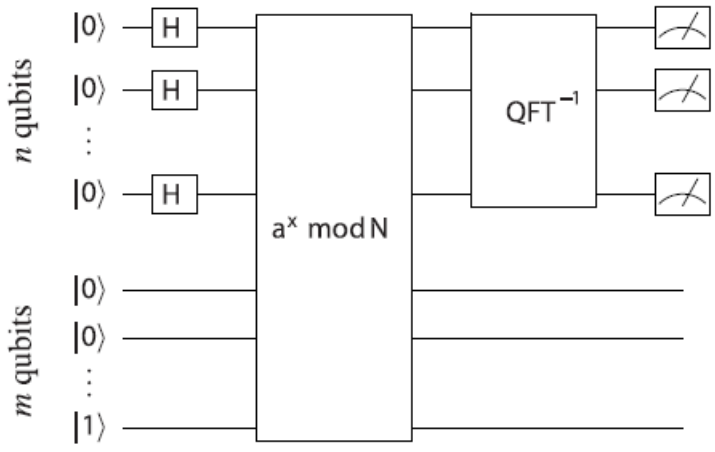

Figure 1: Traditional Circuit for SA

The result of the measurements will give amplitudes that have the form:
$j * \frac{2^n}{r}$, where r is the period; $j = 0, 1, 2..., r - 1$; and n is the number of qubits in the first register.

3 Fermat Numbers

Fermat numbers have the form:

$$f(n) = 2^{2^n} + 1$$

Examples of Fermat numbers: $3, 5, 17, 257, 65537$

A product of Fermat numbers have the form:

$$N = (2^{2^k} + 1)(2^{2^j} + 1)$$

Examples of products of Fermat numbers: $15, 51, 85, 771, 1285$

The products of these numbers have the special property such that for any base a coprime to N, the period of $a \, mod(N)$ is a power of 2. Because of this the phase estimation can be performed by minimal CNOT gates, the number of which depends on the period.

The example circuit in Figure 2 has a base $a = 16$ and that the period $r = 2$ only one CNOT gate is necessary to perform the algorithm.

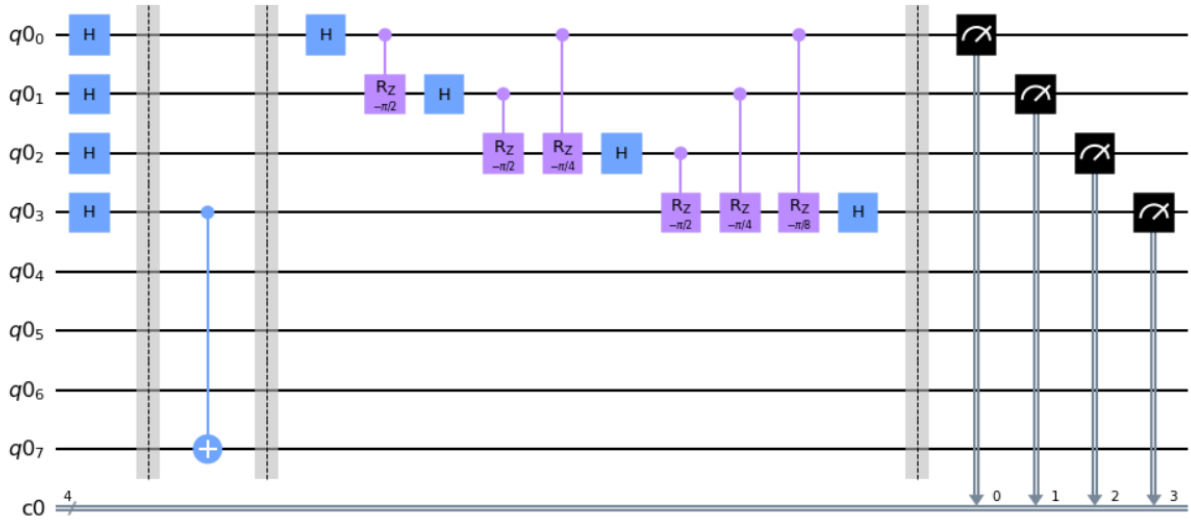

Figure 2: Simplified Circuit for Factoring 51

As well as simplifying the phase estimation, the number of qubits required as also greatly reduced. With an upper bound of approximately:
$l_{max} \leq 2^k + 2^j - 1$, where j, k are the Fermat numbers used to get the product N.

Example: $3 * 17 = 51$
$f(0) = 3, f(3) = 17$
$l_{max} \leq 2^0 + 2^3 - 1 = 8$

Knowing the period beforehand allows one to use the minimal number of CNOTs necessary, but the purpose of Shor's Algorithm is to find the period of any N. This technique of using products of Fermat primes isn't a true demonstration of SA but rather a unique special case where the algorithm can be optimized for a known type of number.

4 References

Geller, Michael R., and Zhongyuan Zhou. "Factoring 51 and 85 with 8 Qubits." Scientific Reports, vol. 3, no. 1, 2013, https://doi.org/10.1038/srep03023.

Nielsen, Michael A., and Isaac L. Chuang. Quantum Computation and Quantum Information. Cambridge University Press, 2021.