CSE439/510 Week 10: Details of Shor's Algorithm

CSE 439/510 Lecture Tue Oct 29 Fall 2024

Shor's Algorithm, Stating Its Backtrack Points BP1 & BP2

Input: $M = pq$, where p, q are n-bit primes. So $\log_2 M \approx 2n$.

Guess $a < M$. If $\gcd(a,M) > 1$ (a tiny chance) we get a factor right away. So suppose gcd is 1, i.e. a is relatively prime to M ($a \in G_M$).

Goal: Compute the true period: least r such that $a^r \equiv 1$ modulo M. Note: multiples of r are also periods, and we may get them instead.

BP1: a may be unlucky in that even after getting (an) r, it is not true or otherwise the classically randomized part fails. Optimal analysis makes it so this is at most a 50-50 chance of backtracking all the way here where you have to guess a different a.

Shor's Algm, Stating Backtrack Points BP1 and BP2

Input: $M = pq$ M is an n-bit number, so $\log_2 M \approx n$. $2^n \approx M$.

Guess $a < M$. If $\gcd(a,M) > 1$, a tiny chance, we get a factor "forthwith".

We may suppose a is rel. prime to M, i.e. $a \in G_M$. $r \leq |G_M| - 1$

Goal: compute the true period $\equiv$ least r s.t. $a^r \equiv 1 \mod M$
(We may instead get a multiple of r, and will hash that out later.)

BP1: a may be unlucky in that even after getting (an) r the classical part may fail.

Optimal analysis puts this chance at most 50%.

Given r, define r_0 to be the odd number obtained by dividing out all 2's from r.

$k \leq n$ [* If $r = 2^k r_0$ and a has period r, then $a' = a^{2^k}$ has period r_0]

Steps of the Quantum Part: (after guessing a but maybe considering all of $a, a^2, a^4, a^8, \ldots$ along with it)

(Q1) Fatten up the domain of $f_a(x) = a^x \bmod M$ to include all x up to $Q = 2^\ell$ where $\ell = \lceil \log_2(M^2) \rceil$ so Q is the least power of 2 above M^2. (We actually only need to guarantee $Q > rM$, noting that $r < |G_M| < M$.)
Then $\ell \approx 4n$ since $\log(M^2) = 2\log M \approx 2 \cdot 2n$.
The range stays mod M. Thus if $f_a(x) = y$, x has ℓ bits and y has n bits. (Text says y has ℓ bits at plot bottom — a minor typo.)

(Q2) Prepare the state $\mathbf{a} = \frac{1}{\sqrt{Q}} \sum_{x<Q} |x\rangle|f_a(x)\rangle$.

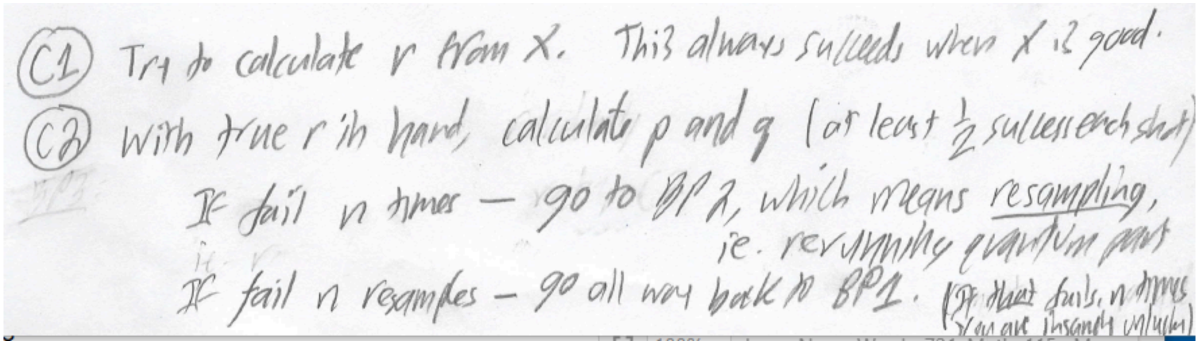
(Q3) Apply QFT_ℓ (or its inverse) to the $|x\rangle$ part to get $\mathbf{b}$

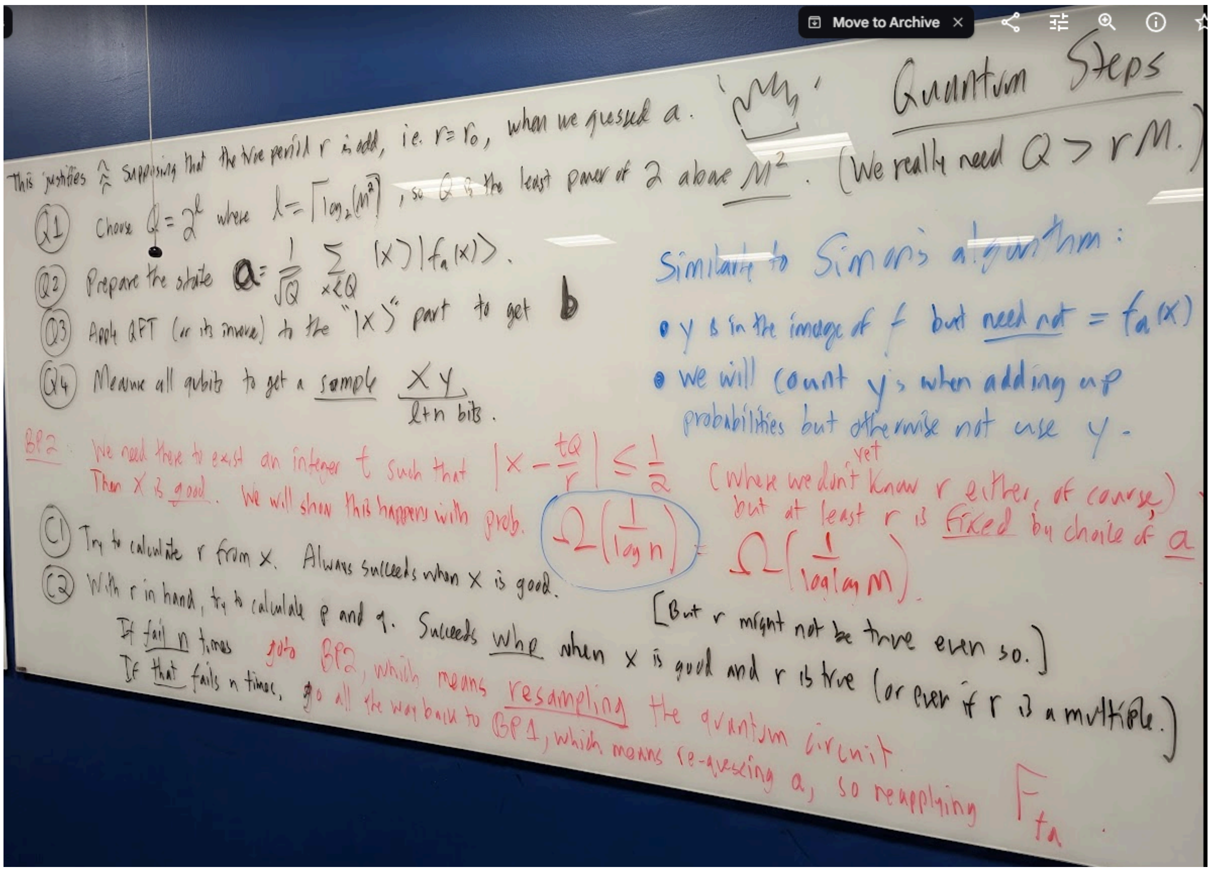
(Q4) Measure all qubits to get a sample $\underbrace{xy}_{\ell+n \text{ bits}}$. Similarly to Simon's algorithm:

- y is in the range of f_a but need not equal $f_a(x)$.
- We will include the # of y's when adding up amplitudes and probabilities, but otherwise we ignore y. We work further only with x.

The second backtrack point comes after the measurement. A quantum technote: Because the measurement "collapses" the quantum state **b**, in the actual quantum algorithm, backtracking here requires rebuilding the whole functional superposition---i.e., redoing the whole circuit. But in my brute-force quantum simulator, it can do another sample without having to re-create all the Boolean formulas that simulate the superposed applications of $f_a(x) = a^x \bmod M$.

BP2: We need there to exist an integer t such that $\left|x - \frac{tQ}{r}\right| \leq \frac{1}{2}$, where we don't know r either, of course, but r is fixed. We also need t to be relatively prime to r, so that tQ/r does not simplify. Then x is good. Chance that x is good: we will show $\Omega\left(\frac{1}{\log n}\right)$. So we do under linearly many backtracks to here.





Analytical Goals of Shor's Algorithm (looking ahead to chapter 12)

The top-down goal is to find a number X such that $X^2 \equiv 1$ modulo M but X is not $\equiv 1$ or $\equiv -1$ modulo M. Then $X^2 - 1 \;=\; (X-1)(X+1)$ is a multiple of M but neither factor is zero. When $M = pq$ with p, q prime, this means p and q each divide one or both factors. We need to split them across the factors, so that $\gcd(X-1, M)$ and/or $\gcd(X+1, M)$ will find p and q as opposed to just giving M back again. Thus we want to guess a such that:

1. The period r of a is even, so that $r/2$ is defined;
2. $X = a^{r/2} \not\equiv M-1$ modulo M.
3. Either $X-1$ or $X+1$ is a multiple of one of p, q **but not both**.

If our value of a fails either of these ("unlucky"), we just try again from the start of guessing $a < M$.

Our treatment (blog post and chapter 12) also desires r to be a multiple of $p-1$ or $q-1$. It can be shown that many a give this "helpful" property, which requires $r \geq \sqrt{(p-1)(q-1)} \approx \sqrt{M}$.

(It is not clear whether we show this. It could be an exercise: Consider numbers r that divide a product mn of two nearly-equal composite numbers. Conditioned on $r \geq \min\{m, n\}$, give a lower bound for the proportion that are a multiple of m or a multiple of n. Note that m and n need not be themselves relatively prime; $p-1$ and $q-1$ are both even, for instance. It would still need to be argued that most a give such an r. But I am not sure that the "helpful" property is needed either.)

Chapter 12 does handle the argument in property 3, given that r is "helpful"---which also subsumes issue 1 since $p-1$ and $q-1$ are even. Issue 2 is handled by a random argument.

We will see that the closer r is to $\sqrt{M}$ as opposed to being order-of M, the more challenging for a potential classical simulation of Shor's algorithm.

Another thing to observe is that when M is a **Blum integer**, meaning p and q are both congruent to 3 modulo 4, then $(p-1)(q-1)$ is divisible by 4 but no higher even number. There are always four square roots of 1 modulo $M = pq$, so we need to argue that the a's such that $a^{r/2}$ is one of the good ones are as plentiful as the bad ones. (Note that r depends only on a.) Here is an example for the smallest Blum integer: $21 = 3*7$. The **quadratic residues** are:

$1:1,\quad 2:4,\quad 3:9,\quad 4:16,\quad 5:4,\quad 6:15,\quad 7:7,\ 8:1,\quad 9:18,\quad 10:16,$
$20:1,\ 19:4,\ 18:9,\ 17:16,\ 16:4,\ 15:15,\ 14:7,\ 13:1,\ 12:18,\ 11:16$

Now $(p-1)(q-1) = 12$. The numbers $Y = 8-1, 8+1, 13+1$, and $13-1$ all give a factor via $\gcd(21, Y)$.

$a = 1$: $r = 1$; of course doesn't work.
$a = 2:\ 2, 4, 8, 16, 11, 1.$ Works
$a = 4:\ 16, 1$ (period 3 is odd)
$a = 5:\ 4, 20, 16, 17, 1$; doesn't work because $20 \equiv -1$.
$a = 8:\ 8^2 \equiv 1$. Period $r = 2$ is "helpful" and $8^{r/2} = 8$ is not -1. So works.
$a = 10:\ 16, 13, 4, 19, 1.$ Works
The other values are mirror images.

A more interesting Blum integer IMHO is $77 = 7*11$. Then $(p-1)(q-1) = 60$. "Helpful" means the period is a multiple of 6 or of 10. Note: $34^2 = 1156 = 77*15 + 1$ is a nontrivial square root of 1 and $43^2 = 1849 = 77*24 + 1$ is the other one. Does 2 work?

$2 : 4, 8, 16, 32, 64, 51, 25, 50, 23, 46, 15, 30, 60, 43, 9, 18, 36, 72, 67, 57, 37, 74,$ etc.: yes.

The next question is whether it is OK for the quantum part to obtain a multiple $r' = br$ of a helpful r. If b is even than certainly not, because $a^{r'/2}$ will be 1. But if b is odd---? In any event, we can obviate this question because we can single out the minimum r with sufficiently high probability.

The key auxiliary technical notion is a number x that is "good" to help find r.

11.2 Good Numbers

Let Q be a power of two, $Q = 2^\ell$, such that $M^2 \leq Q < 2M^2$. Say an integer x in the range $0, 1, \ldots, Q-1$ is **good** provided there is an integer t relatively prime to the period r such that

$$tQ - xr = k, \quad \text{where} \quad -r/2 \leq k \leq r/2. \tag{11.1}$$

The first key part (used later) is the multiple t of Q being relatively prime to r. The second key part is that there is a 1-to-1 correspondence between t's and good x's. So the number of good x's equals the size of G_r. Now unlike with $|G_M| = (p-1)(q-1)$, which is $\sim M$, we don't know $|G_r|$ since r could have any manner of factors. But there is a bound that is almost as good as proportionality:

If $tQ = k \bmod' r$, where $\bmod'$ means using $[-r/2, r/2]$ rather than $[0, r-1]$ for the modular values, then we get $tQ = k + xr$ for some unique x, where $-r/2 \leq k \leq r/2$.

LEMMA 11.1 There are $\Omega(\frac{r}{\log\log r})$ good numbers.

Proof. The key insight is to think of equation (11.1) as an equation modulo r. Then it becomes

$$tQ \equiv k \bmod r,$$

where $-r/2 \le k \le r/2$. But as t varies from 0 to $r-1$, the value of k can be arranged to be always in this range, so the only constraint on t is that it must be relatively prime to r. The number of values t that are relatively prime to r defines Euler's *totient* function, which is denoted by $\phi(r)$. Note that for each value of t there is a different value of x, so counting ts is the same as counting xs. Thus, the lemma reduces to a lower bound on Euler's function. But it is known that

$$\phi(z) = \Omega\left(\frac{z}{\log\log z}\right).$$

Indeed, the constant in Ω approaches $e^{-\gamma}$, where $\gamma = 0.5772156649\ldots$ is the famous Euler-Mascheroni constant. In any event, this proves the lemma. □

The general drift is that a good x gives a good chance of finding r exactly, by purely classical means. Of note:

If r is close to M, then by choosing Q close to M rather than M^2, we would stand a good chance of finding a good x just by picking about $\log \ell$-many of them classically at random. However, this does not help when r is smaller. The genius of Shor's algorithm is that the quantum Fourier transform can be used to drive amplitude toward good numbers in all cases.

This makes $r \approx M^{1-\epsilon}$ where $0 < \epsilon < 1$ the "vat" of hard cases: too sparse to guess at random. For the quantum part, however, we need $Q > rM$.

LEMMA 11.7 If x is good, then in classical polynomial time, we can determine the value of r.

Proof. Recall that x being good means that there is a t relatively prime to r so that (by symmetry)

$$xr - tQ = k \qquad \text{where} \qquad -\frac{r}{2} \leq k \leq \frac{r}{2}.$$

Assume that $k \geq 0$; the argument is the same in the case where it is negative. We can divide by rQ and get the equation

$$\left| \frac{x}{Q} - \frac{t}{r} \right| \leq \frac{1}{2Q}.$$

We next claim that r and t are unique. Suppose there is another t'/r'. Then

$$\left| \frac{t}{r} - \frac{t'}{r'} \right| \geq \frac{1}{rr'} \geq \frac{1}{M^2}.$$

But then both fractions are close, which makes Q smaller than M^2, a contradiction.

Because r is unique, it follows that t is too. So we can treat

$$xr - tQ = k$$

as an integer program in a fixed number of variables: the variables are r, t, and two slack variables used to state

$$-r/2 \leq k \leq r/2$$

as two equations. While integer programs are hard in general, for a fixed number of variables they are solvable in polynomial time. This proves the lemma. □

Simulation Interlude

Before we go to this analysis, let's see a brute-force simulation of Shor's algorithm. It pretty much builds the concrete "mazes" for $\ell + n$ qubits and simulates all the legal "Feynman mouse paths" through them. The run of my simulator on $M = 21$ and $a = 5$ succeeded on the second try:

```
About to do try 1 of sampling QFT applied to 1010101011010010100 with status now PROBS_ENUMERA
Sampling with status PROBS_ENUMERATED:
Base probability for conditionals: 0.166015625000
Current: 0 with probability 0.083007813 on rolling 0.325191374; last 0 prob = 0.500000000
Current: 00 with probability 0.055282593 on rolling 0.563273639; last 0 prob = 0.665992647
Current: 001 with probability 0.027659269 on rolling 0.559076137; last 0 prob = 0.499674899
Current: 0010 with probability 0.027418884 on rolling 0.941772811; last 0 prob = 0.991309060
Current: 00101 with probability 0.027183985 on rolling 0.139894580; last 0 prob = 0.008567052
Current: 001010 with probability 0.026380861 on rolling 0.938149097; last 0 prob = 0.970455980
Current: 0010101 with probability 0.025648040 on rolling 0.595421001; last 0 prob = 0.02777850
Current: 00101010 with probability 0.020074378 on rolling 0.114898273; last 0 prob = 0.7826866
Current: 001010101 with probability 0.018908726 on rolling 0.791199151; last 0 prob = 0.058066
sampled output vector: 00101010110100
time cost: 1.23308 milliseconds.

Measured 001010101 as 85 giving 0.166015625
Fractional approximation is 1/6
; Possible period is 6
; Unable to determine factors, we'll try again.
Let's take a free random crack at it without the QFT application...
Fractional approximation is 2/3
; Odd denominator, trying to expand by 2.
; Possible period is 6
; Unable to determine factors, we'll try again.

About to do try 2 of sampling QFT applied to 1010101011010010100 with status now PROBS_ENUMERA
Sampling with status PROBS_ENUMERATED:
Base probability for conditionals: 0.166015625000
Current: 1 with probability 0.083007813 on rolling 0.527169932; last 0 prob = 0.500000000
Current: 10 with probability 0.055282593 on rolling 0.051374227; last 0 prob = 0.665992647
Current: 100 with probability 0.027623324 on rolling 0.277237177; last 0 prob = 0.499674899
Current: 1000 with probability 0.027576410 on rolling 0.189192738; last 0 prob = 0.998301645
Current: 10000 with probability 0.027567765 on rolling 0.562397971; last 0 prob = 0.999686499
Current: 100000 with probability 0.027564179 on rolling 0.523783427; last 0 prob = 0.999869929
Current: 1000000 with probability 0.027562462 on rolling 0.694951445; last 0 prob = 0.99993770
Current: 10000000 with probability 0.027561612 on rolling 0.646817553; last 0 prob = 0.9999691
Current: 100000000 with probability 0.027561188 on rolling 0.353241189; last 0 prob = 0.999984
sampled output vector: 10000000010100
time cost: 1.2329 milliseconds.

Measured 100000000 as 256 giving 0.500000000
Fractional approximation is 1/2
; Possible period is 2
; Success: 21 = 3 * 7
Success after 2 xy sample(s) plus 2 QFT sample(s).
```

[Show demo]