Shor's Algorithm

NYC Quantum Meetup
12/20/2017

FACTORING NUMBERS WITH PERIOD ESTIMATION

MUIR KUMPH

IBM RESEARCH 2017

Shor’s Algorithm

a quantum algorithm which can factor numbers

- Shor’s algorithm uses period finding and the quantum Fourier transform (QFT) to factor numbers
- It is a probabilistic algorithm and it succeeds more than 50% of the time
- If run on a quantum computer, it would take of order $O(logN)^2$... quantum gates to find the factors of an integer *N*

SIAM REVIEW
Vol. 41, No. 2, pp. 303–332

Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer*

Peter W. Shor†

A little background

math and qubit concepts

- numbers can be stored in base 2 using either bits or qubits
  - classical bits: 10 decimal is 1010 in base 2
  - qubits: 10 in base 2 is |1010>
- qubits can also be in a superposition of all their states
  - ie. c|0> + d|1>, where c and d are complex numbers
- modulo arithmetic is one where there is a maximum number
  - *modulo 15* means that there is no number above 15
  - (14 + 1) mod 15 = 0
  - can also be written (14+1)%15 = 0
  - eg. (14+5)%15=4
- greatest common divider (gcd)
  - gcd(15, 70)=5
  - 15=3x5, 70=2x5x7

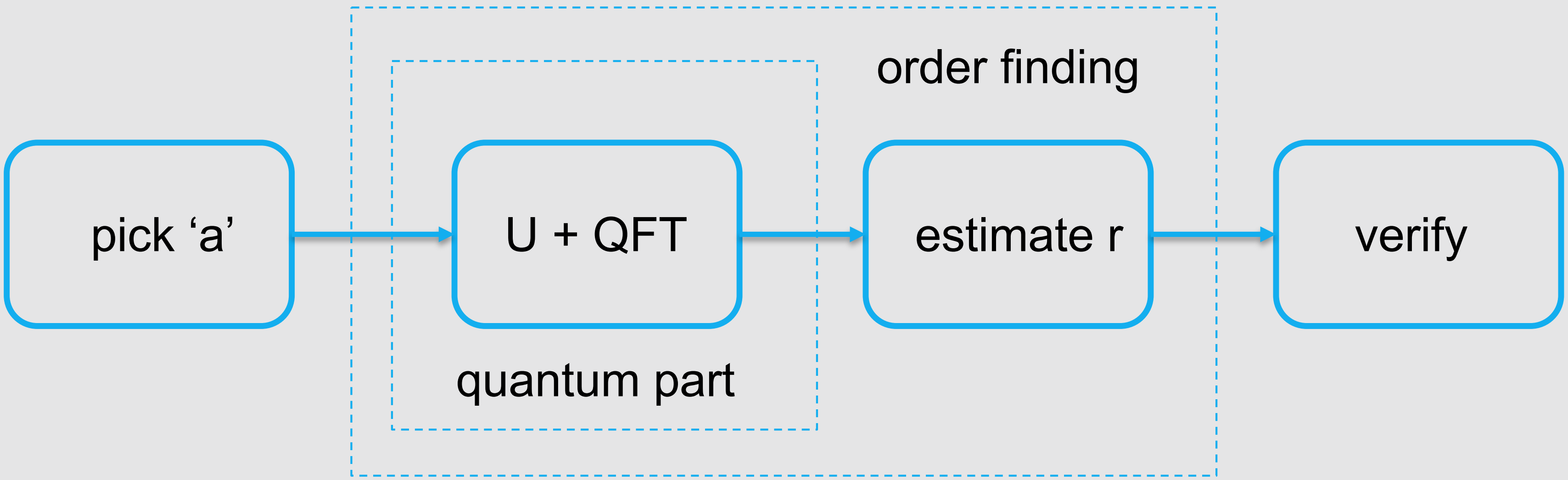
Steps of the Algorithm

How to factor a number N:

FROM WIKIPEDIA:

1. Pick a random number a < N.
2. Compute gcd(a, N).
3. If gcd(a, N) ≠ 1, then this number is a nontrivial factor of N, so we are done.
4. Otherwise, use the period-finding subroutine to find r, the period of the following function: $f(x)=a^x \bmod N$, i.e. the order r of a in $(\mathbb{Z}_N)^\times$, which is the smallest positive integer r for which $f(x+r)=f(x)$, or $f(x+r)=a^{x+r} \bmod N \equiv a^x \bmod N$.
5. If r is odd, go back to step 1.
6. If $a^{r/2} \equiv -1$ (mod N), go back to step 1.
7. $gcd(a^{r/2}+1,N)$ and $gcd(a^{r/2}-1,N)$ are both nontrivial factors of N.

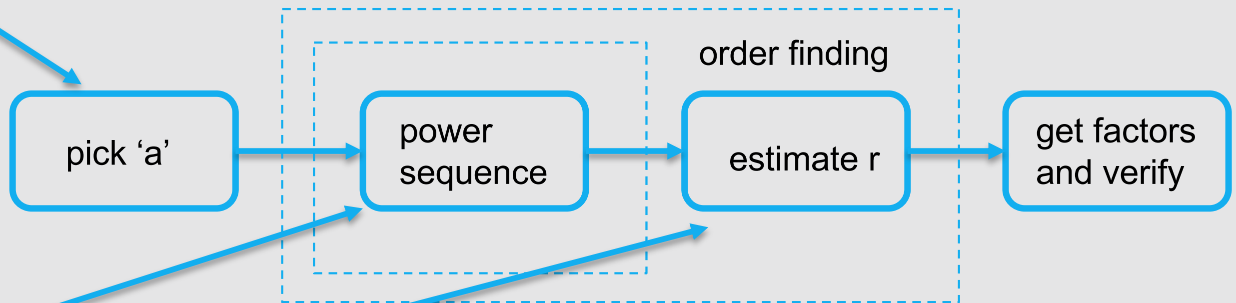
Algorithm Flow



Steps of the Algorithm

no quantum

For example:
to factor N=15
pick a=7

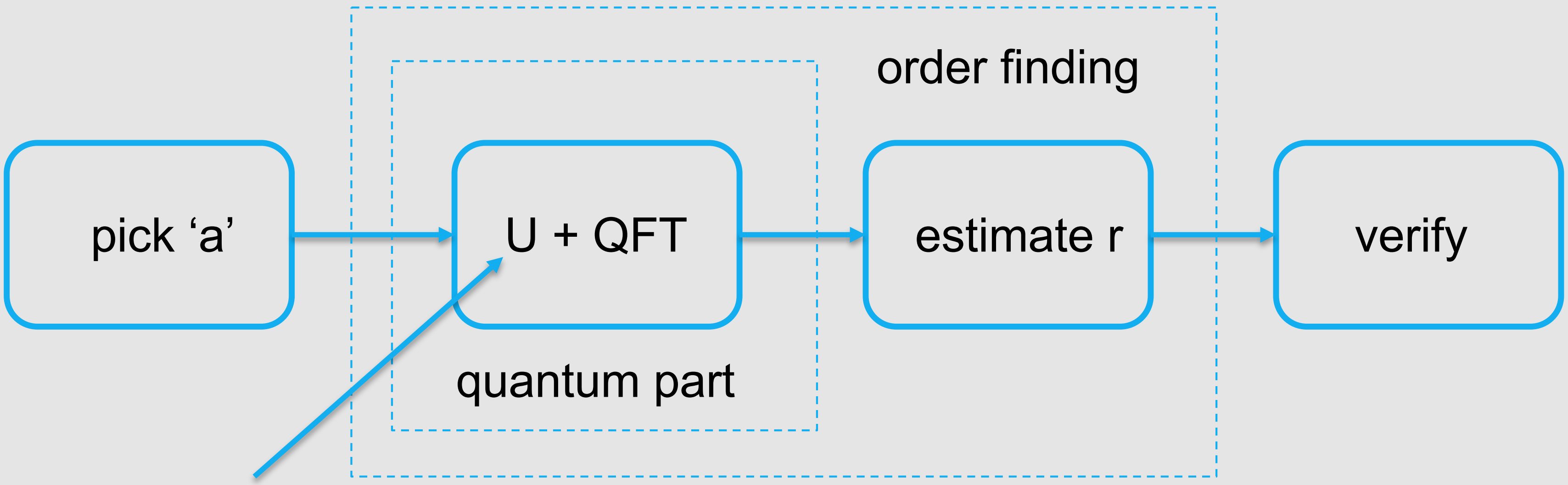


$a^0=1$
$a^1=7$
$a^2= 49\%15 = 4$
$a^3= 343\%15 =13$
$a^4= 2401\%15 =1$

r=4

$gcd(a^{(r/2)}+/-1, N)$ -> gcd(50, 15), gcd(48,15) -> 5, 3
5x3=15

Algorithm Flow



apply a gate which allows one to find the order r of (a mod N)

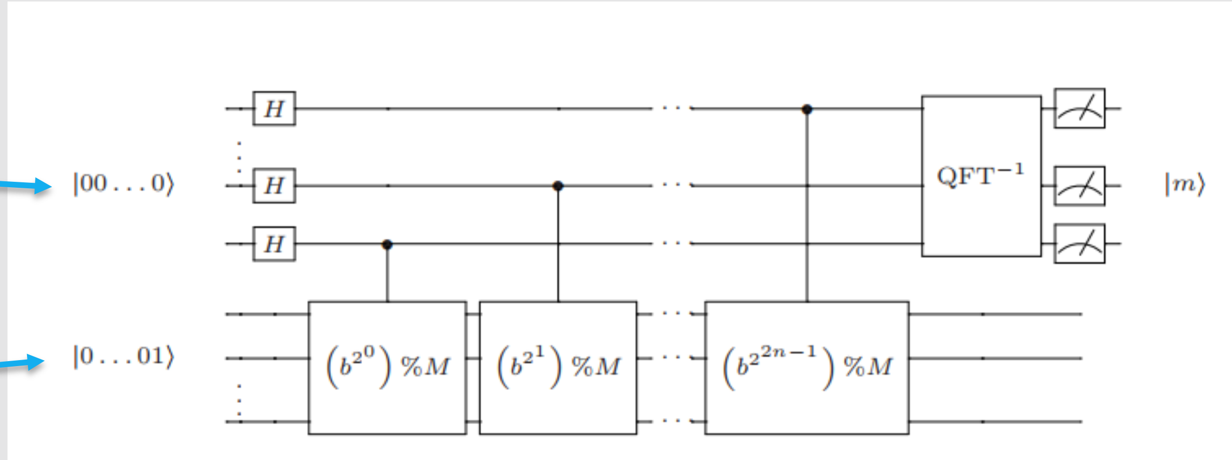
Replace Classical Order Finding with Quantum Methods

qubit registers are divided into two parts

top: phase estimation

bottom: modulo arithmetic

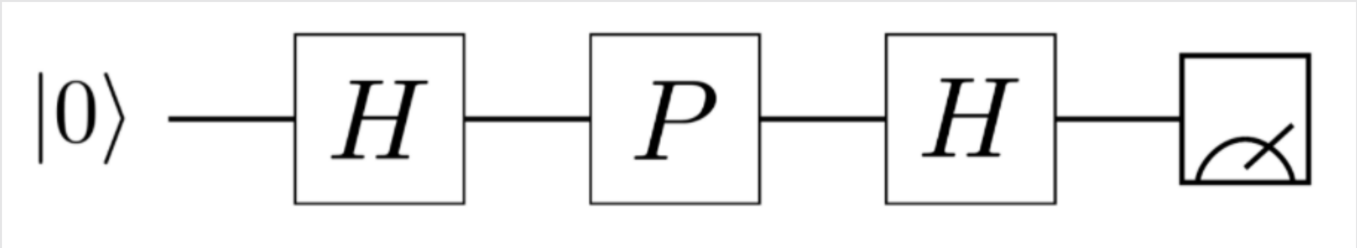
once, N and a have been chosen, the quantum circuit looks like



The phase estimation part

phase estimation is getting the period of a function

a simple phase estimation circuit:



H (Hadamard gate) takes the qubit into superposition state |0> + |1>
P (applies a phase)
H (Hadamard gate) takes the qubit back to |0> if no phase
H is the quantum Fourier transform for a single qubit

Consider the single qubit phase gate (Z)

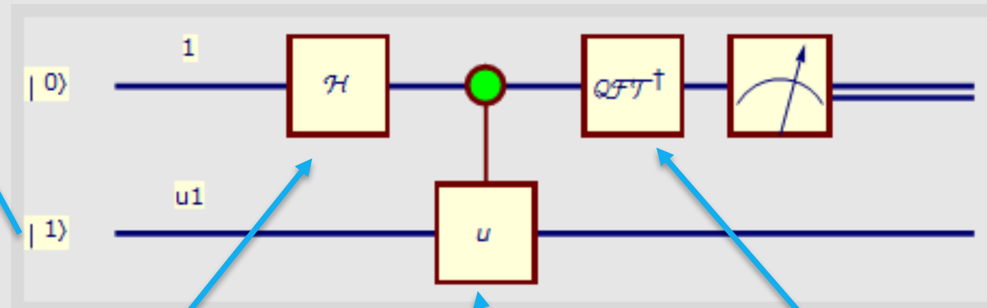
Eigenvalue	Eigenvector
-1	$\mid 1\rangle$
1	$\mid 0\rangle$

This gate will map

|0> + |1> to

|0> - |1>

to measure the phase of it, one needs a controlled version. – see to the right



|01> -> (|01> + |11>) -> (|01> - |11>) -> |11>

What do phases and order finding have to do with each other?

for eigenstates:

U|q>=u|q>

U is an operator

u is a number

the equation to the left is true for eigenstates |q> of the operator **U**

U is an operator which maps one quantum state to another
for certain quantum states |q> (eigenstates), the state remains the same except for a constant factor u (the eigenvalue)

the eigenvalue can be a complex number

if **U** is a quantum operator which multiplies the state by 'a' modulo N, then for eigenstates, the phase φ it accumulates on a single application of u is $\frac{2k\pi}{r}$, where k is some integer between 0 and r

$$\mathrm{u} = e^{-i\varphi}$$

Consider the problem of trying to factor 15 – it's almost trivial

for Shor's algorithm we need to pick 'a', in this example we use a=7

then we need a modulo arithmetic order finding gate u

it's already clear that r=4, because we can see under the hood of the algorithm

	Input	Output
0	$\vert 0000\rangle$	$\vert 0000\rangle$
1	$\vert 0001\rangle$	$\vert 0111\rangle$
2	$\vert 0010\rangle$	$\vert 1110\rangle$
3	$\vert 0011\rangle$	$\vert 0110\rangle$
4	$\vert 0100\rangle$	$\vert 1101\rangle$
5	$\vert 0101\rangle$	$\vert 0101\rangle$
6	$\vert 0110\rangle$	$\vert 1100\rangle$
7	$\vert 0111\rangle$	$\vert 0100\rangle$
8	$\vert 1000\rangle$	$\vert 1011\rangle$
9	$\vert 1001\rangle$	$\vert 0011\rangle$
10	$\vert 1010\rangle$	$\vert 1010\rangle$
11	$\vert 1011\rangle$	$\vert 0010\rangle$
12	$\vert 1100\rangle$	$\vert 1001\rangle$
13	$\vert 1101\rangle$	$\vert 0001\rangle$
14	$\vert 1110\rangle$	$\vert 1000\rangle$
15	$\vert 1111\rangle$	$\vert 1111\rangle$

$$\begin{aligned} 7^2 &= 4 \pmod{15} \\ 7^3 &= 4 \cdot 7 = 13 \pmod{15} \\ 7^4 &= 13 \cdot 7 = 1 \pmod{15} \end{aligned}$$

$$u : \begin{cases} |1\rangle \to |7\rangle \to |4\rangle \to |13\rangle \to |1\rangle \\ |2\rangle \to |14\rangle \to |8\rangle \to |11\rangle \to |2\rangle \end{cases}$$

Multiplication by 7 modulo 15

Order finding 7 modulo 15

We can also look at the eigenvalues and vectors of u

we can use phase estimation to determine what the eigenvalues of these eigenvectors are

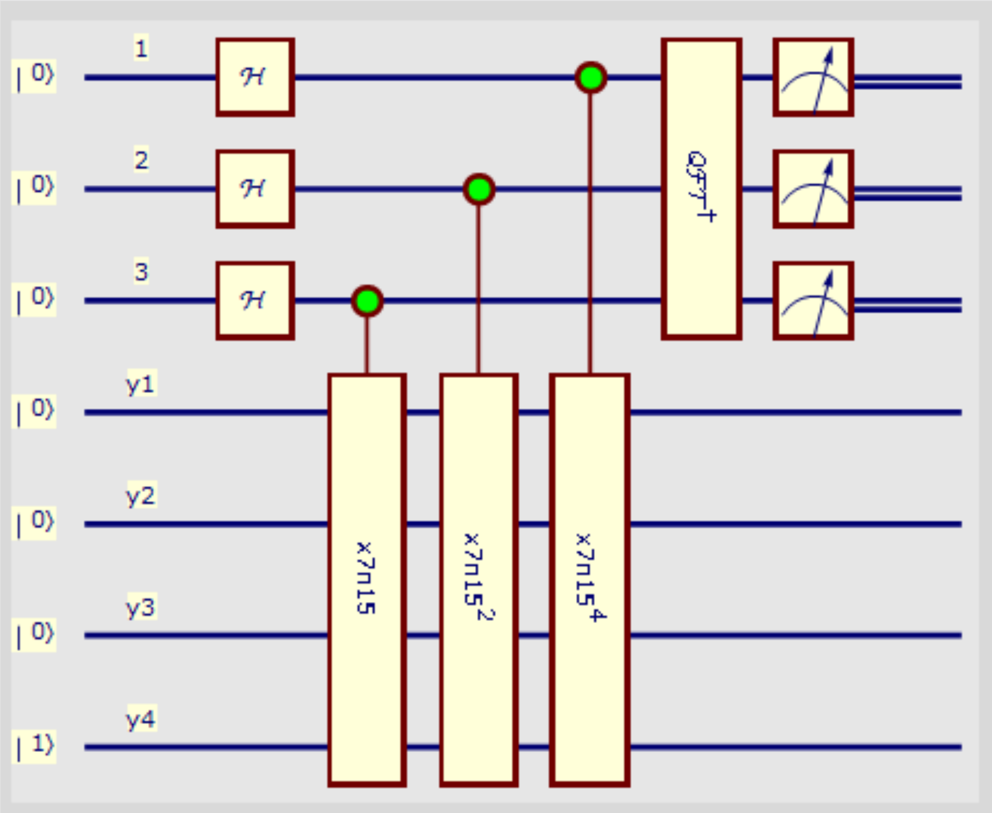
the phases of the eigenvalues are in the form $\frac{2k\pi}{r}$, where r=4

Eigenvalue	Eigenvector
-1	$-\frac{1}{2}\ \|0010\rangle - \frac{1}{2}\ \|1000\rangle + \frac{1}{2}\ \|1011\rangle + \frac{1}{2}\ \|1110\rangle$
-1	$-\frac{1}{2}\ \|0001\rangle - \frac{1}{2}\ \|0100\rangle + \frac{1}{2}\ \|0111\rangle + \frac{1}{2}\ \|1101\rangle$
-1	$\frac{1}{2}\ \|0011\rangle - \frac{1}{2}\ \|0110\rangle - \frac{1}{2}\ \|1001\rangle + \frac{1}{2}\ \|1100\rangle$
i	$\frac{1}{2} i\ \|0010\rangle - \frac{1}{2} i\ \|1000\rangle - \frac{1}{2}\ \|1011\rangle + \frac{1}{2}\ \|1110\rangle$
i	$-\frac{1}{2} i\ \|0001\rangle + \frac{1}{2} i\ \|0100\rangle - \frac{1}{2}\ \|0111\rangle + \frac{1}{2}\ \|1101\rangle$
i	$-\frac{1}{2}\ \|0011\rangle + \frac{1}{2} i\ \|0110\rangle - \frac{1}{2} i\ \|1001\rangle + \frac{1}{2}\ \|1100\rangle$
$-i$	$-\frac{1}{2} i\ \|0010\rangle + \frac{1}{2} i\ \|1000\rangle - \frac{1}{2}\ \|1011\rangle + \frac{1}{2}\ \|1110\rangle$
$-i$	$\frac{1}{2} i\ \|0001\rangle - \frac{1}{2} i\ \|0100\rangle - \frac{1}{2}\ \|0111\rangle + \frac{1}{2}\ \|1101\rangle$
$-i$	$-\frac{1}{2}\ \|0011\rangle - \frac{1}{2} i\ \|0110\rangle + \frac{1}{2} i\ \|1001\rangle + \frac{1}{2}\ \|1100\rangle$
1	$-\ \|1111\rangle$
1	$-\frac{1}{2}\ \|0010\rangle - \frac{1}{2}\ \|1000\rangle - \frac{1}{2}\ \|1011\rangle - \frac{1}{2}\ \|1110\rangle$
1	$-\frac{1}{2}\ \|0001\rangle - \frac{1}{2}\ \|0100\rangle - \frac{1}{2}\ \|0111\rangle - \frac{1}{2}\ \|1101\rangle$
1	$-\frac{1}{2}\ \|0011\rangle - \frac{1}{2}\ \|0110\rangle - \frac{1}{2}\ \|1001\rangle - \frac{1}{2}\ \|1100\rangle$
1	$-\ \|1010\rangle$
1	$-\ \|0101\rangle$
1	$-\ \|0000\rangle$

Order finding 7 mod 15

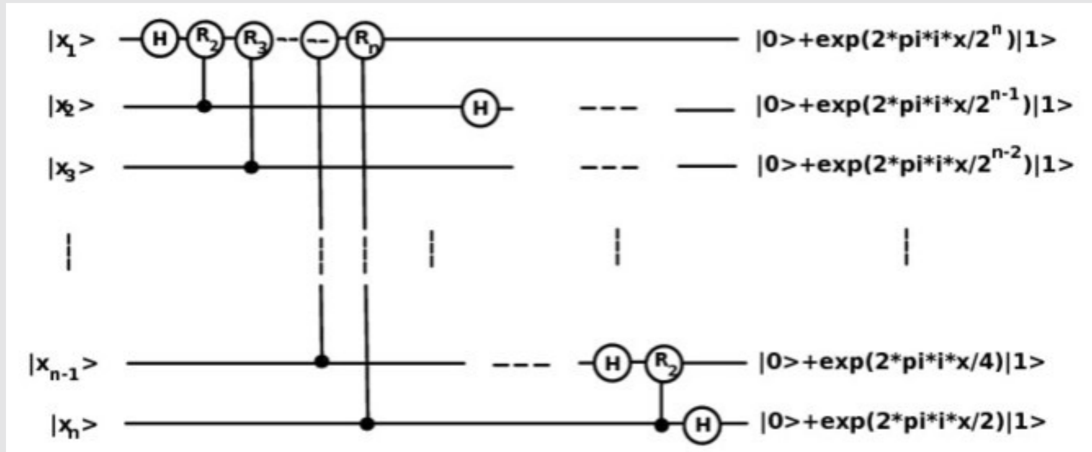
u="x7n15" is the modulo arithmetic, which is controlled on the state of the phase estimation qubits

the order finding is done in even powers of u: u^1, u^2, u^4, u^8 .... doubling the precision of the phase estimation with each qubit



QFT

the quantum Fourier transform (QFT) is the quantum analogue of the discrete Fourier transform (DFT)



$$QFT(|x_1 x_2 \dots x_n\rangle) = \frac{1}{\sqrt{N}} \left(|0\rangle + e^{2\pi i\,[0.x_n]}|1\rangle\right) \otimes \left(|0\rangle + e^{2\pi i\,[0.x_{n-1}x_n]}|1\rangle\right) \otimes \cdots \otimes \left(|0\rangle + e^{2\pi i\,[0.x_1 x_2 \dots x_n]}|1\rangle\right)$$

each qubit gives a binary increase in the precision of the Fourier transform

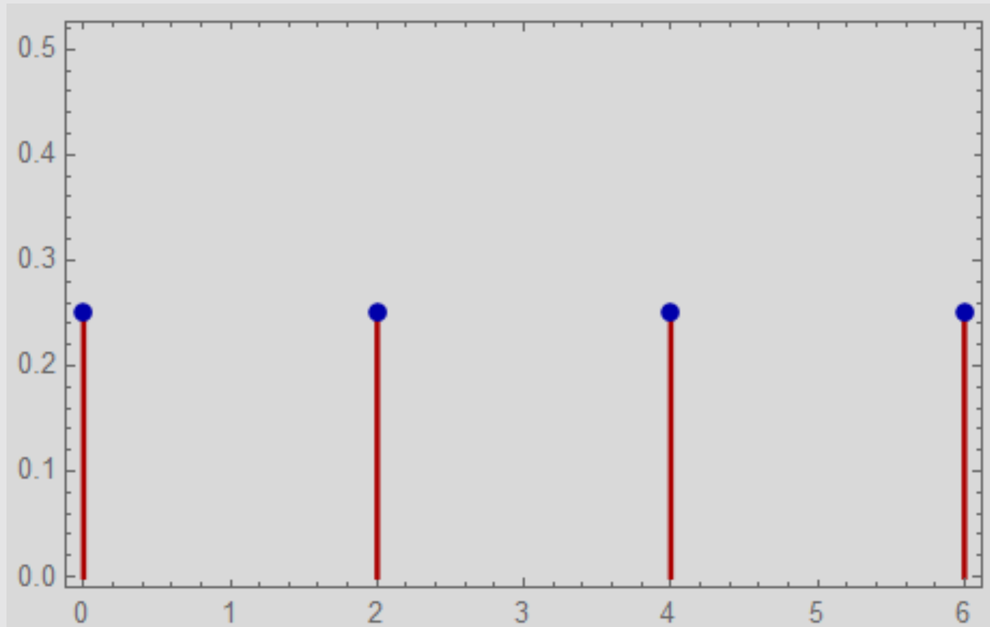
$$QFT_2 = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{bmatrix}$$

$$QFT_4 = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & \frac{i}{2} & -\frac{1}{2} & -\frac{i}{2} \\ \frac{1}{2} & -\frac{1}{2} & \frac{1}{2} & -\frac{1}{2} \\ \frac{1}{2} & -\frac{i}{2} & -\frac{1}{2} & \frac{i}{2} \end{bmatrix}$$

simulation results of circuit

measurement on the phase estimation qubits would give one of 4 possible outcomes

for single runs of the algorithm, half the time you might think that r=2 because the phase was π

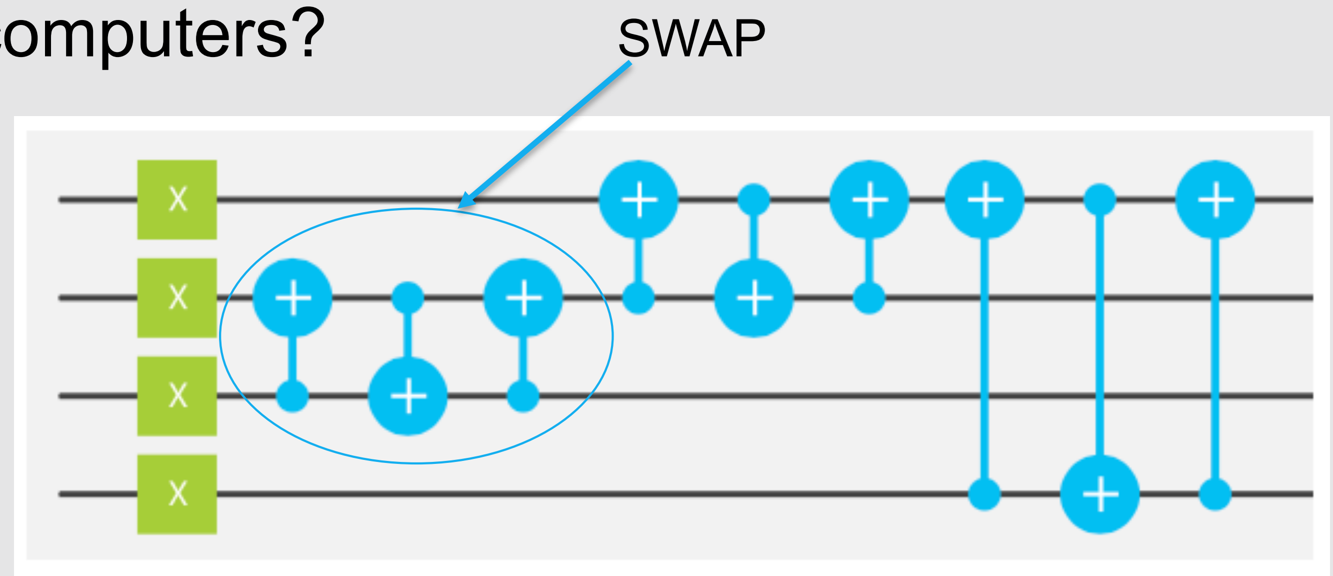


Probability	Measurement	State
0.25	$(0_1 \ 0_2 \ 0_3)$	$0.5(\lvert 0000001\rangle) + 0.5(\lvert 0000100\rangle) + 0.5(\lvert 0000111\rangle) + 0.5(\lvert 0001101\rangle)$
0.25	$(0_1 \ 1_2 \ 0_3)$	$-(0. + 0.5 i)(\lvert 0100111\rangle) + (0. + 0.5 i)(\lvert 0101101\rangle) + 0.5(\lvert 0100001\rangle) - 0.5(\lvert 0100100\rangle)$
0.25	$(1_1 \ 0_2 \ 0_3)$	$0.5(\lvert 1000001\rangle) + 0.5(\lvert 1000100\rangle) - 0.5(\lvert 1000111\rangle) - 0.5(\lvert 1001101\rangle)$
0.25	$(1_1 \ 1_2 \ 0_3)$	$(0. + 0.5 i)(\lvert 1100111\rangle) - (0. + 0.5 i)(\lvert 1101101\rangle) + 0.5(\lvert 1100001\rangle) - 0.5(\lvert 1100100\rangle)$
Probability	Measurement	State

what does x7n15 look like on today's small quantum computers?

but in order to do phase estimation, we would need a controlled version of this circuit

each of the SWAP gates shown here would be replaced with a controlled SWAP (aka FREDKIN) gate to make this a controlled modulo arithmetic



- this is a highly optimized version only valid for a=7, N=15
- in general one would need to build adders, multipliers and then exponential circuits from discrete quantum logic gates

For further reading

- Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer
  - https://arxiv.org/abs/quant-ph/9508027
- Quantum Experience Users Guide to Shor’s Algorithm:
  - https://quantumexperience.ng.bluemix.net/proxy/tutorial/full-user-guide/004-Quantum_Algorithms/110-Shor's_algorithm.html
- Mathematica Add-on for Quantum Mechanics and Quantum Computing
  - http://homepage.cem.itesm.mx/jose.luis.gomez/quantum/
- A 2D Nearest-Neighbor Quantum Architecture for Factoring in Polylogarithmic Depth
  - https://arxiv.org/abs/1207.6655
- Constant-Optimized Quantum Circuits for Modular Multiplication and Exponentiation
  - https://arxiv.org/abs/1202.6614
- Realization of a scalable Shor algorithm
  - https://arxiv.org/pdf/1507.08852.pdf
- Wikipedia
  - https://en.wikipedia.org/wiki/Shor%27s_algorithm

To Do for Next Time

Qiskit has Toffoli gates and QFT built-in

- Show 2nd bit on QFT
- Show how to eliminate most of the phase estimation qubits
  - Kitaev QFT
- The u=7mod15 can then be run on a 5 qubit machine
- Demo Shor's Algorithm with Qiskit
- Deutsch-Jozsa Algorithm

IBM

End of Shor's Algorithm