Seminar 4

Fourier Transform and its Applications

Before going in…

- Today’s seminar is based on chapter 5 in Nielsen & Chuang.
- We are now looking some specific algorithms and their applications today.
- As in the last seminar, there are exercises. The incorporation method is the same as before.

Before Going in…

- There are some prerequisites:
- 1. You must at least understand what FFT does.
- 2. Small knowledge in Number Theory will be useful.

- Don’t worry! It’s not that hard..

Discrete Fourier Transforms

- $y_k = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} e^{2\pi ijk/N} x_j$
- I will not prove how this formula is justified. (Out of Scope!)
- For Quantum Computers:
- $|j\rangle \rightarrow \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} e^{2\pi ijk/N} |k\rangle$
- $\sum_{j=0}^{N-1} x_j |j\rangle \rightarrow \sum_{k=0}^{N-1} y_k |k\rangle$
- The coefficients are called ‘Amplitudes’. (Obviously.)

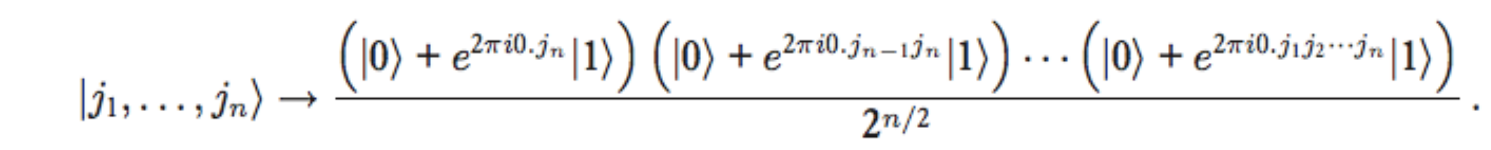
Product Representation

- Fourier Transform will be unitary….
- We can construct the circuit using basic gates… but how?

- Take $N = 2^n$. The basis ket goes from $|0\rangle \ldots |2^n - 1\rangle$
- Let’s represent j in binary: $j = j_1 j_2 j_3 \ldots j_n$
- Binary Fraction: $0.j_1 j_2 \ldots j_m$

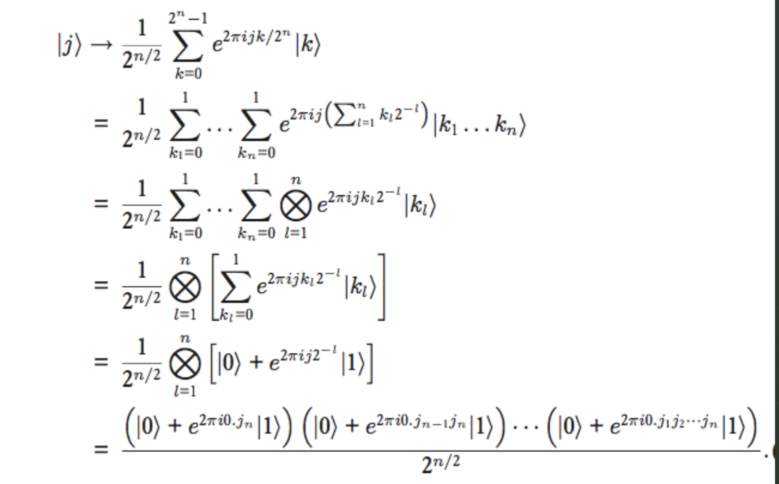
Product Representation

- Little algebra gives



$$|j_1,\ldots,j_n\rangle \rightarrow \frac{\left(|0\rangle + e^{2\pi i 0.j_n}|1\rangle\right)\left(|0\rangle + e^{2\pi i 0.j_{n-1}j_n}|1\rangle\right)\cdots\left(|0\rangle + e^{2\pi i 0.j_1 j_2 \cdots j_n}|1\rangle\right)}{2^{n/2}}.$$

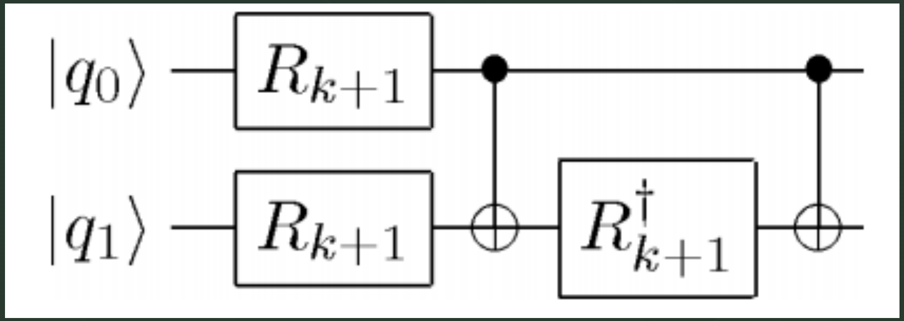
- Proof
- (Yeah.. I’m not doing that!)



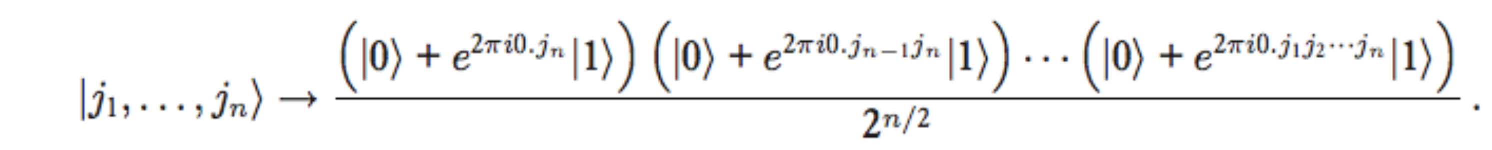
$$\begin{aligned}
|j\rangle &\rightarrow \frac{1}{2^{n/2}} \sum_{k=0}^{2^n-1} e^{2\pi i jk/2^n}|k\rangle \\
&= \frac{1}{2^{n/2}} \sum_{k_1=0}^{1} \cdots \sum_{k_n=0}^{1} e^{2\pi i j\left(\sum_{l=1}^{n} k_l 2^{-l}\right)}|k_1 \ldots k_n\rangle \\
&= \frac{1}{2^{n/2}} \sum_{k_1=0}^{1} \cdots \sum_{k_n=0}^{1} \bigotimes_{l=1}^{n} e^{2\pi i j k_l 2^{-l}}|k_l\rangle \\
&= \frac{1}{2^{n/2}} \bigotimes_{l=1}^{n} \left[\sum_{k_l=0}^{1} e^{2\pi i j k_l 2^{-l}}|k_l\rangle\right] \\
&= \frac{1}{2^{n/2}} \bigotimes_{l=1}^{n} \left[|0\rangle + e^{2\pi i j 2^{-l}}|1\rangle\right] \\
&= \frac{\left(|0\rangle + e^{2\pi i 0.j_n}|1\rangle\right)\left(|0\rangle + e^{2\pi i 0.j_{n-1}j_n}|1\rangle\right)\cdots\left(|0\rangle + e^{2\pi i 0.j_1 j_2 \cdots j_n}|1\rangle\right)}{2^{n/2}}.
\end{aligned}$$

The R_k Gate

- $R_k = \begin{bmatrix} 1 & 0 \\ 0 & e^{2\pi i/2^k} \end{bmatrix}$
- Since this is unitary, we can of course construct controlled-R_k with CNOT and basic single qubit gates.
- Will you give it a go? (MIT homework question!)



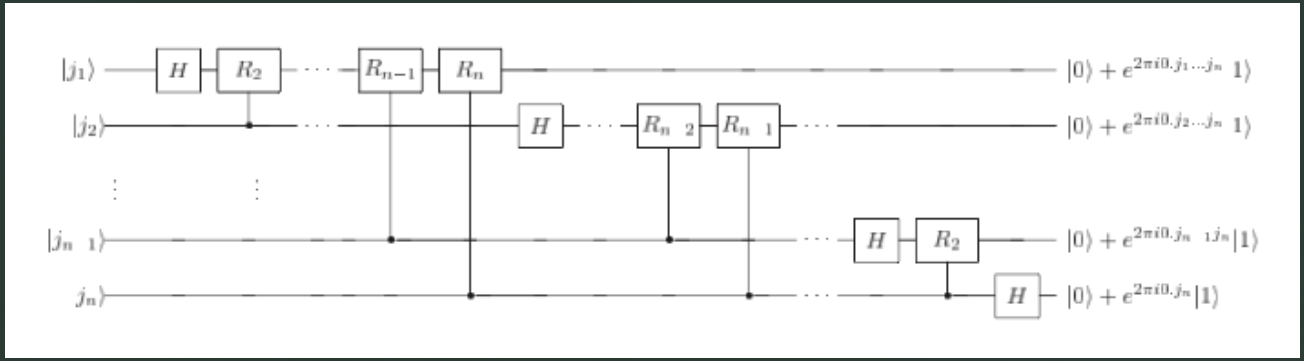
Quantum Fourier Algorithm



$$|j_1,\ldots,j_n\rangle \rightarrow \frac{\left(|0\rangle + e^{2\pi i 0.j_n}|1\rangle\right)\left(|0\rangle + e^{2\pi i 0.j_{n-1}j_n}|1\rangle\right)\cdots\left(|0\rangle + e^{2\pi i 0.j_1 j_2 \cdots j_n}|1\rangle\right)}{2^{n/2}}.$$

- How do we make each terms?
- 1. Apply the Hadamard gate to produce $\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$
- 2. Apply the controlled-R_k gate n-1 times for $|j_1\rangle$. (Control: j_2 to j_n, obviously)
- 3. Repeat for $|j_k\rangle$. For $|j_k\rangle$, apply the gate for n-k times.

Visualisation


$|j_1\rangle$
H
R_2
R_{n-1}
R_n
$|0\rangle + e^{2\pi i 0.j_1 \ldots j_n}|1\rangle$
$|j_2\rangle$
H
R_{n-2}
R_{n-1}
$|0\rangle + e^{2\pi i 0.j_2 \ldots j_n}|1\rangle$
$|j_{n-1}\rangle$
H
R_2
$|0\rangle + e^{2\pi i 0.j_{n-1}j_n}|1\rangle$
$|j_n\rangle$
H
$|0\rangle + e^{2\pi i 0.j_n}|1\rangle$

Note! The Hadamard gate MUST be applied at all basis kets!

Question.

- 1. Construct the 3-qubit QFT (Not Quantum Field Theory!) using H, S, and T gates.
- 2. Compare the efficiency of the classical FFT algorithm and the QFT algorithm. How many operations (approximately) should each algorithms do when they operate on n bases?

Phase Estimation

- A unitary operator can have some complex phases as the eigenvalue, with the phase value unknown.
- Of course, the overall process is close to an approximation, as we all know that the phase itself is an ‘exponential’ term.
- Our Goal: Shor’s Algorithm

The Oracle

- Sometimes called as a ‘black box’
- capable of preparing the state |u⟩ and performing the controlled-U^{2j} operation
- The usage of oracles are NOT algorithms; they are more close to subroutines or modules.
- They are normally combined with other procedures to perform some useful tasks. (e.g. Search Algorithms – Next Seminar!)

First Stage

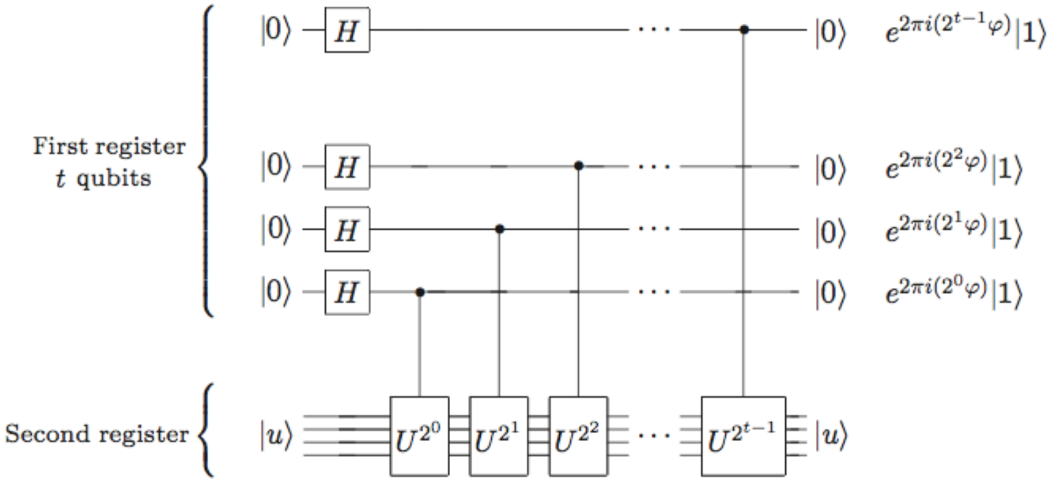
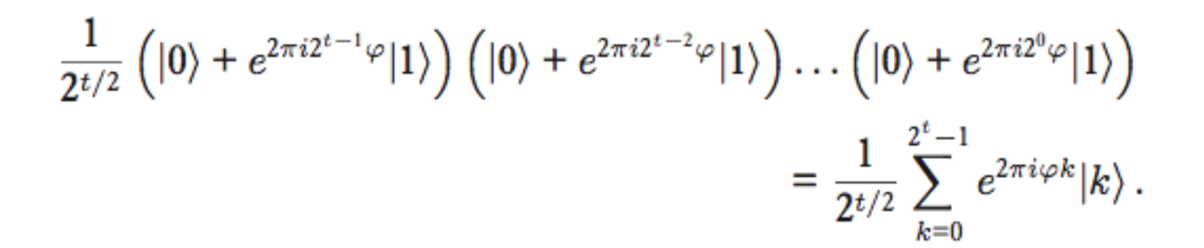

First register
t qubits
Second register
|0⟩
H
|u⟩
U^{2^0}
U^{2^1}
U^{2^2}
$U^{2^{t-1}}$
· · ·
$e^{2\pi i(2^{t-1}\varphi)}|1\rangle$
$e^{2\pi i(2^2\varphi)}|1\rangle$
$e^{2\pi i(2^1\varphi)}|1\rangle$
$e^{2\pi i(2^0\varphi)}|1\rangle$

Figure 5.2. The first stage of the phase estimation procedure. Normalization factors of $1/\sqrt{2}$ have been omitted, on the right.

First Stage

- Obviously, the overall state after the first state is given as



$$\frac{1}{2^{t/2}}\left(|0\rangle + e^{2\pi i 2^{t-1}\varphi}|1\rangle\right)\left(|0\rangle + e^{2\pi i 2^{t-2}\varphi}|1\rangle\right)\ldots\left(|0\rangle + e^{2\pi i 2^{0}\varphi}|1\rangle\right)$$

$$= \frac{1}{2^{t/2}}\sum_{k=0}^{2^t-1} e^{2\pi i\varphi k}|k\rangle .$$

- The last part is the product representation.

Second Stage

- Remember

$$|j_1,\ldots,j_n\rangle \to \frac{\left(|0\rangle + e^{2\pi i 0.j_n}|1\rangle\right)\left(|0\rangle + e^{2\pi i 0.j_{n-1}j_n}|1\rangle\right)\cdots\left(|0\rangle + e^{2\pi i 0.j_1 j_2\cdots j_n}|1\rangle\right)}{2^{n/2}}.$$

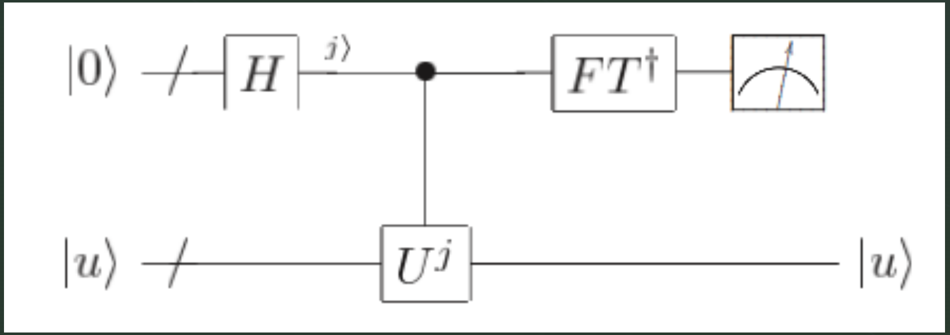
- When we mark $\varphi = \varphi_1\varphi_2 \ldots \varphi_t$, the formula we saw in the last slide changes to]

$$\frac{1}{2^{t/2}}\left(|0\rangle + e^{2\pi i 0.\varphi_t}|1\rangle\right)\left(|0\rangle + e^{2\pi i 0.\varphi_{t-1}\varphi_t}|1\rangle\right)\ldots\left(|0\rangle + e^{2\pi i 0.\varphi_1\varphi_2\cdots\varphi_t}|1\rangle\right)$$

- Okay! Let’s apply the inverse Quantum Fourier Transform and we are done!

Schematic Diagram

- Note. As you can easily deduce, this is an ESTIMATION of the phase until the nth order in binary.
- Of course, the more qubits you have in the first register, the more precise the measurement gets.*



* To acquire the phase accurately to n bits with success rate $1-\epsilon$ requires $t = n + \log(2 + \frac{1}{2\epsilon})$ registers.

RSA Encryption

- Easy explanation: The conventional algorithm for secure data transmission using number theory (i.e. Primes)
- It’s hard to factorise composites rather than to multiply primes!
- How much efficient?: The FASTEST classical factorisation algorithm operates at sub-exponential time scale.

Order-Finding

- Order (Number Theory): The order of x modulo N is defined as the smallest integer such that $x^r \equiv 1 \pmod{N}$
- Question: Prove that for n<r, x^ns have different modulos.
- Think of a unitary operator such tat U |y⟩ ≡ |xy(mod N)⟩
- IF y is larger than N, by convention, U is identity for y larger than N.

Order-Finding

- Think $|u_s\rangle = \frac{1}{\sqrt{r}}\sum_{k=0}^{r-1} e^{-2\pi isk/r}\left|x^k \text{ mod } N\right\rangle$.
- $U|u_s\rangle = \frac{1}{\sqrt{r}}\sum_{k=0}^{r-1} e^{-2\pi isk/r}\left|x^{k+1} \text{ mod } N\right\rangle = e^{2\pi is/r}|u_s\rangle$
- Okay! Now we have to approximate the eigenvalues!
- I know you will have questions regarding this formalism. Let me explain.

Computing the Modular Exponentiation Algorithm

- Question. Can we implement controlled-U^{2j} operations with reasonable (i.e. not exponential) amount of gates?
- It is. If you are curious, look at Box 5.2 at Nielsen & Chuang or http://qudev.phys.ethz.ch/content/courses/QSIT11/presentations/QSIT-ShorTheory.pdf (ETH Zurich lecture PPT)
- Result: The overall performance can be made using $O(L^3)$ gates.

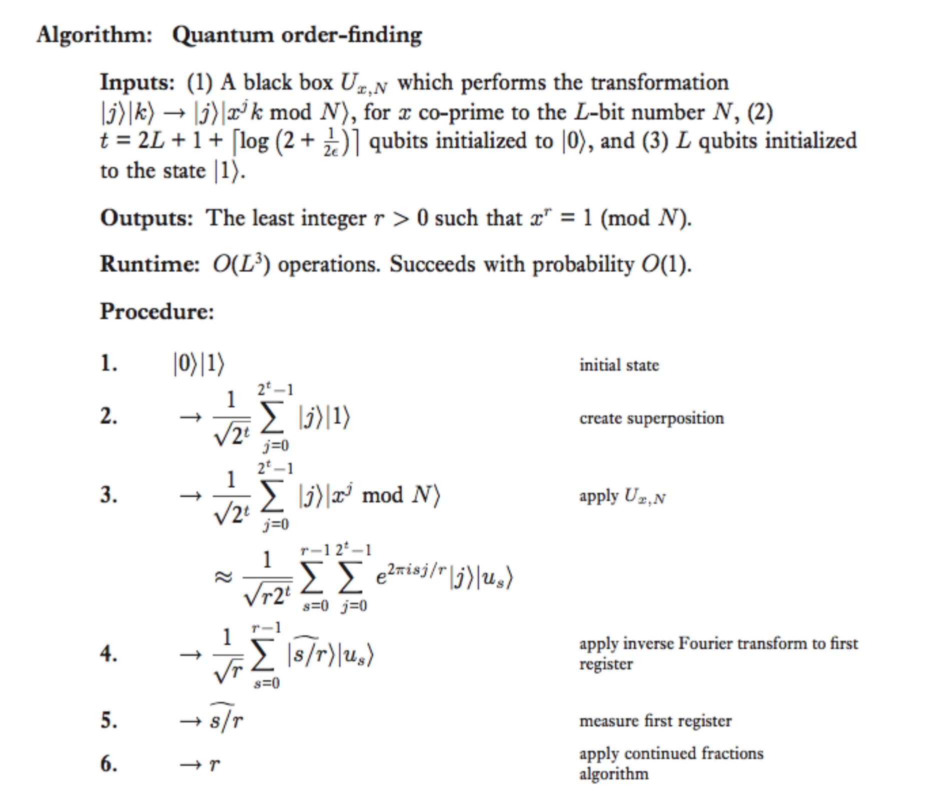
Preparing $|u_s\rangle$

- Wait..does preparing $|u_s\rangle$ require having knowledge on s?
- We can avoid this problem using $\frac{1}{\sqrt{r}}\sum_{s=0}^{r-1}|u_s\rangle = |1\rangle$.
- Apply the Phase Estimation Algorithm: for each s in the range 0 through r − 1, we will obtain an estimate of the phase φ ≈ s/r

Continued Fraction Algorithm

- Let’s reduce the order-finding algorithm to phase estimation.
- Continued Fraction: $[a_0, a_1, \dots, a_m] = a_0 + \frac{1}{a_1 + \frac{1}{a_2 + \frac{1}{\dots + \frac{1}{a_m}}}}$.
- Theorem. Suppose s/r is a rational number such that $\left|\frac{s}{r} - \varphi\right| < \frac{1}{2r^2}$. Then s/r is a convergent of the continued fraction for φ, and thus can be computed in O(L^3) operations using the continued fractions algorithm.
- Using this, s/r can be quickly approximated, thus giving r with a reasonable accuracy.

Summary



Algorithm: Quantum order-finding

Inputs: (1) A black box $U_{x,N}$ which performs the transformation $|j\rangle|k\rangle \to |j\rangle|x^j k \bmod N\rangle$, for x co-prime to the L-bit number N, (2) $t = 2L + 1 + \lceil \log(2 + \frac{1}{2\epsilon}) \rceil$ qubits initialized to $|0\rangle$, and (3) L qubits initialized to the state $|1\rangle$.

Outputs: The least integer $r > 0$ such that $x^r = 1 \pmod{N}$.

Runtime: $O(L^3)$ operations. Succeeds with probability $O(1)$.

Procedure:

1. $|0\rangle|1\rangle$ initial state

2. $\to \frac{1}{\sqrt{2^t}} \sum_{j=0}^{2^t-1} |j\rangle|1\rangle$ create superposition

3. $\to \frac{1}{\sqrt{2^t}} \sum_{j=0}^{2^t-1} |j\rangle|x^j \bmod N\rangle$ apply $U_{x,N}$

   $\approx \frac{1}{\sqrt{r2^t}} \sum_{s=0}^{r-1} \sum_{j=0}^{2^t-1} e^{2\pi i s j/r} |j\rangle|u_s\rangle$

4. $\to \frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} |\widetilde{s/r}\rangle|u_s\rangle$ apply inverse Fourier transform to first register

5. $\to \widetilde{s/r}$ measure first register

6. $\to r$ apply continued fractions algorithm

Factoring

- You all know what factoring is.
- Our task: Reduce the factoring problem to an order-finding problem.

- Step 1. Show that we can compute a factor of N if we can find a non-trivial solution $x \neq \pm 1 (\text{mod } N)$ to the equation $x^2 = 1 (\text{mod } N)$.
- Step 2. Show that randomly chosen y co-prime to N is quite likely to have an EVEN order, and $y^{r/2} \neq \pm 1 \ (\text{mod } N)$

Two Important Theorems

- Theorem 1. Suppose N is an L bit composite number, and x is a non-trivial solution to the equation $x^2 \ = \ 1(\mathrm{mod}\ N)$ in the range 1 ≤ x ≤ N. Then at least one of gcd(x−1,N) and gcd(x+1,N) is a non-trivial factor of N that can be computed using O(L^3) operations.
- Theorem 2. Suppose $N = p_1^{\alpha_1} p_2^{\alpha_2} \dots p_m^{\alpha_m}$, which is the prime factorisation of odd N. Let x be an integer chosen uniformly at random, subject to the requirements that 1 ≤ x ≤ N − 1 and x is co-prime to N . Let r be the order of x modulo N. Then

  $$p\left(r\ even\ and\ x^{\frac{r}{2}} \neq -1\ (\mathrm{mod}\ N)\right) \geq 1 - \frac{1}{2^m}.$$

Shor’s Algorithm

- 1. If N is even, return the factor 2.
- 2. Determine whether $N = a^b$. If so, return a. (This can be done using classical computers.)
- 3. Randomly choose x in the range 1 to N −1. If gcd(x, N) > 1 then return the factor gcd(x, N).
- 4. Use the order-finding subroutine to find the order r of x modulo N.
- 5. If r is even and $x^{r/2} \neq -1 \pmod{N}$, then compute $\gcd\left(x^{\frac{r}{2}} - 1, N\right), \gcd\left(x^{\frac{r}{2}} + 1, N\right)$ and test whether these have some non-trivial factors.

Validity of Shor’s Algorithm

- Step 1 & 2 either returns a factor, or provide that $N = p_1^{\alpha_1} p_2^{\alpha_2} \dots p_m^{\alpha_m}$.
- Step 3 and 4 generates the random number and computes the order r.
- For Step 5, the second theorem first guarantees that r is even and $x^{r/2} \neq -1 \pmod{N}$ with at least 1/2 chance.
- Then, the first theorem shows that either $\gcd\left(x^{\frac{r}{2}} - 1, N\right), \gcd\left(x^{\frac{r}{2}} + 1, N\right)$ will have a non-trivial factor.

Questions

- 1. Factoring 91: Suppose we wish to factor N = 91. Confirm that steps 1 and 2 are passed. For step 3, suppose we choose x = 4, which is co-prime to 91. Compute the order r of x with respect to N, and show that $x^{r/2}mod\,91\ =\ 64/=\ -1(\mathrm{mod}\,91)$, so the algorithm succeeds, giving gcd(64 − 1, 19) = 7.

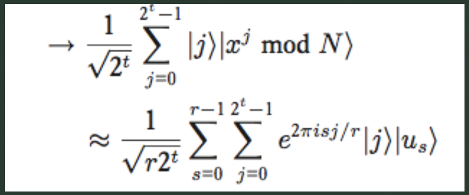
- 2. Using the contents you have learnt, factorise 15 quantum mechanically. This process was physically implemented using NMR.

More information on Shor’s Algorithm

- http://www-bcf.usc.edu/~tbrun/Course/lecture15.pdf
- https://courses.cs.washington.edu/courses/cse599d/06wi/lecturenotes11.pdf
- https://www.cl.cam.ac.uk/teaching/2006/QuantComp/lecture7.pdf
- See these three notes to understand the algorithms better.
- See https://www.uwyo.edu/moorhouse/slides/talk2.pdf on how to factorise large numbers.

Period Finding

- For a function f(x+r)=f(r), can we find a period for this function using quantum algorithms?
- Yes, and we can do it using Phase Estimation.
- Remember

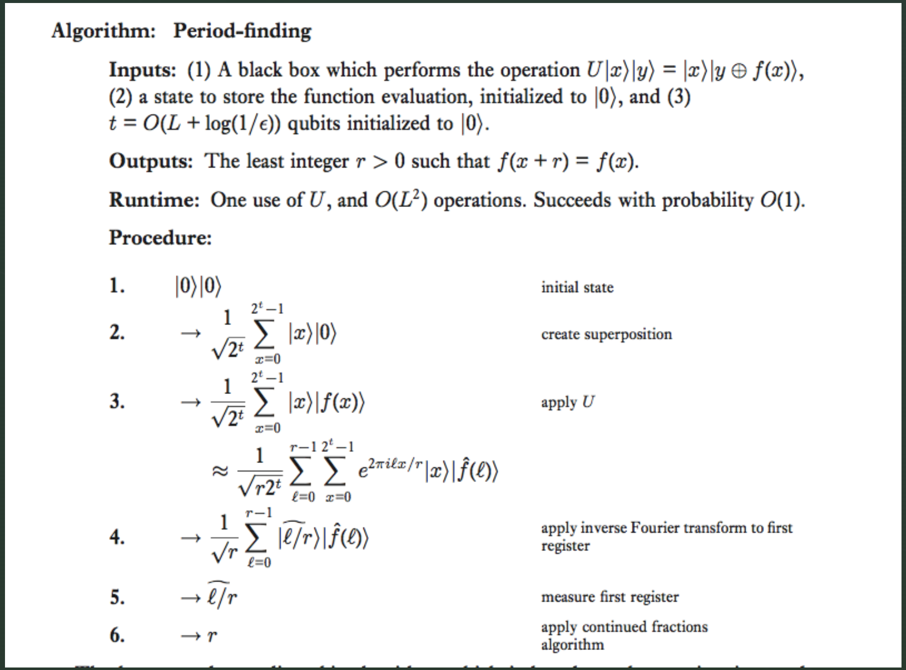


$$\begin{aligned} &\rightarrow \frac{1}{\sqrt{2^t}} \sum_{j=0}^{2^t-1} |j\rangle|x^j \bmod N\rangle \\ &\approx \frac{1}{\sqrt{r2^t}} \sum_{s=0}^{r-1} \sum_{j=0}^{2^t-1} e^{2\pi i s j/r} |j\rangle|u_s\rangle \end{aligned}$$

at the order-finding algo.

- The Phase can be estimated using the continued fractions method, just as before.

Period Finding



Algorithm: Period-finding

Inputs: (1) A black box which performs the operation $U|x\rangle|y\rangle = |x\rangle|y \oplus f(x)\rangle$, (2) a state to store the function evaluation, initialized to $|0\rangle$, and (3) $t = O(L + \log(1/\epsilon))$ qubits initialized to $|0\rangle$.

Outputs: The least integer $r > 0$ such that $f(x + r) = f(x)$.

Runtime: One use of U, and $O(L^2)$ operations. Succeeds with probability $O(1)$.

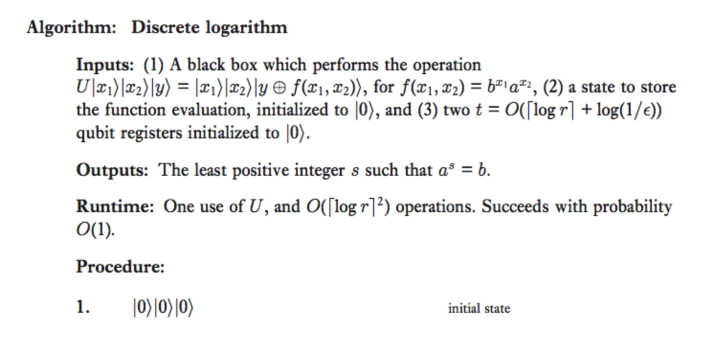
Procedure:

1.	$\|0\rangle\|0\rangle$	initial state
2.	$\rightarrow \frac{1}{\sqrt{2^t}} \sum_{x=0}^{2^t-1} \|x\rangle\|0\rangle$	create superposition
3.	$\rightarrow \frac{1}{\sqrt{2^t}} \sum_{x=0}^{2^t-1} \|x\rangle\|f(x)\rangle$	apply U
	$\approx \frac{1}{\sqrt{r2^t}} \sum_{\ell=0}^{r-1} \sum_{x=0}^{2^t-1} e^{2\pi i \ell x/r} \|x\rangle\|\hat{f}(\ell)\rangle$	
4.	$\rightarrow \frac{1}{\sqrt{r}} \sum_{\ell=0}^{r-1} \|\widetilde{\ell/r}\rangle\|\hat{f}(\ell)\rangle$	apply inverse Fourier transform to first register
5.	$\rightarrow \widetilde{\ell/r}$	measure first register
6.	$\rightarrow r$	apply continued fractions algorithm

Discrete Logarithms

- Little more complex version of the Period Finding Algo.
- Think $f(x_1, x_2) = a^{sx_1+x_2}$.
- This is definitely periodic in 2D (Period (l, -ls))
- Also, this poses the question *If* $b = a^s$ *while a, b are given, what is s?* (Discrete Logarithm Problem)
- Quite expectedly, this is basically the 2D version of Period finding.

Discrete Logarithms



Algorithm: Discrete logarithm

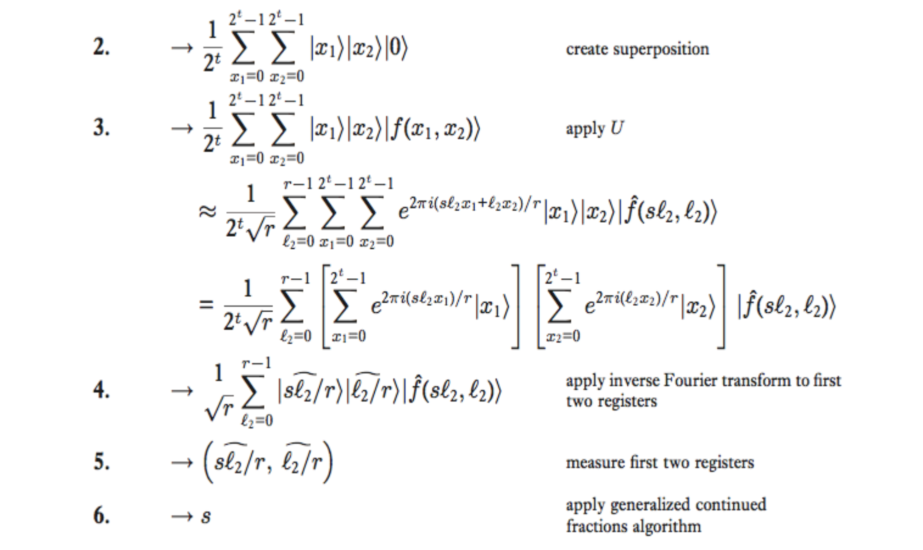
Inputs: (1) A black box which performs the operation $U|x_1\rangle|x_2\rangle|y\rangle = |x_1\rangle|x_2\rangle|y \oplus f(x_1, x_2)\rangle$, for $f(x_1, x_2) = b^{x_1}a^{x_2}$, (2) a state to store the function evaluation, initialized to $|0\rangle$, and (3) two $t = O(\lceil \log r \rceil + \log(1/\epsilon))$ qubit registers initialized to $|0\rangle$.

Outputs: The least positive integer s such that $a^s = b$.

Runtime: One use of U, and $O(\lceil \log r \rceil^2)$ operations. Succeeds with probability $O(1)$.

Procedure:

1. $|0\rangle|0\rangle|0\rangle$ initial state



2. $$\rightarrow \frac{1}{2^t}\sum_{x_1=0}^{2^t-1}\sum_{x_2=0}^{2^t-1}|x_1\rangle|x_2\rangle|0\rangle$$ create superposition

3. $$\rightarrow \frac{1}{2^t}\sum_{x_1=0}^{2^t-1}\sum_{x_2=0}^{2^t-1}|x_1\rangle|x_2\rangle|f(x_1, x_2)\rangle$$ apply U

$$\approx \frac{1}{2^t\sqrt{r}}\sum_{\ell_2=0}^{r-1}\sum_{x_1=0}^{2^t-1}\sum_{x_2=0}^{2^t-1} e^{2\pi i(s\ell_2x_1+\ell_2x_2)/r}|x_1\rangle|x_2\rangle|\hat{f}(s\ell_2, \ell_2)\rangle$$

$$= \frac{1}{2^t\sqrt{r}}\sum_{\ell_2=0}^{r-1}\left[\sum_{x_1=0}^{2^t-1} e^{2\pi i(s\ell_2x_1)/r}|x_1\rangle\right]\left[\sum_{x_2=0}^{2^t-1} e^{2\pi i(\ell_2x_2)/r}|x_2\rangle\right]|\hat{f}(s\ell_2, \ell_2)\rangle$$

4. $$\rightarrow \frac{1}{\sqrt{r}}\sum_{\ell_2=0}^{r-1}|\widetilde{s\ell_2/r}\rangle|\widetilde{\ell_2/r}\rangle|\hat{f}(s\ell_2, \ell_2)\rangle$$ apply inverse Fourier transform to first two registers

5. $$\rightarrow \left(\widetilde{s\ell_2/r}, \widetilde{\ell_2/r}\right)$$ measure first two registers

6. $$\rightarrow s$$ apply generalized continued fractions algorithm

The Hidden Subgroup Problem

- The generalisation for all the work we have done.
- In Group Theory Language: Let f be a function from a finitely generated group G to a finite set X such that f is constant on the cosets of a subgroup K, and distinct on each coset.
- Given a quantum black box for performing the unitary transform $U|g\rangle|h\rangle = |g\rangle|h \oplus f(g)\rangle$, for $g \in G$, $h \in X$, and $\oplus$ an appropriately chosen binary operation on X, find a generating set for K.

The Hidden Subgroup Problem

- For finite or finitely generated Abelian groups, log G gates can effectively solve this problem.
- The Hidden Subgroup problems are usually extremely hard to solve.
- For example, what if there is a group where the continued fraction expansion does not work?