

QUANTUM COMPUTING 6.

Jozef Gruska

Faculty of Informatics

Brno

Czech Republik

November 3, 2010

6. QUANTUM FOURIER TRANSFORM and SHOR'S ALGORITHMS

Perhaps the most significant success of quantum computing so far has been Shor's polynomial time algorithm for factorization to be presented in this section. This is a highly nontrivial algorithm that uses a new technique, that of Quantum Fourier Transform, that will also be illustrated in this chapter.

The fastest classical algorithm to factor m bit numbers requires time $\mathcal{O}(e^{cm^{1/3}(\lg m)^{2/3}})$.

Shor's factorization algorithm requires $\mathcal{O}(m^2 \lg^2 m \lg \lg m)$ time on a quantum computer and polynomial time on a classical computer.

Of interest and importance is also another Shor's polynomial time algorithm, for discrete logarithm, to be also presented in this chapter. We present also another than Shor's approach to quantum factorization.

Correctness and efficiency of Shor's algorithm is based on several simple results from number theory to be presented first.

FIRST REDUCTION

Lemma 0.1 *If there is a polynomial time deterministic (randomized) [quantum] algorithm to find a nontrivial solution of the modular quadratic equations*

$$a^2 \equiv 1 \pmod{n},$$

then there is a polynomial time deterministic (randomized) [quantum] algorithm to factorize integers.

Proof. Let $a \neq \pm 1$ be such that $a^2 \equiv 1 \pmod{n}$. Since

$$a^2 - 1 = (a + 1)(a - 1),$$

if n is not prime, then a prime factor of n has to be a prime factor of either $a + 1$ or $a - 1$.

By using Euclid's algorithm to compute

$$\gcd(a + 1, n) \quad \text{and} \quad \gcd(a - 1, n)$$

we can find, in $\mathcal{O}(\lg n)$ steps, a prime factor of n .

SECOND REDUCTION

The second key concept is that of **period** of the functions

$$f_{n,x}(k) = x^k \pmod n.$$

It is the smallest integer r such that

$$f_{n,x}(k+r) = f_{n,x}(k)$$

for any k , i.e. the smallest r such that

$$x^r \equiv 1 \pmod n.$$

AN ALGORITHM TO SOLVE EQUATION $x^2 \equiv 1 \pmod n$.

1. Choose randomly $1 < a < n$.
2. Compute $\gcd(a, n)$. If $\gcd(a, n) \neq 1$ we have a factor.
3. Find period r of function $a^k \pmod n$.
4. If r is odd or $a^{r/2} \equiv \pm 1 \pmod n$, then go to step 1; otherwise stop.

If this algorithm stops, then $a^{r/2}$ is a non-trivial solution of the equation

$$x^2 \equiv 1 \pmod n.$$

EXAMPLE

Let $n = 15$. Select $a < 15$ such that $\gcd(a, 15) = 1$.

{The set of such a is $\{2, 4, 7, 8, 11, 13, 14\}$ }

Choose $a = 11$. Values of $11^x \bmod 15$ are then

$$11, 1, 11, 1, 11, 1$$

what gives $r = 2$.

Hence $a^{r/2} = 11 \pmod{15}$. Therefore

$$\gcd(15, 12) = 3, \quad \gcd(15, 10) = 5$$

For $a = 14$ we get again $r = 2$, but in this case

$$14^{2/2} \equiv -1 \pmod{15}$$

and the following algorithm fails.

1. Choose randomly $1 < a < n$.
2. Compute $\gcd(a, n)$. If $\gcd(a, n) \neq 1$ we have a factor.
3. Find period r of function $a^k \bmod n$.
4. If r is odd or $a^{r/2} \equiv \pm 1 \pmod{n}$, then go to step 1; otherwise stop.

EFFICIENCY of REDUCTION

Lemma 0.2 *If $1 < a < n$ satisfying $\gcd(n, a) = 1$ is selected in the above algorithm randomly and n is not a power of prime, then*

$$\Pr\{r \text{ is even and } a^{r/2} \not\equiv \pm 1\} \geq \frac{9}{16}.$$

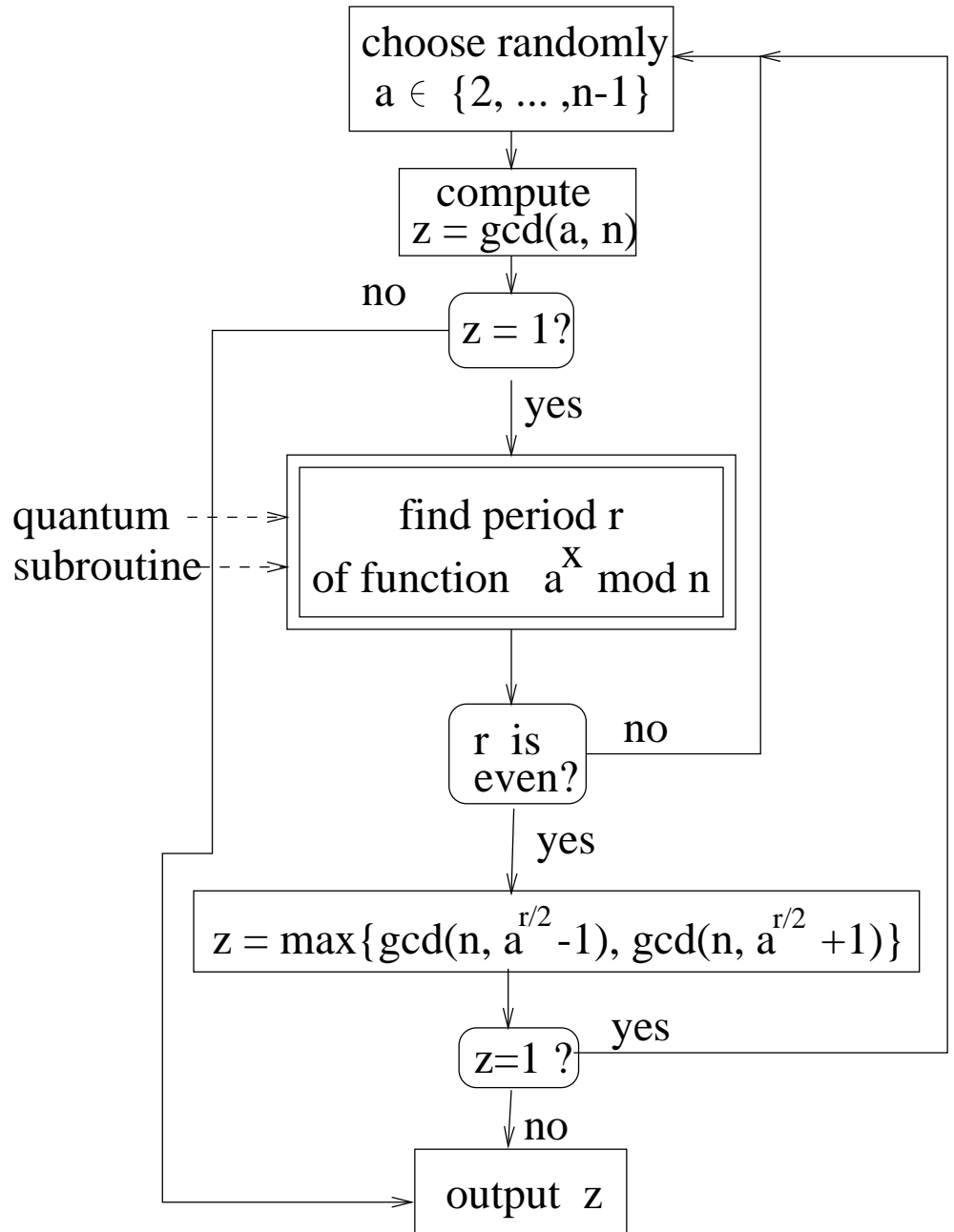
1. Choose randomly $1 < a < n$.
2. Compute $\gcd(a, n)$. If $\gcd(a, n) \neq 1$ we have a factor.
3. Find period r of function $a^k \bmod n$.
4. If r is odd or $a^{r/2} \equiv \pm 1 \pmod{n}$, then go to step 1; otherwise stop.

Corollary 0.3 *If there is a polynomial time randomized [quantum] algorithm to compute the period of the function*

$$f_{n,a}(k) = a^k \bmod n,$$

then there is a polynomial time randomized [quantum] algorithm to find non-trivial solution of the equation $a^2 \equiv 1 \pmod{n}$ (and therefore also to factorize integers).

A GENERAL SCHEME FOR SHOR'S ALGORITHM



SHOR'S ALGORITHM

1. For given $n, q = 2^d, a$ create states

$$\frac{1}{\sqrt{q}} \sum_{x=0}^{q-1} |n, a, q, x, \mathbf{0}\rangle \text{ and } \frac{1}{\sqrt{q}} \sum_{x=0}^{q-1} |n, a, q, x, a^x \bmod n\rangle$$

2. By measuring the last register the state collapses into the state

$$\frac{1}{\sqrt{A+1}} \sum_{j=0}^A |n, a, q, jr + l, y\rangle \text{ or, shortly } \frac{1}{\sqrt{A+1}} \sum_{j=0}^A |jr + l\rangle,$$

where A is the largest integer such that $l + Ar \leq q$, r is the period of $a^x \bmod n$ and l is the offset.

3. In case $A = \frac{q}{r} - 1$, the resulting state has the form.

$$\frac{1}{\sqrt{q}} \sum_{j=0}^{\frac{q}{r}-1} |jr + l\rangle$$

4. By applying quantum Fourier transformation we get then the state

$$\frac{1}{\sqrt{r}} \sum_{j=0}^{r-1} e^{2\pi i l j / r} |j \frac{q}{r}\rangle.$$

5. By measuring the resulting state we get $c = \frac{jq}{r}$ and if $\gcd(j, r) = 1$, what happens with sufficient large probability, then from c and q we can determine the period r .

PERIOD COMPUTATION for $f_{n,a}(x) = a^x \bmod n, q = 2^d$

Hadamard transform applied to the state $|0^{(d)}, 0^{(d)}\rangle$ yields

$$|\phi\rangle = \frac{1}{\sqrt{2^d}} \sum_{x=0}^{q-1} |x, 0^{(d)}\rangle$$

and an application of the unitary transformation

$$U_{f_{n,a}} : |x, 0^{(d)}\rangle \rightarrow |x, a^x \bmod n\rangle$$

provides the state

$$|\phi_1\rangle = U_{f_{n,a}}|\phi\rangle = \frac{1}{\sqrt{2^d}} \sum_{x=0}^{q-1} |x, f_{n,a}(x)\rangle$$

Note 1: All possible values of the function $f_{n,a}$ are encoded in the second register in the state $|\phi_1\rangle$.

Note 2: We are interested in the period of the function $f_{n,a}$ and not in particular values of $f_{n,a}$.

Could we get period by measuring, several times, at first the second register and then the first one?

EXAMPLE

For $n = 15, a = 7, f_{n,a}(x) = 7^x \bmod 15, q = 16$, the state

$$U_{f_{n,a}}|\phi\rangle = \frac{1}{\sqrt{q}} \sum_{x=0}^{q-1} |x, f_{n,a}(x)\rangle$$

has the form

$$\frac{1}{4}(|0\rangle|1\rangle + |1\rangle|7\rangle + |2\rangle|4\rangle + |3\rangle|13\rangle + |4\rangle|1\rangle + |5\rangle|7\rangle + \dots + |14\rangle|4\rangle + |15\rangle|13\rangle).$$

If we measure at this point the second register, then we get as the outcome one of the numbers 1, 4, 7 or 13, and the following table shows the corresponding post-measurement states in the second column.

result	post-measurement state	offset
1	$\frac{1}{2}(0\rangle + 4\rangle + 8\rangle + 12\rangle) 1\rangle$	0
4	$\frac{1}{2}(2\rangle + 6\rangle + 10\rangle + 14\rangle) 4\rangle$	2
7	$\frac{1}{2}(1\rangle + 5\rangle + 9\rangle + 13\rangle) 7\rangle$	1
13	$\frac{1}{2}(3\rangle + 7\rangle + 11\rangle + 15\rangle) 13\rangle$	3

The corresponding sequences of values of the first register are periodic with period 4 but they have different offsets (pre-periods) listed in column 3 of the table.

GRAPHICAL REPRESENTATION of STEPS FOR SHOR'S ALGORITHM

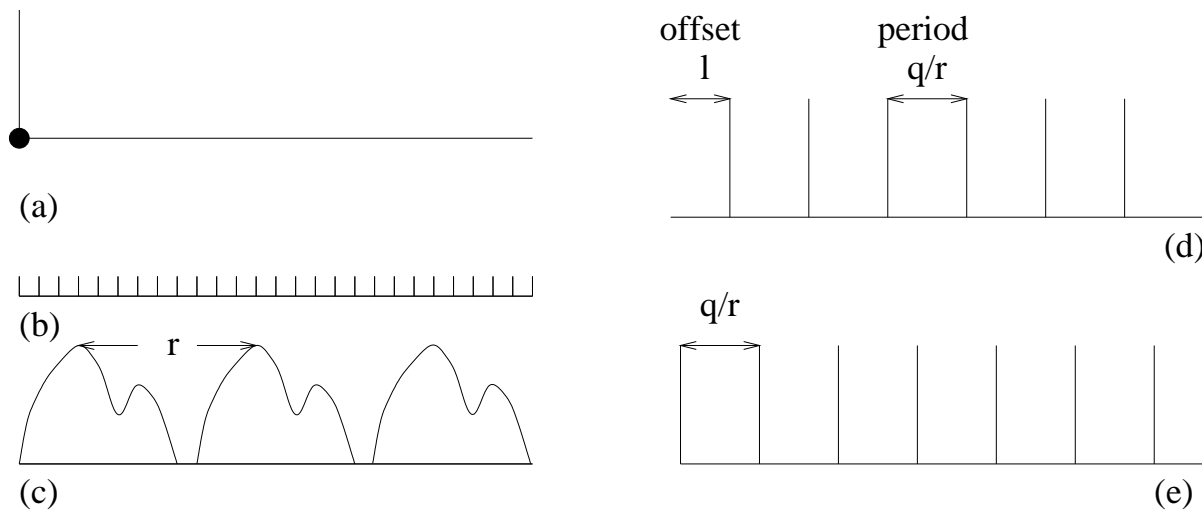


Figure 1: Graphical representation of steps of Shor's algorithm

DISCRETE FOURIER TRANSFORM

Discrete Fourier Transform maps a vector $\mathbf{a} = (a_0, a_1, \dots, a_{n-1})^T$ into the vector $DFT(\mathbf{a}) = A_n \mathbf{a}$, where A_n is an $n \times n$ matrix such that $A_n[i, j] = \frac{1}{\sqrt{n}} \omega^{ij}$ for $0 \leq i, j < n$ and $\omega = e^{2\pi i/n}$ is the n th root of unity.

The matrix A_n has therefore the form

$$A_n = \frac{1}{\sqrt{n}} \begin{pmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & \omega & \omega^2 & \dots & \omega^{(n-1)} \\ 1 & \omega^2 & \omega^4 & \dots & \omega^{2(n-1)} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & \omega^{(n-1)} & \omega^{2(n-1)} & \dots & \omega^{(n-1)^2} \end{pmatrix}.$$

The Inverse Discrete Fourier Transform is the mapping

$$DFT^{-1}(\mathbf{a}) = A_n^{-1} \mathbf{a},$$

where

$$A_n^{-1}[i, j] = \frac{1}{\sqrt{n}} \omega^{-ij}.$$

INSIDES into DFT

There is a close relation between Discrete Fourier Transform and polynomial evaluation and interpolation. Let us consider a polynomial

$$p(x) = \sum_{i=0}^{n-1} a_i x^i.$$

Such a polynomial can be uniquely represented in two ways: either by a list of its coefficients a_0, a_1, \dots, a_{n-1} , or by a list of its values at n distinct points x_0, x_1, \dots, x_{n-1} .

The process of finding the coefficient representation of the polynomial given its values at points x_0, x_1, \dots, x_{n-1} is called **interpolation**.

Computing the Discrete Fourier Transform of a vector $(a_0, a_1, \dots, a_{n-1})$ is equivalent to converting the coefficient representation of the polynomial $\sum_{i=0}^{n-1} a_i x^i$ to its value representation at the points $\omega^0, \omega^1, \dots, \omega^{n-1}$.

Likewise, the Inverse Discrete Fourier Transform is equivalent to interpolating a polynomial given its values at the n -th roots of unity.

QUANTUM FOURIER TRANSFORM

The Quantum Fourier Transform is a quantum variant of the **Discrete Fourier Transform** (DFT). DFT maps a discrete function to another discrete one with equally distant points as its domain. For example it maps a q -dimensional complex vector

$$\{f(0), f(1), \dots, f(q-1)\} \text{ into } \{\bar{f}(0), \bar{f}(1), \dots, \bar{f}(q-1)\},$$

where for $c \in \{0, \dots, q-1\}$

$$\bar{f}(c) = \frac{1}{\sqrt{q}} \sum_{a=0}^{q-1} e^{2\pi i ac/q} f(a), \quad (1)$$

for $c \in \{0, \dots, q-1\}$.

The quantum version of DFT (QFT) is the unitary transformation

$$\text{QFT}_q : |a\rangle \rightarrow \frac{1}{\sqrt{q}} \sum_{c=0}^{q-1} e^{2\pi i ac/q} |c\rangle \quad (2)$$

The quantum version of DFT (QFT) is the unitary transformation

$$\mathbf{QFT}_q : |a\rangle \rightarrow \frac{1}{\sqrt{q}} \sum_{c=0}^{q-1} e^{2\pi i ac/q} |c\rangle \quad (3)$$

for $0 \leq a < q$, with the unitary matrix

$$F_q = \frac{1}{\sqrt{q}} \begin{pmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & \omega & \omega^2 & \dots & \omega^{(q-1)} \\ 1 & \omega^2 & \omega^4 & \dots & \omega^{2(q-1)} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & \omega^{(q-1)} & \omega^{2(q-1)} & \dots & \omega^{(q-1)^2} \end{pmatrix},$$

where $\omega = e^{2\pi i/q}$ is the q th root of unity.

If applied to a quantum superposition, \mathbf{QFT}_q performs as follows;

$$\mathbf{QFT}_q : \sum_{a=0}^{q-1} f(a) |a\rangle \rightarrow \sum_{c=0}^{q-1} \bar{f}(c) |c\rangle,$$

where $\bar{f}(c)$ is defined by (??).

Observe that

$$\mathbf{QFT}_q : |\mathbf{0}\rangle \rightarrow \frac{1}{\sqrt{q}} \sum_{i=0}^{q-1} |i\rangle,$$

SHOR'S ALGORITHM — PHASE 1

Design of states whose amplitudes have the same period as $f_{n,a}$

Given an m bit integer n we choose a $n^2 \leq q \leq 2n^2$ and start with five registers in states $|n, a, q, \mathbf{0}, \mathbf{0}\rangle$, where the last two registers have $m = \lceil \lg n \rceil$ qubits.

An application of the Hadamard transformation to the fourth register yields the state

$$\frac{1}{\sqrt{q}} \sum_{x=0}^{q-1} |n, a, q, x, \mathbf{0}\rangle.$$

and using quantum parallelism we compute $a^x \bmod n$ for all x in one step, to get

$$\frac{1}{\sqrt{q}} \sum_{x=0}^{q-1} |n, a, q, x, a^x \bmod n\rangle.$$

As the next step we perform a measurement on the last register. Let y be the value obtained, i.e. $y = a^{l_y} \bmod n$ for the smallest l_y with this property. If r is the period of $f_{n,a}$, then $a^{l_y} \equiv a^{jr+l_y} \pmod{n}$ for all j . Therefore, the measurement actually selects the sequence of x 's values (in the fourth register), $l_y, l_y + r, l_y + 2r, \dots, l_y + Ar$, where A is the largest integer such that $l_y + Ar \leq q - 1$. Clearly, $A \approx \frac{q}{r}$. The post-measurement state is then

$$|\phi_l\rangle = \frac{1}{\sqrt{A+1}} \sum_{j=0}^A |n, a, q, jr + l_y, y\rangle = \frac{1}{\sqrt{A+1}} \sum_{j=0}^A |jr + l_y\rangle. \quad (4)$$

after omitting some registers being fixed from now on.

SHOR'S ALGORITHM — PHASE 2.

Amplitude amplification by QFT

From now on we consider in detail only a special case. Namely that r divides q . In such a case $A = \frac{q}{r} - 1$. In such a case the last state can be written in the form

$$|\phi_l\rangle = \frac{1}{\sqrt{q}} \sum_{j=0}^{\frac{q}{r}-1} |jr + l_y\rangle$$

and after QFT_q is applied on $|\phi_l\rangle$ we get:

$$\text{QFT}_q |\phi_l\rangle = \frac{1}{\sqrt{q}} \sum_{c=0}^{q-1} \frac{1}{\sqrt{q}} \sum_{j=0}^{\frac{q}{r}-1} e^{2\pi i c(jr+l_y)/q} |c\rangle = \frac{\sqrt{r}}{q} \sum_{c=0}^{q-1} e^{2\pi i l_y c/q} \left(\sum_{j=0}^{\frac{q}{r}-1} e^{2\pi i jcr/q} \right) |c\rangle = \sum_{c=0}^{q-1} \alpha_c |c\rangle.$$

If c is a multiple of $\frac{q}{r}$, then $e^{2\pi i jcr/q} = 1$ and if c is not a multiple of $\frac{q}{r}$, then

$$\sum_{j=0}^{\frac{q}{r}-1} e^{2\pi i jcr/q} = 0,$$

because the above sum is over a set of $\frac{q}{r}$ roots of unity equally spaced around the unit circle. Thus

$$\alpha_c = \begin{cases} \frac{1}{\sqrt{r}} e^{2\pi i l_y c/q}, & \text{if } c \text{ is a multiple of } \frac{q}{r}; \\ 0, & \text{otherwise;} \end{cases}$$

and therefore

$$|\phi_{out}\rangle = \text{QFT}_q |\phi_l\rangle = \frac{1}{\sqrt{r}} \sum_{j=0}^{r-1} e^{2\pi i l_y j/r} |j \frac{q}{r}\rangle.$$

The key point is that the trouble-making offset l_y appears now in the phase factor $e^{2\pi i l_y j/r}$ and has no influence either on the probabilities or on the values in the register.

SHOR'S ALGORITHM — PHASE 3

Period extraction

Each measurement of the state $|\phi_{out}\rangle$ therefore yields one of the multiples $c = \lambda \frac{q}{r}$, $\lambda \in \{0, 1, \dots, r - 1\}$, where each λ is chosen with the same probability $\frac{1}{r}$.

Observe also that in this case the QFT transforms a function with the period r (and an offset l) to a function with the period $\frac{q}{r}$ and offset 0. After each measurement we therefore know c and q and

$$\frac{c}{q} = \frac{\lambda}{r},$$

where λ is randomly chosen.

If $\gcd(\lambda, r) = 1$, then from q we can determine r by dividing q with $\gcd(c, q)$. Since λ is chosen randomly, the probability that $\gcd(\lambda, r) = 1$ is greater than $\Omega(\frac{1}{\lg \lg r})$. If the above computation is repeated $\mathcal{O}(\lg \lg r)$ times, then the success probability can be as close to 1 as desired and therefore r can be determined efficiently.¹

In the general case, i.e., if $A \neq \frac{q}{r} - 1$, there is only a more sophisticated computation of the resulting probabilities and a more sophisticated way to determine r (using a [continuous fraction method](#) to extract the period from its approximation).

¹As observed by Shor (1994) and shown by Cleve et al. (1998), the expected number of trials can be put down to a constant.

GENERAL CASE

Let us now sketch Shor's algorithm to compute the period of a function $f(x) = a^x \bmod n$ for the general case.

QFT_q is applied to the first register of the state $\frac{1}{\sqrt{q}} \sum_{x=0}^{q-1} |x\rangle |f(x)\rangle$ and afterwards the measurement is performed on the first register to provide an $y_0 \in [0, \dots, q-1]$.

To get the period r the following algorithm is realized where $\xi = \frac{y_0}{q}$, $a_0 = \lfloor \xi \rfloor$, $\xi_0 = \xi - a_0$,

$p_0 = a_0, q_0 = 1, p_1 = a_1 a_0 + 1, q_1 = a_1$

for $j = 1$ **until** $\xi_j = 0$ **do**

- compute p_j and q_j using the recursion (for the case $\xi_j \neq 0$);

$$a_j = \left\lfloor \frac{1}{\xi_{j-1}} \right\rfloor, \xi_j = \frac{1}{\xi_{j-1}} - a_j,$$

$$p_j = a_j p_{j-1} + p_{j-2}, \quad q_j = a_j q_{j-1} + q_{j-2}$$

- Test whether $q_j = r$ by computing first $m^{q_j} = \prod_i (m^{2^i})^{q_{j,i}} \bmod n$, where $q_j = \sum_i q_{j,i} 2^i$ is the binary expansion of q_j .

If $a^{q_j} = 1 \bmod n$, then exit with $r = q_j$; if not continue the loop.

The non-easy task is to show, what has been done, that the above algorithm provides the period r with sufficient probability ($> \frac{0.232}{\lg \lg n} (1 - \frac{1}{n})^2$).

COMMENTS on SHOR'S FACTORIZATION ALGORITHM

- Efficient implementations of QFT_q , concerning the number of gates, are known for the the case $q = 2^d$ or q is smooth (that is if factors of q are smaller than $\mathcal{O}(\lg q)$).
- Efficient implementation of modular operations (including exponentiation) are known.
- An estimation says that $300 \lg n$ gates are needed to factor n .
- An estimation says that to factor 130 digit integers would require two weeks on an ideal quantum computer with switching frequency 1 MHz. However to factor 260-digit number only 16 times larger time would be needed.
- It has been shown that there is polynomial time factorization even in the case only one pure qubit is available and the rest of quantumness available is in mixed states.

SHOR'S DISCRETE LOGARITHM ALGORITHM

Shor's quantum algorithm for discrete logarithm will be again presented only for a special case.

The task is to determine an r such that $g^r \equiv x \pmod{p}$ given a prime p , a generator g of the group \mathbf{Z}_p^* and a $0 < x < p$. The special case we consider is that $p - 1$ is smooth.

Using QFT_{p-1} twice, on the third and fourth sub-register of the register $|x, g, \mathbf{0}, \mathbf{0}, \mathbf{0}\rangle$, we get

$$|\phi\rangle = \frac{1}{p-1} \sum_{a=0}^{p-2} \sum_{b=0}^{p-2} |x, g, a, b, \mathbf{0}\rangle,$$

a uniform distribution of all pairs (a, b) , $0 \leq a, b \leq p - 2$. By applying to $|\phi\rangle$ a unitary mapping

$$(x, g, a, b, \mathbf{0}) \rightarrow (x, g, a, b, g^a x^{-b} \pmod{p})$$

we get

$$|\phi'\rangle = \frac{1}{p-1} \sum_{a=0}^{p-2} \sum_{b=0}^{p-2} |x, g, a, b, g^a x^{-b} \pmod{p}\rangle.$$

Since x, g will not be changed in the following computations we will not write them explicitly any longer.

As the next step we apply QFT_{p-1} on $|\phi'\rangle$ twice, once to map $a \rightarrow c$ with amplitude $\frac{1}{\sqrt{p-1}} e^{2\pi i ac/(p-1)}$ and once to map $b \rightarrow d$ with amplitude $\frac{1}{\sqrt{p-1}} e^{2\pi i bd/(p-1)}$. The resulting state is

$$|\phi_1\rangle = \frac{1}{(p-1)^2} \sum_{a,b,c,d=0}^{p-2} e^{\frac{2\pi i}{p-1}(ac+bd)} |c, d, g^a x^{-b} \pmod{p}\rangle.$$

$$|\phi_1\rangle = \frac{1}{(p-1)^2} \sum_{a,b,c,d=0}^{p-2} e^{\frac{2\pi i}{p-1}(ac+bd)} |c, d, g^a x^{-b} \bmod p\rangle.$$

Let us now measure last register and denote by y the value we get.

The state $|\phi_1\rangle$ then collapses into the state (before normalization)

$$|\phi_2\rangle = \sum_{c,d=0}^{p-1} \alpha(c, d) |c, d, y\rangle,$$

where

$$\alpha(c, d) = \frac{1}{(p-1)^2} \sum_{\{(a,b) \mid y=g^a x^{-b} \bmod p\}} e^{\frac{2\pi i}{p-1}(ac+bd)}.$$

We now claim that if $y = g^a x^{-b} \bmod p$, then $y = g^k$ for some k such that

$$a - rb \equiv k \pmod{p-1}.$$

Indeed,

$$y = g^a x^{-b} \equiv g^a (g^r)^{-b} = g^{a-rb}.$$

If $a - rb \equiv k \pmod{p-1}$, then

$$g^{a-rb} \equiv g^k \pmod{p}$$

Therefore

$$\alpha(c, d) = \frac{1}{(p-1)^2} \sum_{\{(a,b) \mid a-rb \equiv k \pmod{p-1}\}} e^{\frac{2\pi i}{p-1}(ac+bd)}$$

For the probability Pr that, for fixed c and d we get by measurement of $|\phi_2\rangle$ a value y is therefore

$$\left| \frac{1}{(p-1)^2} \sum_{a,b=0}^{p-2} \{ e^{\frac{2\pi i}{p-1}(ac+bd)} \mid a-rb \equiv k \pmod{p-1} \} \right|^2.$$

By substituting $a = k + rb + j_b(p-1)$ we get the probability

$$Pr = \left| \frac{1}{(p-1)^2} \sum_{b=0}^{p-2} e^{\frac{2\pi i}{p-1}(kc+cj_b(p-1)+b(d+rc))} \right|^2.$$

$$\left| \frac{1}{(p-1)^2} e^{\frac{2\pi i ac}{p-1}} \sum_{b=0}^{p-2} e^{\frac{2\pi i}{p-1}(b(d+rc))} \right|^2$$

what equals

$$\left| \frac{1}{(p-1)^2} \sum_{b=0}^{p-2} e^{\frac{2\pi i}{p-1}(b(d+rc))} \right|^2$$

The above probability Pr is therefore 0 if

$$d + rc \not\equiv 0 \pmod{p-1}$$

because, as in the previous algorithm, in such a case the sum in the above expression is over a set of $(p-1)$ st roots of unity equally spaced around the unit circle.

On the other hand, if

$$d \equiv -rc \pmod{p-1},$$

then the above sum does not depend on b and it is equal to

$$(p-1)^{-1} e^{(2\pi i k c / (p-1))}.$$

The square of its absolute value, the probability, is therefore $\frac{1}{(p-1)^2}$.

Consequence: the measurements on the first and second register provide a (random) $c < p-1$ and a d such that

$$d \equiv -rc \pmod{p-1}.$$

If $\gcd(c, p-1) = 1$, r can now be obtained as a unique solution of the above congruence equation.

Therefore, the number of computations needed to perform in order to get the probability close to 1 for finding r is polynomial in $\lg \lg p$.

COMMENTS on SHOR's ALGORITHMS

- To factor an integer n Shor's algorithm uses $\mathcal{O}(\lg^3 n)$ steps and success probability is guaranteed to be at least $\Omega(\frac{1}{\lg \lg n})$.
- An analysis of Shor's algorithm therefore shows that by running the algorithm $\mathcal{O}(\lg \lg n)$ times, therefore in total in $\mathcal{O}(\lg^3 n \lg \lg n)$ times we have very high success probability.
- Shor's algorithms make some of the important current cryptosystems, as RSA, ElGamal and so on vulnerable to attacks using quantum computers.
- Shor's result have been generalized to show that a large range of cryptosystems, including elliptic curve cryptosystems, would be vulnerable to attacks using quantum computers.

EXTRAS

HIDDEN SUBGROUP PROBLEM

Given: An (efficiently computable) function $f : G \rightarrow R$, where G group and R a finite set.

Promise: There exists a subgroup $G_0 \leq G$ such that f is constant and distinct on the cosets of G_0 .

Task: Find a generating set for G_0 (in polynomial time (in $\lg |G|$) number of calls to the oracle for f and in the overall polynomial time).²

Deutsch's problem, $G = \mathbf{Z}_2$, $f : \{0, 1\} \rightarrow \{0, 1\}$, $x - y \in G_0 \Leftrightarrow f(x) = f(y)$. Decide whether $G_0 = \{0\}$ (and f is balanced) or $G_0 = \{0, 1\}$ (and f is constant).

Simon's problem, $G = \mathbf{Z}_2^n$, $f : G \rightarrow R$. $x - y \in G_0 \Leftrightarrow f(x) = f(y)$, $G_0 = \{0^{(n)}, s\}$, $s \in \mathbf{Z}_2^n$.
Decide whether $G_0 = \{0^{(n)}\}$ or $G_0 = \{0^{(n)}, s\}$, with an $s \neq 0^{(n)}$ (and in the second case find s).

Order-finding problem, $G = \mathbf{Z}$, $a \in \mathbf{N}$, $f(x) = a^x$, $x - y \in G_0 \Leftrightarrow f(x) = f(y)$, $G_0 = \{rk \mid k \in \mathbf{Z}$ for the smallest r such that $a^r = 1$.} Find r .

Discrete logarithm problem, $G = \mathbf{Z}_r \times \mathbf{Z}_r$, $a^r = 1$, $b = a^m$, $a, b \in \mathbf{N}$, $f(x, y) = a^x b^y$,
 $f(x_1, y_1) = f(x_2, y_2) \Leftrightarrow (x_1, y_1) - (x_2, y_2) \in G_0$. $G_0 = \{(-km, m) \mid k \in \mathbf{Z}_r\}$. Find G_0 (or m).

²A way to solve the problem is to show that in polynomial number of oracle calls (or time) the states corresponding to different candidate subgroups have exponentially small inner product and are therefore distinguishable.

Graph automorphism problem: Consider $G = S_n$, the symmetric group on n elements, which corresponds to the permutations of $\{1, 2, \dots, n\}$. Let \mathbf{G} be a graph on n vertices labelled $\{1, 2, \dots, n\}$. For any permutation $\sigma \in S_n$, let $f_{\mathbf{G}}$ maps S_n to the set of n -vertex graphs by mapping $f_{\mathbf{G}}(\sigma) = \sigma(\mathbf{G})$, where $\sigma(\mathbf{G})$ is the graph obtained by permuting the vertex labels of \mathbf{G} according to σ . For the function $f_{\mathbf{G}}$, the hidden subgroup of \mathbf{G} is the automorphism group of G . Note that for the graph automorphism problem the group G is non-Abelian.

IMPLEMENTATION OF THE FOURIER TRANSFORM in \mathbb{Z}_{2^m}

The clue to the design of a quantum circuit to implement the QFT

$$|x\rangle \rightarrow \frac{1}{\sqrt{2^m}} \sum_{y=0}^{2^m-1} e^{\frac{2\pi ixy}{2^m}} |y\rangle$$

for $|x\rangle = |x_{m-1}\rangle |x_{m-2}\rangle \dots |x_0\rangle$ is the decomposition

$$\sum_{y=0}^{2^m-1} e^{\frac{2\pi ixy}{2^m}} |y\rangle = (|0\rangle + e^{\frac{\pi ix}{2^0}} |1\rangle) (|0\rangle + e^{\frac{\pi ix}{2^1}} |1\rangle) \dots (|0\rangle + e^{\frac{\pi ix}{2^{m-1}}} |1\rangle)$$

The exponent in the l -th factor of the above decomposition can be written as follows

$$\begin{aligned} & \exp\left(\frac{\pi i(2^{m-1}x_{m-1} + 2^{m-2}x_{m-2} + \dots + 2x_1 + x_0)}{2^{l-1}}\right) \\ &= \exp\left(\frac{\pi i(2^{l-1}x_{l-1} + 2^{l-2}x_{l-2} + \dots + 2x_1 + x_0)}{2^{l-1}}\right) \\ &= (-1)^{x_{l-1}} \exp\left(\frac{\pi i x_{l-2}}{2}\right) \dots \exp\left(\frac{\pi i x_1}{2^{l-2}}\right) \exp\left(\frac{\pi i x_0}{2^{l-1}}\right) \end{aligned}$$

DESIGN of CIRCUIT

Starting, for convenience, with the reverse representation of x as $x_0x_1 \dots x_{m-1}$ we show how to implement

$$(|0\rangle + e^{\frac{\pi ix}{2^0}}|1\rangle)(|0\rangle + e^{\frac{\pi ix}{2^1}}|1\rangle) \dots (|0\rangle + e^{\frac{\pi ix}{2^{m-1}}}|1\rangle)$$

for qubits $m-1, m-2, \dots, 0$, step by step, starting with $(m-1)$ -th qubit.

Using Hadamard transform on the $m-1$ -th qubit we get

$$\frac{1}{\sqrt{2}}|x_0\rangle|x_1\rangle \dots |x_{m-2}\rangle(|0\rangle + (-1)^{x_{m-1}}|1\rangle)$$

and then we can complete the phase $(-1)^{x_{m-1}}$ to

$$(-1)^{x_{m-1}} \exp\left(\frac{\pi ix_{m-2}}{2^1}\right) \dots \exp\left(\frac{\pi ix_1}{2^{m-2}}\right) \exp\left(\frac{\pi ix_0}{2^{m-1}}\right)$$

by using **conditionally** phase rotations

$$\exp\left(\frac{\pi i}{2^1}\right), \dots, \exp\left(\frac{\pi i}{2^{m-2}}\right) \exp\left(\frac{\pi i}{2^{m-1}}\right)$$

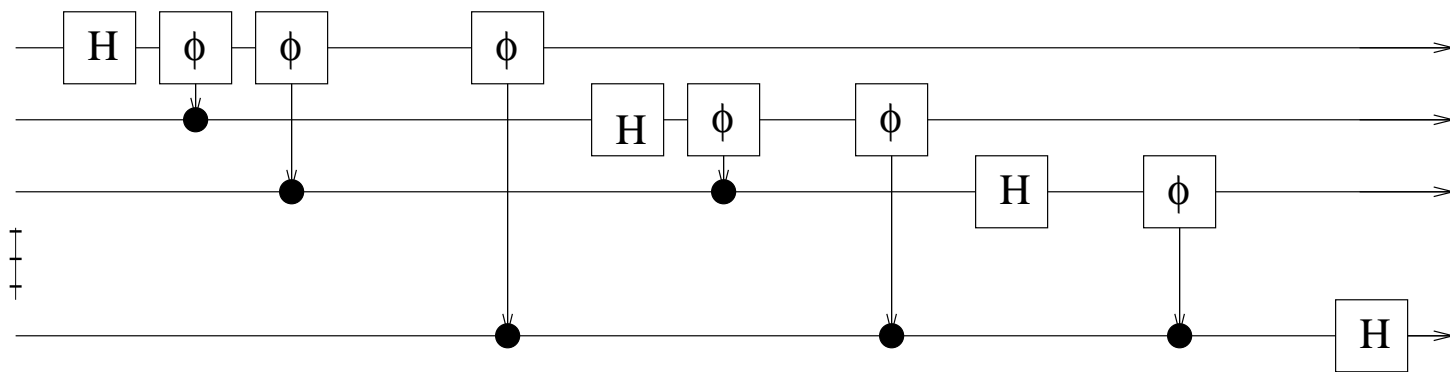
This means that for each $l \in \{1, 2, \dots, m-1\}$ a phase factor $\exp(\frac{\pi i}{2^{m-l}})$ is introduced to the m -th bit if and only if m th and l th qubits are both 1. This will provide the state

$$\frac{1}{\sqrt{2}}|x_0\rangle|x_1\rangle \dots |x_{m-2}\rangle(|0\rangle + e^{\frac{2\pi i x}{2^{m-1}}} |1\rangle)$$

This process can be repeated with other qubits. Each time we use once the Hadamard transform and then the unitary

$$\phi_{kl} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{\frac{\pi i}{2^{l-k}}} \end{pmatrix}$$

which acts on the l th and k th qubit. The resulting circuit has then the following form:



COMPLEXITY of FOURIER TRANSFORM

- The naive algorithm to compute all elements of classical Fourier transform, element by element using basic definition, requires $\mathcal{O}(2^{2m})$ steps.
- A special recursive method, called **Fast Fourier Transform**, that recursively reduces computation of DFT in \mathbb{Z}_{2^m} to computation of two DFT in $\mathbb{Z}_{2^{m-1}}$, requires $\mathcal{O}(m2^m)$ steps - a significant improvement.
- Quantum Fourier Transform in \mathbb{Z}_{2^m} can be done in $\mathcal{O}(m^2)$ quantum steps.

Moreover, in the classical case, physical representation of

$$(f(0), f(1), \dots, f(2^m - 1))$$

requires $\Omega(2^m)$ bits,

but in the quantum case representation of

$$c_0|0\rangle + c_1|1\rangle + \dots + c_{2^m-1}|2^m - 1\rangle$$

requires only m qubits.

FOURIER TRANSFORM on FINITE ABELIAN GROUPS

We show now basics how the concept of Fourier Transform is defined on any finite Abelian group.

CHARACTERS of ABELIAN GROUPS

Let G be an Abelian group written additively, and $|G| = n$. A **character** χ of G is any **morphism** $\chi : G \rightarrow \mathbf{C}/0$. That is for any $g_1, g_2 \in G$ it holds:

$$\chi(g_1 + g_2) = \chi(g_1)\chi(g_2).$$

This implies that $\chi(0) = 1$ and $1 = \chi(n g) = \chi(g)^n$ for any $g \in G$. Therefore, **all values of χ are n th roots of unity**.

If we define multiplication of characters χ_1 and χ_2 by $\chi_1\chi_2(g) = \chi_1(g)\chi_2(g)$, then characters form so-called **dual group** \hat{G} . Groups G and \hat{G} are isomorphic for all Abelian groups G .

Example 1 Any cyclic group of n elements is isomorphic to the group \mathbf{Z}_n and all its characters have the form, for some $y \in \mathbf{Z}_n$:

$$\chi_y(x) = e^{\frac{2\pi i xy}{n}}.$$

Example 2 In the additive group \mathbf{F}_2^m , of all binary strings of length m , all characters have the form, for some binary m -bit strings x and y :

$$\chi_y(x) = (-1)^{x \cdot y},$$

where $x \cdot y = \sum_{i=1}^m x_i y_i \pmod{2}$

ORTHOGONALITY of CHARACTERS

Any function $f : G \rightarrow \mathbf{C}$ on an Abelian group $G = \{g_1, \dots, g_n\}$ can be specified by the vector $(f(g_1), \dots, f(g_n))$, and if the scalar product of two functions is defined in the standard way as

$$\langle f|g \rangle = \sum_{i=1}^n f^*(g_i)h(g_i),$$

then for any characters χ_1 and χ_2 on G it holds

$$\langle \chi_i|\chi_j \rangle = \begin{cases} 0, & \text{if } i \neq j \\ n, & \text{if } i = j \end{cases}$$

Therefore, the functions $\{B_i = \frac{1}{\sqrt{n}}\chi_i\}$ form an orthonormal basis on the set of all functions $f : G \rightarrow \mathbf{C}$.

FOURIER TRANSFORM

Since any $f : G \rightarrow \mathbf{C}$ has a unique representation with respect to the basis $\{B_i = \frac{1}{\sqrt{n}}\chi_i\}_{i=1}^n$,

$$f = \hat{f}_1 B_1 + \dots + \hat{f}_n B_n$$

the function $\hat{f} : G \rightarrow \mathbf{C}$ defined by

$$\hat{f}(g_i) = \hat{f}_i$$

is called the Fourier transform of f .

Since $\hat{f}_i = \langle B_i | f \rangle$, we get

$$\hat{f}(g_i) = \frac{1}{\sqrt{n}} \sum_{k=1}^n \chi_i^*(g_k) f(g_k),$$

and therefore in \mathbf{Z}_n the Fourier transform has the form

$$\hat{f}(x) = \frac{1}{\sqrt{n}} \sum_{y \in \mathbf{Z}_n} e^{-\frac{2\pi i xy}{n}} f(y)$$

and in \mathbf{F}_2^m the Fourier transform has the form

$$\hat{f}(x) = \frac{1}{\sqrt{2^m}} \sum_{y \in \mathbf{F}_2^m} (-1)^{x \cdot y} f(y).$$

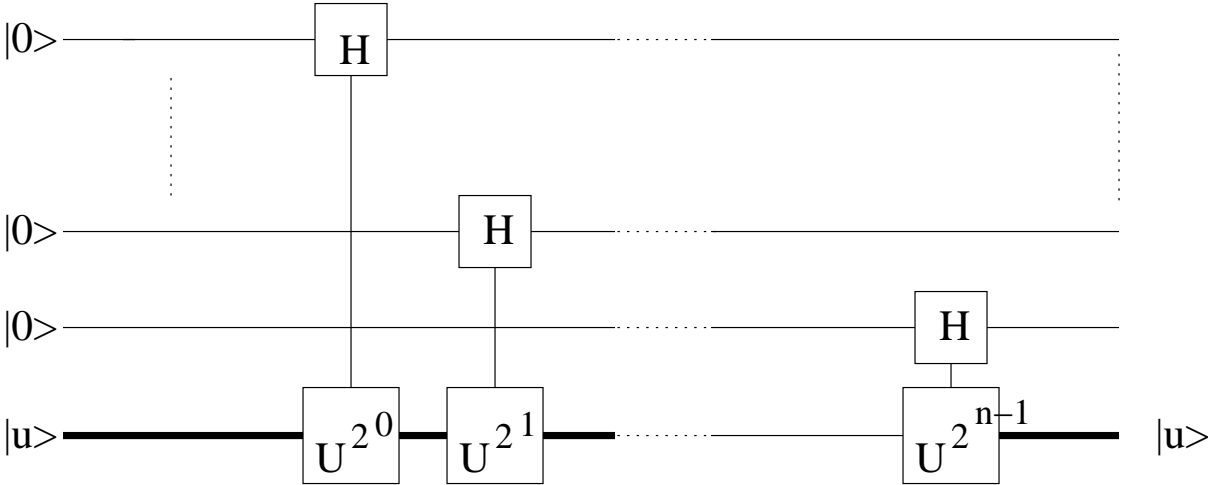
PHASE ESTIMATION

Closely related to the implementation of Quantum Fourier Transform is a method for solution of so called phase estimation problem. Given is an n -qubit unitary operator U , with an eigenvector $|u\rangle$ and its eigenvalue $e^{2\pi i\phi}$, where ϕ is unknown. The task is to determine ϕ .

For a related control- U^j -gate it holds

$$\text{control-}U^j\left(\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)\right)|u\rangle = \frac{1}{\sqrt{2}}(|0\rangle|u\rangle + e^{2\pi i j\phi}|1\rangle|u\rangle) = \frac{1}{\sqrt{2}}(|0\rangle + e^{2\pi i j\phi}|1\rangle)|u\rangle.$$

This means that the first n -qubits of the circuit



produce the state

$$\frac{1}{\sqrt{2^n}} \bigotimes_{t=1}^n (|0\rangle + e^{2\pi i 2^{t-1} \phi} |1\rangle) = \frac{1}{\sqrt{2^n}} \sum_{k=0}^{2^n-1} e^{2\pi i k \phi} |k\rangle$$

The last equality follows from the lemma on next slide.

LEMMA

Let $x \in \{1, \dots, 2^n - 1\}$ and let its binary representation be $x_1 x_2 \dots x_n$. For quantum Fourier transform

$$F|x\rangle = \frac{1}{\sqrt{2^n}} \sum_{k=0}^{2^n-1} e^{2\pi i j k / 2^n} |k\rangle$$

it holds **Lemma**

$$F|x\rangle = \frac{1}{\sqrt{2^n}} [(|0\rangle + e^{2\pi i 0 \cdot x_n} |1\rangle) (|0\rangle + e^{2\pi i 0 \cdot x_{n-1} x_n} |1\rangle) \dots (|0\rangle + e^{2\pi i 0 \cdot x_1 \dots x_n} |1\rangle)].$$

Proof This follows from calculations

$$F|x\rangle = \frac{1}{\sqrt{2^n}} \sum_{k=0}^{2^n-1} e^{2\pi i x k / 2^n} |k\rangle = \frac{1}{\sqrt{2^n}} \sum_{k_1=0}^1 \dots \sum_{k_n=0}^1 \exp(2\pi i x \sum_{l=1}^n k_l 2^{-l}) |x_1 \dots x_n\rangle \quad (5)$$

$$= \frac{1}{\sqrt{2^n}} \sum_{k_1=0}^1 \dots \sum_{k_n=0}^1 \bigotimes_{l=1}^n e^{2\pi i x k_l / 2^l} |k_l\rangle = \frac{1}{\sqrt{2^n}} \bigotimes_{l=1}^n \sum_{k_l=0}^1 e^{2\pi i x k_l / 2^l} |k_l\rangle \quad (6)$$

$$= \frac{1}{\sqrt{2^n}} \bigotimes_{l=1}^n (|0\rangle + e^{2\pi i x / 2^l} |1\rangle) \quad (7)$$

PHASE INVESTIGATION

In case that phase $\phi = 0.\phi_1 \dots \phi_n$ the last circuit produces

$$F|\phi_1\phi_2 \dots \phi_n\rangle$$

and therefore if we apply the inverse quantum Fourier transform to the outcome of the circuit we get

$$|\phi_1\phi_2 \dots \phi_n\rangle$$

and therefore the outcome of the measurement is ϕ .

APPENDIX

EIGENVALUE ESTIMATION APPROACH to INTEGER FACTORIZATION

PRELIMINARIES - PHASE

In general, a **phase** is a complex number of the form $e^{2\pi i\omega}$, for a real ω .

For $x \in \{0, 1\}$ Hadamard transform

$$H|x\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{(-1)^x}{\sqrt{2}}|1\rangle = \frac{1}{\sqrt{2}} \sum_{y \in \{0,1\}} (-1)^{xy} |y\rangle$$

maps value x into the relative phase and its second application maps this relative phase back $|x\rangle$.

Similarly, Hadamard transform on n qubits maps $x \in \{0, 1\}^n$ into relative phases

$$H^{\otimes n}|x\rangle = \frac{1}{\sqrt{2^n}} \sum_{y \in \{0,1\}^n} (-1)^{x \cdot y} |y\rangle$$

and its second application maps the relative phases back to $|x\rangle$.

BASIC IDEAS - STEP by STEP

- We show that the following **Phase estimation problem** can be solved by a polynomial size circuit:

Input : The state $\frac{1}{\sqrt{2^n}} \sum_{y=0}^{2^n-1} e^{2\pi i \omega y} |y\rangle$.

Problem : Determine a good approximation of the phase parameter ω

An efficient solution of the above problem is given by a circuit with property that its inverse is an efficient implementation of Quantum Fourier Transform QFT_{2^n} .

- We show that the following **Eigenvalue estimation problem** can be solved by a polynomial size circuit:

Input A quantum circuit implementing an operator U and an eigenstate $|\psi\rangle$ of U with eigenvalue $e^{2\pi i \omega}$.

Problem : Find a good approximation of ω

This can be determined by showing that the above problem can be transformed to the phase estimation problem by designing at first efficiently the state

$$\frac{1}{\sqrt{2^n}} \sum_{y=0}^{2^n-1} e^{2\pi i \omega y} |y\rangle.$$

It is then shown that eigenvalue estimation circuit can then be applied in a useful way to any state $|\psi\rangle$. If this state is seen as being expressed in the form

$$|\psi\rangle = \sum_{j=0}^{2^n-1} \alpha_j |\psi_j\rangle$$

where $|\psi_j\rangle$ form an orthonormal basis of eigenvectors of U , then the eigenvalue estimation circuit maps it to a state

$$\sum_{j=0}^{2^n-1} \alpha_j |\omega'_j\rangle |\psi_j\rangle$$

where ω'_j is a good approximation of ω_j and by measurement one can get ω'_j with probability $|\alpha_j|^2$.

- It is shown that the operator

$$U_a : |s\rangle \rightarrow |sa \bmod N\rangle, \quad 0 \leq s \leq N,$$

where a and N are co-primes and r is the order of a , that is $a^r \equiv 1 \pmod{N}$, is

r -th root of the identity operator and its eigenvalues have the form

$$e^{2\pi i \frac{k}{r}}$$

for some integers k , and the corresponding eigenvectors are

$$|u_k\rangle = \frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} e^{-2\pi i \frac{k}{r} s} |a^s \bmod N\rangle.$$

Moreover, it can be shown that

$$|1\rangle = \sum_{k=0}^{r-1} |u_k\rangle$$

and therefore if the trick mentioned above is used one gets, with uniform probability, two approximations of $\frac{k}{r}$ that satisfy conditions of the following theorem on continued fractions that allows to find the correct value of r with probability at least 0.399.

Theorem Each rational number $\frac{x}{2^n}$ has a sequence of $\mathcal{O}(n)$ rational approximations, called convergents, $\frac{a_1}{b_1}, \frac{a_2}{b_2}, \dots, \frac{a_m}{b_m}$, where $\frac{a_m}{b_m} = \frac{x}{2^n}$, with the following properties:

- $a_1 < a_2 < \dots < a_m, b_1 < b_2 < \dots < b_m$.
- The above list of convergents can be computed in polynomial time.
- If some fraction $\frac{k}{r}$ satisfies

$$\left| \frac{x}{2^n} - \frac{k}{r} \right| \leq \frac{1}{2r^2}$$

then $\frac{k}{r}$ appears in the list of convergents of $\frac{x}{2^n}$.

FROM PHASE ESTIMATION to QFT CIRCUITS

We show now how to solve efficiently the phase estimation problem: to determine ω given the state

$$|\phi\rangle = \frac{1}{\sqrt{2^n}} \sum_{y=0}^{2^n-1} e^{2\pi i \omega y} |y\rangle$$

for the case $\omega = \frac{x}{2^n}$ and $x < 2^n$.

Our first starting observation is that if

$$\omega = 0.x_1x_2x_3\dots$$

then

$$e^{2\pi i \omega 2^k} = e^{2\pi i 2^k \omega} = e^{2\pi i (0.x_k x_{k+1} \dots)}.$$

Second observation is that for $n = 1$

$$H(|\phi\rangle) = H\left(\frac{1}{\sqrt{2^n}} \sum_{y=0}^{2^n-1} e^{2\pi i (0.x_1)y} |y\rangle\right) = H\left(\frac{1}{\sqrt{2}}(|0\rangle + (-1)^{x_1}|1\rangle)\right) = |x_1\rangle.$$

Last observation is that

$$|\phi\rangle = \frac{1}{\sqrt{2^n}} \sum_{y=0}^{2^n-1} e^{2\pi i \omega y} |y\rangle = \left(\frac{|0\rangle + e^{2\pi i (0.x_n x_{n+1} \dots)} |1\rangle}{\sqrt{2}}\right) \otimes \left(\frac{|0\rangle + e^{2\pi i (0.x_{n-1} x_n \dots)} |1\rangle}{\sqrt{2}}\right) \otimes \dots \otimes \left(\frac{|0\rangle + e^{2\pi i (0.x_1 x_2 \dots)} |1\rangle}{\sqrt{2}}\right)$$

ANOTHER OBSERVATION

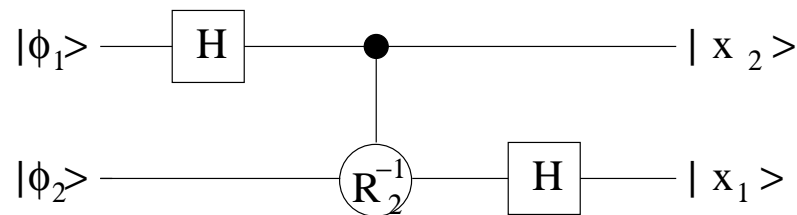
For the gates

$$R_2 = \begin{bmatrix} 1 & 0 \\ 0 & e^{\frac{2\pi i}{2^2}} \end{bmatrix}, \quad R_2^{-1} = \begin{bmatrix} 1 & 0 \\ 0 & e^{-2\pi i(0.01)} \end{bmatrix}$$

we have

$$R_2^{-1} \left(\frac{|0\rangle + e^{2\pi i(0.x_1)} |1\rangle}{\sqrt{2}} \right) = \frac{|0\rangle + e^{2\pi i(0.x_1)} |1\rangle}{\sqrt{2}}.$$

and therefore the circuit



solves the two qubit phase estimation problem, where

$$|\phi_1\rangle = \frac{1}{\sqrt{2}}(|0\rangle + e^{2\pi i(0.x_2)} |1\rangle) \quad |\phi_2\rangle = \frac{1}{\sqrt{2}}(|0\rangle + e^{2\pi i(0.x_1x_2)} |1\rangle)$$

PHASE ESTIMATION for 3-QUBITS

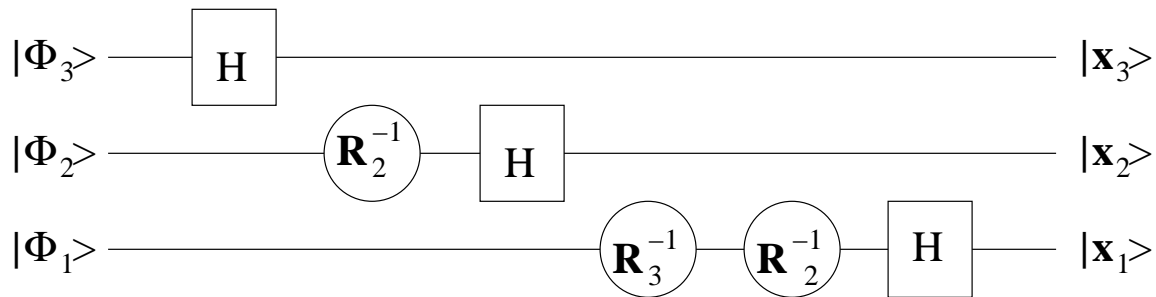
If we define the gate

$$R_k = \begin{bmatrix} 1 & 0 \\ 0 & e^{\frac{2\pi i}{2^k}} \end{bmatrix}$$

then

$$R_k^{-1} : |1\rangle \rightarrow e^{-2\pi i(0.0\dots 01^k)}|1\rangle, \quad R_k^{-1}|0\rangle = |0\rangle.$$

then 3-qubit phase estimation problem can be solved by the circuit



where

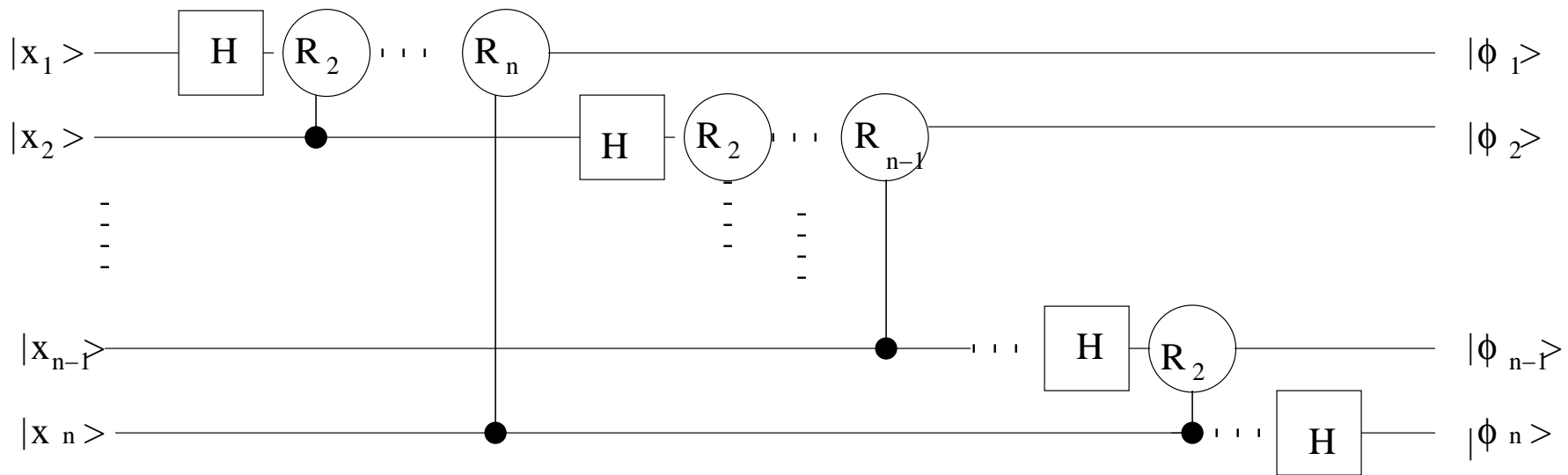
$$|\phi_3\rangle = \frac{1}{\sqrt{2}}(|0\rangle + e^{2\pi i(0.x_3)}|1\rangle) \quad |\phi_2\rangle = \frac{1}{\sqrt{2}}(|0\rangle + e^{2\pi i(0.x_2x_3)}|1\rangle) \quad |\phi_1\rangle = \frac{1}{\sqrt{2}}(|0\rangle + e^{2\pi i(0.x_1x_2x_3)}|1\rangle)$$

FROM PHASE ESTIMATION PROBLEM to QFT

A natural generalization of the above circuits leads to a circuit that solves phase estimation problem

$$\frac{1}{\sqrt{2^n}} \sum_{y=0}^{2^n-1} e^{2\pi i \frac{x}{2^n} y} |y\rangle \rightarrow |x\rangle$$

and its inverse is actually a circuit to implement $\text{QFT}_{2^n} |x\rangle \rightarrow \frac{1}{\sqrt{2^n}} \sum_{y=0}^{2^n-1} e^{2\pi i \frac{x}{2^n} y} |y\rangle$.



where

$$|\phi_i\rangle = \frac{1}{\sqrt{2}} (|0\rangle + e^{2\pi i (0.x_i \dots x_n)} |1\rangle).$$

GENERAL CASE of PHASE ESTIMATION

On the basis of the previous design of a circuit one can show for $m = 2^n$

Theorem There is a uniform family of circuits $C_{m,T}$ with the size polynomial in $\lg m$ and $\log T$ that implements QFT_m with error less than $\frac{1}{T}$.

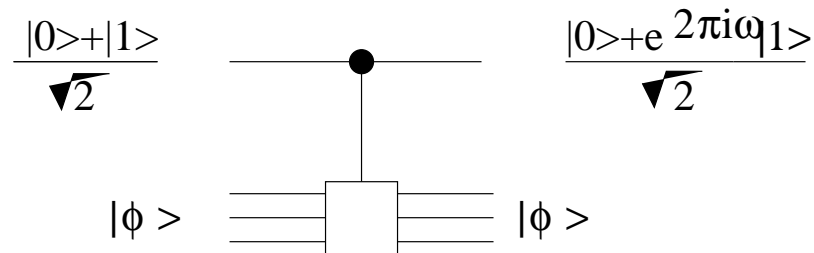
For a general value ω it can be shown

Theorem If $\frac{x}{2^n} \leq \omega \leq \frac{x+1}{2^n}$, then the phase estimation algorithm outputs one of x or $x + 1$ with probability at least $\frac{8}{2^n}$.

EIGENVALUE ESTIMATION

The eigenvalue estimation problem has as an input an operator (unitary) U and its eigenstate $|\phi\rangle$ that has $e^{2\pi i\omega}$ as the corresponding eigenvalue and the task is to determine ω .

Three observations lead to a solution. The first one is the circuit and second one the



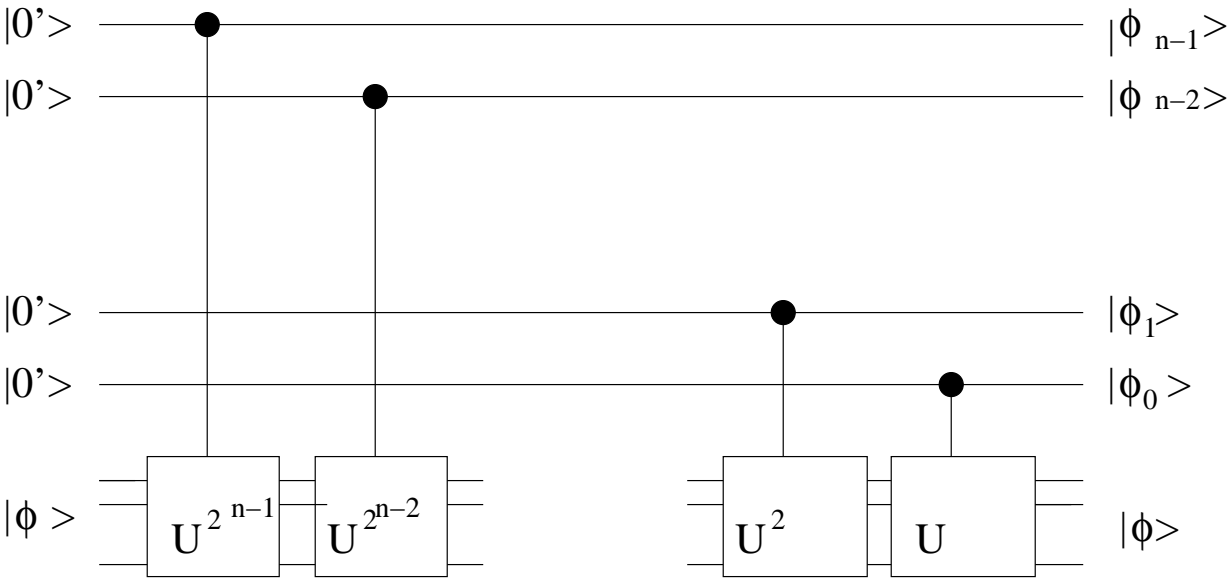
decomposition

$$\frac{1}{\sqrt{2}} \sum_{y=0}^{2^n-1} e^{2\pi i\omega y} = \left(\frac{|0\rangle + e^{2\pi i(2^{n-1}\omega)}|1\rangle}{\sqrt{2}} \right) \left(\frac{|0\rangle + e^{2\pi i(2^{n-2}\omega)}|1\rangle}{\sqrt{2}} \right) \cdots \left(\frac{|0\rangle + e^{2\pi i(2^0\omega)}|1\rangle}{\sqrt{2}} \right)$$

The last one is the fact that $|\phi\rangle$ is also eigenvector of the operator U^{2^j} with the eigenvalue $e^{2\pi i2^j\omega}$.

As a consequence the following circuit computes all factors of the above decomposition of the state

$$\frac{1}{\sqrt{2}} \sum_{y=0}^{2^n-1} e^{2\pi i \omega y}.$$

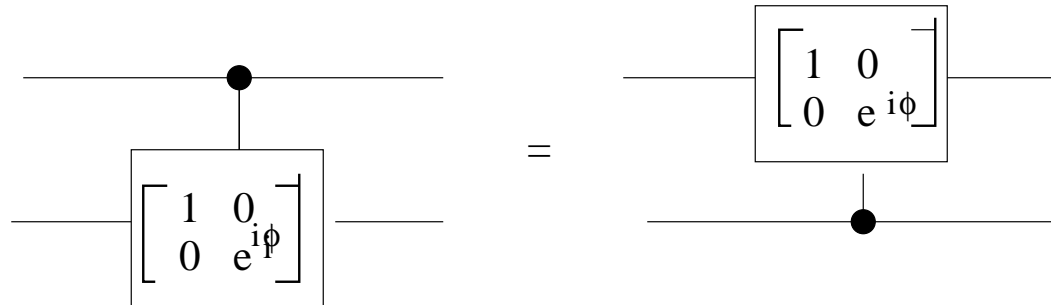


where

$$|\phi_i\rangle = \frac{|0\rangle + e^{2\pi i 2^i \omega} |1\rangle}{\sqrt{2}}$$

If we add at the end of the previous circuit a circuit to compute QFT^{-1} we get an eigenvalue estimation circuit.

The depiction of the last circuit can be simplified as follows



where

$$x = 2^{n-1}x_{n-1} + \dots + 2x_1 + x_0.$$

Eigenvalue estimation algorithm can therefore be summarized as follows:

1. Initialize an n qubit register to $|0\rangle^{\otimes n}$ - called it the *control register*.
2. Apply QFT to the control register.
3. Apply control U^x to the given eigenstate $|\phi\rangle$, controlled on the state of the control register.
4. Apply QFT^{-1} to the control register
5. Measure the control register to obtain a string of bits encoding the integer x .
Output $\frac{x}{2^n}$ as the estimate for ω .

In case the above circuit is applied not to an eigenstate $|\phi\rangle$, but to an arbitrary state $|\psi\rangle$ that can be expressed as a weighted superposition of eigenstates

$$|\psi\rangle = \sum_j \alpha_j |\phi_j\rangle$$

then the outcome, from both control and target registers would be the state

$$\sum_j \alpha_j |\bar{\omega}_j\rangle |\phi_j\rangle$$

A measurement of the output registers then provides a state

$$|\bar{\omega}_j\rangle |\phi_j\rangle$$

with probability $|\alpha_j|^2$

ORDER FINDING based on EIGENVALUE ESTIMATION

We know that factorization problem can be reduced to the problem of order finding.

To solve this problem one can proceed as follows:

For a given N and its co-prime a let us define the operator U_a as follows

$$U_a : |s\rangle \rightarrow |sa \bmod N\rangle, \quad 0 \leq d < N.$$

If r is the order of a with respect to N , then

$$U_a^r : |s\rangle \rightarrow |s\rangle.$$

that is U_a is an r th root of the identity operator and therefore the eigenvalues of U_a are the r th roots of 1, that is of the form $e^{2\pi i \frac{k}{r}}$ for some integer k .

It is then easy to show that states

$$|u_k\rangle = \frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} e^{-2\pi i \frac{k}{r} s} |a^s \bmod N\rangle$$

are eigenstates of U_a with eigenvalues $e^{2\pi i \frac{k}{r}}$.

By applying the eigenvalue estimation algorithm we would get, by last theorem, a good approximation of $\frac{k}{r}$ and that would allow, with high probability, to determine r .

However, without knowing r we actually do not know how to prepare the state $|u_k\rangle$. Fortunately, this is not necessary.

Indeed, it holds (as it is easy to verify),

$$\frac{1}{\sqrt{r}} \sum_{k=0}^{r-1} |u_k\rangle = |1\rangle$$

and therefore the eigenvalue estimation algorithm maps

$$|0\rangle|1\rangle = \frac{1}{\sqrt{r}} \sum_{k=0}^{r-1} |u_k\rangle|1\rangle \rightarrow \frac{1}{\sqrt{r}} \sum_{k=0}^{r-1} |k/r\rangle|u_k\rangle.$$

Therefore, by tracing out the second register from

$$|0\rangle|1\rangle = \frac{1}{\sqrt{r}} \sum_{k=0}^{r-1} |u_k\rangle|1\rangle \rightarrow \frac{1}{\sqrt{r}} \sum_{k=0}^{r-1} |k/r\rangle|u_k\rangle.$$

we get the first register into an equally weighted mixture of the states $|k/r\rangle$, for $k \in \{0, 1, \dots, r-1\}$.

Hence, by measuring the first register at the end of eigenvalue estimation algorithm we get an integer x such that $\frac{x}{2^n}$ is an estimate of $\frac{k}{r}$ for some uniformly random $k \in \{0, 1, \dots, r-1\}$.

This will allow, using continuous fraction algorithm, to determine with high probability the fraction $\frac{k}{r}$ and therefore also r .

The following slide contains the whole algorithm described in a concise way:

ORDER FINDING ALGORITHM

1. Choose an integer n so that $2^n \geq 2r^2$ ($n = \lceil 2\lg N \rceil$ will suffice).
2. Initialize n -qubit *control* register to $|0\rangle^{\otimes n}$.
3. Initialize n -qubit *target* register to the state $|1\rangle$ ($|0\dots 01\rangle$)
4. Apply QFT to the control register.
5. Apply control U_a^x operation to control and target registers.
6. Apply QFT^{-1} to the control registers.
7. Measure control register to get estimate $\frac{x_1}{2^n}$ of a random integer multiple of $\frac{1}{r}$.
8. Use the continued fractions algorithm to obtain integers c_1 and r_1 such that $\left| \frac{x_1}{2^n} - \frac{c_1}{r_1} \right| \leq \frac{1}{2^{\frac{n-1}{2}}}$. If no such pair of integers is found, output "FAIL".
9. Repeat Steps 1-8 to obtain another x_2 and another integers c_2 and r_2 such that $\left| \frac{x_2}{2^n} - \frac{c_2}{r_2} \right| \leq \frac{1}{2^{\frac{n-1}{2}}}$. If no such a pair of integers is found, output "FAIL".
10. Compute $r = \text{LCM}(r_1, r_2)$. Compute $a^r \bmod N$.
11. If $a^r \bmod n = 1$, then output r . Otherwise output "FAIL".