QUANTUM COMPUTING 6.

Jozef Gruska

Faculty of Informatics
Brno
Czech Republic

October 30, 2002

6. 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.

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

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

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

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

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

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

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

EXAMPLE

Let $n = 15$. Select $y < 15$ such that $gcd(y, 15) = 1$.
{The set of such y is $\{2, 4, 7, 8, 11, 13, 14\}$}

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

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

what gives $r = 2$.

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

Therefore

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

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

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

and the following algorithm fails.

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

EFFICIENCY of REDUCTION

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

$$Pr\{r \textit{ is even and } y^{r/2} \not\equiv \pm 1\} \geq \frac{1}{4}.$$

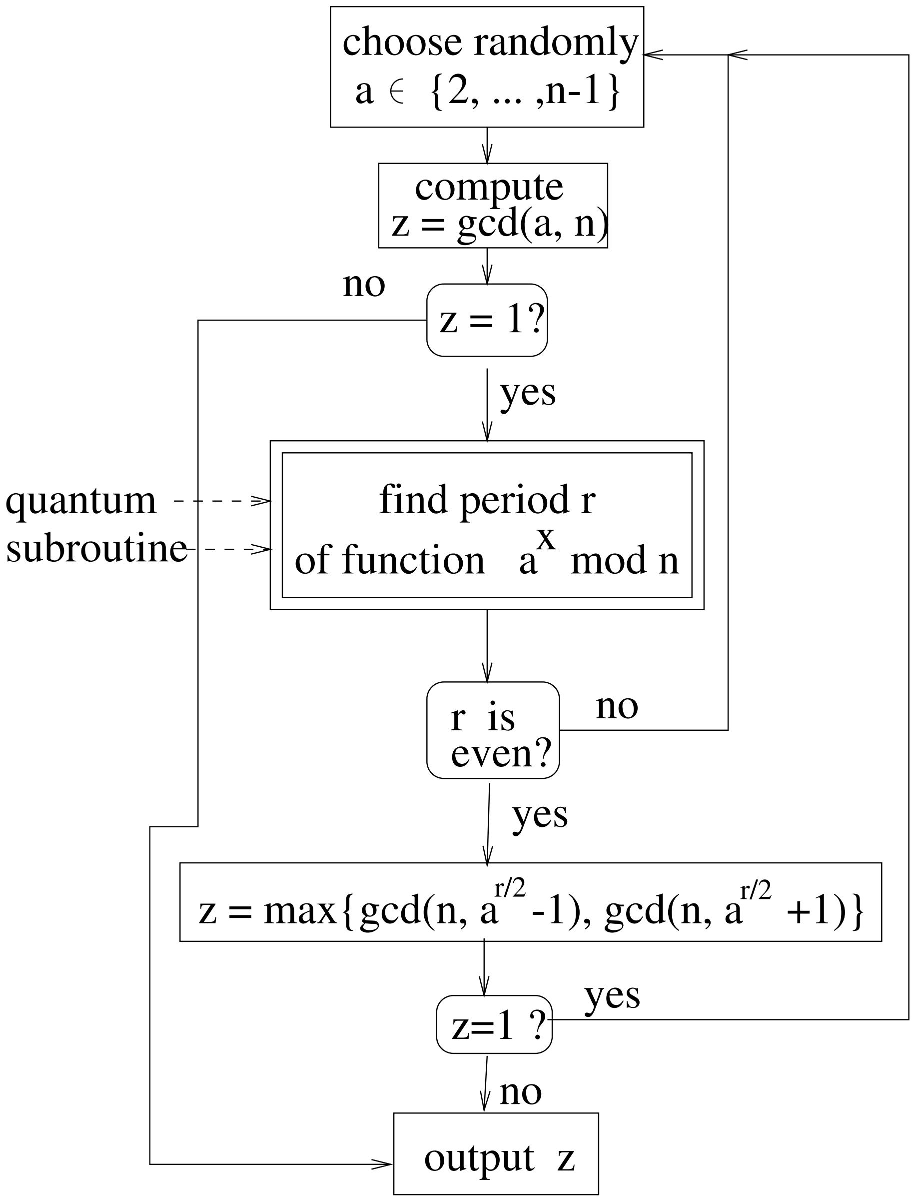
1. Choose randomly $1 < y < n$.
2. Compute $gcd(y, n)$. If $gcd(y, n) \neq 1$ we have a factor.
3. Find period r of function $y^k \bmod n$.
4. If r is odd or $y^{r/2} \equiv \pm 1 \pmod{n}$,then go to step 1; otherwise stop.

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

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

then there is a polynomial time randomized [quantum] algorithm to find non-trivial solution of the equation $x^2 \equiv 1 \pmod{n}$ (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.
$$\sqrt{\frac{r}{q}}\sum_{j=0}^{\frac{q}{r}-1}|jr+l\rangle$$
4. By applying quantum Fourier transformation we get then the state
$$\frac{1}{\sqrt{r}}\sum_{j=0}^{r-1}e^{2\pi ilj/r}|j\frac{q}{r}\rangle.$$
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 g 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 $n = 15, a = 7,\ f_{n,a}(x) = 7^x \mod 15,\ q = 16$, 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

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

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

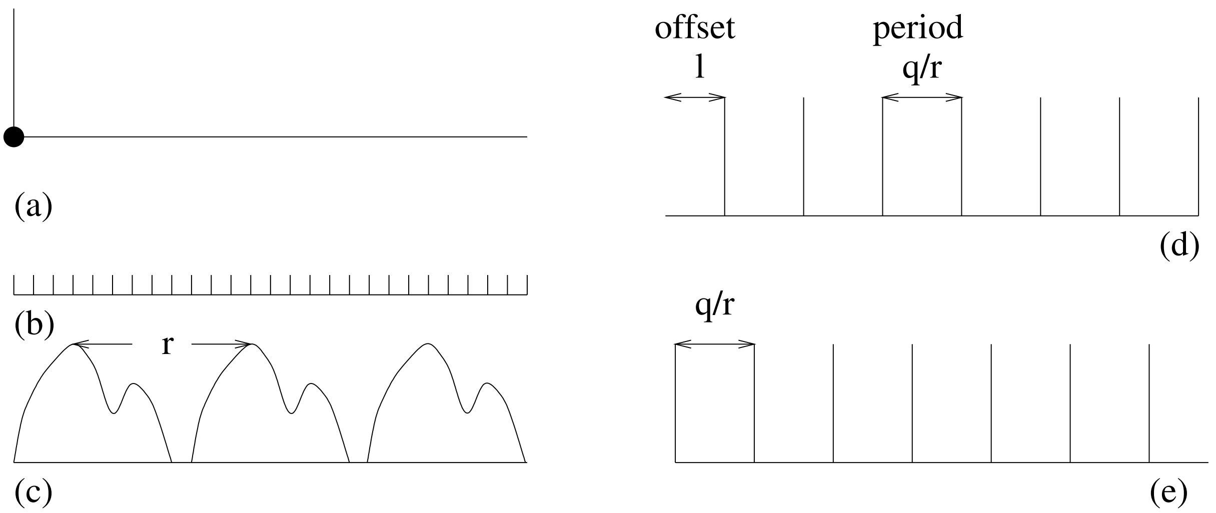


Figure 1: Graphical representation of steps of Shor's algorithm

QUANTUM FOURIER TRANSFORM

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

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

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

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

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

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

$$\mathrm{QFT}_q : |a\rangle \rightarrow \frac{1}{\sqrt{q}} \sum_{c=0}^{q-1} e^{2\pi iac/q} |c\rangle \tag{2}$$

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

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

where $\omega = e^{2\pi i/q}$ is the q**th root of unity**.
If applied to a quantum superposition, QFT_q performs as follows;

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

where $\bar{f}(c)$ is defined by (1).
Observe that

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

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. $y = a^l \bmod n$ for the smallest l with this property. If r is the period of $f_{n,a}$, then $a^l \equiv a^{jr+l} \pmod{n}$ for all j. Therefore, the measurement actually selects the sequence of x's values (in the fourth register), $l, l+r, l+2r, \ldots, l+Ar$, where A is the largest integer such that $l + Ar \leq q - 1$. Clearly, $A \approx \frac{q}{r}$. The post-measurement state is then

$$|\phi_l\rangle = \frac{1}{\sqrt{A+1}} \sum_{j=0}^{A} |n, a, q, jr + l, y\rangle = \frac{1}{\sqrt{A+1}} \sum_{j=0}^{A} |jr + l\rangle. \tag{3}$$

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 $A = \frac{q}{r} - 1$. In such a case the last state can be written in the form

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

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

$$\mathsf{QFT}_q|\phi_l\rangle = \frac{1}{\sqrt{q}} \sum_{c=0}^{q-1} \sqrt{\frac{r}{q}} \sum_{j=0}^{\frac{q}{r}-1} e^{2\pi i c(jr+l)/q}|c\rangle \tag{4}$$

$$= \frac{\sqrt{r}}{q} \sum_{c=0}^{q-1} e^{2\pi i l c/q} \left(\sum_{j=0}^{\frac{q}{r}-1} e^{2\pi i j c r/q} \right) |c\rangle = \sum_{c=0}^{q-1} \alpha_c |c\rangle. \tag{5}$$

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

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

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

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

and therefore

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

The key point is that the trouble-making offset l appears now in the phase factor $e^{2\pi i l j/r}$ 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).

[1] As 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 $\frac{1}{\sqrt{q}}\Sigma_{x=0}^{q-1}|x\rangle|f(x)\rangle$ and afterwords the measurement is performed on the first register to provide an $y_0 \in [0,\ldots,q-1]$.

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 $\xi_j \neq 0$);

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

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

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

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

The non-easy task is to show (see for example Lomonaco (2000)) 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 $\mathcal{O}(\lg q)$).
- The claim that circuits mentioned in the above item are time polynomially efficient for some major technology has been recently challenged.
- 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 polynomial time 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 transformation mapping

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

we get

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

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

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

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

Let us now measure the last register and let us denote by y the value we have receive.

Let us now determine, for afixed c and d, the probability Pr that we get $y \equiv g^k \bmod p$ for some k.

The probability Pr equals the square of the absolute value of the sum of all amplitudes of states with y in the last register, and (fixed) c and d in the first two registers,i.e., Pr is the square of the sum of amplitudes over all a and b satisfying the equality $a - rb \equiv k \pmod{p-1}$ for some k.

This is due to the fact that the computational paths interfere only if $y \equiv g^a (g^r)^{-b} = g^{a-rb} \equiv g^k \bmod p$.

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

$$g^{a-rb} = g^{j(p-1)+k} \equiv g^k \pmod{p}.$$

The probability is therefore

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

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

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

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

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

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

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

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

Therefore

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

For the probability *Pr* that, for fixed b and c we get by measurement y is therefore

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

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

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

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

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

$$d+rc \not\equiv 0 \bmod (p-1)$$

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

$$(p-1)^{-1}e^{(2\pi ikc/(p-1))}.$$

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

- 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 $f : G \to R$, 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 cossets 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, $G = \mathbf{Z}_2$, $f : \{0,1\} \to \{0,1\}$,
$x - y \in G_0 \Leftrightarrow f(x) = f(y)$. Decide whether $G_0 = \{0\}$ (and f is balanced) or $G_0 = \{0,1\}$ (and f is constant).

Simon's problem, $G = \mathbf{Z}_2^n$, $f : G \to R$. $x - y \in G_0 \Leftrightarrow f(x) = f(y)$, $G_0 = \{0^{(n)}, s\}$, $s \in \mathbf{Z}_2^n$. 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, $G = \mathbf{Z}$, $a \in \mathbf{N}$, $f(x) = a^x$,
$x - y \in G_0 \Leftrightarrow f(x) = f(y)$, $G_0 = \{rk \,|\, k \in \mathbf{Z}$ for the smallest r such that $a^r = 1.\}$ Find r.

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

[2] A 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.