Quantum computation

PART 3.

VIII. Finite quantum automata.

As seen before, to design an algorithm on a quantum Turing machine one needs almost superhuman programming skills. Moreover, the question of termination of the computation was not answered in a satisfactory way. Therefore we shall introduce another model of the quantum computation, finite quantum automata.

Let \( \Sigma = \{ \sigma_1, \ldots, \sigma_d \} \) be a finite alphabet and \( \mathcal{H}_d \) the \( d \)-dimensional complex Hilbert space with orthonormal basis \( \{|\sigma_1\rangle, \ldots, |\sigma_d\rangle\} \).

The set of unit length elements in \( \mathcal{H}_d \) are denoted by \( \mathcal{H}^1_d \). A \( d \)-ary quantum digit \( q \) is a mapping \( q : \mathbb{Z} \rightarrow \mathcal{H}^1_d \). Value \( q(t) \) is called the value of quantum digit \( q \) at time \( t \). We use here domain \( \mathbb{Z} \) instead of \( \mathbb{R} \) to indicate that the computation process is performed in discrete steps.

A \( d \)-ary quantum register of length \( m \) is a mapping
\[
r : \mathbb{Z} \rightarrow \mathcal{H}_d \otimes \mathcal{H}_d \otimes \ldots \otimes \mathcal{H}_d = \mathcal{H}_{d^m}.
\]

Space \( \mathcal{H}_{d^m} \) has orthonormal basis
\[
\{|\sigma_{i_1}\rangle \otimes \ldots \otimes |\sigma_{i_m}\rangle | \sigma_{i_j} \in Q\}
\]
so we can identify a \( d \)-ary quantum register of length \( m \) by \( d^m \)-ary quantum digit by
\[
|\sigma_{i_1}\rangle \otimes \ldots \otimes |\sigma_{i_m}\rangle \rightarrow |\sigma_{i_1} \ldots \sigma_{i_m}\rangle.
\]

Binary quantum digits are called qubits.

A finite quantum automaton is a quintuple
\[
\mathcal{F} = (\Sigma, \Gamma, M, W, R),
\]
where \( \Sigma \) and \( \Gamma \) are alphabets of cardinalities \( d \) and \( d' \), \( M \) is memory of a constant size \( K \), \( W \) is the workspace of constant size \( L \) and \( R \) is a finite sequence of the computation rules.

Memory \( M \) is a \( d \)-ary quantum register of length \( K \), workspace is a \( d' \)-ary quantum register of length \( L \) and \( R \) is a sequence whose each member is either a unitary mapping \( U : M \otimes L \rightarrow M \otimes L \) affecting on at most \( 3d \)-dimensional subspace of \( M \) or an observation rule which is specified by a subset of \( \{1, 2, \ldots, K\} \). The configurations, the configuration space and the superpositions are defined as for the quantum Turing machine. The computation of a finite quantum automaton is a sequence of superpositions where each member is determined by the previous one according to the rules of the computation. The time complexity of a finite quantum automaton is defined to be the cardinality of \( R \). The following statements are not too hard to prove:
Theorem VII.1. For any unitary mapping $U : M \otimes W \rightarrow M \otimes W$ there exists a two-tape quantum Turing machine whose finite model performs the computation $c \rightarrow Uc$.

Theorem VII.2. Assume that a two-tape Turing machine $\mathcal{M}$ uses only a constant time for inputs less than a fixed positive integer $M$. Then there exists a finite quantum automaton that simulates $\mathcal{M}$.

VIII. Quantum parallelism.

Assume that $f$ is a Turing-computable function

$$f : \{0, 1, \ldots, 2^m - 1\} \rightarrow \{0, 1, \ldots, 2^n - 1\}.$$ 

All numbers in the domain and range can naturally be represented by using binary quantum registers of $m$ and $n$ respectively. If $n < m$, the function cannot be injective and the computation therefore is not reversible in general. However, allowing workspace large enough, the computation can be forced to be reversible. Consequently there exists a finite quantum automaton $A_f$ using no observation rules that performs the computation

$$A_f \ket{i} \otimes 0 = \ket{f(i)} \otimes w_f,$$

where $A_f$ is a product of unitary mappings each acting on at most $3d$-dimensional subspace of $\mathcal{H}_{2^m}$.

Let $A$ be an unitary matrix given by

$$A = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}.$$

We shall extend $A$ into $A'$ by adding new rules $A_i$ where

$$A_i = I \otimes I \otimes \ldots \otimes A \otimes \ldots \otimes I$$

where $A$ occurs in the $i$th position. Thus, beginning with a configuration

$$\ket{0} \otimes \ket{0} \otimes \ldots \otimes \ket{0} \quad (7-1)$$

it is easy to show, for example by induction on $m$, that after mappings $A_1, A_2, \ldots, A_m$ the state of (7-1) will become

$$\frac{1}{\sqrt{2^m}} (\ket{0} + \ket{1} + \ldots + \ket{2^m - 1}).$$

Notice that here we have not written the workspace explicitely. Then, continuing the computation of $A'$ we will obtain

$$\frac{1}{\sqrt{2^m}} (\ket{f(0)} + \ket{f(1)} + \ldots + \ket{f(2^m - 1)}). \quad (7-2)$$
the time increase was only $m$, but here we have all the values of $f$ (exponentially many in $m$) equally balanced. This phenomenon is called quantum parallelism. Alas, the observation of memory register in (7-2) will yield value $f(i)$ chosen equiprobably and the information of all other values is lost. Thus the quantum parallelism actually does not mean parallel computation. However, some interesting facts about function $f$ may be found.

IX. The discrete Fourier transform.

Here $\mathbb{Z}_n = \mathbb{A}/n\mathbb{Z}$ is the residue class ring of $n$ elements. A well-known fact is that mapping $e : \mathbb{Z}_n \to \mathbb{C}$ defined by $e(x) = e^{\frac{2\pi i x}{n}}$ is an additive character of $\mathbb{Z}_n$ and that all additive characters of $\mathbb{Z}_n$ are of form $\psi_a(x) = e(ax)$. The orthogonality of the characters is also a well-known fact:

$$\sum_{a \in \mathbb{Z}_n} e(ax) = \begin{cases} 0, & \text{if } x \neq 0 \\ n, & \text{if } x = 0. \end{cases}$$

**Definition.** Let $A : \mathbb{Z} \to \mathbb{C}$ be any function. The discrete Fourier transform of $A$ is a function $\mathcal{F}A : \mathbb{Z} \to \mathbb{C}$ given by

$$\mathcal{F}A(x) = \frac{1}{\sqrt{n}} \sum_{y \in \mathbb{Z}_n} A(y)e(xy).$$

The inverse discrete Fourier transform of $A$ is a function $\mathcal{F}^{-1}A : \mathbb{Z}_n \to \mathbb{C}$ defined by

$$\mathcal{F}^{-1}A(x) = \frac{1}{\sqrt{n}} \sum_{y \in \mathbb{Z}_n} A(y)e(-xy).$$

Using the orthogonality of characters we get easily $\mathcal{F}\mathcal{F}^{-1}A = \mathcal{F}^{-1}\mathcal{F}A = A$.

We will now assume that $n = 2^m$ and show how to perform the discrete Fourier transform on $\mathbb{Z}_n$, when the input values are encoded in the weights of a quantum superposition of numbers $0, 1, \ldots, 2^m$. This means that initially the register contains

$$A(0) |0\rangle + A(1) |1\rangle + \ldots + A(2^m - 1) |2^m - 1\rangle,$$

where

$$\sum_{i=0}^{2^m-1} |A(i)|^2 = 1$$

and the desired outcome of a unitary transform is

$$\frac{1}{\sqrt{2^m}}(\mathcal{F}A(0) |0\rangle + \mathcal{F}A(1) |1\rangle + \ldots + \mathcal{F}A(2^m - 1) |2^m - 1\rangle).$$

By linearity it suffices to establish a unitary mapping $U_{\text{QFT}}$ that does the operation

$$U_{\text{QFT}} |a\rangle = \frac{1}{\sqrt{2^m}} \sum_{b=0}^{2^m-1} e^{\frac{2\pi i ab}{2^m}} |b\rangle.$$
We will denote the Hilbert space that forms the binary register by
\[ H = H_{m-1} \otimes H_{m-1} \otimes \ldots \otimes H_0, \]
and assume that the most significant bit is represented by \( H_{m-1} \). Mapping \( A_i \) is defined as in section VIII. Moreover, for each pair \( j, k \) where \( 0 \leq j < k \leq m - 1 \) unitary mapping \( B_{jk} \) is defined by
\[
B_{jk} = \begin{pmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & e^{2\pi i k / (2^m)}
\end{pmatrix}.
\]
Matrix \( B_{jk} \) is written on basis \( \{ |0\rangle \otimes |0\rangle, |0\rangle \otimes |1\rangle, |1\rangle \otimes |0\rangle, |1\rangle \otimes |1\rangle \} \). Clearly \( B_{jk} \) affects on the bits at locations \( j \) and \( k \) only if they both are 1.

The following lemma can be proved for example on induction on \( m \).

**Lemma IX.1.** Let
\[
U_{\text{QFT}} = \prod_{i=0}^{m-1} A_i \prod_{j=i+1}^{m-1} B_{i,j}
\]
be a product of \( \frac{m(m+1)}{2} \) matrices and \( a_{m-1}2^{m-1} + a_{m-2}2^{m-2} + \ldots + a_0 \) the binary representation of \( a \). Then
\[
U_{\text{QFT}} |a\rangle = \frac{1}{\sqrt{2^m}} \sum_{b=0}^{2^m-1} e^{2\pi i b a / 2^m} |b_0\rangle \otimes |b_1\rangle \otimes \ldots \otimes |b_{m-1}\rangle,
\]
where \( b_{m-1}2^{m-1} + b_{m-2}2^{m-2} + \ldots + b_{0} \) is the inverse binary representation of \( b \).

By using \( \lfloor \frac{n}{2} \rfloor \) transpositios which clearly are unitary we can inverse the order of the bits of the register. As a consequence we obtain

**Theorem IX.2.** Quantum Fourier transform on binary register of length \( m \) can be done in time \( O(m^2) \).

Note that the quantum Fourier transform (QFT) operated here on the weights of the register, not on the contents, so finite quantum automaton does not solve the Fourier transform for the input data. Although, QFT is used in the Shor’s factoring algorithm.

**I0. Shor’s algorithm for factoring numbers.**

Given two primes \( p \) and \( q \) it is easy to compute the product \( pq \) by the elementary multiplication algorithm. The time requested for this is \( O(\max\{|p|, |q|\}^2) \), where \( |p| \) is the length of non-unary representation of \( p \) which is of order \( \log p \). However, solving the inverse problem, find out \( p \) and \( q \) when the product \( n = pq \) is given is not known to be solvable with any polynomial time algorithm so far.

The problem of finding the prime factors of a composite number is of course interesting number theoretically, but also in the cryptography. For instance, a
widely used public-key cryptosystem RSA is based on the assumed non-tractability of factorization.

Divisibility by a prime number is a periodic property; every \( p \)th number is divisible by \( p \). Then it should not appear as a surprise that factoring reduces to determining a period of a function.

Peter W. Shor described in 1994 [PART 1.] a probabilistic polynomial time quantum algorithm for factoring integer numbers with two or more different prime factors.

**Shor’s factoring algorithm.** The input is an odd composite number \( N \) which is not a pure prime power. The aim is to find a nontrivial factor of \( N \).

1° Choose an arbitrary natural number \( a < N \).

2° Compute \( d = \gcd(a, N) \).

3° If \( d > 1 \) then output \( d \) and stop.

4° Compute the period of function \( f_N(x) = a^x \pmod{N} \) and denote this by \( r \).

5° If \( r \) is odd or \( a^\frac{r}{2} \equiv \pm 1 \pmod{N} \) then output “failure” and stop, else output \( \gcd(a^\frac{r}{2} \pm 1, N) \).

It is easy to check that everything else but step 4° can be done with classical algorithm in cubic time. Note that if \( N \) is even, the multipliers 2 can easily be detected and divided. There are also polynomial time classical probabilistic algorithms for recognizing pure prime powers.

We will verify that the algorithm works correctly if the output is not “failure”. So, suppose that \( r = \text{ord}_N(a) \) is an even number and that \( a^\frac{r}{2} \not\equiv \pm 1 \pmod{N} \). Then \( N \) divides \( a^r - 1 = (a^\frac{r}{2} - 1)(a^\frac{r}{2} + 1) \), but \( N \) does not divide factors \( a^\frac{r}{2} - 1 \) and \( a^\frac{r}{2} + 1 \). It follows that \( N \) has common prime factors with \( a^\frac{r}{2} + 1 \) and with \( a^\frac{r}{2} - 1 \) and those factors can be found by using the Euclidean’s algorithm that works in cubic time.

\( N \) cannot have more prime factors than \( c|N| \) where \( c \) is a constant independent of \( N \), so repeating the algorithm \( O(N) \) times we get the prime decomposition of \( N \).

The order of a number \( a \) modulo \( N \) is denoted by \( \text{ord}_N(a) \). For the estimate of the probability of output “failure” we represent the following statement:

**Lemma III.7.1.** Let \( N = p_1^{a_1} \cdots p_k^{a_k} \), \( p_i \not\equiv p_j \) unless \( i = j \) be the decomposition of an odd integer into the prime factors and \( a \) a random number such that \( \gcd(N, a) = 1 \). The probability \( P(r = \text{ord}_N(a) \) is even and \( a^\frac{r}{2} \not\equiv \pm 1 \pmod{N} \) \) is greater than \( 1 - \frac{1}{2k-1} \).

**Proof.** The claim follows if we can show that

\[
P(r = \text{ord}_N(a) \) is odd or \( a^\frac{r}{2} \equiv \pm 1 \pmod{N} \) \geq \frac{1}{2k-1}.
\]

In fact we can reduce the latter condition into \( a^\frac{r}{2} \equiv -1 \pmod{N} \) since \( r \) is the least positive number such that \( a^r \equiv 1 \pmod{N} \). Since all numbers \( p_i^{a_i} \) are coprime to each other, the following equivalence holds:

\[
a^\frac{r}{2} \equiv -1 \pmod{N} \iff a^{a_i} \equiv -1 \pmod{p_i^{a_i}} \text{ for each } i.
\]
Let us denote $r_i = \text{ord}_{p_i^a}(a)$. Then clearly $r = \text{lcm}(r_1, \ldots, r_k)$. If we denote $r_i = 2^{t_i} s_i$ where $s_i$ is odd and

$$s = \text{lcm}(s_1, \ldots, s_k)$$

$$t = \max(t_1, \ldots, t_k)$$

then clearly $r = 2^t s$ where $s$ is odd. Next we show that the implication

$$a^\frac{r}{2} \equiv -1 \pmod{p_i^{a_i}} \implies t_i = t$$

also holds. For the sake of contradiction, suppose that $a^\frac{r}{2} \equiv -1$ but $t_i < t$. This means that 2 divides $r$ at least once more than it divides $r_i$, so $r_i | \frac{r}{2}$ because each $s_i$ divides $s$. Thus we can write $\frac{r}{2} = r_i \cdot c$ and compute:

$$1 \equiv a^{r_i} \equiv a^{cr_i} = a^\frac{r}{2} \equiv -1 \pmod{p_i^{a_i}},$$

which is a contradiction since primes 2 was excluded case.

As a conclusion we get equivalences

$$r \text{ odd } \iff \text{each } r_i \text{ odd } \iff t_i = 0 \text{ for all } i.$$  

Using (10-1) we see that

$$P(r \text{ odd or } a^\frac{r}{2} \equiv -1 \pmod{N}) \leq P(\text{all the numbers } t_i \text{ are equal}).$$

Finally we claim that the probability $P(t_i = j)$ cannot be greater than $\frac{1}{2}$ for any numbers $i$ and $j$. To see this we recall that group $\mathbb{Z}_{p^a}^*$ is cyclic, when $p \neq 2$. Let $\gamma$ be a generator of $\mathbb{Z}_{p^a}^*$. From assumption $p \neq 2$ we get that $|\mathbb{Z}_{p^a}^*| = \varphi(p^a) = p^{a-1}(p-1)$ is an even number, let us denote $\varphi(p^a) = 2^m l$ with odd $l$. All elements in $\mathbb{Z}_{p^a}^*$ can be represented uniquely in the form $\gamma^i$ where $i \in \{1, 2, \ldots, 2^m l\}$. Moreover,

$$\text{ord}_{p^a} (\gamma^i) = \frac{2^m l}{\gcd(i, 2^m l)}.$$  

So an order of form $2^m k$ with odd $k$ occurs if and only if $i = 2^{n-m} k$ where $k$ is odd. But multiples of $2^{n-m}$ with odd multiplier occur in the sequence 1, 2, ..., $2^m l$ at most

$$\frac{1}{2} \cdot \frac{2^m l}{2^{n-m}} = 2^{m-1} l \leq |\mathbb{Z}_{p^a}^*| \quad \frac{2^m l}{2^n}$$

times. As a conclusion we get

$$P(\text{all the numbers } t_i \text{ are equal}) = \sum_{j=0}^{n} \prod_{i=1}^{k} P(t_i = j) = P(t_1 = j) \prod_{i=2}^{k} P(t_i = t_1) \leq \prod_{i=2}^{k} \frac{1}{2} = \frac{1}{2^{k-1}}.$$
which completes the proof.

The bound in the lemma is strict with respect to \( k \) in the sense that it can be shown that for prime powers \( N = p^a \) we always have \( r = \text{ord}_N(a) \) either odd or \( a^{\frac{N}{2}} \equiv -1 \pmod{N} \).

We will now see how to design the quantum algorithm for finding the order of an element modulo \( N \). First we choose a number \( M = 2^L \) such that \( N^2 \leq M < 2N^2 \), which is clearly possible. The quantum automaton uses binary memory register of size \( 2L \) (logarithmic in \( N \)), and the automaton is designed to work in time that is polynomial in \( L \).

The memory is thought to be divided into two registers of length \( L \), say \( q_1 \) and \( q_2 \), and the input of the automaton will be the configuration

\[
c = |0\rangle \otimes \ldots \otimes |0\rangle \otimes |n_1\rangle \otimes \ldots \otimes |n_L\rangle \otimes |*\rangle \otimes \ldots \otimes |*\rangle,
\]

where \( n_1, \ldots, n_L \) consists the binary representation of the arbitrarily chosen \( a \) and \( N \). In the continuation we will not write the workspace explicitly and denote this input by \(|0\rangle \otimes |a,N\rangle\).

The first rules of the computation are \( A_1, \ldots, A_L \). The outcome will be

\[
\frac{1}{\sqrt{M}}(|0\rangle \otimes |a,N\rangle + |1\rangle \otimes |a,N\rangle + \ldots + |M-1\rangle \otimes |a,N\rangle),
\]

thus register \( q_1 \) contains the equally weighted superposition of all values in \( \{0,1,\ldots,M-1\} \).

The next set of rules will make use of the reversible Turing machine \( M \) computing the function \( f(c,a,N) = a^c \pmod{N} \). Input \( c \) is taken from register \( q_1 \) and the output is stored in \( q_2 \). After this operation, the contents of the memory will be

\[
\frac{1}{\sqrt{M}}(\sum_{c=0}^{M-1} |c\rangle \otimes |a^c\rangle),
\]

where \( \overline{a^c} \) is the least positive remainder of \( a^c \) modulo \( N \). The next computational rule is an observation rule determined by the set \( \{L+1,\ldots,2L\} \), that means an observation of the register \( q_2 \). If \( r \) is the order of \( a \) modulo \( N \), then \( a^l \equiv a^{l+jr} \pmod{N} \) for all \( j \) and the post-observation value of the memory will be

\[
\frac{1}{K+1} \sum_{j=0}^{K} |jr+l\rangle \otimes |\overline{a^l}\rangle,
\]

where \( l \) has been chosen probabilistically in the observation procedure and \( K \) is the unique integer such that \( M - r \leq l + Kr < M \) which is equivalent of \( \frac{M-r}{r} - 1 \leq K < \frac{M-1}{r} \). Also \( K + 1 \leq \frac{M}{r} \) holds. We can the write (10 - 2) as

\[
\sum_{b=0}^{M-1} f(b) |b\rangle,
\]
where
\[
 f(b) = \begin{cases} 
 \frac{1}{\sqrt{K+1}} & \text{if } b = jr + l \text{ for some } j \\
 0 & \text{otherwise.}
\end{cases}
\]

So \( f \) is a periodic function of period \( r \). However, the periodicity may be slightly ruined in the end of the “cycle” when we identify \( M \) and 0. Next rules used will be the operations forming \( QFT \). The result will be
\[
\sum_{b=0}^{M-1} \left( \frac{1}{\sqrt{M}} \sum_{a=0}^{M-1} f(a)e^{\frac{2\pi i ab}{M}} \right) |b\rangle,
\]
which can be estimated by
\[
|\tilde{f}(b)| = \frac{1}{\sqrt{M}} \sum_{j=0}^{K} \frac{1}{\sqrt{K+1}} e^{\frac{2\pi i (jr+l)b}{M}}.
\]

The next rule is an observation at register \( q_1 \). The probability of seeing \( b \) is given by
\[
P(b) = \frac{1}{M} \left( \sum_{j=0}^{K} e^{\frac{2\pi i jr b}{M}} \right)^2,
\]
which can be estimated by
\[
P(b) = \frac{1}{M^2} \left( \sum_{j=0}^{K} e^{\frac{2\pi i jr b}{M}} \right)^2 \geq \frac{r}{M^2} \left( \sum_{j=0}^{K} e^{\frac{2\pi i j b}{M}} \right)^2.
\]

Now we interrupt the analysis of the general case and, in order to understand the situation better, consider a special case where \( r \) divides \( M \) exactly. Then obviously \( K = \frac{M}{r} - 1 \) and (10-4) becomes
\[
\tilde{f}(b) = \frac{1}{\sqrt{M}} \sum_{j=0}^{\frac{M}{r} - 1} \frac{1}{\sqrt{M/r}} e^{\frac{2\pi i (jr+l)b}{M}}
\]
and (10-5) becomes
\[
P(b) = \frac{1}{M^2} \left( \sum_{j=0}^{\frac{M}{r} - 1} e^{\frac{2\pi i j b}{M/r}} \right)^2
= \frac{r}{M^2} \left( \sum_{j=0}^{\frac{M}{r} - 1} e^{\frac{2\pi i j b}{M/r}} \right)^2
\]

Thus the observation of (10-3) cannot yield any other values but \( k \frac{M}{r} \), where \( k \) is chosen from the set \( \{0, 1, \ldots, r - 1\} \) each with probability \( \frac{1}{r} \). If the observed value
\( b' = k' \frac{M}{r} \) satisfies \( \gcd(k', r) = 1 \) then we can find \( r \) by cancelling \( \frac{k'}{M} = \frac{k'}{r} \) into an irreducible fraction (recall that this can be done with help of Euclidean algorithm in cubic time). It still remains to study the probability of having \( \gcd(k', r) = 1 \), but now we turn into the general case.

In the special case we could observe only those values of \( b \) where \( br - kM = 0 \) for some \( k \in \{0, 1, \ldots, r - 1\} \). In the general case we will seek for the values \( b \) where

\[
|br - kM| \leq \frac{r}{2}.
\]

(10-6)

It is not too hard to show that there are \( r \) such \( b \) values for which we can find \( k \) to satisfy (10-6) as in the special case there were \( r \) possible values to be observed. For such a \( b \) inequality (10-5) becomes

\[
P(b) \geq \frac{r}{M^2} \left| \sum_{j=0}^{K} e^{2\pi i (\varphi_b) j M} \right|^2,
\]

(10-7)

where \( \varphi_b = rb - kM \). Evaluating (10-7) we get

\[
P(b) \geq \frac{r}{M^2} \frac{\sin^2 \frac{\pi \varphi_b}{M} (K + 1)}{\sin^2 \frac{\pi \varphi_b}{2M}}.
\]

(10-8)

Considering (10-8) as a function of \( \varphi_b \) and keeping (10-6) in mind we notice that (10-8) gets the lowest possible value at the points \( \varphi_b = \pm \frac{\pi}{2} \). Therefore

\[
P(b) \geq \frac{r}{M^2} \frac{\sin^2 \frac{\pi \varphi_b}{2M} (K + 1)}{\sin^2 \frac{\pi \varphi_b}{2M}}.
\]

(10-9)

Since \( \frac{M}{r} - 1 < K + 1 \leq \frac{M}{r} \),

\[
\frac{\pi}{2} \left( 1 - \frac{r}{M} \right) < \frac{\pi r}{2M} (K + 1) \leq \frac{\pi}{2}.
\]

Because \( \frac{r}{M} \leq \frac{N}{M} \leq \frac{1}{N} \), the numerator of (10-9) is very close to 1. For instance, is we require \( N \geq 2^{10} \), then \( \sin^2 \frac{\pi r}{2M} (K + 1) \geq 1 - 10^{-9} \). Similarly \( \frac{\pi r}{2M} \) is close to zero and the estimation \( \sin x \leq x \) holds. Thus

\[
P(b) \geq \frac{r}{M^2} \frac{1 - 10^{-9}}{(\frac{\pi r}{2M})^2} \geq \frac{4}{10} \frac{1}{r}.
\]

Since there are \( r \) different values of \( b \) such that (10-6) holds, the possibility of observing one is greater than \( \frac{2}{5} \). It can be shown that \( \frac{\varphi(b)}{r} \geq c \frac{\log \log r}{\log \log N} \), so we can find a \( b \) value such that \( \gcd(k, r) = 1 \) for the corresponding \( k \) and (10-6) holds with probability greater than

\[
\frac{2}{5} \frac{c}{\log \log r} \geq \frac{2}{5} \frac{c}{\log \log N}.
\]
Both conditions above can be tested in cubic time with classical algorithm. The probability of failure is \(1 - \frac{2c}{5 \log |N|}\). Suppose that the failure probability should be at most \(\varepsilon\) and this happens after \(t\) repeats. Then

\[
(1 - \frac{2c}{5 \log |N|})^t \leq \varepsilon,
\]

and using \(\log(1 - x) \leq -x\) we get

\[
t \geq \frac{5 \log |N| \log \varepsilon^{-1}}{2c}.
\]

Thus repeating the algorithm \(\mathcal{O}(\log |N| \log \varepsilon^{-1})\) times we can find a suitable \(b\) with probability at least \(1 - \varepsilon\).

Finally, condition (10-6) can be written as

\[
\left| \frac{b}{M} - \frac{k}{r} \right| \leq \frac{1}{2M}.
\]  \hspace{1cm} (10-10)

Now \(b\) and \(M\) are known, and there is exactly one fraction \(\frac{k}{r}\) with denominator at most \(N\) in the range (10-10). This can be found efficiently using the continued fraction expansion of \(\frac{b}{M}\).