We are going to extend this simple result in three ways. First, we will assume that W measures f with some error probability ε; we will find out with what precision the above procedure corresponds to a garbage-free measurement (the answer is $\sqrt{2\varepsilon}$). Second, the formula for the conditional probabilities (12.4) makes sense for an arbitrary orthogonal decomposition $\mathcal{B}^{\otimes N} = \bigoplus_{y\in\Delta} \mathcal{M}_y$. Third, instead of copying the result, we can apply any operator V that is measuring with respect to the indicated decomposition. The copying corresponds to $V : |y, z, v\rangle \mapsto |y, z,\ y \oplus v\rangle$.

[2!] **Problem 12.2.** Let $\mathcal{N} = \bigoplus_{j\in\Omega} \mathcal{L}_j$ and $\mathcal{B}^{\otimes N} = \bigoplus_{y\in\Delta} \mathcal{M}_y$ be orthogonal decompositions. Consider two measuring operators on $\mathcal{N} \otimes \mathcal{K} \otimes \mathcal{B}^{\otimes N}$, such that $\mathcal{B}^{\otimes N}$ serves as the "instrument" for one and the "object" for the other:

$$W = \sum_{j\in\Omega} \Pi_{\mathcal{L}_j} \otimes I_{\mathcal{K}} \otimes R_j, \qquad V = \sum_{y\in\Delta} I_{\mathcal{N}} \otimes Q_y \otimes \Pi_{\mathcal{M}_y}.$$

Suppose that W *measures a function* $f : \Omega \to \Delta$ *with error probability* $\leq \varepsilon$, i.e., the conditional probabilities $\mathbf{P}(y|j) = \langle 0^N | R_j^\dagger \Pi_{\mathcal{M}_y} R_j | 0^N \rangle$ satisfy $\mathbf{P}(f(j)|j) \geq 1 - \varepsilon$. Then the operator $\tilde{U} = W^{-1}VW$ approximates the operator

$$U = \sum_{j\in\Omega} \Pi_{\mathcal{L}_j} \otimes Q_{f(j)} \ : \ \mathcal{N} \otimes \mathcal{K} \to \mathcal{N} \otimes \mathcal{K}$$

with precision $2\sqrt{\varepsilon}$, using $\mathcal{B}^{\otimes N}$ as the ancillary space. If V is the copy operator, then the precision is $\sqrt{2\varepsilon}$.

13. Quantum algorithms for Abelian groups

The only nontrivial quantum algorithm we have considered so far is Grover's algorithm for the solution of the universal search problem (see Section 9.2). Unfortunately, there we achieved only a polynomial increase in speed. For this reason Grover's algorithm does not yield any serious consequences (of type BQP $\supset$ BPP) for complexity theory. At present time, there is no proof that quantum computation is super-polynomially faster than classical probabilistic computation. But there are several pieces of indirect evidence in favor of such an assertion. The first of these is an example of a *problem with oracle* (cf. Definition 2.2 on page 26 and the beginning of Section 9.2), for which there exists a polynomial quantum algorithm, while any classical probabilistic algorithm is exponential.[10] This example, constructed by D. Simon [**66**], is called the hidden subgroup problem for $(\mathbb{Z}_2)^k$. The famous

[10] One should keep in mind that the complexity of problems with oracle frequently differs from the complexity of ordinary computational problems. A classical example is the theorem stating that IP = PSPACE [**58, 59**]. The oracle analogue of this assertion is not true [**25**]!

factoring algorithm by P. Shor [**62**] is based on a similar idea. After discussing these examples, we will solve the hidden subgroup problem for the group $\mathbb{Z}^k$, which generalizes both results.

THE HIDDEN SUBGROUP PROBLEM. Let G be a group with a specified representation of its elements by binary words. There is a device (an oracle) that computes some function $f : G \to \mathcal{B}^n$ with the following property:

$$f(x) = f(y) \iff x - y \in D, \tag{13.1}$$

where $D \subseteq G$ is an initially unknown subgroup. It is required to find that subgroup. (The result should be presented in a specified way.)

13.1. The problem of hidden subgroup in $(\mathbb{Z}_2)^k$; Simon's algorithm. We consider the problem formulated above for the group $G = (\mathbb{Z}_2)^k$. The elements of this group can be represented by length k words of zeros and ones; the group operation is bitwise addition modulo 2. We may regard G as the k-dimensional linear space over the field $\mathbb{F}_2$. Any subgroup of G is a linear subspace, so it can be represented by a basis.

It is easy to show that a "hidden subgroup" cannot be found quickly using a classical probabilistic machine. (A classical machine sends words $x_1, \dots, x_l$ to the input of the "black box" and receives answers $y_1, \dots, y_l$. Each subsequent query x_j depends on the previous answers $y_1, \dots, y_{j-1}$ and some random number r that is generated in advance.)

Proposition 13.1. *Let $n \geq k$. For any classical probabilistic algorithm making no more than $2^{k/2}$ queries to the oracle, there exist a subgroup $D \subseteq (\mathbb{Z}_2)^k$ and a corresponding function $f : (\mathbb{Z}_2)^k \to \mathcal{B}^n$ for which the algorithm is wrong with probability $> 1/3$.*

Proof. For the same subgroup D there exist several different oracles f. We assume that one of them is chosen randomly and uniformly. (If the algorithm is wrong with probability $> 1/3$ for the randomized oracle, then it will be wrong with probability $> 1/3$ for some particular oracle.) The randomized oracle works as follows. If the present query is x_j, and $x_j - x_s \in D$ for some $s < j$, the answer y_j coincides with the answer y_s that was given before. Otherwise, y_j is uniformly distributed over the set $\mathcal{B}^n \setminus \{y_1, \dots, y_{j-1}\}$. The randomized oracle is not an oracle in the proper sense, meaning that its answer may depend on the previous queries rather than only on the current one. In this manner, the randomized oracle is equivalent to a device with memory, which, being asked the question x_j, responds with the smallest number $s_j \leq j$ such that $x_j - x_{s_j} \in D$. Indeed, if we have a classical probabilistic machine making queries to the randomized oracle, we can adapt it for the use with the device just described. For this, the machine should be altered in such a way that it will transform each answer s_j to y_j and

proceed as before. (That is, $y_j = y_s$ if $s_j < j$, or y_j is uniformly distributed over $\mathcal{B}^n \setminus \{y_1, \ldots, y_{j-1}\}$ if $s_j = j$.)

Let the total number of queries be $l \leq 2^{k/2}$. Without loss of generality all queries can be assumed different. In the case $D = \{0\}$, all answers are also different, i.e., $s_j = j$ for all j. Now we consider the case $D = \{0, z\}$, where z is chosen randomly, with equal probabilities, from the set of all nonzero elements of the group $(\mathbb{Z}_2)^k$. Then, regardless of the algorithm that is used, $z \notin \{x_j - x_1, \ldots, x_j - x_{j-1}\}$ with probability $\geq 1 - (j-1)/(2^k - 1)$. The condition for z implies that $s_j = j$. This is true for all $j = 1, \ldots, l$ with probability $\geq 1 - l(l-1)/(2(2^{k-1} - 1)) > 1/2$. Recall that we have two random parameters: z and r. We can fix z in such a way that the probability of obtaining the answers $s_j = j$ (for all j) will still be greater than $1/2$. Let us see what a classical machine would do in such a circumstance. If it gives the answer "$D = \{0\}$" with probability $\geq 2/3$, we set $D = \{0, z\}$, and then the resulting answer will be wrong with probability $> (2/3) \cdot (1/2) = 1/3$. If, however, the probability of the answer "$D = \{0\}$" is smaller than $2/3$, we set $D = \{0\}$. □

Let us define a quantum analog of the oracle f. The corresponding quantum oracle is a unitary operator

$$U : |x, y\rangle \to |x, y \oplus f(x)\rangle \tag{13.2}$$

($\oplus$ denotes bitwise addition). We note that the quantum oracle allows linear combinations of the various queries; therefore it is possible to use it more efficiently than the classical oracle.

Now we describe Simon's algorithm for finding the hidden subgroup in $\mathbb{Z}_2^k$. Let $E = G/D$ and let E^* be the group of characters on E. (By definition, a *character* is a homomorphism $E \to \mathbf{U}(1)$.) In the case $G = (\mathbb{Z}_2)^k$ we can characterize the group E^* in the following manner:

$$E^* = \{h \in (\mathbb{Z}_2)^k \text{ such that } h \cdot z = 0 \text{ for all } z \in D\},$$

where $h \cdot z$ denotes the inner product modulo 2. (The character corresponding to h has the form $z \mapsto (-1)^{h \cdot z}$.) Let us show how one can generate a random element $h \in E^*$ using the operator U. After generating sufficiently many random elements, we will find the group E^*, hence the desired subgroup D.

We begin by preparing the state

$$|\xi\rangle = 2^{-k/2} \sum_{x \in G} |x\rangle = H^{\otimes k} |0^k\rangle$$

in the first quantum register. In the second register we place the state $|0^n\rangle$ and apply the operator U. Then we discard the second register, i.e., we will

no longer make use of its contents. Thus we obtain the mixed state

$$\rho = \mathrm{Tr}_2\Big(U\big(|\xi\rangle\langle\xi| \otimes |0\rangle\langle 0|\big)U^\dagger\Big) = 2^{-k} \sum_{x,y:\, x-y\in D} |x\rangle\langle y|.$$

Now we apply the operator $H^{\otimes k}$ to the remaining first register. This yields a new mixed state

$$\gamma = H^{\otimes k}\rho H^{\otimes k} = 2^{-2k} \sum_{a,b} \sum_{x,y:\, x-y\in D} (-1)^{a\cdot x - b\cdot y} |a\rangle\langle b|.$$

It is easy to see that $\sum_{x,y:\, x-y\in D}(-1)^{a\cdot x-b\cdot y}$ is different from zero only in the case where $a = b \in E^*$. Hence

$$\gamma = \frac{1}{|E^*|} \sum_{a\in E^*} |a\rangle\langle a|.$$

This is precisely the density matrix corresponding to the uniform probability distribution on the group E^*. It remains to apply the following lemma, which we formulate as a problem.

[2!] **Problem 13.1.** Let $h_1, \ldots, h_l$ be independent uniformly distributed random elements of an Abelian group X. Prove that they generate the entire group X with probability $\geq 1 - |X|/2^l$.

Therefore, $2k$ random elements suffice to generate the entire group E^* with probability of error $\leq 2^{-k}$, where "error" refers to the case where $h_1, \ldots, h_{2k}$ actually generate a proper subgroup of E^*. (Note that such a small — compared to $1/3$ — probability of error is obtained without much additional expense. To make it still smaller, it is most efficient to use the standard procedure: repeat all calculations several times and choose the most frequent outcome.)

Summing up, to find the "hidden subgroup" D, we need $O(k)$ queries to the quantum oracle. The overall complexity of the algorithm is $O(k^3)$.

13.2. Factoring and finding the period for raising to a power. A second piece of evidence in favor of the hypothesis BQP $\supset$ BPP is the fast quantum algorithm for factoring integers into primes and for another number-theoretic problem — finding the discrete logarithm. They were found by P. Shor [**62**]. Let us discuss the first of these two problems.

Factoring (an integer into primes). Suppose we are given a positive integer y. It is required to find its decomposition into prime factors

$$y = p_1^{\alpha_1} p_2^{\alpha_2} \cdots p_k^{\alpha_k}.$$

This problem is thought to be so complex that practical cryptographic algorithms are based on the hypothetical difficulty of its solution. From the theoretical viewpoint, the situation is somewhat worse: there is neither a

reduction of problems of class NP to the factoring problem, nor any other "direct" evidence in favor of its complexity. (The word "direct" is put in quotation marks because at present the answer to the question $\mathrm{P} \stackrel{?}{=} \mathrm{NP}$ is unknown.) Therefore, the conjecture about the complexity of the factoring problem complements the abundant collection of unproved conjectures in the computational complexity theory. It is desirable to decrease the number of such problems. Shor's result is a significant step in this direction: if we commit an "act of faith" and believe in the complexity of the factoring problem, then the need for yet another act of faith (regarding the greater computational power of the quantum computer) disappears.

We will construct a fast quantum algorithm for solving not the factoring problem, but another problem, called PERIOD FINDING, to which the factoring problem is reduced with the aid of a classical probabilistic algorithm.

PERIOD FINDING. Suppose we are given an integer $q > 1$ that can be written using at most n binary digits (i.e., $q < 2^n$) and another integer a such that $1 \leq a < q$ and $\gcd(a, q) = 1$ (where $\gcd(a, q)$ denotes the greatest common divisor). It is required to find the period of a with respect to q, i.e., the smallest nonnegative number t such that $a^t \equiv 1 \pmod{q}$.

In other words, the period is the order of the number a in the multiplicative group of residues $(\mathbb{Z}/q\mathbb{Z})^*$. We will denote the period of a with respect to q by $\mathrm{per}_q(a)$.

Below we will examine a quantum algorithm for the solution of the period finding problem. But we will begin by describing the classical probabilistic reduction of the factoring problem to the problem of finding the period. We suggest that the reader reviews the probabilistic test for primality presented in Part 1 (see Section 4.2).

13.3. Reduction of factoring to period finding. Thus, let us assume that we know how to find the period. It is clear that we can factor the number y by running $O(\log y)$ times a subprogram which, for any composite number, finds a nontrivial divisor with probability at least $1/2$. (Of course, it is also necessary to use the standard procedure for amplification of success probability; see formula (4.1) on p. 37 and the paragraph preceding it.)

Procedure for finding a nontrivial divisor.

Input. An integer y $(y > 1)$.

Step 1. Check y for parity. If y is even, then give the answer "2"; otherwise proceed to Step 2.

Step 2. Check whether y is the k-th power of an integer for $k = 2, \dots, \log_2 y$. If $y = m^k$, then give the answer "m"; otherwise proceed to Step 3.

Step 3. Choose an integer a randomly and uniformly between 1 and $y-1$. Compute $b = \gcd(a, q)$ (say, by Euclid's algorithm). If $b > 1$, then give the answer "b"; otherwise proceed to Step 4.

Step 4. Compute $r = \mathrm{per}_y(a)$ (using the period finding algorithm that we assume we have). If r is odd, then the answer is "y is prime" (which means that we give up finding a nontrivial divisor). Otherwise proceed to Step 5.

Step 5. Compute $d = \gcd(a^{r/2} - 1, y)$. If $d > 1$, then the answer is "d"; otherwise the answer is "y is prime".

Analysis of the divisor finding procedure. If the above procedure yields a number, it is a nontrivial divisor of y. The procedure fails and gives the answer "y is prime" in two cases: 1) when $r = \mathrm{per}_y(a)$ is odd, or 2) when r is even but $\gcd(a^{r/2} - 1, y) = 1$, i.e., $a^{r/2} - 1$ is invertible modulo y. However, $(a^{r/2} + 1)(a^{r/2} - 1) \equiv a^r - 1 \equiv 0 \pmod{y}$, hence $a^{r/2} + 1 \equiv 0 \pmod{y}$ in this case. The converse is also true: if r is even and $a^{r/2} + 1 \equiv 0 \pmod{y}$, then the answer is "$y$ is prime".

Let us prove that our procedure succeeds with probability at least $1 - 1/2^{k-1}$, where k is the number of distinct prime divisors of y. (Note that this probability vanishes for prime y, so that the the procedure also works as a primality test.) In the proof we will need the Chinese Remainder Theorem (Theorem A.5 on page 241) and the fact that the multiplicative group of residues modulo p^α, p prime, is cyclic (see Theorem A.11).

Let $y = \prod_{j=1}^k p_j^{\alpha_j}$ be the decomposition of y into prime factors. We introduce the notation

$$a_j \equiv a \pmod{p_j^{\alpha_j}}, \qquad r_j = \mathrm{per}_{(p_j^{\alpha_j})}\, a_j = 2^{s_j} r'_j, \quad \text{where } r'_j \text{ is odd.}$$

By the Chinese Remainder Theorem, r is the least common multiple of all the r_j. Hence $r = 2^s r'$, where $s = \max\{s_1, \dots, s_k\}$ and r' is odd.

We now prove that the procedure yields the answer "y is prime" if and only if $s_1 = s_2 = \dots = s_k$. Indeed, if $s_1 = \dots = s_k = 0$, then r is odd. If $s_1 = \dots = s_k \geq 1$, then r is even, but $a_j^{r_j/2} \equiv -1 \pmod{p_j^{\alpha_j}}$ (using the cyclicity of the group $(\mathbb{Z}/p_j^{\alpha_j}\mathbb{Z})^*$), hence $a^{r/2} \equiv -1 \pmod{y}$ (using the Chinese Remainder Theorem). Thus the procedure yields the answer "y is prime" in both cases. Conversely, if not all the s_j are equal, then r is even and $s_m < s$ for some m, so that $a_m^{r/2} \equiv 1 \pmod{p_m^{\alpha_m}}$. Hence $a^{r/2} \not\equiv -1 \pmod{y}$, i.e., the procedure yields a nontrivial divisor.

To give a lower bound of the success probability, we may assume that the procedure has reached Step 4. Thus a is chosen according to the uniform distribution over the group $(\mathbb{Z}/q\mathbb{Z})^*$. By the Chinese Remainder Theorem, the uniform random choice of a is the same as the independent uniform random choice of $a_j \in (\mathbb{Z}/p_j^{\alpha_j}\mathbb{Z})^*$ for each j. Let us fix j, choose some $s \geq 0$ and estimate the probability of the event $s_j = s$ for the uniform distribution of a_j. Let g_j be a generator of the cyclic group $(\mathbb{Z}/p_j^{\alpha_j}\mathbb{Z})^*$. The order of this group may be represented as $p_j^{\alpha_j} - p_j^{\alpha_j - 1} = 2^{t_j} q_j$, where q_j is odd. Then

$$\begin{aligned} \bigl|\{a_j : s_j = s\}\bigr| &= \Bigl|\bigl\{g_j^l : l = 2^{t_j - s} m, \text{ where } m \text{ is odd}\bigr\}\Bigr| \\ &= \begin{cases} q_j & \text{if } s = 0, \\ (2^s - 2^{s-1}) q_j & \text{if } s = 1, \ldots, t_j. \end{cases} \end{aligned}$$

For any given s, the probability of the event $s_j = s$ does not exceed $1/2$. Now let $s = s_1$ be a random number (depending on a_1); then $\mathbf{Pr}[s_j = s] \leq 1/2$ for $j = 2, \ldots, k$. It follows that

$$\mathbf{Pr}[s_1 = s_2 = \cdots = s_k] \leq (1/2)^{k-1}.$$

This yields the desired estimate of the success probability for the entire procedure: with probability at least $1 - 1/2^{k-1}$ the procedure finds a nontrivial divisor of y.

13.4. Quantum algorithm for finding the period: the basic idea. Thus, the problem is this: given the numbers q and a, construct a polynomial size quantum circuit that computes $\mathrm{per}_q(a)$ with error probability $\epsilon \leq 1/3$. The circuit will operate on a single n-qubit register, as well as on many other qubits, some of which may be considered classical. The n-qubit register is meant to represent residues modulo q (recall that $q < 2^n$).

Let us examine the operator that multiplies the residues by a, acting by the rule

$$U_a : |x\rangle \mapsto |ax \bmod q\rangle.$$

(A more accurate notation would be $U_{q,a}$, indicating the dependence on q. However, q is fixed throughout the computation, so we suppress it from the subscript. We keep a because we will also use the operators U_b for arbitrary b.) This operator permutes the basis vectors for $0 \leq x < q$ (recall that $(a, q) = 1$). However, we represent $|x\rangle$ by n qubits, so x may take any value between 0 and $2^n - 1$. We will assume that the operator U_a acts trivially on such basis vectors, i.e., $U_a : |x\rangle = |x\rangle$ for $q \leq x < 2^n$.

Since for the multiplication of the residues there is a Boolean circuit of polynomial — $O(n^2)$ — size, there is a quantum circuit (with ancillas) of about the same size.

The permutation given by the operator U_a can be decomposed into cycles. The cycle containing 1 is $(1, a, a^2, \ldots, a^{\mathrm{per}_q(a)-1})$; it has length $\mathrm{per}_q(a)$. The algorithm we are discussing begins at the state $|1\rangle$, to which the operator U_a gets applied many times. But such transformations do not take us beyond the orbit of 1 (the set of elements which constitute the cycle described above). Therefore we consider the restriction of the operator U_a to the subspace generated by the orbit of 1.

$$\text{Eigenvalues of } U_a: \quad \lambda_k = e^{2\pi i \cdot k/t}, \text{ where } t \text{ is the period;}$$

$$\text{Eigenvectors of } U_a: \quad |\xi_k\rangle = \frac{1}{\sqrt{t}} \sum_{m=0}^{t-1} e^{-2\pi i \cdot km/t} |a^m\rangle.$$

It is easy to verify that the vectors $|\xi_k\rangle$ are indeed eigenvectors. It suffices to note that the multiplication by a leads to a shift of the indices in the sum. If we change the variable of summation in order to remove this shift, we get the factor $e^{2\pi i \cdot k/t}$.

If we are able to measure the eigenvalues of the operator U_a, then we can obtain the numbers k/t. First let us analyze how this will help us in determining the period.

Suppose we have a machine which in each run gives us the number k/t, where t is the sought-for period and k is a random number uniformly distributed over the set $\{0, \ldots, t-1\}$. We suppose that k/t is represented as an irreducible fraction k'/t' (if the machine were able to give the number in the form k/t, there would be no problem at all).

Having obtained several fractions of the form k'_1/t'_1, $k'_2/t'_2, \ldots, k'_l/t'_l$, we can, with high probability, find the number t by reducing these fractions to a common denominator.

Lemma 13.2. *If $l \geq 2$ fractions are obtained, then the probability that their least common denominator is different from t is less than $3 \cdot 2^{-l}$.*

Proof. The fractions k'_1/t'_1, $k'_2/t'_2, \ldots, k'_l/t'_l$ can be obtained as reductions of fractions $k_1/t, \ldots, k_l/t$ (i.e., $k'_j/t'_j = k_j/t$), where $k_1, \ldots, k_l$ are independently distributed random numbers. The least common multiple of $t'_1, \ldots, t'_l$ equals t if and only if the greatest common divisor of $k_1, \ldots, k_l$ and t is equal to 1.

The probability that $k_1, \ldots, k_l$ have a common prime divisor p does not exceed $1/p^l$. Therefore the probability of not getting t after reducing to a common denominator does not exceed $\sum_{k=2}^{\infty} \frac{1}{k^l} < 3 \cdot 2^{-l}$ (the range of the index k in this sum obviously includes all prime divisors of t). □

Now we construct the machine M that generates the number k/t (in the form of an irreducible fraction) for random uniformly distributed k.

This will be a quantum circuit which realizes the measuring operator $W = \sum_{k=0}^{t-1} V_k \otimes \Pi_{\mathcal{L}_k}$, where $\mathcal{L}_k = \mathbb{C}(|\xi_k\rangle)$, the subspace generated by $|\xi_k\rangle$. The operators V_k are the form $|0\rangle \mapsto \sum_{y,z} |y, z\rangle$, where y is an irreducible fraction and z is garbage. In this, the conditional probabilities should satisfy the inequality

$$\mathbf{P}\Big(\boxed{\tfrac{k}{t}}\,\Big|\,k\Big) \stackrel{\text{def}}{=} \sum_z \Big|\Big\langle \boxed{\tfrac{k}{t}}, z\,\Big|\,V_k\Big|0\Big\rangle\Big|^2 \geq 1-\varepsilon, \tag{13.3}$$

where $\boxed{\frac{k}{t}}$ denotes the irreducible fraction equal to the rational number k/t.

The construction of such a measuring circuit is rather complex, so we first explain how it is used to generate the outcome y with the desired probability $w_y = \sum_{k\in M_y} \frac{1}{t}$, where $M_y = \Big\{k : \boxed{\frac{k}{t}} = y\Big\}$. Let us take the state $|1\rangle$ as the initial state. A direct computation (the reader is advised to carry it through) shows that

$$|1\rangle = \frac{1}{\sqrt{t}} \sum_{k=0}^{t-1} |\xi_k\rangle.$$

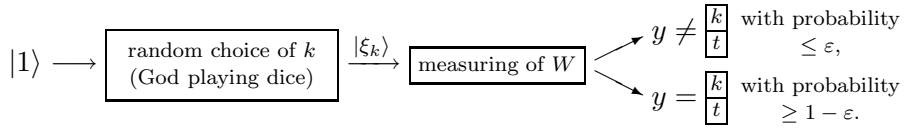
If we perform the measurement on this state, then by the formula for total probability we obtain

$$\mathbf{Pr}[\text{outcome} = y] \;= \mathbf{P}\big(W(|0\rangle \otimes |1\rangle), y\big) = \sum_k \mathbf{P}(y|k)\, \mathbf{P}(|1\rangle, \mathcal{L}_k).$$

The probabilities of all $|\xi_k\rangle$ are equal: $\mathbf{P}(|1\rangle, \mathcal{L}_k) = |\langle \xi_k|1\rangle|^2 = 1/t$, which corresponds to the uniform distribution of k. The property (13.3) guarantees that we obtain the outcome $\boxed{\frac{k}{t}}$ with probability $\geq 1-\varepsilon$. Well, the reader may find this statement not rigorous because k does not have a certain value. To be completely pedantic, we need to derive an inequality similar to (10.4), namely,

$$\sum_y \Big|\mathbf{Pr}[\text{outcome} = y] - w_y\Big| \leq 2\epsilon.$$

Schematically, the machine M functions as follows:



The random choice of k happens automatically, without applying any operator whatsoever. Indeed, the formula of total probability is arranged in such a way as if: before the measurement begins, a random k was generated, which then remains constant. (Of course, the formula is only true when the operator W is measuring with respect to the given subspaces $\mathcal{L}_k$.)

13.5. The phase estimation procedure. Now we will construct the operator that measures the eigenvalues of U_a. The eigenvalues have the form

$$\lambda_k = e^{2\pi i \varphi_k}, \quad \text{where } \varphi_k = \frac{k}{t} \bmod 1.$$

The phase φ_k is a real number modulo 1, i.e., $\varphi_k \in \mathbb{R}/\mathbb{Z}$. (The set $\mathbb{R}/\mathbb{Z}$ can be conveniently represented as a circle of unit length.) The procedure for determining φ_k is called *phase estimation*.

As we already mentioned, we can limit ourselves to the study of the action of the operator U_a on the input vector $|\xi_k\rangle$. The construction is divided into four stages.

1. We construct a measuring operator such that the conditional probabilities depend on the value of $\varphi = \varphi_k$. Thus a single use of this operator will give us *some information* about φ (like flipping a biased coin tells something about the bias, though inconclusively) (see 13.5.1).
2. We localize the value of φ *with modest precision*. It is the moment to emphasize that, in all the arguments, there are two parameters: the *probability of error* ε and the *precision* δ. As the result of a measurement, we obtain some number y, for which the condition $|y-\varphi|_{\bmod 1} < \delta$ must hold with probability at least $1-\varepsilon$. (Here $|\cdot|_{\bmod 1}$ denotes the distance on the unit length circle, e.g., $|0.1 - 0.9|_{\bmod 1} = 0.2$.) For the time being, a modest precision will do, say $\delta = 1/16$ (see 13.5.2).
3. Now we must *increase the precision*. Specifically, we determine φ with precision $1/2^{2n+2}$ (see 13.5.3).
4. We need to pass from the approximate value of φ to the exact one, represented *in the form of an irreducible fraction*. It is essential to be able to distinguish between numbers of the form $\varphi = k/t$, where $0 \le k < t < 2^n$. Notice that if $k_1/t_1 \ne k_2/t_2$, then $|k_1/t_1 - k_2/t_2|_{\bmod 1} \ge 1/(t_1 t_2) > 1/2^{2n}$. Therefore, knowing $\varphi = k/t$ with precision $1/2^{2n+1}$, one can, in principle, determine its exact value. Moreover, this can be done efficiently by the use of continued fractions (see 13.5.4).

At stage 3, we will use the operator U_b for *arbitrary* b (not just for $b = a$, the number for which we seek the period). To this end, we introduce an operator U that sends $|b, x\rangle$ to $|b,\ bx \bmod q\rangle$ whenever $\gcd(b, q) = 1$. How the operator U acts in the remaining cases is not important; this action can be defined in an arbitrary computationally trivial way, so that U be represented by a quantum circuit of size $O(n^2)$. In fact, all the earlier arguments about the simulation of Boolean circuits by quantum circuits hold true for the simulation of circuits that compute partially defined functions.

[1] **Problem 13.2.** Using the operator U, realize the operator $\Lambda(U_b)$ for arbitrary b relatively prime to q.

13.5.1. ***How to get some information about the phase.*** In Section 12 we introduced the operator $\Xi(U_a) = (H \otimes I)\Lambda(U_a)(H \otimes I)$, which measures the eigenvalues. In our case $\lambda_k = e^{2\pi i\varphi_k}$ and we can write the operator in the form

$$\Xi(U_a) = \sum_k V_k \otimes \Pi_{\mathcal{L}_k}, \quad V_k = \frac{1}{2}\begin{pmatrix} 1+e^{2\pi i\varphi_k} & 1-e^{2\pi i\varphi_k} \\ 1-e^{2\pi i\varphi_k} & 1+e^{2\pi i\varphi_k} \end{pmatrix},$$

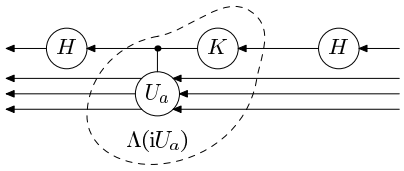
and its action in the form

$$|0\rangle \otimes |\xi_k\rangle \xrightarrow{\Xi(U_a)} \left(\frac{1+e^{2\pi i\varphi_k}}{2}|0\rangle + \frac{1-e^{2\pi i\varphi_k}}{2}|1\rangle\right) \otimes |\xi_k\rangle,$$

so that for the conditional probabilities we get the expressions

$$\mathbf{P}(0|k) = \left|\frac{1+e^{2\pi i\varphi_k}}{2}\right|^2 = \frac{1+\cos(2\pi\varphi_k)}{2}, \qquad \mathbf{P}(1|k) = \frac{1-\cos(2\pi\varphi_k)}{2}.$$

Although the conditional probabilities depend on φ_k, they do not allow one to distinguish between $\varphi_k = \varphi$ and $\varphi_k = -\varphi$. That is why another type of measurement is needed. We will use the operator $\Xi(iU_a)$. Its realization is shown in the diagram below. It uses the operator $K = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix}$ from



the standard basis. The encircled part of the diagram realizes the operator $\Lambda(iU_a)$. Indeed, K multiplies only $|1\rangle$ by i, but this is just the case where the operator U_a is applied (by the definition of $\Lambda(U_a)$). For the operator $\Xi(iU_a)$ the conditional probabilities are

$$\mathbf{P}(0|k) = \frac{1-\sin(2\pi\varphi_k)}{2}, \qquad \mathbf{P}(1|k) = \frac{1+\sin(2\pi\varphi_k)}{2}.$$

The complexity of the realization of the operators $\Xi(U_a)$ and $\Xi(iU_a)$ depends on the complexity of the operator $\Lambda(U_a)$, which is not much higher than the complexity of the operator U (cf. Problem 13.2). Thus, $\Xi(U_a)$ and $\Xi(iU_a)$ can be realized by quantum circuits of size $O(n^2)$ in the standard basis.

13.5.2. ***Determining the phase with constant precision.*** We want to localize the value of $\varphi = \varphi_k$, i.e., to infer the inequality $|\varphi - y|_{\bmod 1} < \delta$ for some (initially unknown) y and a given precision δ. To get such an estimate, we apply the operators $\Xi(U_a)$ and $\Xi(iU_a)$ to the same "object of

measurement" but different "instruments" (auxiliary qubits). The reasoning is the same for both operators, so we limit ourselves to the case $\Xi(U_a)$.

We have the quantum register A that contains $|\xi_k\rangle$. Actually, this register initially contains $|1\rangle = \frac{1}{\sqrt{t}} \sum_{k=0}^{t-1} |\xi_k\rangle$, but we consider each $|\xi_k\rangle$ separately. (We can do this because we apply only operators that are measuring with respect to the orthogonal decomposition $\bigoplus_k \mathbb{C}(|\xi_k\rangle)$, so that different eigenvectors do not mix.) Let us introduce a large number s of auxiliary qubits. Each of them will be used in applying the operator $\Xi(U_a)$.

As was proved in Section 12 (see p. 114), the conditional probabilities in such a case multiply. For the operators $\prod_{r=1}^{s} \Xi(U_a)[r, A]$, the conditional probabilities are equal to $\mathbf{P}(v_1, \ldots, v_s|k) = \prod_{r=1}^{s} \mathbf{P}(v_r|k)$ (here v_r denotes the value of the r-th auxiliary qubit).

From this point on, the qubits containing the results of the "experiments" will only be operated upon classically. Since the conditional probabilities multiply, we can assume that we are estimating the probability $p_* = \mathbf{P}(1|k)$ of the outcome 1 ("head" in a coin flip) by performing a series of Bernoulli trials.

If the coin is tossed s times (where s is large), then the observed frequency $(\sum v_r)/s$ of the outcome 1 is close to its probability p_*. What is the accuracy of this estimate? The exact question: with what probability does the number $(\sum v_r)/s$ fail to approximate p_* with a given precision δ? The answer is given by *Chernoff's bound*:

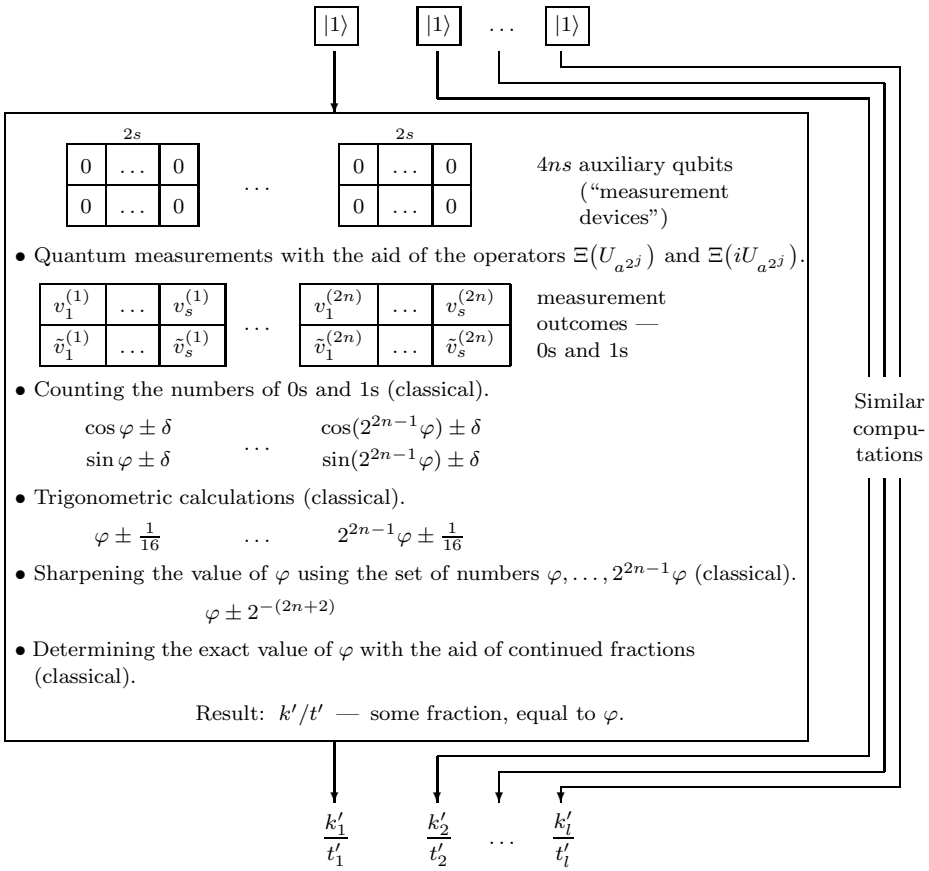
$$\mathbf{Pr}\Big[\Big|s^{-1}\sum_{r=1}^{s} v_r - p_*\Big| \geq \delta\Big] \leq 2e^{-2\delta^2 s}. \tag{13.4}$$

(This inequality is a generalization of the inequality (4.1) which was used to prove the amplification of success probability in the definitions of BPP and BQP.) Thus, for a fixed δ we can find a suitable constant $c = c(\delta)$ such that the error is smaller than ε when $s = \lceil c \log(1/\varepsilon) \rceil = \Theta(\log(1/\varepsilon))$ trials are made.

So, we have learned how to find $\cos(2\pi\varphi)$ and $\sin(2\pi\varphi)$ with any given precision δ. Now we choose δ so that the value φ can be determined from the values of the sine and the cosine with precision $1/16$. This still takes $\Theta(\log(1/\varepsilon))$ trials. The second stage is completed.

[3] **Problem 13.3.** Prove the inequality (13.4).

13.5.3. ***Determining the phase with exponential precision.*** To increase the precision, we will use, along with $\Lambda(U_a)$, the operators $\Lambda\big((U_a)^{2^j}\big)$ for all $j \leq 2n-1$. We can quickly raise numbers to a power, but, in general, computing a power of an operator is difficult. However, the operation U_a of



– Calculation of the least common denominator (classical).

Answer: t (with probability of error $< 3 \cdot 2^{-l} + nle^{-\Omega(s)}$).

Table 13.1. General scheme of the period finding algorithm. Shown in a box is the phase estimation part.

(mod q)-multiplication by a possesses the following remarkable property:

$$(U_a)^p = U_{a^p} = U_{(a^p \bmod q)}.$$

Consequently, $\Lambda\big((U_a)^{2^j}\big) = \Lambda(U_b)$, where $b \equiv a^{2^j} \pmod q$. The required values for the parameter b can be calculated using a circuit of polynomial size; then we can apply the result of Problem 13.2.

Let us return to the circuit described in 13.5.1. We found the eigenvalue $\lambda_k = \lambda = e^{2\pi i\varphi}$ for some eigenvector $|\xi_k\rangle$. This same vector is an eigenvector for any power of the operator U_a, so that in the same quantum register we can look for an eigenvalue of $U_a^2 = U_{a^2}$ (it equals $\lambda^2 = e^{2\pi i\cdot 2\varphi}$), of $U_a^4 = U_{a^4}$ (it equals $\lambda^4 = e^{2\pi i\cdot 4\varphi}$), etc.

In other words, we can determine with precision $1/16$ the values of φ, $2\varphi,\ldots,$ $2^{2n-1}\varphi$ modulo 1. But this allows us to *determine φ with precision $1/2^{2n+2}$ efficiently* (in linear time with constant memory). The idea is based on the following obvious fact: if $|y - 2\varphi|_{\bmod 1} < \delta < 1/2$, then

$$\text{either } |y_0' - \varphi|_{\bmod 1} < \delta/2 \quad \text{or } |y_1' - \varphi|_{\bmod 1} < \delta/2,$$

where y_0', y_1' are the solutions to the equation $2y' \equiv y \pmod 1$. Thus we can start from $2^{2n-1}\varphi$ and increase the precision as we proceed toward φ. The approximate values of $2^{j-1}\varphi$ $(j = 2n, 2n-1, \ldots, 1)$ will allow us to make the correct choices.

Let $m = 2n$. For $j = 1, \ldots, m$ we replace the known approximate value of $2^{j-1}\varphi$ by β_j, the closest number from the set $\{\frac{0}{8}, \frac{1}{8}, \frac{2}{8}, \frac{3}{8}, \frac{4}{8}, \frac{5}{8}, \frac{6}{8}, \frac{7}{8}\}$. This guarantees that

$$|2^{j-1}\varphi - \beta_j|_{\bmod 1} < 1/16 + 1/16 = 1/8.$$

Let us introduce a notation for binary fractions: $\overline{.\alpha_1\cdots\alpha_p} = \sum_{j=1}^{p} 2^{-j}\alpha_j$ $(\alpha_j \in \{0,1\})$. Our algorithm is as follows.

Algorithm for sharpening the value of φ. Set $\overline{.\alpha_m\alpha_{m+1}\alpha_{m+2}} = \beta_m$ and proceed by iteration:

$$\alpha_j = \begin{cases} 0 & \text{if } \left|\overline{.0\alpha_{j+1}\alpha_{j+2}} - \beta_j\right|_{\bmod 1} < 1/4, \\ 1 & \text{if } \left|\overline{.1\alpha_{j+1}\alpha_{j+2}} - \beta_j\right|_{\bmod 1} < 1/4 \end{cases} \qquad \text{for } j = m-1, \ldots, 1$$

(each time, exactly one of the two cases holds). The result satisfies the inequality

$$|\overline{.\alpha_1\alpha_2\cdots\alpha_{m+2}} - \varphi|_{\bmod 1} < 2^{-(m+2)}.$$

The proof is a simple induction: $\left|\overline{.\alpha_j\cdots\alpha_{m+2}} - 2^{j-1}\varphi\right|_{\bmod 1} < 2^{-(m+3-j)}$ at each step.

This procedure is an example of computation by a finite-state automaton (see Problem 2.11). The state of the automaton is the pair $(\alpha_{j+1}, \alpha_{j+2})$, whereas the input symbols are β_j. It follows that the computation can be represented by a Boolean circuit of size $O(m)$ and depth $O(\log m)$.

13.5.4. ***Determining the exact value of the phase.*** We have found a number y satisfying $|y - k/t| < 1/2^{2n+1}$. We represent it as a continued fraction (see Section A.7) and try all convergents of y until we find a fraction k'/t' such that $|y - k'/t'| < 1/2^{2n+1}$. The second part of Theorem A.13 guarantees that the number k/t is contained among the convergents, and

therefore will be found unless the algorithm stops earlier. But it cannot stop earlier because there is at most one fraction with denominator $\le 2^n$ that approximates a given number with precision $1/2^{2n+1}$. The running time of this algorithm is $O(n^3)$.

Important observations. 1. It is essential that the vector $|\xi_k\rangle$ does not deteriorate during the computation.

2. The entire period finding procedure depends on the parameters l and s; they should be adjusted so that the error probability be small enough. The error can occur in determining the period t as the least common denominator (see Lemma 13.2) or in estimating the cosine and the sine of φ_k with constant precision δ (see inequality (13.4)). The total probability of error does not exceed $3 \cdot 2^{-l} + nle^{-\Omega(s)}$. If it is required to get the result with probability of error $\le 1/3$, then we must set $l = 4$, $s = \Theta(\log n)$. In this way we get a quantum circuit of size $O(n^3 \log n)$. (In fact, there is some room for optimization; see Corollary 13.3.1 below.)

13.6. Discussion of the algorithm. We discuss two questions that arise naturally with regard to the algorithm that has been set forth.

— **Which eigenvalues do we find?** We find a randomly chosen eigenvalue. The distribution over the set of all eigenvalues can be controlled by appropriately choosing the initial state. In our period finding algorithm, it was $|1\rangle = \frac{1}{\sqrt{t}} \sum_{k=0}^{t-1} |\xi_k\rangle$, which corresponded to the uniform distribution on the set of eigenvalues associated with the orbit of 1.

— **Is it possible to find eigenvalues of other operators in the same way as in the algorithm for determining the period?** Let us be accurate: by *finding an eigenvalue with precision δ and error probability $\le \varepsilon$* we mean constructing a measuring operator with garbage, as in equation (12.3), where $j \in \Omega$ is an index of an eigenvalue $\lambda_j = e^{2\pi i \varphi_j}$, $\mathcal{L}_j$ is the corresponding eigenspace, and $y = \overline{.\alpha_1 \cdots \alpha_n}$ $(n = \lceil \log_2(1/\delta) \rceil)$. The conditional probabilities (12.4) should satisfy

$$\mathbf{Pr}\Big[|y - \varphi_j|_{\bmod 1} < \delta\Big] = \sum_{y:\, |y-\varphi_j|_{\bmod 1} < \delta} \mathbf{P}(y|j) \ge 1 - \varepsilon. \tag{13.5}$$

The answer to the question is "yes" — it is only necessary to implement $\Lambda(U)$, which is usually easy. (For example, if $U|0\rangle = |0\rangle$, we can use the result of Problem 8.4.) However, in general, the attainable precision is not great and depends polynomially on the number of times the operator $\Lambda(U)$ is used. If one can efficiently compute the powers of U, e.g., if one can implement the operator

$$\Upsilon_m(U) \,:\, |p\rangle \otimes |\xi\rangle \mapsto |p\rangle \otimes U^p|\xi\rangle \quad (0 \le p < 2^m), \tag{13.6}$$

then the precision can be made exponential, $\delta = \exp(-\Omega(m))$.

13.7. Parallelized version of phase estimation. Applications. Remarkably, the phase estimation procedure (except for its last part — the continued fraction algorithm) can be realized by a quantum circuit of small depth. This result is due to R. Cleve and J. Watrous [**18**], but our proof is different from theirs.

Theorem 13.3. *Eigenvalues of a unitary operator U can be determined with precision $\delta = 2^{-n}$ and error probability $\le \varepsilon = 2^{-l}$ by an $O(n(l+\log n))$-size, $O(\log n + \log l)$-depth quantum circuit over the standard basis, with the additional gate $\Upsilon_m(U)$, $m = n + \log(l+\log n) + O(1)$. This gate is used in the circuit only once.*

Proof. At the core of the usual phase estimation procedure is a sequence of operators $\Lambda(U^{2^k})$, $k = 1, \dots, n-1$, applied to the same main register A with distinct control qubits $1, \dots, t$. (Here $t = 2ns$, which corresponds to $2n$ series of Bernoulli trials, each consisting of $s = \Theta(l + \log n)$ coin tosses. Each series is made to determine a single number $\cos(2^k\varphi_j)$ or $\sin(2^k\varphi_j)$ with the suitable constant precision and error probability $2^{-l}/(2n)$.) We need to parallelize these sequences. This can be done as follows: instead of applying the circuit

$$\Lambda(U^{p_t})[t, A] \cdots \Lambda(U^{p_1})[1, A],$$

we compute $p = p(u_1, \dots, u_t) = u_1p_1 + \cdots + u_tp_t$ (where $u_1, \dots, u_t \in \mathbb{B}$ are the values of the control qubits), use p as the control parameter for the operator $\Upsilon_m(U)$, and uncompute p.

To optimize the computation of p, we notice that each p_r is of the form $p_r = 2^{k_r}$. The terms in the sum $\sum_{r=1}^{t} u_r p_r$ can be divided into $2s$ groups in such a way that the numbers k_r be distinct within each group. Therefore, each group corresponds to an n-digit integer, and there are $2s = O(l+\log n)$ of such integers. The sum can be computed by a circuit of size $O(n(l+\log n))$ and depth $O(\log n + \log l)$ (see Problem 2.13a).

Let us estimate the complexity of the remaining part of the procedure. Each gate $\Lambda(U^{2^k})$ is accompanied by two H gates and possibly by one K gate, which contribute $O(t)$ to the size and $O(1)$ to the depth. Further, one needs to count the number of "heads" in each of the $2n$ series of coin flips. This is done by circuits of size $O(s)$ and depth $O(\log s)$. The subsequent trigonometric calculations are performed with constant precision, so each instance of such calculation is done by a circuit of constant size. Finally, sharpening of the value of φ_j is carried out by a circuit of size $O(n)$ and depth $O(\log n)$. All these numbers stay within the required bounds. $\square$

Unfortunately, Theorem 13.3 does not imply that the algorithms for period finding and factoring can be fully parallelized. However, one can derive the following corollary.

Corollary 13.3.1. *Period finding and factoring can be performed by a uniform sequence of $O(n^3)$-size, $O((\log n)^2)$-depth quantum circuits, with some classical pre-processing and post-processing. The pre-processing and post-processing are realized by uniform sequences of $O(n^3)$-size Boolean circuits.*

Note that if we use Definition 9.2, classical pre-processing does not count, since it can be included into the machine that generates the circuit. (However, the post-processing does count.) The division into three stages is clear from Table 13.1. The pre-processing amounts to modular exponentiation, $(q, a, p) \mapsto a^p \bmod q$. No small depth circuit is known for the solution of this problem. Thus we must compute the numbers $a^{2^j} \bmod q$ $(j = 0, \ldots, 2n-1)$ in advance. The post-processing includes finding the exact value of φ (by the continued fraction algorithm) and the calculation of the least common denominator (by Euclid's algorithm).

Proof of Corollary 13.3.1. The operator

$$\Upsilon(U_a) \,:\, \big|p, x\big\rangle \mapsto \big|p,\, (a^p x \bmod q)\big\rangle$$

is realized using the construction from the proof of Theorem 7.3. We need to compute $(a^p x \bmod q)$ and $(a^{-p} x \bmod q)$ by circuits of small depth. With pre-computed values of $(a^{2^j} \bmod q)$ and $(a^{-2^j} \bmod q)$, the computation of $(a^p x \bmod q)$ or $(a^{-p} x \bmod q)$ amounts to multiplying $O(n)$ numbers and calculating the residue mod q, which is done by a circuit of size $O(n^3)$ and depth $O((\log n)^2)$. □

Remark 13.1. R. Cleve and J. Watrous [**18**] also noticed that the depth can be decreased at the cost of increase in size. Indeed, the multiplication of $O(n)$ n-digit numbers can be performed with depth $O(\log n)$ and size $O(n^5(\log n)^2)$ (see [**9**]); therefore the same bound applies to period finding and factoring.

Now we can also prove Theorem 8.3. We will actually consider a more general situation: instead of realizing a single operator, we will try to simulate a circuit (see Theorem 13.5 below). Let us begin with a lemma.

Lemma 13.4. *The operator $\Upsilon_n(e^{2\pi i/2^n}) : |l\rangle \mapsto e^{2\pi i l/2^n}|l\rangle$ $(0 \le l < 2^n)$ can be realized with precision $\delta = 2^{-n}$ by an $O(n^2 \log n)$-size $O((\log n)^2)$-depth circuit C_n over the standard basis, using ancillas. The circuit C_n can be constructed algorithmically in time* $\mathrm{poly}(n)$.

Proof. Let us assume that we have at our disposal an n-qubit register in the state

$$|\psi_{n,k}\rangle = \frac{1}{\sqrt{2^n}} \sum_{j=0}^{2^n-1} \exp\left(-2\pi i \frac{kj}{2^n}\right) |j\rangle, \tag{13.7}$$

where k is odd. We will now see how it helps to achieve our goal of realizing the operator $\Upsilon_n(e^{2\pi i/2^n})$.

The vector $|\psi_{n,k}\rangle$ is an eigenvector of the permutation operator $X : |j\rangle \mapsto |(j+1) \bmod 2^n\rangle$,

$$X|\psi_{n,k}\rangle = e^{2\pi i\varphi_k}|\psi_{n,k}\rangle, \qquad \varphi_k = k/2^n.$$

Application of a power of X to the target state $|\psi_{n,k}\rangle$ results in multiplication by a phase factor,

$$X^p|\psi_{n,k}\rangle = e^{2\pi i(kp/2^n)}|\psi_{n,k}\rangle.$$

If k is odd, we can choose p to satisfy $kp \equiv l \pmod{2^n}$, which will provide the required phase factor $e^{2\pi il/2^n}$. Thus, for the realization of the operator $|l\rangle \mapsto e^{2\pi il/2^n}|l\rangle$ we use the value of l to compute p, apply the operator

$$\Upsilon_n(X) : |p, j\rangle \mapsto \big|p, (j+p) \bmod 2^n\big\rangle, \quad p, j \in \{0, \ldots, 2^n - 1\}, \tag{13.8}$$

controlled by this p, and "uncompute" p. The operator $\Upsilon_n(X)$ can be realized by a circuit of size $O(n)$ and depth $O(\log n)$ over the standard basis.

Ideally, we would want to use the vector $|\psi_{n,k}\rangle$ for some particular k, say, $k = 1$. But constructing such a vector is not easy, so we will start from a superposition of all odd values of k, namely,

$$|\eta\rangle = \frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|2^{n-1}\rangle = \frac{1}{\sqrt{2^{n-1}}} \sum_{s=1}^{2^{n-1}} |\psi_{n,2s-1}\rangle.$$

Then we will measure $k = 2s - 1$ and solve the equation for p. We now describe the required actions.

1. Create the vector $|\eta\rangle = \sigma^z[1]\, H[1]\, |0^n\rangle$.
2. Measure k with error probability $\le \varepsilon = \delta^2/4$. To find k, it suffices to determine the phase $\varphi_k = k/2^n$ with precision $\delta = 2^{-n}$. By Theorem 13.3, such phase estimation is realized by a circuit of size $O(n^2)$ and depth $O(\log n)$. The measured value should be odd, $k = 2s - 1$. (If it has happened to be even, set $k = 1$.)
3. Find $p = p(s, l)$ satisfying the equation $(2s-1)p \equiv l \pmod{2^n}$ (see below).
4. Apply X^p to the n-bit register (which presumably contains $|\psi_{n,2s-1}\rangle$). This will effect the desired phase shift.

5. Reverse the computation done at Steps 1–3.

Apart from Step 1 and its reverse, the above procedure can be described symbolically as $W^{-1}VW$, where W represents Steps 2 and 3, and V represents Step 4. Hence the result of Problem 12.2 applies — the procedure realizes the operator $U = \Upsilon_n(e^{2\pi i/2^n})$ with precision $2\sqrt{\varepsilon} = \delta$.

Step 3 is the most demanding for resources. The solution to the equation $(2s-1)p \equiv l \pmod{2^n}$ can be obtained as follows:

$$p \equiv -l \sum_{j=0}^{m-1} (2s)^j \equiv -l \prod_{r=1}^{t-1} \bigl(1 + (2s)^{2^r}\bigr) \mod 2^n, \quad m = 2^t, \quad t = \lceil \log_2 n \rceil.$$

This calculation is done by a circuit of size $O(n^2 \log n)$ and depth $O((\log n)^2)$ (cf. solution to Problem 2.14a). □

Theorem 13.5. *Any circuit C of size L and depth d over a fixed finite basis $\mathcal{C}$ can be simulated with precision δ by an $O\bigl(Ln + n^2 \log n\bigr)$-size $O\bigl(d \log n + (\log n)^2\bigr)$-depth circuit $\tilde{C}$ over the standard basis (using ancillas), where $n = O(\log(L/\delta))$.*

Proof. Due to the results of Problems 8.1 and 8.2, each gate of the original basis $\mathcal{C}$ can be replaced by a constant size circuit over the basis $\mathcal{Q} \cup \bigl\{\Lambda(e^{i\varphi}) : \varphi \in \mathbb{R}\bigr\}$. Thus the circuit C is transformed into a circuit C' of size $L' = O(L)$ and depth $d' = O(d)$ over the new basis. Each gate $\Lambda(e^{i\varphi})$ can be approximated with precision $\delta' = \delta/(3L')$ by a gate of the form $\Lambda(e^{2\pi i l/2^n})$, where $n = \lceil \log_2(1/\delta') \rceil$, and l is an integer. The operator $\Lambda(e^{2\pi i l/2^n})$ is a special case of $\Upsilon_n(e^{2\pi i/2^n})$, hence we can use Lemma 13.4. However, the resulting circuit is somewhat larger than required, although it will suffice for the proof of Theorem 8.3 (which corresponds to the case $L = d = 1$).

To optimize the above simulation procedure, let us examine the proof of Lemma 13.4. Most of the resource usage can attributed to solving the equation $kp \equiv l \pmod{2^n}$. But this step is redundant if $k = 1$. In fact, the operator $\Upsilon_n(e^{2\pi i/2^n})$ can be realized by applying $\Upsilon_n(X)$ (see (13.8)) to the target state $|\psi_{n,1}\rangle$; this is done by a circuit of size $O(n)$ and depth $O(\log n)$. Thus we need to create L' copies of the state $|\psi_{n,1}\rangle$ and use one copy per gate in the simulation of the circuit C'. The exact sequence of actions is as follows.

1. Create the state $|\psi_{n,0}\rangle = H^{\otimes n}|0^n\rangle$.
2. Turn it into $|\psi_{n,1}\rangle = \Upsilon_n(e^{-2\pi i/2^n})|\psi_{n,0}\rangle$ by the procedure of Lemma 13.4. This is done with precision $\delta' = 2^{-n} \le \delta/3$. The corresponding circuit has size $O(n^2 \log n)$ and depth $O((\log n)^2)$.
3. Make L' copies of the state $|\psi_{n,1}\rangle$ out of one copy (see below).

4. Simulate the circuit C' with precision $\delta/3$, using one copy of $|\psi_{n,1}\rangle$ per gate.
5. Reverse Steps 1–3.

To produce multiple copies of the state $|\psi_{n,1}\rangle$, we can use the equation

$$|\psi_{n,k}\rangle^{\otimes m} = W^{-1}\Big(|\psi_{n,0}\rangle^{\otimes(m-1)} \otimes |\psi_{n,k}\rangle\Big),$$
$$W \,:\, |x_1, \ldots, x_{m-1}, x_m\rangle \mapsto |x_1, \ldots, x_{m-1},\, x_1 + \cdots + x_m\rangle.$$

The operators W and W^{-1} are realized using the construction from the proof of Theorem 7.3. This involves the addition of m n-digit numbers, which is done by a Boolean circuit of size $O(nm)$ and depth $O(\log n + \log m)$ (see Problem 2.13a). In our case $m = L' = O(L)$. The overall size and depth of the resulting quantum circuit are as required. □

[3!] **Problem 13.4.** Let $q \geq 1,\ n = \lceil \log_2 q \rceil,\ \delta = 2^{-l}$. Realize the Fourier transform on the group $\mathbb{Z}_q$ with precision δ by a quantum circuit of size $\mathrm{poly}(n, l)$ and depth $\mathrm{poly}(\log n, \log l)$ over the standard basis. Estimate the size and the depth of the circuit more accurately. (For definition of the quantum Fourier transform, see Problem 9.4c.)

13.8. The hidden subgroup problem for $\mathbb{Z}^k$. The algorithms discovered by Simon and Shor can be generalized to a rather broad class of problems connected with Abelian groups. The most general of these is the hidden subgroup problem for $\mathbb{Z}^k$ [**12**], to which the hidden subgroup problem in an arbitrary finitely generated Abelian group G can be reduced. (Indeed, G can be represented as a quotient group of $\mathbb{Z}^k$ for some k.)

A "hidden subgroup" $D \subseteq \mathbb{Z}^k$ (as defined on page 117) has finite index: the order of the group $E = \mathbb{Z}^k/D$ does not exceed 2^n. Therefore $D \cong \mathbb{Z}^k$. From the computational viewpoint, D is given by a basis $(g_1, \ldots, g_k)$ whose binary representation has length $\mathrm{poly}(k, n)$. Any such basis gives a solution to the problem. (The equivalence of two bases can be verified by a polynomial algorithm.)

The problem of computing the period is a special case of the hidden subgroup problem in $\mathbb{Z}$. Recall that $\mathrm{per}_q(a) = \min\{t \geq 1 : a^t \equiv 1 \pmod q\}$. The function $f : x \mapsto a^x \pmod q$ satisfies condition (13.1), where $D = \{m\ \mathrm{per}_q(a) : m \in \mathbb{Z}\}$. This function is polynomially computable, hence an arbitrary polynomial algorithm for finding a hidden subgroup can be transformed into a polynomial algorithm for calculating the period.

The well-known problem of calculating the discrete logarithm can be reduced to the hidden subgroup problem for $\mathbb{Z}^2$. The smallest positive integer s such that $\zeta^s = a$, where ζ is a generator of the group $(\mathbb{Z}/q\mathbb{Z})^*$, is called the *discrete logarithm* of a number a at base ζ. Consider the function

$f : (x_1, x_2) \mapsto \zeta^{x_1} a^{x_2} \bmod q$. This function also satisfies condition (13.1), where $D = \{(x_1, x_2) \in \mathbb{Z}^2 : \zeta^{x_1} a^{x_2} \equiv 1 \bmod q\}$. If we know a basis of the subgroup $D \subseteq \mathbb{Z}^2$, it is easy to find an element of the form $(s, -1) \in D$. Then $\zeta^s = a$, i.e., s is the discrete logarithm of a at base ζ.

Let us describe a quantum algorithm that solves the hidden subgroup problem for $G = \mathbb{Z}^k$. It is analogous to the algorithm for the case $G = (\mathbb{Z}_2)^k$, but instead of the operator $H^{\otimes k}$ we use the procedure for measuring the eigenvalues. Instead of a basis for the group D we will look for a system of generators of the *character group* $E^* = \mathrm{Hom}(E, U(1))$ (the transition from E^* to D is realized by a polynomial algorithm; see, for example, [**54**, vol. 1]). The character

$$(g_1, \ldots, g_k) \mapsto \exp\Bigl(2\pi i \sum_j \varphi_j g_j\Bigr)$$

is determined by the set $\varphi_1, \ldots, \varphi_k$ of numbers modulo 1. These are rational numbers with denominators not exceeding $|E^*| \le 2^n$.

If we produce $l = n + 3$ uniformly distributed random characters $(\varphi_1^{(1)}, \ldots, \varphi_k^{(1)}), \ldots, (\varphi_1^{(l)}, \ldots, \varphi_k^{(l)})$, then they will generate the entire group E^* with probability $\ge 1 - 1/2^{l-n} = 1 - 1/8$ (see Problem 13.1). It suffices to know each $\varphi_j^{(r)}$ with precision δ and the probability of error $\le \varepsilon$, where

$$\delta \le \frac{1}{2^{2n+1}}, \qquad \varepsilon \le \frac{1}{5kl}. \tag{13.9}$$

The last condition guarantees that the total probability of error does not exceed $1/8 + 1/5 < 1/3$.

Let us choose a sufficiently large number $M = 2^m$ (a concrete estimate can be obtained by analyzing the algorithm). We will work with integers between 0 to $M - 1$.

Let us prepare, in one quantum register of length km, the states

$$|\xi\rangle = M^{-k/2} \sum_{g \in \Delta} |g\rangle, \text{ where } \Delta = \{0, \ldots, M-1\}^k.$$

In another register we put $|0^n\rangle$. We apply the quantum oracle (13.2) and then discard the second register. We obtain the mixed state

$$\rho = \mathrm{Tr}_{[km+1,\ldots,km+n]}\Bigl(U\bigl(|\xi\rangle\langle\xi| \otimes |0^n\rangle\langle 0^n|\bigr)U^\dagger\Bigr) = M^{-k} \sum_{g,h \in \Delta:\, g-h \in D} |g\rangle\langle h|.$$

Now we are going to measure the eigenvalues of the shift (mod M) operators

$$V_j : \bigl(g_1, \ldots, g_j, \ldots, g_k\bigr) \mapsto \bigl(g_1, \ldots, (g_j + 1) \bmod M, \ldots, g_k\bigr)$$

(only the j-th component changes). These operators commute, so that they have a common basis of eigenvectors, and therefore we can determine their eigenvalues simultaneously. The eigenvalues have the form $e^{2\pi i s_j/M}$. The corresponding eigenvectors are

$$|\xi_{s_1,\dots,s_k}\rangle \;=\; M^{-k/2} \sum_{(g_1,\dots,g_k)\in\Delta} \exp\Big(-2\pi i \sum_{j=1}^{k} \frac{g_j s_j}{M}\Big)\, |g_1,\dots,g_k\rangle.$$

The probability that a given set $(s_1,\dots,s_k)$ will be realized equals

$$\begin{aligned}\mathbf{P}(\rho,\, \mathcal{L}_{s_1,\dots,s_k}) &= \langle \xi_{s_1,\dots,s_k}|\, \rho\, |\xi_{s_1,\dots,s_k}\rangle \\ &= M^{-2k} \sum_{g,h\in\mathbb{Z}^k} \chi_D(g-h)\, \chi_\Delta(g)\chi_\Delta(h)\, \exp\Big(2\pi i \sum_{j=1}^{k} \frac{(g_j-h_j)s_j}{M}\Big),\end{aligned}$$

where $\chi_A(\cdot)$ denotes the characteristic function of the set A. The Fourier transform of the product is the convolution of the Fourier transforms of the factors. Therefore,

$$\mathbf{P}(\rho,\, \mathcal{L}_{s_1,\dots,s_k}) = \frac{1}{|E^*|} \sum_{(\varphi_1,\dots,\varphi_k)\in E^*} p_{\varphi_1,\dots,\varphi_k}(s_1,\dots,s_k), \tag{13.10}$$

where

$$p_{\varphi_1,\dots,\varphi_k}(s_1,\dots,s_k) = \prod_{j=1}^{k} \left(\frac{\sin(M\pi(s_j/M-\varphi_j))}{M\sin(\pi(s_j/M-\varphi_j))}\right)^2.$$

For a given element $(\varphi_1,\dots,\varphi_k)\in E^*$ the function $p_{\varphi_1,\dots,\varphi_k}$ is a probability distribution. Therefore equation (13.10) can be modeled by the following process: first, a random uniformly distributed element $(\varphi_1,\dots,\varphi_k)\in E^*$ is generated; second, the parameters $s_1,\dots,s_k$ are set according to the conditional probabilities $p_{\varphi_1,\dots,\varphi_k}(s_1,\dots,s_k)$. The conditional probabilities have the following property:

$$\mathbf{Pr}\Big[|s_j/M-\varphi_j|>\beta\Big] \le \frac{1}{M\beta}$$

for any $\beta>0$. If we estimate the quantities s_j/M with precision β and error probability $\le 1/M\beta$, we obtain the values of $\varphi_1,\dots,\varphi_k$ with precision $\delta=2\beta$ and error probability $\le\varepsilon=2/M\beta$. It remains to choose the numbers M and β so that inequality (13.9) be satisfied.

Complexity of the algorithm. We need $O(n)$ queries to the oracle, each query being of length $O(k(n+\log k))$. The size of the quantum circuit is estimated as $O(kn^3)\,\mathrm{poly}(\log k,\log n)$.