## IV. IMPORTANT QUANTUM ALGORITHMS

### A. Introduction

Up until now two main classes of quantum algorithms can be distinguished:

- Quantum Fourier transform based algorithms. The most prominent member of this class is Shor's [11] algorithm with its exponential speedup of number factoring as compared to classical algorithms.
- Quantum searching algorithms, for example the one by Grover [30, 31] with its quadratic speedup for a "needle in a haystack" search in an unstructured data base.

One of the main sources for the power of quantum information processing is the fact a quantum computer is able to deal with a superposition of states at a time instead of one state after another. This feature is called *quantum parallelism* and probably its simplest application is the Deutsch algorithm which we will discuss before the more difficult Grover and Shor algorithms. The downside of quantum parallelism is the fact that all possible results are superposed too and that we have to filter the desired result from this superposition.
It seems that the quantum algorithms discovered so far are best suited to study global properties of a function or a sequence as a whole, like finding the period of a function, the median of a sequence, etc., and not individual details [7] .

### B. The Deutsch algorithm: Looking at both sides of a coin at a time

Consider a one-bit-to-one-bit function $f(x)$. The quantum way to compute such a function is a unitary operator $\mathbf{U}_f$ acting on a two-qubit state $|x, y\rangle$:

$$\mathbf{U}_f|x, y\rangle = |x, y \oplus f(x)\rangle$$

where $\oplus$ means (as usual) addition modulo 2, or XOR. Of course we have to show that $\mathbf{U}$ is unitary, which we do by brute force, that is, by inspecting all possible cases. There are four possible one-bit-to-one-bit functions which can be encoded as $2 \times 2$ matrices (compare section II):

$$f = \mathbf{1}, \mathbf{X}, \mathbf{P}_+ + \mathbf{S}_+, \mathbf{P}_- + \mathbf{S}_-.$$

The first two functions are *balanced* (both outputs 0 and 1 are possible with equal probabilities), the other two are constant. Now let us write down the matrices for $\mathbf{U}_f$ in the usual computational basis $(|00\rangle, |01\rangle, |10\rangle, |11\rangle)$ for the four possible functions $f$. The resulting $4 \times 4$ matrices are block-diagonal (with $2 \times 2$ blocks) and the blocks are either $\mathbf{X}$ or the $2 \times 2$ unit matrix. All these matrices are thus real symmetric and self-inverse and consequently unitary, *q.e.d.*
For computing $f(x)$ we just initialize $y$ to zero:

$$\mathbf{U}_f|x, 0\rangle = |x, f(x)\rangle.$$

Recalling the Hadamard gate

$$\mathbf{H} = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix},$$

that is,

$$\mathbf{H}|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \quad ; \quad \mathbf{H}|1\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$$

we can perform

$$\mathbf{U}_f\mathbf{H}_x|00\rangle = \frac{1}{\sqrt{2}}\mathbf{U}_f(|00\rangle + |10\rangle)$$

$$= \frac{1}{\sqrt{2}}(|0, f(0)\rangle + |1, f(1)\rangle).$$

(Where $\mathbf{H}_x$ means the Hadamard gate applied to the $x$ qubit.) By applying $\mathbf{U}_f$ just *once* we have thus obtained information about $f$ for *both* possible input values. This is not too impressive, but consider a function with $n$ input qubits, but still one output qubit. Here the application of $n$ Hadamard gates (one for each input qubit) to the state

$$|0\rangle_1|0\rangle_2 \cdots |0\rangle_{n-1}|0\rangle_n|0\rangle_{n+1}$$

generates the state

$$\left(\frac{|0\rangle + |1\rangle}{\sqrt{2}}\right)_1 \left(\frac{|0\rangle + |1\rangle}{\sqrt{2}}\right)_2 \cdots \left(\frac{|0\rangle + |1\rangle}{\sqrt{2}}\right)_n |0\rangle_{n+1},$$

(qubit $n+1$ is the output) which is an equal-weight superposition of *all* $2^n$ possible input states. This extremely efficient operation ($n$ operations generate $2^n$ objects) is called the Hadamard transform or Walsh-Hadamard transform. It is related to the Fourier transform. By a single application of the gate $\mathbf{U}_f$ which evaluates the function $f$ to the above state we get (in a sense) all possible outputs of the function $f$. The difficult point is how to extract the desired information from this superposition; a simple measurement would just yield *one* of the $2^n$ possible combinations $|\vec{x}, f(\vec{x})\rangle$ with equal probability. ($\vec{x}$ is shorthand for the $n$ input qubits.) An important ingredient of quantum information processing is the fact that one may exploit *interference* to extract that information from a superposition. The simplest example for this strategy is the Deutsch algorithm which uses a single function evaluation to determine $f(0) \oplus f(1)$, a "global" property of the one-bit function $f$. This algorithm dates back to [32]; we will present an improved version. A generalization to several input qubits is the Deutsch-Jozsa algorithm ([33], improved in [34] and generalized to mixed (thermal) states in [35]) which we will discuss below. All these algorithms do not have a great practical value as compared to the Shor and Grover algorithms but they are easy to understand and they illustrate some general principles.

Consider the two-qubit system discussed above and the following sequence of operations and states:

$$\mathbf{H}_x\mathbf{U}_f\mathbf{H}_x\mathbf{H}_y|0,1\rangle = \mathbf{H}_x\mathbf{U}_f|\psi_1\rangle = \mathbf{H}_x|\psi_2\rangle = |\psi_3\rangle.$$

Obviously

$$|\psi_1\rangle = \frac{1}{2}\left(|0\rangle + |1\rangle\right)\left(|0\rangle - |1\rangle\right) = \frac{1}{2}\left(|00\rangle + |10\rangle - |01\rangle - |11\rangle\right)$$

and

$$|\psi_2\rangle = \frac{1}{2}(|0, f(0)\rangle + |1, f(1)\rangle - |0, 1 \oplus f(0)\rangle - |1, 1 \oplus f(1)\rangle)$$

$$= \frac{1}{2}\begin{cases} (|0, f(0)\rangle + |1, f(0)\rangle - |0, 1 \oplus f(0)\rangle - |1, 1 \oplus f(0)\rangle) & \text{for } f(0) = f(1) \\ (|0, f(0)\rangle + |1, 1 \oplus f(0)\rangle - |0, 1 \oplus f(0)\rangle - |1, f(0)\rangle) & \text{for } f(0) \neq f(1) = 1 \oplus f(0) \end{cases}$$

$$= \frac{1}{2}\begin{cases} (|0\rangle + |1\rangle)(|f(0)\rangle - |1 \oplus f(0)\rangle) & \text{for } f(0) = f(1) \\ (|0\rangle - |1\rangle)(|f(0)\rangle - |1 \oplus f(0)\rangle) & \text{for } f(0) \neq f(1) \end{cases} = \frac{1}{2}(|0\rangle \pm |1\rangle)(|f(0)\rangle - |1 \oplus f(0)\rangle)$$

with the plus sign for $f(0) = f(1)$ (that is, $f(0) \oplus f(1) = 0$ and the minus sign otherwise. The final application of the Hadamard gate on the $x$ qubit yields

$$|\psi_3\rangle = |f(0) \oplus f(1)\rangle \left(\frac{|f(0)\rangle - |1 \oplus f(0)\rangle}{\sqrt{2}}\right)$$

and a measurement of qubit $x$ tells us if the function $f$ is balanced ($f(0) \oplus f(1) = 1$) or constant. One function evaluation is thus enough to determine whether $f$ is balanced or constant: A pictorial way to describe this is "looking at both sides of a coin at the same time": if the two sides of a coin are equal, it is forged (not too cleverly, however), if not, chances are that it is good.

The Deutsch-Jozsa algorithm performs the same test (balanced or constant) on a $n$-bit function $f(\vec{x})$. If one imagines that $n$ may be large and $f$ may be costly to evaluate, then the advantage of having only one function evaluation (as compared to $\mathcal{O}(2^n)$) is clear. It is, however, important to stress that the function must be *promised* to be either balanced or constant; for a more general function the Deutsch-Jozsa algorithm will give an ambiguous answer.

The algorithm for the $n$-qubit case with input qubit vector $\vec{x}$ and a single output qubit $y$ is completely analogous to the one-qubit case:

$$\mathbf{H}_{\vec{x}}\mathbf{U}_f\mathbf{H}_{\vec{x}}\mathbf{H}_y|\vec{0}, 1\rangle = \mathbf{H}_{\vec{x}}\mathbf{U}_f|\psi_1\rangle = \mathbf{H}_{\vec{x}}|\psi_2\rangle = |\psi_3\rangle.$$

Here

$$\mathbf{H}_{\vec{x}} = \prod_{i=1}^{n} \mathbf{H}_i$$

with $\mathbf{H}_i$ the Hadamard gate acting on qubit $i$. We have

$$|\psi_1\rangle = \sum_{\vec{x}} \frac{|\vec{x}\rangle}{2^n}\left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right).$$

Next we observe

$$\mathbf{U}_f|\vec{x}\rangle(|0\rangle - |1\rangle) = |\vec{x}\rangle(|f(\vec{x})\rangle - |1 \oplus f(\vec{x})\rangle)$$

$$= \begin{cases} |\vec{x}\rangle(|0\rangle - |1\rangle) & \text{for } f(\vec{x}) = 0 \\ |\vec{x}\rangle(|1\rangle - |0\rangle) & \text{for } f(\vec{x}) = 1 \end{cases}$$

and thus

$$|\psi_2\rangle = \sum_{\vec{x}} (-1)^{f(\vec{x})} \frac{|\vec{x}\rangle}{2^n}\left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right).$$

The possible function values are now stored in the signs of the amplitudes in the superposition state. We have to find out what the final Hadamard transform does to this state. To see this we consider a one-qubit example first:

$$\mathbf{H}|x\rangle = \frac{1}{\sqrt{2}}(|0\rangle + (-1)^x|1\rangle) = \frac{1}{\sqrt{2}}\sum_z (-1)^{xz}|z\rangle.$$

This generalizes to the $n$-qubit case:

$$\mathbf{H}_{\vec{x}}|\vec{x}\rangle = \frac{1}{\sqrt{2^n}}\sum_{\vec{z}} (-1)^{\vec{x}\cdot\vec{z}}|\vec{z}\rangle$$

where $\vec{x} \cdot \vec{z} = \sum_i x_i z_i$ is the bitwise scalar product of the two $n$-qubit vectors $\vec{x}$ and $\vec{z}$. The final state of the process is

$$|\psi_3\rangle = \frac{1}{2^n}\sum_{\vec{z}}\sum_{\vec{x}} (-1)^{\vec{x}\cdot\vec{z}+f(\vec{x})}|\vec{z}\rangle\left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right).$$

Now measure the amplitude of the state $|\vec{z}\rangle = |\vec{0}\rangle$, which is

$$2^{-n}\sum_{\vec{x}} (-1)^{f(\vec{x})} = \begin{cases} 1 \text{ for } f \text{ constant} \\ 0 \text{ for } f \text{ balanced} \end{cases},$$

and obviously some intermediate value if $f$ is neither balanced nor constant.
As was already said above, the Deutsch-Jozsa algorithm does not solve a particularly urgent problem. Furthermore, as the function is promised to be either balanced or constant, one may also use a classical algorithm to randomly calculate and inspect a number of function values. If these are not all equal, the function is certainly balanced, and if they are all equal, the function is constant with a very high probability. Nevertheless, the algorithm illustrates how interference, and in a way, the Fourier transform (which is related to the Hadamard transform), are employed in quantum information processing. Another Fourier-based algorithm which is more difficult, and potentially much more interesting is Shor's algorithm for finding prime factors.

### C. The Shor algorithm: It's prime time

Shor's algorithm draws from two main sources. One source is number theory, which we will not treat too deeply, and which shows that factoring can be reduced to finding the period of a certain function. Finding a period is of course related to the physicist's everyday business of Fourier transformation, which is the second source of Shor's algorithm. A quantum computer can very effectively compute the desired number-theoretic function for many input values in parallel, and it can also perform certain aspects of the Fourier transform so efficiently that already the term "quantum Fourier transformation" (QFT) has been coined.
Why is it interesting to find prime factors of large numbers? The scientist's motivation is, because it is a hard problem. It turns out that this is one of the extremely rare cases where the same motivation is shared by scientists, bankers, and the military. The reason is cryptography, the secret transmission of (for example financial or military) data by so-called public key cryptographic schemes. In these schemes a large number (the public key) is used to generate a coded message which is then sent to a recipient. The message can only be decoded using the prime factors of the public key. These prime factors (the private key) are only known to the recipient (bank, chief of staff,...). An extremely low-level example is the number 29083=127·229. With pencil and paper only it will probably take you some time to find the prime factors, whereas the inverse operation (the multiplication) should not take you more than about a minute.

#### 1. A little bit of number theory

Let $N \geq 3$ be an odd integer which we want to factorize, and $a < N$ an integer. Let us assume that the greatest common divisor $\gcd(N, a) = 1$, that is, $N$ and $a$ are coprime (*teilerfremd* in German). (If they are not coprime, $f = \gcd(N, a)$ is already a nontrivial prime factor of $N$ and we restart with $N/f$.) The gcd can be determined by a nice little piece of classical Greek culture, Euclid's algorithm, which is, by modern terms, an efficient algorithm.

#### Euclid's algorithm

- $a$,$b$ two numbers, $a > b$, $c = \gcd(a, b)$
- $\Longrightarrow$ $a$ and $b$ are multiples of $c$.
- $\Longrightarrow$ $a - b, \quad a - 2b$, ... are multiples of $c$
- The remainder $r = a - k \cdot b < b$ is also a multiple of $c$
  (if $r = 0, \quad \gcd(a, b) = b$)
- $\Longrightarrow$ $\gcd(a, b) = \gcd(b, r)$.
- Repeat with $(b, r)$ in place of $(a, b)$, ...
- Last nonzero remainder is $\gcd(a, b)$.

Consider the powers $a^x$ of $a$, modulo $N$ (that is, calculate the remainder of $a^x$ with respect to division by $N$). The smallest positive integer $r$ such that

$$a^r \mod N = 1$$

is called the *order* of $a \mod N$. This means that

$$a^r = k \cdot N + 1$$

for some $k$, and consequently

$$a^{r+1} = k \cdot N \cdot a + a$$

such that

$$a^{r+1} \mod N = a \mod N$$

which shows that $r$ is the *period* of the function

$$F_N(x) = a^x \mod N.$$

(Incidentally, this shows that $r \leq N$ because $F_N(x)$ cannot assume more than $N$ different values before repeating.)
Three cases may arise

1) $r$ is odd
2) $r$ is even and $a^{r/2} \mod N = -1$
3) $r$ is even and $a^{r/2} \mod N \neq -1$.

In case 3) at least one of the numbers $\gcd(N, a^{r/2} \pm 1)$ is a nontrivial factor of $N$. The other cases are irrelevant for factorization.
We now show that case 3) leads to a nontrivial factor of $N$. For ease of notation let us call $a^{r/2} = x$. From $x^2 \mod N = 1$ it follows that $x^2 - 1 = (x+1)(x-1)$

is divided by $N$ and thus $N$ must have a common factor with $x+1$ or $x-1$. That common factor cannot be $N$ itself, since $x \mod N \neq -1$ and thus $x+1$ is not a multiple of $N$; neither can $x-1$ be a multiple of $N$ since if it were, $a^{r/2} \mod N = 1$ and the order would be $r/2$, not $r$. (Remember that the order was defined as the *smallest* number such that...) The common factor we are looking for must then be one of the numbers $\gcd(N, a^{r/2} \pm 1)$, and the gcd can be efficiently computed by Euclid's algorithm.
Next we must make sure that case 3) above has a fair chance to occur if we try some numbers $a$. The following facts give us hope:

- If $N$ is a pure prime power $N = p^s$ ($s \geq 2$) there are fast algorithms to find this out. (See, for example, Exercise 5.17 in [3].)
- If $N$ is an odd composite number $N = p_1^{\alpha_1} \cdots p_m^{\alpha_m}$ and $a$ a randomly chosen integer $1 \leq a \leq N-1$ coprime to $N$, and $a^r = 1 \mod N$ (that is, $r$ is the order of $a \mod N$), then the probability

$$\text{prob}(r \text{ even and } a^{r/2} \neq -1 \mod N) \geq 1 - \frac{1}{2^m} \geq \frac{3}{4}.$$

This means that for each time we calculate the order of $a \mod N$ we have a chance of better than 75% to find a nontrivial prime factor of $N$. Computing the order $m$ times reduces the chance of failure to $4^{-m}$. The chance of finding a prime factor (if one exists!) can thus be brought arbitrarily close to 1, but it is important to note that Shor's is a *probabilistic* algorithm.
The proof of this number-theoretic result cam be found in [3], Appendix 4. It is not difficult, but it involves a few more pieces of classical culture, such as the Chinese Remainder Theorem (German: Hauptsatz über simultane Kongruenzen) which is more than 750 years old. The proof can also be found in Appendix B of the excellent 1996 paper [36] by Ekert and Jozsa.
We are now able to give an algorithm which (with high probability) returns a non-trivial factor of any composite $N$. All steps can be performed effectively on a classical computer, except for the task of computing the order, which is where quantum computing comes in.

1) If $N$ is even, return the factor 2.
2) Determine whether $N = a^b$ for integers $a \geq 1$ and $b \geq 2$, and if so return the factor $a$.
3) Randomly chose $x$ in the range 1 to $N-1$. If $\gcd(x, N) > 1$ then return the factor $\gcd(x, N)$.
4) Use the order-finding subroutine to find the order of $x$ modulo $N$.
5) If $r$ is even and $x^{r/2} \neq -1 \mod N$ then compute $\gcd(x^{r/2} \pm 1, N)$ and test to see if one of these is a non-trivial factor, returning that factor if so. Otherwise, the algorithm fails in which case one must restart at step 3).

In Section VI of [36] the authors discuss the complete application of the algorithm to the simplest possible case, $N = 15$.

### 2. *Computing the order*

Remember that the order $r$ of $a \mod N$ was the period of the function

$$F_N(x) = a^x \mod N.$$

The strategy behind Shor's algorithm is to calculate the function $F_N(x)$ for many values of $x$ in parallel and use Fourier techniques to detect the period in the sequence of function values. To do this for a given $N$ two quantum registers are needed:
a source register with $K$ qubits such that $N^2 \leq Q := 2^K \leq 2N^2$ and
a target register with $N$ or more basis states, that is, at least $\log_2 N$ qubits.
*Step 1* of the algorithm is the initialization

$$|\psi_1\rangle = |0\rangle|0\rangle.$$

*Step 2* is the "Quantum Fourier transformation" of the source register. The quantum Fourier transformation is nothing but the ordinary discrete Fourier transformation of a set of data of length $Q$. The corresponding unitary operator is defined by

$$\mathbf{U}_{F_Q} : |q\rangle \mapsto \frac{1}{\sqrt{Q}} \sum_{q'=0}^{Q-1} \exp\left(2\pi i \frac{q'q}{Q}\right) |q'\rangle.$$

The number $q$ between 0 and $Q-1$ has the binary expansion $q = \sum_{j=0}^{K-1} q_j 2^j$, and $|q\rangle$ is shorthand for $|q_{K-1} \ldots q_1 q_0\rangle$. (We will discuss the QFT and its properties below.) The target register is not modified, so the state after step 2 is

$$|\psi_2\rangle = (\mathbf{U}_{F_Q} \otimes \mathbf{1})|\psi_1\rangle = Q^{-1/2} \sum_{q=0}^{Q-1} |q\rangle|0\rangle;$$

all the Fourier phase factors are equal to unity since all source qubits were zero. Note that this particular output can also be generated by a Hadamard transform of the source register.
*Step 3* is the application of the gate $\mathbf{U}_a$ which implements the modular exponentiation $q \mapsto a^q \mod N$. (We will not discuss in detail how to build this gate.) The result is

$$|\psi_3\rangle = \mathbf{U}_a|\psi_2\rangle = Q^{-1/2} \sum_{q=0}^{Q-1} |q\rangle|a^q \mod N\rangle.$$

Here $Q > N^2$ function values of the function $F_N(q)$ are computed in parallel in one step, and since $r < N$ the period $r$ must show up somewhere in this sequence of function values.
*Step 4*: Apply the quantum Fourier transform again to the source register

$$|\psi_4\rangle = (\mathbf{U}_{\mathbf{F_Q}} \otimes \mathbf{1})|\psi_3\rangle$$

$$= Q^{-1} \sum_{q=0}^{Q-1} \sum_{q'=0}^{Q-1} e^{2\pi i \frac{qq'}{Q}} |q\rangle |a^{q'} \mod N\rangle.$$

*Step 5*: Measure the source qubits in the computational basis. Using arguments from number theory and complex analysis it is possible to show that the probability to find the source register in the state q is a rather peaky-looking function, much like the diffraction pattern of a grating in optics. A nice example for $Q = 256$ and $r = 10$ is Fig. IV.1 (compare Fig. 40 in [7]). From the regularities of that pattern the order $r$ can be deduced with a high probability (not with certainty) if the positions of a sufficiently large number of "diffraction peaks" are taken into account.

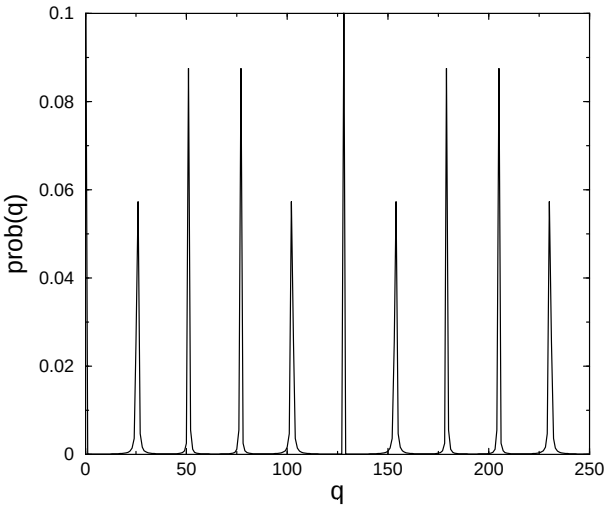


FIG. IV.1: Probability for measuring $q$, with $Q = 256$ and $r = 10$.

What remains to be understood is the implementation of modular exponentiation and of the discrete Fourier transform. We skip all details of the modular exponentiation except for one remark related to the efficient computation of (high) powers $x^a$ of some number $x$ : square $x$, square the result, etc. Get (by squaring $M$ times) the $M + 1$ numbers $x, x^2, x^4, \ldots, x^{2^M}$. Determine the binary expansion of $a$: $a = \sum_{i=0}^{M} a_i 2^i$ (with $a_i = 0, 1$) and multiply the (at most $M + 1$) numbers $x^{2^i}$ for which $a_i = 1$. Thus the large power $a$ is computed using only of the order of $\log_2 a$ multiplications. The only other ingredient needed is an algorithm for multiplying two integers by means of quantum gates, which is available.

### 3. *The quantum Fourier transform*

It is useful to recall the "classical" discrete Fourier transform which maps a complex input vector with components $x_0, x_1, \ldots, x_{N-1}$ to the output vector $y_0, y_1, \ldots, y_{N-1}$:

$$y_k = N^{-1/2} \sum_{j=0}^{N-1} x_j \exp\left(\frac{2\pi i}{N} jk\right).$$

The "quantum Fourier transform" maps the basis states of an $N$-dimensional Hilbert space as follows

$$|j\rangle \mapsto N^{-1/2} \sum_{k=0}^{N-1} \exp\left(\frac{2\pi i}{N} jk\right) |k\rangle$$

such that an arbitrary state is mapped as

$$\sum_{j=0}^{N-1} x_j |j\rangle \mapsto \sum_{k=0}^{N-1} y_k |k\rangle$$

with $y_k$ given by the discrete Fourier transform formula. The unitarity of this transformation is fairly obvious:

$$\sum_{k=0}^{N-1} |y_k|^2 = N^{-1} \sum_{k=0}^{N-1} \left| \sum_{j=0}^{N-1} x_j \exp\left(\frac{2\pi i}{N} jk\right) \right|^2$$

$$= N^{-1} \sum_{k=0}^{N-1} \sum_{j=0}^{N-1} \sum_{l=0}^{N-1} x_j x_l^* e^{\frac{2\pi i}{N}(j-l)k} = \sum_{l=0}^{N-1} |x_l|^2$$

where in the last step we have used the identity

$$\sum_{k=0}^{N-1} \exp\left(\frac{2\pi i}{N}(j-l)k\right) = N\delta_{jl}$$

familiar, for example, from elementary solid state physics. Let us now assume that $N = 2^n$ such that the basis states $\{|0\rangle \ldots |2^n - 1\rangle\}$ form the computational basis for an $n$-qubit quantum computer. We will denote these basis states either by $j$, or by the sequence $j_1 j_2 \ldots j_n$ from the binary representation

$$j = j_1 2^{n-1} + \cdots + j_n 2^0 = \sum_{\nu=0}^{n} j_\nu 2^{n-\nu}.$$

We will also need the notion of a *binary fraction*

$$0.j_l j_{l+1} \ldots j_m = j_l 2^{-1} + j_{l+1} 2^{-2} + \cdots + j_m 2^{-m+l-1}$$

We take another look at the quantum Fourier transform

$$|j\rangle \mapsto 2^{-n/2} \sum_{k=0}^{2^n-1} \exp\left(\frac{2\pi i}{2^n} jk\right) |k\rangle.$$

inserting the binary expansion of $k$ yields

$$|j\rangle \mapsto 2^{-n/2} \sum_{k_1=0}^{1} \cdots \sum_{k_n=0}^{1} e^{\left(\frac{2\pi i}{2^n} j \left(\sum_{l=1}^{n} k_l 2^{n-l}\right)\right)} |k_1 \ldots k_n\rangle$$

$$= 2^{-n/2} \sum_{k_1=0}^{1} \cdots \sum_{k_n=0}^{1} \bigotimes_{l=1}^{n} \exp(2\pi i j k_l 2^{-l}) |k_l\rangle$$

$$= 2^{-n/2} \bigotimes_{l=1}^{n} \left[ \sum_{k_l=0}^{1} \exp(2\pi i j k_l 2^{-l}) |k_l\rangle \right]$$

$$= 2^{-n/2} \bigotimes_{l=1}^{n} \left[ |0\rangle_l + \exp(2\pi i j 2^{-l}) |1\rangle_l \right]$$

where $\bigotimes$ denotes the tensor product and where the sums have been rearranged in the second to last step (in the manner used, for example, also in the standard statistical mechanics calculation of the partition function of a Fermi gas). A closer look at the exponent reveals

$$j2^{-l} = \sum_{\nu=1}^{n} j_\nu 2^{n-\nu-l} = j_1 j_2 \dots j_{n-l} . j_{n-l+1} \dots j_n$$

the integer part (left of the decimal point) of which is irrelevant because $e^{i2\pi k} = 1$ and we can write the quantum Fourier transform as

$$|j\rangle \mapsto 2^{-n/2} \left( |0\rangle_1 + e^{i2\pi 0.j_n} |1\rangle_1 \right) \left( |0\rangle_2 + e^{i2\pi 0.j_{n-1} j_n} |1\rangle_2 \right) \cdots \left( |0\rangle_n + e^{i2\pi 0.j_1 j_2 \dots j_n} |1\rangle_n \right).$$

The quantum Fourier transform is thus nothing but a simple qubit-wise phase shift: the $|1\rangle$ state of each qubit is given an extra phase factor. That operation can be performed by a quantum circuit combining some simple quantum gates. Let us define the unitary (phase shift) operator

$$\mathbf{R}_k = \begin{pmatrix} 1 & 0 \\ 0 & e^{2\pi i 2^{-k}} \end{pmatrix}$$

and the corresponding controlled-$\mathbf{R}_k$ gate which applies $\mathbf{R}_k$ to the target qubit if the control qubit is in state $|1\rangle$. In the corresponding symbol (Fig. IV.2) for the "wiring diagram" of a quantum computer performing the quantum Fourier transform the upper wire denotes the target qubit, the lower wire the control qubit, and data are processed from left to right. The controlled-$\mathbf{R}_k$ gate (for various $k$ values) and the Hadamard gate are sufficient for the quantum Fourier transform circuit shown in Fig. IV.3. To analyze how the circuit of Fig. IV.3 performs the quantum Fourier transform, consider the input state $|j_1 j_2 \dots j_n\rangle$. The Hadamard gate applied to the first qubit generates the state

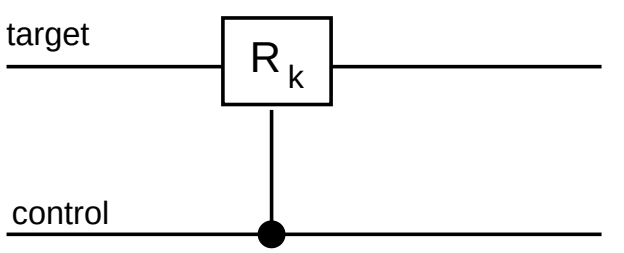


FIG. IV.2: The controlled-$\mathbf{R}_k$ gate.

$$2^{-1/2} \left( |0\rangle + e^{2\pi i 0.j_1} |1\rangle \right) |j_2 \dots j_n\rangle,$$

since $e^{2\pi i 0.j_1} = (-1)^{j_1}$. The controlled-$\mathbf{R}_2$ gate produces

$$2^{-1/2} \left( |0\rangle + e^{2\pi i 0.j_1 j_2} |1\rangle \right) |j_2 \dots j_n\rangle,$$

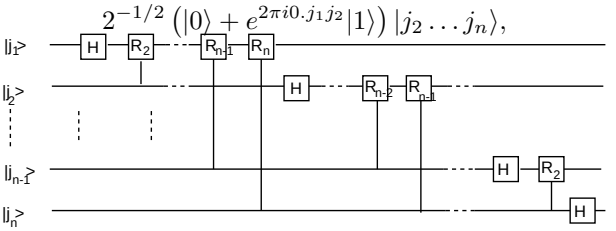


FIG. IV.3: A circuit for the quantum Fourier transform. Not shown are the swap gates necessary to rearrange the output into the desired form. See main text for explanations.

and the following controlled-$\mathbf{R}$ gates keep appending bits to the exponent of the phase factor of $|1\rangle_1$, leading finally to

$$2^{-1/2} \left( |0\rangle + e^{2\pi i 0.j_1 j_2 \dots j_n} |1\rangle \right) |j_2 \dots j_n\rangle.$$

The second qubit is treated in a similar way. The Hadamard gate generates

$$2^{-2/2} \left( |0\rangle + e^{2\pi i 0.j_1 j_2 \dots j_n} |1\rangle \right) \left( |0\rangle + e^{2\pi i 0.j_2} |1\rangle \right) |j_3 \dots j_n\rangle$$

and the controlled-$\mathbf{R}_2$ through $\mathbf{R}_{n-1}$ gates take care of the lower-order bits in the exponent of the phase factor of $|1\rangle_2$, leading to

$$2^{-2/2} \left( |0\rangle + e^{2\pi i 0.j_1 j_2 \dots j_n} |1\rangle \right) \left( |0\rangle + e^{2\pi i 0.j_2 \dots j_n} |1\rangle \right) |j_3 \dots j_n\rangle.$$

Continuing this process we obtain the final state

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

This is almost the desired result, except for the order of the qubits which can be rearranged by SWAP gates. The total number of operations (gates) for the quantum Fourier transform is easily counted. The first qubit is acted on by a Hadamard gate and $n-1$ controlled-**R** gates, a total of $n$ gates. The next qubit needs one controlled-**R** gate less, and so on. The total number of gates shown (implicitly) in Fig. IV.3 thus is $n + (n-1) + \cdots + 1 = n(n+1)/2$. In addition one needs about $n/2$ SWAP gates, each containing three CNOTs. The quantum Fourier transform thus needs of the order of $n^2$ gates (operations) to Fourier transform $2^n$ input data. This is much better than the famous classical fast Fourier transform of Cooley and Tukey (actually of Gauß) which needs $n2^n$ steps, where the "naive" algorithm suggested by the definition of the Fourier transform would take of the order of $2^{2n}$ steps. Note, however, that it is not possible to get out *all* of the amplitudes of the final state of the quantum Fourier transform, nor is it possible to efficiently prepare the input state for arbitrary amplitudes.

### D. NMR Implementation

The Shor algorithm was implemented in an NMR system [37] by a group at IBM Almaden Research Center near San Jose. The smallest integer to which the Shor algorithm can be applied is N=15 (remember: N must be odd and not the power of a prime).

#### 1. *Nuclear Magnetic Resonance*

Nuclear magnetic resonance (NMR) was the first experimental scheme that successfully implemented all required operations for processing of quantum information. Today it remains the only physical systems where quantum algorithms have been implemented.

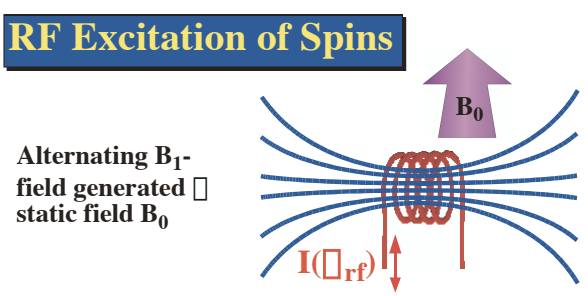


FIG. IV.4: Excitation of nuclear spin qubits.

The scheme identifies the logical states of a qubit with the $|\uparrow\rangle$ and $|\downarrow\rangle$ states of a spin $S = 1/2$. The spins are placed in a static magnetic field, which lifts the degeneracy of the spin states through the Zeeman effect. Unitary operations on the qubits are implemented by radio frequency pulses, which are applied to the spin system through an alternating magnetic field perpendicular to the static field. In addition, free precession periods are used, during which the system evolves under the internal Hamiltonian.

Single qubit operations can be implemented by radio frequency pulses whose frequency is tuned to the resonance frequency of the qubits that are addressed. To permit addressing of individual spins, they must have different resonance frequencies. This differentiation can be achieved either by using different nuclear spin species (in the present example $^{13}C$ and $^{19}F$), and by chemical shift differences between nuclei of the same spin species. Chemical shifts can be understood as an attenuation (called shielding) of the external magnetic field by the surrounding electrons. It can be described phenomenologically by a Hamiltonian

$$H_Z = \sum_i \omega_i I_z^i$$

where the index $i$ runs over all spins. The individual Larmor frequencies $\omega_i$ are proportional to the external field, but individualized through the chemical shift. As a result of these chemical shift differences, NMR spectra of a single type of spin contain distinguishable resonance lines.

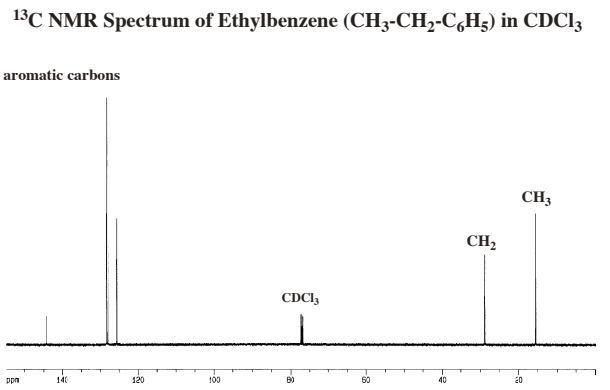


FIG. IV.5: $^{13}C$ NMR spectrum with distinct chemical shifts for 8 different spins.

The separation of these resonances in frequency space allows one to address the spins individually by tuning the RF frequency into the appropriate range.

For 2-qubit operations, couplings between the selected nuclei must be present. In liquid state NMR, such

couplings exist as so-called indirect couplings that are mediated by the electrons. The Hamiltonian for such a coupling has the form

$$H_{ij} = 2\pi\hbar J_{ij} I_z^i I_z^j,$$

with typical coupling constants $J_{ij}$ of the order of 1-100 Hz. Each coupling splits the resonance lines of the coupled spins into 2S+1 lines, which can be labeled with the spin states of the connected spins.

Additional details on NMR implementation of quantum information processing can be found in the lecture notes of the first course, `http://e3.physik.uni-dortmund.de/~suter/Vorlesung/QIV_WS01/6_Implementations.pdf`.

### 2. Qubit Implementation

For the implementation of Shor's factoring algorithm, Vandersypen et al. used a custom-designed molecule with five $^{19}F$ and two $^{13}C$ nuclear spins.

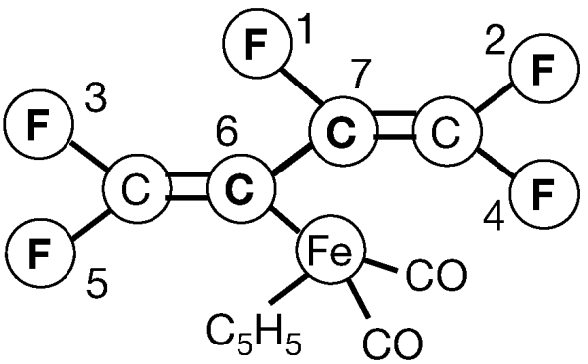


FIG. IV.6: Custom designed molecule with seven nuclear spin qubits.

Two additional carbon nuclei were not used in this experiment. $^{19}F$ and $^{13}C$ are both spins 1/2, have generally long decoherence times and a large chemical shift range that allows fast gating of the qubits.

| $i$ | $\omega_i/2\pi$ | $T_{1,i}$ | $T_{2,i}$ | $J_{7i}$ | $J_{6i}$ | $J_{5i}$ | $J_{4i}$ | $J_{3i}$ | $J_{2i}$ |
|---|---|---|---|---|---|---|---|---|---|
| 1 | −22052.0 | 5.0 | 1.3 | −221.0 | 37.7 | 6.6 | −114.3 | 14.5 | 25.16 |
| 2 | 489.5 | 13.7 | 1.8 | 18.6 | −3.9 | 2.5 | 79.9 | 3.9 | |
| 3 | 25088.3 | 3.0 | 2.5 | 1.0 | −13.5 | 41.6 | 12.9 | | |
| 4 | −4918.7 | 10.0 | 1.7 | 54.1 | −5.7 | 2.1 | | | |
| 5 | 15186.6 | 2.8 | 1.8 | 19.4 | 59.5 | | | | |
| 6 | −4519.1 | 45.4 | 2.0 | 68.9 | | | | | |
| 7 | 4244.3 | 31.6 | 2.0 | | | | | | |

FIG. IV.7: Resonance frequencies of and coupling constants between the qubits. Qubits 1-5 are fluorine nuclei, 6 and 7 are carbon.

The chemical shift separation between the qubits is typically of the order of 1 kHz, thus allowing single qubit gates of the order of 1 millisecond. Each qubit is coupled to every other spin, although some of the coupling constants are relatively small. While the large number of coupling constants allows direct implementation of all 2-qubit gates, it leads to a rather complicated spectrum and implies a rather complicated refocusing scheme: for every single qubit operation, all but one coupling must be refocused by suitable echo pulses.

The coupling implies that the resonance line of every spin is split into two for every other spin. Since every spin is coupled to six other spins, we expect $2^6 = 64$ resonance lines for every spin.

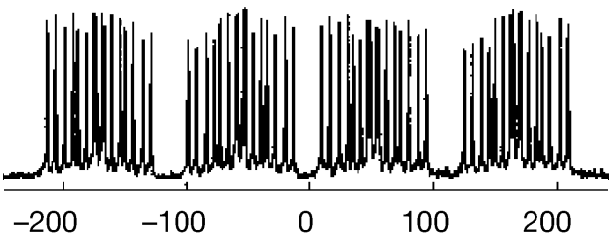


FIG. IV.8: Resonance lines associated with spin 1 for a thermal state.

In the case of spin 1, the total number of distinguishable resonance lines is 64 (although the resolution of the figure, which was taken from [37] is limited). In the spectra of the other spins, several of the resonance lines coincide, thus reducing the total number of lines and making their amplitudes unequal. Each of these resonance lines corresponds to a specific spin state of the other spins. Such a spectrum provides therefore (in principle) a single shot readout of the state of all spins.

### 3. Pseudopure State Preparation

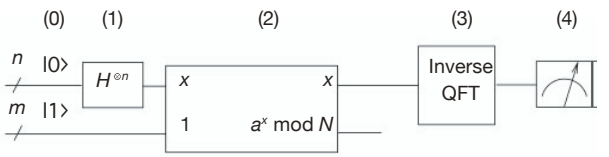


FIG. IV.9: Shor's algorithm.

Shor's algorithm starts with the initial state

$$|\psi_0\rangle = |0000001\rangle$$

The first (most significant) three qubits encode the input register, qubits 4-7 are used as target register for the modular exponentiation.

A spin system used in liquid state NMR, however, is initially in a mixed state determined by the Boltzmann statistics

$$\rho_{th} = \exp(-\frac{H_0}{k_B T})\frac{1}{2^7},$$

determined through the Boltzman statistics and the Zeeman Hamiltonian. Since the thermal energy $k_BT$ ¿¿ $\hbar\omega$ is much larger than the Zeeman energy, the system is in an almost maximally mixed state, which can be expanded as

$$\rho_{th} \approx (1 - \frac{H_0}{k_BT})\frac{1}{2^7}.$$

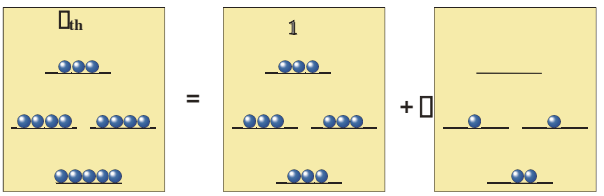


FIG. IV.10: Decomposition of the thermal state into a totally mixed part and a differential part.

The first part corresponds to the completely mixed state, while the second term describes the deviations from complete mixing.
Most quantum algorithms require a pure state as the starting point, but it is not possible to convert a thermal state into a pure state by radio frequency pulses, since these operations are unitary transformations that do not change the entropy of the system.
Rather than working with true pure states, NMR quantum computers use so-called pseudo-pure states, which correspond to the sum of a totally mixed state plus a pure state multiplied with a scaling factor that is of the order of the Boltzmann factor. The task is then to convert the second term, which describes the deviation from complete mixing, into a pseudo-pure state. Different ways have been shown to achieve this goal; Vandersypen et al. [37] chose temporal averaging. The principle of this scheme can be explained for a two-spin system, such as the one shown in the figure.

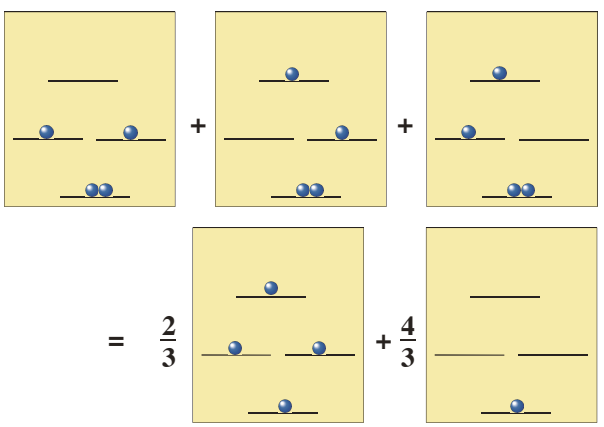


FIG. IV.11: Principle of pseudo-pure state generation.

Starting from the thermal state discussed above, one has to average over three different states (for two qubits), as shown in the figure. The sum of these three states can be decomposed into another contribution to the unity operator plus a pure state (typically the ground state), multiplied by a small coefficient. For the seven qubit system used for the factorization experiment, the pseudo pure state preparation required averaging over 36 different experiments.
Three different ways of averaging have been demonstrated, which are known as temporal averaging, spatial averaging, and logical labeling. Vanderuypen et al. chose the first type, which corresponds to a sequence of experiments, where each experiment starts not with a pure state, but with a mixed state corresponding to one of the states in the figure. The averaging is performed by adding the results of all three (for two qubits) experiments. The experimental result obtained with such a scheme is identical to the result that would be obtained in an experiment with a true pure state.

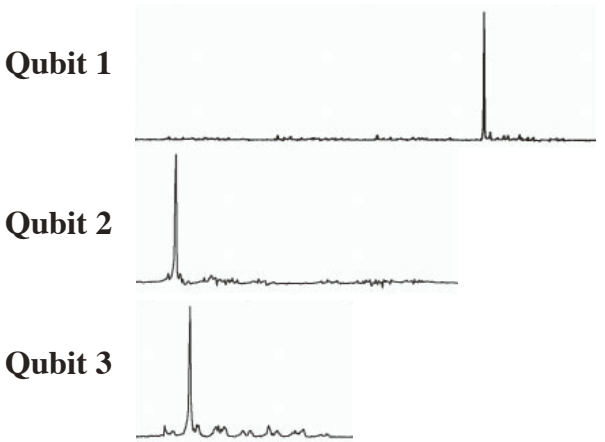


FIG. IV.12: Demonstration of pure state preparation in the spectra of qubits 1-3.

The success of the preparation scheme can be checked easily through a spectrum: If the system is in a pure (or pseudo-pure) state, each spin should have a well defined frequency, i.e. only one of the resonance lines that are generated by spin-spin coupling appears. This is apparently fulfilled to an excellent approximation in the spectra of the first three qubits. While the source register is initiated in the state $|0\rangle$, the target register is initially in state $|1\rangle$. This is achieved by first initiating it into state $|0\rangle$ and subsequently flipping bit 7.

### *4. Hadamard Transform*

The next step is the generation of the superposition of all spin states of qubits 1-3 (the input qubits) through the Hadamard transformation:

$$|\psi_1\rangle = 2^{-n/2}\sum_{q=0}^{2^n-1}|q\rangle.$$

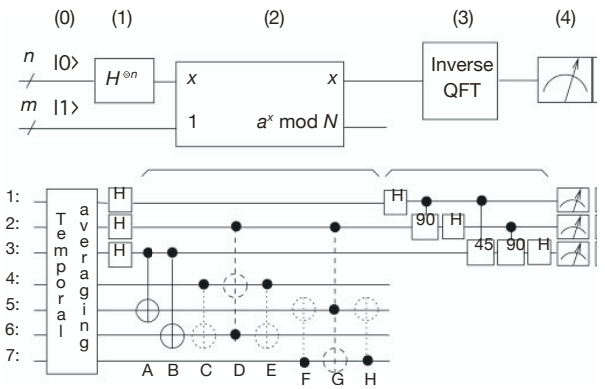


FIG. IV.13: Implementation of Shor's algorithm by gates for N=15 and a=7.

This step can be done either by the Hadamard of QFT transform; for the given input state, the results are identical. The Hadamard gates were implemented by spin-selective 90 degree pulses on the first three qubits, while all couplings between spins had to be removed by additional pulses on all spins.

### 5. *Modular Exponentiation*

One of the crucial steps of Shor's algorithm (as well as of corresponding classical algorithms) is the modular exponentiation $f(q) = a^q \mod N$ for $2^n$ values in parallel. As discussed in subsection C, this is done qubit by qubit with the help of the identity

$$a^q = a^{2^{n-1}q_{n-1}} ... a^{2q_1} a^{q_0},$$

where $q_n$ are the bits of the binary representation of q. While the period of f(q) can be as large as N, only the values 2 and 4 appear for N=15. Since a must be coprime with N, the possible choices of a for N=15 are 2, 4, 7, 8, 11, 13 and 14. For the choices a = 2, 7, 8, and 13, one finds $a^4 \mod 15 = 1$, while $a^2 \mod 15 = 1$ for a = 4, 11 and 14. According to the above expansion, one therefore needs only the two least significant bits of q, i.e. $q_0$ and $q_1$. Vandersypen et al chose to use three bits for encoding q; the additional qubit may be used for test purposes. Together with the with the $m = \log_2 15 = 4$ qubits needed to encode f(x), a total of seven qubits were used.

To implement the exponentiation efficiently, the powers of a were precomputed on a classical computer. The exponentiation is then computed in the target register through CNOT operations.

The first step is a multiplication mod 15 with $a^{q_0}$, i.e. multiplication if qubit 3 is 1, NOP if qubit 3 is 0. Since the target register is now in state $|1\rangle$, multiplication by a can be done by adding (a-1), again controlled by qubit 3. This addition can be implemented by two CNOT operations: for a = 7, with CNOT $_{35}$ CNOT $_{36}$ (see figure), for a = 11 as CNOT $_{34}$ CNOT $_{36}$. For a=7, the second step is multiplication with $7^2 \mod 14 = 4$. This can be done by swapping $y_0$ with $y_2$ and $y_1$ with $y_3$. These swap operations are controlled by qubit $q_1$, i.e. bit 2 in the notation of the figure. The resulting sequence of operations is labeled C-H in the figure.

This step is the most complicated part of Shor's algorithm. Vandersypen et al. used a number of simplifications (="compiler optimizations") to simplify or eliminate specific gates, taking advantage of the special situation. These simplifications are indicated in the figure as dotted gates (can be eliminated) or dashed gates (can be simplified). Gate C can be eliminated because the control qubit is zero, thus reducing the gate to the unity operation. The doubly controlled gates D and G act on target bits that are in basis states (not superposition states), which allows additional simplifications. Gate F can be simplified to a NOT operation, since the control qubit is always 1. Finally, gates E and H can be omitted, since they act on qubits that are no longer accessed afterwards and therefore do not affect the result.

### 6. *QFT*

The implementation of Shor's algorithm requires an (inverse) QFT, in this case on the three most significant qubits. It contains Hadamard gates and phase gates (i.e. z-rotations) of 45 and 90 degrees. In practice, the phase gates are usually turned into rotations of the coordinate axes: rather than apply actual z-pulses (which can be implemented by composite rotations (x)(y)(-x), one simply shifts the phases of all earlier pulses by the corresponding amount.

### 7. *Readout*

One normally assumes that the readout procedure projects the state of the qubit onto either the logical state $|0\rangle$ or $|1\rangle$, independent of the state of the other qubits, as will be discussed in more detail in the following section.

The NMR experiment differs from this idealized quantum mechanical measurement in two respects: It does not yield signals proportional to populations, but instead proportional to coherences. In addition, one does not get a single result for a specific measurement, since the measurement is performed on an ensemble. Instead, the observed amplitudes directly reflect the probabilities of measuring the corresponding state, averaged over the ensemble of some $10^{20}$ molecular quantum computers.

Classically, one observes the magnetisation component that is transverse with respect to the magnetic field and therefore undergoes Larmor precession around the field. The oscillating magnetisation component in the direction of the RF coil changes the magnetic flux through this coil. According to Faraday's law, the change in magnetic flux induces a voltage proportional to this change. One therefore observes an oscillating voltage that decays as the trans-

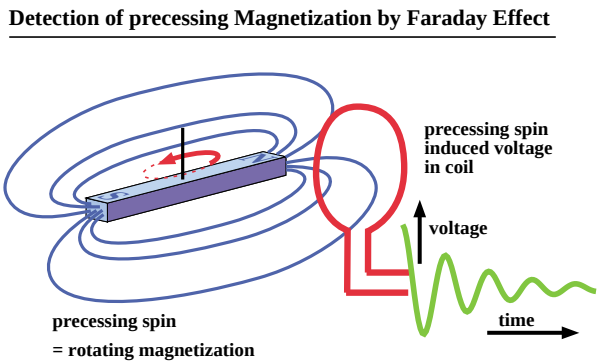


FIG. IV.14: NMR signal from Faraday effect of precessing spins.

verse magentisation slowly vanishes. The decaying signal is referred to as free induction decay or FID. Fourier transformation of this FID yields a signal that corresponds to the absorption and dispersion of the NMR spectrum.

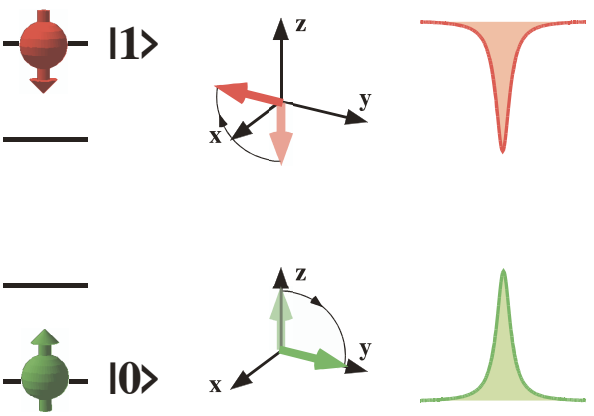


FIG. IV.15: Readout of spin qubits by selective RF pulses: a spin in state $|0\rangle$ gives a positive signal, a spin in state $|1\rangle$ a negative signal.

To apply this measurement to quantum computation, the standard algorithm must be modified slightly. At the end of the standard algorithm, the information is stored in the populations of the spin state. Since they cannot be observed directly, one has to convert them into observable transverse magnetisation with an RF pulse. For a spin that is initially in the ground state $|0\rangle$, i.e. aligned along the field, a suitable RF pulse creates magnetization along the x-direction. If the spin is initially in state $|1\rangle$, the same rf pulse creates -x magnetization; the resulting spectrum will therefore show an emissive behavior.
The three spectra shown here display the resulting state of the three qubits for an input of a = 11. The resulting spectra of the three qubits, which are shown in the figure, contain only positive lines for qubits 1 and 2, indicating that they are in state $|0\rangle$. Qubit 3 has one positive and one negative line, indicating that it is in a superposition state $|0\rangle \pm |1\rangle$.

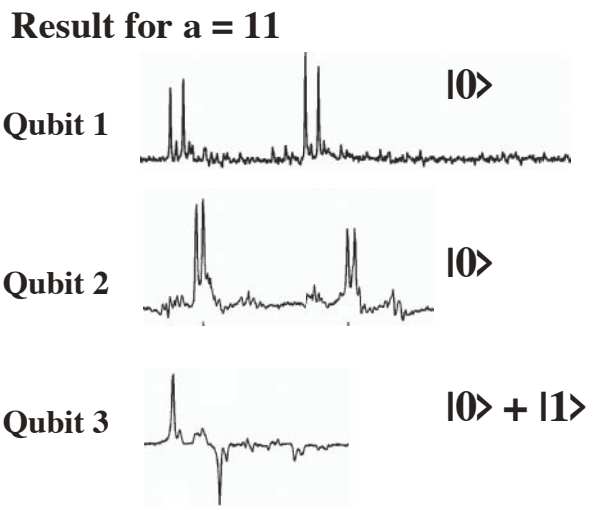


FIG. IV.16: Spectra of the three result qubits for the input a = 11.

After the inverse QFT, qubit 3 is the most significant bit. The resulting state is therefore a mixture of $|100\rangle = |4\rangle$ and $|000\rangle = |0\rangle$. This indicates that the periodicity is $n = 4$ and $r = 2^n/4 = 2$. A classical calculation yields the g.c.d. of $11^{2/2} \pm 1$ and 15 as 3 and 5, and thus directly the prime factors of N.

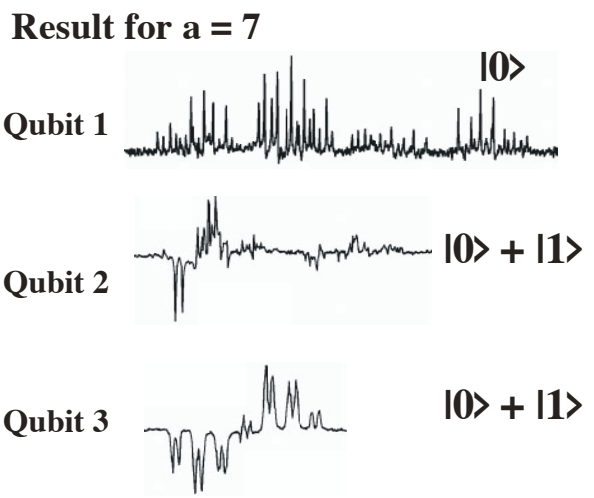


FIG. IV.17: Spectra of the three result qubits for the input a = 7.

If the input a = 7 is used instead, both qubits 2 and 3 are in superposition states. The possible results are therefore the states $|000\rangle = |0\rangle$, $|010\rangle = |2\rangle$, $|100\rangle = |4\rangle$, and $|110\rangle = |6\rangle$, indicating a period of 2. We conclude that $r = 8/2 = 4$ and $g.c.d(7^{4/2} \pm 1, 15) = 3, 5$ as before. Obviously both inputs produce the expected result.

### 8. Decoherence

The experimental implementation of Shor's algorithm represents a milestone for quantum information processing, not because of the result itself, but because it provides the possibility to study limitations to quantum information processing on a working example.

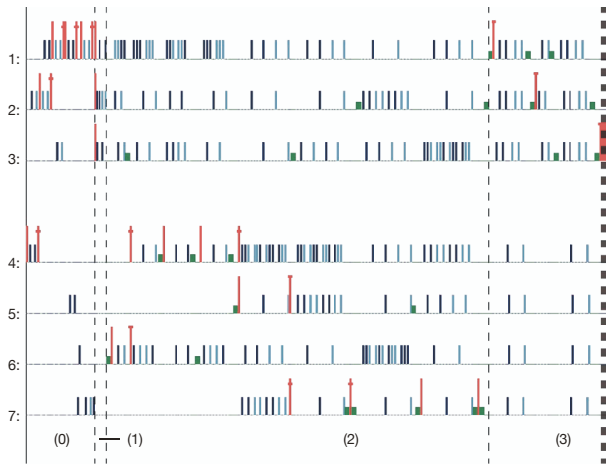


FIG. IV.18: Pulse sequence used for the implementation.

The IBM group used some 300 radio frequency pulses to implement the algorithm. Most of the pulses were used not for the processing itself, but to compensate for unwanted effects, such as spin-spin couplings and magnetic field inhomogeneity. The overall sequence lasted almost 1 second, which is longer than some of the relevant relaxation times (=decoherence times). This caused a significant loss of information and therefore deviations of the experimental measurements from the idealized behavior.

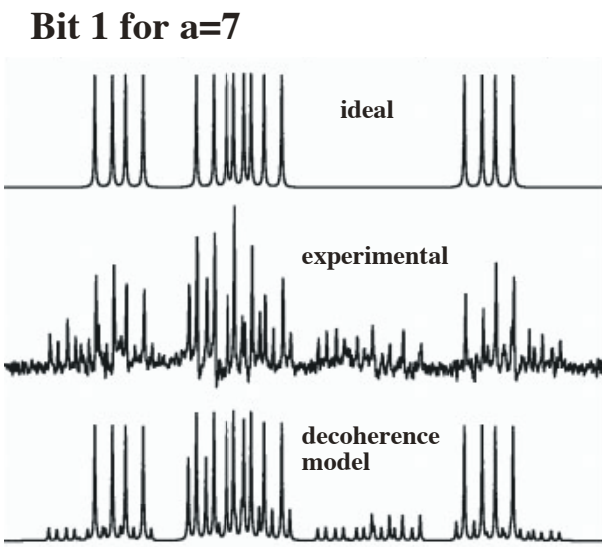


FIG. IV.19: Ideally expected result for a=7 (top), experimental result (middle) and result expected from a model that includes effects of decoherence.

Vandersypen et al. analyzed these deviations with a model for the relevant decoherence processes and found that they could explain most of the differences with their model.

## E. The Grover algorithm: Looking for a needle in a haystack

The Grover algorithm [30, 31] is useful for a search in an unstructured database. This is a very important problem in data processing because every data base is an unstructured one if the problem does not fit to the original design structure of the data base. Just think of trying to find out the name of a person living at a given street address from the usual alphabetic phone directory of a big city. If the phone directory contains $N$ entries this will require checking $N/2$ entries on average (provided there is only one person who lives at the particular address). Grover's algorithm reduces the number of calls to $\mathcal{O}(\sqrt{N})$, which is a significant reduction for large $N$.

In this section we will not deal with the practical implementation of Grover's algorithm, that is, how to couple an existing classical data base to this quantum algorithm etc. We will only outline how this beautiful algorithm allows the solution to "grow" out of the noise by interating a simple procedure. As with all growing things, however, it is important to do the harvesting at the right time. It turns out that the same procedures can be used to grow the solution and to determine the time for harvest.

### 1. The search algorithm

Let the search space of our problem have $N$ elements (entries in the phone directory, in the introductory example), indexed 0 to $N-1$, and for simplicity, $N = 2^n$. Let the search problem have $M$ solutions (persons living at the given street address). The solutions can be characterized by some function $f$ with the property

$$f(x) = \begin{cases} 1 & \text{if } x \text{ is a solution} \\ 0 & \text{if } x \text{ is not a solution.} \end{cases}$$

We are able, by some kind of "detector" to recognize a solution if we are confronted with the $x$th element of the data base. In our example this is simple: we just check the item "street address" in the telephone directory entry number $x$ and output a 1 if it fits and a zero otherwise. In other examples this step may be much more complicated and it is the aim of Grover's algorithms to minimize the number of calls to this "detector" function, or *oracle* function as it is commonly called.

Our quantum search algorithm in fact needs a quantum oracle function, that is, a unitary operator $\mathbf{O}$ acting on the combination (tensor product) of the quantum register holding the index $x$ and a single oracle qubit $|q\rangle$ in the following way:

$$\mathbf{O}|x\rangle|q\rangle = |x\rangle|q \oplus f(x)\rangle,$$

that is, the oracle qubit is flipped when the data base item with the number $x$ is a solution of the search problem. If we initialize the oracle qubit in the state

$$|q_0\rangle := \frac{|0\rangle - |1\rangle}{\sqrt{2}},$$

application of the quantum oracle will lead to

$$\mathbf{O}|x\rangle|q_0\rangle = (-1)^{f(x)}|x\rangle|q_0\rangle.$$

Note that the oracle qubit is not changed, and in fact remains in its initial state during the whole calculation. We will henceforth omit it from out calculations (but will not forget that it is needed). So from now on we will abbreviate the above equation in the following way:

$$\mathbf{O}|x\rangle = (-1)^{f(x)}|x\rangle$$

The oracle *marks* the solutions of the search problem by a minus sign. We will see that only $\mathcal{O}(\sqrt{\frac{N}{M}})$ calls to the quantum oracle will be necessary to solve the search problem. We wish to stress again that the oracle does not by some magic *know* the solution, it is only able *recognize* if a candidate is a solution. Think of the prime factoring problem to note the difference: it is easy to *check* if a proposed candidate divides a number. An appropriate circuit performing test divisions may be used as an oracle in this case.

The key point of the search algorithm will be to use the phase factors (minus signs) marking the solutions to let the amplitudes of the solution states grow out of the set of all possible states, and to "harvest" them at the right time, as said above. We will now first list the steps of the search algorithm and then analyze what these steps do.

*Step 1*: Initialize the $n$-qubit index register

$$|\psi_1\rangle = |0\rangle$$

(All $n$ qubits are set to their $|0\rangle$ states.)

*Step 2*: Apply the Hadamard transform

$$|\psi_2\rangle = \mathbf{H}^{\otimes n}|0\rangle = N^{-1/2}\sum_{x=0}^{N-1}|x\rangle \quad (N = 2^n)$$

to generate an equal-weight, equal-phase superposition.

*Steps 3 and following*: Iterate with the Grover operator $\mathbf{G}$

$$|\psi_{k+1}\rangle = \mathbf{G}|\psi_k\rangle$$

where the Grover operator consists of four substeps:

*Substep 1*: Apply the oracle

$$|\psi_{k+1/4}\rangle = \mathbf{O}|\psi_k\rangle$$

(we use fractional indices to symbolize that these are just substeps of the Grover iteration step).

*Substep 2*: Apply the Hadamard transform

$$|\psi_{k+1/2}\rangle = \mathbf{H}^{\otimes n}|\psi_{k+1/4}\rangle.$$

*Substep 3*: Apply a conditional $\pi$ phase shift, that is, reverse the signs of all computational basis states except $|0\rangle$:

$$\mathbf{C}_\pi|x\rangle = (-1)^{\delta_{x0}-1}|x\rangle$$

$$|\psi_{k+3/4}\rangle = \mathbf{C}_\pi|\psi_{k+1/2}\rangle.$$

*Substep 4*: Apply the Hadamard transform again

$$|\psi_{k+1}\rangle = \mathbf{H}^{\otimes n}|\psi_{k+3/4}\rangle.$$

Substeps 2,3, and 4 can be efficiently implemented on a quantum computer: remember that $\mathbf{H}^{\otimes n}$ creates $2^n$ states (in a superposition) with just $n$ operations; conditional phase shifts are also easy to construct from a complete set of quantum gates. The oracle *may* be computationally expensive, but we use it only *once* per iteration step.

Let us analyze what the Grover iteration step does, other than calling the oracle. The conditional phase shift may be written as

$$\mathbf{C}_\pi = -\mathbf{1} + 2|0\rangle\langle 0|$$

where $\mathbf{1}$ is the $n$-qubit unit operator and $|0\rangle\langle 0|$ is the projection operator on the basis state $|0\rangle$. We know already that

$$\mathbf{H}^{\otimes n}|0\rangle = |\psi_2\rangle( \text{ and } \langle\psi_2| = \langle 0|\mathbf{H}^{\otimes n})$$

where $|\psi_2\rangle$ is the equal-weight (and equal-phase) superposition. The Grover operator thus can be written as

$$\mathbf{G} = (2|\psi_2\rangle\langle\psi_2| - \mathbf{1})\,\mathbf{O}.$$

This operation has a nice algebraic interpretation; it turns out that the amplitudes of the computational basis states are "inverted about their average" (or mean) as is often said. However, we will not employ this algebraic interpretation (which is explained in Chapter 6 of Nielsen and Chuang's book [3]), because it turns out that there is an even nicer geometrical interpretation. The Grover iteration is a rotation in the 2d space spanned by the starting vector $|\psi_2\rangle$ (the uniform superposition of *all* basis states) and the uniform superposition of the states corresponding to the $M$ solutions of the search problem, and we will see that the rotation moves the state into the right direction.

To see this we need a bit of new notation. Let $\sum_x'$ denote the sum over all $M$ solutions of the search problem and $\sum_x''$ the sum over all *other* states of the computational basis. We can then define the following normalized states:

$$|\alpha\rangle = \frac{1}{\sqrt{N-M}}\sum_x''|x\rangle$$

(the unwanted state) and

$$|\beta\rangle = \frac{1}{\sqrt{M}}\sum_x'|x\rangle$$

(the desired state). We can then write the state $|\psi_2\rangle$ in the search algorithm as a superposition of $|\alpha\rangle$ and $|\beta\rangle$:

$$|\psi_2\rangle = \sqrt{\frac{N-M}{N}}|\alpha\rangle + \sqrt{\frac{M}{N}}|\beta\rangle = \cos\frac{\theta}{2}|\alpha\rangle + \sin\frac{\theta}{2}|\beta\rangle$$

which defines the angle $\theta$. Now recall that the oracle marks solutions of the search problem with a minus sign such that

$$\mathbf{O}|\psi_2\rangle = \cos\frac{\theta}{2}|\alpha\rangle - \sin\frac{\theta}{2}|\beta\rangle.$$

The $|\beta\rangle$ component of the initial state thus gets reversed, whereas the $|\alpha\rangle$ component remains the same. In the $|\alpha\rangle, |\beta\rangle$ plane this is a *reflection* about the $|\alpha\rangle$ axis. (See Fig. IV.20.) The remaining three substeps of $\mathbf{G}$ in fact perform another reflection: Note that

$$2|\psi_2\rangle\langle\psi_2| - \mathbf{1} = |\psi_2\rangle\langle\psi_2| - (\mathbf{1} - |\psi_2\rangle\langle\psi_2|) = \mathbf{P}_2 - \mathbf{P}_2^\perp$$

where $\mathbf{P}_2$ is the projector on the initial state $|\psi_2\rangle$ and $\mathbf{P}_2^\perp$ is the projector on the subspace of all Hilbert space vectors perpendicular to $|\psi_2\rangle$. The component perpendicular to $|\psi_2\rangle$ thus gets reversed so that we have performed a *reflection about* $|\psi_2\rangle$. A look at the figure tells us that we have reached the state

$$\mathbf{G}|\psi_2\rangle = \cos\frac{3\theta}{2}|\alpha\rangle + \sin\frac{3\theta}{2}|\beta\rangle,$$

that is, $\mathbf{G}$ has performed a $\theta$ rotation. Iteration then yields

$$\mathbf{G}|\psi_2\rangle = \cos\frac{2k+1}{2}\theta|\alpha\rangle + \sin\frac{2k+1}{2}\theta|\beta\rangle,$$

and we only have to choose $k$ such that the $|\beta\rangle$ component is as large as possible. Measurement in the computational basis will then with high probability produce one of the components of $|\beta\rangle$, the solutions of the search problem. For a detailed description of the search algorithm in a space with four states (not too big), see [3] or the popular article [38] by Grover.

How often do we have to repeat the Grover algorithm? From figure IV.20 and the definition of the angle $\theta$ we see that the necessary number of iterations is the closest integer (abbreviated CI) to $\frac{\pi-\theta}{2\theta}$,

$$R := \mathrm{CI}\left(\frac{\pi}{2\theta} - \frac{1}{2}\right)$$

$$= \mathrm{CI}\left(\frac{\pi}{4\arcsin\sqrt{\frac{M}{N}}} - \frac{1}{2}\right) \leq \frac{\pi}{4}\sqrt{\frac{N}{M}}$$

since $\arcsin x > x$. This moves the state quite close to the desired one: as each Grover iteration rotates the state by $\theta$ we end up at most $\theta/2$ away from $|\beta\rangle$. For the interesting case $\frac{M}{N} \ll 1$ the error probability (square of the $|\alpha\rangle$ component in the final state) is

$$p \leq \sin^2\frac{\theta}{2} = \frac{M}{N}.$$

It is important to note that

- iterating more than $R$ times worsens the result
- in this version of the algorithm it is necessary to know $M$, the number of solutions.

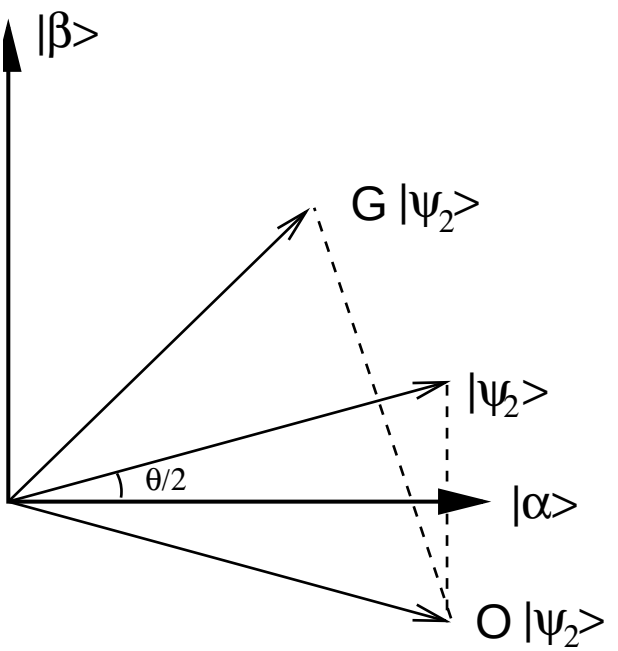


FIG. IV.20: The Grover iteration as a twofold reflection, or a rotation (see text for details).

### 2. *Quantum counting*

Here we discuss how the number $M$ of solutions to the search problem can be counted by a quantum algorithm involving the Grover operator $\mathbf{G}$ again. The idea is simple: recall that in a suitable two-dimensional subspace $\mathbf{G}$ was just a rotation and the angle of rotation is related to $M$. This angle of rotation can be determined by quantum Fourier transform techniques.

The rotation matrix for $\mathbf{G}$ in the basis $(|\alpha\rangle, |\beta\rangle)$ is

$$\mathbf{G} = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}.$$

The eigenvectors of this matrix are $\frac{1}{\sqrt{2}}\begin{pmatrix} 1 \\ \pm i \end{pmatrix}$ with eigenvalues $e^{\pm i\theta}$. Recall that $\sin\frac{\theta}{2} = \sqrt{\frac{M}{N}}$. (Some problems may arise if $M > N/2$, because then $\theta > \pi/2$; however, these problems may always be circumvented by enlarging the search space from $N$ to $2N$ by adding some fictitious directions to the Hilbert space, as discussed in [3]. We will ignore these problems altogether for simplicity.) The problem of (approximately) counting the number $M$ of solutions is thus reduced to estimating the phase $\theta$ of the unitary operator $\mathbf{G}$, the Grover gate. This task of *phase estimation* is worth a section of its own because it is employed, for example, in Shor's algorithm too.

### 3. Phase estimation

For a given unitary operator $\mathbf{U}$ we are given an eigenvector $|u\rangle$:

$$\mathbf{U}|u\rangle = e^{2\pi i\phi}|u\rangle$$

where $\phi$ (between 0 and 1) is to be estimated. Let us assume we have available "black boxes" to
• prepare $|u\rangle$
• perform controlled-$\mathbf{U}^{(2^j)}$ operations ($j = 0, 1, ..$).
The phase estimation algorithm needs two registers. The first register contains $t$ qubits, initially all in the state $|0\rangle$ ($t$ depending on demanded *accuracy* and *success probability* of the algorithm). The second qubit holds the state $|u\rangle$ initially. The algorithm works as follows.
*Step 1*: Apply $\mathbf{H}^{\otimes t}$ to the first register, to generate the state

$$\frac{1}{\sqrt{2^t}}\sum_{x=1}^{2^t}|x\rangle$$

which is the by now well-known equal-weight, equal-phase superposition.
*Step 2.k* ($k = 0, ..., t-1$): Apply the controlled-$\mathbf{U}^{(2^j)}$ operation to register 2, using qubit $k$ of the first register as control qubit. This puts register 2 in state

$$|u\rangle \text{ if qubit } k \text{ is } |0\rangle$$

and in state

$$e^{2\pi i 2^k\phi}|u\rangle \text{ if qubit } k \text{ is } |1\rangle.$$

Note that register 2 always stays in the state $|u\rangle$, up to phase factors which we can collect next to the qubits of register 1 which control them. The state of the first register thus can be written

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

$$\left(|0\rangle + e^{2\pi i 2^0\phi}|1\rangle\right) = \frac{1}{2^{t/2}}\sum_{k=0}^{2^{t-1}} e^{2\pi i\phi k}|k\rangle.$$

(Remember that we have omitted the second register which is in state $|u\rangle$ anyway.
For ease of discussion, assume that $\phi$ is a $t$-bit binary fraction, $\phi = 0, \phi_1\phi_2\ldots\phi_t$ (remember $\phi \leq 1$). The state of register 1 is just

$$\frac{1}{2^{t/2}}\left(|0\rangle + e^{2\pi i 0.\phi_t}|1\rangle\right)\left(|0\rangle + e^{2\pi i 0.\phi_{t-1}\phi_t}|1\rangle\right)\cdots$$

$$\left(|0\rangle + e^{2\pi i 0.\phi_1\phi_2\ldots\phi_t}|1\rangle\right)$$

since $e^{2\pi i m} = 1$ for integer $m$.
We now recall the discussion of the quantum Fourier transform from Shor's algorithm. There we constructed a quantum circuit performing the quantum Fourier transformation

$$|j_1\ldots j_n\rangle \longrightarrow \frac{1}{2^{n/2}}\left(|0\rangle + e^{2\pi i 0.j_n}|1\rangle\right)$$

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

The *inverse* quantum Fourier transform can be performed by simply reversing the QFT circuit. Applying the inverse QFT to the state of our register 1 leads to the state

$$|\phi_1\ldots\phi_t\rangle$$

and therefore we can measure $\phi$ exactly (in this example, where $\phi$ has exactly $t$ bits). The more general case is discussed in Chapter 5.2.1 of [3]. It turns out that using $t$ qubits one can measure $\phi$ accurate to $t - \text{int}(\log_2(2 + \frac{1}{2\epsilon}) - 1$ with probability of success at least $1 - \epsilon$. (int denotes the integer part.)
An important point which remains to be clarified in general is the preparation of the eigenstate $|u\rangle$. In the worst case we are not able to prepare a specific eigenstate, but only some state $|\psi\rangle$ which can then be expanded in $\mathbf{U}$-eigenstates,

$$|\psi\rangle = \sum_u c_u|u\rangle, \text{ where } \mathbf{U}|u\rangle = e^{2\pi i\phi_u}|u\rangle.$$

Running the phase estimation algorithm with input $|\psi\rangle$ in the second register leads (due to linearity) to the output

$$\sum_u c_u|\tilde{\phi}_u\rangle|u\rangle$$

where $\tilde{\phi}_u$ is an approximation to the phase $\phi_u$. We thus obtain the possible phase values of $\mathbf{U}$ with their respective probabilities $|c_u|^2$ as given by the initial state.
In the special case of the Grover algorithm it turns out that we are lucky. Recall that the starting vector of the Grover algorithm was a combination of $|\alpha\rangle$ and $|\beta\rangle$, or equivalently, of the two eigenstates of the unitary operator $\mathbf{G}$ (the Grover operator) so that the phase estimation algorithm will give us approximations to either $\theta$ or $(2\pi) - \theta$ with both of which we will be content, because knowing $\theta$ will enable us to iterate $\mathbf{G}$ so often that we will find a solution to the search problem with high probability.
We will not discuss how to *really* search an unstructured data base etc, and we will also not go into the detailed perfomance and probability estimates. Some remarks on these topics may be found in Chapter 6 of [3], and some generalizations and references to interesting applications are in [7].
Instead we turn (very briefly) to an interesting paper [39] from the 1 April 2002 issue of the *Physical Review Letters* which nevertheless is completely serious.

### 4. Implementation of Grover's algorithm by classical optics

In this implementation it was demonstrated that Grover's algorithm can also be run on purely classical hardware. The implementation employs a laser beam of roughly 1 mm$^2$ size. The beam is divided into

"cells" or "patches" which correspond to the qubits. The information is stored in the electric field of the light wave front on the patch. (Note that the electric field has both amplitude and phase.) A 300 ps ( 10 cm long) laser pulse is reflected back and forth in a 2 m long cavity. One cycle of reflections corresponds to one iteration of the Grover algorithm. The necessary Fourier transforms and othe operations are performed by optical elements (lenses and delay plates).
The important lesson to be learned from this experiment is that Grover's algorithm does not absolutely need quantum properties such as entanglement, but only precise control of wave properties. In the experiment a $N = 32$ database was searched and it was etsimated that $N \sim 10^6$ might be possible by the same technique in the future. For larger $N$ entanglement, and thus a quantum system would be needed.

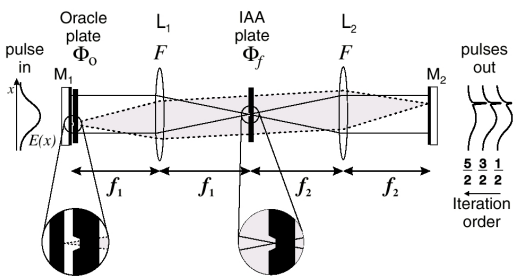


FIG. IV.21: Classical implementation of Grover's algorithm.