Introduction to Quantum Information Processing

E. Knill, R. Laflamme, H. Barnum, D. Dalvit, J. Dziarmaga,
J. Gubernatis, L. Gurvits, G. Ortiz, L. Viola and W. H. Zurek

July 31, 2002

Contents

QUANTUM INFORMATION PROCESSING, SCIENCE OF - The theoretical, experimental and technological areas covering the use of quantum mechanics for communication and computation.

Kluwer Encyclopedia of Mathematics, Supplement III

Research of the last few decades has established that quantum information, or information based on quantum mechanics, has capabilities that exceed those of traditional "classical" information. For example, in communication, quantum information enables quantum cryptography [1, 2], which is a method for communicating in secret. Secrecy is guaranteed because eavesdropping attempts necessarily disturb the exchanged quantum information without revealing the content of the communication. In computation, quantum information enables efficient simulation of quantum physics [3], a task for which general purpose, efficient, classical algorithms are not known to exist. Quantum information also leads to efficient algorithms for factoring of large numbers [4, 5], which is believed to be difficult for classical computers. An efficient factoring algorithm would break the security of commonly used public key cryptographic codes used for authenticating and securing internet communications. A fourth application of quantum information improves the efficiency with which unstructured search problems can be solved [6]. Quantum unstructured search may make it possible to solve significantly larger instances of optimization problems such as the scheduling and traveling salesman problems.

As a result of the capabilities of quantum information, the science of quantum information processing is now a prospering, interdisciplinary field focused on better understanding the possibilities and limitations of the underlying theory, on developing new applications of quantum information and on physically realizing controllable quantum devices. The purpose of this primer is to provide an elementary introduction to quantum information processing (Sect. 2), and then to briefly explain how we hope to exploit the advantages of quantum information (Sect. 3). These two sections can be read independently. For reference, we have included a glossary (Sect. 4) of the main terms of quantum information.

When we use the word "information", we generally think of the things we can talk about, broadcast, write down, or otherwise record. Such records can exist in many forms, such as sound waves, electrical signals in a telephone wire, characters on paper, pit patterns on an optical disk, or magnetization on a computer hard disk. A crucial property of information is that it is "fungible": It can be represented in many different physical forms and easily converted from one form to another without changing its meaning. In this sense information exists independently of the devices used to represent it, but requires at least one physical representation to be useful.

We call the familiar information stored in today's computers "classical" or "deterministic" to distinguish it from quantum information. It is no accident that classical information is the basis of all human knowledge. Any information passing through our senses is best modeled by classical discrete or continuous information. Therefore, when considering any other kind of information, we need to provide a method for extracting classically meaningful information. We begin by recalling the basic ideas of classical information in a way that illustrates the general procedure for building an information processing theory.

1 Classical Information

The basic unit of classical deterministic information is the “bit”. A bit is an abstract entity or “system” that can be in one of the two states symbolized by 0 and 1. At this point, the symbols for the two states have no numeric meaning. That is why we have used a font different from that for the numbers 0 and 1. By making a clear distinction between the bit and its states we emphasize that a bit should be physically realized as a system or device whose states correspond to the ideal bit’s states. For example, if you are reading this primer on paper, the system used to realize a bit is a reserved location on the surface, and the state depends on the pattern of ink (0 or 1) in that location. In a computer, the device realizing a bit can be a combination of transistors and other integrated circuit elements with the state of the bit determined by the distribution of charge.

In order to make use of information it must be possible to manipulate (or “process”) the states of information units. The elementary operations that can be used for this purpose are called “gates”. Two one-bit gates are the $\mathbf{not}$ and the $\mathbf{reset}$ gates. Applying the $\mathbf{not}$ gate to a bit has the effect of “flipping” the state of the bit. For example, if the initial state of the bit is 0, then the state after applying $\mathbf{not}$ is $\mathbf{not}(0) = 1$. We can present the effect of the gate in the following form:

$$\begin{array}{ccl} \text{Initial State} & & \text{Final State} \\ 0 & \rightarrow & \mathbf{not}(0) = 1, \\ 1 & \rightarrow & \mathbf{not}(1) = 0. \end{array} \tag{1}$$

The $\mathbf{reset}$ gate sets the state to 0 regardless of the input:

$$\begin{array}{ccl} \text{Initial State} & & \text{Final State} \\ 0 & \rightarrow & \mathbf{reset}(0) = 0, \\ 1 & \rightarrow & \mathbf{reset}(1) = 0. \end{array} \tag{2}$$

By applying a combination of $\mathbf{not}$ and $\mathbf{reset}$ gates one can transform the state of a bit in every possible way.

Information units can be combined to represent more information. Bits are typically combined into sequences. The states of such a sequence are symbolized by strings of state symbols for the constituent bits. For example a two-bit sequence can be in one of the following four states: $00, 01, 10$ and 11. The different bits are distinguished by their position in the sequence.

The one-bit gates can be applied to any bit in a sequence. For example, the $\mathbf{not}$ gate applied to the second bit of a three-bit sequence in the state 011 changes the state to 001.

One-bit gates act independently on each bit. To compute with multiple bits, we need gates whose action can correlate the states of two or more bits. One such gate is the $\mathbf{nand}$ (“not and”) gate, which acts on two bits in a bit sequence. Its effect is to set the state of the first bit to 0 if both the first and the second bit are 1, otherwise it sets it to 1. Here is what happens when $\mathbf{nand}$ is applied to two consecutive bits:

$$\begin{array}{ccl} \text{Initial State} & & \text{Final State} \\ 00 & \rightarrow & \mathbf{nand}(00) = 10, \\ 01 & \rightarrow & \mathbf{nand}(01) = 11, \\ 10 & \rightarrow & \mathbf{nand}(10) = 10, \\ 11 & \rightarrow & \mathbf{nand}(11) = 01. \end{array} \tag{3}$$

The $\mathbf{nand}$ gate can be applied to any two bits in a sequence. For example, it can be applied to the fourth and second bits (in this order) of four bits, in which case the initial state $\mathtt{1101}$ is transformed to $\mathtt{1100}$, setting the fourth bit to $\mathtt{0}$.

Other operations on bit sequences include adding a new bit to the beginning ($\mathbf{prepend}$) or end ($\mathbf{append}$) of a bit sequence. The new bit is always initialized to $\mathtt{0}$. It is also possible to discard the first or last bit, regardless of its state. Versions of these operations that are conditional on the state of another bit may also be used. An example is the conditional append operation: "if the k'th bit is in the state $\mathtt{0}$ then append a bit."

The gates just introduced suffice for implementing arbitrary state transformations of a given bit sequence. Instructions for applying gates in a particular order are called a "circuit". An important part of investigations in information processing is to determine the minimum resources required to perform information processing tasks. For a given circuit, the two primary resources are the number of gates and the total number of bits used. The "circuit complexity" of a desired transformation is the minimum number of gates needed to implement it.

The model of computation defined by the ability to apply gates in a fixed sequence is called the "circuit model". Classical computation extends the circuit model by providing a means for repeating blocks of instructions indefinitely or until a desired condition is achieved. In principle, it is possible to conceive of a general purpose computer as a device that repeatedly applies the same circuit to the beginnings of several bit sequences. In this introduction, we take for granted a traditional programmable computer based on classical information. Thus a "quantum algorithm" is a program written for such a computer with additional instructions for applying gates to quantum information. The computational power of this model is equivalent to that of other general purpose models of quantum computation, such as quantum Turing machines [7].

For an introduction to algorithms and their analysis, see [8]. A useful textbook on computational complexity with an introduction to classical computation and computational machine models is [9].

2 Quantum Information

The foundations of an information processing theory can be constructed by the procedure we followed in the previous section:

1. Define the basic unit of information.
2. Give the means for processing one unit.
3. Describe how multiple units can be combined.
4. Give the means for processing multiple units.
5. Show how to convert the content of any of the extant units to classical information.

Note that the last step was not required for classical information processing.

In this section, we follow the general procedure for defining an information processing theory to introduce quantum information processing. A simple example that exhibits the advantages of quantum information is given in Sect. 2.8. A version of the quantum factoring algorithm is described in Sect. 2.10.

2.1 The Quantum Bit

The fundamental resource and basic unit of quantum information is the quantum bit (qubit), which behaves like a classical bit enhanced by the superposition principle (see below). From a physical point of view, a qubit is represented by an ideal two-state quantum system. Examples of such systems include photons (vertical and horizontal polarization), electrons and other spin-$\frac{1}{2}$ systems (spin up and down), and systems defined by two energy levels of atoms or ions. From the beginning the two-state system played a central role in studies of quantum mechanics. It is the most simple quantum system, and in principle all other quantum systems can be modeled in the state space of collections of qubits.

From the information processing point of view, a qubit's state space contains the two "logical", or "computational", states $|0\rangle$ and $|1\rangle$. The so-called "ket" notation for these states was introduced by P. Dirac, and its variations are widely used in quantum physics. One can think of the pair of symbols "$|$" and "$\rangle$" as representing the qubit system. Their content specifies a state for the system. In this context 0 and 1 are system-independent state labels. When, say, 0 is placed within the ket, the resulting expression $|0\rangle$ represents the corresponding state of a specific qubit.

The initial state of a qubit is always one of the logical states. Using operations to be introduced later, we can obtain states which are "superpositions" of the logical states. Superpositions can be expressed as sums $\alpha|0\rangle + \beta|1\rangle$ over the logical states with complex coefficients. The complex numbers α and β are called the "amplitudes" of the superposition. The existence of such superpositions of distinguishable states of quantum systems is one of the basic tenets of quantum theory called the "superposition principle". Another way of writing a general superposition is as a vector

$$\alpha|0\rangle + \beta|1\rangle \leftrightarrow \begin{pmatrix} \alpha \\ \beta \end{pmatrix}, \tag{4}$$

where the two-sided arrow "$\leftrightarrow$" is used to denote the correspondence between expressions that mean the same thing.

The qubit states that are superpositions of the logical states are called "pure" states: A superposition $\alpha|0\rangle + \beta|1\rangle$ is a pure state if the corresponding vector has length 1, that is $|\alpha|^2 + |\beta|^2 = 1$. Such a superposition or vector is said to be "normalized". (For a complex number given by $\gamma = x + iy$, one can evaluate $|\gamma|^2 = x^2 + y^2$. Here, x and y are the real and imaginary part of γ, and the symbol i is a square root of -1, that is, $i^2 = -1$. The conjugate of γ is $\overline{\gamma} = x - iy$. Thus $|\gamma|^2 = \overline{\gamma}\gamma$.) Here are a few examples of states given in both the ket and the vector notation:

$$\begin{aligned} |\psi_1\rangle &= \Big(|0\rangle + |1\rangle\Big)/\sqrt{2} \leftrightarrow \begin{pmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \end{pmatrix}, && (5) \\ |\psi_2\rangle &= \frac{3}{5}|0\rangle - \frac{4}{5}|1\rangle \leftrightarrow \begin{pmatrix} 3/5 \\ -4/5 \end{pmatrix}, && (6) \\ |\psi_3\rangle &= \frac{i3}{5}|0\rangle - \frac{i4}{5}|1\rangle \leftrightarrow \begin{pmatrix} i3/5 \\ -i4/5 \end{pmatrix}. && (7) \end{aligned}$$

The state $|\psi_3\rangle$ is obtained from $|\psi_2\rangle$ by multiplication with i. It turns out that two states cannot be distinguished if one of them is obtained by multiplying the other by a "phase" $e^{i\theta}$. Note how we have generalized the ket notation by introducing expressions such as $|\psi\rangle$ for arbitrary states.

The superposition principle for quantum information means that we can have states that are sums of logical states with complex coefficients. There is another, more familiar type of information whose states are combinations of logical states. The basic unit of this type of information is the probabilistic bit (pbit). Intuitively, a pbit can be thought of as representing the as-yet-undetermined outcome of a coin flip. Since we need the idea of probability to understand how quantum information converts to classical information, we briefly introduce pbits.

A pbit's state space is a probability distribution over the states of a bit. One very explicit way to symbolize such a state is by using the expression $\{p{:}\mathsf{0}, (1-p){:}\mathsf{1}\}$, which means that the pbit has probability p of being $\mathsf{0}$ and $1-p$ of being $\mathsf{1}$. Thus a state of a pbit is a "probabilistic" combination of the two logical states, where the coefficients are nonnegative real numbers summing to 1. A typical example is the unbiased coin in the process of being flipped. If "tail" and "head" represent $\mathsf{0}$ and $\mathsf{1}$, respectively, the coin's state is $\{\frac{1}{2}{:}\mathsf{0}, \frac{1}{2}{:}\mathsf{1}\}$. After the outcome of the flip is known, the state "collapses" to one of the logical states $\mathsf{0}$ and $\mathsf{1}$. In this way, a pbit is converted to a classical bit. If the pbit is probabilistically correlated with other pbits, the collapse associated with learning the pbit's logical state changes the overall probability distribution by a process called "conditioning on the outcome".

A consequence of the conditioning process is that we never actually "see" a probability distribution. We only see classical deterministic bit states. According to the frequency interpretation of probabilities, the original probability distribution can only be inferred after one looks at many independent pbits in the same state $\{p{:}\mathsf{0}, (1-p){:}\mathsf{1}\}$: In the limit of infinitely many pbits, p is given by the fraction of pbits seen to be in the state $\mathsf{0}$. As we will explain, we can never "see" a general qubit state either. For qubits there is a process analogous to conditioning. This process is called "measurement" and converts qubit states to classical information.

Information processing with pbits has many advantages over deterministic information processing with bits. One advantage is that algorithms are often much easier to design and analyze if they are probabilistic. Examples include many optimization and physics simulation algorithms. In some cases, the best available probabilistic algorithm is more efficient than any known deterministic algorithm. An example is an algorithm for determining whether a number is prime or not. It is not known whether every probabilistic algorithm can be "derandomized" efficiently. There are important communication problems that can be solved probabilistically but not deterministically. For a survey, see [10].

What is the difference between bits, pbits and qubits? One way to visualize the difference and see the enrichment provided by pbits and qubits is shown in Fig. 1.

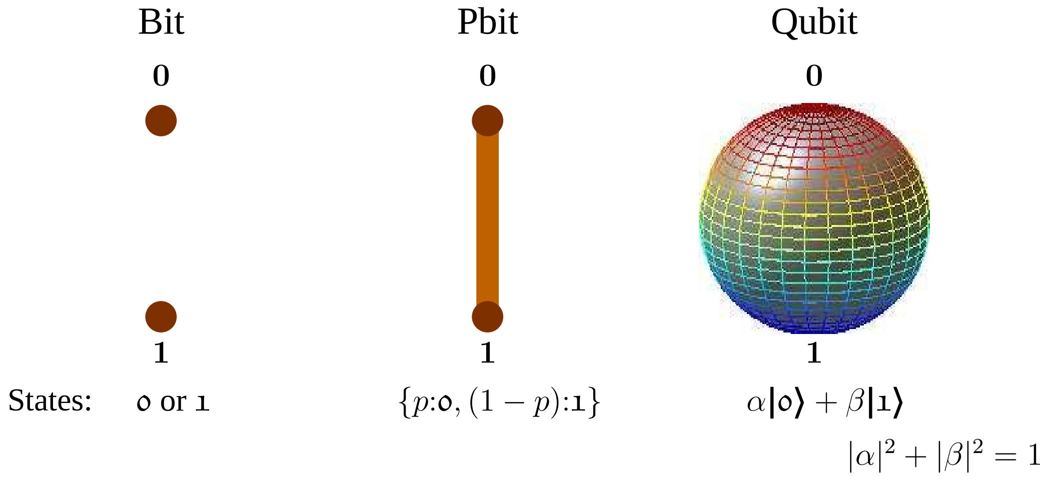

Bit
Pbit
Qubit
0
0
0
1
1
1
States:
0 or 1
$\{p{:}0, (1-p){:}1\}$
$\alpha|0\rangle + \beta|1\rangle$
$|\alpha|^2 + |\beta|^2 = 1$

FIG. 1: Visual comparison of the state spaces of different information units. The states of a bit correspond to two points. The states of a pbit can be thought of as "convex" combinations of the states of a bit and therefore can be visualized as lying on the line connecting the two bit states. A qubit's pure states correspond to the surface of the unit sphere in three dimensions, where the logical states correspond to the poles. This representation of qubit states is called the "Bloch sphere". The explicit correspondence is discussed at the end of Sect. 2.7. See also the definition and use of the Bloch sphere in [11]. The correspondence between the pure states and the sphere is physically motivated and comes from a way of viewing a spin-$\frac{1}{2}$ system as a small quantum magnet. Intuitively, a state is determined by the direction of the north pole of the magnet.

2.2 Processing One Qubit

The quantum version of the **not** gate for bits exchanges the two logical states. That is, using ket notation,

$$\mathbf{not}\Big(\alpha|0\rangle + \beta|1\rangle\Big) = \alpha|1\rangle + \beta|0\rangle = \beta|0\rangle + \alpha|1\rangle. \tag{8}$$

In vector notation this equation becomes

$$\mathbf{not}\begin{pmatrix}\alpha\\ \beta\end{pmatrix} = \begin{pmatrix}\beta\\ \alpha\end{pmatrix}. \tag{9}$$

Another way of expressing the effect of **not** is by multiplying the vector by a matrix representing **not**:

$$\mathbf{not}\begin{pmatrix}\alpha\\ \beta\end{pmatrix} = \begin{pmatrix}0 & 1\\ 1 & 0\end{pmatrix}\begin{pmatrix}\alpha\\ \beta\end{pmatrix} = \begin{pmatrix}\beta\\ \alpha\end{pmatrix}, \tag{10}$$

so we that can identify the action of **not** with the matrix $\sigma_x = \begin{pmatrix}0 & 1\\ 1 & 0\end{pmatrix}$. An even simpler gate is the one that does nothing. We call this the **noop** gate, and its matrix form is the identity matrix as shown in the following equation:

$$\mathbf{noop}\begin{pmatrix}\alpha\\ \beta\end{pmatrix} = \begin{pmatrix}1 & 0\\ 0 & 1\end{pmatrix}\begin{pmatrix}\alpha\\ \beta\end{pmatrix} = \begin{pmatrix}\alpha\\ \beta\end{pmatrix}. \tag{11}$$

The $\mathbf{noop}$ and $\mathbf{not}$ gates are "reversible". In other words, we can undo their actions by applying other gates. For example, the action of the $\mathbf{not}$ gate can be undone by another $\mathbf{not}$ gate. The action of every reversible quantum gate can be represented by matrix multiplication, where the matrix has the additional property of preserving the length of vectors. Such matrices are called "unitary" and are characterized by the equation $A^\dagger A = \mathbb{1}$, where $A^\dagger$ is the conjugate transpose of A and $\mathbb{1}$ is the identity matrix. (The conjugate transpose of a matrix is computed by flipping the matrix across the main diagonal and conjugating the complex numbers.) For gates represented by a matrix, the unitarity condition is necessary and sufficient for ensuring that pure states get mapped to pure states.

Because qubit states can be represented as points on a sphere, reversible one-qubit gates can be thought of as rotations of the Bloch sphere. This is why such quantum gates are often called "rotations". As explained in detail in [11], rotations around the x, y and z axis are in a sense generated by the three Pauli matrices

$$\sigma_x = \begin{pmatrix}0 & 1\\ 1 & 0\end{pmatrix}\ ,\ \ \sigma_y = \begin{pmatrix}0 & -i\\ i & 0\end{pmatrix}\ ,\ \ \sigma_z = \begin{pmatrix}1 & 0\\ 0 & -1\end{pmatrix}\ , \tag{12}$$

each of which represents a one-qubit gate. For example, a rotation around the x-axis by an angle ϕ is given by $e^{-i\sigma_x\phi/2} = \cos(\phi/2)\mathbb{1} - i\sin(\phi/2)\sigma_x$. To obtain this identity, one can use the power series for e^A, $e^A = \sum_{k=0}^{\infty}\frac{1}{k!}A^k$, and exploit the fact that $\sigma_x^2 = \mathbb{1}$ to simplify the expression. Here are some gates that can be defined with the help of rotations:

$$\begin{array}{lrcl}
90^\circ\ x\text{-rotation}: & \mathbf{rotx}_{90^\circ} & = & \frac{1}{\sqrt{2}}\begin{pmatrix}1 & -i\\ -i & 1\end{pmatrix}\\[2ex]
90^\circ\ y\text{-rotation}: & \mathbf{roty}_{90^\circ} & = & \frac{1}{\sqrt{2}}\begin{pmatrix}1 & -1\\ 1 & 1\end{pmatrix}\\[2ex]
\phi\ z\text{-rotation}: & \mathbf{rotz}_{\phi} & = & \begin{pmatrix}e^{-i\phi/2} & 0\\ 0 & e^{i\phi/2}\end{pmatrix}\\[2ex]
\text{Hadamard gate}: & \mathbf{H} & = & \frac{1}{\sqrt{2}}\begin{pmatrix}1 & 1\\ 1 & -1\end{pmatrix}
\end{array} \tag{13}$$

The rotation gates often show up in controlling spins or ions with radio-frequency pulses or lasers. The Hadamard gate is used primarily by quantum programmers. It can be expressed as a product of a 90° y-rotation and σ_z.

To check directly that the rotation gates are reversible one can determine their inverses. In this case and as expected, the inverse of a rotation is the rotation around the same axis in the opposite direction. For example, the inverses of the $\mathbf{roty}_{90^\circ}$ and $\mathbf{rotz}_{\phi}$ gates are given by

$$\begin{aligned}\mathbf{roty}_{-90^\circ} &= \tfrac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ -1 & 1 \end{pmatrix} \\ \mathbf{rotz}_{-\phi} &= \begin{pmatrix} e^{i\phi/2} & 0 \\ 0 & e^{-i\phi/2} \end{pmatrix}\end{aligned} \tag{14}$$

Another useful property of the rotation gates is that the angles add when rotations are applied around the same axis. For example, $\mathbf{rotz}_{\phi}\mathbf{rotz}_{\theta} = \mathbf{rotz}_{\phi+\theta}$.

The ket notation can be extended so that we can write gates in a compact form that readily generalizes to multiple qubits. To do so we have to introduce expressions such as $\langle\psi| = \alpha\langle 0| + \beta\langle 1|$. This is called the "bra" notation. The terminology comes from the term "bracket": The 'bra" is the left and the "ket" is the right part of a matched pair of brackets. From the vector point of view, $\langle\psi|$ corresponds to the row vector (α, β). Note that a column vector multiplied by a row vector yields a matrix. In the bra-ket notation, this corresponds to multiplying a ket $|\psi\rangle$ by a bra $\langle\phi|$, written as $|\psi\rangle\langle\phi|$. Since this represents an operator on states, we expect to be able to compute the effect of $|\psi\rangle\langle\phi|$ on a state $|\varphi\rangle$ by forming the product. To be able to evaluate such products with one-qubit kets and bras, we need the following two rules.

Distributivity. You can rewrite sums and products using distributivity. For example,

$$\Big(\frac{3}{5}\langle 0| + \frac{4}{5}\langle 1|\Big) i|1\rangle = \frac{i3}{5}\langle 0||1\rangle + \frac{i4}{5}\langle 1||1\rangle. \tag{15}$$

Observe that we can combine the amplitudes of terms, but we cannot rearrange the order of the bras and kets in a product.

Inner product evaluation. The product of a logical "bra" and a logical "ket" is evaluated according to the identities

$$\begin{aligned}\langle 0||0\rangle &= 1, \\ \langle 0||1\rangle &= 0, \\ \langle 1||0\rangle &= 0, \\ \langle 1||1\rangle &= 1.\end{aligned} \tag{16}$$

It follows that for logical states, if a bra multiplies a ket, the result cancels unless the states match, in which case the answer is 1. Applying inner product evaluation to the example (Eq. 15) results in

$$\frac{i3}{5}\langle 0||1\rangle + \frac{i4}{5}\langle 1||1\rangle = \frac{i3}{5}0 + \frac{i4}{5}1 = \frac{i4}{5}. \tag{17}$$

To simplify the notation, we can omit one of the two vertical bars in products such as $\langle a||b\rangle$ and write $\langle a|b\rangle$.

To understand inner product evaluation, think of the expressions as products of row and column vectors. For example,

$$\langle 0|1\rangle \leftrightarrow \quad \begin{pmatrix} 1 & 0 \end{pmatrix}\begin{pmatrix} 0 \\ 1 \end{pmatrix} = 0, \tag{18}$$

That is, as vectors the two states $|\mathfrak{0}\rangle$ and $|\mathfrak{1}\rangle$ are orthogonal. In general, if $|\phi\rangle$ and $|\psi\rangle$ are states, then $\langle\phi|\psi\rangle$ is the "inner product" or "overlap" of the two states. In the expression for the overlap, $\langle\phi|$ is computed from $|\phi\rangle = \alpha|\mathfrak{0}\rangle + \beta|\mathfrak{1}\rangle$ by conjugating the coefficients and converting the logical kets to bras: $\langle\phi| = \overline{\alpha}\langle\mathfrak{0}| + \overline{\beta}\langle\mathfrak{1}|$. In the vector representation, this is the conjugate transpose of the column vector for $|\phi\rangle$, so the inner product agrees with the usual one. Two states are orthogonal if their overlap is zero. We write $|\phi\rangle^\dagger = \langle\phi|$ and $\langle\phi|^\dagger = |\phi\rangle$.

Every linear operator on states can be expressed with the bra-ket notation. For example, the bra-ket expression for the $\mathbf{noop}$ gate is $\mathbf{noop} = |\mathfrak{0}\rangle\langle\mathfrak{0}| + |\mathfrak{1}\rangle\langle\mathfrak{1}|$. To apply $\mathbf{noop}$ to a qubit, you multiply its state on the left by the bra-ket expression:

$$\begin{aligned}
\mathbf{noop}\Big(\alpha|\mathfrak{0}\rangle + \beta|\mathfrak{1}\rangle\Big) &= \Big(|\mathfrak{0}\rangle\langle\mathfrak{0}| + |\mathfrak{1}\rangle\langle\mathfrak{1}|\Big)\Big(\alpha|\mathfrak{0}\rangle + \beta|\mathfrak{1}\rangle\Big) \\
&= |\mathfrak{0}\rangle\langle\mathfrak{0}|\Big(\alpha|\mathfrak{0}\rangle + \beta|\mathfrak{1}\rangle\Big) + |\mathfrak{1}\rangle\langle\mathfrak{1}|\Big(\alpha|\mathfrak{0}\rangle + \beta|\mathfrak{1}\rangle\Big) \\
&= \alpha|\mathfrak{0}\rangle\langle\mathfrak{0}|\mathfrak{0}\rangle + \beta|\mathfrak{0}\rangle\langle\mathfrak{0}|\mathfrak{1}\rangle + \alpha|\mathfrak{1}\rangle\langle\mathfrak{1}|\mathfrak{0}\rangle + \beta|\mathfrak{1}\rangle\langle\mathfrak{1}|\mathfrak{1}\rangle \\
&= \alpha|\mathfrak{0}\rangle 1 + \beta|\mathfrak{0}\rangle 0 + \alpha|\mathfrak{1}\rangle 0 + \beta|\mathfrak{1}\rangle 1 \\
&= \alpha|\mathfrak{0}\rangle + \beta|\mathfrak{1}\rangle
\end{aligned} \tag{19}$$

One way to think about an operator such as $|a\rangle\langle b|$ is to notice that when it is used to operate on a ket expression, the $\langle b|$ picks out the matching kets in the state, which are then changed to $|a\rangle$. For example, we can write the $\mathbf{not}$ operation as $\mathbf{not} = |\mathfrak{0}\rangle\langle\mathfrak{1}| + |\mathfrak{1}\rangle\langle\mathfrak{0}|$.

The coefficients of the $|a\rangle\langle b|$ in a bra-ket representation of a gate correspond to matrix entries in the matrix representation. The relationship is defined by

$$\alpha_{00}|\mathfrak{0}\rangle\langle\mathfrak{0}| + \alpha_{01}|\mathfrak{0}\rangle\langle\mathfrak{1}| + \alpha_{10}|\mathfrak{1}\rangle\langle\mathfrak{0}| + \alpha_{11}|\mathfrak{1}\rangle\langle\mathfrak{1}| \leftrightarrow \begin{pmatrix} \alpha_{00} & \alpha_{01} \\ \alpha_{10} & \alpha_{11} \end{pmatrix}. \tag{20}$$

2.3 Two Quantum Bits

Some states of two quantum bits can be symbolized by the juxtaposition (or multiplication) of states of each quantum bit. In particular, the four logical states $|\mathfrak{0}\rangle|\mathfrak{0}\rangle, |\mathfrak{0}\rangle|\mathfrak{1}\rangle, |\mathfrak{1}\rangle|\mathfrak{0}\rangle$, and $|\mathfrak{1}\rangle|\mathfrak{1}\rangle$ are acceptable pure states for two quantum bits. In these expressions, we have distinguished the qubits by position (first or second). It is easier to manipulate state expressions if we explicitly name the qubits, say A and B. We can then distinguish the kets by writing, for example, $|\psi\rangle_{\mathsf{A}}$ for a state of qubit A. Now the state $|\mathfrak{0}\rangle|\mathfrak{1}\rangle$ can be written with explicit qubit names (or "labels") as

$$|\mathfrak{0}\rangle_{\mathsf{A}}|\mathfrak{1}\rangle_{\mathsf{B}} = |\mathfrak{1}\rangle_{\mathsf{B}}|\mathfrak{0}\rangle_{\mathsf{A}} = |\mathfrak{01}\rangle_{\mathsf{AB}} = |\mathfrak{10}\rangle_{\mathsf{BA}}. \tag{21}$$

Having explicit labels allows us to unambiguously reorder the states in a product of states belonging to different qubits. We say that kets for different qubits "commute".

So far we have seen four states of two qubits, which are the logical states that correspond to the states of two bits. As in the case of one qubit, the superposition principle can be used to get all the other pure states. Each state of two qubits is therefore of the form

$$\alpha|\mathfrak{00}\rangle_{\mathsf{AB}} + \beta|\mathfrak{01}\rangle_{\mathsf{AB}} + \gamma|\mathfrak{10}\rangle_{\mathsf{AB}} + \delta|\mathfrak{11}\rangle_{\mathsf{AB}}, \tag{22}$$

where α, β, γ, and δ are complex numbers. Again, there is a column vector form for the state:

$$\begin{pmatrix} \alpha \\ \beta \\ \gamma \\ \delta \end{pmatrix}, \tag{23}$$

and this vector has to be of unit length, that is $|\alpha|^2 + |\beta|^2 + |\gamma|^2 + |\delta|^2 = 1$. When using the vector form for qubit states, one has to be careful about the convention used for ordering the coefficients.

Other examples of two-qubit states in ket notation are the following:

$$\begin{aligned}
|\psi_1\rangle_{\mathsf{AB}} &= \frac{1}{\sqrt{2}}\Big(|0\rangle_{\mathsf{A}} + |1\rangle_{\mathsf{A}}\Big)|1\rangle_{\mathsf{B}}, \\
|\psi_2\rangle_{\mathsf{AB}} &= \frac{1}{\sqrt{2}}\Big(|0\rangle_{\mathsf{A}} - |1\rangle_{\mathsf{A}}\Big)\frac{1}{\sqrt{2}}\Big(|0\rangle_{\mathsf{B}} + i|1\rangle_{\mathsf{B}}\Big) \\
&= \frac{1}{2}\Big(|00\rangle_{\mathsf{AB}} + i|01\rangle_{\mathsf{AB}} - |10\rangle_{\mathsf{AB}} - i|11\rangle_{\mathsf{AB}}\Big) \\
|\psi_3\rangle_{\mathsf{AB}} &= \frac{1}{\sqrt{2}}\Big(|00\rangle_{\mathsf{AB}} + |11\rangle_{\mathsf{AB}}\Big), \\
|\psi_4\rangle_{\mathsf{AB}} &= \frac{1}{\sqrt{2}}\Big(|01\rangle_{\mathsf{AB}} - |10\rangle_{\mathsf{AB}}\Big).
\end{aligned} \tag{24}$$

The first two of these states have the special property that they can be written as a product $|\phi_1\rangle_{\mathsf{A}}|\phi_2\rangle_{\mathsf{B}}$ of a state of qubit A and a state of qubit B. The second expression for $|\psi_2\rangle$ shows that the product decomposition is not always easy to see. Such states are called “product” states. The last two states, $|\psi_3\rangle_{\mathsf{AB}}$ and $|\psi_4\rangle_{\mathsf{AB}}$ are two of the famous Bell states. They have no such representation as a product of independent states of each qubit. They are said to be “entangled” because they contain a uniquely quantum correlation between the two qubits. Pbits can also have correlations that cannot be decomposed into product states, but the entangled states have additional properties that make them very useful. For example, if Alice and Bob each have one of the qubits that together are in the state $|\psi_3\rangle_{\mathsf{AB}}$, they can use them to create a secret bit for encrypting their digital communications.

2.4 Processing Two Qubits

The simplest way of modifying the state of two qubits is to apply one of the one-qubit gates. If the gates are expressed in the bra-ket notation, all we need to do is add qubit labels so that we know which qubit each bra or ket belongs to. For example, the **not** gate for qubit B is written as

$$\mathbf{not}^{(\mathsf{B})} = |0\rangle_{\mathsf{B}}{}^{\mathsf{B}}\langle 1| + |1\rangle_{\mathsf{B}}{}^{\mathsf{B}}\langle 0|. \tag{25}$$

The labels for bra expressions occur as left superscripts. To apply expressions like this to states, we need one more rule:

Commutation. Kets and bras with different labels can be interchanged in products (they "commute"). This is demonstrated by the following example:

$$\begin{aligned}
\left(|0\rangle_{\mathsf{B}}{}^{\mathsf{B}}\langle 1|\right)|01\rangle_{\mathsf{AB}} &= |0\rangle_{\mathsf{B}}{}^{\mathsf{B}}\langle 1||0\rangle_{\mathsf{A}}|1\rangle_{\mathsf{B}} \\
&= |0\rangle_{\mathsf{A}}\ |0\rangle_{\mathsf{B}}{}^{\mathsf{B}}\langle 1|\ |1\rangle_{\mathsf{B}} \\
&= |0\rangle_{\mathsf{A}}\ |0\rangle_{\mathsf{B}}\,{}^{\mathsf{B}}\langle 1|1\rangle_{\mathsf{B}} \\
&= |0\rangle_{\mathsf{A}}|0\rangle_{\mathsf{B}} = |00\rangle_{\mathsf{AB}}.
\end{aligned} \tag{26}$$

Note that we cannot merge the two vertical bars in expressions such as ${}^{\mathsf{B}}\langle 1||0\rangle_{\mathsf{A}}$ because the two terms belong to different qubits. The bars can only be merged when the expression is an inner product, which requires that the two terms belong to the same qubit.

With the rules for bra-ket expressions in hand, we can apply the $\mathbf{not}$ gate to one of our Bell states to see how it acts:

$$\begin{aligned}
\mathbf{not}^{(\mathsf{B})}\frac{1}{\sqrt{2}}\left(|00\rangle_{\mathsf{AB}}+|11\rangle_{\mathsf{AB}}\right) &= \left(|0\rangle_{\mathsf{B}}{}^{\mathsf{B}}\langle 1|+|1\rangle_{\mathsf{B}}{}^{\mathsf{B}}\langle 0|\right)\frac{1}{\sqrt{2}}\left(|00\rangle_{\mathsf{AB}}+|11\rangle_{\mathsf{AB}}\right) \\
&= \frac{1}{\sqrt{2}}\left(|0\rangle_{\mathsf{B}}{}^{\mathsf{B}}\langle 1|\left(|00\rangle_{\mathsf{AB}}+|11\rangle_{\mathsf{AB}}\right)+|1\rangle_{\mathsf{B}}{}^{\mathsf{B}}\langle 0|\left(|00\rangle_{\mathsf{AB}}+|11\rangle_{\mathsf{AB}}\right)\right) \\
&= \frac{1}{\sqrt{2}}\left(|0\rangle_{\mathsf{B}}{}^{\mathsf{B}}\langle 1||00\rangle_{\mathsf{AB}}+|0\rangle_{\mathsf{B}}{}^{\mathsf{B}}\langle 1||11\rangle_{\mathsf{AB}}+|1\rangle_{\mathsf{B}}{}^{\mathsf{B}}\langle 0||00\rangle_{\mathsf{AB}}+|1\rangle_{\mathsf{B}}{}^{\mathsf{B}}\langle 0||11\rangle_{\mathsf{AB}}\right) \\
&= \frac{1}{\sqrt{2}}\left(|0\rangle_{\mathsf{A}}|0\rangle_{\mathsf{B}}{}^{\mathsf{B}}\langle 1||0\rangle_{\mathsf{B}}+|1\rangle_{\mathsf{A}}|0\rangle_{\mathsf{B}}{}^{\mathsf{B}}\langle 1||1\rangle_{\mathsf{B}}+|0\rangle_{\mathsf{A}}|1\rangle_{\mathsf{B}}{}^{\mathsf{B}}\langle 0||0\rangle_{\mathsf{B}}+|1\rangle_{\mathsf{A}}|1\rangle_{\mathsf{B}}{}^{\mathsf{B}}\langle 0||1\rangle_{\mathsf{B}}\right) \\
&= \frac{1}{\sqrt{2}}\left(|0\rangle_{\mathsf{A}}|0\rangle_{\mathsf{B}}\ 0+|1\rangle_{\mathsf{A}}|0\rangle_{\mathsf{B}}\ 1+|0\rangle_{\mathsf{A}}|1\rangle_{\mathsf{B}}\ 1+|1\rangle_{\mathsf{A}}|1\rangle_{\mathsf{B}}\ 0\right) \\
&= \frac{1}{\sqrt{2}}\left(|1\rangle_{\mathsf{A}}|0\rangle_{\mathsf{B}}+|0\rangle_{\mathsf{A}}|1\rangle_{\mathsf{B}}\right) = \frac{1}{\sqrt{2}}\left(|01\rangle_{\mathsf{AB}}+|10\rangle_{\mathsf{AB}}\right).
\end{aligned} \tag{27}$$

The effect of the gate was to flip the state symbols for qubit B, which results in another Bell state.

The gate $\mathbf{not}^{(\mathsf{B})}$ can also be written as a 4×4 matrix acting on the vector representation of a two-qubit state. However, the relationship between this matrix and the one-qubit matrix is not as obvious as for the bra-ket expression. The matrix is

$$\mathbf{not}^{(\mathsf{B})} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}, \tag{28}$$

which swaps the top two and the bottom two entries of a state vector.

One way to see the relationship between the one and the two-qubit representation of the gate $\mathbf{not}^{(\mathsf{B})}$ is to notice that because the $\mathbf{noop}$ gate acts as the identity, and because we can act on different qubits independently, $\mathbf{noop}^{(\mathsf{A})}\mathbf{not}^{(\mathsf{B})} \simeq \mathbf{not}^{(\mathsf{B})}$. The matrix for $\mathbf{not}^{(\mathsf{B})}$ can be expressed as a "Kronecker product" ("$\otimes$") of the matrices for $\mathbf{noop}$ and $\mathbf{not}$:

$$\mathbf{noop}^{(\mathsf{A})}\mathbf{not}^{(\mathsf{B})} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \otimes \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}.$$

$$
\begin{aligned}
&= \begin{pmatrix} 1\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} & 0\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \\ \\ 0\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} & 1\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \end{pmatrix} \\
&= \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}.
\end{aligned}
\tag{29}
$$

The Kronecker product of two matrices expands the first matrix by multiplying each entry by the second matrix. A disadvantage of the matrix representation of quantum gates is that it depends on the number and order of the qubits. However, it is often easier to visualize what the operation does by writing down the corresponding matrix.

One cannot do much with one-bit classical gates. Similarly, the utility of one-qubit gates is limited. In particular, it is not possible to obtain a Bell state starting from $|00\rangle_{AB}$ or any other product state. We therefore need to introduce at least one two-qubit gate not expressible as the product of two one-qubit gates. The best known such gate is the "controlled-not" ($\mathbf{cnot}$) gate. Its action can be described by the statement, "if the first bit is 1, flip the second bit, otherwise do nothing". The bra-ket and matrix representations for this action are

$$
\begin{aligned}
\mathbf{cnot}^{(AB)} &= |0\rangle_{A}{}^{A}\langle 0| + |1\rangle_{A}{}^{A}\langle 1|\Big(|0\rangle_{B}{}^{B}\langle 1| + |1\rangle_{B}{}^{B}\langle 0|\Big) \\
&= \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}.
\end{aligned}
\tag{30}
$$

The $\mathbf{cnot}$ gate is reversible because its action is undone if a second $\mathbf{cnot}$ is applied. This outcome is easy to see by computing the square of the matrix for $\mathbf{cnot}$, which yields the identity matrix. As an exercise in manipulating bras and kets, let us calculate the product of two $\mathbf{cnot}$ gates by using the bra-ket representation.

$$
\mathbf{cnot}^{(AB)}\mathbf{cnot}^{(AB)} = \Big(|0\rangle_{A}{}^{A}\langle 0| + |1\rangle_{A}{}^{A}\langle 1|\Big(|0\rangle_{B}{}^{B}\langle 1| + |1\rangle_{B}{}^{B}\langle 0|\Big)\Big)\Big(|0\rangle_{A}{}^{A}\langle 0| + |1\rangle_{A}{}^{A}\langle 1|\Big(|0\rangle_{B}{}^{B}\langle 1| + |1\rangle_{B}{}^{B}\langle 0|\Big)\Big). \tag{31}
$$

The first step is to expand this expression by multiplying out. Expressions such as $|0\rangle_{A}{}^{A}\langle 0|\,|1\rangle_{A}{}^{A}\langle 1|$ cancel because of the inner product evaluation rule, ${}^{A}\langle 0|1\rangle_{A} = 0$. One can also reorder bras and kets with different labels and rewrite $|0\rangle_{A}{}^{A}\langle 0|\,|0\rangle_{A}{}^{A}\langle 0| = |0\rangle_{A}{}^{A}\langle 0|$ to get

$$
\begin{aligned}
\mathbf{cnot}^{(AB)}\mathbf{cnot}^{(AB)} &= |0\rangle_{A}{}^{A}\langle 0| + |1\rangle_{A}{}^{A}\langle 1|\Big(|0\rangle_{B}{}^{B}\langle 1| + |1\rangle_{B}{}^{B}\langle 0|\Big)\Big(|0\rangle_{B}{}^{B}\langle 1| + |1\rangle_{B}{}^{B}\langle 0|\Big) \\
&= |0\rangle_{A}{}^{A}\langle 0| + |1\rangle_{A}{}^{A}\langle 1|\Big(|0\rangle_{B}{}^{B}\langle 0| + |1\rangle_{B}{}^{B}\langle 1|\Big) \\
&= |0\rangle_{A}{}^{A}\langle 0| + |1\rangle_{A}{}^{A}\langle 1|\mathbf{noop}^{(B)}
\end{aligned}
$$

$$\begin{aligned} &\simeq |0\rangle_{\mathsf{A}}{}^{\mathsf{A}}\langle 0| + |1\rangle_{\mathsf{A}}{}^{\mathsf{A}}\langle 1| \\ &= \mathbf{noop}^{(\mathsf{A})} \\ &\simeq 1, \end{aligned} \tag{32}$$

where we used the fact that when the bra-ket expression for $\mathbf{noop}$ is applied to the ket expression for a state it acts the same as (here denoted by the symbol "$\simeq$") multiplication by the number 1.

2.5 Using Many Quantum Bits

To use more than two, say five, qubits, we can just start with the state $|0\rangle_{\mathsf{A}}|0\rangle_{\mathsf{B}}|0\rangle_{\mathsf{C}}|0\rangle_{\mathsf{D}}|0\rangle_{\mathsf{E}}$ and apply gates to any one or two of these qubits. For example, $\mathbf{cnot}^{(\mathsf{DB})}$ applies the $\mathbf{cnot}$ operation from qubit D to qubit B. Note that the order of D and B in the label for the $\mathbf{cnot}$ operation matters. In the bra-ket notation, we simply multiply the state with the bra-ket form of $\mathbf{cnot}^{(\mathsf{DB})}$ from the left. One can express everything in terms of matrices and vectors, but now the vectors have length $2^5 = 32$, and the Kronecker product expression for $\mathbf{cnot}^{(\mathsf{DB})}$ requires some reordering to enable inserting the operation so as to act on the intended qubits. Nevertheless, to analyze the properties of all reversible (that is, unitary) operations on these qubits, it is helpful to think of the matrices, because a lot of useful properties about unitary matrices are known. One important result from this analysis is that every matrix that represents a reversible operation on quantum states can be expressed as a product of the one- and two-qubit gates introduced so far. We say that this set of gates is "universal".

For general purpose computation, it is necessary to have access to arbitrarily many qubits. Instead of assuming that there are infinitely many from the start, it is convenient to have an operation to add a new qubit, namely, $\mathbf{add}$. To add a new qubit labeled X in the state $|0\rangle_{\mathsf{X}}$, apply $\mathbf{add}^{(\mathsf{X})}$ to the current state. This operation can only be used if there is not already a qubit labeled X. In the bra-ket notation, we implement the $\mathbf{add}^{(\mathsf{X})}$ operation by multiplying the ket expression for the current state by $|0\rangle_{\mathsf{X}}$.

2.6 Qubit Measurements

In order to classically access information about the state of qubits we use the measurement operation $\mathbf{meas}$. This is an intrinsically probabilistic process that can be applied to any extant qubit. For information processing, one can think of $\mathbf{meas}$ as a subroutine or function that returns either 0 or 1 as output. The output is called the "measurement outcome". The probabilities of the measurement outcomes are determined by the current state. The state of the qubit being measured is "collapsed" to the logical state corresponding to the outcome. Suppose we have just one qubit, currently in the state $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$. Measurement of this qubit has the effect

$$\mathbf{meas}\Big(\alpha|0\rangle + \beta|1\rangle\Big) = \begin{cases} 0{:}|0\rangle & \text{with probability } |\alpha|^2, \\ 1{:}|1\rangle & \text{with probability } |\beta|^2. \end{cases} \tag{33}$$

The classical output is given before the new state for each possible outcome. This measurement behavior explains why the amplitudes have to define unit length vectors: Up to a phase, they are associated with square roots of probabilities.

For two qubits the process is more involved. Because of possible correlations between the two qubits, the measurement affects the state of the other one too, similar to conditioning for pbits after one "looks" at one of them. As an example, consider the state

$$|\psi\rangle_{\mathsf{AB}} = \frac{2}{3}|01\rangle_{\mathsf{AB}} + \frac{i2}{3}|10\rangle_{\mathsf{AB}} + \frac{1}{3}|00\rangle_{\mathsf{AB}}. \tag{34}$$

To figure out what happens when we measure qubit A, we first rewrite the current state in the form $\alpha|0\rangle_{\mathsf{A}}|\phi_0\rangle_{\mathsf{B}} + \beta|1\rangle_{\mathsf{A}}|\phi_1\rangle_{\mathsf{B}}$, where $|\phi_0\rangle_{\mathsf{B}}$ and $|\phi_1\rangle_{\mathsf{B}}$ are pure states for qubit B. It is always possible to do that. For the example of Eq. 34:

$$\begin{aligned} |\psi\rangle_{\mathsf{AB}} &= \frac{2}{3}|0\rangle_{\mathsf{A}}|1\rangle_{\mathsf{B}} + \frac{1}{3}|0\rangle_{\mathsf{A}}|0\rangle_{\mathsf{B}} + \frac{i2}{3}|1\rangle_{\mathsf{A}}|0\rangle_{\mathsf{B}} \\ &= |0\rangle_{\mathsf{A}}\Big(\frac{2}{3}|1\rangle_{\mathsf{B}} + \frac{1}{3}|0\rangle_{\mathsf{B}}\Big) + |1\rangle_{\mathsf{A}}\frac{i2}{3}|0\rangle_{\mathsf{B}} \\ &= \frac{\sqrt{5}}{3}|0\rangle_{\mathsf{A}}\Big(\frac{1}{\sqrt{5}}|0\rangle_{\mathsf{B}} + \frac{2}{\sqrt{5}}|1\rangle_{\mathsf{B}}\Big) + \frac{i2}{3}|1\rangle_{\mathsf{A}}\Big(|0\rangle_{\mathsf{B}}\Big), \end{aligned} \tag{35}$$

so $\alpha = \frac{\sqrt{5}}{3}$, $\beta = \frac{i2}{3}$, $|\phi_0\rangle_{\mathsf{B}} = \frac{1}{\sqrt{5}}|0\rangle_{\mathsf{B}} + \frac{2}{\sqrt{5}}|1\rangle_{\mathsf{B}}$ and $|\phi_1\rangle_{\mathsf{B}} = |0\rangle_{\mathsf{B}}$. The last step required pulling out the factor of $\frac{\sqrt{5}}{3}$ to make sure that $|\phi_0\rangle_{\mathsf{B}}$ is properly normalized for a pure state. Now that we have rewritten the state, the effect of measuring qubit A can be given as follows:

$$\mathbf{meas}^{(\mathsf{A})}\Big(\alpha|0\rangle_{\mathsf{A}}|\phi_0\rangle_{\mathsf{B}} + \beta|1\rangle_{\mathsf{A}}|\phi_1\rangle_{\mathsf{B}}\Big) = \begin{cases} 0{:}|0\rangle_{\mathsf{A}}|\phi_0\rangle_{\mathsf{B}} & \text{with probability } |\alpha|^2, \\ 1{:}|1\rangle_{\mathsf{A}}|\phi_1\rangle_{\mathsf{B}} & \text{with probability } |\beta|^2. \end{cases} \tag{36}$$

For the example, the measurement outcome is 0 with probability $\frac{5}{9}$, in which case the state collapses to $|0\rangle_{\mathsf{A}}\Big(\frac{1}{\sqrt{5}}|0\rangle_{\mathsf{B}} + \frac{2}{\sqrt{5}}|1\rangle_{\mathsf{B}}\Big)$. The outcome is 1 with probability $\frac{4}{9}$, in which case the state collapses to $|1\rangle_{\mathsf{A}}|0\rangle_{\mathsf{B}}$. The probabilities add up to 1 as they should.

The same procedure works for figuring out the effect of measuring one of any number of qubits. Say we want to measure qubit B among qubits $\mathsf{A}, \mathsf{B}, \mathsf{C}, \mathsf{D}$, currently in state $|\psi\rangle_{\mathsf{ABCD}}$. First rewrite the state in the form $\alpha|0\rangle_{\mathsf{B}}|\phi_0\rangle_{\mathsf{ACD}} + \beta|1\rangle_{\mathsf{B}}|\phi_1\rangle_{\mathsf{ACD}}$, making sure that the ACD superpositions are pure states. Then the outcome of the measurement is 0 with probability $|\alpha|^2$ and 1 with probability $|\beta|^2$. The collapsed states are $|0\rangle_{\mathsf{B}}|\phi_0\rangle_{\mathsf{ACD}}$ and $|1\rangle_{\mathsf{B}}|\phi_1\rangle_{\mathsf{ACD}}$, respectively.

Probabilities of the measurement outcomes and the new states can be calculated systematically. For example, to compute the probability and state for outcome 0 of $\mathbf{meas}^{(\mathsf{A})}$ given the state $|\psi\rangle_{\mathsf{AB}}$, one can first obtain the unnormalized ket expression $|\phi_0'\rangle_{\mathsf{B}} = {}^{\mathsf{A}}\langle 0||\psi\rangle_{\mathsf{AB}}$ by using the rules for multiplying kets by bras. The probability is given by $p_0 = {}^{\mathsf{B}}\langle \phi_0'|\phi_0'\rangle_{\mathsf{B}}$, and the collapsed, properly normalized pure state is

$$|0\rangle_{\mathsf{A}}|\phi_0'\rangle_{\mathsf{B}}/\sqrt{p_0} = |0\rangle_{\mathsf{A}}{}^{\mathsf{A}}\langle 0||\psi\rangle_{\mathsf{AB}}/\sqrt{p_0}, \tag{37}$$

The operator $P_{0} = |0\rangle_{\mathsf{A}}{}^{\mathsf{A}}\langle 0|$ is called a "projection operator" or "projector" for short. If we perform the same computation for the outcome 1, we find the projector $P_{1} = |1\rangle_{\mathsf{A}}{}^{\mathsf{A}}\langle 1|$. The two operators satisfy ${P_{\mathsf{a}}}^2 = P_{\mathsf{a}}$,

${P_{\mathfrak{a}}}^\dagger = P_{\mathfrak{a}}$ and $P_{\mathfrak{o}} + P_{\mathfrak{1}} = \mathbb{1}$. In terms of the projectors, the measurement's effect can be written as follows:

$$\mathbf{meas}^{(\mathsf{A})}|\psi\rangle_{\mathsf{AB}} = \begin{cases} \mathfrak{o}{:}P_{\mathfrak{o}}|\psi\rangle_{\mathsf{AB}}/\sqrt{p_0} & \text{with probability } p_0, \\ \\ \mathfrak{1}{:}P_{\mathfrak{1}}|\psi\rangle_{\mathsf{AB}}/\sqrt{p_1} & \text{with probability } p_1, \end{cases} \tag{38}$$

where $p_0 = {}^{\mathsf{AB}}\langle\psi|P_{\mathfrak{o}}|\psi\rangle_{\mathsf{AB}}$ and $p_1 = {}^{\mathsf{AB}}\langle\psi|P_{\mathfrak{1}}|\psi\rangle_{\mathsf{AB}}$. In quantum mechanics, any pair of projectors satisfying the properties given above is associated with a potential measurement whose effect can be written in the same form. This is called a binary "von Neumann", or "projective", measurement.

2.7 Mixtures and Density Operators

The measurement operation "reads out" information from qubits to pbits. What if we discard the pbit that contains the measurement outcome? The result is that the qubits are in a probabilistic "mixture" of two pure states. Such mixtures are a generalization of pure states. The obvious way to think about a mixture is that we have a probability distribution over pure quantum states. For example, after discarding the pbit and qubit A in Eq. 36, we can write the state of B as $\rho = \{|\alpha|^2{:}|\phi_0\rangle_{\mathsf{B}}\ ,\ |\beta|^2{:}|\phi_1\rangle_{\mathsf{B}}\}$, using the notation for probability distributions introduced earlier.

Mixtures frequenty form when using irreversible operations such as measurement. Except for measurement, the quantum gates that we have introduced so far are reversible and therefore transform pure states to pure states, so that no mixtures can be formed. One of the fundamental results of reversible classical and quantum computation is that there is no loss in power in using only reversible gates. Specifically, it is possible to change a computation that includes irreversible operations to one that accomplishes the same goal, has only reversible operations and is efficient in the sense that it uses at most polynomial additional resources. However, the cost of using only reversible operations is not negligible. In particular, for ease of programming, and more importantly, when performing repetitive error-correction tasks (see [12]), the inability to discard or reset qubits can be very inconvenient. We therefore introduce additional operations that enable resetting and discarding.

Although resetting has a so-called "thermodynamic" cost (think of the heat generated by a computer), it is actually a simple operation. The $\mathbf{reset}$ operation applied to qubit A can be thought of as the result of first measuring A, then flipping A if the measurement outcome is $|\mathfrak{1}\rangle$, and finally discarding the measurement result. Using the notation of Eq. 36, the effect on a pure state $|\psi\rangle_{\mathsf{AB}}$ is given by:

$$\mathbf{reset}^{(\mathsf{A})}|\psi\rangle_{\mathsf{AB}} = \{|\alpha|^2{:}|\mathfrak{o}\rangle_{\mathsf{A}}|\phi_0\rangle_{\mathsf{B}}\ ,\ |\beta|^2{:}|\mathfrak{o}\rangle_{\mathsf{A}}|\phi_1\rangle_{\mathsf{B}}\}. \tag{39}$$

To apply $\mathbf{reset}$ to an arbitrary probability distribution, you apply it to each of the distribution's pure states and combine the results to form an expanded probability distribution. The $\mathbf{discard}^{(\mathsf{A})}$ operation is $\mathbf{reset}^{(\mathsf{A})}$ followed by discarding qubit A. Therefore, in the expression for the state after $\mathbf{reset}^{(\mathsf{A})}$, all the $|\mathfrak{o}\rangle_{\mathsf{A}}$ are removed. It is an important fact that every physically realizable quantum operation, whether reversible or not, can be expressed as a combination of $\mathbf{add}$ operations, gates from the universal set and $\mathbf{discard}$ operations.

The representation of mixtures using probability distributions over pure states is redundant. That is, there are many probability distributions that are physically indistinguishable. A non-redundant description of a quantum state can be obtained by using “density operators”. The density operator for the mixture ρ given in Eq. 39 is given by

$$\hat{\rho} = |\alpha|^2 |\phi_0\rangle_{\mathsf{B}}{}^{\mathsf{B}}\langle\phi_0| + |\beta|^2 |\phi_1\rangle_{\mathsf{B}}{}^{\mathsf{B}}\langle\phi_1|. \tag{40}$$

The general rule for calculating the density operator from a probability distribution is as follows: For each pure state $|\phi\rangle$ in the distribution, calculate the operators $|\phi\rangle\langle\phi|$ and sum them weighted by their probabilities.

There is a way to apply gates to the density operators defined by states. If the gate acts by the unitary operator U, then the effect of applying it to $\hat{\rho}$ is given by $U\hat{\rho}U^\dagger$, where $U^\dagger$ is the conjugate transpose of U. (In the bra-ket expression for U, $U^\dagger$ is obtained by replacing all complex numbers by their conjugates, and terms such as $|\phi\rangle\langle\varphi|$ by $|\varphi\rangle\langle\phi|$.)

The relationship between a qubit’s state space and a sphere can be explained in terms of the qubit’s density operators. In matrix form, this operator is a 2×2 matrix, which can be written uniquely as a sum $(\mathbb{1} + x\sigma_x + y\sigma_y + z\sigma_z)/2$. One can check that if the density operator $|\psi\rangle\langle\psi|$ for a qubit’s pure state is written as such a sum,

$$|\psi\rangle\langle\psi| = (\mathbb{1} + x\sigma_x + y\sigma_y + z\sigma_z)/2, \tag{41}$$

then the vector (x, y, z) thus obtained is on the surface of the unit sphere in three dimensions. In fact, for every vector (x, y, z) on the unit sphere, there is a unique pure state satisfying Eq. 41. Since the density operators for mixtures are arbitrary, convex (that is probabilistic) sums of pure states, the set of (x, y, z) thus obtained for mixtures fills out the unit ball. The rotations introduced earlier modify the vector (x, y, z) in the expected way, by rotation of the vector around the appropriate axis. See [11] for more details.

2.8 Quantum Computation

The model of computation defined by the one- and two-qubit gates and the operations of adding ($\mathbf{add}$), measuring ($\mathbf{meas}$) and discarding ($\mathbf{discard}$) qubits is called the “quantum network model”. A sequence of instructions for applying these operations is called a “quantum network”. Quantum computation extends the network model by providing a means for repeating blocks of instructions. Such means can be specified by a formal machine model of computation. There are several such models of classical and quantum computers. One of the best known is the Turing machine, which has a quantum analogue, the quantum Turing machine. This model has its uses for formal studies of computation and complexity, but is difficult to program. Fortunately, as mentioned in Sect. 1, there is no loss of computational power if the means for repeating instructions is provided by a classical computer that can apply gates and other operations to qubits. A general quantum algorithm is a program written for such a computer.

There are three practical methods that can be used to write quantum networks and algorithms. The first is to use the names for the different operations and algebraically multiply them. The second is to draw quantum networks, which are pictorial representations of the sequence of steps in time, somewhat like flowcharts without loops. The third is to use a generic programming language enhanced with statements for accessing and modifying quantum bits. The first two methods work well as long as the sequence is short and we do not use many operations that depend on measurement outcomes or require loops. They are often used to describe subroutines of longer algorithms presented either in words or by use of the third method.

To see how to use the different methods and also to illustrate the power of quantum computation, we work out a short quantum algorithm that solves the following problem:

The Parity Problem: Given is a "black box" quantum operation $\mathbf{BB}^{(\mathsf{ABC})}$ that has the following effect when applied to a logical basis state:

$$\mathbf{BB}^{(\mathsf{ABC})}|a_{\mathsf{A}}a_{\mathsf{B}}\rangle_{\mathsf{AB}}|a_{\mathsf{C}}\rangle_{\mathsf{C}} = |a_{\mathsf{A}}a_{\mathsf{B}}\rangle_{\mathsf{AB}}|a_{\mathsf{C}} \oplus (b_{\mathsf{A}}a_{\mathsf{A}} \oplus b_{\mathsf{B}}a_{\mathsf{B}})\rangle_{\mathsf{C}}, \tag{42}$$

where b_{A} and b_{B} are 0 or 1. The actual values of b_{A} and b_{B} are unknown. The problem is to determine what b_{A} and b_{B} are by using the black box only once.

The terminology and the definition of the operation $\mathbf{BB}^{(\mathsf{ABC})}$ require explanation. In computation, we say that an operation is a black box or an "oracle" if we have no access whatsoever to how the operation is implemented. In a black box problem, we are promised that the black box implements an operation from a specified set of operations. In the case of the parity problem, we know that the operation is to add one of four possible parities (see below). The problem is to determine which parity is added by using the black box in a network. Black box problems serve many purposes. One is to study the differences between models of computation, just as we are about to do. In fact, black box problems played a crucial role in the development of quantum algorithms by providing the first and most convincing examples of the power of quantum computers [13, 14]. Some of these examples involve generalizations of the parity problem. Another purpose of black box problems is to enable us to focus on what can be learned from the "input/output" behavior of an operation without having to analyze its implementation. This is useful because in many cases of interest, it is very difficult to exploit knowledge of the implementation to determine a desirable property of the operation. A classical example is the well-known satisfiability problem, in which we are given a classical circuit with one output bit and we need to determine whether there is an input for which the output is 1. Instead of trying to analyze the circuit, one can look for and use a general purpose black-box search algorithm to find the "satisfying" input.

In the definition of the effect of $\mathbf{BB}^{(\mathsf{ABC})}$, the operation "$\oplus$" is addition modulo 2, so $1 \oplus 1 = 0$, and all the other sums are as expected. As the state symbols now have a numeric meaning, we use the number font for states. To see what $\mathbf{BB}$ does, suppose that b_{A} and b_{B} are both 1. Then $\mathbf{BB}$ adds (modulo 2) the parity of the logical state in AB to the logical state of C. The parity of a logical state is 0 if the number of 1's is even and 1 if it is odd. The action of $\mathbf{BB}$ for this example is given by:

$$\begin{aligned}
\mathbf{BB}^{(\mathsf{ABC})}|00\rangle_{\mathsf{AB}}|0\rangle_{\mathsf{C}} &= |00\rangle_{\mathsf{AB}}|0\rangle_{\mathsf{C}} \\
\mathbf{BB}^{(\mathsf{ABC})}|01\rangle_{\mathsf{AB}}|0\rangle_{\mathsf{C}} &= |01\rangle_{\mathsf{AB}}|0 \oplus 1\rangle_{\mathsf{C}} \\
&= |01\rangle_{\mathsf{AB}}|1\rangle_{\mathsf{C}} \\
\mathbf{BB}^{(\mathsf{ABC})}|10\rangle_{\mathsf{AB}}|1\rangle_{\mathsf{C}} &= |10\rangle_{\mathsf{AB}}|1 \oplus 1\rangle_{\mathsf{C}} \\
&= |10\rangle_{\mathsf{AB}}|0\rangle_{\mathsf{C}} \\
\mathbf{BB}^{(\mathsf{ABC})}|11\rangle_{\mathsf{AB}}|0\rangle_{\mathsf{C}} &= |11\rangle_{\mathsf{AB}}|0\rangle_{\mathsf{C}}
\end{aligned} \tag{43}$$

The action of the black box is extended to superpositions by "linear extension". This means that to apply $\mathbf{BB}$ to a superposition of the logical states, simply apply it to each logical summand and add the results. Different values of b_{A} and b_{B} correspond to different parities. For example, if $b_{\mathsf{A}} = 1$ and $b_{\mathsf{B}} = 0$, then the parity of the state in A is added to the state in C. In this sense, what is added is the parity of a subset of the two qubits AB. Thus, one way of thinking about the problem is that we wish to find out which subset's parity the black box is using.

We can give an algorithm that solves the parity problem using each of the three methods for describing quantum networks. Here is an algebraic description of a solution, $\mathbf{qparity}^{(\mathsf{ABC})}$, given as a product of quantum gates that involves one use of the black box. We defer the explanation of why this solution works until after we show how to represent the algorithm pictorially using quantum networks.

$$\mathbf{qparity}^{(\mathsf{ABC})} = \mathbf{meas}^{(\mathsf{B})}\mathbf{H}^{(\mathsf{B})}\mathbf{meas}^{(\mathsf{A})}\mathbf{H}^{(\mathsf{A})}\mathbf{BB}^{(\mathsf{ABC})}\mathbf{H}^{(\mathsf{C})}\mathbf{not}^{(\mathsf{C})}\mathbf{add}^{(\mathsf{C})}\mathbf{H}^{(\mathsf{B})}\mathbf{add}^{(\mathsf{B})}\mathbf{H}^{(\mathsf{A})}\mathbf{add}^{(\mathsf{A})}. \tag{44}$$

The output of the algorithm is given by the classical outputs of the measurements of qubit A, which yields b_{A}, and qubit B, which yields b_{B}. As is conventional, in writing products of linear operators, the order of application in Eq. 44 is right to left, as in a product of matrices applied to a column vector. This order of terms in a product is, however, counterintuitive, particularly for operations to be performed one after the other. It is therefore convenient to use left to right notation, as is done in describing laser or radio-frequency pulse sequences. One way to make it clear that left to right order is used involves putting dots between gates as in the following version of Eq. 44:

$$\mathbf{qparity}^{(\mathsf{ABC})} = \mathbf{add}^{(\mathsf{A})}.\mathbf{H}^{(\mathsf{A})}.\mathbf{add}^{(\mathsf{B})}.\mathbf{H}^{(\mathsf{B})}.\mathbf{add}^{(\mathsf{C})}.\mathbf{not}^{(\mathsf{C})}.\mathbf{H}^{(\mathsf{C})}.\mathbf{BB}^{(\mathsf{ABC})}.\mathbf{H}^{(\mathsf{A})}.\mathbf{meas}^{(\mathsf{A})}.\mathbf{H}^{(\mathsf{B})}.\mathbf{meas}^{(\mathsf{B})}. \tag{45}$$

In this representation, the first operation is $\mathbf{add}^{(\mathsf{A})}$, the second is $\mathbf{H}^{(\mathsf{A})}$ (the Hadamard gate on qubit A) and so on.

The algebraic specification of the algorithm as products of gates does not make it easy to see why the algorithm works. It is also difficult to see which operations depend on each other. Such dependencies are used to determine whether the operations can be "parallelized". Quantum networks make these tasks simpler. The quantum network for the above sequence is shown in Fig. 2.

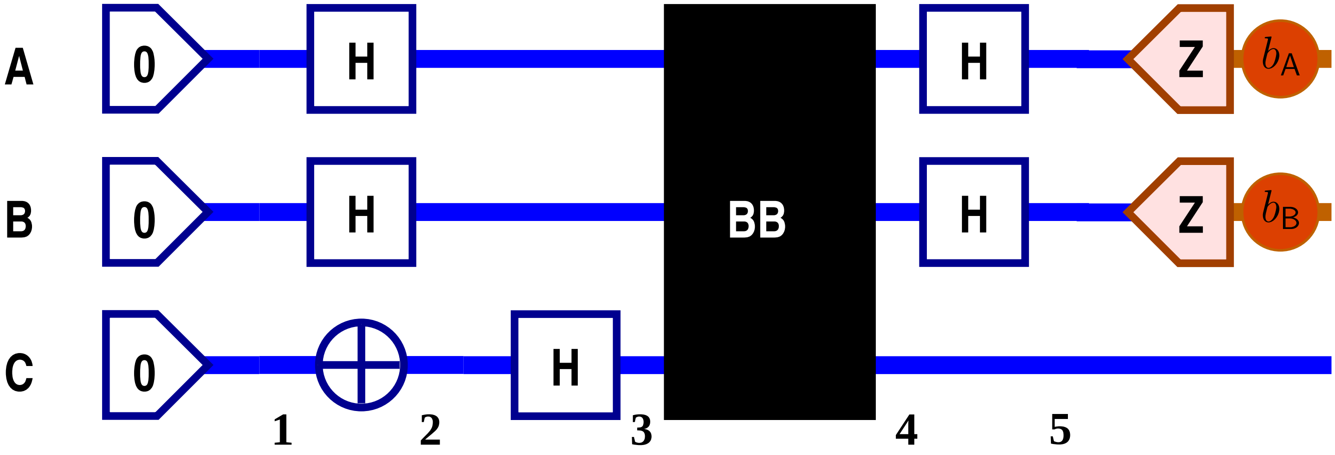


FIG. 2: Quantum network for solving the parity problem. A quantum network has a (horizontal in this case) line for each qubit. The line can be thought of as the time-line for the qubit and is shown in blue. Each gate is drawn as a box, circle, or other element intercepting the lines of the qubits it acts on. In this case, time runs from left to right. Each qubit's time-line starts at the point where it is added. In this example, the qubits' time-lines end when they are measured, at which point a classical bit (brown time line) containing the measurement outcome is introduced. The operation $\mathbf{BB}$ is illustrated as a black box. The numbers underneath the network refer to checkpoints used to explain how the network solves the parity problem.

To understand how the quantum network of Fig 2 solves the parity problem, we can follow the states as the network is "executed" from left to right, using the indicated checkpoints. Using vector notation for the states, at checkpoint **1** the state is

$$|\psi\rangle_1 = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix} \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \tag{46}$$

where we used Kronecker product notation to denote the states of A, B and C, in this order. In the next time step, the network involves applying Hadamard gates (Eq. 13) to A and B and a $\mathbf{not}$ gate (Eq. 9) to C. At checkpoint **2**, this operation results in the state

$$|\psi\rangle_2 = \begin{pmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \end{pmatrix} \otimes \begin{pmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \end{pmatrix} \otimes \begin{pmatrix} 0 \\ 1 \end{pmatrix}. \tag{47}$$

Next, a Hadamard gate is applied to C, so that at checkpoint **3** we have,

$$|\psi\rangle_3 = \begin{pmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \end{pmatrix} \otimes \begin{pmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \end{pmatrix} \otimes \begin{pmatrix} 1/\sqrt{2} \\ -1/\sqrt{2} \end{pmatrix}. \tag{48}$$

The next event involves applying the black box. To understand what happens, note that the effect of the black box can be described as "conditional on each logical state of AB, if the parity according to b_{A} and b_{B} is 1, then apply $\mathbf{not}$ to C" The current state of C is such that if $\mathbf{not}$ is applied, only the sign changes:

$$\begin{aligned} \mathbf{not}\begin{pmatrix} 1/\sqrt{2} \\ -1/\sqrt{2} \end{pmatrix} &= \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}\begin{pmatrix} 1/\sqrt{2} \\ -1/\sqrt{2} \end{pmatrix} \\ &= -\begin{pmatrix} 1/\sqrt{2} \\ -1/\sqrt{2} \end{pmatrix}. \end{aligned} \tag{49}$$

Now AB is in a superposition of each of the logical states, and conditional on the logical state and the (hidden) parity, the sign changes. As a result, although the state of C does not change, a phase is "kicked back" to AB. A generalization of this effect is at the heart of A. Kitaev's version of P. Shor's quantum factoring algorithm (Sect. 2.10). At the next checkpoint, and after some arithmetic to check which logical states change sign, we can write the state as

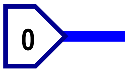
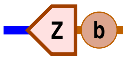
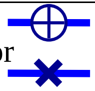
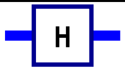
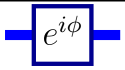
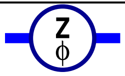
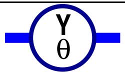
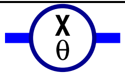
$$|\psi\rangle_4 = \begin{pmatrix} 1/\sqrt{2} \\ (-1)^{b_{\mathsf{A}}}/\sqrt{2} \end{pmatrix} \otimes \begin{pmatrix} 1/\sqrt{2} \\ (-1)^{b_{\mathsf{B}}}/\sqrt{2} \end{pmatrix} \otimes \begin{pmatrix} 1/\sqrt{2} \\ -1/\sqrt{2} \end{pmatrix}. \tag{50}$$

Notice that qubits A and B are in orthogonal states for different values of $b_{\mathsf{A}}, b_{\mathsf{B}}$. It suffices to apply the Hadamard transform again to A and B to get

$$|\psi\rangle_4 = \begin{pmatrix} 1-b_{\mathsf{A}} \\ b_{\mathsf{A}} \end{pmatrix} \otimes \begin{pmatrix} 1-b_{\mathsf{B}} \\ b_{\mathsf{B}} \end{pmatrix} \otimes \begin{pmatrix} 1/\sqrt{2} \\ -1/\sqrt{2} \end{pmatrix}. \tag{51}$$

Measurements of A and B now reveal the previously unknown b_{A} and b_{B}.

As can be seen, the visual representation of a quantum network eases the tasks of following what happens. This is why it is used extensively for presenting basic subroutines and algorithms in quantum computation. A guide to the commonly used network elements is given in Fig. 3.

Name	Gate	Symbols	Algebraic	Matrix
Add/prepare		**add**	If applied to existing qubit: $\{|0\rangle\langle 0|, |0\rangle\langle 1|\}$ (operator mixture)	$\begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}, \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}$
Measure		**meas**	$\{0{:}|0\rangle\langle 0|, 1{:}|1\rangle\langle 1|\}$	$\begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}, \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}$
Not	or 	**not**, σ_x	$|0\rangle\langle 1| + |1\rangle\langle 0|$	$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$
Hadamard		**H**	$e^{-i\sigma_y\pi/4}\sigma_z$	$\frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$
Phase change		$\mathbf{S}(e^{i\phi})$	$e^{i\phi/2}e^{-i\sigma_z\phi/2}$	$\begin{pmatrix} 1 & 0 \\ 0 & e^{i\phi} \end{pmatrix}$
z-Rotation		$\mathbf{Z}_\phi$	$e^{-i\sigma_z\phi/2}$	$\begin{pmatrix} e^{-i\phi/2} & 0 \\ 0 & e^{i\phi/2} \end{pmatrix}$
y-Rotation		$\mathbf{Y}_\theta$	$e^{-i\sigma_y\theta/2}$	$\begin{pmatrix} \cos(\theta/2) & -\sin(\theta/2) \\ \sin(\theta/2) & \cos(\theta/2) \end{pmatrix}$
x-Rotation		$\mathbf{X}_\theta$	$e^{-i\sigma_x\theta/2}$	$\begin{pmatrix} \cos(\theta/2) & -i\sin(\theta/2) \\ -i\sin(\theta/2) & \cos(\theta/2) \end{pmatrix}$

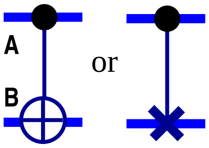
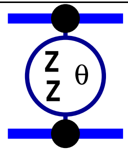
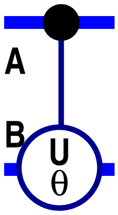
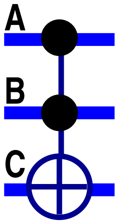
Controlled not		**cnot**	$\vert 0\rangle_{A}{}^{A}\langle 0\vert + \vert 1\rangle_{A}{}^{A}\langle 1\vert\sigma_x{}^{(B)}$ $e^{-i\sigma_z{}^{(A)}\pi/4}e^{-i\frac{1}{2}(\mathbb{1}-\sigma_z{}^{(A)})\sigma_x{}^{(B)}\pi/2}$	$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$
ZZ rotation		$(\mathbf{ZZ})_\theta$	$e^{-i\sigma_z{}^{(A)}\sigma_z{}^{(B)}\theta/2}.$	$\begin{pmatrix} e^{-i\theta/2} & 0 & 0 & 0 \\ 0 & e^{i\theta/2} & 0 & 0 \\ 0 & 0 & e^{i\theta/2} & 0 \\ 0 & 0 & 0 & e^{-i\theta/2} \end{pmatrix}$
Controlled rotation		$\mathbf{cU}_\theta$	$\vert 0\rangle_{A}{}^{A}\langle 0\vert + \vert 1\rangle_{A}{}^{A}\langle 1\vert e^{-i\sigma_U{}^{(B)}\theta/2}$	$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \multicolumn{2}{c}{\multirow{2}{*}{$e^{-i\sigma_U\theta/2}$}} \\ 0 & 0 & & \end{pmatrix}$
Toffoli gate		$\mathbf{c^2not}$	$\mathbb{1} - \vert 11\rangle_{AB}{}^{AB}\langle 11\vert + \vert 11\rangle_{AB}{}^{AB}\langle 11\vert\sigma_x{}^{(C)}$	

FIG. 3: Quantum network elements.

When designing or describing complicated algorithms for quantum computers, providing everything in terms of quantum networks can become difficult, particularly when an important part of the algorithm consists of computations that are best done on a classical computer. For example, a full description of Shor's algorithm for factoring whole numbers (see Sect. 2.10) includes a significant amount of classical preprocessing, which determines choices made in the quantum algorithm, and classical postprocessing, which computes a factor from the measured result by a continued fraction algorithm. For such algorithms, one can use a programming language similar to Pascal, BASIC or C enhanced with statements to access quantum bits and to apply quantum operations. For algorithm design, computer scientists often use a semi-formal language called "pseudocode" [8]. With a simple extension called "quantum pseudocode", the algorithm for the parity problem can be written as follows:

BBPARITY($\mathbf{BB}$)
Input: Access to a quantum black box $\mathbf{BB}$ that acts on three qubits by adding a parity function of the first two qubits to the third.
Output: The two bits b_A and b_B of the parity function.

```
    foreach $i \in \{A, B, C\}$
```

```
        $\ulcorner a_i \urcorner \leftarrow |0\rangle$
            C: Initialize three one-qubit registers $\ulcorner a_i \urcorner$, $i = \mathsf{A}, \mathsf{B}, \mathsf{C}$. The corner bracket annotation declares $a_i$ as a quantum register.
    end
    $\ulcorner a_C \urcorner \leftarrow \sigma_x \ulcorner a_C \urcorner$
    foreach $i \in \{\mathsf{A}, \mathsf{B}, \mathsf{C}\}$
        $\ulcorner a_i \urcorner \leftarrow \mathbf{H} \ulcorner a_i \urcorner$
    end
    $\ulcorner a \urcorner \leftarrow \mathbf{BB} \ulcorner a \urcorner$
        C: $\ulcorner a \urcorner$ refers to the three qubit register consisting of the $\ulcorner a_i \urcorner$
    foreach $i \in \{\mathsf{A}, \mathsf{B}\}$
        $\ulcorner a_i \urcorner \leftarrow \mathbf{H} \ulcorner a_i \urcorner$
        $b_i \leftarrow \mathbf{meas} \ulcorner a_i \urcorner$
    end
    return $b_\mathsf{A}, b_\mathsf{B}$
end
```

Any classical programming language can be extended with statements to access and manipulate quantum registers.

Now that we have looked at the quantum solution of the parity problem, let us consider the question of the least number of black-box applications required by a classical algorithm: Each classical use of the black box can only give us one bit of information. In particular, one use of the black box with input $a_\mathsf{A}a_\mathsf{B}$ reveals only the parity of $a_\mathsf{A}a_\mathsf{B}$ according to the hidden parameters b_A and b_B. Each use of the black box can therefore only help us distinguish between two subsets of the four possible parities. At least two uses of the black box are therefore necessary. Two uses are also sufficient: To determine which of the four parities is involved, use the black box first with input $a_\mathsf{A}a_\mathsf{B} = 10$ and then with input $a_\mathsf{A}a_\mathsf{B} = 01$. As a result of this argument, one can consider the parity problem as a simple example of a case in which there is a more efficient quantum algorithm than is possible classically. However, it is worth noting that the comparison is not entirely fair: A truly classical oracle answering parity questions or implementing the black box on the states of classical bits is useless to a quantum algorithm. To take advantage of such an algorithm it must be possible to use superpositions that are not implicitly collapsed. Collapse can happen if the oracle makes a measurement or otherwise "remembers" the question that it was asked.

2.9 Resource Accounting

When trying to solve a problem using quantum information processing, an important issue is to determine what physical resources are available and how much of each resource is needed for the solution. As mentioned before, in classical information, the primary resources are bits and operations. The number of bits used by an algorithm is called its "space" requirement. The number of operations used is called its "time" requirement. If parallel computation is available, one can distinguish between the total number of operations ("work") and the number of parallel steps ("time").

When quantum information processing is used, the classical resources are still relevant for running the computer that controls the quantum system and performs any pre- and post-processing tasks. The main quantum resources are analogous to the classical ones: “quantum space” is the number of qubits needed, and “quantum time” the number of quantum gates. Because it turns out that reset operations have a thermodynamic cost, one can count irreversible quantum operations separately. This accounting of the resource requirements of algorithms and of the minimum resources needed to solve problems forms the foundations of quantum complexity theory.

As a simple example of resource accounting, consider the algorithm for the parity problem. No classical computation is required to decide which quantum gates to apply, or to determine the answer from the measurement. The quantum network consists of a total of 11 quantum gates (including the $\mathbf{add}$’s and $\mathbf{meas}$’s operations) and one oracle call (the application of the black box). In the case of oracle problems, one usually counts the number of oracle calls first, as we have done in discussing the algorithm. The network is readily parallelized to reduce the time resource to 6 steps.

2.10 From Factoring to Phase Estimation

The publication of Shor’s quantum algorithm for efficiently factoring numbers [4, 5] was the key event that stimulated many theoretical and experimental investigations of quantum computation. One of the reasons why this algorithm is so important is that the security of widely used public key cryptographic protocols relies on the conjectured difficulty of factoring large numbers. An elementary overview of these protocols and the quantum algorithm for breaking them is in [15]. Here, we outline the relationship between factoring and the powerful technique of phase estimation. This relationship helps in understanding many of the existing quantum algorithms and was first explained in [16], motivated by Kitaev’s version [17] of the factoring algorithm.

The factoring problem requires writing a whole number N as a product of primes. (Primes are whole numbers greater than 1 that are divisible without remainder only by 1 and themselves.) Shor’s algorithm solves this problem by reducing it to instances of the order-finding problem, which will be defined below. The reduction is based on basic number theory and involves efficient classical computation. At the core of Shor’s algorithm is a quantum algorithm that solves the order-finding problem efficiently. In this case, an algorithm is considered efficient if it uses resources bounded by a polynomial in the number of digits of N. For more information on the requisite number theory, see any textbook on number theory [18, 19].

We begin by showing that factoring reduces to order finding. The first observation is that to factor a whole number it is sufficient to solve the factor-finding problem, whose statement is: Given a whole number N find a proper factor of N, if one exists. A “factor” of N is a whole number f that satisfies $N = fg$ for some whole number g. The factor f is “proper” if $f \neq 1$ and $f \neq N$. For example, if $N = 15$, then 3 and 5 are its proper factors. For some numbers it is easy to find a proper factor. For example, you can tell that N is even from the least significant digit (in decimal or binary), in which case 2 is a proper factor (unless $N = 2$, a prime). But many numbers are not so easy. As an example, you can try to find a proper factor of $N = 149573$ by hand[1]. You can complete the factorization of a whole number by recursively applying an algorithm for the factor-finding problem to all the proper factors found.

[1] 149573=373*401

Before we continue the reduction of factoring to order finding, we briefly explain modular arithmetic, which both simplifies the discussion and is necessary to avoid computing with numbers that have exponential numbers of digits. We say that a and b are "equal modulo N", written as $a = b \bmod N$, if $a - b$ is divisible by N (without remainder). For example, $3 = 18 \bmod 15 = 33 \bmod 15$. Equality modulo N is well-behaved with respect to addition and multiplication. That is, if $a = b \bmod N$ and $c = d \bmod N$, then $a + c = b + d \bmod N$ and $ac = bd \bmod N$. For factoring N, we will be looking for whole numbers a that are divisible by a proper factor of N. If a has this property, then so does any b with $b = a \bmod N$. We therefore perform all arithmetic "modulo N". One way to think about this is that we only use whole numbers a that satisfy $0 \leq a \leq N - 1$. We can implement an arithmetic operation modulo N by first applying the operation in the usual way and then computing the remainder after division by N. For example, to obtain $ab \bmod N$, we first compute ab. The unique c such that $0 \leq c \leq N - 1$ and $c = ab \bmod N$ is the remainder after division of ab by N. Thus c is the result of multiplying a by b modulo N. Consistent with this procedure, we can think of the expression $a \bmod N$ as referring to the remainder of a after division by N.

The second observation in the reduction of factoring to order finding is that it is sufficient to find a whole number r with the property that $r^2 - 1$ is a multiple of N but $r - 1$ and $r + 1$ are not. Using the language of modular arithmetic, the property is expressed as $r^2 = 1 \bmod N$ but $r \neq 1 \bmod N$ and $r \neq -1 \bmod N$. Because $1 \bmod N$ and $-1 \bmod N$ are the obvious square roots of $1 \bmod N$, we say that r is a "non-trivial square root of unity" (modulo N). For such an r, one can write $r^2 - 1 = (r-1)(r+1) = mN$ for some whole number m. This implies that every prime factor p of N divides either $(r - 1)$ or $(r + 1)$ so that either $(r - 1)$ or $(r + 1)$ is or shares a factor with N. Suppose that $r - 1$ is or shares such a factor. Because $r - 1$ is not a multiple of N, the greatest common divisor of $r - 1$ and N is a proper factor of N. Since there exists an efficient classical algorithm (the "Euclidean algorithm") for finding the greatest common divisor, we can easily find the desired proper factor.

The examples of $N = 15$ and $N = 21$ serve to illustrate the key features of the algorithm. For $N = 15$, possible choices for r are $r = 4\ \ (4^2 - 1 = 1 * 15)$ and $r = 11\ \ (11^2 - 1 = 120 = 8 * 15)$. For the first choice, the proper factors emerge immediately: $4 - 1 = 3, 4 + 1 = 5$. For the second, it is necessary to determine the greatest common divisors. Let gcd(x, y) stand for the greatest common divisor of x and y. The proper factors are gcd$(11 - 1, 15)$ = gcd$(10, 15) = 5$ and gcd$(11 + 1, 15)$ = gcd$(12, 15) = 3$. For $N = 21$, one can take $r = 8$, as $8^2 - 1 = 63 = 3 * 21$. In this case, $8 - 1 = 7$ is a proper factor and gcd$(8 + 1, 21) = 3$ is another.

For N even or a power of a prime it is not always possible to find a non-trivial square root of unity. Because both of these cases can be handled efficiently by known classical algorithms, we can exclude them. In every other case, such numbers r exist. One way to find such an r is to start from any whole number q with $1 < q < N$. If gcd$(q, N) = 1$, then according to a basic result in number theory there is a smallest whole number $k > 1$ such that $q^k - 1 = 0 \bmod N$. The number k is called the "order" of q modulo N. If k is even, say $k = 2\,l$, then $(q^l)^2 = 1 \bmod N$, so q^l is a (possibly trivial) square root of unity. For the example of $N = 15$, we can try $q = 2$. The order of 2 modulo 15 is 4, which gives $r = 2^2 = 4$, the first of the two choices in the previous paragraph. For $N = 21$, again with $q = 2$, the order is 6: $2^6 - 1 = 63 = 3 * 21$. Thus, $r = 2^3 = 8$. We can also try $q = 11$, in which case with foresight it turns out that $11^6 - 1$ is divisible by 21. A possible problem appears, namely, the powers q^k that we want to compute are extremely large. But modular arithmetic can be used to avoid this problem. For example, to find the order of 11 modulo 21 by a direct search, we can perform the following computation:

$$\begin{array}{rcrcrcrcrcr}
11^2 &=& 121 &=& 5*21+16 &=& 16 \bmod 21 & & & & \\
11^3 &=& 11*11^2 &=& & & 11*16 \bmod 21 &=& 11*(-5) \bmod 21 & & \\
 & & &=& & & -55 \bmod 21 &=& -3*21+8 \bmod 21 &=& 8 \bmod 21 \\
11^4 &=& 11*11^3 &=& & & 11*8 \bmod 21 &=& 4*21+4 \bmod 21 &=& 4 \bmod 21 \\
11^5 &=& 11*11^4 &=& & & 11*4 \bmod 21 &=& 2 \bmod 21 & & \\
11^6 &=& 11*11^5 &=& & & 11*2 \bmod 21 &=& 1 \bmod 21 & &
\end{array} \tag{52}$$

In general such a direct search for the order of q modulo N is very inefficient, but as we will see, there is an efficient quantum algorithm that can determine the order.

A factor-finding algorithm based on the above observations is the following:

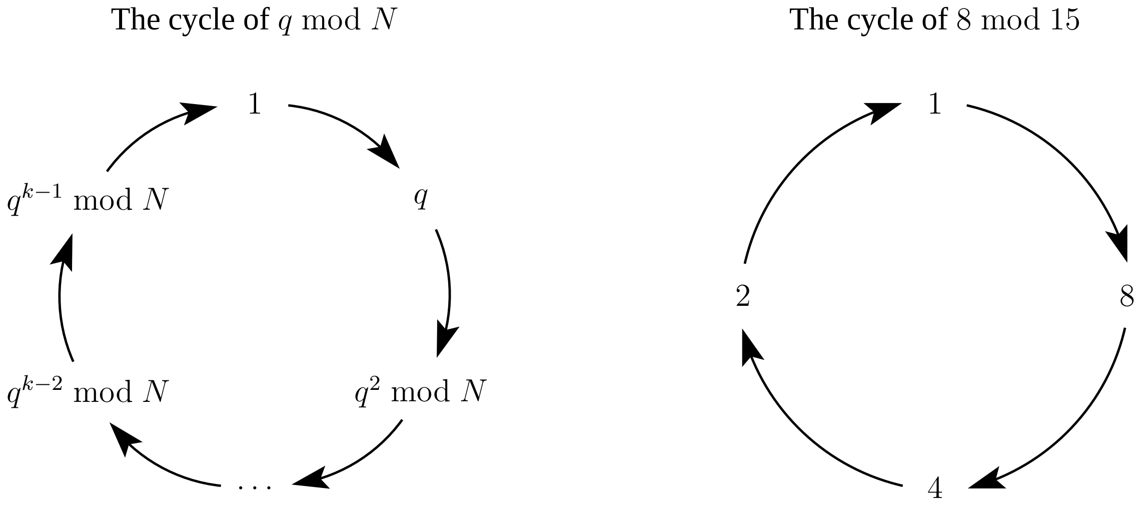
```
FACTORFIND($N$)
Input: A positive, non-prime whole number $N$.
Output: A proper factor $f$ of $N$, that is $f$ is a whole number such that $1 < f < N$ and $N = fg$ for some whole number $g$.

1. If $N$ is even, return $f = 2$.
2. If $N = p^k$ for $p$ prime, return $p$.
3. Randomly pick $1 < q < N - 1$.
    3.a. If $f = \text{gcd}(q, N) > 1$ return $f$.
4. Determine the order $k$ of $q$ modulo $N$ using the quantum order-finding algorithm.
    4.a. If $k$ is not even, repeat at step 3.
5. Write $k = 2l$ and determine $r = q^l \bmod N$ with $1 < r < N$.
    5.a. If $1 < f = \text{gcd}(r - 1, N) < N$, return $f$.
    5.b. If $1 < f = \text{gcd}(r + 1, N) < N$, return $f$.
    5.c. If we failed to find a proper factor, repeat at step 3.
```

The efficiency of this algorithm depends on the probability that a randomly chosen q at step 3 results in finding a factor. By using an analysis of the group of numbers q that satisfy $\text{gcd}(q, N) = 1$, it can be shown that this probability is sufficiently large.

The main problem that remains to be solved is that of finding the order of $q \bmod N$. A direct search for the order of $q \bmod N$ involves computing the sequence

$$1 \rightarrow q \rightarrow q^2 \bmod N \rightarrow \ldots \rightarrow q^{k-1} \bmod N \rightarrow 1 = q^k \bmod N. \tag{53}$$

This sequence can be conveniently visualized as a cycle whose length is the order of $q \bmod N$ (Fig. 4).



The cycle of $q \bmod N$

The cycle of $8 \bmod 15$

FIG. 4: Multiplicative cycles of $q \bmod N$. Each number on a cycle is obtained from the previous one by multiplication by $q \bmod N$.

To introduce the quantum algorithm, we first associate the logical quantum states $|0\rangle, |1\rangle, \ldots |N-1\rangle$ with the numbers $0, 1, \ldots, N-1$. The map f which takes each number on the cycle to the next number along the cycle is given by $f(x) = qx \bmod N$. For q satisfying gcd$(q, N) = 1$, the map f permutes not only the numbers on the cycle, but all the numbers modulo N. As a result, the linear operator $\hat{f}$ defined by $\hat{f}|x\rangle = |f(x)\rangle = |qx \bmod N\rangle$ is unitary. The quantum algorithm deduces the length of the cycle for q by making measurements to determine properties of the action of $\hat{f}$ on superpositions of the states $|q^s \bmod N\rangle$. To illustrate the basic ideas, we work out the example of $N = 15$ and $q = 8$. The action of $\hat{f}$ on the states $|1\rangle, |8\rangle, |4\rangle, |2\rangle$ in the cycle of $8 \bmod 15$ is completely determined by the eigenstates and eigenvalues of $\hat{f}$. For cyclicly acting permutations, a basis of eigenstates is given by the "Fourier" basis for the space spanned by the states in a cycle. For the cycle of interest, the Fourier basis consists of the states

$$\begin{array}{rcl}
|\psi_0\rangle &=& \frac{1}{2}\Big(|1\rangle + |8\rangle + |4\rangle + |2\rangle\Big) \\
|\psi_1\rangle &=& \frac{1}{2}\Big(|1\rangle + i|8\rangle - |4\rangle - i|2\rangle\Big) \\
|\psi_2\rangle &=& \frac{1}{2}\Big(|1\rangle - |8\rangle + |4\rangle - |2\rangle\Big) \\
|\psi_3\rangle &=& \frac{1}{2}\Big(|1\rangle - i|8\rangle - |4\rangle + i|2\rangle\Big)
\end{array} \qquad (54)$$

The phases of the l'th state of the cycle occurring in the sum for $|\psi_m\rangle$ can be written as i^{lm}. It follows that $\hat{f}|\psi_m\rangle = i^m|\psi_m\rangle$, that is, the eigenvalue of $\hat{f}$ for $|\psi_m\rangle$ is i^m. Note that in the complex numbers, the powers of i are all the fourth roots of unity. In general, the Fourier basis for the cycle $\ldots \rightarrow |q^l \bmod N\rangle \rightarrow \ldots$ consists of the states $|\psi_m\rangle = \sum_l \omega^{lm}|q^l \bmod N\rangle$, where $\omega = e^{i2\pi/k}$ is a primitive k'th root of unity. (The complex number x is a primitive k'th root of unity if k is the smallest whole number $k > 0$ such that $x^k = 1$. For example, both -1 and i are fourth roots of unity, but only i is primitive.)

It is, of course, possible to express the logical state $|1\rangle$ using the Fourier basis:

$$|1\rangle = \frac{1}{2}\Big(|\psi_0\rangle + |\psi_1\rangle + |\psi_2\rangle + |\psi_3\rangle\Big). \tag{55}$$

The key step of the quantum algorithm for order finding consists of a measurement to estimate a random eigenvalue of $\hat{f}$ whose associated eigenstate occurs in the expression for $|1\rangle$ in terms of the Fourier basis. If the eigenvalue found is a primitive k'th root of unity, we infer that the cycle length is divisible by k and check (using a classical algorithm) whether this is the order of q. In the example, the random eigenvalues are 1 (the only primitive first root of unity), i and $-i$ (primitive fourth roots of unity) and -1 (the primitive second root of unity). The order is found if the random eigenvalue is a primitive fourth root of unity, which happens with probability $1/2$ in this case.

The quantum algorithm for obtaining an eigenvalue is called the "phase estimation" algorithm. It exploits a more general version of the phase kick back we encountered in the solution of the parity problem. The phase kick back transfers the eigenvalue of an eigenstate of $\hat{f}$ to a Fourier basis on a number of additional qubits called "helper" or "ancilla" qubits. Which Fourier state results is then determined by a subroutine called the "measured quantum Fourier transform". We introduce these elements in the next paragraphs. Their combination for solving the general order-finding problem is shown in Fig. 10.

Fig. 5 shows how to kick back the eigenvalue of an eigenstate of $\hat{f}$ using a network implementing the controlled-$\hat{f}$ operation.

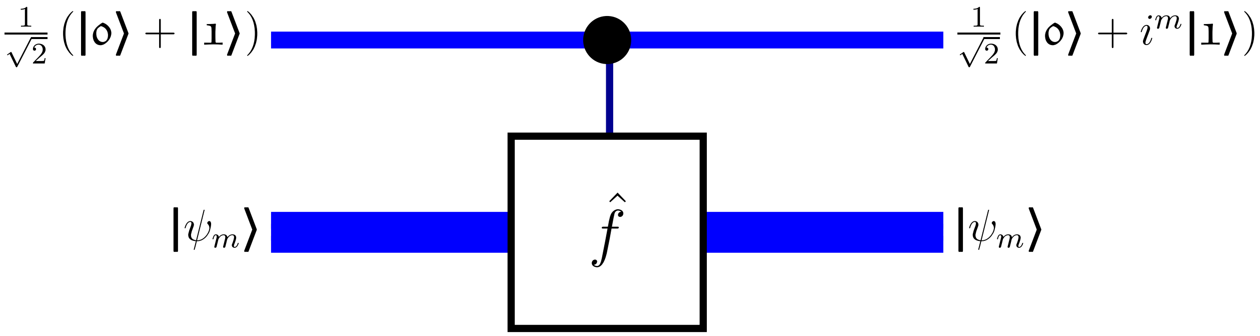

$\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$
$\frac{1}{\sqrt{2}}(|0\rangle + i^m|1\rangle)$
$|\psi_m\rangle$
$\hat{f}$
$|\psi_m\rangle$

FIG. 5: Phase estimation with one qubit. The input is a product state on one ancilla qubit and on a second quantum system as shown. The state $|\psi_m\rangle$ on the second system is an eigenstate of $\hat{f}$. For the example under discussion (see Eq. 54), the eigenvalue is i^m. A controlled-$\hat{f}$ operation is applied to the input, that is, $\hat{f}$ is applied to the second system conditional on $|1\rangle$ for the ancilla qubit. In the bra-ket notation, the total operation can be written as $|0\rangle\langle 0| + |1\rangle\langle 1|\hat{f}$ (system labels have been omitted). Since $\hat{f}$ changes only the phase of its input, the second system is unchanged, but the phase modifies the ancilla qubit's superposition as shown.

The network in Fig. 5 can be used with input $|1\rangle$ on the second system. From Eq. 55 and the superposition principle, it follows that the output correlates the different phase kickback states with the four eigenvectors

$|\psi_m\rangle$. That is, the network implements the following transformation:

$$\frac{1}{2\sqrt{2}}\left(|0\rangle+|1\rangle\right)\begin{pmatrix}|\psi_0\rangle \\ \quad +|\psi_1\rangle \\ \qquad +|\psi_2\rangle \\ \qquad\quad +|\psi_3\rangle\end{pmatrix} \longrightarrow \frac{1}{2\sqrt{2}}\begin{pmatrix}(|0\rangle+i^0|1\rangle)\,|\psi_0\rangle \\ \quad +(|0\rangle+i^1|1\rangle)\,|\psi_1\rangle \\ \qquad +(|0\rangle+i^2|1\rangle)\,|\psi_2\rangle \\ \qquad\quad +(|0\rangle+i^3|1\rangle)\,|\psi_3\rangle\end{pmatrix} \tag{56}$$

The hope is that a measurement of the first qubit can distinguish between the four possible phases that can be kicked back. However, because the four states are not mutually orthogonal, they are not unambiguously distinguishable by a measurement. To solve this problem, we use a second qubit and a controlled-$\hat{f}^2$ as shown in Fig. 6.

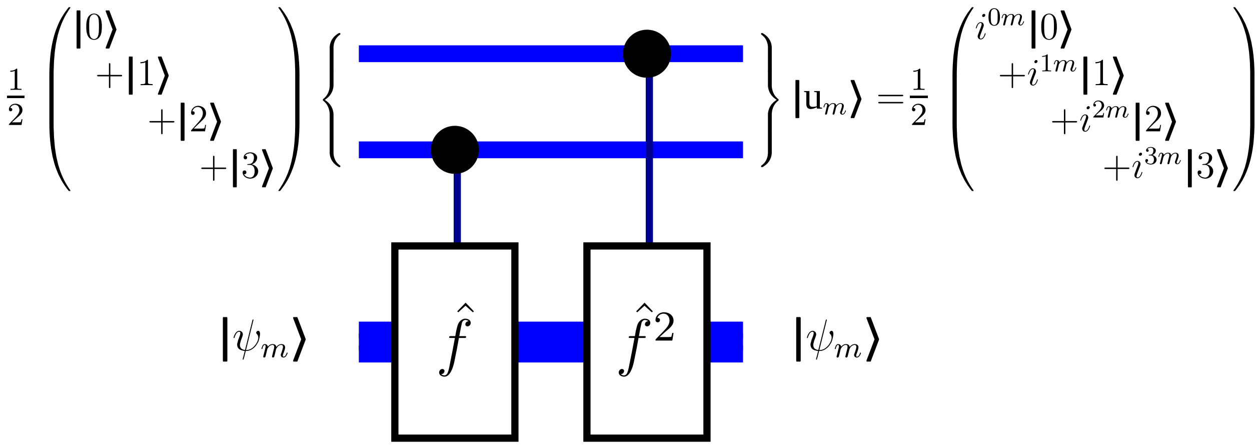

$\frac{1}{2}\begin{pmatrix}|0\rangle \\ +|1\rangle \\ +|2\rangle \\ +|3\rangle\end{pmatrix}$
$|u_m\rangle = \frac{1}{2}\begin{pmatrix}i^{0m}|0\rangle \\ +i^{1m}|1\rangle \\ +i^{2m}|2\rangle \\ +i^{3m}|3\rangle\end{pmatrix}$
$|\psi_m\rangle$
$\hat{f}$
$\hat{f}^2$
$|\psi_m\rangle$

FIG. 6: Phase estimation with two qubits. Using two qubits ensures distinguishability of the eigenvalues of $\hat{f}$ for the states $|\psi_m\rangle$. The states of the input qubits are used to represent the numbers from 0 to 3 in binary. The most significant bit (the "two"'s digit in the binary representation) is carried by the top qubit. That is, we make the following identification: $|0\rangle = |00\rangle$, $|1\rangle = |01\rangle$, $|2\rangle = |10\rangle$ and $|3\rangle = |11\rangle$. It follows that the network has the effect of applying $\hat{f}^m$ conditional on the input qubits' logical state being $|m\rangle$.

The four possible states $|u_m\rangle$ that appear on the ancilla qubits in the network of Fig. 6 are the Fourier basis for the cycle $0 \rightarrow 1 \rightarrow 2 \rightarrow 3 \rightarrow 0$ and are therefore orthonormal. If we apply the network of Fig. 6 with $|1\rangle$ instead of $|\psi_m\rangle$ at the lower input, the output correlates the four $|\psi_m\rangle$ in the superposition with the $|u_m\rangle$, which makes the information about the eigenvalues of $\hat{f}$ available in the Fourier basis of the two ancilla qubits. This approach has the advantage that the states are known, whereas in the Fourier basis for the cycle of $q \bmod N$, the states depend on the numbers in the cycle, which are not known in advance (except in very simple cases, such as the example we are working with).

To learn one of the eigenvalues of $\hat{f}$, the last step is to make a "measurement in the Fourier basis". For one qubit representing the binary numbers 0 and 1, the Fourier basis is $\frac{1}{\sqrt{2}}\left(|0\rangle+|1\rangle\right)$ and $\frac{1}{\sqrt{2}}\left(|0\rangle-|1\rangle\right)$, which is constructed as discussed after Eq. 54, but using the square root of unity $\omega=-1$ instead of the fourth root i. To make a measurement that determines which of the two basis vectors is present, it suffices to apply the Hadamard transform **H** and make a standard measurement, just as we did twice in the network of Fig. 2. A more complicated network works with two qubits representing the binary numbers from 0 to 3. Such a network is shown in Fig. 7.

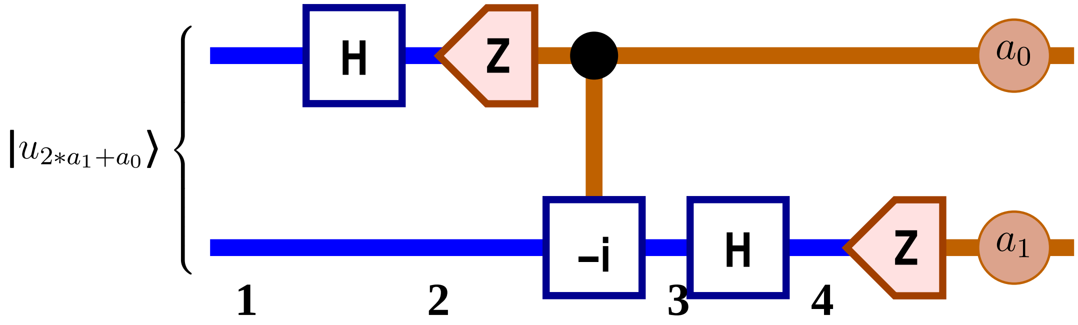


FIG. 7: The measured quantum Fourier transform [20] on two qubits representing the numbers $0, 1, 2, 3$. If the input is one of the Fourier states $|u_a\rangle$, where the binary digits of a are determined by $a = 2 * a_1 + a_0$, then the measurement outcomes are a_0 and a_1, as shown. The numbers under the network are checkpoints used for analyzing the network.

To see how the network extracts the bits in the index of $|u_a\rangle$, we can follow the states as the network is executed. The input state at checkpoint **1** in Fig. 7 is given by

$$|\phi_1\rangle = |u_a\rangle = \frac{1}{2}\begin{pmatrix} i^{0*a}|0\rangle \\ +i^{1*a}|1\rangle \\ +i^{2*a}|2\rangle \\ +i^{3*a}|3\rangle \end{pmatrix} = \frac{1}{2}\begin{pmatrix} i^{(0*2^1+0*2^0)(a_1*2^1+a_0*2^0)}|00\rangle \\ +i^{(0*2^1+1*2^0)(a_1*2^1+a_0*2^0)}|01\rangle \\ +i^{(1*2^1+0*2^0)(a_1*2^1+a_0*2^0)}|10\rangle \\ +i^{(1*2^1+1*2^0)(a_1*2^1+a_0*2^0)}|11\rangle \end{pmatrix}. \tag{57}$$

In the last sum, the relevant numbers have been fully expanded in terms of their binary digits to give a flavor of the general principles underlying the measured Fourier transform. The next step of the network applies a Hadamard gate to the qubit carrying the most significant digit. To understand how it succeeds in extracting a_0, the least significant bit of a, let b with binary digits b_0 and b_1 represent one of the logical states of the two qubits. As before, the most significant bit b_1 is represented by the top/first qubit that the first Hadamard gate is applied to. The phase of $|b\rangle$ in Eq. 57 is given by $i^{(b_1*2^1+b_0*2^0)(a_1*2^1+a_0*2^0)}$. Next, we determine how this phase depends on b_1:

$$\begin{aligned} i^{(b_1*2^1+b_0*2^0)(a_1*2^1+a_0*2^0)} &= i^{b_1*2^1*(a_1*2^1+a_0*2^0)}\ i^{b_0*2^0*(a_1*2^1+a_0*2^0)} \\ &= i^{b_1*a_1*2^2} i^{b_1*a_0*2^1}\ i^{b_0*2^0*(a_1*2^1+a_0*2^0)} \end{aligned}$$

$$\begin{aligned} &= \left(i^4\right)^{b_1*a_1}\left(i^2\right)^{b_1*a_0}\ i^{b_0*2^0*(a_1*2^1+a_0*2^0)} \\ &= (-1)^{b_1*a_0}\ i^{b_0*2^0*(a_1*2^1+a_0*2^0)}. \end{aligned} \tag{58}$$

It follows that if $a_0 = 0$, the phase does not depend on b_1, and if $a_0 = 1$, it changes sign with b_1. This sign change can be detected by performing the Hadamard transform and measuring, as can be seen explicitly by computing the state after the Hadamard transform at checkpoint **2**:

$$\begin{aligned} |\phi_2\rangle &= \frac{1}{\sqrt{2}}\left(i^{0*2^0*(a_1*2^1+a_0*2^0)}|a_0\rangle|0\rangle + i^{1*2^0*(a_1*2^1+a_0*2^0)}|a_0\rangle|1\rangle\right) \\ &= |a_0\rangle\frac{1}{\sqrt{2}}\left(i^{0*2^0*(a_1*2^1+a_0*2^0)}|0\rangle + i^{1*2^0*(a_1*2^1+a_0*2^0)}|1\rangle\right). \end{aligned} \tag{59}$$

The phases still show a dependence on a_0 via the terms $i^{b_0*2^0*a_0*2^0} = i^{b_0a_0}$. The purpose of the phase shift gate conditioned on the measurement outcome is to remove that dependence. The result is the following state on the remaining qubit at checkpoint **3**:

$$\begin{aligned} |\phi_3\rangle &= \frac{1}{\sqrt{2}}\left(i^{0*2^0*a_1*2^1}|0\rangle + i^{1*2^0*a_1*2^1}|1\rangle\right) \\ &= \frac{1}{\sqrt{2}}\left((-1)^{0*a_1}|0\rangle + (-1)^{1*a_1}|1\rangle\right) \\ &= \frac{1}{\sqrt{2}}\left(|0\rangle + (-1)^{a_1}|1\rangle\right). \end{aligned} \tag{60}$$

The final Hadamard transform followed by a measurement therefore results in the bit a_1, as desired.

The elements that we used to determine the order of 8 modulo 15 can be combined and generalized to determine the order of any q modulo N with gcd$(q, N) = 1$. The general network is shown in Fig. 10. Two features of the generalization are not apparent from the example. First, in order for the quantum network to be efficient, an efficient implementation of the controlled $\hat{f}^{2^l}$ operation is required. To obtain such an implementation, first note that to calculate $f^{2^l}(x) = q^{2^l}x \bmod N$ it suffices to square q repeatedly modulo N using $\left(q^{2^m}\right)^2 \bmod N = q^{2^{m+1}} \bmod N$ until we obtain $q^{2^l} \bmod N$. The result is then multiplied by $x \bmod N$. This computation is efficient. For any given q, it can be converted to an efficient network consisting of Toffoli and controlled-not gates acting on the binary representation of x. The conversion can be accomplished with standard techniques from the theory of reversible classical computation. The result is an efficient network for $\hat{f}^{2^l}$. Basic network theory can then be used to implement the controlled version of this operation [21].

The understand the second feature, note that we were lucky that the order of 8 modulo 15 was a power of 2, which nicely matched the measured Fourier transform we constructed on two qubits. The measured Fourier transform on m ancilla qubits can detect exactly only eigenvalues that are powers of the 2^m'th root of unity $e^{i\pi/2^{m-1}}$. The phase kicked back by the controlled operations corresponds to a k'th root of unity. Given a Fourier state on the cycle of $q \bmod N$, the resulting state on the ancilla qubits has phases that go as powers of a k'th root of unity. Fortunately, the ancilla's Fourier basis is such that the measured Fourier transform picks up primarily those basis states whose generating phase is close to the kick back phase. Thus we are likely to detect a nearby $\omega = e^{i\,l\pi/2^{m-1}}$. It is still necessary to infer (a divisor of) k

from knowledge of such an ω. Since we know that the order k is bounded by N, the number of possible phases kicked back that are near the measured ω is limited. To ensure that there is only one possible such phase, it is necessary to choose m such that $2^m > N^2$. See also the caption of Fig. 10.

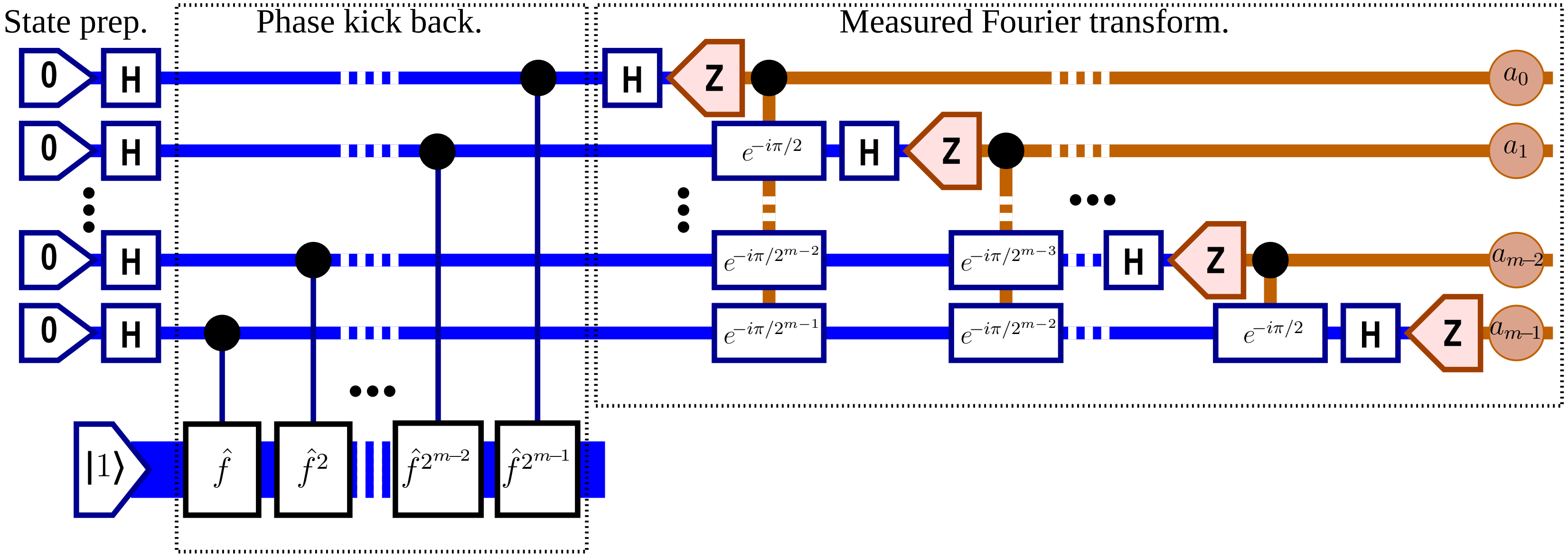


FIG. 9: Network for quantum order finding and phase estimation. The number m of qubits used for the phase kick back has to be chosen such that $m > 2 * \log_2(k_u)$, where k_u is a known upper bound on the order k of $q \bmod N$. Because $N > k$, one can set $m = 2\lceil\log_2(N)\rceil$, where $\lceil x \rceil$ is the least whole number $s \geq x$. There is an eigenvalue $\lambda_l = e^{i\,2l\pi/k}$ of one of the Fourier eigenvectors associated with the cycle of $q \bmod N$ such that the number a whose binary digits are the measurement outcomes satisfies $e^{i\pi a/2^{m-1}} \approx e^{i\,2\pi l/k}$. More precisely, with probability above $.405$, there exists l such that $|a/2^m - l/k| \leq 1/2^{m+1}$ [16]. Since any two distinct rational numbers with denominator at most k_u differ by at least $1/k_u^2 > 2/2^{m+1}$, the theory of rational approximations guarantees that we can uniquely determine the number l/k. There is an efficient classical algorithm based on continued fractions that computes r and s with $r/s = l/k$ and $s = k/\text{gcd}(l,k)$. The probability that $\text{gcd}(l,k) = 1$ is at least $1/(\log_2(k)+1)$, in which case we learn that $s = k$, and this is the order of $q \bmod N$. Note that the complexity of the network depends on the complexity of implementing the controlled $\hat{f}^{2^l}$ operations. Because these operations can be implemented efficiently, the network and hence the determination of the order of $q \bmod N$ are efficient in the sense that on average, polynomial resources in $\log_2(N)$ suffice.

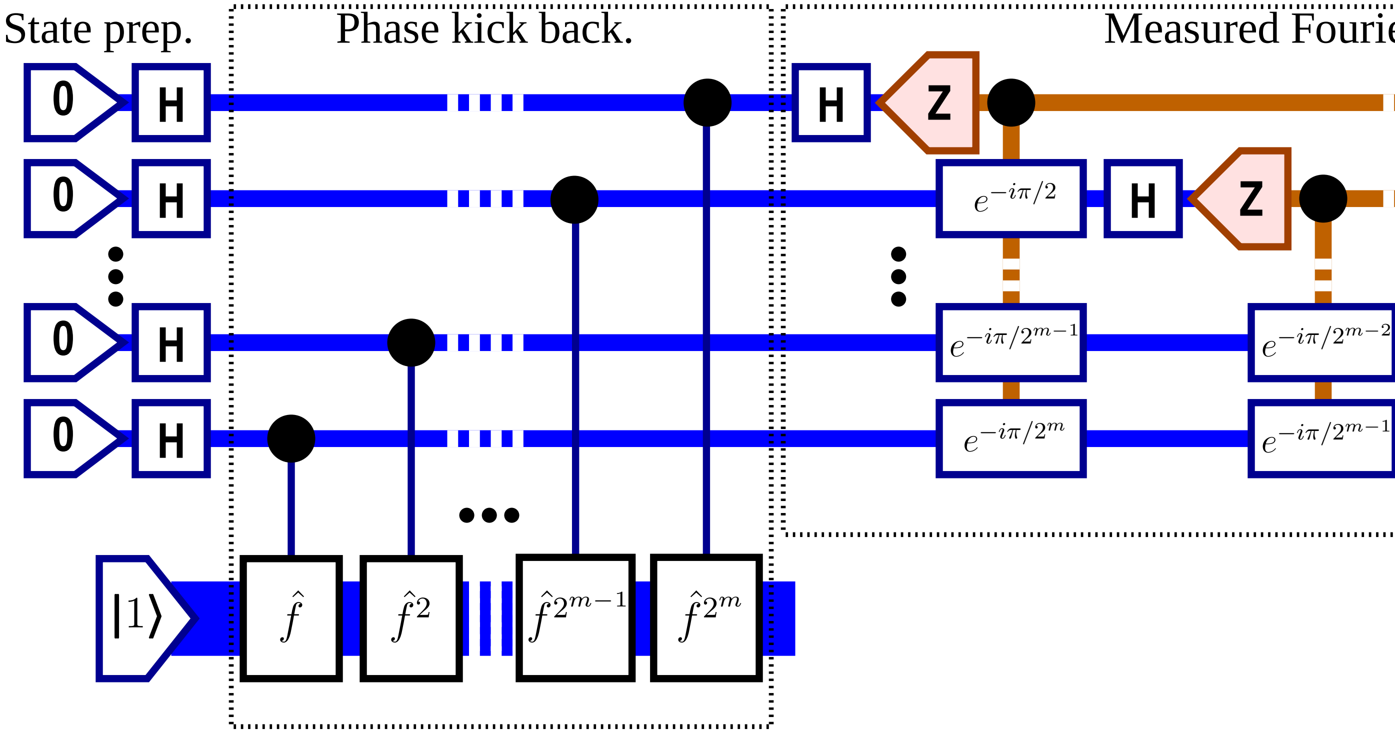

State prep.
Phase kick back.
Measured Fourie
0
H
0
H
0
H
0
H
H
Z
Z
H
$e^{-i\pi/2}$
$e^{-i\pi/2^{m-1}}$
$e^{-i\pi/2^{m}}$
$e^{-i\pi/2^{m-2}}$
$e^{-i\pi/2^{m-1}}$
$|1\rangle$
$\hat{f}$
$\hat{f}^2$
$\hat{f}^{2^{m-1}}$
$\hat{f}^{2^m}$

FIG. 10: Network for quantum order finding and phase estimation. The number m of qubits used for the phase kick back has to be chosen such that $m > 2{*}\log_2(k_u)$, where k_u is a known upper bound on the order k of $q \bmod N$. Because $N > k$, one can set $m = 2\lceil\log_2(N)\rceil$, where $\lceil x\rceil$ is the least whole number $s \geq x$. There is an eigenvalue $\lambda_l = e^{i\,2l\pi/k}$ of one of the Fourier eigenvectors associated with the cycle of $q \bmod N$ such that the number a whose binary digits are the measurement outcomes satisfies $e^{i\pi a/2^{m-1}} \approx e^{i\,2\pi l/k}$. More precisely, with probability above $.405$, there exists l such that $|a/2^m - l/k| \leq 1/2^{m+1}$ [16]. Since any two distinct rational numbers with denominator at most k_u differ by at least $1/k_u^2 > 2/2^{m+1}$, the theory of rational approximations guarantees that we can uniquely determine the number l/k. There is an efficient classical algorithm based on continued fractions that computes r and s with $r/s = l/k$ and $s = k/\text{gcd}(l,k)$. The probability that $\text{gcd}(l,k) = 1$ is at least $1/(\log_2(k)+1)$, in which case we learn that $s = k$, and this is the order of $q \bmod N$. Note that the complexity of the network depends on the complexity of implementing the controlled $\hat{f}^{2^l}$ operations. Because these operations can be implemented efficiently, the network and hence the determination of the order of $q \bmod N$ are efficient in the sense that on average, polynomial resources in $\log_2(N)$ suffice.

3 Advantages of Quantum Information

The notion of quantum information as explained in this primer was established in the 1990s. It emerged from research focused on understanding how physics affects our capabilities to communicate and to process information. The recognition that usable types of information need to be physically realizable was repeatedly emphasized by R. Landauer who proclaimed that "information is physical" [22]. Beginning in the 1960s, R. Landauer studied the thermodynamic cost of irreversible operations in computation [23]. C. Bennett showed that by using reversible computation, this cost can be avoided [24]. Limitations of measurement in quantum mechanics were investigated early by researchers such as J. von Neumann [25, 26], and later by A. Holevo [27] and C. Helstrom [28]. A. Holevo introduced the idea of quantum communication channels and found bounds on their capacity for transmitting classical information [29]. Initially, most work focused on determining the physical limitations placed on classical information processing. The fact that pairs of two-level systems can have correlations not possible for classical systems was proven by J. Bell [30] in 1964. Subsequently, indications that quantum mechanics offers advantages to information processing came from S. Wiesner's studies of cryptographic applications [1] in the late 1960s. S. Wiesner's work was not recognized until the 1980s, when C. Bennett, G. Brassard, S. Breidbart and S. Wiesner [2] introduced the idea of quantum cryptography, which can be used to communicate in secret.

Initially, the term "quantum computation" was mostly used to refer to classical computers realized using quantum mechanical systems. In the 1980s, P. Benioff [31], R. Feynman [3] and Y. I. Manin [32] introduced the idea of a quantum computer based on quantum information. They noted that the apparent exponential complexity of simulating quantum mechanics on a classical computer might be overcome if we could use a computer that is itself based on quantum mechanics. A formal model of quantum Turing machines was soon defined by D. Deutsch [33], who later also introduced quantum networks [34]. D. Deutsch and R. Jozsa [35] were the first to introduce a black box problem that can be solved deterministically on a quantum computer in fewer steps than on a classical computer.

In spite of suggestions that it could lead to large efficiency improvements in simulating physics, quantum information processing was still largely an academic subject. Based on work by E. Bernstein and U. Vazirani [13] that formalized quantum complexity theory, D. Simon [14] showed that, for black-box problems, quantum computers can be exponentially more efficient than classical deterministic or probabilistic computers, giving the first indication of a strong advantage for quantum information processing. It was Shor's algorithm for factoring large whole numbers [4, 5] that finally convinced a larger community that quantum information was more than just a tool for realizing classical computers. This change in attitudes was in no small part due to the fact that the security of commonly used cryptographic protocols is based on the hardness of factoring.

At that point, it was still generally believed that the fragility of quantum states made it unlikely for reasonably large quantum computers to be realized in practice. But the discovery by Shor [36] and A. Steane [37] that quantum error-correction was possible soon changed that view, see [12] for an introductory overview.

As a result of the recognition of the utility and realizability of quantum information, the science of quantum information processing is a rapidly growing field. As quantum information becomes increasingly accessible by technology, its usefulness will be more apparent. The next few sections briefly discuss what we currently know about applications of quantum information processing. A useful reference text on quantum computation and information with historical notes is the book by M. Nielsen and I. Chuang [38].

3.1 Quantum Algorithms

Shor's factoring algorithm, which precipitated much of the current work in quantum information processing, is based on a quantum realization of the fast Fourier transform. The most powerful version of this technique is now represented by the phase-estimation algorithm of A. Kitaev [17] as formalized by R. Cleve *et al.* [16]. See Sect. 2.10 for an explanation of the factoring algorithm and phase estimation. The best known application of quantum factoring is to cryptanalysis, where it can be used to efficiently break the currently used public-key cryptographic codes. Whether there are any constructive applications of quantum factoring and its generalizations remains to be determined. For users of public key cryptography, a crucial question is: "How long can public key codes based on factoring continue to be used safely?" To attempt to answer this question, one can note that to break a code with a typical key size of 1000 bits requires more than 3000 qubits and 10^8 quantum gates, which is well out of reach of current technology. However, it is conceivable that a recording of encrypted information transmitted in 2000 can be broken in the next "few" decades.

Shor's quantum factoring algorithm was not the first with a significant advantage over classical algorithms. The first quantum algorithms to be proposed with this property were algorithms for simulating quantum mechanical systems. These algorithms simulate the evolution of a reasonably large number of interacting quantum particles, for example, the electrons and nuclei in a molecule. The algorithms' outputs are what would be measurable physical quantities of the system being simulated. The known methods for obtaining these quantities on classical computers scale exponentially with the number of particles, except in special cases.

The idea of using quantum computers for simulating quantum physics spurred the work that eventually lead to the quantum factoring algorithm. However, that idea did not have the broad scientific impact that the quantum factoring algorithm had. One reason is that because of its cryptographic applications, factoring is a heavily studied problem in theoretical computer science and cryptography. Because so many people have tried to design efficient algorithms for factoring and failed, the general belief that factoring is hard for classical computers has a lot of credibility. In contrast, the problem of quantum physics simulation has no simple formulation as an algorithmic problem suitable for study in theoretical computer science. Furthermore, many researchers still believe that the physically relevant questions can be answered with efficient classical algorithms, requiring only more cleverness on the part of the algorithms designers. Another reason for the lack of impact is that many of the fundamental physical quantities of interest are not known to be efficiently accessible even on quantum computers. For example, one of the first questions about a physical system with a given Hamiltonian (energy observable), is: What is the ground state energy? It is known that the ability to efficiently answer this question for physically reasonable Hamiltonians leads to efficient algorithms for hard problems such as the traveling salesman or the scheduling problems. In spite of occasional claims to the contrary, an efficient quantum solution to these problems is widely considered unlikely.

Most quantum algorithms for physics simulations are based on a direct emulation of the evolution of a quantum mechanical system. The focus of the original proposals by Feynman and others was on how to implement the emulation using a suitable formulation of general-purpose quantum computers. After such computers were formalized by Deutsch, the implementation of the emulation was generalized and refined by S. Lloyd [39], Wiesner [40] and C. Zalka [41]. The ability to emulate the evolution of quantum systems is actually widely used by classical "Monte-Carlo" algorithms for simulating physics, where the states amplitudes are, in effect, represented by expectations of random variables that are computed during the simulation. As in the case of the quantum algorithms for physics emulation, the Monte-Carlo algorithms efficiently evolve the representation of the quantum system. The inefficiency of the classical algorithm arises only in determining a physical quantity of interest. In the case of Monte-Carlo algorithms, the "measurement" of a physical quantity suffers from the so-called "sign problem", often resulting in exponentially large, random errors that can be reduced only by repeating the computation extremely many times. In contrast, the quantum algorithms for emulation can determine many (but not all) of the interesting physical quantities with polynomially bounded statistical errors. How to efficiently implement measurements of these quantities has been the topic of more recent work in this area, much of which is based on variants of the phase estimation algorithm [42, 43, 44, 45, 46].

Although several researchers have suggested that there are interesting quantum physics simulations that can be implemented with well below 100 qubits, one of the interesting problems in this area of research is to come up with a specific simulation algorithm that uses small numbers of qubits and quantum gates, and that computes an interesting physical quantity not easily obtainable using available classical computers.

Another notable algorithm for quantum computers, unstructured quantum search, was described by L. Grover [6]. Given is a black box that computes a binary function f on inputs x with $0 \leq x < N$. The function f has the property that there is a unique input a for which $f(a) = 1$. The standard quantum version of this black box implements the transformation $\hat{f}|x\rangle|b\rangle = |x\rangle|b \oplus f(x)\rangle$, where b is a bit and $b \oplus f(x)$ is computed modulo 2. Unstructured quantum search finds a quadratically faster, that is, in time of order $N^{1/2}$, than the best classical black-box search, which requires time of order N. The context for this algorithm is the famous $P \neq NP$ conjecture, which is captured by the following algorithmic problem: Given is a classical circuit C that computes an output. Is there an input to the circuit for which the circuit's output is 1? Such an input is called a "satisfying" input or "assignment". For any given input, it is easy to check the output, but an efficient algorithm that finds a satisfying input is conjectured to be impossible. This is the $P \neq NP$ conjecture. Generalizations of Grover's search algorithm ("amplitude amplification" [47]) can be used to find satisfying inputs faster than the naive classical search, which tries each possible input in some, possibly random, order. It is worth noting, howoever, that if sufficient classical parallelism is available, quantum search loses many of its advantages.

The three algorithms just described capture essentially all the known algorithmic advantages of quantum computers. Almost all algorithms that have been described are applications of phase estimation or of amplitude amplification. These algorithms well justify developing special purpose quantum information processing technology. Will general purpose quantum computers be useful? More specifically, what other algorithmic advantages do quantum computers have?

3.2 Quantum Communication

Quantum communication is an area in which quantum information has proven (rather than conjectured) advantages. The best known application is quantum cryptography, whereby two parties, Alice and Bob, can generate a secret key using a quantum communication channel (for example, photons transmitted in optical fiber) and an authenticated classical channel (for example, a telephone line). Any attempt at learning the key by eavesdropping is detected. A quantum protocol for generating a secret key is called a "quantum key exchange" protocol. There are no equally secure means for generating a secret key by using only classical deterministic channels. Few quantum operations are needed to implement quantum key exchange, and as a result there are working prototype systems [48, 49, 50]. To overcome the distance limitations (tens of kilometers) of current technology requires the use of quantum error-correction and hence more demanding quantum technology.

Quantum key exchange is one of an increasing number of multi-party problems that can be solved more efficiently with quantum information. The area of research concerned with how several parties at different locations can solve problems while minimizing communication resources is called "communication complexity". For quantum communication complexity (R. Cleve and H. Burhman [51]), the communication resources include either shared entangled qubits or a means for transmitting quantum bits. A seminal paper by Burhman, Cleve and W. Van Dam [52] shows how the non-classical correlations present in maximally entangled states lead to protocols based on such states that are more efficient than any classical deterministic or probabilistic protocol achieving the same goal. R. Raz [53] showed that there is an exponential improvement in communication resources for a problem in which Alice and Bob have to answer a question about the relationship between a vector known to Alice and a matrix known to Bob. Although this problem is artificial, it suggests that there are potentially useful advantages to be gained from quantum information in this setting.

3.3 Quantum Control

According to G. Moore's law of semiconductor technology, the size of transistors is decreasing exponentially, by a factor of about $.8$ every year. If this trend continues, then over the next few decades devices will inevitably be built whose behavior will be primarily quantum mechanical. For the purpose of classical computation, the goal is to remove the quantum behavior and stabilize classical information. But quantum information offers an alternative: It is possible to directly use the quantum effects to advantage. Whether or not this advantage is useful (and we believe it is), the ideas of quantum information can be used to systematically understand and control quantum mechanical systems.

The decreasing size of semiconductor components is a strong motivation to strive for better understanding the behavior of condensed matter quantum mechanical systems. But there is no reason to wait for Moore's law: There are a rapidly increasing number of experimental systems in which quantum mechanical effects are being used and investigated. Examples include many optical devices (lasers, microwave cavities, entangled photon pairs), nuclear magnetic resonance with molecules or in solid state, trapped ion or atom systems, Rydberg atoms, superconducting devices (Josephson junctions, SQUIDs) and spintronics (electron spins in semiconductor devices). Many of these systems are being considered as candidates for realizing quantum information processing. Yet, regardless of the future of quantum information processing, there is ample motivation for studying these systems.

3.4 Outlook

The science of quantum information processing is promising to have a significant impact on how we process information, solve algorithmic problems, engineer nano-scale devices and model fundamental physics. It is already changing the way we understand and control matter at the atomic scale, making the quantum world more familiar, accessible and understandable. Whether or not we do most of our everyday computations by using the classical model, it is likely that the physical devices that support these computations will exploit quantum mechanics and integrate the ideas and tools that have been developed for quantum information processing.

Acknowledgements: We thank Nikki Cooper and Ileana Buican for their extensive encouragement and editorial help.

Addresses:	E. Knill:	Los Alamos National Laboratory	knill@lanl.gov
	R. Laflamme:	University of Waterloo and Perimeter Institute	laflamme@iqc.ca
	H. Barnum:	Los Alamos National Laboratory	barnum@lanl.gov
	D. Dalvit:	”	dalvit@lanl.gov
	J. Dziarmaga:	”	jpd@lanl.gov
	J. Gubernatis:	”	jg@lanl.gov
	L. Gurvits:	”	gurvits@lanl.gov
	G. Ortiz:	”	g_ortiz@lanl.gov
	L. Viola:	”	lviola@lanl.gov
	W. H. Zurek:	”	whz@lanl.gov

References

[1] S. Wiesner. Conjugate coding. *Sigact News, (original manuscript ∼1969)*, 15:78–88, 1983.

[2] C. H. Bennett, G. Brassard, S. Breidbart, and S. Wiesner. Quantum cryptography, or unforgeable subway tokens. In *Advances in Cryptology: Proceedings of Crypto'82*, pages 267–275. Plenum Press, 1982.

[3] R. P. Feynman. Simulating physics with computers. *Int. J. Theor. Phys.*, 21:467–488, 1982.

[4] P. W. Shor. Algorithms for quantum computation: Discrete logarithms and factoring. In *Proceedings of the 35'th Annual Symposium on Foundations of Computer Science*, pages 124–134, Los Alamitos, California, 1994. IEEE Press.

[5] P. W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM J. Comput.*, 26:1484–1509, 1997.

[6] L. K. Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the 28th Annual ACM Symposium on the Theory of Computation*, pages 212–219, New York, New York, 1996. ACM press.

[7] A. Yao. Quantum circuit complexity. In *Proceedings of the 34th Annual Symposium on Foundations of Computer Science*, pages 352–360, Los Alamitos, California, 1993. IEEE Press.

[8] T. H. Cormen, C. E. Leiserson, and R. L. Rivest. *Introduction to Algorithms*. MIT Press, Cambridge, Mass, 1990.

[9] C. H. Papadimitriou. *Computational Complexity*. Addison-Wesley, Reading, Mass, 1994.

[10] R. Gupta, S. A. Smolka, and S. Bhaskar. On randomization in sequential and distributed algorithms. *ACM Comp. Surv.*, 26:7–86, 1994.

[11] R. Laflamme, E. Knill, D. Cory, E. M. Fortunato, T. Havel, C. Miquel, R. Martinez, C. Negrevergne, G. Ortiz, M. A. Pravia, S. Sinha, R. Somma, and L. Viola. Introduction to NMR quantum information processing. Technical Report LAUR-02-6132, Los Alamos National Laboratory, 2001. To appear in LA Science.

[12] E. Knill, R. Laflamme, A. Ashikhmin, H. Barnum, L. Viola, and W. Zurek. Introduction to quantum error correction. Technical Report LAUR-01-6115, Los Alamos National Laboratory, 2001. To appear in LA Science.

[13] E. Bernstein and U. Vazirani. Quantum complexity theory. In *Proceedings of the 25th Annual ACM Symposium on the Theory of Computation*, pages 11–20, New York, New York, 1993. ACM press.

[14] D. R. Simon. On the power of quantum computation. In *Proceedings of the 35th Annual Symposium on Foundations of Computer Science*, pages 116–123, Los Alamitos, California, 1994. IEEE Press.

[15] A. Ekert. From quantum code-making to quantum code-breaking. In *The geometric universe*, pages 195–214. Oxford Univ. Press, Oxford, 1998.

[16] R. Cleve, A. Ekert, C. Macchiavello, and M. Mosca. Quantum algorithms revisited. *Proc. R. Soc. Lond. A*, 454:339–354, 1998. quant-ph/9708016.

[17] A. Yu. Kitaev. Quantum measurements and the Abelian stabilizer problem. quant-ph/9511026, 1995.

[18] E. D. Bolker. *Elementary Number Theory: An Algebraic Approach*. W. A. Benjamin, Inc., New York, 1970.

[19] G. H. Hardy and E. M. Wright. *An Introduction to the Theory of Numbers*. Oxford University Press, London, fifth edition edition, 1979.

[20] R. B. Griffiths and C-S Niu. Semiclassical Fourier transform for quantum computation. *Phys. Rev. Lett.*, 76:3228–3231, 1996.

[21] A. Barenco, C. H. Bennett, R. Cleve, D. P. DiVincenzo, N. Margolus, P. Shor, T. Sleator, J. Smolin, and H. Weinfurter. Elementary gates for quantum computation. *Phys. Rev. A*, 52:3457–3467, 1995.

[22] R. Landauer. Information is physical. *Phys. Today*, 44:22–29, 1991.

[23] R. Landauer. Irreversibility and heat generation in the computing process. *IBM J. Res. Dev.*, 5:183–192, 1961. See also [54].

[24] C. H. Bennett. Logical reversibility of computation. *IBM J. Res. Dev.*, 17:525–532, 1973.

[25] J. von Neumann. Measurement and reversibility. In *Mathematische Grundlagen der Quantenmechanik*, page Ch. V. Springer, Berlin, 1932.

[26] J. von Neumann. The measuring process. In *Mathematische Grundlagen der Quantenmechanik*, page Ch. VI. Springer, Berlin, 1932.

[27] A. S. Holevo. Statistical problems in quantum physics. In G. Maruyama and J. V. Prokhorov, editors, *Proceedings of the Second Japan–USSR Symposium on Probability Theory, Lecture Notes in Mathematics **330***, Berlin, 1973. Springer Verlag.

[28] C. W. Helstrom. *Quantum Detection and Estimation Theory*. Mathematics in Science and Engineering **123**. Academic Press, New York, 1976.

[29] A. S. Holevo. Bounds for the quantity of information transmitted by a quantum communication channel. *Prob. Inf. Trans.*, 9:177–183, 1973.

[30] J. S. Bell. On the einstein podolsky rosen paradox. *Physics*, 1:195–200, 1964.

[31] P. Benioff. The computer as a physical system: A microscopic quantum mechanical hamiltonian model of computers as represented by turing machines. *J. Stat. Phys.*, 22:563–591, 1980.

[32] Y. I. Manin. *The Computable and the Not Computable*. Sovetskoye Radio, Moscow, 1980. In Russian.

[33] D. Deutsch. Quantum theory, the Church-Turing principle and the universal quantum computer. *Proc. R. Soc. Lond. A*, 400:97–117, 1985.

[34] D. Deutsch. Quantum computational networks. *Proc. R. Soc. Lond. A*, 425:73–90, 1989.

[35] D. Deutsch and R. Jozsa. Rapid solution of problems by quantum computation. *Proc. R. Soc. Lond. A*, 439:553–558, 1992.

[36] P. W. Shor. Scheme for reducing decoherence in quantum computer memory. *Phys. Rev. A*, 52:2493–2496, 1995.

[37] A. Steane. Multiple particle interference and quantum error correction. *Proc. R. Soc. Lond. A,* 452:2551–2577, 1996.

[38] M. A. Nielsen and I. L. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, 2001.

[39] S. Lloyd. Universal quantum simulators. *Science*, 273:1073–1078, 1996.

[40] S. Wiesner. Simulations of many-body quantum systems by a quantum computer. quant-ph/9603028, 1996.

[41] C. Zalka. Simulating quantum-systems on a quantum computer. *Proc. R. Soc. Lond. A*, 454:313–322, 1998.

[42] B. M. Terhal and D. P. DiVincenzo. Problem of equilibration and the computation of correlation functions on a quantum computer. *Phys. Rev. A*, 61:022301/1–22, 2000.

[43] E. Knill and R. Laflamme. On the power of one bit of quantum information. *Phys. Rev. Lett.*, 81:5672–5675, 1998.

[44] D. S. Abrams and S. Lloyd. Quantum algorithm providing an exponential speed increase for finding eigenvalues and eigenvectors. *Phys. Rev. Lett.*, 83:5162–5165, 1999.

[45] G. Ortiz, J. E. Gubernatis, E. Knill, and R. Laflamme. Quantum algorithms for fermionic simulations. *Phys. Rev. A*, 64:022319/1–14, 2001.

[46] C. Miquel, J. P. Paz, M. Saraceno, E. Knill, R. Laflamme, and C. Negrevergne. Interpretation of tomography and spectroscopy as dual forms of quantum computations. *Nature*, 418:59–62, 2002. quant-ph/0109072.

[47] G. Brassard, P. Høyer, and A. Tapp. Quantum counting. In K. G. Larsen, S. Skyum, and G. Winskel, editors, *Automata, Languages and Programming, Proceedings of ICALP'98*, volume 1443 of *Lecture Notes in Computer Science*, pages 820–831, Berline, Germany, 1998. Springer Verlag.

[48] R. J. Hughes, G. L. Morgan, and C. G. Peterson. Quantum key distribution over a 48km optical fibre network. *J. Mod. Optics*, 47:533–547, 2000.

[49] P. D. Townsend. Quantum cryptography on optical fiber networks. *Opt. Fiber Tech.: Mat., Dev., Sys.*, 4:345–370, 1998.

[50] G. Ribordy, J. Brendel, J.-D. Gautier, N. Gisin, and H. Zbinden. Long-distance entanglement-based quantum key distribution. *Phys. Rev. A*, 63:012309/1–12, 2001.

[51] R. Cleve and H. Buhrman. Substituting quantum entanglement for communication. *Phys. Rev. A*, 56:1201–1204, 1997.

[52] H. Buhrman, R. Cleve, and W. Van Dam. Quantum entanglement and communication complexity. *SIAM J. Comput.*, 30:1829–1841, 2000.

[53] R. Raz. Exponential separation of quantum and classical communication complexity. In *Proceedings of the 31st Annual ACM Symposium on the Theory of Computation (STOC)*, pages 358–367, El Paso, Texas, 1999. ACM Press.

[54] R. Landauer. Irreversibility and heat generation in the computing process. *IBM J. Res. Dev.*, 44:261–269, 2000.

4 Glossary

Algorithm. A set of instructions to be executed by a computing device. What instructions are available depends on the computing device. Typically, instructions include commands for manipulating the contents of memory and means for repeating blocks of instructions indefinitely or until a desired condition is met.

Amplitude. A quantum system with a chosen orthonormal basis of “logical” states $|i\rangle$ can be in any superposition $\sum_i \alpha_i|i\rangle$ of these states, where $\sum_i |\alpha_i|^2 = 1$. In such a superposition, the complex numbers α_i are called the amplitudes. Note that the amplitudes depend on the chosen basis.

Ancillas. Helper systems used to assist in a computation involving other information systems.

Bell basis. For two qubits A and B, the Bell basis consists of the four states $\frac{1}{\sqrt{2}}\left(|00\rangle_{\mathsf{AB}} \pm |11\rangle_{\mathsf{AB}}\right)$ and $\frac{1}{\sqrt{2}}\left(|01\rangle_{\mathsf{AB}} \pm |10\rangle_{\mathsf{AB}}\right)$.

Bell states. The members of the Bell basis.

Bit. The basic unit of deterministic information. It is a system that can be in one of two possible states, 0 and 1.

Bit sequence. A way of combining bits into a larger system whose constituent bits are in a specific order.

Bit string. A sequence of 0’s and 1’s that represents a state of a bit sequence. Bit strings are the words of a binary alphabet.

Black box. A computational operation whose implementation is unknown. Typically, a black box implements one of a restricted set of operations, and the goal is to determine which of these operations it implements by using it with different inputs. Each use of the black box is called a “query”. The smallest number of queries required to determine the operation is called the “query complexity” of the restricted set. Determining the query complexity of sets of operations is an important problem area of computational complexity.

Bloch sphere. The set of pure states of a qubit represented as points on the surface of the unit sphere in three dimensions.

Bra. A state expression of the form $\langle\psi|$, which is considered to be the conjugate transpose of the ket expression $|\psi\rangle$.

Bra-ket notation. A way of denoting states and operators of quantum systems with kets (for example, $|\psi\rangle$) and bras (for example, $\langle\phi|$).

Circuit. A combination of gates to be applied to information units in a prescribed order. To draw circuits, one often uses a convention for connecting and depicting gates. See also “network”.

Circuit complexity. The circuit complexity of an operation on a fixed number of information units is the smallest number of gates required to implement the operation.

Classical information. The type of information based on bits and bit strings and more generally on words formed from finite alphabets. This is the information used for communication between people. Classical information can refer to deterministic or probabilistic information, depending on the context.

Computation. The execution of the instructions provided by an algorithm.

Computational states. See the entry for “logical states”.

Computer. A device that processes information.

Density matrix or operator. A representation of pure and mixed states without redundancy. For a pure state $|\psi\rangle$, the corresponding density operator is $|\psi\rangle\langle\psi|$. A general density operator is a probabilistic combination $\sum_i \lambda_i |\psi_i\rangle\langle\psi_i|$, with $\sum_i \lambda_i = 1$.

Deterministic information. The type of information that is based on bits and bit strings. Deterministic information is classical, but it explicitly excludes probabilistic information.

Distinguishable states. In quantum mechanics, two states are considered distinguishable if they are orthogonal. In this case, a measurement exists that is guaranteed to determine which of the two states a system is in.

Efficient computation. A computation is efficient if it requires at most polynomially many resources as a function of input size. For example, if the computation returns the value $f(x)$ on input x, where x is a bit string, then it is efficient if there exists a power k such that the number of computational steps used to obtain $f(x)$ is bounded by $|x|^k$, where $|x|$ is the length (number of bits) of x.

Entanglement. A non-classical correlation between two quantum systems most strongly exhibited by the maximally entangled states such as the Bell states for two qubits, and considered to be absent in mixtures of product states (which are called “separable” states). Often states that are not separable are considered to be entangled. However, nearly separable states do not exhibit all the features of maximally entangled states. As a result, studies of different types of entanglement are an important component of quantum information theory.

Gate. An operation applied to information for the purpose of information processing.

Global phase. Two quantum states are indistinguishable if they differ only by a global phase. That is, $|\psi\rangle$ and $e^{i\phi}|\psi\rangle$ are in essence the same state. The global phase difference is the factor $e^{i\phi}$. The equivalence of the two states is apparent from the fact that their density matrices are the same.

Hilbert space. An n-dimensional Hilbert space consists of all complex n-dimensional vectors. A defining operation in a Hilbert space is the inner product. If the vectors are thought of as column vectors, then the inner product $\langle x, y\rangle$ of x and y is obtained by forming the conjugate transpose $x^\dagger$ of x and calculating $\langle x, y\rangle = x^\dagger y$. The inner product induces the usual squared norm $|x|^2 = \langle x, x\rangle$.

Information. Something that can be recorded, communicated, and computed with. Information is fungible; that is, its meaning can be identified regardless of the particulars of the physical realization. Thus, information in one realization (such as ink on a sheet of paper) can be easily transferred to another (for example, spoken words). Types of information include deterministic, probabilistic and quantum information. Each type is characterized by “information units”, which are abstract systems whose states represent the simplest information of each type. The information units define the “natural” representation of the information. For deterministic information the information unit is the bit, whose states are symbolized by 0 and 1. Information units can be put together to form larger systems and can be processed with basic operations acting on a small number of them at a time.

Inner product. The defining operation of a Hilbert space. In a finite dimensional Hilbert space with a chosen orthonormal basis $\{e_i : 1 \leq i \leq n\}$, the inner product of two vectors $x = \sum_i x_i e_i$ and $y = \sum_i y_i e_i$ is given by $\sum_i \overline{x}_i y_i$. In the standard column representation of the two vectors, this is the number obtained by computing the product of the conjugate transpose of x with y. For real vectors, this agrees with the usual “dot” product. The inner product of x and y is often written in the form $\langle x, y\rangle$. Pure quantum states are unit vectors in a Hilbert space. If $|\phi\rangle$ and $|\psi\rangle$ are two quantum states expressed in the ket-bra notation, there inner product is given by $(|\phi\rangle)^\dagger\, |\psi\rangle = \langle\phi|\psi\rangle$.

Ket. A state expression of the form $|\psi\rangle$ representing a quantum state. Usually $|\psi\rangle$ is thought of as a superposition of members of a logical state basis $|i\rangle$. One way to think about the notation is to consider the two symbols "$|$" and "$\rangle$" as delimiters denoting a quantum system and ψ as a symbol representing a state in a standard Hilbert space. The combination $|\psi\rangle$ is the state of the quantum system associated with ψ in the standard Hilbert space via a fixed isomorphism. In other words, one can think of $\psi \leftrightarrow |\psi\rangle$ as an identification of the quantum system's state space with the standard Hilbert space.

Linear extension of an operator. The unique linear operator that implements a map defined on a basis. Typically, we define an operator U on a quantum system only on the logical states $U : |i\rangle \mapsto |\psi_i\rangle$. The linear extension is defined by $U(\sum_i \alpha_i |i\rangle) = \sum_i \alpha_i |\psi_i\rangle$.

Logical states. For quantum systems used in information processing, the logical states are a fixed orthonormal basis of pure states. By convention, the logical basis for qubits consists of $|0\rangle$ and $|1\rangle$. For larger dimensional quantum systems, the logical basis is often indexed by the whole numbers, $|0\rangle, |1\rangle, |2\rangle, \ldots$. The logical basis is often also called the "computational" basis, or sometimes, the "classical" basis.

Measurement. The process used to extract classical information from a quantum system. A general projective measurement is defined by a set of projectors P_i satisfying $\sum_i P_i = \mathbb{1}$ and $P_i P_j = \delta_{ij} P_i$. Given the quantum state $|\psi\rangle$, the outcome of a measurement with the set $\{P_i\}_i$ is one of the classical indeces i associated with a projector P_i. The index i is the measurement outcome. The probability of outcome i is $p_i = |P_i|\psi_i\rangle|^2$, and given outcome i, the quantum state "collapses" to $P_i|\psi_i\rangle/\sqrt{p_i}$.

Mixture. A probabilistic combination of pure states of a quantum system. Mixtures can be represented without redundancy with density operators. Thus a mixture is of the form $\sum_i \lambda_i |\psi_i\rangle\langle\psi_i|$, with $\lambda_i \geq 0$, $\sum_i \lambda_i = 1$ being the probabilities of the states $|\psi_i\rangle$. This expression for mixtures defines the set of density operators, which can also be characterized as the set of operators ρ satisfying $\text{tr}(\rho) = 1$ and for all $|\psi\rangle$, $\langle\psi|\rho|\psi\rangle \geq 0$ ("positive semidefinite operator").

Network. In the context of information processing, a network is a sequence of gates applied to specified information units. We visualize networks by drawing horizontal lines to denote the time line of an information unit. The gates are represented by graphical elements that intercept the lines at specific points. A realization of the network requires applying the gates to the information units in the specified order (left to right).

Operator. A function that transforms the states of a system. Operators may be restricted depending on the system's properties. For example, in talking about operators acting on quantum systems, one always assumes that they are linear.

Oracle. An information processing operation that can be applied. A use of the oracle is called a "query". In the oracle model of computation, a standard model is extended to include the ability to query an oracle. Each oracle query is assumed to take one time unit. Queries can reduce the resources required for solving problems. Usually, the oracle implements a function or solves a problem not efficiently implementable by the model without the oracle. Oracle models are used to compare the power of two models of computation when the oracle can be defined for both models. For example, in 1994, D. Simon showed that quantum computers with a specific oracle $\mathcal{O}$ could efficiently solve a problem that had no efficient solution on classical computers with access to the classical version of $\mathcal{O}$. At the time, this result was considered to be the strongest evidence for an exponential gap in power between classical and quantum computers.

Overlap. The inner product between two quantum states.

Pauli operators. The Hermitian matrices $\sigma_x, \sigma_y, \sigma_z$ acting on qubits, which are two-level quantum systems. They are defined in Eq. 12. It is often convenient to consider the identity operator to be included in the set of Pauli operators.

Polynomial resources. To say that an algorithm computing the function $f(x)$, where x is a bit string, uses polynomial resources (in orther words, "is efficient") means that the number of steps required to compute $f(x)$ is bounded by $|x|^k$ for some fixed k. Here $|x|$ denotes the length of the bit string x.

Probabilistic bit. The basic unit of probabilistic information. It is a system whose state space consists of all probability distributions over the two states of a bit. The states can be thought of as describing the outcome of a biased coin flip before the coin is flipped.

Probabilistic information. The type of information obtained by extending the state spaces of deterministic information to include arbitrary probability distributions over the deterministic states. This is the main type of classical information to which quantum information is compared.

Probability amplitude. The squared norm of an amplitude with respect to a chosen orthonormal basis $\{|i\rangle\}$. Thus, the probability amplitude is the probability with which the state $|i\rangle$ is measured in a complete measurement that uses this basis.

Product state. For two quantum systems A and B, product states are of the form $|\psi\rangle_{\mathsf{A}}|\phi\rangle_{\mathsf{B}}$. Most states are not of this form.

Program. An algorithm expressed in a language that can be understood by a particular type of computer.

Projection operator. A linear operator P on a Hilbert space that satisfies $P^2 = P^\dagger P = P$. The projection onto a subspace V with orthogonal complement W is defined as follows: If $x \in V$ and $y \in W$, then $P(x+y) = x$.

Pseudo-code. An semi-formal computer language that is intended to be executed by a standard "random access machine", which is a machine model with a central processing unit and access to a numerically indexed unbounded memory. This machine model is representative of the typical one-processor computer. Pseudo-code is similar to programming languages such as BASIC, Pascal, or C, but does not have specialized instructions for human interfaces, file management, or other "external" devices. Its main use is to describe algorithms and enable machine-independent analysis of the algorithms' resource usage.

Pure state. A state of a quantum system that corresponds to a unit vector in the Hilbert space used to represent the system's state space. In the ket notation, pure states are written in the form $|\psi\rangle = \sum_i \alpha_i|i\rangle$, where the $|i\rangle$ form a logical basis and $\sum_i |\alpha_i|^2 = 1$.

Quantum information. The type of information obtained when the state space of deterministic information is extended by normalized superpositions of deterministic states. Formally, each deterministic state is identified with one of an orthonormal basis vector in a Hilbert space and normalized superpositions are unit-length vectors that are expressible as complex linear sums of the chosen basis vectors. It is convenient to extend this state space further by permitting probability distributions over the quantum states (see the entry for "mixtures"). This extension is still called quantum information.

Qubit. The basic unit of quantum information. It is the quantum extension of the deterministic bit, which implies that its state space consists of the unit-length vectors in a two dimensional Hilbert space.

Read-out. A method for obtaining human-readable information from the state of a computer. For quantum computers, read-out refers to a measurement process used to obtain classical information about a quantum system.

Reversible gate. A gate whose action can be undone by a sequence of gates.

Separable state. A mixture of product states.

States. The set of states for a system characterizes the system's behavior and possible configurations.

Subspace. For a Hilbert space, a subspace is a linearly closed subset of the vector space. The term can be used more generally for a system Q of any information type: A subspace of Q or, more specifically, of the state space of Q is a subset of the state space that preserves the properties of the information type represented by Q.

Superposition principle. One of the defining postulates of quantum mechanics according to which if states $|1\rangle, |2\rangle, \ldots$ are distinguishable then $\sum_i \alpha_i |i\rangle$ with $\sum_i |\alpha_i|^2 = 1$ is a valid quantum state. Such a linear combination is called a normalized superposition of the states $|i\rangle$.

System. An entity that can be in any of a specified number of states. An example is a desktop computer whose states are determined by the contents of its various memories and disks. Another example is a qubit, which can be thought of as a particle whose state space is identified with complex, two-dimensional, length-one vectors. Here, a system is always associated with a type of information that determines the properties of the state space. For example, for quantum information the state space is a Hilbert space. For deterministic information, it is a finite set called an alphabet.

Unitary operator. A linear operator U on a Hilbert space that preserves the inner product. That is, $\langle Ux, Uy\rangle = \langle x, y\rangle$. If U is given in matrix form, then this expression is equivalent to $U^\dagger U = \mathbb{1}$.

Universal set of gates. A set of gates that satisfies the requirement that every allowed operation on information units can be implemented by a network of these gates. For quantum information, it means a set of gates that can be used to implement every unitary operator. More generally, a set of gates is considered universal if for every operator U, there are implementable operators V arbitrarily close to U.