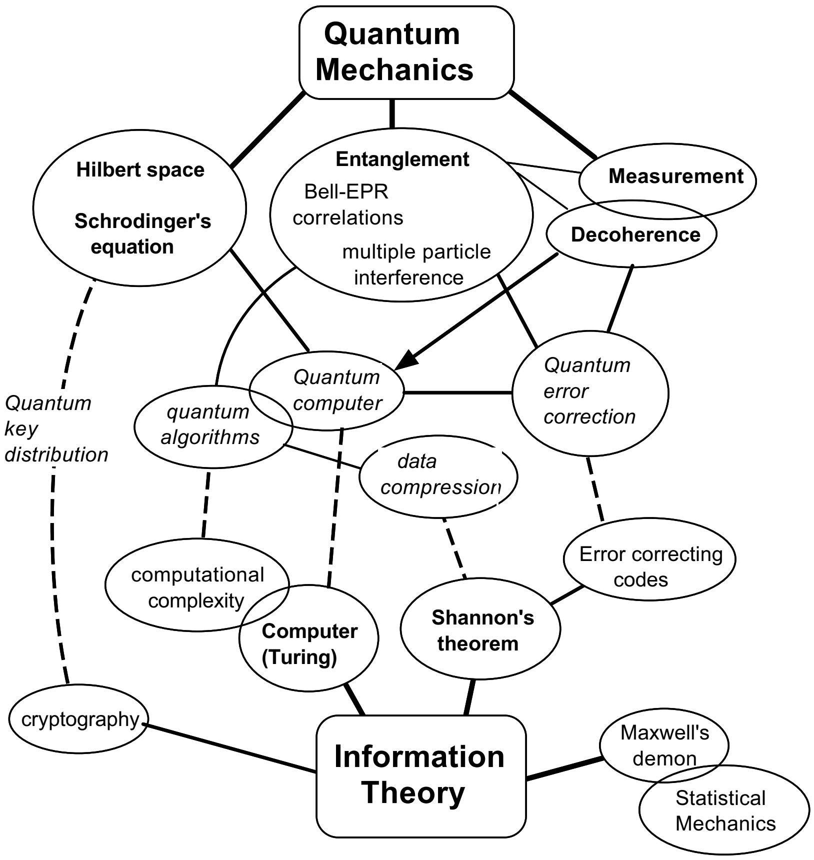

Figure 14: Overview of the different topics in connection with quantum computing.

5 The quantum computer: principles

We have touched so far a list of distinct topics having to do with quantum computing, many more are actually relevant in this field, see Fig. 14. Let us now discuss the functioning of quantum computer, in theory.

5.1 Quantum parallelism

Let us consider a general quantum gate, as the one illustrated in Fig. 10, which maps some input $|x\rangle$ to some output-function $|f(x)\rangle$, see Fig. 4 via

$$|x\rangle \otimes |0\rangle \;\longmapsto\; |x\rangle \otimes |f(x)\rangle \ , \tag{17}$$

the first ket describing the state of the input register and the second the state of the output register.

One of the most powerful features of quantum computation is *quantum parallelism*. The superposition principle of quantum mechanics allows us to prepare the computer in a coherent superposition of a set I in the Hilbert-space $\mathcal{H}$ of $|x\rangle$ of input states. After a single run, all of the corresponding outputs $f(x)$ appear in the final state, according to the time evolution

$$\sum_{x\in I} |x\rangle \otimes |0\rangle \;\longmapsto\; \sum_{x\in I} |x\rangle \otimes |f(x)\rangle \ . \tag{18}$$

Superposition of all states – A simple way to construct a linear superposition of *all* states in $\mathcal{H}$ uses the Hadamard gates. Let $|x\rangle = \otimes_{i=1}^{n}|x_i\rangle$ and $|0\rangle = \otimes_{i=1}^{n}|0\rangle$.

$$\bigotimes_{i=1}^{n} H_i|0\rangle \;=\; \frac{1}{\sqrt{2}^n}\sum_{x\in\mathcal{H}} |x\rangle$$

is then a linear superposition, with equal weights, of all states in $\mathcal{H}$, where H_i is the Hadamard gate (6) acting on the qubit i.

No free lunch – Unfortunately, Eq. (18) is no "pay one, take n". In fact, the result is an entangled state of the input and output registers and there is no single measurement allowing us to extract from it *all* the computed values of $f(x)$. However, it may well be possible to distill from this final state some *global property* of the function, thus exploiting quantum parallelism. One of the most famous examples was presented by D. Deutsch *et al.* [15], who showed that the problem of classifying a given function

$$f : \{0, \ldots, n\} \;\longrightarrow\; \{0, 1\} \tag{19}$$

as "not constant" or "not balanced" [§] can be solved in polynomial time by means of a quantum computer (the time required by a classical computer growths exponential).

Quantum algorithms – Above is typical example: Quantum computers lead to exponential speedup only for certain problems (algorithms). Examples are the Deutsch-Jozsa algorithm mentioned above, the Grover algorithm [31] for database search and Shor's algorithm [12] for factorization of large integer numbers .

5.2 Quantum software vs. quantum hardware

A normal computer is composed by hardware, doing the computation, and software, which tells the computer what to do. For quantum computers this distinction does not necessary hold. We can consider to kinds of concepts for functioning quantum computers which we will discuss now. The simplest concepts involves time-dependent Hamiltonians, the more advanced (and Feynmann's original proposal) considers time-independent Hamiltonian-evolution.

[§] "Not balanced" means that the number one's and zero's on output is not the same.

5.2.1 Time-dependent activation of quantum gates.

Every quantum gate U_n corresponds via the Schrödinger equation to a microscopic Hamiltonian H_n:

$$U_n \;=\; \exp\left(-itH_n\right) \;. \tag{20}$$

Hamiltonian for the NOT-gate – Let us give an example. We consider a single qubit and the NOT-gate, the Hamiltonian $H_{\rm NOT}$ is defined by

$$e^{i\Delta t H_{\rm NOT}} \;=\; {\rm NOT} \;=\; \left(\begin{array}{cc} 0 & 1 \\ 1 & 0 \end{array}\right) \;,$$

where Δt it time interval for propagation. We will generally set $\Delta t = 1$. The Hamiltonian which does the job is

$$H_{\rm NOT} \;=\; \frac{\pi}{2}\left(\begin{array}{cc} 1 & 0 \\ 0 & 1 \end{array}\right) - \pi\left(\begin{array}{cc} 0 & 1 \\ 1 & 0 \end{array}\right) \;, \tag{21}$$

since

$$e^{-iH_{\rm NOT}} \;=\; e^{-i\pi/2}\left(\begin{array}{cc} 1 & 0 \\ 0 & 1 \end{array}\right)\cdot\left[\cos(\frac{\pi}{2})\left(\begin{array}{cc} 1 & 0 \\ 0 & 1 \end{array}\right) + i\sin(\frac{\pi}{2})\left(\begin{array}{cc} 0 & 1 \\ 1 & 0 \end{array}\right)\right] \;=\; \left(\begin{array}{cc} 0 & 1 \\ 1 & 0 \end{array}\right) \;.$$

The Hamiltonian is defined only modulo multiples of $2\pi/(\Delta t)$.

Quantum circuit – Let us consider a simple quantum circuit, like the decomposition of the controlled-U gate illustrated in Fig. 13. In Sec. 4.3 we had proven the relation

$$U_{CU} \;=\; \prod_{n=1}^{N} U_n \;, \tag{22}$$

where the U_n are the A, B, C, the CNOT's and the CPHASE gate respectively. It would be tempting to use Eq. (20) and write in general

$$U^{\rm tot} \;=\; \prod_{n=1}^{N} e^{-itH_n} \;=\; \exp\big(-it\sum_n H_n\big) \;.$$

This relation is actually incorrect. The product of quantum-gates in Eq. (22), and actually everywhere else up-to-now, assumed that we have performed the unitary transformations one after another!

Time dependent Hamiltonians – In order to make quantum-mechanical sense of Eq. (22) we have to assume that the H_n are active only for certain time-intervals:

$$H_n \;\longrightarrow\; \Big(\theta(t_{n-1}) - \theta(t_n)\Big)\, H_n \;, \tag{23}$$

($\theta(x) = 0/1$ for $x < 0$ and $x > 0$ respectively.) where we have assumed as series of time-intervals

$$t_0 \;<\; t_1 \;<\; \ldots \;<\; t_N \;.$$

In principle there could be also idle times in between the action of the individual quantum gates, which we can model by appropriated $H_n \equiv 0$.

Classical computer controls quantum computer – Using Eq. (23) we can now write correctly

$$U^{\mathrm{tot}} \;=\; U^{\mathrm{tot}}(t_N, t_0) \;=\; \prod_{n=1}^{N} \mathrm{e}^{-i(t_n - t_{n-1})H_n} \;. \tag{24}$$

This means that we have activated *from the outside* the individual Hamiltonians H_n. The software for the quantum computer resides consequently in the lab-computer which controls the experiment, i.e. in a NMR quantum computer the computer which controls the RF-impuls sequence. We need consequently a classical computer to control the quantum computer.

Software vs. hardware – The distinction in between soft- and hardware blurs for a quantum-computer. In principle we would think of logical gates, like AND in classical computers or the CNOT in quantum computers to be part of the hardware. In the NMR-implementation the logical gates do not exist as solid-state devices, they are "made on the run by transient electromagnetic pulses".

Addressability – In a universal quantum computer we would like to be able to address all possible quantum gates (actually to create them from the outside) individually. The number of addressable gates would then scale linearly with the number of qubits N_q. Every single gate would then be activated from the outside one after another, see Eq. (24), total time processing time would therefor increase at least linearly with the number of qubits. This is undesirable. Current concepts for quantum computers are mostly based on parallel gates H_n, acting in identical manner on many qubits in parallel, like the classical vector-processor implemented in the original Gray-"supercomputers". This concept makes, off course, only sense when all qubits are part of a single input, i.e. when they can be treated identically.

5.2.2 Fixed-wired quantum circuits

Let us consider now the case that we are able to build hard-wired quantum circuits. In this case the Hamilton operators H_n of the individual quantum gates $n = 1, \ldots, N$ which make up the circuit are time-independent. From the knowledge of the individual gate-Hamiltonians H_n we would like to interfere the properties of the total circuit $U^{\mathrm{tot}}(t, 0)$. This is non-trivial problem and we will devote the next section to study this problem. Let us note at this point an important consequence: The properties of *quantum-hardware* are non-trivial functions of its individual components. The up-to-date practical routes towards a quantum computer therefore involve "programmable gates' like (23)

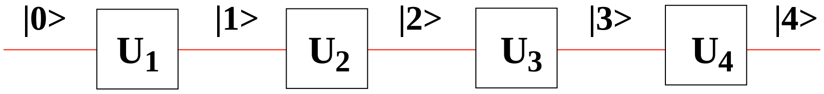

Figure 15: Illustration of a quantum circuit with four quantum gates.

5.3 Quantum hardware

Hilbert space – An Illustration of a general quantum circuit is given in Fig. 15. How large is the Hilbert-space? Let n be the number of qubits involved (input/output and at every step). Let N be the number of elementary quantum gates. "Elementary" means here that we know the respective Hamiltonians H_n. The dimensional of the Hilbert space is now

$$2^{nN} ,$$

the wavefunction is coherent all over the circuit.

Complex problem – For fixed-wired quantum circuits we have to solve for the dynamics of a complex quantum-mechanical problem. In order to do so we introduce one additional concept from quantum mechanics, the notion of creation and anhilation operators.

5.3.1 Creation and anhilation operators

We use quantum mechanical operators in order to describe quantum gates in general. To this end we first map the bits to qubits of which we only use the basis states $|0\rangle = |\uparrow\rangle$ and $|1\rangle = |\downarrow\rangle$. From Sec. 1.5.4 we know how to flip these states by the spin raising and lowering operators:

$$S^+|\downarrow\rangle \;=\; |\uparrow\rangle\,; \qquad S^-|\uparrow\rangle \;=\; |\downarrow\rangle\,.$$

Here it is convenient to use a slightly different language: if we interpret $|0\rangle_a$ and $|1\rangle_a$ as absence and presence of a particle at qubit ("site") a, respectively, and if we use the language of second quantization, that is, creation and annihilation operators:

$$a^\dagger|0\rangle \;=\; |1\rangle\,, \qquad a|1\rangle \;=\; |0\rangle,$$

and the number operator

$$a^\dagger a|x\rangle \;=\; x|x\rangle \qquad (x=0,1)\,.$$

Note that $a^\dagger a + aa^\dagger = 1$, that is, these particles resemble fermions in that no more than one of them can be present.

More than one qubit – Introducing the same notation for the additional qubits, like b and c in Fig. 16, it is very easy to write down the unitary transformation for the elementary gates:

$$\begin{aligned} \text{NOT}\ (a) &= (a + a^\dagger)\ , \\ \text{CNOT}\ (a,b) &= (b + b^\dagger) a^\dagger a + 1_b (1_a - a^\dagger a) \\ = (b + b^\dagger) a^\dagger a + 1_b a a^\dagger &= (b + b^\dagger - 1_b) a^\dagger a + 1 \end{aligned}$$

where a is the control bit. In order to avoid sign trouble we assume that operators for different qubits (sites) commute, that is, behave like Bosons in this respect. Together with the Fermi-like on-site commutation relation this shows that these particles are neither Bosons nor Fermions. (Sometimes they are called "hard-core Bosons" or "Paulions".) The mixed commutator structure is cumbersome if standard many-body calculational techniques are to be applied (which we do not intend to).

After the above exercises it is easy to code the Toffoli gate CCNOT as an operator:

$$\theta^{(3)}(a,b,c) = 1 + (c^\dagger + c - 1)\, a^\dagger a\, b^\dagger b\ .$$

Full time evolution – The operator for the half adder can be written down reading the diagram in Fig. 16 starting from the left and writing down the elementary operators from the right:

$$\text{CNOT}\ (a,b)\, \theta^{(3)}(a,b,c) \equiv U_2 U_1 |a,b,c\rangle = \exp\left(-it\mathbf{H}t\right) |a,b,c\rangle\ .$$

Are there a Hamiltonian $\mathbf{H}$ and a time t which satisfy this equation? Obviously this is no easy question, since

$$\exp\left(-i\mathbf{H}t\right) = \mathbf{1} + (-i\mathbf{H}t) + \frac{1}{2}(-i\mathbf{H}t)^2 + \frac{1}{6}(-i\mathbf{H}t)^3 + \cdots$$

and thus the right hand side of the above equation for the half-adder will be a superposition of states where $\mathbf{H}$ has acted any number of times, from zero to infinity.

5.3.2 Hamiltonians do not commute generally

Before we come to the solution of the general time-evolution problem for quantum-computing hardware, we will show that a simple solution is not possible, do to the non-trivial commutation relations in between different elementary Hamiltonians. For simplicity we consider only a single qubit. The Hamiltonian for the NOT gate is (21):

$$H_{\text{NOT}} = \pi \left(a + a^\dagger\right) - \frac{\pi}{2} 1_a \tag{25}$$

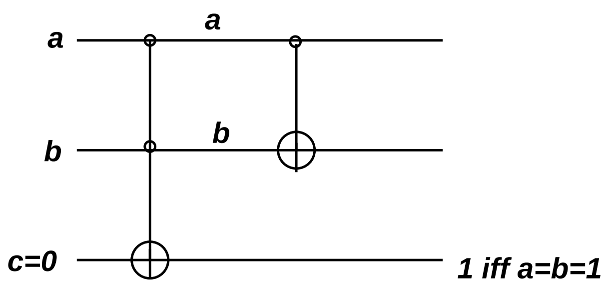

Figure 16: An adder (half adder) circuit with an CCNOT (Toffoli gate) and a CNOT.

in second quantization. We now consider also the Hadamard gate

$$U_{\mathrm{H}} \;=\; \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \;=\; \frac{1}{\sqrt{2}}\left(a+a^\dagger\right) \;+\; \frac{1}{\sqrt{2}}\left(1-2\,a^\dagger a\right)\;. \qquad (26)$$

Eigenstates of the Hadamard gate– The NOT and the Hadamard gate do not commute: $U_{\mathrm{NOT}}U_{\mathrm{H}} \neq U_{\mathrm{H}}U_{\mathrm{NOT}}$. We therefore expect that the respective Hamiltonians do not commute either. We consider first the eigenstates $\mathbf{x}_\pm$

$$\mathbf{x}_\pm \;=\; \frac{1}{\sqrt{4\pm 2\sqrt{2}}}\begin{pmatrix} 1\pm\sqrt{2} \\ 1 \end{pmatrix} \;=\; \frac{1}{\sqrt{4\pm 2\sqrt{2}}}\left((1\pm\sqrt{2})1_a + a^\dagger\right)|0\rangle$$

of the Hadamard gate, with eigenvalues $\lambda_\pm = \pm 1$. Consequently we can write the Hadamard gate as

$$U_{\mathrm{H}} \;=\; |x_+\rangle\langle x_+| \;-\; |x_-\rangle\langle x_-| \;=\; x_+^\dagger x_+ \;-\; x_-^\dagger x_- \;=\; e^{-i\pi x_-^\dagger x_-}\;,$$

where the $x_\pm^\dagger$ and the $x_\pm$ are with

$$x_\pm^\dagger \;=\; \frac{1}{\sqrt{4\pm 2\sqrt{2}}}\left((1\pm\sqrt{2})1_a + a^\dagger\right)$$

the respective creation/anhilation operators for the eigenstates of the Hadamard gate.

Hamiltonian for the Hadamard gate– The Hamiltonian for the Hadamard gate is consequently

$$H_{\mathrm{H}} \;=\; \pi\, x_-^\dagger x_- \;=\; \frac{\pi}{4\pm 2\sqrt{2}}\left((1\pm\sqrt{2})1_a + a^\dagger\right)\left((1\pm\sqrt{2})1_a + a\right)\;, \qquad (27)$$

it does not commute with H_{NOT} since

$$[a+a^\dagger, a^\dagger a] \;=\; a - a^\dagger\;.$$

We therefor have

$$U_{\mathrm{NOT}}U_{\mathrm{H}} \;=\; \mathrm{e}^{-iH_{\mathrm{NOT}}}\mathrm{e}^{-iH_{\mathrm{H}}} \;\neq\; \mathrm{e}^{-i(H_{\mathrm{NOT}}+H_{\mathrm{H}})}\;.$$

The last equation would hold only if $[H_{\mathrm{NOT}}, H_{\mathrm{H}}] = 0$.

5.3.3 Full time evolution

The full time evolution of a general fixed-wired quantum gate like the one in Fig. 15 can be computed by the introduction of an auxiliary particle. The trick is to keep a record of which of the U operators have already acted on the input state. This bookkeeping is done by auxiliary (or "slave") particles.

Suppose you want to calculate

$$|k\rangle \;=\; U_k U_{k-1} \cdots U_1 |0\rangle \;\equiv\; \mathrm{e}^{iH^{tot}} |0\rangle$$

for an n-qubit state $|\psi_i\rangle$. Here we consider the case where the U_k are some elementary quantum gates. Since most, but not all, elementary quantum gates are hermitian, in addition to be unitary, we consider $U_k = U_k^+ = U^{-1}$.

Program counter qubits – We introduce a "chain" of $k+1$ new *program counter qubits* named $i = 0 \cdots k$, with corresponding creation and annihilation operators q_i,$q_i^\dagger$. The desired Hamiltonian then reads

$$\mathbf{H} \;\equiv\; H^{tot} \;=\; \sum_{i=0}^{k-1} \left(q_{i+1}^\dagger q_i \, U_{i+1} + U_{i+1} \, q_i^\dagger q_{i+1} \right) \;=\; \sum_{i=0}^{k-1} \left(q_{i+1}^\dagger q_i + q_i^\dagger q_{i+1} \right) U_{i+1} \;,$$

where we have used $U_k = U_k^+$ and that the q operators commute with the A operators. Note that the number of "q particles" $\sum_{i=0}^{k} q_i^\dagger q_i$ is a constant; we will be interested exclusively in the case of a single particle.

Program counter qubits hopping – The action of the Hamiltonian is represented pictorially in Fig. 15:

Whenever the program counter particle moves from site i to $i+1$ or vice versa the operator A_{i+1} acts on the "register qubits" where the calculation is performed. The calculation starts with the register qubits in the input state $|0\rangle = |\psi_0\rangle$ and the program counter particle at site 0. The action of $\mathbf{H}^\nu$ then yields

$$\mathbf{H}^\nu |1000\cdots 0\rangle |\psi_0\rangle \;=\; \mathbf{H}^{\nu-1} |0100\cdots 0\rangle U_1 |\psi_0\rangle \;=$$
$$=\; \mathbf{H}^{\nu-2} \left(|0010\cdots 0\rangle U_2 U_1 |\psi_0\rangle \;+\; |1000\cdots 0\rangle \underbrace{U_1 U_1}_{\mathbf{1}} |\psi_0\rangle \right) \;=\; \cdots$$

since $U_i^2 = U_i U_i^\dagger = U_i U_i^{-1} = 1$. We see that if the program counter particle is at site l, the last operator which has been active is U_l:

$$|0 \cdots \underbrace{1}_{l} \cdots 0\rangle U_l \cdots |\psi_0\rangle.$$

The next application of $\mathbf{H}$ then leads to two possibilities:

- $l \to l-1; U_l$ is squared (=erased)
- $l \to l-1; U_{l+1}$ is prepended to the string of U operators.

(This argument can of course be transformed into a rigorous proof by induction.) We conclude that if our final state contains a component with the counter particle at site k, we are finished. We only have to project out the desired component:

$$\alpha|00\cdots 01\rangle|\psi_k\rangle \;=\; q_k^\dagger q_k \exp\left(-i\mathbf{H}t\right)|100\cdots 0\rangle|\psi_0\rangle \;, \tag{28}$$

where α is a normalization factor which depends on t and the (infinite) number of times the program counter particle hops forth and back. Eq. (28) is a proof, by construction, that the time evolution of an arbitrary circuit of elementary quantum gates exists. For the practical implementation, one need than to find physical processes which do exactly correspond to H^{tot}.

Exercise – Construct for $n = 1$ and $k = 2$ the Hamiltonian H^{tot} for $U^{tot} = U_{\mathrm{NOT}}U_{\mathrm{H}}$.