Quantum information theory (20110401)

Lecturer: Jens Eisert

Chapter 10: Quantum computational models

Contents

10 Quantum computational models **5**
10.1 Adiabatic quantum computing 5
10.2 Measurement-based quantum computing 7
10.2.1 Cluster states 7
10.2.2 Simulating quantum circuits in the cluster-state model 9
10.2.3 A few final words 11
10.3 Other models of quantum computing 11
10.3.1 Quantum computing by Hamiltonian evolution 11
10.3.2 Dissipative quantum computing 12

Chapter 10

Quantum computational models

10.1 Adiabatic quantum computing

So far, we have encountered the *circuit model* for quantum computing, and have made it look as if it was the only model for quantum computing. It is not. There is a large class of models for quantum computing. At the end, all a model for quantum computing needs to be able to produce is an efficient simulation of any other model for quantum computation. And the latter holds true in particular for the circuit model for quantum computation. A particularly intriguing other model for quantum computing is the so-called model of *adiabatic quantum computing*. In 2000, Farhi, Goldstone, Gutmann, and Sipser introduced a new concept to the study of quantum algorithms, based on the adiabatic theorem of quantum mechanics. The idea is the following: let $f : \{0,1\}^N \longrightarrow \mathbb{R}$ be a cost function of which we would like to find the global minimum, assumed in $x \in \{0,1\}^N$. In fact, any local combinatorial search problem can be formulated in this form. For simplicity, suppose that this global minimum is unique. Introducing the *problem Hamiltonian*

$$H_T = \sum_{z\in\{0,1\}^N} f(z)|z\rangle\langle z|, \tag{10.1}$$

the problem of finding the $x \in \{0,1\}^N$ where f attains its minimum amounts to identifying the eigenvector $|x\rangle$ of H_T corresponding to the smallest eigenvalue $f(x)$, i.e., the *ground state energy* associated with H_T. But how does one find the ground state in the first place? The key idea is to consider another Hamiltonian H_0, with the property that the system can easily be prepared in its ground state, which is again assumed to be unique. One then interpolates between the two Hamiltonians, for example linearly

$$H(t) = \frac{t}{T}H_T + \big(1 - \frac{t}{T}\big)H_0, \tag{10.2}$$

with $t \in [0, T]$, where T is the *run time* of the adiabatic quantum algorithm. This Hamiltonian governs the time evolution of the quantum state of the system from time $t = 0$ until $t = T$. According to the Schrödinger equation, the state vector evolves as $i\partial_t|\psi(t)\rangle = H(t)|\psi(t)\rangle$. In a last step one performs a measurement in the computational basis. If one obtains the outcome associated with $|x\rangle$, then the measurement result is just x, the minimal value of the function f. In this case the probabilistic algorithm is successful, which happens with the *success probability* $p = |\langle x|\psi(T)\rangle|^2$.

What are the requirements for such an algorithm to work, i.e., to result in x with a large success probability? The answer to this question is provided by the *quantum adiabatic theorem*: If the Hamiltonian $H(t)$ exhibits a non-zero spectral gap between the smallest and the second-to-smallest eigenvalue for all $t \in [0, T]$, then the final state vector $|\psi(T)\rangle$ will be close to the state vector $|x\rangle$ corresponding to the ground state of H_T, if the interpolation happens sufficiently slowly, meaning that T is sufficiently large. The initial state is then said to be adiabatically transferred to arbitrary accuracy into the desired ground state of the problem Hamiltonian, which encodes the solution to the problem. The typical problem of encountering local minima that are distinct from the global minimum can in principle not even occur. This kind of quantum algorithm is referred to as an *adiabatic algorithm*.

Needless to say, the question is how large the time T has to be chosen. Let us denote with

$$\Delta = \min_{t\in[0,T]} (E_t^{(0)} - E_t^{(1)}) \tag{10.3}$$

the minimal spectral gap over the time interval $[0, T]$ between the smallest $E_t^{(0)}$ and the second-to-smallest eigenvalue $E_t^{(1)}$ of $H(t)$, associated with eigenvectors $|\psi_t^{(0)}\rangle$ and $|\psi_t^{(1)}\rangle$, respectively, and with

$$\Theta = T \max_{t\in[0,T]} |\langle\psi_t^{(1)}|\partial_t H(t)|\psi_t^{(0)}\rangle| = \max_{t\in[0,T]} |\langle\psi_t^{(1)}|(H_T - H_0)|\psi_t^{(0)}\rangle. \tag{10.4}$$

Then, according to the quantum adiabatic theorem, the success probability satisfies

$$p = |\langle\psi_T^{(0)}|\psi(T)\rangle|^2 \geq 1 - \varepsilon^2 \tag{10.5}$$

if

$$T\varepsilon \geq \frac{\Theta}{\Delta^2}. \tag{10.6}$$

The quantity Θ is typically polynomially bounded in N for the problems one is interested in, so the crucial issue is the behaviour of the minimal gap Δ. Time complexity is now quantified in terms of the run time T of the adiabatic algorithm. If one knew the spectrum of $H(t)$ at all times, then one could immediately see how fast the algorithm can be performed. Roughly speaking, the larger the gap, the faster the algorithm can be implemented. The problem is that the spectrum of $H(t)$, which can be represented as a $2^N \times 2^N$ matrix, is in general unknown. Even to find lower bounds for the minimal spectral gap is extraordinarily difficult, unless a certain symmetry highly simplifies the problem of finding the spectrum. After all, in order for the

Hamiltonian to be ‘reasonable’, it is required that it is *local*, i.e., it is a sum of operators that act only on a bounded number of qubits in N. This is a very natural restriction, as it means that the physical interactions involve always only a finite number of quantum systems. Note that an indication whether the chosen run time T for an adiabatic algorithm was appropriate, one may start with the initial Hamiltonian and prepare the system in its ground state, interpolate to the problem Hamiltonian and – using the same interpolation – back to the original Hamiltonian. A necessary condition for the algorithm to have been successful is that finally, the system is to a good approximation in the ground state of the initial Hamiltonian. This is a method that should be accessible to an experimental implementation.

Adiabatic algorithms are known to reproduce the quadratic speedup in the Grover algorithm for unstructured search problems. But adiabatic algorithms can also be applied to other instances of search problems: Adiabatic algorithms have been compared with simulated annealing algorithms, finding settings in which the quantum adiabatic algorithm succeeded in polynomial time, but for simulated annealing exponential time was necessary. There is after all some numerical evidence that for structured NP hard problems like MAX CLIQUE and 3-SAT, it may well be that adiabatic algorithms offer an exponential speedup over the best classical algorithm, again, assuming that $P \neq NP$. In fact, it can be shown that adiabatic algorithms can be efficiently simulated on a quantum computer based on the quantum circuit model, provided that the Hamiltonian is local in the above sense (see also the subsequent section). Hence, whenever an efficient adiabatic algorithm can be found for a specific problem, this implies an efficient quantum algorithm. The concept of adiabatic algorithms may be a key tool to establish new algorithms beyond the hidden subgroup problem framework.

10.2 Measurement-based quantum computing

This section is taken from M. A. Nielsen, quant-ph/0504097, with small modifications. Mike holds the copyright for this explanation, but it is quite beautiful.

10.2.1 Cluster states

A cluster-state computation begins with the preparation of a special entangled many-qubit quantum state, known as a *cluster state*, followed by an adaptive sequence of single-qubit measurements, which process the cluster, and finally read-out of the computation’s result from the remaining qubits. We now discuss each of these steps in detail.

The term “cluster state” refers not to a single quantum state, but rather to a family of quantum states. The idea is that to any graph G on n vertices we can define an associated n-qubit cluster state, by first associating to each vertex a corresponding qubit, and then applying a graph-dependent preparation procedure to the qubits, as described below. As an example, the following graph represents a six-qubit cluster state, The cluster state associated to the graph may be defined as the result of applying the following preparation procedure.

Cluster states:

1. Prepare each of the n qubits in the state vector $|+\rangle = (|0\rangle + |1\rangle)/\sqrt{2}$.
2. Apply controlled-PHASE gates between qubits whose corresponding graph vertices are connected.

Note that controlled-PHASE gates commute with one another, so we do not need to specify the order in which the gates are applied. Also, although we have described the preparation of the cluster in terms of applying quantum gates, later in the paper we briefly describe how to prepare clusters using measurements alone, and so the cluster-state model may be regarded as a truly measurement-only model of quantum computation.

Note that the states we have called cluster states are sometimes also known as *graph states*. Originally, the term "cluster state" was introduced by Raussendorf and Briegel to refer to the case where the graph G is a two-dimensional square lattice. This was the class of states which they showed could be used as a substrate for quantum computation. The term "graph state" originally referred to the family of states associated with more general graphs G. This distinction was blurred by the introduction of schemes for quantum computing based on Raussendorf and Briegel's ideas, but using different graphs.

Once the cluster state is prepared, the next step in the computation is to perform a sequence of *processing measurements* on the state. These measurements satisfy: (1) they are single-qubit measurements; (2) the choice of measurement basis may depend on the outcomes of earlier measurements, i.e., feedforward of classical measurement results is allowed; and (3) measurement results may be processed by a classical computer to assist in the feedforward, so the choice of basis may be a complicated function of earlier measurement results. Note that for the cluster-state computation to be *efficient* we must constrain the classical computation to be of polynomial size.

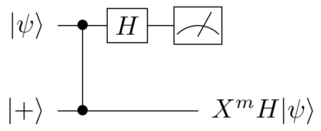
The output of the cluster-state computation may be defined in two different ways, both useful. The first is to regard the computation as having a *quantum state* as output, namely, the quantum state of the qubits which remain when the sequence of processing measurements has terminated. The second definition is to add a set of *read-out measurements*, a sequence of single-qubit measurements applied to the qubits which remain when the processing measurements are complete. In this case the output of the computation is a classical bit string.

A concrete example of these ideas is the following cluster-state computation:

Labels indicate qubits on which processing measurements occur, while unlabeled qubits are those which remain as the output of the computation when the processing measurements are complete. Note that the qubits are labeled by a positive integer n and a single-qubit unitary, which we refer to generically as U; here $U = HZ_{\pm\alpha_j}, HZ_{\pm\beta_j}$. The n label indicates the time-ordering of the processing measurements, with qubits having the same label capable of being measured in either order, or simultaneously. The time order is important, because it determines which measurement results can be fedforward to control later measurement bases. The U label indicates the basis in which the qubit is measured, denoting a rotation by the unitary U, followed by a computational basis measurement. Equivalently, a single-qubit measurement in the basis $\{U^\dagger|0\rangle, U^\dagger|1\rangle\}$ is performed. The $\pm$ notation in $HZ_{\pm\alpha_2}$ and $HZ_{\pm\beta_2}$ indicates that the choice of sign depends on the outcomes of earlier measurements, in a manner to be specified separately. We'll give an example of how this works later.

10.2.2 Simulating quantum circuits in the cluster-state model

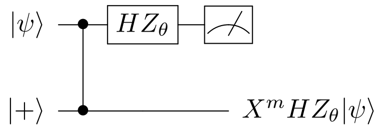
We now explain how quantum circuits can be simulated using a cluster-state computation. The key idea underlying the simulation is a simple circuit identity, sometimes known as one-bit teleportation,



(10.7)

Here, m is the outcome (zero or one) of the computational basis measurement on the first qubit. This identity may be verified by expanding $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$, so the state vector after the controlled-PHASE and Hadamard gates is $\alpha|+,+\rangle + \beta|-,-\rangle$, by the gate definitions given earlier. This state may be re-expressed as $(|0\rangle \otimes H|\psi\rangle + |1\rangle \otimes XH|\psi\rangle)/\sqrt{2}$, from which the result follows.

The identity of (10.7) is easily generalized to the following identity



(10.8)

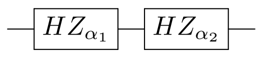
The proof is to note that Z_θ commutes with the controlled-PHASE gate, and thus the output of the circuit is the same as would have been output from the circuit in Equation (10.7) had $Z_\theta|\psi\rangle$ been input, instead of $|\psi\rangle$.

The proof of (10.8) is elementary, but the result is nonetheless remarkable. Observe that although the first qubit is measured, no quantum information is lost, for no matter what the measurement outcome, the posterior state vector of the second qubit is related by a known unitary transformation to the original input, $|\psi\rangle$.

It is tempting to regard this as unsurprising. After all, suppose we replaced the controlled-PHASE gate by a SWAP gate, which merely interchanges the state of the two qubits. Then we would not expect a measurement on the first qubit to destroy any quantum information, since all the quantum information would have been transferred from qubit one to qubit two *before* the measurement on qubit one.

However, this is not what happens, as can be seen from the fact that by varying the basis in which the first qubit is measured, i.e., by varying θ, we can vary the unitary transformation effected on the second qubit, without destroying any quantum information. This may be regarded as a generalization of the EPR effect, and may also be viewed as an instance of a quantum error-correcting code.

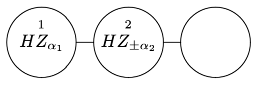
We can use (10.8) to explain how cluster-state computation can simulate quantum circuits. We begin by explaining how to simulate a single-qubit circuit of the form



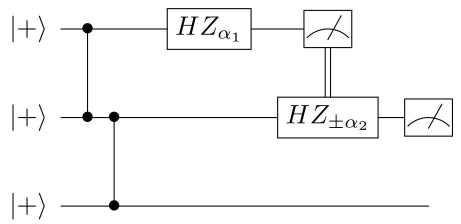
(10.9)

This apparently trivial case contains the most important ideas used in the general case. Note that we assume the qubit starts in the $|+\rangle$ state vector, and that single-qubit gates are of the form HZ_α. These assumptions are made for convenience, and do not cause any loss of generality, since it is clear that an arbitrary single-qubit circuit can be simulated using the ability to simulate these operations.

The cluster-state computation to simulate circuit (10.9) is

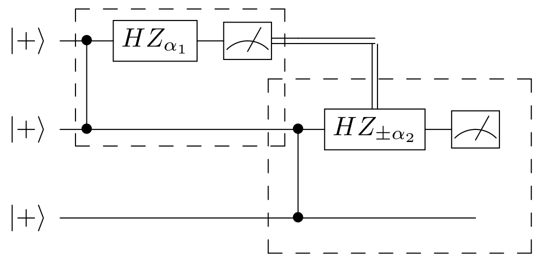


By definition, this cluster-state computation has an output equal to the output of the following quantum circuit[1]:



(10.10)

Equivalently, we can delay the operations on the second and third qubits until *after* the measurement on the first qubit is complete:



(10.11)

To determine the output, observe that the two highlighted boxes are both of the form of (10.8), and thus the output of the circuit is $X^{m_2}HZ_{\pm\alpha_2}X^{m_1}HZ_{\alpha 1}|+\rangle$, where

[1]Note that the double vertical lines emanating from the meter on the top qubit indicate classical feed-forward and control of later operations. We use this and similar notations often later in the paper.

m_1 and m_2 are the outputs of the measurements on the first and second qubits, respectively. Observe that feedforward can be used to choose the sign of $\pm\alpha_2$ so that $Z_{\pm\alpha_2}X^{m_1} = X^{m_1}Z_{\alpha_2}$. We also have $HX^{m_1} = Z^{m_1}H$, and thus the output may be rewritten as $X^{m_2}Z^{m_1}HZ_{\alpha_2}HZ_{\alpha_1}|+\rangle$, which, up to the known Pauli matrix $X^{m_2}Z^{m_1}$, is identical to the output of the conventional single-qubit quantum circuit (10.9).

This example generalizes easily to larger single-qubit circuits containing gates of the form HZ_α. The general proof strategy is: (1) rewrite the cluster-state computation in terms of an equivalent quantum circuit; (2) reinterpret the quantum circuit as a sequence of circuits of the form (10.8); (3) in the resulting expression for the output state, commute operators of the form X^m all the way to the left, using feedforward to choose signs on the terms of the form $Z_{\pm\alpha}$ to ensure that after commutation they are of the form Z_α. The result is a state which, up to a known Pauli matrix, is equivalent to the output of the single-qubit quantum circuit.

These ideas generalize also to multi-qubit quantum circuits. For example, the circuit: can be simulated using the above cluster-state computation. The proof of this

equivalence follows exactly the same lines as in the single-qubit case, and is only notationally more complicated. We omit the details, and suggest the interested reader fill them in.

Summing up, we have shown how the cluster-state model of computation can be used to efficiently simulate any quantum circuit whose inputs are all $|+\rangle$ state vectors, and whose gates are either controlled-PHASE gates, or gates of the form HZ_α. This set of resources is universal for quantum computation, and thus the cluster-state model is capable of efficiently simulating any quantum circuit. Conversely, it is straightforward to see that any cluster-state computation may be efficiently simulated in the quantum circuit model, and thus the two models are computationally equivalent.

10.2.3 A few final words

For many years, the cluster state based model was the only model for measurement-based quantum computing known. This is for good reason, as it is not quite obvious how to devise a model that is not based on commuting quantum gates. That said, later, many models have been found, including ones based on ground states of gapped local Hamiltonians. One can even find full computational phases of matter so that within every point in a phase of matter, one can do measurement-based quantum computing.

10.3 Other models of quantum computing

10.3.1 Quantum computing by Hamiltonian evolution

There are other models of quantum computing, other than adiabatic quantum computing and measurement-based quantum computing. One only has to be able to efficiently simulate the circuit model of quantum computing. There is the *Hamiltonian* *model of quantum computing*, where one measures a qubit after performing a unitary time evolution, with O acting on a single qubit only,

$$p = \mathrm{tr}(e^{-itH}\rho e^{itH}(O \otimes \mathbb{I})), \tag{10.12}$$

for a suitable $t > 0$, where

$$H = \sum_j h_j \tag{10.13}$$

is a local Hamiltonian, i.e., a sum of terms each of which acts on a small number of qubits only.

10.3.2 Dissipative quantum computing

Even more compelling could be the *dissipative model for quantum computing* in which one performs a suitable dissipative map and estimates for a suitable $t > 0$

$$p = \mathrm{tr}(e^{t\mathcal{L}}(\rho)(O \otimes \mathbb{I})), \tag{10.14}$$

where the *Liouvillian* is given by

$$\mathcal{L}(\rho) = -i[H, \rho] + \sum_j \left(L_j^\dagger \rho L_j^\dagger - \frac{1}{2} L_j^\dagger L_j \rho - \frac{1}{2} \rho L_j^\dagger L_j \rho \right). \tag{10.15}$$

Here, again, each of the terms $\{L_j\}$, referred to as Lindblad operators, are local and act on a small number of qubits only. The first part is the familiar part of the Schrödinger time evolution. The latter is a *Markovian master equation* capturing dissipative dynamics. This is interesting, as dissipation is usually seen as an enemy of coherent quantum evolution, and not a friend. So even with controlled dissipative dynamics alone, one can in principle realize a quantum computer.