QUANTUM INFORMATION & COMPUTATION

Nilanjana Datta, DAMTP Cambridge

1 Circuit model of quantum computation

The circuit model of classical computation above has a straightforward generalisation to the quantum setting. For inputs of size n the starting string $b_1 \ldots b_n 00 \ldots 0$ is replaced by a sequence of qubits in the corresponding computational basis state $|b_1\rangle \ldots |b_n\rangle |0\rangle |0\rangle \ldots |0\rangle$. A computational step is the application of a *quantum gate* which is a prescribed unitary operation applied to a prescribed choice of qubits. We do not need any randomised input qubits here as for example, a random choice of $|0\rangle$ and $|1\rangle$ can be generated by a measurement on $H|0\rangle$.

For each input size n we have a quantum circuit C_n which is a prescribed sequence of such steps. The output of the computation is the result of performing a quantum measurement (in the computational basis) on a specified subset of the qubits (this being part of the description of C_n).
Remark: More generally we could allow measurements along the way (rather than only at the end) and allow the choice of subsequent gates to depend on the measurement outcomes. However it can be shown that this further generality adds nothing extra: any such circuit can be re-expressed as an equivalent circuit in which measurements are performed at the end only. $\square$

A quantum computation or quantum algorithm is defined by a (uniform) family of quantum circuits $(C_1, C_2, \ldots)$.

Classical or quantum circuits can be depicted pictorially as a circuit diagram. Each input bit or qubit is represented by a horizontal line running across the diagram, which is read from left to right. The applied gates are represented by labelled boxes (or other symbols attached to the relevant lines), read in order from left to right.

Universal sets of quantum gates

In classical computation we restrict our circuits to be composed of gates chosen from a (small) universal set that act on only a few bits each. One such choice is the set {NOT, AND, OR}. Actually OR may even be deleted from this set since b_1 OR b_2 = NOT(NOT(b_1) AND NOT(b_2)).
Remark (optional): It may be shown that no sets of 2-bit *reversible* gates are universal (see Preskill p241-2) but there are 3-bit reversible gates G that are universal even just by themselves i.e. any reversible Boolean function may be constructed as a circuit of G's alone, so long as we have available constant extra inputs set to 0 or 1. Two examples of such gates (assuming our starting bit string can also have bits set to 1 in the extra working space) are the Fredkin gate $F(0b_2b_2) = 0b_2b_3$ and $F(1b_2b_3) = 1b_3b_2$ i.e. a controlled SWAP, controlled by the value of the first bit, and the Toffoli gate $\mathrm{Toff}(0b_2b_3) = 0b_2b_3$

and Toff$(1b_2b_3) = 1CX(b_2b_3)$ i.e. a controlled-controlled-X gate in which X is applied to bit 3 iff the first two bits are 1 and Toff is the identity otherwise. $\Box$

Approximately universal sets of quantum gates
In the quantum case all gates are reversible (unitary) by definition and there are similar universality results but the situation is a little more complicated: quantum gates are parameterised by *continuous* parameters (in contrast to classical gates which form a discrete set) so no finite set can generate them all *exactly* via (even unboundedly large) finite circuits. But many small finite sets of quantum gates are still *approximately universal* in the sense that they can generate any unitary gate with any prescribed accuracy $\epsilon > 0$. Such approximations (for suitably small ϵ) will suffice for all our purposes and for clarity of discussion we will generally ignore this issue of approximability and just allow use of any exact gate that we need.

More precisely, introduce a notion of closeness of unitary operators U and V (on the same space) by defining $||U - V|| \leq \epsilon$ to mean that $\max ||U |\psi\rangle - V |\psi\rangle || \leq \epsilon$. Here the maximum is taken over all normalised vectors $|\psi\rangle$ and $||..||$ in the maximum is the usual length of vectors. Then a set of quantum gates (acting on qubits) is defined to be *approximately universal* if for any unitary W on any number n of qubits and any $\epsilon > 0$ there is a circuit C of the given gates whose overall unitary action (also denoted C) satisfies $||W - C|| \leq \epsilon$. The set is called *exactly universal* if we have $\epsilon = 0$ in the preceding condition.

For either exactly or approximately universal sets of gates, the size of the circuit C for W will generally be exponential in the number of qubits n on which W acts (but in some important special cases of W, it can be poly-sized e.g. notably for the quantum Fourier transform on n qubits, cf later! Another important issue is how the size of C grows with decreasing accuracy parameter ϵ. Here we just quote a fundamental result: the Solovay-Kitaev theorem asserts that if $\mathcal{G}$ is an approximately universal set of gates, then (under some further mild technical conditions on $\mathcal{G}$) the size of C can be taken to be bounded by poly$(\log(1/\epsilon))$ as a function of accuracy parameter i.e. polynomial in the number of digits $(\log(1/\epsilon)$ of accuracy (For a proof see e.g. appendix in Nielsen and Chuang). The degree of the polynomial p here depends (generally exponentially) on the number of qubits n of W, but for fixed n, p does not depend on the gate W being approximated.

Remark (optional)
Some examples of approximately universal sets of quantum gates are the following: $\{CX, \text{all 1-qubit gates}\}$, $\{CX, H, T = \begin{pmatrix} 1 & 0 \\ 0 & \exp i\pi/4 \end{pmatrix}\}$ and $\{\text{Toffoli 3-qubit gate}, H\}$ the latter actually being universal for all gates with *real* entries, which can be shown to suffice for full universal quantum computation i.e. for any quantum circuit there is a corresponding circuit comprising only real gates, that generates the same output probability distribution for any computational basis input. The infinite set $\{CX, \text{all 1-qubit gates}\}$ is actually *exactly* universal too (with continuous parameters provided by the 1-qubit gates). For more details and proofs see Nielsen and Chuang §4.5. $\Box$

Polynomial time quantum computations and BQP

The complexity class **BQP** (**b**ounded error **q**uantum **p**olynomial time) is defined as a direct generalisation of **BPP** viz. **BQP** is the class of languages L such that there is a polynomial time quantum algorithm for deciding membership of L i.e. for each input size n we have a quantum circuit C_n whose size is bounded by poly(n) and for any input string the output answer is correct with probability at least $2/3$.

BQP is our mathematical formalisation of "computations that are feasible on a quantum computer". From the definitions it can be shown that **BPP** $\subseteq$ **BQP** (any poly sized classical circuit can be replaced by an equivalent circuit of classical reversible gates, still of poly size and the latter is also a quantum circuit albeit comprising gates that preserve the computational basis as a set). Thus with these computational definitions the question "Is quantum computing more powerful than classical computing?" can be expressed formally as "Is **BQP** strictly larger than **BPP**?". This question remains unsolved although it is generally believed that the classes are unequal. For example the decision problem FACTOR(M, N) (viz. does M have a nontrivial factor less than N?) is in **BQP** (as we'll see in detail later) but it is not known to be in **BPP** (although we have no proof that it is not in **BPP**!)

More generally we will be especially interested in any kind of computational task that can demonstrate any kind of computational resource benefit (especially an exponential benefit) for solution by quantum vs. classical computation. Historically the notion of query complexity and promise problems that we introduced above, provided the first source of such examples, and we'll consider some of them as our first quantum algorithms below. But before giving explicit algorithms we need a further result about black boxes (oracles) in the context of quantum vs. classical computations.

Reversible version of any Boolean function

If $f : B_m \to B_n$ is any Boolean function it can be expressed in an equivalent *reversible* form $\tilde{f} : B_{m+n} \to B_{m+n}$ as follows. We introduce an addition operation, denoted $\oplus$, for n-bit strings: if $b = b_1 \dots b_n$ and $c = c_1 \dots c_n$ then $b \oplus c = (b_1 \oplus c_1) \dots (b_n \oplus c_n)$ i.e. $b \oplus c$ is the n-bit string obtained by adding mod 2, the corresponding bits of b and c in each slot separately. For example $011 \oplus 110 = 101$. Note that for any n-bit string we have $b \oplus b = 0 \dots 0$ where $0 \dots 0$ denotes the n-bits string of all zeroes.

Now for any $f : B_m \to B_n$ define $\tilde{f} : B_{m+n} \to B_{m+n}$ by

$$\tilde{f}(b, c) = (b, c \oplus f(b)) \quad \text{for any } m\text{-bit string } b \text{ and any } n\text{-bit string } c.$$

Note that $\tilde{f}$ is easily computable if we can compute f and the (simple) addition operation $\oplus$ on bit strings. Conversely given $\tilde{f}$ we can easily recover $f(b)$ for any b by setting $c = 0 \dots 0$ and looking at the last n bits of output of $\tilde{f}$.

Furthermore we have the key property: for any f, $\tilde{f}$ is a *reversible* (i.e. invertibele function on $m + n$ bits. In fact $\tilde{f}$ is always self-inverse i.e. $\tilde{f}$ applied twice is the identity operation (an easy consequence of the fact that $b \oplus b = 00 \dots 0$ for any bit string b).

It should be intuitively clear that any classical algorithm using an oracle for f can be equally well performed using an oracle for the reversible version $\tilde{f}$ instead. In quantum computation, gates are always reversible (unitary) by definition so we will always use (a quantum version of) $\tilde{f}$ for any oracle problem involving f. More specifically the **quantum oracle for any Boolean function** $f : B_m \to B_n$ will be the quantum gate denoted U_f on $m+n$ qubits, defined by its action on basis states as follows:

$$U_f \left|x\right\rangle \left|y\right\rangle = \left|x\right\rangle \left|y \oplus f(x)\right\rangle \qquad \text{for all } x \in B_m \text{ and } y \in B_n$$

i.e. U_f acts exactly like the classical function $\tilde{f}$ on the labels $(x, y) \in B_{m+n}$ of the computational basis states (and it acts on arbitrary states of $m+n$ qubits by linear extension). We sometimes refer to the m-qubit register $\left|x\right\rangle$ and the n-qubit register $\left|y\right\rangle$ as the **input** and **output registers** respectively.
Remark. U_f as defined above is always guaranteed to be a *unitary* operation. Indeed if $g : B_k \to B_k$ is any *reversible* Boolean function on k bits then it is just a permutation of all k-bit strings. Hence the linear map V on k qubits defined by $V \left|i_1 \dots i_k\right\rangle = \left|g(i_1 \dots i_k)\right\rangle$ will be represented by a permutation matrix in the computational basis i.e. each column is all 0's with a single 1 entry, and different columns have the 1 entry in different rows, so V is unitary. $\square$

Computation by quantum parallelism

Note that as a quantum operation (in contrast to classical oracles) U_f can act on (jointly) superposed inputs of both registers. Indeed if we set the input register to an equal superposition of all 2^m possible m-bit strings we get (by linearity)

$$U_f : \frac{1}{\sqrt{2^m}} \sum_{\text{all } x} \left|x\right\rangle \left|0\right\rangle \to \left|\psi_f\right\rangle \equiv \frac{1}{\sqrt{2^m}} \sum_{\text{all } x} \left|x\right\rangle \left|f(x)\right\rangle$$

i.e. in *one run* of U_f we obtain a final state which depends on *all* of the function values. Such a computation on superposed inputs is called **computation by quantum parallelism**. By further quantum processing and measurement on the state $\left|\psi_f\right\rangle$ we are able to obtain "global" information about the nature of the function f (e.g. determine some joint properties of all the values) with just one run of U_f, and these properties may be difficult to get classically without *many* classical evaluations of f (as each such evaluation reveals only one further value). This simple idea of running computations in quantum superposition is a powerful ingredient in quantum vs. classical algorithms. In Appendix 1 (at the end of the notes) we discuss some further issues relating to the interpretation of superpositions in quantum computation.

It is instructive to consider more explicitly how we can actually create the input state of a uniform superposition over all x values that is needed in the above process. Recall that $H\left|0\right\rangle = \frac{1}{\sqrt{2}}(\left|0\right\rangle + \left|1\right\rangle)$ so if we apply H to each of n qubits initially in state $\left|0\right\rangle$ and multiply out all the state tensor products, we get

$$\begin{aligned} H \otimes \dots \otimes H(\left|0\right\rangle \dots \left|0\right\rangle) &= \tfrac{1}{\sqrt{2^n}}(\left|0\right\rangle + \left|1\right\rangle) \dots (\left|0\right\rangle + \left|1\right\rangle) \\ &= \tfrac{1}{\sqrt{2^m}} \textstyle\sum_{x_1, x_2, \dots, x_m = 0}^{1} \left|x_1\right\rangle \left|x_2\right\rangle \dots \left|x_m\right\rangle = \tfrac{1}{\sqrt{2^m}} \textstyle\sum_{x \in B_n} \left|x\right\rangle . \end{aligned}$$

An important feature of this process (recalling the fundamental significance of poly vs. exponential growth in complexity theory) is that we have created a superposition of *exponentially many* (viz. 2^m) terms with only a *linear* number of elementary operations viz. application of H just m times.