C/CS/Phys C191 Reversible Computing, Randomized Computation, Deferred Measurements 9/29/07
Fall 2009 Lecture 10

1 Readings

Reversible computation: Benenti et al, Ch.1.5 - 1.6; Kaye et al, Ch. 1.5

Randomized computation: Kaye et al, Ch. 6. 1

Energy as a resource: Nielsen and Chuang, Ch. 3.2.5

Deferred Measurements: Nielsen and Chuang, Ch. 4.4

2 Reversible Computation

The classical NAND gate is irreversible. Landauer first pointed out that there is a minimal amount of energy dissipated from the computer into the environment with every irreversible computation. This minimal energy loss is equal to $kTln2$ and derives from the entropy decrease of the information on erasure (initialization) of one bit. Entropy decrease of the information is accompanied by an entropy increase of the environment and dissipation of energy into this. Note that information content is equivalent to our ignorance of the message, i.e., of the actual state of the bit.

Energy dissipation in electronic circuits has decreased substantially over the last 40 years, by a factor of about 10 every 4 years, going from about 1 x 10^{-2}J per logical operation in 1940 to about 1 x 10^{-7}J per logical operation today. Extrapolation of this trend (is this valid?) would imply that the energy dissipated per logical operation will reach the thermal limit kT at $T = 300$ K within 10-15 years from now. In that situation spontaneous fluctuations could cause the circuits to switch and computations to cease being reliable. The anticipation of this situation led to the development of reversible computation schemes.

We shall illustrate this here by supposing that we are given a classical circuit, for example for primality testing of an input number M (note that the length of the input M is $\lceil \log_2 M \rceil$), and showing how to construct a corresponding reversible circuit.

Any classical circuit can be built from three basic pieces: the AND gate $\wedge(a,b) = a \cdot b$, the NOT gate $NOT(a) = 1 - a$, and fan-out (copying). The NOT gate is a reversible, unitary one-qubit operation. But the AND gate clearly cannot be reversible; since it takes two bits to just one, some information must be lost. (In particular, after applying a NOT gate, we cannot distinguish the cases where the inputs were 00 versus 01 or 10.) The AND gate erases information, which is not reversible. However, copying of classical states is reversible, $(a,0) \rightarrow (a,a)$. There are a number of ways to construct a reversible circuit corresponding to a nonreversible circuit. For example, one can build the necessary pieces out of a controlled swap gate (Fredkin gate, another 3-bit gate), a CNOT gate, and a NOT gate. Here, we will construct a corresponding reversible circuit using the Toffoli gate and the NOT gate. We encountered the Toffoli gate a couple of lectures ago: it is a doubly-controlled NOT gate with action (a,b,c) to $(a,b,c+ab \bmod 2)$. It is its own inverse, so it is by definition reversible.

Now we consider the standard, possibly irreversible, circuit C taking input x to $y = C(x)$, shown in Figure 1. The Toffoli and NOT gates can be used to replace each of the circuit components to build a reversible circuit

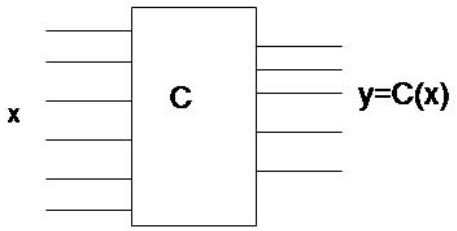

Figure 1: The classical, possibly nonreversible circuit C

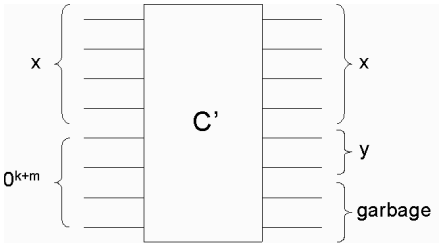

Figure 2: The reversible circuit C' that produces extra garbage (which should be reset reversibly)

$\hat{C}$ taking $(x, 0^k)$ to (x, y). Actually, some number of ancilla bits, initialized to 0 will also be useful, so $\hat{C}$ takes $(x, 0^k, 0^m)$ to $(x, y, 0^m)$. (Notice that the m ancilla bits are unchanged... which will require putting them back to their initial states.) The replacement procedure is summarized below and the resulting reversible circuit is shown in Figure 2.

- To achieve an AND gate on a and b, we use the Toffoli gate on input $(a, b, 0)$. The output is $(a, b, a \wedge b)$, so the third output wire has the result of the AND.
- To copy a bit a, use the Toffoli gate on input $(a, 1, 0)$. The output is $(a, 1, a)$, so we have copied a. Note that this uses both 0 and 1 ancillas; to get a 1 ancilla, we can simply apply a NOT gate to a constant 0 wire.

This method allows us to construct a circuit C' corresponding to C which reversibly takes $(x, 0^k, 0^m)$ to $(x, C(x), \text{garbage}_x)$. The garbage is left over in the original ancilla wires because our operations had extra inputs and outputs. For example to achieve a NOT we had three output wires, whereas the nonreversible NOT gate only has one output wire. The extra two wires are just garbage which we can reset to their initial states for a clean circuit. This does not have to introduce irreversibility - given sufficient extra bits it it can be done reversibly using the method we show below. See also Nielsen and Chuang, p. 158, for a related approach.

We shall show below how to return the garbage states back to their initial values in a reversible fashion. But first let us consider how things look with quantum circuits. As a historical note, quantum computation actually was originally (in the late 70s and early 80s) studied to understand whether unitary constraint on quantum evolution provided *limits* beyond those explored in classical computation. A unitary transformation taking basis states to basis states must be a permutation. (Indeed, if $U|x\rangle = |u\rangle$ and $U|y\rangle = |u\rangle$, then $|x\rangle = U^{-1}|u\rangle = |y\rangle$.) Therefore quantum mechanics imposes the constraint that its classical analog must be reversible computation.

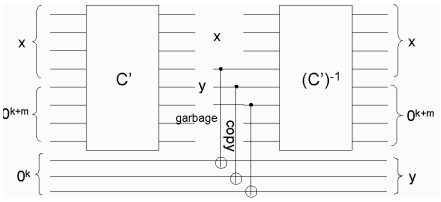

Figure 3: The clean reversible circuit $\hat{C}$ built out of C' and $(C')^{-1}$.

Furthermore, from the perspective of a quantum algorithm, any extra garbage left over is quite significant because it can prevent desirable destructive interference when we run the circuit on states which are not just computational basis states. Thus in a quantum circuit we definitely want a clean output. Fortunately, there is a simple trick for removing all the garbage reversibly; we just run the circuit in reverse to bring the garbage back to its initial state! Of course, we don't want to forget our final answer, so before running C' in reverse we need to copy $y = C(x)$ to some additional ancilla wires. This copying just uses one additional 1 ancilla (from the reversible copy with the Toffoli gate described above), and does not create any further garbage. The sequence of steps is then

$$(x, 0^k, 0^m, 0^k, 1) \xrightarrow{C'} (x, y, \text{garbage}_x, 0^k, 1) \xrightarrow{\text{copy } y} (x, y, \text{garbage}_x, y, 1) \xrightarrow{(C')^{-1}} (x, 0^k, 0^m, y, 1) \ .$$

Overall, this gives us a clean reversible circuit $\hat{C}$ corresponding to C, shown in Figure 3.

3 Review of quantum circuit model

Recall that in the quantum circuit model we have n qubits that we can manipulate in the following ways:

1. Initialization: The qubits can be initialized to the state $|0^n\rangle$. Preparing a different input state $|x\rangle$, $x \in \{0,1\}^n$ can be done by flipping the required bits.

2. Universal set of gates: Certain sets of one- and two-qubit gates can approximate any constant dimensional unitary transformation sufficiently closely. For example, the CNOT gate together with all one qubit transformations (rotations on the Bloch sphere) forms a universal gate set.

3. Measurement of some (or all) of the qubits output by the quantum circuit. For example if $|\psi\rangle = \sum_x \alpha_x |x\rangle$ and we do a full measurement in the standard/computational basis, then we measure x with probability $|\alpha_x|^2$. If we only measure the first k bits of x, then the probability of measuring $z \in \{0,1\}^k$ is $\sum_{x:z \text{ is a prefix of} x} |\alpha_x|^2$. The resulting partially collapsed quantum state is, up to normalization, $\sum_{x:z \text{ is a prefix of} x} \alpha_x |x\rangle$.

4. Classical postprocessing of the measured value to get the solution to the problem being solved. Quantum computers are expensive and rare (!), so we would probably prefer to use classical processing as much as possible.

The size of a quantum circuit is the number of gates in the circuit. We are interested in finding *efficient* circuits for problems, i.e., circuits for which the total size of the circuit is polynomially bounded in the number of input bits. For example, a family of circuits of size $c \cdot n^5$ is good. But exponentially large – 2^n – circuits are bad.

4 Randomized computation

Many important classical algorithms are randomized. For example, the most common primality testing algorithm needs as input a random string which is then used to construct a test of whether the number is prime (see, e.g., "primality test" in http://Mathworld.wolfram.com). (Note that a deterministic polynomial time algorithm was discovered in 2002 by Agrawal et al.)

Rabin-Miller Strong Pseudoprime test:

For odd integer n, let $n = 2^r s + 1$ with s odd, choose a random integer α, $1 \leq \alpha \leq n-1$. If $\alpha^s = 1 (\mathrm{mod}\, n)$ or $\alpha^{2^j s} = -1 (\mathrm{mod}\, n)$ for some $0 \leq j \leq r-1$, then n passes the test, i.e. it is prime. A true prime number will pass the test for all α. Thus this protocol can give us erroneous result for some random α (i.e. tell us that n is prime, when it isn't), but with a small probability.

To simulate quantumly:

1. First create the corresponding reversible circuit with inputs x, α and ancilla 0's.
2. To randomize α, feed each $|0\rangle$ qubit wire through a Hadamard gate, giving $\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$. Immediately after applying the Hadamard gate, measure each qubit of α.
3. Instead of measuring the qubits of α, it is sufficient to copy (with CNOT gates) the outputs of the Hadamard gates into fresh qubits. For example, we change $\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)|0\rangle$ to $\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$. Since they are entangled, measuring the bits into which we copied each computational basis state of α is equivalent to measuring the bits of α itself.
4. In fact, though, it doesn't matter whether we measure the fresh qubits before or after running the quantum circuit. In fact, we can delay their measurement arbitrarily long, or just avoid it altogether. This is known as the "principle of deferred measurement." Measurement is equivalent to entanglement of the system with its environment.

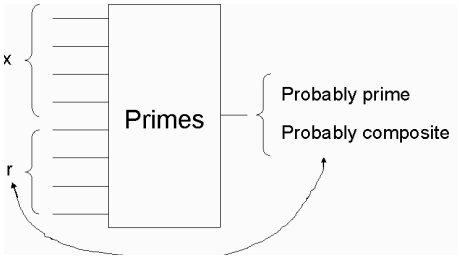

Figure 4: A circuit for primality testing which takes as additional input a random string $r \equiv \alpha$.

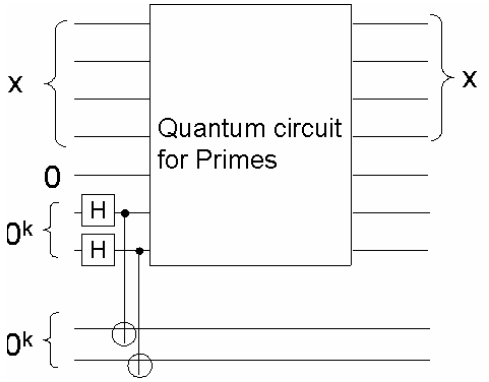

Figure 5: The corresponding quantum circuit; copying with a CNOT the qubits $\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ is equivalent to measuring them, giving a random string α.

For each x, for most choices of α, the circuit computes the correct answer.

5 Deferred measurements

Another example of this principle of deferred measurement can be shown for teleportation. Figure 6 shows the usual teleportation circuit in which Alice (on left) performs the measurements of qubits 1 and 2, then sends the classical output of these measurements to Bob (on right). Figure 7 shows the equivalent circuit in which the measurements are done at the end, rather than in the middle. Instead of unitaries condition on the result of Alice's measurements, Bob makes controlled unitary operations on qubit 3. You can convince yourselves of the equivalence by writing out the states and actions of the measurements and controlled unitaries on then. The notation here is such that $|\beta_{00}\rangle = |\phi^+\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$.

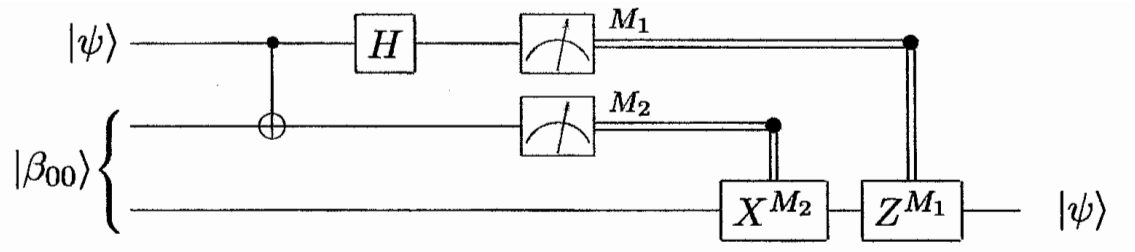

Figure 6: The usual teleportation circuit with measurement performed by Alice on qubits 1 and 2, who then sends the classical information from this to Bob who performs single qubit unitaries conditional on the results on qubit 3.

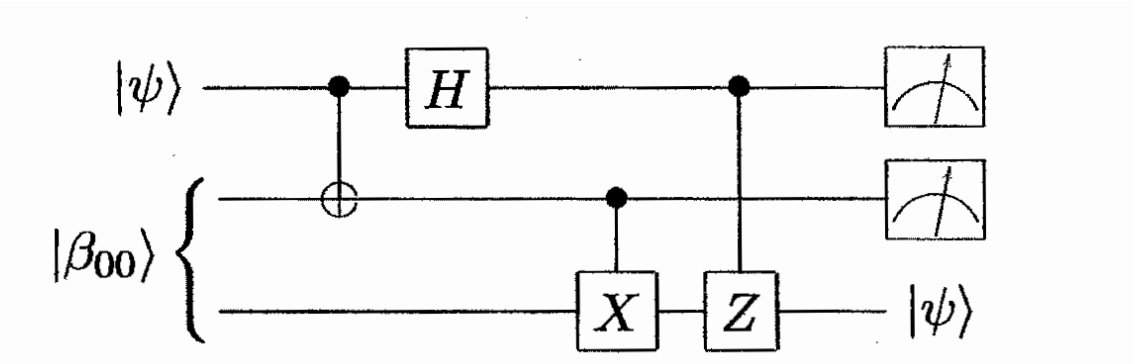

Figure 7: The deferrred measurement quantum teleportation circuit.