E. Servan-Schreiber
Vazirani Class #5
September 16

Quantum Simulations of a Probabilistic Circuit

1 Fourier Basis

The state of an n qubit is described by a unit vector in $\mathbb{C}^{2^n}$. A particularly convenient (orthonormal) basis for this space is the computational basis, in which there is a basis vector for every classical configuration of the system. We shall label these basis vectors with n bit strings or whenever convenient with elements of Z_2^n. Thus the state may be written as $\sum_{x\in\{0,1\}^n} \alpha_x |x\rangle$.

There is another natural basis for the space — the Fourier basis, also called the parity basis. These basis vectors are also labeled by elements of Z_2^n. Thus for $u \in Z_2^n$:

$$|\chi_u\rangle = \frac{1}{2^{n/2}} \sum_{x\in\mathbb{Z}_2^n} (-1)^{x\cdot u} |x\rangle$$

where $x \cdot u$ is shorthand for $\sum_j x_j u_j$. Notice that if we measure the state of the system when it is in any of the fourier basis states $|\chi_u\rangle$, we get each $x \in \mathbb{Z}_2^n$ with equal probability, since the magnitude of the x^{th} component is $\frac{1}{2^{n/2}}$ independent of x. Thus in the Fourier basis vectors all the information resides in the phase, which is $+-1$. In $\mathbb{C}^2$, the parity basis is the now familiar result of the one bit Hadamard gate (see figure 1).

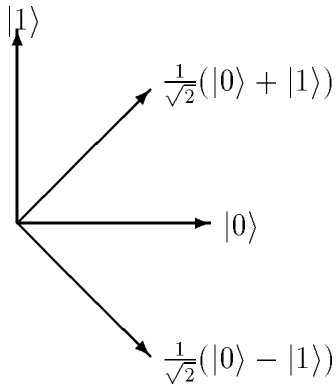

Figure 1: The Fourier Basis in $\mathbb{C}^2$

It is easy to verify that the Fourier basis vectors are indeed orthonormal (with respect to the inner-product implicit in the computational basis). To check orthogonality, write the dot product

$$|\chi_u\rangle \cdot |\chi_v\rangle = \sum_{x\in\mathbb{Z}_2^n} (-1)^{x\cdot u}(-1)^{xv}\frac{1}{2^n}$$

If $u \neq v$, there is a position j such that $u_j \neq v_j$. Assume $u_j = 0$. Now, if we pair up the elements of $\mathbb{Z}_2^n$ that differ only in the j^{th} bit, both elements of the pair have the same phase in $|\chi_u\rangle$, but have opposite phase in $|\chi_v\rangle$. Therefore the product $(-1)^{x\cdot u}(-1)^{xv}$ is 1 for one element of the pair and -1 for the other element. Therefore each pair in the partition contributes 0 to the above sum, and so the dot product is 0.

One way to picture the Fourier basis is by identifying Z_2^n with the vertices of the boolean n dimensional cube (see figure 2). Now a vector in $\alpha \in \mathbb{C}^{2^n}$ is a function on the vertices of the cube, which assigns the number α_x to vertex x. The computational basis or the assignment basis consists of delta functions - functions that assign the value 1 to exactly one vertex of the cube and 0 to all others. The fourier basis vector $|\chi_0\rangle$ assigns value $\frac{1}{2^{n/2}}$ to all the vertices of the hypercube. Each $u \neq 0$ defines a subcube consisting of half the vertices of the cube — those which have an even number of $1's$ in the positions picked out by u (positions j such that $u_j = 1$). The fourier basis vector $|\chi_u\rangle$ assigns value $\frac{1}{2^{n/2}}$ this subcube, and value $\frac{-1}{2^{n/2}}$ to the rest of the vertices in the hypercube.

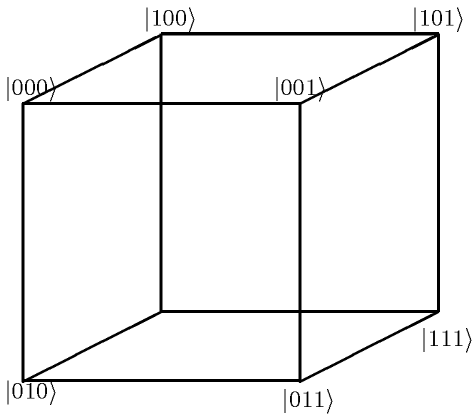

Figure 2: A 3-dimensional hypercube

The fourier transform over $\mathbb{Z}_2^n$ is the linear transform that maps the computational basis vector $|u\rangle$ to the fourier basis vector $|\chi_u\rangle$. The circuit to perform this transformation is particularly simple: run each bit of the input through a Hadamard gate.

2 Simulating a Probabilistic Circuit

A probabilistic circuit is a circuit which takes its input along with a set of random bits and outputs the correct answer with high probability. (see figure 3).

Our objective is to simulate this behavior with a quantum circuit.

Now that we know how to replace deterministic circuits with quantum circuits, the only problem we must resolve is how we can pick a uniformly random string to feed in the circuit.

One way to do that is easy: suppose we need m random bits, just feed the $|0^m\rangle$ vector to a Hadamard gate of m inputs (referred to as H_m), we will get the following superposition:

$$\sum_{R \in \mathbb{Z}_2^m} \frac{1}{2^{m/2}} |R\rangle$$

This superposes all the vectors of $\mathbb{Z}_2^m$ evenly. If we measure that, we will get any particular R with probability $\frac{1}{2^m}$ which is the uniform distribution over all m bit vectors, i.e. what we want (see figure 4).

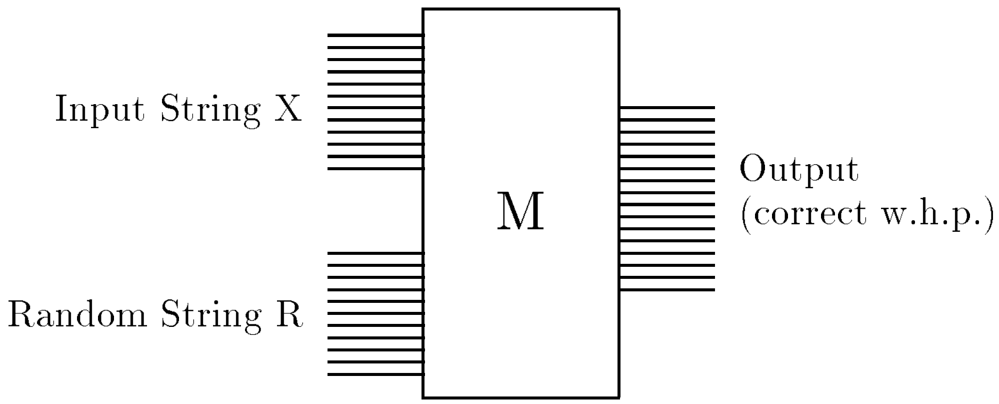

Figure 3: A generic probabilistic circuit

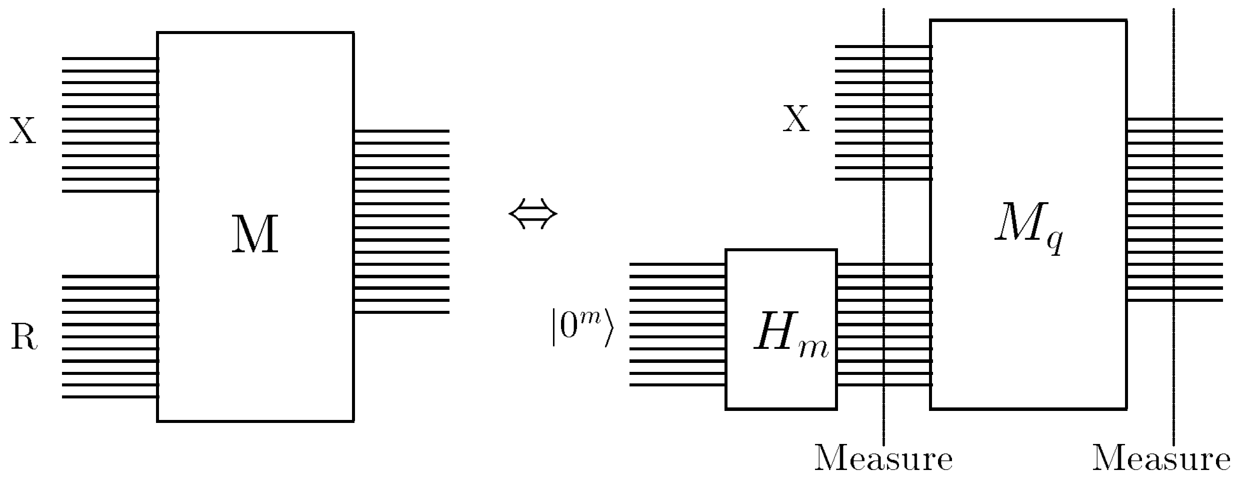

Figure 4: Measuring early is equivalent to generating a random string

3 Delaying measurement

But what if we wanted to wait until the end of the computation before doing any measurement. Is there a way we can set things up so that, given the quantum circuit M_q which performs the computation, we can be certain that waiting until the end to measure will not make any difference.

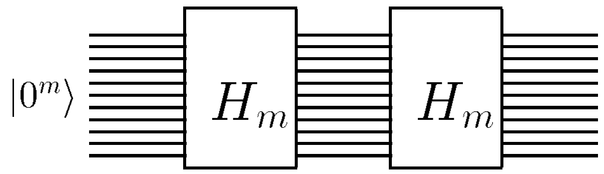

Figure 5: A quantum circuit where measuring before the end makes a difference

It is fair to ask what kind of problem could occur. The following example will make that clear. Consider the circuit in figure 5: two Hadamard circuits on m bits following each other.

case 1 If we measure just before the second Hadamard gate, we will observe some m-bit string R with probability $\frac{1}{2^m}$. Then the output of the second gate will be the superposition $\sum_{x\in\mathbb{Z}_2^m} \frac{(-1)^{R\cdot x}}{2^{m/2}}|x\rangle$. If we measure now, we will observe any given m-bit string with probability $\frac{1}{2^m}$.

case 2 If, on the other hand, we only make one measurement at the end, after the second Hadamard gate, then since H_m is its own inverse, the output superposition will be exactly what we started with, namely the $|0^m\rangle$ vector, which will measure to the 0 vector with probability 1.

The point of this example is to show that a uniform superposition of all the m-bit vectors cannot simply be thought of as a random string. It is only “random” in the computational basis, but in the Fourier basis it is one of the basis vectors.

So the problem still stands: what can we do to make sure that no matter what the circuit M_q does, if we make no measurement until the end, the behavior will be exactly the same as the original probabilistic circuit.

4 Copying the random bits

We can use a simple trick which consists of “copying” the random bits on a separate set of wires and promising that the circuit M_q will not touch them at all (see figure 6). At this point it is fair to ask why this should make any difference at all? What if the M_q is the Hadamard gate on m bits again, why is this not going to be a problem? Why indeed...

The major difference is that by adding another set of m bits which we entangle with our original m bits, we are now working in a much larger space ($\mathbb{C}^{2^{2m}}$ as opposed to $\mathbb{C}^{2^m}$), and whatever we do the first set of m bits, their configurations cannot interfere anymore. To interfere the whole $2m$-bit configurations have to match. But if we agree beforehand that we will not touch the "copy" set of bits, that cannot happen. In a sense, by performing this copy, we created enough dimensions so that each m-bit string can now roam in its own orthogonal subspace of 2^m dimensions. Within this subspace, it can assume any superposition of it wishes without interfering with the original superposition.

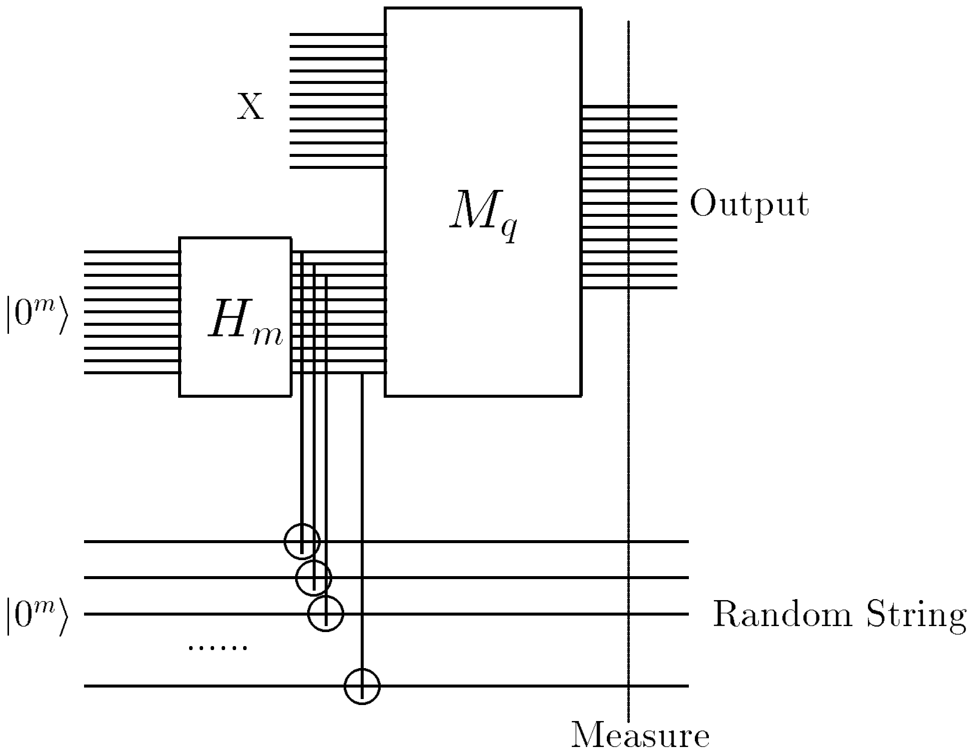

Figure 6: Copying the bits avoids the intermediate measurement

Note that the only important promise here is that M_q will not touch the second set of bits. But we may perform any operations we wish on that set.

Let us examine all this with a 1-bit example: Here $m = 1$ and $M_q = H_1$.

case 1 No copying. The uniform superposition of $|0\rangle$ and $|1\rangle$ is fed into H_1. As before, the output is just $|0\rangle$, not random at all.

case2 We make a copy of the entering qubit. The superposition is now $\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$. Now if we let the first bit go through the Hadamard gate and do nothing to the second one, the result becomes:

$$(H \otimes I)(\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) = \frac{1}{2}(|00\rangle + |10\rangle) + \frac{1}{2}(|01\rangle - |11\rangle)$$

Each vector has equal norm so we will observe, after measuring, each vector with probability 1/4, and in particular the first bit will be perfectly random.

case 3 Now let us see what happens if we also make the copy bit go through a Hadamard gate. Will it affect the randomness of the output?

$$(H\otimes H)(\frac{1}{\sqrt{2}}(|00\rangle+|11\rangle)) = \frac{1}{2^{3/2}}((|0\rangle+|1\rangle)\otimes(|0\rangle+|1\rangle))+\frac{1}{2^{3/2}}((|0\rangle-|1\rangle)\otimes(|0\rangle-|1\rangle)) = \frac{1}{\sqrt{2}}(|00\rangle+|11\rangle)$$

The randomness of the first bit remains intact.

Let us look again at this phenomenon in QTM notation. Let us call U_{M_q} the unitary transformation performed by the circuit M_q and U_c the (arbitrary) unitary transformation performed on the copy bits.

$$\begin{aligned} \frac{1}{2^{m/2}} \sum_{R\in\mathbb{Z}_2^m} |R\rangle|R,x\rangle \quad &\Rightarrow \quad \frac{1}{2^{m/2}} \sum_R U_c \otimes U_{M_q}(|R\rangle|R,x\rangle) \\ &= \quad \frac{1}{2^{m/2}} \sum_R U_c|R\rangle \otimes U_{M_q}|R,x\rangle \end{aligned}$$

What this means is that when we perform a measurement at the end of the computation, we will observe with probability $\frac{1}{2^m}$ for any fixed m-bit string R the output $U_c|R\rangle, U_{M_q}|Z_R\rangle$. That is exactly what we want! It is equivalent to saying that the circuit M_q is run with any particular string R with probability $\frac{1}{2^m}$.

5 Summary

In this lecture, we saw how to simulate any probabilistic circuit with a quantum circuit in two different ways. The first (easy) way is to create a uniform superposition of all the random strings and make a measurement before feeding it to the circuit. The second (interesting) way allows all measurements to be delayed until the end of the computation. To insure that the circuit will not interfere with the uniform superposition of the random strings, you must make a copy of the qubits and promise that the copied qubits will not be touched by the original circuit.