Lecture I
Probability and linear algebra
Regsiters, states, measurements

Jaikumar Radhakrishnan
jaikumar.radhakrishnan@icts.res.in

Monday 22nd January, 2024, 23:40

Please let me know if you spot an error.

We discuss classical deterministic circuits and model the states of the registers and the computation on them as vectors and linear transformations. Extending this discussion to randomized computation, and model randomized states as probability vectors and stochastic matrices. We introduce the *temporary* notation of $|v\rangle\!\rangle$ and $\langle\!\langle v|$ in the context of classical computation and motivate the use of $|v\rangle$ and $\langle v|$ in the context of quantum computation.

Classical computation performed by a circuit looks like this. The *wires* represent *registers*. Each register holds a bit, that is $\{0,1\}$. The state of the registers is modified by gates. The goal of computation is transform the *input state* to the desired *output state* by applying a small number of gates. The input state of the system with n-input registers is an element of $\{0,1\}^n$, that is bit-strings of length n; similarly, the state of output registers is an element of $\{0,1\}^m$. The first bit corresponds to the topmost register in the picture, and the last bit corresponds to the bottommost.

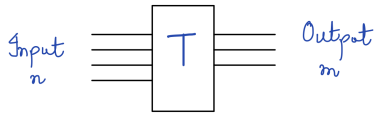

Figure 1: A computation with n input registers and m output registers

Example 1 *Consider the operation of the* AND *gate. There are two input registers and one output register. Thus,* $\text{AND} : \{0,1\}^n \to \{0,1\}^m$.

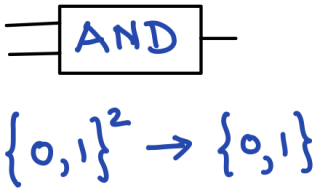

Figure 2: The two-input AND gate

Let us consider the operation of the AND gate in this example. The *truth table* specifies this operation.

Input	Output
00	0
01	0
10	0
11	1

Alternatively, the operation of the AND gate can be specified by its 1×2 matrix.

$$M_{\text{AND}} = \begin{pmatrix} 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

In general, to describe a computation involving n input and m output registers, we use a $2^m \times 2^n$ matrix. We index the columns of the matrix by the input states and the rows by the output states. If $\mathtt{a} \in \{0,1\}^m$ and $\mathtt{b} \in \{0,1\}^n$, then $M_T[\mathtt{a},\mathtt{b}]$ is 1 if $T(\mathtt{b}) = \mathtt{a}$, and is 0 otherwise. Note that since the computation is deterministic, each

column of M_T has exactly one 1. Note also that the way the matrix is written depends on the order in which the output and input states are listed as indices for rows and columns: we will normally use the lexicographic ordering, e.g., 000, 001, 010, 011, 100, 101, 110, 111. If we choose to describe the input-output behaviour of the computation using matrices, then it is convenient to represent the states of the registers as a column vector.

The input and output states for the two-input AND gate above can be described using state vectors

$$00 \mapsto \begin{pmatrix}1\\0\\0\\0\end{pmatrix} \qquad 01 \mapsto \begin{pmatrix}0\\1\\0\\0\end{pmatrix} \qquad 10 \mapsto \begin{pmatrix}0\\0\\1\\0\end{pmatrix} \qquad 11 \mapsto \begin{pmatrix}0\\0\\0\\1\end{pmatrix}$$

$$0 \mapsto \begin{pmatrix}1\\0\end{pmatrix} \qquad 1 \mapsto \begin{pmatrix}0\\1\end{pmatrix}$$

If $\vec{v}$ represents the input state and $\vec{w}$ represents the output state, then

$$\vec{w} = M_t \vec{v},$$

where we use the familiar method to multiply matrices.

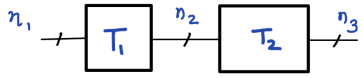

Figure 3: T_1 and T_2 in series

Now, suppose our computation T consists of two smaller computations executed in *series*, T_1 (with n_1 input registers and n_2 output registers) followed by T_2 (with n_2 input registers and n_3 output registers). Then, the $2^{n_3} \times 2^{n_1}$ matrix of the computation T can be recovered from the $2^{n_2} \times 2^{n_1}$ matrix of M_{T_1} and the $2^{n_3} \times 2^{n_2}$ matrix M_{T_2} using matrix multiplication: $M_T = M_{T_2} \cdot M_{T_1}$. What if the computations were performed in *parallel*? Let us first understand the two simple cases.

Figure 4: The two-input AND gate in parallel with identity

In the first, the AND gate is applied to the first two registers, and the third register is undisturbed; in the second case, the AND is applied to the last two registers. Clearly, the matrix M_T of the final computation can be obtained by inspecting the matrices for M_{T_1} and M_{T_2}. How exactly? If we examine the input-output behaviour of the two circuits, we obtain the following matrices in the two cases

$$M_{T_1} = \begin{pmatrix}1&1&1&0\\0&0&0&1\end{pmatrix} \quad M_{T_2} = \begin{pmatrix}1&0\\0&1\end{pmatrix} \qquad M_T = \begin{pmatrix}1&0&1&0&1&0&0&0\\0&1&0&1&0&1&0&0\\0&0&0&0&0&0&1&0\\0&0&0&0&0&0&0&1\end{pmatrix}$$

$$M_{T_1} = \begin{pmatrix}1&0\\0&1\end{pmatrix} \qquad M_{T_2} = \begin{pmatrix}1&1&1&0\\0&0&0&1\end{pmatrix} \quad M_T = \begin{pmatrix}1&1&1&0&&&&\\0&0&0&1&&&&\\&&&&1&1&1&0\\&&&&0&0&0&1\end{pmatrix}$$

We learn that the general rule for obtaining the matrix for the computation T on built out of the parallel execution of the operations T_1 and T_2 is the following: (i) M_T is a $2^{m_1+m_2} \times 2^{n_1+n_2}$ matrix; (ii) For $x_1 \in \{0,1\}^{n_1}, x_2 \in \{0,1\}^{n_2}, y_1 \in \{0,1\}^{m_1}, y_2 \in \{0,1\}^{m_2}$, we have

$$M_T[y_1y_2, x_1x_2] = M_{T_1}[y_1, x_1] \cdot M_{T_2}[y_2, x_2].$$

We then say that M_T is the *tensor product* of M_{T_1} and M_{T_2}, and write $M_T = M_{T_1} \otimes M_{T_2}$. Once again, when writing down the matrix explicitly, one needs to pay some attention to how the columns and rows are indexed.

Reverisible computation

The operations we commonly use for implementing deterministic computation are not all reversible. Indeed, if the number of input registers is more than then number of output registers, then there must be two input states that lead to the same output state. If we have fewer input registers than output registers, then the 'inverse' will have an excess of input registers. While we aim to show that quantum computation offers us more power in certain circumstances, we will need to rely on tasks that we already know to perform deterministically. It will be important for us then that our classical computation is *reversible* and *clean*.

We know that all classical computation can be performed using two-input AND gates and NOT gates. The NOT gate is reversible.

$$M_{\text{NOT}} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

The actions of the AND gate are not reversible. However, it is possible to realize an AND gate or an OR gate (and also the NOT gate) by suitably specializing reversible gates, the Toffoli gate and Fredkin gate, both of which operate on three registers.

Clearly, these gates are their own inverses. All classical computation can be performed using either of these gates alone, with some acceptable loss in efficiency. For example, we may compute the three input AND(x, y, z) using two two-input Toffoli gates.

Restoring the workspace

Computations often require additional workspace, registers that are involved in intermediate computations. Suppose we have a computation that involves some workspace registers. Note that in the end the workspace contains information that we are not interested in. In classical computation, the presence of 'garbage' in the workspace poses

Figure 5: The NOT gate

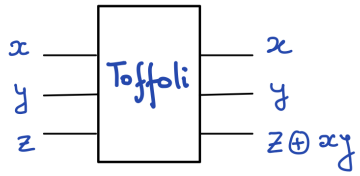

Figure 6: The Toffoli gate

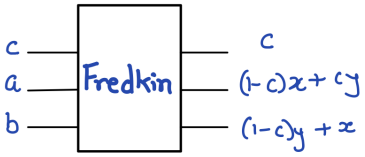

Figure 7: The Fredkin gate

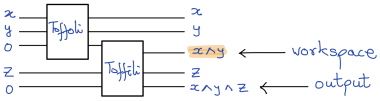

Figure 8: A three-input AND build out of Toffoli gates

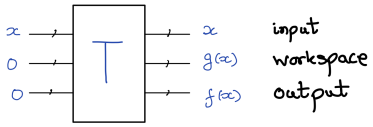

Figure 9: Garbage collects in the workspace

no problem. However, as will see, for such classical computation to be deployed as part of a bigger quantum algorithm, we must restore the workspace to its original state. This is actually easily achieved: copy the output and reverse the computation.

Note that if the output register started of in the state b, then its final state would be $b \oplus f(x)$.

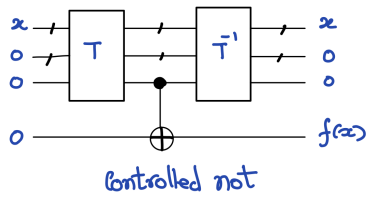

Figure 10: Restoring the workspace

Classical randomized computation

Consider the state of n registers on which a randomized computation has been performed. The state is a probability distribution on $\{0,1\}^n$, and can be written as a probability vector. E.g., the state where each of the 2^n possibilities has equal probabilities is represented by the probability vector

$$\begin{pmatrix} 1/2^n \\ 1/2^n \\ \vdots \\ 1/2^n \end{pmatrix}.$$

We will use a \$ gate (a *dollar* gate) to model a coin toss; when this gate is applied to a register, the resulting state is equally likely to be 0 or 1 irrespective of what the register contained initially, and independently of the action of the other \$ gates; it is natural, therefore, to describe the state after the application of a \$ gate by $\begin{pmatrix} 1/2 \\ 1/2 \end{pmatrix}$. Suppose n registers start off in the state $00\cdots0$. Now if the \$ gate is applied to all but the first register. The joint state of these registers then will be a uniform distribution on the possibilities of the form $0x$, where $x \in \{0,1\}^{n-1}$. The probability vector corresponding to this state is

$$\begin{pmatrix} 1/2^{n-1} \\ 1/2^{n-1} \\ \vdots \\ 1/2^{n-1} \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}.$$

Note that the classical deterministic states can be thought of as special randomized states, where all the probability is entirely concentrated on only one outcome. Consider the circuit with two registers where \$ gate applied to the first register, and then a controlled-NOT is applied with the first register acting as control. In the state that results in the end, the outcomes 00 and 11 have probability 1/2, and

the other two outcome have probability 0, so it corresponds to the probability vector is

$$\begin{pmatrix} 1/2 \\ 0 \\ 0 \\ 1/2 \end{pmatrix}.$$

Just as we did for deterministic computation, the computation of circuits that involve \$ gates can be described by matrices. For example, the \$ gate itself corresponds to the corresponds to the 2×2 matrix $\begin{pmatrix} 1/2 & 1/2 \\ 1/2 & 1/2 \end{pmatrix}$. What 4×4 matrix describes the computation of the above circuit?

States, random variables, events

We will give randomized states names: $|p\rangle\!\rangle$, $|q\rangle\!\rangle$, ..., $|\alpha\rangle\!\rangle$, $|\beta\rangle\!\rangle$, etc. As we saw, these states can be described by probability vectors, but it is often easier to manipulate the states using their names than working with probability vectors. Consider a randomized system with two registers. Then, we have the following special states: $|00\rangle\!\rangle, |01\rangle\!\rangle, |10\rangle\!\rangle, |11\rangle\!\rangle$. The uniform distribution on two-registers is the state:

$$|u\rangle\!\rangle = \frac{1}{4}|00\rangle\!\rangle + \frac{1}{4}|01\rangle\!\rangle + \frac{1}{4}|10\rangle\!\rangle + \frac{1}{4}|11\rangle\!\rangle.$$

If the registers have names, say A, B, and we want to emphasize them, then we write them as subscripts: $|u\rangle\!\rangle_{AB}$, $|10\rangle\!\rangle_{AB}$, etc.

Random variables assign values to the outcomes of a probabilistic experiment. E.g., in a system with two registers, the function $wt : \{0,1\}^2 \to \mathbb{R}$, that maps the outcome a_1a_2 to $a_1 + a_2$ is a random variable. Now, if $|p\rangle\!\rangle$ is a state of the two registers, then we may compute the expectation of a random variable X as follows:

$$\mathrm{E}[X] = \sum_{\mathtt{a}} X(\mathtt{a})p(\mathtt{a}),$$

where a ranges over the possible outcome of the system. It will be convenient to describe random variables as row vectors (for we will later make them interact with states, which we have chosen to describe as column vectors), e.g.,

$$wt = \begin{pmatrix} wt(00) & wt(01) & wt(10) & wt(11) \end{pmatrix} = \begin{pmatrix} 0 & 1 & 1 & 2 \end{pmatrix}.$$

Then, we can obtain the expectation of wt in the state $|p\rangle\!\rangle$ by multiplying the row with a column:

$$\begin{pmatrix} 0 & 1 & 1 & 2 \end{pmatrix} \begin{pmatrix} p_{00} \\ p_{01} \\ p_{10} \\ p_{11} \end{pmatrix}.$$

In this way, every random variable maps a probabilistic state to a scalar. This function is linear on the states. We denote the linear function associated with the random variable X by $\langle\!\langle X|$, and write $\langle\!\langle X|p\rangle\!\rangle$ to denote the scalar that results on the application of this function to the state $|p\rangle\!\rangle$; thus, $\langle\!\langle X|p\rangle\!\rangle = \mathrm{E}_p[X]$. A special case of random variables are indicator functions of event. Recall that we formally define an event as a subset of outcomes; e.g., for a system with three registers, we have

$$\begin{aligned}\text{odd parity} &= \{\mathtt{001},\mathtt{010},\mathtt{100},\mathtt{111}\};\\ \text{majority} &= \{\mathtt{011},\mathtt{101},\mathtt{110},\mathtt{111}\};\\ \text{the third bit is } \mathtt{1} &= \{\mathtt{001},\mathtt{011},\mathtt{101},\mathtt{111}\}.\end{aligned}$$

For each of them, we may define a 0-1 indicator random variable, which takes the value 1 precisely when the outcome is in the set; e.g.,

$$\text{odd parity}(a_1a_2a_3) = 1 \text{ iff } a_1 + a_2 + a_3 = 1 \pmod 2.$$

What row vectors correspond to the linear functions $\langle\!\langle$odd parity$|$, $\langle\!\langle$majority$|$, $\langle\!\langle$the third bit is 1$|$? Note that if I_A is the indicator random variable for the event A, then $\langle\!\langle I_A|p\rangle\!\rangle = \mathrm{E}_p[I_A] = \Pr_p[A]$.

Returning to random variables in general, note that we may add random variables: $(f+g)(w) = f(w) + g(w)$; the corresponding linear function is written as $\langle\!\langle f| + \langle\!\langle g|$. We then, have the rule $(\langle\!\langle f| + \langle\!\langle g|)|p\rangle\!\rangle = \langle\!\langle f|p\rangle\!\rangle + \langle\!\langle g|p\rangle\!\rangle$. We may also scale a random variable: $(cf)(w) = cf(w)$, and check that $\langle\!\langle cf|p\rangle\!\rangle = c\langle\!\langle f|p\rangle\!\rangle$. We may derive the *linearity* property of the expectation for random variables as a consequence of the above discussion:

$$\mathrm{E}_p[aX + bY] = \langle\!\langle aX|p\rangle\!\rangle + \langle\!\langle bY|p\rangle\!\rangle = a\langle\!\langle X|p\rangle\!\rangle + b\langle\!\langle Y|p\rangle\!\rangle = a\,\mathrm{E}_p[X] + b\,\mathrm{E}_p[Y].$$

Randomized transformations

Recall that with each outcome $b \in \{\mathtt{0},\mathtt{1}\}^n$ of a system of n-registers, there is a linear function $\langle\!\langle b|$ (e.g., we have $\langle\!\langle \mathtt{00}|$, $\langle\!\langle \mathtt{00}|$), which takes the value 1 on the deterministic state $|b\rangle\!\rangle$ and 0 on $|b'\rangle\!\rangle$ for $b' \in \{\mathtt{0},\mathtt{1}\}^n \setminus \{b\}$; the column vectors corresponding to these states have exactly one 1. If a randomized computation take the deterministic input state b to a state $|p_b\rangle\!\rangle$, then the transformation performed by this computation can be written as:

$$\sum_{b\in\{\mathtt{0},\mathtt{1}\}^n} |p_b\rangle\!\rangle\langle\!\langle b|.$$

When we substitute the corresponding column and row vectors for $|p_b\rangle\!\rangle$ and $\langle\!\langle b|$, then the expression becomes precisely the matrix corre-

sponding to the computation. For example, the randomized transformation (two input register, one output register)

$$\begin{pmatrix} 1 & 1/2 & 1/3 & 0 \\ 0 & 1/2 & 2/3 & 1 \end{pmatrix}$$

can be written as

$$|0\rangle\!\rangle\langle\!\langle 00| + \left(\frac{1}{2}|0\rangle\!\rangle + \frac{1}{2}|1\rangle\!\rangle\right)\langle\!\langle 01| + \left(\frac{1}{3}|0\rangle\!\rangle + \frac{2}{3}|1\rangle\!\rangle\right)\langle\!\langle 10| + |1\rangle\!\rangle\langle\!\langle 11|.$$

Summary

- We use $|p\rangle\!\rangle$, $|q\rangle\!\rangle$, etc., to represent classical states; for randomized computation, these states can be associated with probability vectors, e.g., the result of applying the \$ gate on n-registers corresponds to the probability vector with 2^n entries, each $1/2^n$.
- Classical random variables (including indicator random variables of events) are represented using $\langle\!\langle X|$, $\langle\!\langle Y|$, ..., $\langle\!\langle$odd parity$|$, etc., and associated with appropriate row vectors. In particular, the expectation of the random variable X is $\langle\!\langle X|p\rangle\!\rangle$, and the probability of an event A is $\langle\!\langle I_A|p\rangle\!\rangle$.
- An expression of the form $|p\rangle\!\rangle\langle\!\langle b|$ can be thought of as a (partial) input output transformation that maps the deterministic input state $\langle\!\langle b|$ to the probabilistic state $|p\rangle\!\rangle$; we can combine such expressions to obtain expressions for transformations performed by randomized computation.

Looking ahead: quantum states and measurements

The state[1] of a quantum system with n registers is described by associating *amplitudes* with each outcome in $\{0,1\}^n$. So the state of a single qubit register associates amplitudes with each of the outcomes 0 and 1 (just as the randomized state associated probabilities with them). We write these states as

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle\,,$$

where α and β are complex numbers[2] such that $|\alpha|^2 + |\beta|^2 = 1$. We represent these states by column vectors of the form $\begin{pmatrix}\alpha\\ \beta\end{pmatrix}$. States of the form $|b\rangle$ ($b \in \{0,1\}^n$) are *computational basis states*; the amplitude is 1 for the 'outcome' b and zero for all other outcomes in $\{0,1\}^n$.

The role that was earlier played by indicators random variable of an event is played by objects that we write as $\langle\psi|$. E.g., for a system

[1] Here we only consider *pure* states; to deal with quantum systems in full-generality, we must consider *mixed* states, which can be thought of as probabilistic generalizations of pure states.

[2] That these coefficients are complex will not be crucial for most of the discussion. In fact, all quantum algorithms can be implemented without recourse to quantum coefficients.

with two registers, we have $\langle 00|$, such that $\langle 00\rangle b = 1$ if $b = 00$ and 0 if $b \in \{0,1\}^2 \setminus \{00\}$. In general, the vector $\langle\psi|$ is associated with a complex row vector of unit length; the sum of the square of the absolute values of its entries should be 1. In general, quantum state $|\psi\rangle$ (which need not be one of the computational basis states). In general, we have for each state $|\psi\rangle$, we have a dual vector $\langle\psi|$ such that $\langle\psi|\psi\rangle = 1$ and $\langle\psi|\psi\rangle = 0$ for all states that are *orthogonal* to $|\psi\rangle$. The row vector associated with $\langle\psi|$ is the conjugate transpose of the column vector corresponding to $|\psi\rangle$.

The quantity $|\langle\psi|\phi\rangle|^2$ (note the square) is closely related to the probability of the outcomes that result from quantum operation called *measurement*. Understanding these will be one of the goals of this boot camp.