Lecture Notes on Quantum Computing and Quantum Information Processing.

Instructor: Sudebkumar Prasant Pal, Scribe: Sima Das

5th March, 2005

1 Quantum algorithms

1.1 Simon's Algorithm

Earlier we studied Deutsch's problem and the Deutsch-Jozsa problem. Deterministic answers of whether a given function is constant or balanced over a single bit input and an n-bit input string, respectively, are computed using a single controlled quantum function gate provided as a black box. The promise in the second case is that the function is either constant or balanced. Similarly, the Bernstein-Vazirani problem can be solved using a single quantum controlled function gate. In this problem we find an unknown bit string a deterministically. It turns out that we achieve a linear speedup over the best classical solution. In this discussion, we encounter a different type of problem where we derive a probabilistic solution with linear computation time in contrast to the provably exponential time requirement in the classical case. The problem and the solution famously known as Simon's, go as follows.

Consider a function defined as $f : \{0,1\}^n \to \{0,1\}^n$ that is a periodic two-to-one mapping. The problem deals with computing the period of the function. More precisely, for two distinct elements x, y (n-bit integers) from the domain, $f(x) = f(y)$ if and only if $y = x \oplus s$, where the $\oplus$ operation is bitwise modulo 2 addition. We have to find out two distinct integers x and y with $f(x) = f(y)$ and then find the period $x \oplus y$. Finding such a match and hence the period requires an exponential number of trials in the classical case. More precisely, we state the problem as follows. For $n \geq 1$ and a function $f : F_2^n \to R$, where R is finite set, and promise that there exists a nonzero element $s \in F_2^n$ such that for all $x, y \in F_2^n, f(x) = f(y)$ given $x <> y$ iff $x = y \oplus s$, find s. As the domain of the problem is of n bits, there will be a total of $N = 2^n$ elements. Below we give the argument for an exponential number of trials required for solving the problem classically.

We pick elements at random. The probability that we fail to find a match after the first element is picked is $(N-2)/(N-1)$, as we have $N-1$ remaining elements for selection in future trials and $N-2$ of them are unfavourable (exactly one is favourable for the first pick). After the second trial we have $N-2$ remaining elements to select from and $N-4$ unfavourable cases out of these $N-2$ (as the first two picks will match with two out of remaining $N-2$). So the probability of failure after the second trial is $(N-4)/(N-2)$. Continuing in this manner, the $(m+1)$st trial has probability of failure $(N-2m)/(N-m)$. Failure probability after $(m+1)$ trials is therefore $P = (N-2)/(N-1) * (N-4)/(N-2) * \ldots * (N-2m)/(N-m) = (1-2/N)/(1-1/N) * (1-4/N)/(1-2/N) * \ldots * (1-2m/N)/(1-m/N)$. For small x we can approximate $1-x \approx e^{-x}$, to simplify the above expression as $= e^{-2/N} * e^{-4/N} * \ldots e^{-2m/N} / e^{-1/N} * e^{-2/N} * \ldots e^{-m/N} = e^{-(2/N+4/N+\ldots+2m/N)+(1/N+2/N+\ldots+m/N)} = e^{-(1/N+2/N+\ldots+m/N)} = e^{-m(m+1)/2N}$. This failure probability approaches unity if $m(m+1)/2$ is sufficiently small with respect to N. So, for appreciable success probability m should be of the order of $\sqrt{N}$. Therefore, we have to make an exponential number of trials before getting a match. If $n = 100$ bits then we need approximately $2^{n/2} = 2^{50} = 10^{15}$ trials to get a good chance for finding s. Now we will see how quantum parallelism and superposition provide a linear time solution.

The procedure is as follows: The inpitial state is $|0\rangle_n|0\rangle_n$. Applying the Hadamard operator on each qubit of the first $|0\rangle_n$ qubit vector, gives the superposition state $1/2^{n/2}\sum_{x=0}^{2^n-1}|x\rangle_n|0\rangle_n$. The unitary evolution operator $(U_f : |x,y\rangle \rightarrow |x, y\oplus f(x)\rangle)$ converts it to $1/2^{n/2}\sum_{x=0}^{2^n-1}|x\rangle_n|f(x)\rangle_n$. Again applying $H^{\otimes n}$ on only the first n-qubit register gives $1/2^n\sum_{x=0}^{2^n-1}\sum_{y=0}^{2^n-1}(-1)^{x\cdot y}|y\rangle_n|f(x)\rangle_n$. We measure the resulting state to get $|y, f(x)\rangle$. For two distinct vectors x say x', as per the assumption in the in the problem definition, $f(x) = f(x') \Leftrightarrow x' = x\oplus s$ and $s \neq 0_n$. So, for each x and x', $|y, f(x)\rangle$ and $|y, f(x\oplus s)\rangle$ are identical. Their total amplitude is $1/2^n((-1)^{x\cdot y} + -(1)^{(x\oplus s)\cdot y})$. If $y.s = 0$ mod 2 then $x.y = (x\oplus s).y$ mod 2. So the total amplitude becomes $(-1)^{x\cdot y} * 2 * 2^{-n} = 1/2^{n-1}(-1)^{x\cdot y}$. By applying the above steps $n-1$ times we obtain $n-1$ vectors y_i, such that $y_1.s = 0, y_2.s = 0, y_3.s = 0, \ldots y_{n-1}.s = 0$.

THe application of $H^{\otimes n}$ twice is above is repeated until we obtain a set of $n-1$ linearly independent y's. Solving the set of $n-1$ linearly independent equations we get the value of s, the required period. As in the beginning we have defined f to be an two-one mapping, we do not have to check for $f(0)$ or $f(s)$. If the definition of f is not restricted in the beginning then we need to check whether f is one-one $(f(0) \neq f(s))$ or two-one$(f(0) = f(s))$. The total computation time for this entire algorithm is bounded by the number of repetations times the time $t(n)$ required for a single evaluation of f on input n-qubit string and the time $g(n)$ required to solve the $n-1$ linearly independent equations. The required time is $O(nt(n) + g(n))$.

Now we will see how the probability of sucess (that is the probability of obtaining $n-1$ linearly independent equations) is $\geq 1/4$. Consider $y^1, y^2, \ldots, y^{n-1}$ be the $n-1$ linearly independent vectors. For any set of $i-1$ vectors there are 2^{i-1} vectors that are resulted as linear combination of these vectors. Hence the probability that a vector y^i out of the 2^{n-1} vectors is linearly independent of the first $i-1$ vectors say $y^1, \ldots, y^{i-1}$ is $(2^{n-1} - 2^{i-1})/2^{n-1} = 1 - 2^{i-1}/2^{n-1} = 1 - 1/2^{n-i}$. Now using principle of induction we obtain probability that the vector $y^1$is independent= $1 - 1/2^{n-1}$. Probability that the vectory^2 is linearly independent with y^1 is $(2^{n-1} - 2^1)/2^{n-1} = 1 - 1/2^{n-2}$. Proceeding similarly we obtain y^k is linearly independent of $y^1, \ldots, y^{k-1}$ is $1 - 1/2^{n-k}$. So y^{k+1} being linearly independent of $y^1, \ldots, y^k$ is $1 - 1/2^{n-k-1}$. So the probability that all $n-1$ vectors are linearly independent is $(1 - 1/2^{n-1})(1 - 1/2^{n-2})\ldots(1 - 1/2)$. We know $(1-a)(1-b)\ldots > 1 - a - b - \ldots$. So $(1 - 1/2^{n-1})(1 - 1/2^{n-2})\ldots(1 - 1/2) > 1 - 1/2^{n-2} - 1/2^{n-2} - \ldots - 1/2$. Probability that $y^1, \ldots, y^{n-2}$ are linearly independent is $(1-1/2^{n-1})(1-1/2^{n-2})\ldots(1-1/4)$. But $(1-1/2^{n-1})(1-1/2^{n-2})\ldots(1-1/4) > 1 - 1/2^{n-1} - 1/2^{n-2}\ldots - 1/4$. $1/2^{n-1} + 1/2^{n-2}\ldots + 1/4 \leq 1/2$. Substituting we obtain $(1-1/2^{n-1})(1-1/2^{n-2})\ldots(1-1/4) > 1 - 1/2^{n-1} - 1/2^{n-2}\ldots - 1/4 \geq 1/2$. The probability that y^{n-1} is linearly independent of the remaining vectors is $(2^{n-1} - 2^{n-2})/2^{n-1} = 1/2$. So, the probability that all $n-1$ equations are linearly independent is the probability that $y^1, \ldots, y^{n-2}$ are linearly independent times the probability that y^{n-1} is linearly independent with remaining set. This probability is therefore $\geq 1/4$.