Quantum Computation

(Lecture 7)

Luís Soares Barbosa

Universidade do Minho

HASLab HIGH-ASSURANCE SOFTWARE LABORATORY

INL INTERNATIONAL IBERIAN NANOTECHNOLOGY LABORATORY

UNITED NATIONS UNIVERSITY UNU-EGOV

Quantum Computing Course Unit

Universidade do Minho, 2021

Quantum algorithms

The use of superposition as a basic quantum resource was been essential for all algorithms studied until now, illustrating

- the phase kick-back technique (Deutsch-Joza)
- the phase amplification technique (Grover)

Superposition introduces 'quantum parallelism', whose miracle is, to a great extent, only apparent.

Actually, the result of the calculation is not 2^n evaluations of f: those evaluations characterize the form of the state that describes the output of the computation.

Quantum algorithms

What works indeed?

- What remains is the fact that the random selection of the x, for which $f(x)$ can be learned, is made only after the computation has been carried out.
- Note that asserting that the selection was made before the computation corresponds to look at a superposition as merely a probabilistic phenomenon (i.e. the qubit described by a superposition is actually in one or the other of the basis states).
- Further computation makes possible to extract useful information about relations between several different values of x, which a classical computer could get only by making several independent evaluations.

Quantum algorithms

What works indeed?

- The price to be paid is the loss of the possibility of learning the actual value $f(x)$ for any individual x — cf Heisenberg uncertainty principle.
- cf the mistaken view that the quantum state encodes a property inherent to the qubits: it rather encodes only the possibilities available for the extraction of information from them.

Two further algorithms

1. Bernstein-Vazirani algorithm
2. Simon's algorithm, linking to the next lecture on the quantum Fourier transform and the hidden subgroup problem.

The Bernstein-Vazirani algorithm

The problem

Let w be an unknown non-negative integer less than 2^n and consider a function $f(x) = w \cdot x$, where

$$w \cdot x \;=\; w_1 x_1 + w_2 x_2 + \cdots + w_n x_n$$

i.e. the bitwise product of x and z, modulo 2.

How many times one has to call f to determine the value of the integer w?

- Classically, n times: the n values $w \cdot 2^m$, for $0 \leq m < n$.
- In a quantum computer a single invocation is enough, regardless of the number n of bits.

The Bernstein-Vazirani algorithm

- Re-use the Deutsch-Joza circuit
- Superposition

$$H^{\otimes n}|x\rangle = \frac{1}{\sqrt{2^n}} \sum_{y_n=0}^{1} \cdots \sum_{y_1=0}^{1} (-1)^{\sum_{j=1}^{n} x_j y_j} |y_n\rangle \cdots |y_1\rangle$$

$$= \frac{1}{\sqrt{2^n}} \sum_{y=0}^{2_n-1} (-1)^{x \cdot y} |y\rangle_n$$

cf

$$H|x\rangle = \frac{1}{\sqrt{2}}(|0\rangle + (-1)^x|1\rangle) = \frac{1}{\sqrt{2}} \sum_{y=0}^{1} (-1)^{xy}|y\rangle$$

The Bernstein-Vazirani algorithm

Putting everything together,

$$
\begin{aligned}
&(H^{\otimes n} \otimes H) U_f (H^{\otimes n} \otimes H)\, |0\rangle|1\rangle \\
&= (H^{\otimes n} \otimes H) U_f \left(\frac{1}{\sqrt{2^n}} \sum_{x=0}^{2^n-1} |x\rangle \right) \frac{1}{\sqrt{2}} (|0\rangle - |1\rangle) \\
&= \frac{1}{\sqrt{2^n}} H^{\otimes n} \left(\sum_{x=0}^{2^n-1} (-1)^{f(x)} |x\rangle \right) H \left(\frac{1}{\sqrt{2}} (|0\rangle - |1\rangle) \right) \\
&= \frac{1}{2^n} \sum_{x=0}^{2^n-1} \sum_{y=o}^{2^n-1} (-1)^{f(x)+x\cdot y} |y\rangle|1\rangle \\
&= |w\rangle|1\rangle
\end{aligned}
$$

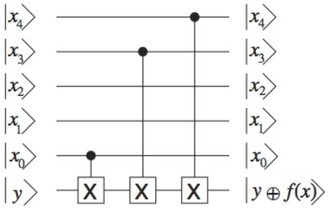
The Bernstein-Vazirani algorithm: another explanation

Some oracles can be implemented by simple circuits.

- In this case the action of U_f on the computational basis is to flip the 1 qubit target register once, whenever a bit of x and the corresponding bit of w are both 1.
- Put one CNOT for each nonzero bit of w, controlled by the qubit representing the corresponding bit of x.
- Their combined effect on every computational basis state is precisely that of U_f.

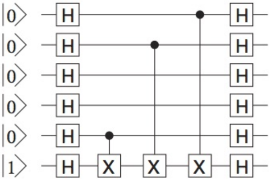
The Bernstein-Vazirani algorithm: another explanation

Example of the encoding for $w = 11001$



The Bernstein-Vazirani algorithm: another explanation

Enveloping U_f into the algorithm



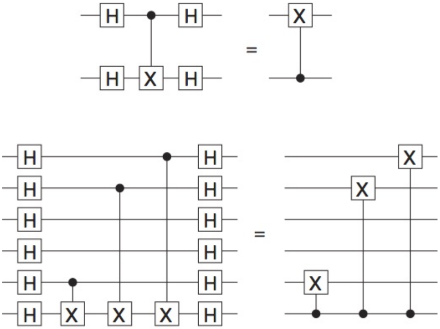
The effect is to convert every CNOTgate in the equivalent representation of U_f from C_{ij} to

$$C_{ji} = (H_i H_j) C_{ij} (H_i H_j)$$

reversing the target and control qubits.

The Bernstein-Vazirani algorithm: another explanation

Actually,



The Bernstein-Vazirani algorithm: another explanation

Thus

- After the reversal, the target register controls every one of the CNOT gates, and since the state of the target register is $|1\rangle$, every one of the NOT operators acts.
- That action flips just those qubits of the control register for which the corresponding bit of w is 1.
- Since the control register starts in the state $|0\rangle$, this changes the state of each qubit of the control to $|1\rangle$, iff it corresponds to a nonzero bit of w.
- Thus, in the end, the state of the input register changes from $|0\rangle$ to $|w\rangle$.

Simon's algorithm

The problem

Let $f : 2^n \longrightarrow 2^n$ be such that for some $s \in 2^n$,

$$f(x) = f(y) \text{ iff } x = y \text{ or } x = y \oplus s$$

Find s.

Equivalent formulation as a period-finding problem

Determine the period s of a function f periodic under $\oplus$:

$$f(x \oplus s) = f(x)$$

Note that f is bijective if $s = 0$ (because $x \oplus y = 0$ iff $x = y$), and two-to-one otherwise (because, for a given s there is only a pair of values x, y such that $x \oplus y = s$).

Simon's algorithm, classically

Compute f for sequence of values until finding a value x_j such that $f(x_j) = f(x_i)$ for a previous x_i. Then

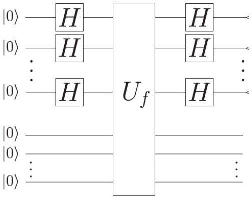
$$s = x_j \oplus x_i$$

- At any previous stage, if this procedure has picked m different values of x, then one concludes that $s \neq x_j \oplus x_i$ for all such values.
- Thus, at most
$$\frac{1}{2}m(m-1)$$
possible values for s have been discarded (vs $2^n - 1$ possible values for s).
- The procedure is unlike to succeed until m becomes of the order of $\sqrt{2^n}$ — the execution time grows exponentially with the number of bits n.

Going quantum

Reuse the circuit from the Deutsch-Joza algorithm but expand both registers to n qubits

The circuit



where

$$U_f = |x\rangle|c\rangle \mapsto |x\rangle|c \oplus f(x)\rangle$$

Going quantum

The oracle maps

$$\frac{1}{\sqrt{2^n}} \sum_{x \in 2^n} |x\rangle|0\rangle \quad \text{to} \quad \frac{1}{\sqrt{2^n}} \sum_{x \in 2^n} |x\rangle|f(x)\rangle$$

because $0 \oplus x = x$.

A measurement of the target register choose randomly one of the 2^{n-1} possible outcomes of f as f gives the same output for x and $x \oplus s$, to 2^n possible inputs correspond 2^{n-1} possible outcomes

This measurement is not very useful (why?).

Note, however, if $f(k)$ was measured, the control register contains superposition

$$\frac{1}{\sqrt{2}}(|k\rangle + |k \oplus s\rangle)$$

as they are the unique values yielding $f(k)$

Basic insight: the effect of $H^{\otimes n}$

Recall

$$H|x\rangle = \frac{1}{\sqrt{2}} \sum_{z \in 2} (-1)^{xz} |z\rangle$$

which extends to a n-qubit as follows

$$\begin{aligned}
H^{\otimes n}|x\rangle &= H|x_1\rangle H|x_2\rangle \cdots H|x_n\rangle \\
&= \frac{1}{\sqrt{2}} \sum_{z_1 \in 2^n} (-1)^{x_1 z_1} |z_1\rangle + \frac{1}{\sqrt{2}} \sum_{z_2 \in 2^n} (-1)^{x_2 z_2} |z\rangle \cdots \frac{1}{\sqrt{2}} \sum_{z_n \in 2^n} (-1)^{x_n z_n} |z_n\rangle \\
&= \frac{1}{\sqrt{2^n}} \sum_{z_1, z_2, \cdots, z_n \in 2^n} (-1)^{x_1 z_1 + x_2 z_2 + \cdots + x_n z_n} |z_1 z_2 \cdots z_n\rangle \\
&= \frac{1}{\sqrt{2^n}} \sum_{z \in 2^n} (-1)^{x \cdot z} |z\rangle
\end{aligned}$$

Basic insight: the effect of $H^{\otimes n}$

Consider now a particular case: applying $H^{\otimes n}$ to a superposition of two basis states, e.g. $|0\rangle$ and $|s\rangle$:

$$H^{\otimes n}\left(\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|s\rangle\right) = \frac{1}{\sqrt{2^{n+1}}}\sum_{z\in 2^n}|z\rangle + \frac{1}{\sqrt{2^{n+1}}}\sum_{z\in 2^n}(-1)^{s\cdot z}|z\rangle$$
$$= \frac{1}{\sqrt{2^{n+1}}}\sum_{z\in 2^n}((1+(-1)^{s\cdot z})|z\rangle$$

- $s \cdot z = 1 \Rightarrow$ basis state $|z\rangle$ vanishes (because $1 + (-1)^1 = 0$)
- $s \cdot z = 0 \Rightarrow$: basis state $|z\rangle$ is kept with amplitude $\frac{2}{\sqrt{2^{n+1}}} = \frac{1}{\sqrt{2^{n-1}}}$

Basic insight: the effect of $H^{\otimes n}$

$$
\begin{aligned}
H^{\otimes n}\left(\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|s\rangle\right) &= \frac{1}{\sqrt{2^{n-1}}} \sum_{z \in \{x \in 2^n \mid s \cdot z = 0\}} |z\rangle \\
&= \frac{1}{\sqrt{2^{n-1}}} \sum_{z \in S^{\perp}} |z\rangle
\end{aligned}
$$

$S^{\perp}$, for $S = \{0, s\}$ is the orthogonal complement of subspace S, with $\dim(S^{\perp}) = n - 1$
(because $\dim(S) = 1$, as S is the subspace generated by s).

Recall that for a subspace F of V, $F^{\perp} = \{v \in V \mid \forall_{x \in F}.\ x.v = 0\}$

Basic insight: the effect of $H^{\otimes n}$

In general,

$$\begin{aligned} H^{\otimes n}\left(\frac{1}{\sqrt{2}}|x\rangle + \frac{1}{\sqrt{2}}|y\rangle\right) &= \frac{1}{\sqrt{2^{n+1}}}\sum_{z\in 2^n}(-1)^{x\cdot z}|z\rangle + \frac{1}{\sqrt{2^{n+1}}}\sum_{z\in 2^n}(-1)^{y\cdot z}|z\rangle \\ &= \frac{1}{\sqrt{2^{n+1}}}\sum_{z\in 2^n}\underbrace{\left((-1)^{x\cdot z} + (-1)^{y\cdot z}\right)}_{(\star)}|z\rangle \\ &= \frac{1}{\sqrt{2^{n-1}}}\sum_{z\in\{0,x\oplus y\}^{\perp}}(-1)^{x\cdot z}|z\rangle \end{aligned}$$

because expression $(\star)$ yields 0 whenever $x \oplus y = 1$.

Putting everything together

Given the initial state $\frac{1}{\sqrt{2^n}} \sum_{x \in 2^n} |x\rangle|0\rangle$, the oracle produces

$$\frac{1}{\sqrt{2^n}} \sum_{2^n} |x\rangle|f(x)\rangle$$

which, as seen above, can be rewritten as

$$\frac{1}{\sqrt{2^{n-1}}} \sum_{x \in I} \frac{1}{\sqrt{2}} (|x\rangle + |x \oplus s\rangle)|f(x)\rangle$$

because 2^n can be partitioned into 2^{n-1} sets of strings $\{x, x \oplus s\}$.
Set I is composed of one representative of each such set.

Note

Technically each pair of strings is a coset of the subgroup $S = \{0, s\}$.

Recall: coset

The coset of a subgroup S of a group $(G, .)$ wrt $g \in G$ is

$$gS \; = \; \{g.s \mid s \in S\}$$

In this case the vector space $(Z_2)^n$, whose elements are n-tuples over 2, with dimension n, forms a group $((Z_2)^n, \oplus)$, thus,

$$xS \; = \; \{x \oplus 0, x \oplus s\}$$

Question

Why are there only 2^{n-1} cosets for this group?

Putting everything together

Applying $H^{\otimes n}$ to the control register yields a uniform superposition of elements of $S^{\perp}$:

$$H^{\otimes n}\left(\frac{1}{\sqrt{2}}(|x\rangle + |x \oplus s\rangle)\right) = \frac{1}{\sqrt{2^{n-1}}}\sum_{z \in S^{\perp}} (-1)^{x \cdot z}|z\rangle$$

Such a measurement returns one such z with probability $\frac{1}{2^{n-1}}$.

Putting everything together

Repeating this procedure until n linearly independent values $\{z_1, z_2. \cdots, z_{n-1}\}$ over $(Z_2)^n$ are found, entails the possibility of solving the set of equations:

$$\begin{aligned} z_1 \cdot s &= 0 \\ z_2 \cdot s &= 0 \\ &\vdots \\ z_{n-1} \cdot s &= 0 \end{aligned}$$

The only solutions to this set of equations are 0 and s, so, finally, s is found.

Note that the span of $\{z_1, z_2. \cdots, z_{n-1}\}$ is $S^{\perp}$.

The algorithm

1. Prepare the initial state $\frac{1}{\sqrt{2^n}} \sum_{x \in 2^n} |x\rangle|0\rangle$ and make $i := 1$
2. Apply the oracle U_f to obtain the state

$$\frac{1}{\sqrt{2^n}} \sum_{x \in 2^n} |x\rangle|f(x)\rangle$$

   which can be re-written as

$$\frac{1}{\sqrt{2^{n-1}}} \sum_{x \in I} \frac{1}{\sqrt{2}}(|x\rangle + |x \oplus s\rangle)|f(x)\rangle$$

3. Apply $H^{\otimes n}$ to the control register yielding a uniform superposition of elements of $S^{\perp}$.

The algorithm

4. Measure the first register and record the value observed z_i, which is a randomly selected element of $S^{\perp}$.
5. If the dimension of the span of $\{z_1, z_2, \cdots, z_i\}$ is less than $n-1$, increment i and to go step 2; else proceed.
6. Then

$$\text{span}\{z_1, z_2, \cdots, z_i\} = S^{\perp}$$

Thus, s will be the unique non-zero solution of

$$Z\,s = 0$$

where Z is the matrix whose line i corresponds to vector z_i. Compute this system of linear equations to find s by Gaussian elimination modulo 2 (in time polynomial in n).

Can we do better?

The algorithm computes a solution in polynomial expected running time

- In each iteration i the probability of z_i being linearly independent of the values previously computed is at least 0.5.
- Thus, after $2(n-1)$ iterations the probability of having found a basis for $S^{\perp}$ is also at least 0.5
- The corresponding equations can be solved to find s in $\mathcal{O}(n^2)$
- Thus, with high likelihood s is expected to be found with $\mathcal{O}(n-1)$ calls to the oracle, followed by $\mathcal{O}(n^2)$ steps to solve the equations.

Can we do better?

Can we obtain a polynomial worst-case running time?

There is a basic result on analysing probabilistic algorithms stating that any algorithm that terminates with an expected number of queries equal to n will terminate after at most $3n$queries, with probability at least $\frac{2}{3}$.

This means that one may abandon the iterative process if a solution is not found in $3n$ iterations and find the solution with probability $\frac{2}{3}$.

The revised algorithm

5. If $i \leq 3n$ increment i and to go step 2; else proceed.
6. Solve

$$Z\,s \;=\; 0$$

   Compute this system of linear equations and let s_1, s_2, ... s_n be the generators of the solution space.

7. If the solution space has dimension 1, spanned by s_1, output $s = s_1$, else fail.

This solves Simon's problem with probability $\frac{2}{3}$ using $3n$ evaluations of f.

Generalised Simon's algorithm

The problem

Let $f : 2^n \longrightarrow X$, for some X finite, be such that,

$f(x) = f(y)$ iff $x - y \in S$, for some subspace $S \leq (Z_2)^n$, of dimension m

Find a basis $s_1, s_2, \cdots s_m$ for S.

Generalised Simon's algorithm

- If $S = \{0, x_1, \cdots, x_{2^m-1}\}$ is a subspace of dimension m of Z_2^n, 2^n can be decomposed into 2^{n-m} cosets of the form $y, y \oplus x_1, y \oplus x_2, \cdots, y \oplus x_{2^m-1}$ (abbreviated to $y + S$)
- Step 3 yields

$$\sum_{x \in 2^n} |x\rangle|f(x)\rangle = \frac{1}{\sqrt{2^{n-m}}} \sum_{y \in I} |y + S\rangle|f(x)\rangle$$

where I be a subset of 2^n consisting of one representative of each 2^{n-m} disjoint cosets, and

$$|y + S\rangle = \sum_{s \in S} \frac{1}{\sqrt{2^m}} |f(x)\rangle$$

Generalised Simon's algorithm

- In step 4 the first register is left in a state of the form $|y + S\rangle$ for a random y.
- After applying the Hadamard transformation, the first register contains a uniform superposition of elements of $S^{\perp}$ and its measurement yields a value w_i sampled uniformly at random from $S^{\perp}$.

leading to the revised algorithm:

5. If the dimension of the span of $\{z_1, z_2, \cdots, z_i\}$ is less than $n - m$, increment i and to go step 2; else proceed.
6. Compute the system of linear equations

$$Z s = 0$$

and let $s_1, s_2, \cdots, s_m$ be the generators of the solution space. They form the envisaged basis.

The hidden subgroup problem

The group S is often called the hidden subgroup.

Simon's algorithm is an instance of a much general scheme, leading to exponential advantage, that will be studied next.