Quantum Computation

(Lecture 3)

Luís Soares Barbosa

Quantum Computing Course Unit

Universidade do Minho, 2020

Quantum algorithms

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

- the phase push-up 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 in 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 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

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

i.e. the bitwise product of x and z, modulo 2. Note that addition modulo 2 corresponds to $\oplus$ (xor).

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

- Prepare the single qubit output register as $H|1\rangle$ since oracle U_f applied to $|x\rangle_n|y\rangle_1$ flips the value y of the output register iff $f(x) = 1$. Thus,

$$U_f\,|x\rangle_n \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \;=\; (-1)^{f(x)}\,|x\rangle_n \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$$

converting a bit flip to an overall change of sign.

The Bernstein-Vazirani algorithm

- Superposition

$$
\begin{aligned}
H^{\otimes n}|x\rangle_n &= \frac{1}{\sqrt{2^n}} \sum_{y_n=0}^{1} \cdots \sum_{y_1=0}^{1} (-1)^{\sum_{j=1}^{n} x_j y_j} |y_n\rangle \cdots |y_1\rangle \\
&= \frac{1}{\sqrt{2^n}} \sum_{y=0}^{2_n-1} (-1)^{x\cdot y} |y\rangle_n
\end{aligned}
$$

cf

$$
H|x\rangle_1 = \frac{1}{\sqrt{2}}(|0\rangle + (-1)^x|1\rangle) = \frac{1}{\sqrt{2}} \sum_{y=0}^{1} (-1)^{xy}|y\rangle
$$

The Bernstein-Vazirani algorithm

Putting everything together,

$$\begin{aligned}
&(H^{\otimes n} \otimes I)\,U_f\,(H^{\otimes n} \otimes H)\;|0\rangle_n|1\rangle_1 \\
&\quad = (H^{\otimes n} \otimes I)\,U_f \left(\frac{1}{\sqrt{2^n}} \sum_{x=1}^{2^n} |x\rangle \right) \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \\
&\quad = \frac{1}{\sqrt{2^n}} \left(H^{\otimes n} \sum_{x=1}^{2^n} (-1)^{f(x)} |x\rangle \right) \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \\
&\quad = \frac{1}{2^n} \sum_{x=1}^{2^n} \sum_{y=1}^{2^n} (-1)^{f(x)+x\cdot y} |y\rangle \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \\
&\quad = |w\rangle_n|1\rangle_1
\end{aligned}$$

because

$$\sum_{x=1}^{2^n} (-1)^{w\cdot x} (-1)^{y\cdot x} = \prod_{j=1}^{n} \sum_{x_j=0}^{1} (-1)^{(w_j+y_j)x_j}$$

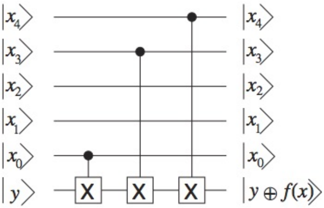
The Bernstein-Vazirani algorithm: another explanation

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 output 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.

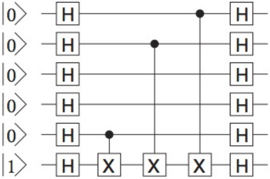
The Bernstein-Vazirani algorithm: another explanation

Example of the encoding for $w = 11001$



The Bernstein-Vazirani algorithm: another explanation

Enveloping U_f into the algorithm



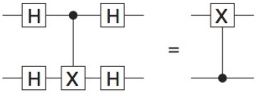
The effect is to convert every CNOTgate in the equivalent representation of U_f from C_{ij} to

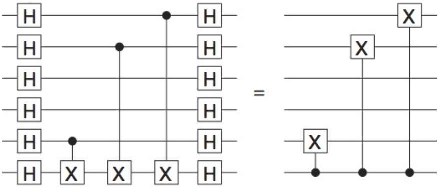
$$C_{ji} = (H_i H_j) C_{ij} (H_i H_j)$$

reversing the target and control qubits.

The Bernstein-Vazirani algorithm: another explanation

Actually,





The Bernstein-Vazirani algorithm: another explanation

Thus

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

Simon's algorithm

The problem

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

$$f(x) = f(y) \text{ iff } x \oplus y \in \{0, s\}.$$

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

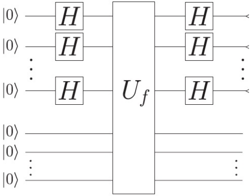
Compute f for sequence of values until finding a value x_j such that $f(x_j) = f(x_i)$ for a previous x_i. Then

$$s \;=\; x_j \oplus x_i$$

- 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: intuition

The circuit



where

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

By repeating the activation of the circuit 'enough' times, find $n-1$ linearly independent n-bit strings $w_1 w_2 ... w_n$ such that, for each i, $w_i \oplus s = 0$, obtaining $n-1$ linear equations in n unknowns (corresponding to the bits of s). Solve the system to find s.

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 $|s\rangle$:

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

- $s.z = 1 \Rightarrow$ basis state $|z\rangle$ vanishes (because $1 + (-1)^1 = 0$)
- $s.z = 0 \Rightarrow$: 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 $H^{\otimes n}$

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

$S^\perp$, 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 $H^{\otimes n}$

Similarly,

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

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

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

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

which can be re-written as

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

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.

The algorithm

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?

The algorithm

3. Apply $H^{\otimes n}$ to the first register yielding a uniform superposition of elements of $S^{\perp}$.
4. Measure the first register and record the value observed w_i, which is a randomly selected element of $S^{\perp}$.
5. If the dimension of the span of $\{w_1, w_2, \cdots, w_i\}$ is less than $n-1$, increment i and to go step 2; else proceed.
6. Then

$$\text{span}\{w_1, w_2, \cdots, w_i\} = S^{\perp}$$

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

$$W\,s = 0$$

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

Can we do better?

Complexity

The expected number of evaluations of f in the execution of the algorithm is less than n

Simons's algorithm computes a solution in polynomial expected running time. 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 $\frac{2}{3}$.

The revised algorithm

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

$$W\, s \;=\; 0$$

   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.

The theorem underneath "zero-error" solutions

Let $E(X) = \sum_{x \in D} x\, Pr(X = x)$ be the expected value of a discrete random variable X over a countable domain S and taking non-negative values. Then, for any constant c,

$$Pr(X \geq c\, E(X)) \;\leq\; \frac{1}{3}$$

Proof

Let W be a random variable st $W = \begin{cases} 0 & \Leftarrow 0 \leq X < cE(X) \\ cE(X) & \Leftarrow X \geq cE(X) \end{cases}$ Since $X \geq W$,

$$E(X) \geq E(W) \;=\; 0Pr(Y = 0) + cE(X)Pr(W = 1) \;=\; cE(X)Pr(X \geq c\, E(X))$$

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 $\{w_1, w_2, \cdots, w_i\}$ is less than $n - m$, increment i and to go step 2; else proceed.
6. Compute the system of linear equations

$$W\,s \;=\; 0$$

   and let $s_1, s_2, \cdots, s_3$ 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.