Quantum Computation
(Lecture 4)

Luís Soares Barbosa

MSc Physics Engineering

Universidade do Minho, 2021-22

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 the set of all 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 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 warming up algorithms

1. Bernstein-Vazirani algorithm
2. Simon's algorithm, bridging to the quantum Fourier transform and the hidden subgroup problem. (to be discussed in the next lecture)

The Bernstein-Vazirani algorithm

The problem

Let w be an unknown non-negative integer less than 2^n, encoded as a bit string, and consider a function which hides secret w as follows:
$f(x) = x \cdot w$, where

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

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

How many times one has to call f to determine w?

- Classically, n times: the n values $2^m \cdot w$, for $0 \leq m < n$. Actually for each 1 in position i,

$$f(00 \cdots 1_i \cdots 0) \;=\; w_i$$

- In a quantum computer a single invocation is enough, regardless of the number n of bits.

The Bernstein-Vazirani algorithm

The components

- An oracle $U_f|x\rangle|z\rangle = |x\rangle|z \oplus f(x)\rangle$, which when applied to $|x\rangle|-\rangle$ transforms $|x\rangle|-\rangle$ into $(-1)^{f(x)}|x\rangle|-\rangle$
(as used in the Deutsch-Joza algorithm.)
- Superposition

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

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

recalling that

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

Putting everything together

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

Putting everything together

$$\cdots = \frac{1}{2^n} \sum_{x=0}^{2^n-1} \sum_{y=0}^{2^n-1} (-1)^{x\cdot(w\oplus y)} |y\rangle|1\rangle = \cdots$$

For each y, $\frac{1}{2^n}\sum_{x=0}^{2^n-1}(-1)^{x\cdot(w\oplus y)}$ is 1 iff $(w \oplus y) = 0$, which happens only if $w = y$ In all other cases $\frac{1}{2^n}\sum_{x=0}^{2^n-1}(-1)^{x\cdot(w\oplus y)}$ is 0.

The reason is easy to guess:

- for $w \oplus y = 0$ $\frac{1}{2^n}\sum_{x=0}^{2^n-1}(-1)^{x\cdot(w\oplus y)} = \frac{1}{2^n}\sum_{x=0}^{2^n-1} 1 = 1$.
- for $w \oplus y \neq 0$, as x spans all numbers from 0 to $2^n - 1$, half of the 2^n factors in the sum will be -1 and the other half 1, thus summing up to 0.

Thus, the only non zero amplitude is the one associated to w.

The Bernstein-Vazirani algorithm: another explanation

The explanation of the algorithm is based, as usual, on the combination

quantum parallelism + suitable manipulation of the resulting superposition

... but, in a sense, this is just an explanation ...

Let us see a different, simpler on (due to David Mermin)

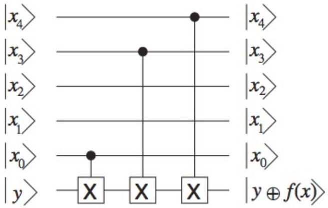
The Bernstein-Vazirani algorithm: another explanation

Observe that 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.

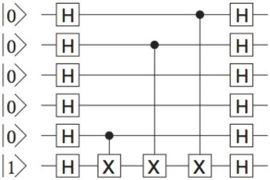
The Bernstein-Vazirani algorithm: another explanation

Example of the encoding for $w = 11001$



The Bernstein-Vazirani algorithm: another explanation

Enveloping U_f into the algorithm



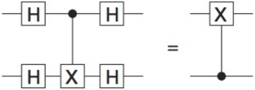
The effect is to convert every CNOTgate in the equivalent representation of U_f from C_{ij} to

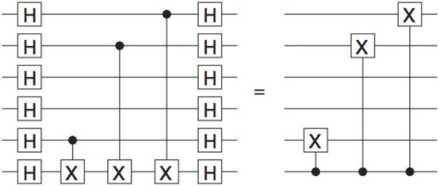
$$C_{ji} = (H_i H_j) C_{ij} (H_i H_j)$$

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 $|w\rangle$.