Description of Grover's algorithm

In this section, we'll describe Grover's algorithm. We'll begin by discussing *phase query gates* and how to build them, followed by the description of Grover's algorithm itself. Finally, we'll briefly discuss how this algorithm is naturally applied to searching.

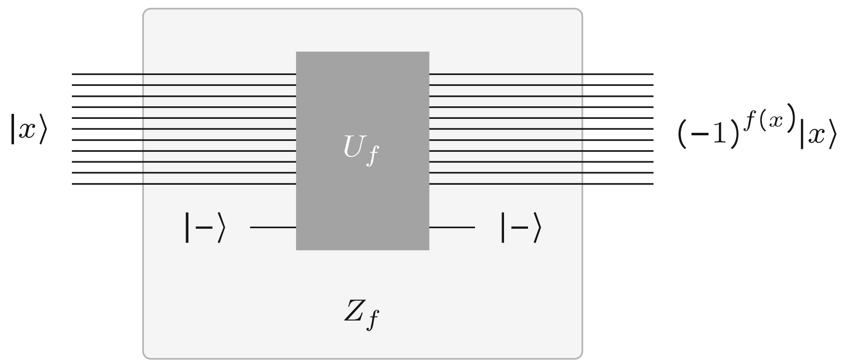
Phase query gates

Grover's algorithm makes use of operations known as *phase query gates*. In contrast to an ordinary query gate U_f, defined for a given function f in the usual way described previously, a phase query gate for the function f is defined as

$$Z_f|x\rangle = (-1)^{f(x)}|x\rangle$$

for every string $x \in \Sigma^n$.

The operation Z_f can be implemented using one query gate U_f as this diagram suggests:



This implementation makes use of the phase kickback phenomenon, and requires that one workspace qubit, initialized to a $|-\rangle$ state, is made available. This qubit remains in the $|-\rangle$ state after the implementation has completed, and can be reused (to implement subsequent Z_f gates, for instance) or simply discarded.

In addition to the operation Z_f, we will also make use of a phase query gate for the n-bit OR function, which is defined as follows for each string $x \in \Sigma^n$.

$$\mathrm{OR}(x) = \begin{cases} 0 & x = 0^n \\ 1 & x \neq 0^n \end{cases}$$

Explicitly, the phase query gate for the n-bit OR function operates like this:

$$Z_{\mathrm{OR}}|x\rangle = \begin{cases} |x\rangle & x = 0^n \\ -|x\rangle & x \neq 0^n. \end{cases}$$

To be clear, this is how Z_{OR} operates on standard basis states; its behavior on arbitrary states is determined from this expression by linearity.

The operation Z_{OR} can be implemented as a quantum circuit by beginning with a Boolean circuit for the OR function, then constructing a U_{OR} operation (that is, a standard query gate for the n-bit OR function) using the procedure described in the *Quantum algorithmic foundations* lesson, and finally a Z_{OR} operation using the phase kickback phenomenon as described above. Notice that the operation Z_{OR} has no dependence on the function f and can therefore be implemented by a quantum circuit having no query gates.

Description of the algorithm

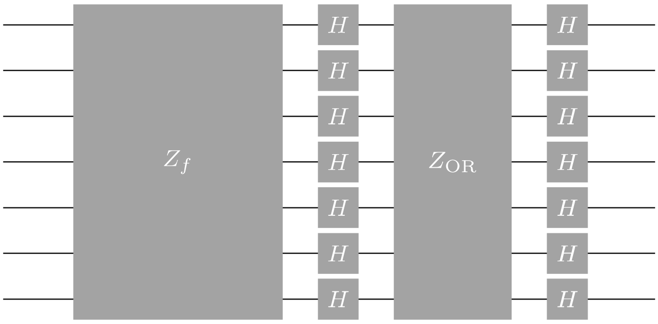
Now that we have the two operations Z_f and Z_{OR}, we can describe Grover's algorithm.

The algorithm refers to a number t, which is the number of *iterations* it performs (and therefore the number of *queries* to the function f it requires). This number t isn't specified by Grover's algorithm as we're describing it, and we'll discuss later in the lesson how it can be chosen.

Grover's algorithm

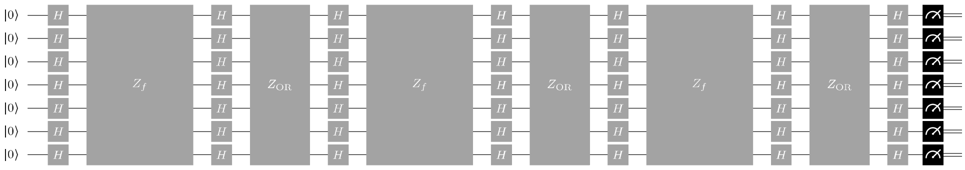
1. Initialize an n qubit register Q to the all-zero state $|0^n\rangle$ and then apply a Hadamard operation to each qubit of Q.
2. Apply t times the unitary operation $G = H^{\otimes n} Z_{\mathrm{OR}} H^{\otimes n} Z_f$ to the register Q
3. Measure the qubits of Q with respect to standard basis measurements and output the resulting string.

The operation $G = H^{\otimes n} Z_{\mathrm{OR}} H^{\otimes n} Z_f$ iterated in step 2 will be called the *Grover operation* throughout the remainder of this lesson. Here is a quantum circuit representation of the Grover operation when $n = 7$:



In this diagram, the Z_f operation is depicted as being larger than Z_{OR} as an informal visual clue to suggest that it is likely to be the more costly operation. In particular, when we're working within the query model, Z_f requires one query while Z_{OR} requires no queries. If instead we have a Boolean circuit for the function f, and then convert it to a quantum circuit for Z_f, we can reasonably expect that the resulting quantum circuit will be larger and more complicated than one for Z_{OR}.

Here's a diagram of a quantum circuit for the entire algorithm when $n = 7$ and $t = 3$. For larger values of t we can simply insert additional instances of the Grover operation immediately before the measurements.



Application to search

Grover's algorithm can be applied to the *Search* problem as follows:

- Choose the number t in step 2. (This is discussed later in the lesson.)
- Run Grover's algorithm on the function f, using whatever choice we made for t, to obtain a string $x \in \Sigma^n$.
- Query the function f on the string x to see if it's a valid solution:
    - If $f(x) = 1$, then we have found a solution, so we can stop and output x.
    - Otherwise, if $f(x) = 0$, then we can either run the procedure again, possibly with a different choice for t, or we can decide to give up and output "no solution."

Once we've analyzed how Grover's algorithm works, we'll see that by taking $t = O(\sqrt{N})$, we obtain a solution to our search problem (if one exists) with high probability.

Previous page Next page