Workshop on Quantum Computation using IBM Q

Salvador E. Venegas-Andraca
Juan Arturo Silva-Ordaz

Tecnológico de Monterrey, Escuela de Ingeniería y Ciencias

Table of Contents

Problem statement

Grover's Algorithm is a quantum algorithm that finds with high probability the unique input to a **black box** function that produces a particular output value, using $\mathcal{O}(\sqrt{N})$ evaluations of the function, where N is the size of the function's domain. The analogous problem in classical computation cannot be solved in fewer than $\mathcal{O}(N)$ evaluations. It was proved by Bernstein, Bennet, Brassard, and Vazirani that Grover's algorithm is asymptotically optimal.

Applications

Although the purpose of Grover's algorithm is usually described as "searching a database", it may be more accurate to describe it as an "inverting function".

Applications

Although the purpose of Grover's algorithm is usually described as "searching a database", it may be more accurate to describe it as an "inverting function".
If a function $y = f(x)$ can be evaluated on a quantum computer, Grover's algorithm calculates x when given y. Inverting a function is related to the searching of a database because we could come up with a function that produces one particular value of y (e.g. `TRUE`) if x matches a desired entry in a database, and another value of y (e.g. `FALSE`) for other values of x.

Setup

Consider an unsorted database with N entries. The algorithm requires an $N-$ dimensional state space H, which can be supplied with $n = \log_2 N$qubits. Consider the problem of determining the index of the database entry that satisfies some such criterion.

Setup

Consider an unsorted database with N entries. The algorithm requires an $N-$ dimensional state space H, which can be supplied with $n = \log_2 N$qubits. Consider the problem of determining the index of the database entry that satisfies some such criterion.

Let f be the function that maps database entries to 0 or 1, where $f(x) = 1$ if and only if x satisfies the search criterion $(x = \omega)$

Setup

We are provided with quantum black box access to a subroutine in the form of a unitary operator U_ω that acts as follows:

$$\begin{cases} U_\omega |x\rangle = -|x\rangle & \text{, for } x = \omega \\ U_\omega |x\rangle = |x\rangle & \text{, for } x \neq \omega \end{cases}$$

Tecnológico de Monterrey

IBM

Setup

If we use an ancillary qubit, the operator acts in the following way:

$$\begin{cases} U_\omega \left|x\right\rangle \left|y\right\rangle = \left|x\right\rangle \left|\neg y\right\rangle & \text{, for } x = \omega \\ U_\omega \left|x\right\rangle \left|y\right\rangle = \left|x\right\rangle \left|y\right\rangle & \text{, for } x \neq \omega \end{cases}$$

Setup

If we use an ancillary qubit, the operator acts in the following way:

$$\begin{cases} U_\omega |x\rangle |y\rangle = |x\rangle |\neg y\rangle & \text{, for } x = \omega \\ U_\omega |x\rangle |y\rangle = |x\rangle |y\rangle & \text{, for } x \neq \omega \end{cases}$$

This is the same as

$$U_\omega |x\rangle |y\rangle = |x\rangle |y \oplus x\rangle$$

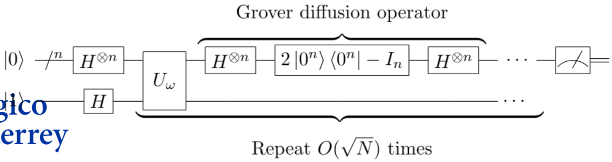
Setup

If we use an ancillary qubit, the operator acts in the following way:

$$\begin{cases} U_\omega |x\rangle |y\rangle = |x\rangle |\neg y\rangle & , \text{ for } x = \omega \\ U_\omega |x\rangle |y\rangle = |x\rangle |y\rangle & , \text{ for } x \neq \omega \end{cases}$$

This is the same as

$$U_\omega |x\rangle |y\rangle = |x\rangle |y \oplus x\rangle \,(U_f |x\rangle = (-1)^{f(x)} |x\rangle$$



Algorithm steps

Let $|s\rangle$ denote the uniform superposition over all states

$$|s\rangle = \frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} |x\rangle$$

Algorithm steps

Let $|s\rangle$ denote the uniform superposition over all states

$$|s\rangle = \frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} |x\rangle$$

The operator

$$U_s = 2\,|s\rangle\,\langle s| - I$$

is known as the Grover diffussion operator.

Algorithm steps

1. Initialize the system to the state

$$|s\rangle = \frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} |x\rangle$$

2. Perform the Grover iteration $r(N)$ times.
    1. Apply the operator U_ω
    2. Apply the operator U_s
3. Perform the measurement.