

Quantum Computing

Lab 2. Algorithms

Frank C Langbein

```
[1]: from qutip import *  
import numpy as np
```

Oracles

Create a quantum circuit to implement the following oracles as sign and bit oracle:

1. f_1 given by

x_2	x_1	$f_1(x_1, x_2)$
0	0	0
0	1	0
1	0	1
1	1	1

- 2.

$$f_2(x_1, x_2, x_3, x_4) = (x_1 \text{ XOR } x_2) \text{ AND } (x_3 \text{ AND } x_4)$$

- 3.

$$f_3(x_1, x_2, x_3) = \text{NOT}(x_1 \text{ AND } x_2 \text{ AND } x_3) \text{ OR } (x_1 \text{ AND } x_2 \text{ AND } x_3)$$

Solution

1. Bit Oracles
 - f_1 Bit Oracle
 - f_2 Bit Oracle
 - f_3 Bit Oracle
2. Sign Oracles
 - f_1 Sign Oracle
 - f_2 Sign Oracle
 - f_3 Sign Oracle

Grover's Search

Create a quantum circuit to implement Grover's search to find $|1011\rangle$.

Solution

Grover circuit for $|1011\rangle$

Or

Grover circuit for $|1011\rangle$

Reconstructing an Oracle

Assume you are given a sign oracle O_f for a function of the type

$$f(x_1, \dots, x_n) = x_1 a_1 \oplus x_2 a_2 \oplus \dots \oplus x_n a_n$$

where x_l and a_l are n -bit strings and \oplus is the addition modulo 2. I.e. f calculates the modulo-two sum of the x_l for which $a_l = 1$.

Create a quantum algorithm that can find the bit-string a_l from evaluating O_f for four bits. The solution circuits below contain an (unknown) oracle which you should try to reconstruct.

Show that a quantum algorithm requires less evaluations of f than a classical algorithm to solve this problem and explain how your algorithm works.

Solution

Circuit for oracle 1

Note, oracle 1 (O_1) is across all four qubits, but does not act on the first qubit, so in quirk only spans 3 qubits.

Circuit for oracle 2

A classical algorithm could reconstruct the bit-string a by calling f for each state with only a single bit set, as $f(0\dots 010\dots 0) = a_l$ for the 1 at position l . So we need n calls to f to reconstruct a .

The sign oracle for f is $O_f|x\rangle = (-1)^{\sum_l x_l a_l} |x\rangle$ (the modulo-two does not actually matter in the sum to find the sign).

As Deutsch-Josza, the algorithm applies a stack of Hadamard operations twice. The first application mapped all states to $|+\rangle$. The second application maps the states which picked up a phase from O_f from $|-\rangle$ to $|1\rangle$, while the qubits that did not pick up a phase from O_f are mapped from $|+\rangle$ to $|0\rangle$. Hence, the algorithm perfectly reconstructs a with a single call to f .

[]: