Algoritmo de Grover: Ket vs Qiskit

Gabriel Carneiro

May 26, 2022

Problema

Encontrar determinado elemento em uma lista desordenada de tamanho 2^n.

$$f(x) = \begin{cases} 0, x \neq x_0 \\ 1, x = x_0 \end{cases}$$

Cada elemento da lista será codificado em um estado da base computacional.

Algoritmo

1. $H^{\otimes n} \left|0\right\rangle^{\otimes n}$
2. Aplicar operador de Grover G k vezes:
   2.1 Aplicar oráculo;
   2.2 Aplicar a porta de Hadamard em todos os qubits;
   2.3 Aplicar o operador $2\left|0\right\rangle\left\langle 0\right| - I$;
   2.4 Aplicar a porta de Hadamard em todos os qubits.

Notação Auxiliar

$\mathbb{B}_n$: conjunto de todas as palavras de n bits.

$$N = 2^n \begin{cases} n : \text{número de qubits.} \\ N : \text{número de itens.} \end{cases}$$

M : número de itens desejados.

$$|\alpha\rangle := \sum_{\substack{x \in \mathbb{B}_n \\ f(x)=0}} \frac{|x\rangle}{\sqrt{N-M}}$$

$$|\beta\rangle := \sum_{\substack{x \in \mathbb{B}_n \\ f(x)=1}} \frac{|x\rangle}{\sqrt{M}} : \text{itens desejados.}$$

$$S := \mathrm{span}_{\mathbb{R}}\{|\alpha\rangle, |\beta\rangle\}.$$

Primeira Aplicação de G

$$\begin{aligned}
|\psi_0\rangle &= |+\rangle^{\otimes n} \\
&= \sum_{x \in \mathbb{B}_n} \frac{|x\rangle}{\sqrt{N}} \\
&= \sum_{\substack{x \in \mathbb{B}_n \\ x \neq x_0}} \frac{|x\rangle}{\sqrt{N}} + \frac{|x_0\rangle}{\sqrt{N}} \\
&= \frac{\sqrt{N-1}}{\sqrt{N}} \sum_{\substack{x \in \mathbb{B}_n \\ x \neq x_0}} \frac{|x\rangle}{\sqrt{N-1}} + \frac{|x_0\rangle}{\sqrt{N}} \\
&= \frac{\sqrt{N-1}}{\sqrt{N}} |\alpha\rangle + \frac{1}{\sqrt{N}} |\beta\rangle
\end{aligned}$$

Primeira Aplicação de G: Oráculo

$$
\begin{aligned}
|\psi_1\rangle &= O_F |\psi_0\rangle \\
&= \frac{\sqrt{N-1}}{\sqrt{N}} |\alpha\rangle - \frac{1}{\sqrt{N}} |\beta\rangle
\end{aligned}
$$

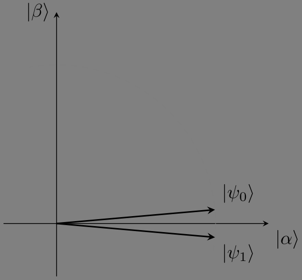


Figure: Oráculo equivale a uma reflexão em relação ao eixo $|\alpha\rangle$.

Primeira Aplicação de G: Difusor

$$
\begin{aligned}
2\,|0\rangle\,\langle 0| - I &= 2\cdot\begin{bmatrix}1\\0\\\vdots\\0\end{bmatrix}_{n\times 1}\begin{bmatrix}1 & 0 & \cdots & 0\end{bmatrix}_{1\times n} - \begin{bmatrix}1 & 0 & \cdots & 0\\0 & 1 & \cdots & 0\\\vdots & \vdots & \ddots & 0\\0 & 0 & \cdots & 1\end{bmatrix}_{n\times n}\\
&= \begin{bmatrix}2 & 0 & \cdots & 0\\0 & 0 & \cdots & 0\\\vdots & \vdots & \ddots & 0\\0 & 0 & \cdots & 0\end{bmatrix}_{n\times n} - \begin{bmatrix}1 & 0 & \cdots & 0\\0 & 1 & \cdots & 0\\\vdots & \vdots & \ddots & 0\\0 & 0 & \cdots & 1\end{bmatrix}_{n\times n}\\
&= \begin{bmatrix}1 & 0 & \cdots & 0\\0 & -1 & \cdots & 0\\\vdots & \vdots & \ddots & 0\\0 & 0 & \cdots & -1\end{bmatrix}_{n\times n}\\
&= |0\ldots0\rangle\,\langle 0\ldots0| - |0\ldots1\rangle\,\langle 0\ldots1| - \cdots - |1\ldots1\rangle\,\langle 1\ldots1|
\end{aligned}
$$

Primeira Aplicação de G: Difusor

$$
\begin{aligned}
2\left|0\right\rangle\left\langle 0\right| - I &= 2 \cdot \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}_{n\times 1} \begin{bmatrix} 1 & 0 & \cdots & 0 \end{bmatrix}_{1\times n} - \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & 0 \\ 0 & 0 & \cdots & 1 \end{bmatrix}_{n\times n} \\
&= \begin{bmatrix} 2 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & 0 \\ 0 & 0 & \cdots & 0 \end{bmatrix}_{n\times n} - \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & 0 \\ 0 & 0 & \cdots & 1 \end{bmatrix}_{n\times n} \\
&= \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & -1 & \cdots & 0 \\ \vdots & \vdots & \ddots & 0 \\ 0 & 0 & \cdots & -1 \end{bmatrix}_{n\times n} \\
&= \left|0\ldots 0\right\rangle\left\langle 0\ldots 0\right| - \left|0\ldots 1\right\rangle\left\langle 0\ldots 1\right| - \cdots - \left|1\ldots 1\right\rangle\left\langle 1\ldots 1\right| \\
&= -\left|0\ldots 0\right\rangle\left\langle 0\ldots 0\right| + \left|0\ldots 1\right\rangle\left\langle 0\ldots 1\right| + \cdots + \left|1\ldots 1\right\rangle\left\langle 1\ldots 1\right|
\end{aligned}
$$

Primeira Aplicação de G: Difusor

$$
\begin{aligned}
|\psi_2\rangle &= (2\,|\psi_0\rangle\langle\psi_0| - I)\,|\psi_1\rangle \\
&= 2\,\langle\psi_0 \mid \psi_1\rangle\,|\psi_0\rangle - |\psi_1\rangle
\end{aligned}
$$

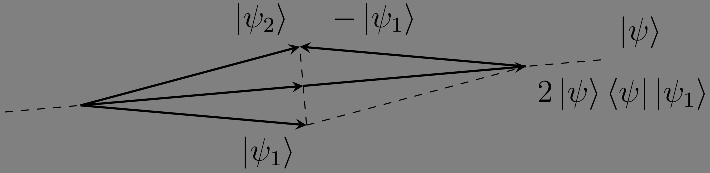


Figure: Difusor equivale a uma reflexão em relação $|\psi\rangle$.

Primeira Aplicação de G

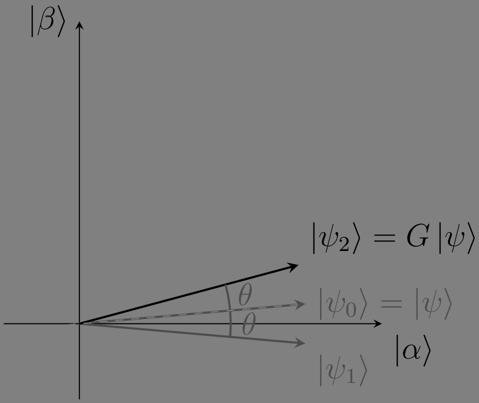


Figure: Aplicar G equivale à rotação de θ no sentido anti-horário.

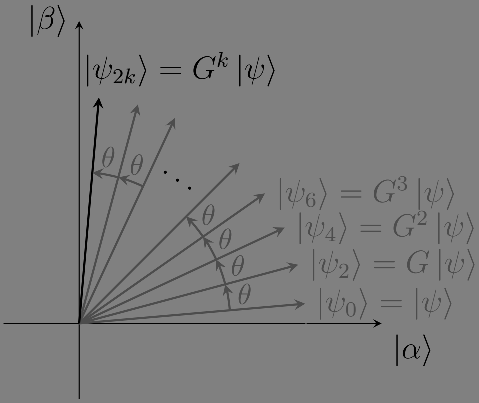


Figure: Aplicações sucessivas de G.

Acerto, k e θ

$$
\begin{aligned}
k &= \frac{\pi}{4}\sqrt{\frac{N}{M}} \\
\theta &= \arccos\left(\frac{N-2}{N}\right) \\
P_a &= \frac{N-1}{N}
\end{aligned}
$$

Ket vs Qiskit

Ket	Qiskit

Ket vs Qiskit

Ket	Qiskit
$\lvert q_0 \ldots q_n \rangle$	$\lvert q_n \ldots q_0 \rangle$

Ket vs Qiskit

Ket	Qiskit
$\lvert q_0 \dots q_n \rangle$	$\lvert q_n \dots q_0 \rangle$
`H(qubits)`	`qc.h(qubits)`

Ket vs Qiskit

Ket	Qiskit
$\lvert q_0 \dots q_n \rangle$	$\lvert q_n \dots q_0 \rangle$
`H(qubits)`	`qc.h(qubits)`
`ctrl()`	`qc.mcx()`

Ket vs Qiskit: Código

```python
def phase_oracle(qubits: quant, state: int) -> None:
    ctrl(qubits, Z, qubits[-1], on_state=state)

    return None
```

Ket vs Qiskit: Código

```
def phase_oracle(qc: QuantumCircuit, state: int) -> None:
    state: str = bin(state)[2:]
    state = "0" * (qc.num_qubits - len(state)) + state
    flip_qubits: List[Qubit] = []
    state = state[::-1]
    for i in range(len(state)):
        if (state[i] == "0"):
            flip_qubits.append(qc.qubits[i])

    if flip_qubits:
        qc.x(flip_qubits)
    qc.h(qc.qubits[-1])
    qc.mcx(qc.qubits[:-1], qc.qubits[-1])
    qc.h(qc.qubits[-1])
    if flip_qubits:
        qc.x(flip_qubits)

    return None
```

Ket vs Qiskit: Código

```python
def grover_diffuser(qubits: quant):
    H(qubits)

    ctrl(qubits, Z, qubits[-1], on_state=0)

    H(qubits)

    return None
```

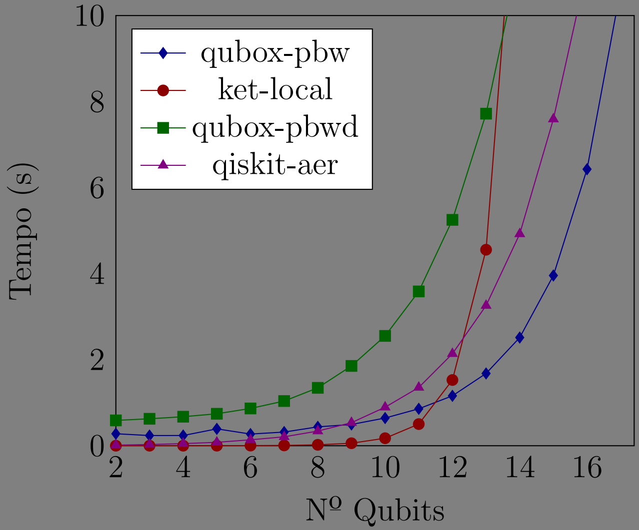
Ket vs Qiskit: Código

```python
def grover_diffuser(qc: QuantumCircuit) -> None:
    qc.h(qc.qubits)
    qc.x(qc.qubits)

    # Make a multi controlled z gate
    qc.h(qc.num_qubits - 1)
    qc.mcx(qc.qubits[:-1], qc.num_qubits - 1)
    qc.h(qc.num_qubits - 1)

    qc.x(qc.qubits)
    qc.h(qc.qubits)

    return None
```

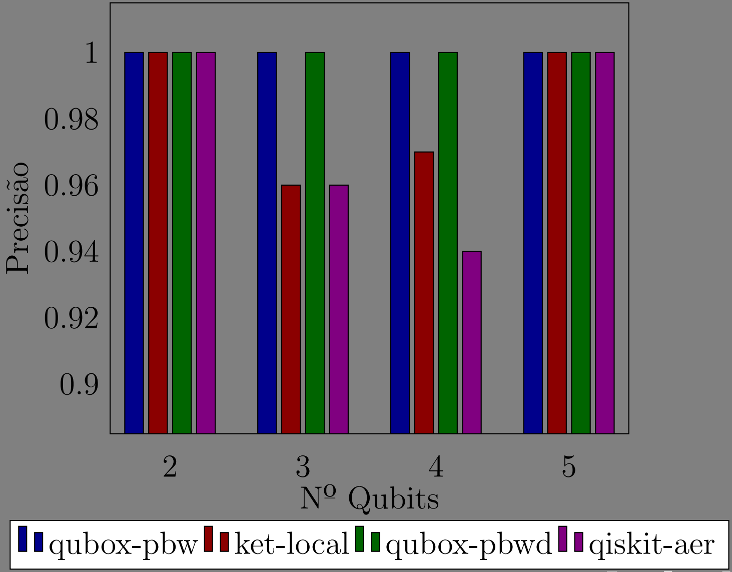
Parâmetros dos Testes

- 100 replicações.
- n qubits, $n \in [2, 20)$.
- 1 estado aleatório marcado.

Tempo de Execução

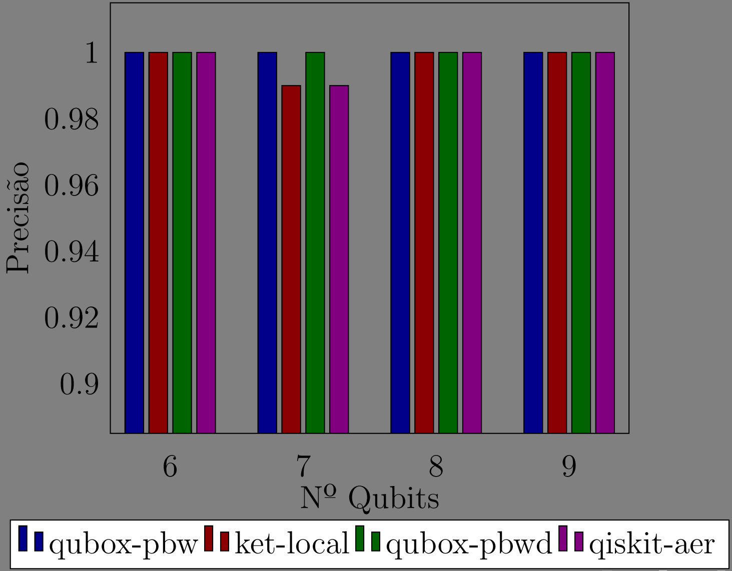


Precisão

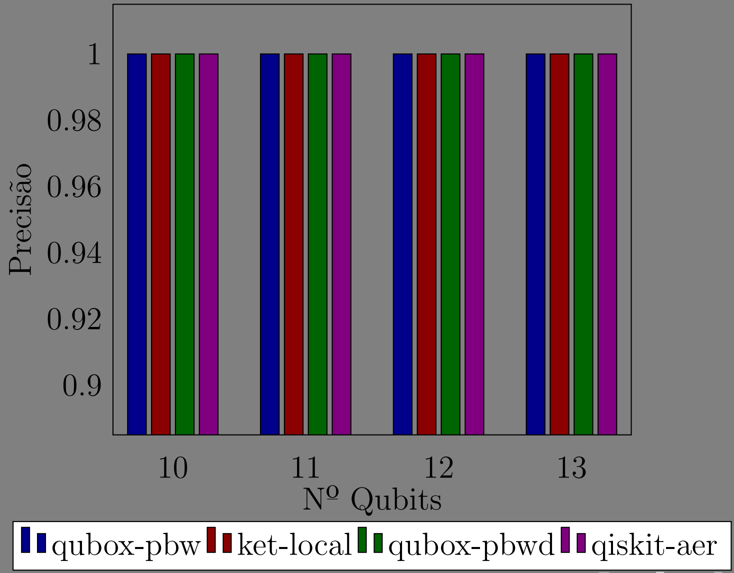


1
0.98
0.96
0.94
0.92
0.9
Precisão
2
3
4
5
Nº Qubits
qubox-pbw
ket-local
qubox-pbwd
qiskit-aer

Precisão

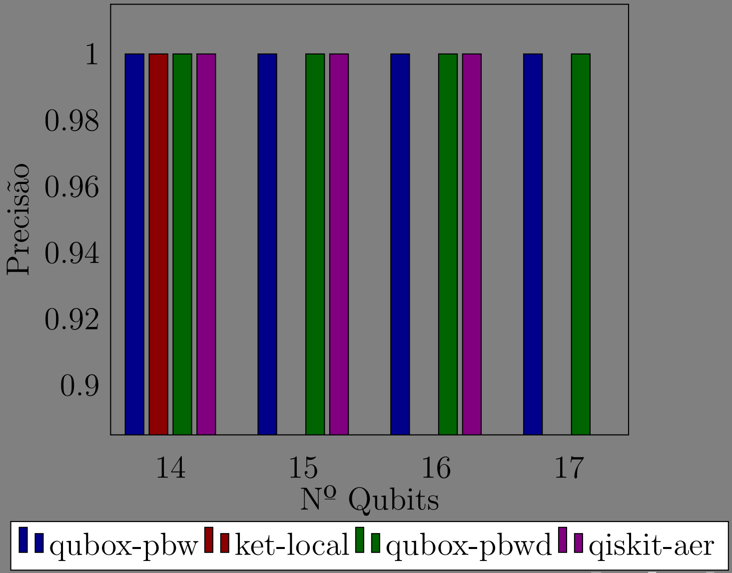

Precisão
1
0.98
0.96
0.94
0.92
0.9
6
7
8
9
Nº Qubits
qubox-pbw
ket-local
qubox-pbwd
qiskit-aer

Precisão



1
0.98
0.96
0.94
0.92
0.9
Precisão
10
11
12
13
Nº Qubits
qubox-pbw
ket-local
qubox-pbwd
qiskit-aer

Precisão



1
0.98
0.96
0.94
0.92
0.9
Precisão
14
15
16
17
Nº Qubits
qubox-pbw
ket-local
qubox-pbwd
qiskit-aer

Qiskit: Exemplo