Introduction to quantum algorithms

Arjan Cornelissen

IRIF, Paris, France

April 27th, 2023

INSTITUT
DE RECHERCHE
EN INFORMATIQUE
FONDAMENTALE

Introduction to quantum algorithms

Non-permanent's seminar?

Arjan Cornelissen

IRIF, Paris, France

April 27th, 2023

INSTITUT
DE RECHERCHE
EN INFORMATIQUE
FONDAMENTALE

Introduction to quantum algorithms

Non-permanent's seminar? – Doc-postdoc seminar?

Arjan Cornelissen

IRIF, Paris, France

April 27th, 2023

Introduction to quantum algorithms

Non-permanent's seminar? – Doc-postdoc seminar? – Junior seminar?

Arjan Cornelissen

IRIF, Paris, France

April 27th, 2023

Introduction to quantum algorithms

Non-permanent's seminar? – Doc-postdoc seminar? – Junior seminar? – P $\neq$ NP seminar?

Arjan Cornelissen

IRIF, Paris, France

April 27th, 2023

Introduction to quantum algorithms

Non-permanent's seminar? – Doc-postdoc seminar? – Junior seminar? – P $\neq$ NP seminar? – ???

Arjan Cornelissen

IRIF, Paris, France

April 27th, 2023

INSTITUT DE RECHERCHE EN INFORMATIQUE FONDAMENTALE

Introduction to quantum algorithms

Non-permanent's seminar? – Doc-postdoc seminar? – Junior seminar? – P $\neq$ NP seminar? – ???

Arjan Cornelissen

IRIF, Paris, France

April 27th, 2023 – (Koningsdag / King's day)

INSTITUT
DE RECHERCHE
EN INFORMATIQUE
FONDAMENTALE

Koningsdag / King's day

Koningsdag / King's day

Koningsdag / King's day

Koningsdag / King's day

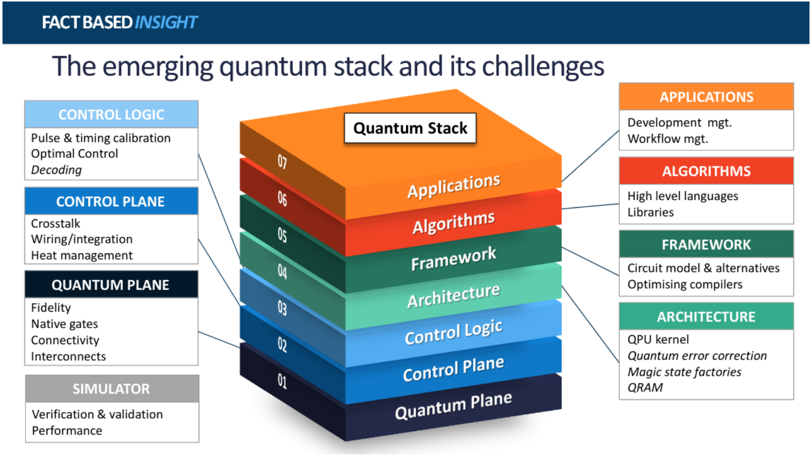
Koningsdag / King's day

Overview

Plan for today:

1. Quantum algorithms
2. Grover's algorithm
3. Application: collision finding

Quantum algorithms



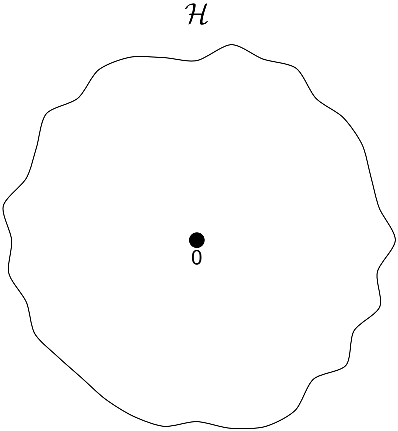
Quantum algorithms

Ingredients:

Quantum algorithms

Ingredients:

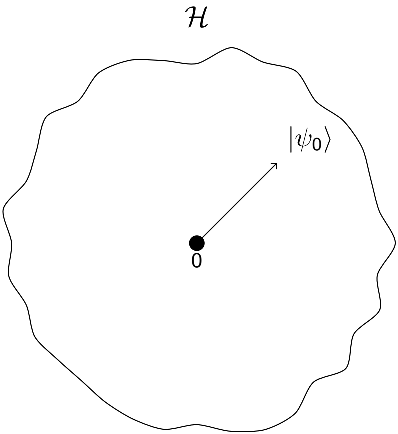
1. *State space* – $\mathcal{H}$.



Quantum algorithms

Ingredients:

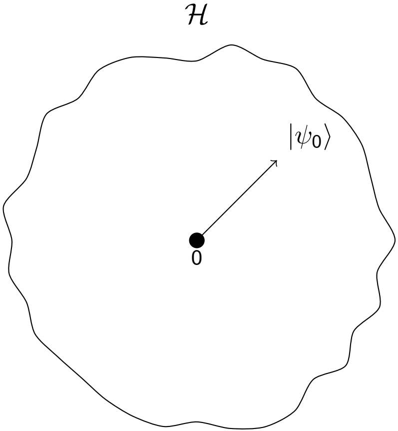
1. *State space* – $\mathcal{H}$.
2. *Initial state* – $|\psi_0\rangle \in \mathcal{H}$, $\||\psi_0\rangle\| = 1$.



Quantum algorithms

Ingredients:

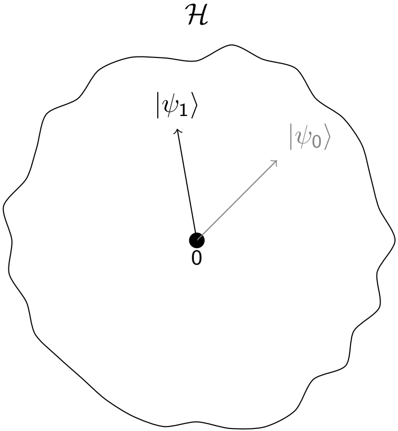
1. *State space* – $\mathcal{H}$.
2. *Initial state* – $|\psi_0\rangle \in \mathcal{H}$, $\||\psi_0\rangle\| = 1$.
3. *Operations* – unitary operators $U_1, \dots, U_T \in \mathcal{U}(\mathcal{H})$.
   $|\psi_0\rangle \overset{U_1}{\mapsto} |\psi_1\rangle \overset{U_2}{\mapsto} |\psi_2\rangle \overset{U_3}{\mapsto} \cdots \overset{U_T}{\mapsto} |\psi_T\rangle$.



Quantum algorithms

Ingredients:

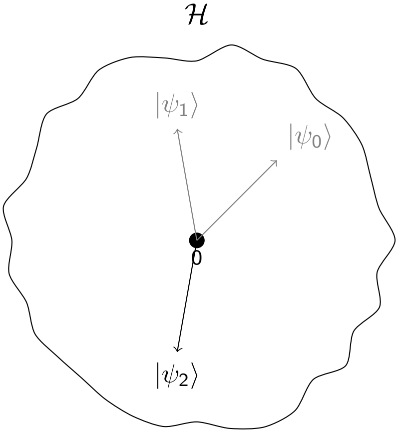
1. *State space* – $\mathcal{H}$.
2. *Initial state* – $|\psi_0\rangle \in \mathcal{H}$, $\||\psi_0\rangle\| = 1$.
3. *Operations* – unitary operators $U_1, \dots, U_T \in \mathcal{U}(\mathcal{H})$.
   $|\psi_0\rangle \overset{U_1}{\mapsto} |\psi_1\rangle \overset{U_2}{\mapsto} |\psi_2\rangle \overset{U_3}{\mapsto} \cdots \overset{U_T}{\mapsto} |\psi_T\rangle.$



Quantum algorithms

Ingredients:

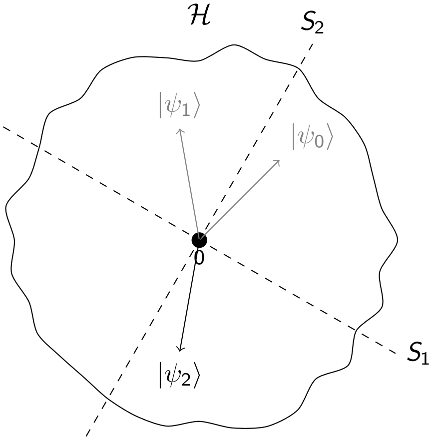
1. *State space* – $\mathcal{H}$.
2. *Initial state* – $|\psi_0\rangle \in \mathcal{H}$, $\||\psi_0\rangle\| = 1$.
3. *Operations* – unitary operators $U_1, \dots, U_T \in \mathcal{U}(\mathcal{H})$.
   $|\psi_0\rangle \overset{U_1}{\mapsto} |\psi_1\rangle \overset{U_2}{\mapsto} |\psi_2\rangle \overset{U_3}{\mapsto} \cdots \overset{U_T}{\mapsto} |\psi_T\rangle$.



Quantum algorithms

Ingredients:

1. *State space* – $\mathcal{H}$.
2. *Initial state* – $|\psi_0\rangle \in \mathcal{H}$, $\||\psi_0\rangle\| = 1$.
3. *Operations* – unitary operators $U_1, \dots, U_T \in \mathcal{U}(\mathcal{H})$.
   $|\psi_0\rangle \overset{U_1}{\mapsto} |\psi_1\rangle \overset{U_2}{\mapsto} |\psi_2\rangle \overset{U_3}{\mapsto} \cdots \overset{U_T}{\mapsto} |\psi_T\rangle$.
4. *Measurement* – $S_1, \dots, S_m \subseteq \mathcal{H}$ s.t. $\mathcal{H} = \bigoplus_{j=1}^m S_j$.

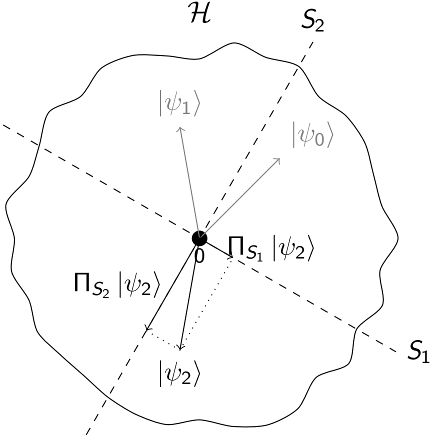


Ingredients:

1. *State space* – $\mathcal{H}$.
2. *Initial state* – $|\psi_0\rangle \in \mathcal{H}$, $\||\psi_0\rangle\| = 1$.
3. *Operations* – unitary operators $U_1, \dots, U_T \in \mathcal{U}(\mathcal{H})$.
   $|\psi_0\rangle \overset{U_1}{\mapsto} |\psi_1\rangle \overset{U_2}{\mapsto} |\psi_2\rangle \overset{U_3}{\mapsto} \cdots \overset{U_T}{\mapsto} |\psi_T\rangle$.
4. *Measurement* – $S_1, \dots, S_m \subseteq \mathcal{H}$ s.t. $\mathcal{H} = \bigoplus_{j=1}^m S_j$.

Result: Probability of outcome $j \in \{1, \dots, m\}$:

$$\mathbb{P}[j] = \left\|\Pi_{S_j} |\psi_T\rangle\right\|^2.$$



Unstructured search

Unstructured search

Problem: (Unstructured search)

1. *Input:* $x \in \{0,1\}^n$.

Unstructured search

Problem: (Unstructured search)

1. *Input:* $x \in \{0,1\}^n$.
2. *Output:*
   - If $x \neq 0^n$, output $j \in \{1,\dots,n\}$ s.t. $x_j = 1$.
   - If $x = 0^n$, output "NO SOLUTION".

Unstructured search

Problem: (Unstructured search)

1. *Input:* $x \in \{0,1\}^n$.
2. *Output:*
   - If $x \neq 0^n$, output $j \in \{1, \ldots, n\}$ s.t. $x_j = 1$.
   - If $x = 0^n$, output "NO SOLUTION".

Classical access model: $j \mapsto x_j$.

Unstructured search

Problem: (Unstructured search)

1. *Input:* $x \in \{0,1\}^n$.
2. *Output:*
   - If $x \neq 0^n$, output $j \in \{1, \ldots, n\}$ s.t. $x_j = 1$.
   - If $x = 0^n$, output "NO SOLUTION".

Classical access model: $j \mapsto x_j$.

Classical algorithm:

1. Query all bits, stop when you find a 1.

$$\underbrace{\cdot\ \cdot\ \cdot\ \cdot\ \cdot\ \cdot\ \cdot}_{n \text{ bits}}$$

Unstructured search

Problem: (Unstructured search)

1. *Input:* $x \in \{0,1\}^n$.
2. *Output:*
   - If $x \neq 0^n$, output $j \in \{1,\dots,n\}$ s.t. $x_j = 1$.
   - If $x = 0^n$, output "NO SOLUTION".

Classical access model: $j \mapsto x_j$.

Classical algorithm:

1. Query all bits, stop when you find a 1.

$$\underbrace{0\cdots\cdots}_{n \text{ bits}}$$

Unstructured search

Problem: (Unstructured search)

1. *Input:* $x \in \{0,1\}^n$.
2. *Output:*
   - If $x \neq 0^n$, output $j \in \{1,\dots,n\}$ s.t. $x_j = 1$.
   - If $x = 0^n$, output "NO SOLUTION".

Classical access model: $j \mapsto x_j$.

Classical algorithm:

1. Query all bits, stop when you find a 1.

$$\underbrace{00\cdots\cdots}_{n \text{ bits}}$$

Unstructured search

Problem: (Unstructured search)

1. *Input:* $x \in \{0,1\}^n$.
2. *Output:*
   - If $x \neq 0^n$, output $j \in \{1, \ldots, n\}$ s.t. $x_j = 1$.
   - If $x = 0^n$, output "NO SOLUTION".

Classical access model: $j \mapsto x_j$.

Classical algorithm:

1. Query all bits, stop when you find a 1.

$$\underbrace{001\cdots\cdot}_{n \text{ bits}}$$

Unstructured search

Problem: (Unstructured search)

1. *Input:* $x \in \{0,1\}^n$.
2. *Output:*
   - If $x \neq 0^n$, output $j \in \{1, \dots, n\}$ s.t. $x_j = 1$.
   - If $x = 0^n$, output "NO SOLUTION".

Classical access model: $j \mapsto x_j$.

Classical algorithm:

1. Query all bits, stop when you find a 1.

$$\underbrace{001\cdots\cdot}_{n \text{ bits}}$$

2. Worst case: n queries.

Unstructured search

Problem: (Unstructured search)

1. *Input:* $x \in \{0,1\}^n$.
2. *Output:*
   - If $x \neq 0^n$, output $j \in \{1,\ldots,n\}$ s.t. $x_j = 1$.
   - If $x = 0^n$, output "NO SOLUTION".

Classical access model: $j \mapsto x_j$.

Classical algorithm:

1. Query all bits, stop when you find a 1.

$$\underbrace{001\cdots\cdot}_{n \text{ bits}}$$

2. Worst case: n queries.

Quantum access model:

1. *Oracle:* $O_x : |j\rangle \mapsto (-1)^{x_j}|j\rangle$.

$$O_x = \begin{bmatrix} (-1)^{x_1} & 0 & \cdots & 0 \\ 0 & (-1)^{x_2} & & 0 \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \cdots & (-1)^{x_n} \end{bmatrix}.$$

Unstructured search

Problem: (Unstructured search)

1. *Input:* $x \in \{0,1\}^n$.
2. *Output:*
   - If $x \neq 0^n$, output $j \in \{1,\dots,n\}$ s.t. $x_j = 1$.
   - If $x = 0^n$, output "NO SOLUTION".

Classical access model: $j \mapsto x_j$.

Classical algorithm:

1. Query all bits, stop when you find a 1.

$$\underbrace{001\cdots}_{n \text{ bits}}$$

2. Worst case: n queries.

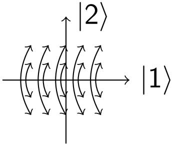
Quantum access model:

1. *Oracle:* $O_x : |j\rangle \mapsto (-1)^{x_j}|j\rangle$.

$$O_x = \begin{bmatrix} (-1)^{x_1} & 0 & \cdots & 0 \\ 0 & (-1)^{x_2} & & 0 \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \cdots & (-1)^{x_n} \end{bmatrix}.$$

2. *Example:* $x = 01 \in \{0,1\}^2$:

$$O_x = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$$



Unstructured search

Problem: (Unstructured search)

1. *Input:* $x \in \{0,1\}^n$.
2. *Output:*
   - If $x \neq 0^n$, output $j \in \{1, \dots, n\}$ s.t. $x_j = 1$.
   - If $x = 0^n$, output "NO SOLUTION".

Classical access model: $j \mapsto x_j$.

Classical algorithm:

1. Query all bits, stop when you find a 1.

$$\underbrace{001\cdots\cdot}_{n \text{ bits}}$$

2. Worst case: n queries.

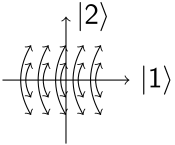
Quantum access model:

1. *Oracle:* $O_x : |j\rangle \mapsto (-1)^{x_j} |j\rangle$.

$$O_x = \begin{bmatrix} (-1)^{x_1} & 0 & \cdots & 0 \\ 0 & (-1)^{x_2} & & 0 \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \cdots & (-1)^{x_n} \end{bmatrix}.$$

2. *Example:* $x = 01 \in \{0,1\}^2$:

$$O_x = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$$



Quantum algorithm: of the form

$$|\psi_0\rangle \overset{O_x}{\mapsto} |\psi_1\rangle \overset{U_2}{\mapsto} |\psi_2\rangle \overset{O_x}{\mapsto} |\psi_3\rangle \overset{U_4}{\mapsto} \cdots \overset{U_T}{\mapsto} |\psi_T\rangle.$$

Grover's algorithm [Gro'96] (1/2)

Grover's algorithm [Gro'96] (1/2)

1. *Assumption:* $|x| = 1$.

Grover's algorithm [Gro'96] (1/2)

1. *Assumption:* $|x| = 1$.
2. *Example:* when $x = 1000 \in \{0,1\}^4$:

$$|\psi_0\rangle := \frac{1}{\sqrt{4}} \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

Grover's algorithm [Gro'96] (1/2)

1. *Assumption:* $|x| = 1$.
2. *Example:* when $x = 1000 \in \{0,1\}^4$:

$$|\psi_0\rangle := \frac{1}{\sqrt{4}} \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \qquad O_x = \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

Grover's algorithm [Gro'96] (1/2)

1. *Assumption:* $|x| = 1$.
2. *Example:* when $x = 1000 \in \{0,1\}^4$:

$$|\psi_0\rangle := \frac{1}{\sqrt{4}} \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} = \frac{1}{\sqrt{4}} \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} + \frac{1}{\sqrt{4}} \begin{bmatrix} 0 \\ 1 \\ 1 \\ 1 \end{bmatrix} \qquad O_x = \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

Grover's algorithm [Gro'96] (1/2)

1. *Assumption:* $|x| = 1$.
2. *Example:* when $x = 1000 \in \{0,1\}^4$:

$$|\psi_0\rangle := \frac{1}{\sqrt{4}} \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} = \frac{1}{\sqrt{4}} \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} + \frac{1}{\sqrt{4}} \begin{bmatrix} 0 \\ 1 \\ 1 \\ 1 \end{bmatrix} = \frac{1}{\sqrt{4}} \underbrace{\begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}}_{|s\rangle} + \sqrt{\frac{3}{4}} \cdot \underbrace{\frac{1}{\sqrt{3}} \begin{bmatrix} 0 \\ 1 \\ 1 \\ 1 \end{bmatrix}}_{|s^\perp\rangle}, \quad O_x = \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

Grover's algorithm [Gro'96] (1/2)

1. *Assumption:* $|x| = 1$.
2. *Example:* when $x = 1000 \in \{0,1\}^4$:

$$|\psi_0\rangle := \frac{1}{\sqrt{4}}\begin{bmatrix}1\\1\\1\\1\end{bmatrix} = \frac{1}{\sqrt{4}}\begin{bmatrix}1\\0\\0\\0\end{bmatrix} + \frac{1}{\sqrt{4}}\begin{bmatrix}0\\1\\1\\1\end{bmatrix} = \frac{1}{\sqrt{4}}\underbrace{\begin{bmatrix}1\\0\\0\\0\end{bmatrix}}_{|s\rangle} + \sqrt{\frac{3}{4}} \cdot \underbrace{\frac{1}{\sqrt{3}}\begin{bmatrix}0\\1\\1\\1\end{bmatrix}}_{|s^\perp\rangle}, \quad O_x = \begin{bmatrix}-1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1\end{bmatrix}.$$

3. *In general:* We can write $x_s = 1$. Then: $O_x : |j\rangle \mapsto (-1)^{x_j}|j\rangle$

$$|\psi_0\rangle := \frac{1}{\sqrt{n}}\sum_{j=1}^{n}|j\rangle$$

Grover's algorithm [Gro'96] (1/2)

1. *Assumption:* $|x| = 1$.
2. *Example:* when $x = 1000 \in \{0,1\}^4$:

$$|\psi_0\rangle := \frac{1}{\sqrt{4}}\begin{bmatrix}1\\1\\1\\1\end{bmatrix} = \frac{1}{\sqrt{4}}\begin{bmatrix}1\\0\\0\\0\end{bmatrix} + \frac{1}{\sqrt{4}}\begin{bmatrix}0\\1\\1\\1\end{bmatrix} = \frac{1}{\sqrt{4}}\underbrace{\begin{bmatrix}1\\0\\0\\0\end{bmatrix}}_{|s\rangle} + \sqrt{\frac{3}{4}}\cdot\underbrace{\frac{1}{\sqrt{3}}\begin{bmatrix}0\\1\\1\\1\end{bmatrix}}_{|s^\perp\rangle}, \quad O_x = \begin{bmatrix}-1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1\end{bmatrix}.$$

3. *In general:* We can write $x_s = 1$. Then:

$$O_x : |j\rangle \mapsto (-1)^{x_j}|j\rangle$$

$$|\psi_0\rangle := \frac{1}{\sqrt{n}}\sum_{j=1}^{n}|j\rangle = \frac{1}{\sqrt{n}}|s\rangle + \frac{1}{\sqrt{n}}\sum_{\substack{j=1\\j\neq s}}^{n}|j\rangle$$

Grover's algorithm [Gro'96] (1/2)

1. *Assumption:* $|x| = 1$.
2. *Example:* when $x = 1000 \in \{0,1\}^4$:

$$|\psi_0\rangle := \frac{1}{\sqrt{4}}\begin{bmatrix}1\\1\\1\\1\end{bmatrix} = \frac{1}{\sqrt{4}}\begin{bmatrix}1\\0\\0\\0\end{bmatrix} + \frac{1}{\sqrt{4}}\begin{bmatrix}0\\1\\1\\1\end{bmatrix} = \frac{1}{\sqrt{4}}\underbrace{\begin{bmatrix}1\\0\\0\\0\end{bmatrix}}_{|s\rangle} + \sqrt{\frac{3}{4}}\cdot\underbrace{\frac{1}{\sqrt{3}}\begin{bmatrix}0\\1\\1\\1\end{bmatrix}}_{|s^\perp\rangle}, \quad O_x = \begin{bmatrix}-1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1\end{bmatrix}.$$

3. *In general:* We can write $x_s = 1$. Then: $\quad O_x : |j\rangle \mapsto (-1)^{x_j}|j\rangle$

$$|\psi_0\rangle := \frac{1}{\sqrt{n}}\sum_{j=1}^{n}|j\rangle = \frac{1}{\sqrt{n}}|s\rangle + \frac{1}{\sqrt{n}}\sum_{\substack{j=1\\ j\neq s}}^{n}|j\rangle = \frac{1}{\sqrt{n}}|s\rangle + \sqrt{\frac{n-1}{n}}\cdot\underbrace{\frac{1}{\sqrt{n-1}}\sum_{\substack{j=1\\ j\neq s}}^{n}|j\rangle}_{|s^\perp\rangle}.$$

Grover's algorithm [Gro'96] (1/2)

1. *Assumption:* $|x| = 1$.
2. *Example:* when $x = 1000 \in \{0,1\}^4$:

$$|\psi_0\rangle := \frac{1}{\sqrt{4}}\begin{bmatrix}1\\1\\1\\1\end{bmatrix} = \frac{1}{\sqrt{4}}\begin{bmatrix}1\\0\\0\\0\end{bmatrix} + \frac{1}{\sqrt{4}}\begin{bmatrix}0\\1\\1\\1\end{bmatrix} = \frac{1}{\sqrt{4}}\underbrace{\begin{bmatrix}1\\0\\0\\0\end{bmatrix}}_{|s\rangle} + \sqrt{\frac{3}{4}}\cdot\underbrace{\frac{1}{\sqrt{3}}\begin{bmatrix}0\\1\\1\\1\end{bmatrix}}_{|s^\perp\rangle}, \quad O_x = \begin{bmatrix}-1 & 0 & 0 & 0\\0 & 1 & 0 & 0\\0 & 0 & 1 & 0\\0 & 0 & 0 & 1\end{bmatrix}.$$

3. *In general:* We can write $x_s = 1$. Then: $O_x : |j\rangle \mapsto (-1)^{x_j}|j\rangle$

$$|\psi_0\rangle := \frac{1}{\sqrt{n}}\sum_{j=1}^n |j\rangle = \frac{1}{\sqrt{n}}|s\rangle + \frac{1}{\sqrt{n}}\sum_{\substack{j=1\\j\neq s}}^n |j\rangle = \frac{1}{\sqrt{n}}|s\rangle + \sqrt{\frac{n-1}{n}}\cdot\underbrace{\frac{1}{\sqrt{n-1}}\sum_{\substack{j=1\\j\neq s}}^n |j\rangle}_{|s^\perp\rangle}.$$

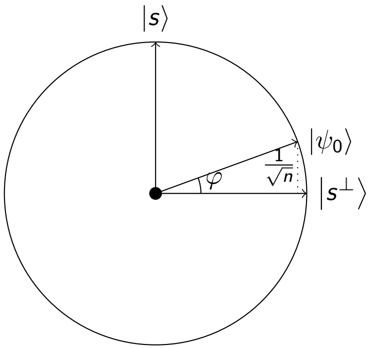
4. *Conclusion:*
   1. $|\psi_0\rangle = \frac{1}{\sqrt{n}}|s\rangle + \sqrt{1-\frac{1}{n}}|s^\perp\rangle$.
   2. $O_x|s\rangle = -|s\rangle$ and $O_x|s^\perp\rangle = |s^\perp\rangle$.

Grover's algorithm [Gro'96] (2/2)

1. *Assumption:* $|x| = 1$.
2. *Observations:*
   1. $|\psi_0\rangle = \frac{1}{\sqrt{n}}|s\rangle + \sqrt{1 - \frac{1}{n}}|s^\perp\rangle$
   2. $O_x|s\rangle = -|s\rangle$ and $O_x|s^\perp\rangle = |s^\perp\rangle$.

Grover's algorithm [Gro'96] (2/2)

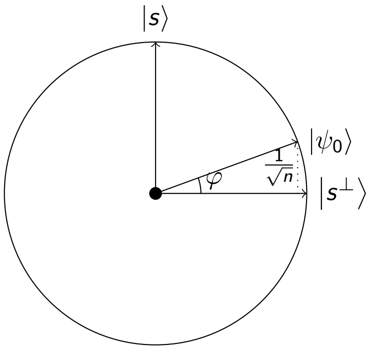
1. *Assumption:* $|x| = 1$.
2. *Observations:*
   1. $|\psi_0\rangle = \frac{1}{\sqrt{n}}|s\rangle + \sqrt{1 - \frac{1}{n}}|s^\perp\rangle$
   2. $O_x|s\rangle = -|s\rangle$ and $O_x|s^\perp\rangle = |s^\perp\rangle$.



$$\varphi = \arcsin\frac{1}{\sqrt{n}}$$

Grover's algorithm [Gro'96] (2/2)

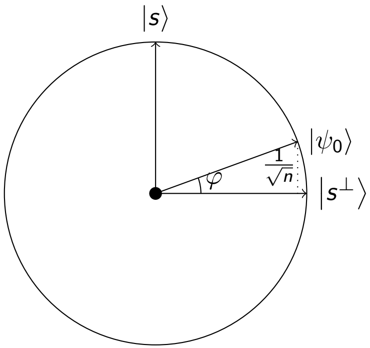
1. *Assumption:* $|x| = 1$.
2. *Observations:*
   1. $|\psi_0\rangle = \frac{1}{\sqrt{n}}|s\rangle + \sqrt{1 - \frac{1}{n}}|s^\perp\rangle$
   2. $O_x|s\rangle = -|s\rangle$ and $O_x|s^\perp\rangle = |s^\perp\rangle$.
3. *Grover's algorithm:*
   1. *State space:* $\mathcal{H} = \mathbb{C}^n$.
   2. *Initial state:* $|\psi_0\rangle = \frac{1}{\sqrt{n}}\sum_{j=1}^n |j\rangle$.



$$\varphi = \arcsin\frac{1}{\sqrt{n}}$$

Grover's algorithm [Gro'96] (2/2)

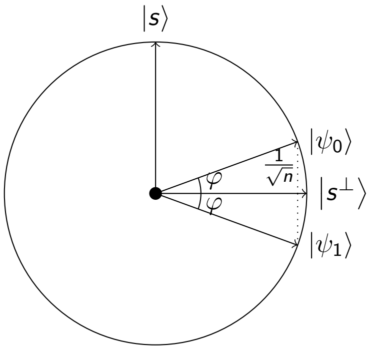
1. *Assumption:* $|x| = 1$.
2. *Observations:*
   1. $|\psi_0\rangle = \frac{1}{\sqrt{n}}|s\rangle + \sqrt{1 - \frac{1}{n}}|s^\perp\rangle$
   2. $O_x|s\rangle = -|s\rangle$ and $O_x|s^\perp\rangle = |s^\perp\rangle$.
3. *Grover's algorithm:*
   1. *State space:* $\mathcal{H} = \mathbb{C}^n$.
   2. *Initial state:* $|\psi_0\rangle = \frac{1}{\sqrt{n}}\sum_{j=1}^n |j\rangle$.
   3. *Operations:*
      1. Apply O_x.
      2. Reflect through $|\psi_0\rangle$.



$\varphi = \arcsin\frac{1}{\sqrt{n}}$

Grover's algorithm [Gro'96] (2/2)

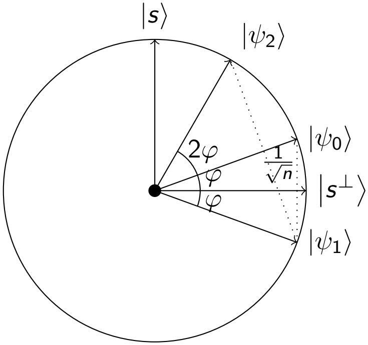
1. *Assumption:* $|x| = 1$.
2. *Observations:*
   1. $|\psi_0\rangle = \frac{1}{\sqrt{n}}|s\rangle + \sqrt{1 - \frac{1}{n}}|s^\perp\rangle$
   2. $O_x|s\rangle = -|s\rangle$ and $O_x|s^\perp\rangle = |s^\perp\rangle$.
3. *Grover's algorithm:*
   1. *State space:* $\mathcal{H} = \mathbb{C}^n$.
   2. *Initial state:* $|\psi_0\rangle = \frac{1}{\sqrt{n}}\sum_{j=1}^n |j\rangle$.
   3. *Operations:*
      1. Apply O_x.
      2. Reflect through $|\psi_0\rangle$.



$\varphi = \arcsin \frac{1}{\sqrt{n}}$

Grover's algorithm [Gro'96] (2/2)

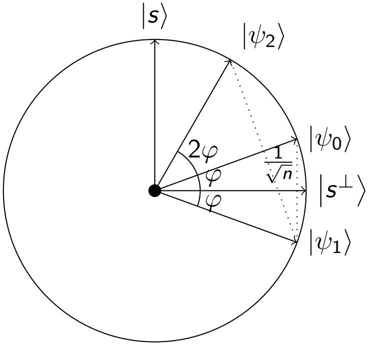
1. *Assumption:* $|x| = 1$.
2. *Observations:*
   1. $|\psi_0\rangle = \frac{1}{\sqrt{n}}|s\rangle + \sqrt{1-\frac{1}{n}}|s^\perp\rangle$
   2. $O_x|s\rangle = -|s\rangle$ and $O_x|s^\perp\rangle = |s^\perp\rangle$.
3. *Grover's algorithm:*
   1. *State space:* $\mathcal{H} = \mathbb{C}^n$.
   2. *Initial state:* $|\psi_0\rangle = \frac{1}{\sqrt{n}}\sum_{j=1}^n |j\rangle$.
   3. *Operations:*
      1. Apply O_x.
      2. Reflect through $|\psi_0\rangle$.



$\varphi = \arcsin\frac{1}{\sqrt{n}}$

Grover's algorithm [Gro'96] (2/2)

1. *Assumption:* $|x| = 1$.
2. *Observations:*
   1. $|\psi_0\rangle = \frac{1}{\sqrt{n}}|s\rangle + \sqrt{1 - \frac{1}{n}}|s^\perp\rangle$
   2. $O_x|s\rangle = -|s\rangle$ and $O_x|s^\perp\rangle = |s^\perp\rangle$.
3. *Grover's algorithm:*
   1. *State space:* $\mathcal{H} = \mathbb{C}^n$.
   2. *Initial state:* $|\psi_0\rangle = \frac{1}{\sqrt{n}}\sum_{j=1}^n |j\rangle$.
   3. *Operations:*
      1. Apply O_x.
      2. Reflect through $|\psi_0\rangle$.

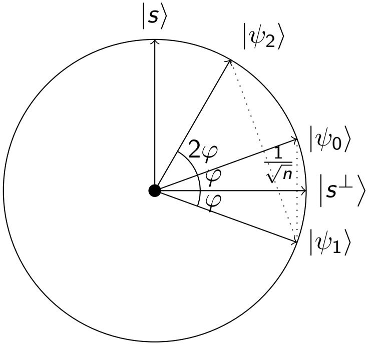


$\varphi = \arcsin \frac{1}{\sqrt{n}}$

$(2k+1)\varphi \approx \frac{\pi}{2}$.

Grover's algorithm [Gro'96] (2/2)

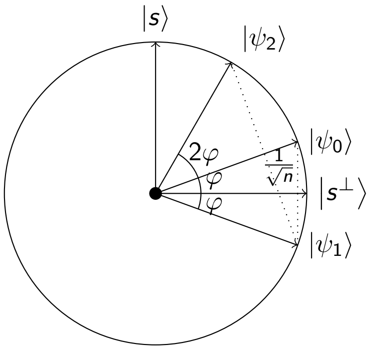
1. *Assumption:* $|x| = 1$.
2. *Observations:*
    1. $|\psi_0\rangle = \frac{1}{\sqrt{n}}|s\rangle + \sqrt{1 - \frac{1}{n}}|s^\perp\rangle$
    2. $O_x|s\rangle = -|s\rangle$ and $O_x|s^\perp\rangle = |s^\perp\rangle$.
3. *Grover's algorithm:*
    1. *State space:* $\mathcal{H} = \mathbb{C}^n$.
    2. *Initial state:* $|\psi_0\rangle = \frac{1}{\sqrt{n}}\sum_{j=1}^n |j\rangle$.
    3. *Operations:* Repeat $k = \lfloor \frac{\pi}{4\varphi} \rfloor$ times:
        1. Apply O_x.
        2. Reflect through $|\psi_0\rangle$.



$$\varphi = \arcsin \frac{1}{\sqrt{n}}$$
$$(2k+1)\varphi \approx \frac{\pi}{2}.$$

Grover's algorithm [Gro'96] (2/2)

1. *Assumption:* $|x| = 1$.
2. *Observations:*
   1. $|\psi_0\rangle = \frac{1}{\sqrt{n}}|s\rangle + \sqrt{1 - \frac{1}{n}}|s^\perp\rangle$
   2. $O_x|s\rangle = -|s\rangle$ and $O_x|s^\perp\rangle = |s^\perp\rangle$.
3. *Grover's algorithm:*
   1. *State space:* $\mathcal{H} = \mathbb{C}^n$.
   2. *Initial state:* $|\psi_0\rangle = \frac{1}{\sqrt{n}}\sum_{j=1}^n |j\rangle$.
   3. *Operations:* Repeat $k = \lfloor \frac{\pi}{4\varphi} \rfloor$ times:
      1. Apply O_x.
      2. Reflect through $|\psi_0\rangle$.
   4. *Measurement:* $S_j = \mathrm{Span}\{|j\rangle\}$.

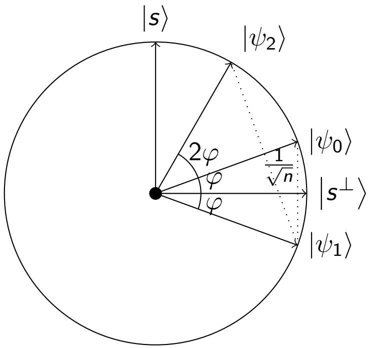


$\varphi = \arcsin \frac{1}{\sqrt{n}}$

$(2k+1)\varphi \approx \frac{\pi}{2}$.

Grover's algorithm [Gro'96] (2/2)

1. *Assumption:* $|x| = 1$.
2. *Observations:*
   1. $|\psi_0\rangle = \frac{1}{\sqrt{n}}|s\rangle + \sqrt{1 - \frac{1}{n}}|s^\perp\rangle$
   2. $O_x|s\rangle = -|s\rangle$ and $O_x|s^\perp\rangle = |s^\perp\rangle$.
3. *Grover's algorithm:*
   1. *State space:* $\mathcal{H} = \mathbb{C}^n$.
   2. *Initial state:* $|\psi_0\rangle = \frac{1}{\sqrt{n}}\sum_{j=1}^n |j\rangle$.
   3. *Operations:* Repeat $k = \lfloor \frac{\pi}{4\varphi} \rfloor$ times:
      1. Apply O_x.
      2. Reflect through $|\psi_0\rangle$.
   4. *Measurement:* $S_j = \mathrm{Span}\{|j\rangle\}$.
4. $\mathbb{P}[s] \geq 1 - 1/n$.

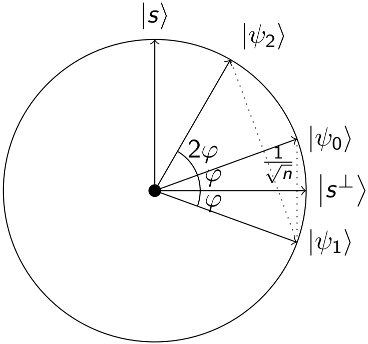


$$\varphi = \arcsin\frac{1}{\sqrt{n}}$$

$$(2k+1)\varphi \approx \frac{\pi}{2}.$$

Grover's algorithm [Gro'96] (2/2)

1. *Assumption:* $|x| = 1$.
2. *Observations:*
   1. $|\psi_0\rangle = \frac{1}{\sqrt{n}}|s\rangle + \sqrt{1 - \frac{1}{n}}|s^\perp\rangle$
   2. $O_x|s\rangle = -|s\rangle$ and $O_x|s^\perp\rangle = |s^\perp\rangle$.
3. *Grover's algorithm:*
   1. *State space:* $\mathcal{H} = \mathbb{C}^n$.
   2. *Initial state:* $|\psi_0\rangle = \frac{1}{\sqrt{n}}\sum_{j=1}^n |j\rangle$.
   3. *Operations:* Repeat $k = \lfloor \frac{\pi}{4\varphi} \rfloor$ times:
      1. Apply O_x.
      2. Reflect through $|\psi_0\rangle$.
   4. *Measurement:* $S_j = \mathrm{Span}\{|j\rangle\}$.
4. $\mathbb{P}[s] \geq 1 - 1/n$.
5. $k = O(\sqrt{n})$ queries – *Quadratic improvement!*



$$\varphi = \arcsin\frac{1}{\sqrt{n}} = \Omega\left(\frac{1}{\sqrt{n}}\right).$$
$$(2k+1)\varphi \approx \frac{\pi}{2}.$$

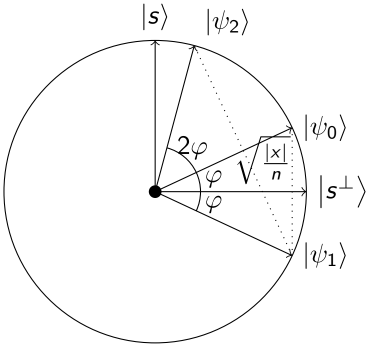
Improvements of Grover's algorithm

Improvements of Grover's algorithm

1. If $|x| > 0$:
$$|s\rangle = \frac{1}{\sqrt{|x|}} \sum_{\substack{j=1 \\ x_j=1}}^{n} |j\rangle, \quad |s^{\perp}\rangle = \frac{1}{\sqrt{n-|x|}} \sum_{\substack{j=1 \\ x_j=0}}^{n} |j\rangle.$$

Improvements of Grover's algorithm

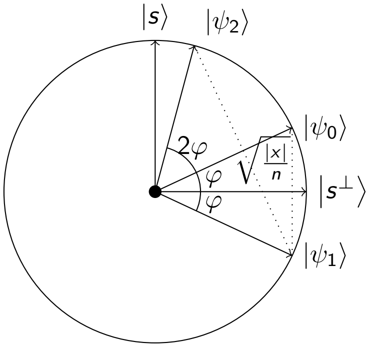
1. If $|x| > 0$:
$$|s\rangle = \frac{1}{\sqrt{|x|}} \sum_{\substack{j=1 \\ x_j=1}}^{n} |j\rangle, \quad |s^\perp\rangle = \frac{1}{\sqrt{n-|x|}} \sum_{\substack{j=1 \\ x_j=0}}^{n} |j\rangle.$$



$$\varphi = \arcsin\sqrt{\frac{|x|}{n}} = \Omega\left(\sqrt{\frac{|x|}{n}}\right).$$
$$(2k+1)\varphi \approx \frac{\pi}{2}.$$

Improvements of Grover's algorithm

1. If $|x| > 0$:
$$|s\rangle = \frac{1}{\sqrt{|x|}} \sum_{\substack{j=1 \\ x_j=1}}^{n} |j\rangle, \quad |s^\perp\rangle = \frac{1}{\sqrt{n-|x|}} \sum_{\substack{j=1 \\ x_j=0}}^{n} |j\rangle.$$
2. If we know $|x|$:
   - $k = \lfloor \frac{\pi}{4\varphi} \rfloor$ iterations.
   - $k = O(\sqrt{n/|x|})$ queries.



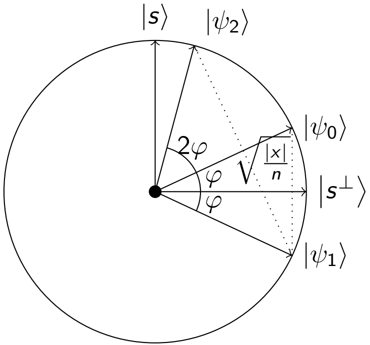
$$\varphi = \arcsin\sqrt{\frac{|x|}{n}} = \Omega\left(\sqrt{\frac{|x|}{n}}\right).$$
$$(2k+1)\varphi \approx \frac{\pi}{2}.$$

Improvements of Grover's algorithm

1. If $|x| > 0$:

$$|s\rangle = \frac{1}{\sqrt{|x|}} \sum_{\substack{j=1 \\ x_j=1}}^{n} |j\rangle, \quad |s^\perp\rangle = \frac{1}{\sqrt{n-|x|}} \sum_{\substack{j=1 \\ x_j=0}}^{n} |j\rangle.$$

2. If we know $|x|$:
   - $k = \lfloor \frac{\pi}{4\varphi} \rfloor$ iterations.
   - $k = O(\sqrt{n/|x|})$ queries.
3. If we don't know $|x|$:
   - Guess $|x| = n$, $|x| = n/2$, $|x| = n/4$, etc.
   - Check if outcome j satisfies $x_j = 1$.
   - Output "NO SOLUTION" if all tries failed.



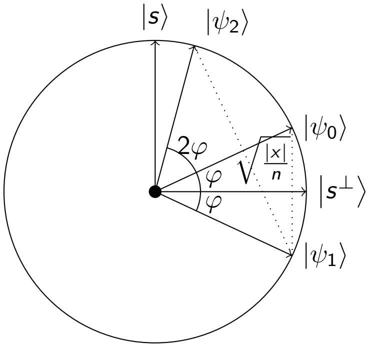
$$\varphi = \arcsin\sqrt{\frac{|x|}{n}} = \Omega\left(\sqrt{\frac{|x|}{n}}\right).$$

$$(2k+1)\varphi \approx \frac{\pi}{2}.$$

Improvements of Grover's algorithm

1. If $|x| > 0$:
$$|s\rangle = \frac{1}{\sqrt{|x|}} \sum_{\substack{j=1 \\ x_j=1}}^{n} |j\rangle, \quad |s^{\perp}\rangle = \frac{1}{\sqrt{n-|x|}} \sum_{\substack{j=1 \\ x_j=0}}^{n} |j\rangle.$$
2. If we know $|x|$:
   - $k = \lfloor \frac{\pi}{4\varphi} \rfloor$ iterations.
   - $k = O(\sqrt{n/|x|})$ queries.
3. If we don't know $|x|$:
   - Guess $|x| = n$, $|x| = n/2$, $|x| = n/4$, etc.
   - Check if outcome j satisfies $x_j = 1$.
   - Output "NO SOLUTION" if all tries failed.

Total queries to O_x: $O(\sqrt{n})$.



$$\varphi = \arcsin\sqrt{\frac{|x|}{n}} = \Omega\left(\sqrt{\frac{|x|}{n}}\right).$$
$$(2k+1)\varphi \approx \frac{\pi}{2}.$$

Application: Collision finding [BHT'97]

Application: Collision finding [BHT'97]

Problem: (Collision finding)

1. *Input:* $x \in \mathcal{D}^n$, $|\mathcal{D}| = n/2$.
   - Every element appears exactly twice in x.

Application: Collision finding [BHT'97]

Problem: (Collision finding)

1. *Input:* $x \in \mathcal{D}^n$, $|\mathcal{D}| = n/2$.
   - Every element appears exactly twice in x.
   - *Example:* $\mathcal{D} = \{A, B, C\}$, $n = 6$, $x = BCACAB$.

Application: Collision finding [BHT'97]

Problem: (Collision finding)

1. *Input:* $x \in \mathcal{D}^n$, $|\mathcal{D}| = n/2$.
   - Every element appears exactly twice in x.
   - *Example:* $\mathcal{D} = \{A, B, C\}$, $n = 6$, $x = BCACAB$.
2. *Output:* j, j' such that $x_j = x_{j'}$ and $j \neq j'$.

Application: Collision finding [BHT'97]

Problem: (Collision finding)

1. *Input:* $x \in \mathcal{D}^n$, $|\mathcal{D}| = n/2$.
    - Every element appears exactly twice in x.
    - *Example:* $\mathcal{D} = \{A, B, C\}$, $n = 6$, $x = BCACAB$.
2. *Output:* j, j' such that $x_j = x_{j'}$ and $j \neq j'$.

Classical algorithm:

1. Query in random order.

. . . . . .

Application: Collision finding [BHT'97]

Problem: (Collision finding)

1. *Input:* $x \in \mathcal{D}^n$, $|\mathcal{D}| = n/2$.
   - Every element appears exactly twice in x.
   - *Example:* $\mathcal{D} = \{A, B, C\}$, $n = 6$, $x = BCACAB$.
2. *Output:* j, j' such that $x_j = x_{j'}$ and $j \neq j'$.

Classical algorithm:

1. Query in random order.

$$\cdot\, C \cdot \cdot \cdot \cdot$$

Application: Collision finding [BHT'97]

Problem: (Collision finding)

1. *Input:* $x \in \mathcal{D}^n$, $|\mathcal{D}| = n/2$.
   - Every element appears exactly twice in x.
   - *Example:* $\mathcal{D} = \{A, B, C\}$, $n = 6$, $x = BCACAB$.
2. *Output:* j, j' such that $x_j = x_{j'}$ and $j \neq j'$.

Classical algorithm:

1. Query in random order.

$$\cdot\, C \cdot \cdot \cdot B$$

Application: Collision finding [BHT'97]

Problem: (Collision finding)

1. *Input:* $x \in \mathcal{D}^n$, $|\mathcal{D}| = n/2$.
   - Every element appears exactly twice in x.
   - *Example:* $\mathcal{D} = \{A, B, C\}$, $n = 6$, $x = BCACAB$.
2. *Output:* j, j' such that $x_j = x_{j'}$ and $j \neq j'$.

Classical algorithm:

1. Query in random order.

$$\cdot\, C \cdot C \cdot B$$

Application: Collision finding [BHT'97]

Problem: (Collision finding)

1. *Input:* $x \in \mathcal{D}^n$, $|\mathcal{D}| = n/2$.
   - Every element appears exactly twice in x.
   - *Example:* $\mathcal{D} = \{A, B, C\}$, $n = 6$, $x = BCACAB$.
2. *Output:* j, j' such that $x_j = x_{j'}$ and $j \neq j'$.

Classical algorithm:

1. Query in random order.

   $\cdot\, C \cdot C \cdot B$
2. $O(\sqrt{n})$ queries suffice *(birthday paradox)*.

Application: Collision finding [BHT'97]

Problem: (Collision finding)

1. *Input:* $x \in \mathcal{D}^n$, $|\mathcal{D}| = n/2$.
   - Every element appears exactly twice in x.
   - *Example:* $\mathcal{D} = \{A, B, C\}$, $n = 6$, $x = BCACAB$.
2. *Output:* j, j' such that $x_j = x_{j'}$ and $j \neq j'$.

Classical algorithm:

1. Query in random order.
   $\cdot\, C \cdot C \cdot B$
2. $O(\sqrt{n})$ queries suffice *(birthday paradox)*.

Quantum algorithm:

1. Query k random elements in the list.

$$\begin{array}{c|cccccc} x & B & C & A & C & A & B \\ & B & C & \cdot & \cdot & \cdot & \cdot \end{array}$$

Application: Collision finding [BHT'97]

Problem: (Collision finding)

1. *Input:* $x \in \mathcal{D}^n$, $|\mathcal{D}| = n/2$.
   - Every element appears exactly twice in x.
   - *Example:* $\mathcal{D} = \{A, B, C\}$, $n = 6$, $x = BCACAB$.
2. *Output:* j, j' such that $x_j = x_{j'}$ and $j \neq j'$.

Classical algorithm:

1. Query in random order.
   $\cdot\, C \cdot C \cdot B$
2. $O(\sqrt{n})$ queries suffice *(birthday paradox)*.

Quantum algorithm:

1. Query k random elements in the list.
2. Let $y \in \{0,1\}^n$ with y_j if j forms a collision with any of the already queries entries.

$$\begin{array}{c|cccccc} x & B & C & A & C & A & B \\ & B & C & \cdot & \cdot & \cdot & \cdot \\ y & 0 & 0 & 0 & 1 & 0 & 1 \end{array}$$

Application: Collision finding [BHT'97]

Problem: (Collision finding)

1. *Input:* $x \in \mathcal{D}^n$, $|\mathcal{D}| = n/2$.
   - Every element appears exactly twice in x.
   - *Example:* $\mathcal{D} = \{A, B, C\}$, $n = 6$, $x = BCACAB$.
2. *Output:* j, j' such that $x_j = x_{j'}$ and $j \neq j'$.

Classical algorithm:

1. Query in random order.

   $\cdot\, C \cdot C \cdot B$
2. $O(\sqrt{n})$ queries suffice *(birthday paradox)*.

Quantum algorithm:

1. Query k random elements in the list.
2. Let $y \in \{0,1\}^n$ with y_j if j forms a collision with any of the already queries entries.

$$\begin{array}{c|cccccc} x & B & C & A & C & A & B \\ & B & C & \cdot & \cdot & \cdot & \cdot \\ y & 0 & 0 & 0 & 1 & 0 & 1 \end{array}$$

   - $|y| = k$.

Application: Collision finding [BHT'97]

Problem: (Collision finding)

1. *Input:* $x \in \mathcal{D}^n$, $|\mathcal{D}| = n/2$.
   - Every element appears exactly twice in x.
   - *Example:* $\mathcal{D} = \{A, B, C\}$, $n = 6$, $x = BCACAB$.
2. *Output:* j, j' such that $x_j = x_{j'}$ and $j \neq j'$.

Classical algorithm:

1. Query in random order.
   $\cdot\, C \cdot C \cdot B$
2. $O(\sqrt{n})$ queries suffice *(birthday paradox)*.

Quantum algorithm:

1. Query k random elements in the list.
2. Let $y \in \{0,1\}^n$ with y_j if j forms a collision with any of the already queries entries.

$$\begin{array}{c|cccccc} x & B & C & A & C & A & B \\ & B & C & \cdot & \cdot & \cdot & \cdot \\ y & 0 & 0 & 0 & 1 & 0 & 1 \end{array}$$

   - $|y| = k$.
   - Grover: $O(\sqrt{n/k})$ queries.

Application: Collision finding [BHT'97]

Problem: (Collision finding)

1. *Input:* $x \in \mathcal{D}^n$, $|\mathcal{D}| = n/2$.
   - Every element appears exactly twice in x.
   - *Example:* $\mathcal{D} = \{A, B, C\}$, $n = 6$, $x = BCACAB$.
2. *Output:* j, j' such that $x_j = x_{j'}$ and $j \neq j'$.

Classical algorithm:

1. Query in random order.
   $\cdot\, C \cdot C \cdot B$
2. $O(\sqrt{n})$ queries suffice *(birthday paradox)*.

Quantum algorithm:

1. Query k random elements in the list.
2. Let $y \in \{0,1\}^n$ with y_j if j forms a collision with any of the already queries entries.

$$\begin{array}{c|cccccc} x & B & C & A & C & A & B \\ & B & C & \cdot & \cdot & \cdot & \cdot \\ y & 0 & 0 & 0 & 1 & 0 & 1 \end{array}$$

   - $|y| = k$.
   - Grover: $O(\sqrt{n/k})$ queries.
3. Total queries: $O(k + \sqrt{n/k})$.

Application: Collision finding [BHT'97]

Problem: (Collision finding)

1. *Input:* $x \in \mathcal{D}^n$, $|\mathcal{D}| = n/2$.
   - Every element appears exactly twice in x.
   - *Example:* $\mathcal{D} = \{A, B, C\}$, $n = 6$, $x = BCACAB$.
2. *Output:* j, j' such that $x_j = x_{j'}$ and $j \neq j'$.

Classical algorithm:

1. Query in random order.
   $\cdot\, C \cdot C \cdot B$
2. $O(\sqrt{n})$ queries suffice *(birthday paradox)*.

Quantum algorithm:

1. Query k random elements in the list.
2. Let $y \in \{0,1\}^n$ with y_j if j forms a collision with any of the already queries entries.

$$\begin{array}{c|cccccc} x & B & C & A & C & A & B \\ & B & C & \cdot & \cdot & \cdot & \cdot \\ y & 0 & 0 & 0 & 1 & 0 & 1 \end{array}$$

   - $|y| = k$.
   - Grover: $O(\sqrt{n/k})$ queries.
3. Total queries: $O(k + \sqrt{n/k})$.
4. Minimized for $k = \Theta(n^{1/3})$.
5. $O(n^{1/3})$ queries – *subquadratic improvement!*

Summary

Summary:

1. Quantum algorithms.
2. Grover's algorithm:
   - Quadratic improvement.
3. Application: collision finding:
   - Subquadratic improvement.

Summary

Summary:

1. Quantum algorithms.
2. Grover's algorithm:
   - Quadratic improvement.
3. Application: collision finding:
   - Subquadratic improvement.

Thanks for your attention!

`cornelissen@irif.fr`