Complexity of recognizing Dyck languages of bounded height with quantum query algorithms.

Maxime CAUTRÈS

Faculty of Computing
University of Latvia

31/08/2022

Sommaire

Classical and quantum computers are both made with simple components.

Figure: A Boolean circuit (Full adder).

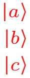


Figure: A Quantum circuit.

Classical and quantum computers are both made with simple components.

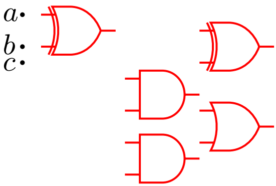


Figure: A Boolean circuit (Full adder).

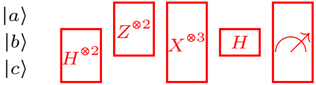


Figure: A Quantum circuit.

Classical and quantum computers are both made with simple components.

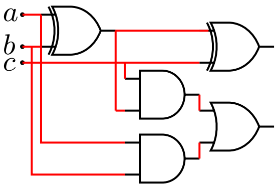


Figure: A Boolean circuit (Full adder).

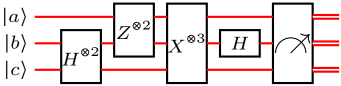


Figure: A Quantum circuit.

Classical and quantum computers are both made with simple components.

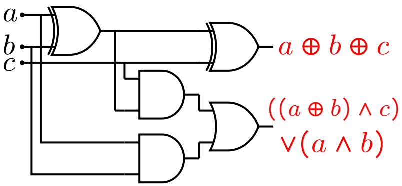


Figure: A Boolean circuit (Full adder).

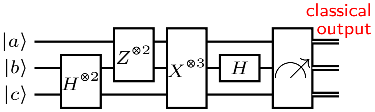


Figure: A Quantum circuit.

Interacting with qubits is more complexe.

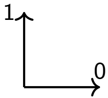


Figure: A classical bit

Interacting with qubits is more complexe.

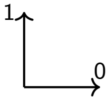


Figure: A classical bit

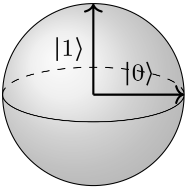


Figure: A quantum bit.

Interacting with qubits is more complexe.

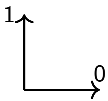


Figure: A classical bit

A	B	$A \oplus B$
0	0	0
0	1	1
1	0	1
1	1	0

Figure: Truth table on 2 bits.

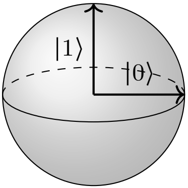


Figure: A quantum bit.

Interacting with qubits is more complexe.

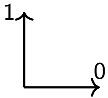


Figure: A classical bit

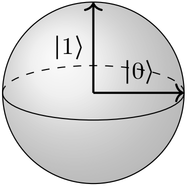


Figure: A quantum bit.

A	B	$A \oplus B$
0	0	0
0	1	1
1	0	1
1	1	0

Figure: Truth table on 2 bits.

$$H^{\otimes 2} = \frac{1}{2}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix}$$

Figure: Unitary matrix on 2 qubits.

Quantum query algorithm is just a quantum circuit.

$$x = \underbrace{100101\ldots01011}_{n}$$

Figure: Structure of a quantum query algorithm.

Quantum query algorithm is just a quantum circuit.

$$x = \underbrace{100101\ldots01011}_{n}$$

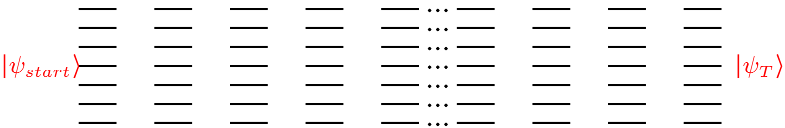


Figure: Structure of a quantum query algorithm.

Quantum query algorithm is just a quantum circuit.

$$x = \underbrace{100101\ldots01011}_{n}$$

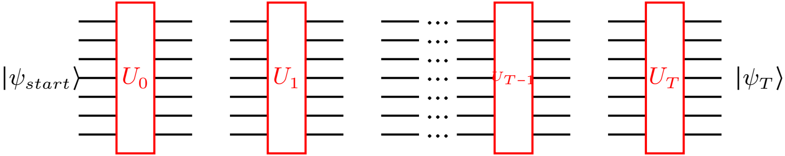


Figure: Structure of a quantum query algorithm.

Quantum query algorithm is just a quantum circuit.

$$x = \underbrace{100101\ldots01011}_{n}$$

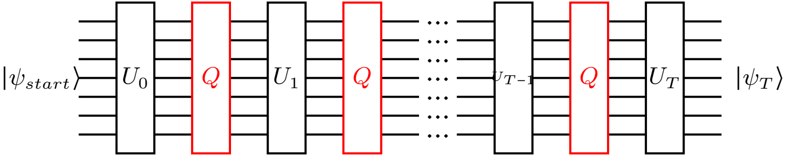


Figure: Structure of a quantum query algorithm.

Quantum query algorithm is just a quantum circuit.

$$x = \underbrace{100101\ldots01011}_{n}$$

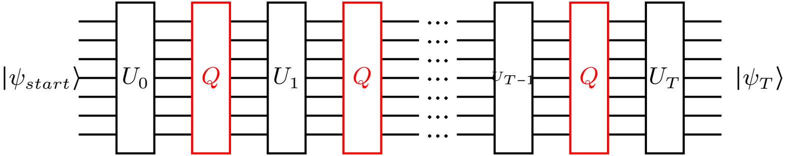


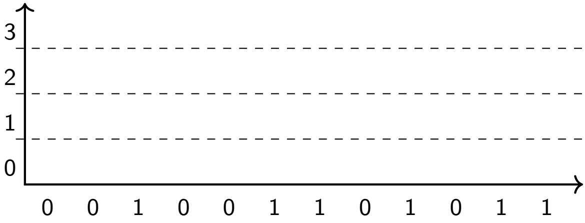
Figure: Structure of a quantum query algorithm.

$$Q(f)$$

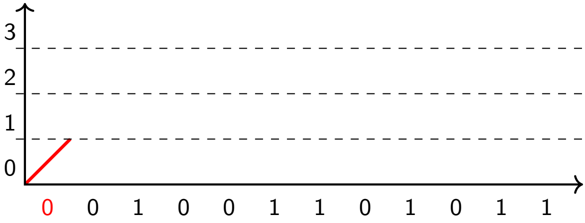
Dyck words of bounded height are a natural restriction of Dyck words.

0 0 1 0 0 1 1 0 1 0 1 1

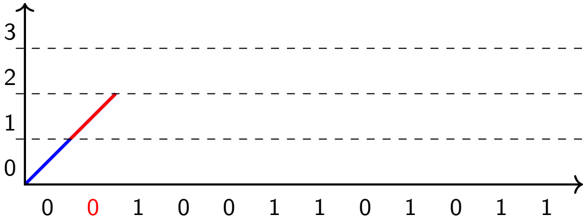
Dyck words of bounded height are a natural restriction of Dyck words.



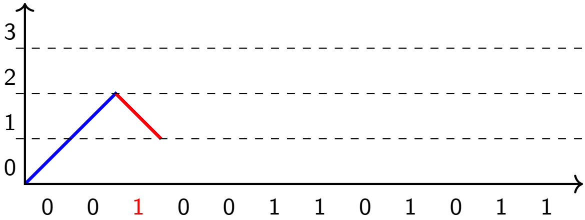
Dyck words of bounded height are a natural restriction of Dyck words.



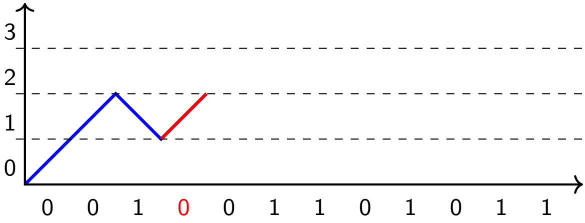
Dyck words of bounded height are a natural restriction of Dyck words.



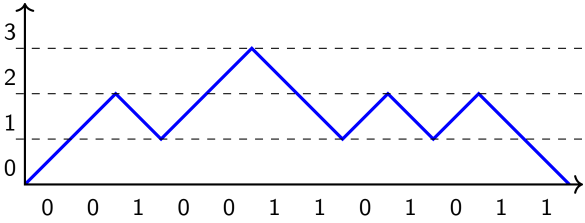
Dyck words of bounded height are a natural restriction of Dyck words.



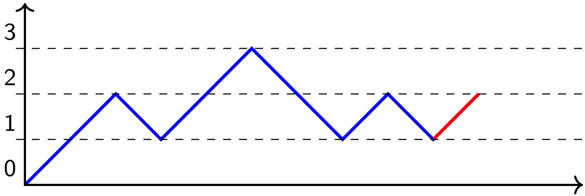
Dyck words of bounded height are a natural restriction of Dyck words.



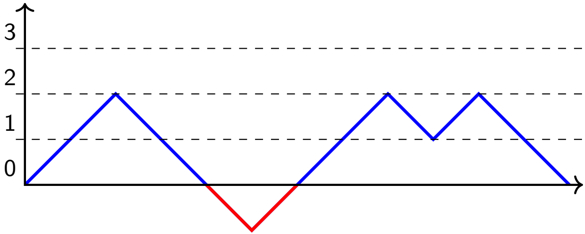
Dyck words of bounded height are a natural restriction of Dyck words.



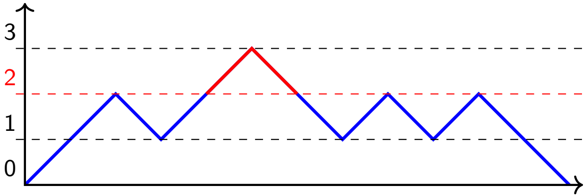
Dyck words of bounded height are a natural restriction of Dyck words.



Dyck words of bounded height are a natural restriction of Dyck words.



Dyck words of bounded height are a natural restriction of Dyck words.



$$\textsc{Dyck}_k$$

A general result that help but not close the $Q(\textsc{Dyck}_k)$ problem.

The Trichotomy theorem:(Aaronson, Grier and Schaeffer [1, 2019])

$$\text{Star Free Languages} \implies \tilde{\Theta}\left(\sqrt{n}\right)$$

A general result that help but not close the $Q(\textsc{Dyck}_k)$ problem.

The Trichotomy theorem:(Aaronson, Grier and Schaeffer [1, 2019])

$$\text{Star Free Languages} \implies \tilde{\Theta}\left(\sqrt{n}\right)$$

Application:

$$\textsc{Dyck}_k \in \text{Star free languages}$$

A general result that help but not close the $Q(\textsc{Dyck}_k)$ problem.

The Trichotomy theorem:(Aaronson, Grier and Schaeffer [1, 2019])

$$\text{Star Free Languages} \implies \tilde{\Theta}\left(\sqrt{n}\right)$$

Application:

$$\textsc{Dyck}_k \in \text{Star free languages}$$

Implication:

$$Q(\textsc{Dyck}_{k,n}) = \Theta\left(\sqrt{n}\log_2(n)^{p(k)}\right)$$

First step, one try to have a good upper bounds.

- **Algorithms:**

Require: $n \geq 0$ and $k \geq 1$
Ensure: $|x| = n$
$x \leftarrow 1^k x 0^k$
$v \leftarrow \textsc{FindAny}_{k+1}(0, n + 2 * k - 1, \{1, -1\})$
return v $= \textsc{Null}$

Figure: Ambainis' algorithm (small piece).

First step, one try to have a good upper bounds.

- **Algorithms:**

Require: $n \geq 0$ and $k \geq 1$
Ensure: $|x| = n$
$x \leftarrow 1^k x 0^k$
$v \leftarrow \textsc{FindAny}_{k+1}(0, n + 2 * k - 1, \{1, -1\})$
return v $= \textsc{Null}$

Figure: Ambainis' algorithm (small piece).

$$O\left(\sqrt{n}(\log_2(n))^{0.5k}\right)$$

First step, one try to have a good upper bounds.

- **Algorithms:**

Require: $n \geq 0$ and $k \geq 1$
Ensure: $|x| = n$
$x \leftarrow 1^k x 0^k$
$v \leftarrow \textsc{FindAny}_{k+1}(0, n + 2 * k - 1, \{1, -1\})$
return v $= \textsc{Null}$

Figure: Ambainis' algorithm (small piece).

$$O\left(\sqrt{n}(\log_2(n))^{0.5k}\right)$$

- **Reductions to:**

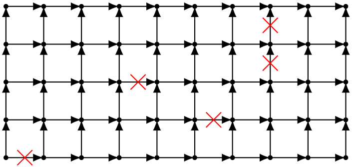


Figure: A reduction to 2D directed grid connectivity.

First step, one try to have a good upper bounds.

- **Algorithms:**

Require: $n \geq 0$ and $k \geq 1$
Ensure: $|x| = n$
$x \leftarrow 1^k x 0^k$
$v \leftarrow \textsc{FindAny}_{k+1}(0, n + 2 * k - 1, \{1, -1\})$
return v $= \textsc{Null}$

Figure: Ambainis' algorithm (small piece).

$$O\left(\sqrt{n}(\log_2(n))^{0.5k}\right)$$

- **Reductions to:**

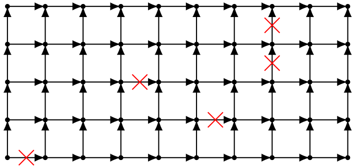


Figure: A reduction to 2D directed grid connectivity.

$$O\left(\sqrt{n}(\log_2(n))^{0.5(k-1)}\right)$$

Second step, one try to prove the optimality with a matching lower bound.

- **Adversary methods:**

No result yet

Second step, one try to prove the optimality with a matching lower bound.

- **Adversary methods:**

No result yet

- **Reduction from:**

$$\mathrm{Ex}_{2m}^{m|m+1}(x) = 0 \Leftrightarrow |x|_0 - |x|_1 = 2$$
$$\mathrm{Ex}_{2m}^{m|m+1}(x) = 1 \Leftrightarrow |x|_0 - |x|_1 = 0$$

$$\Omega\left(\sqrt{n c^k}\right)$$

A natural goal is to made the bounds match.

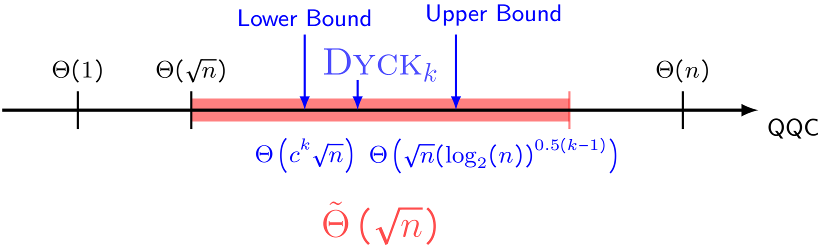


Figure: Representation of the different bounds.

Sommaire

Every k is not as simple as 1.

- $k = 1$:

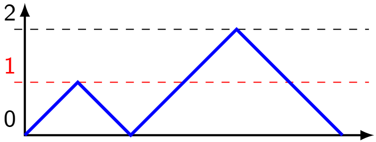


Figure: A dyck word of height 2.

Every k is not as simple as 1.

- $k = 1$:

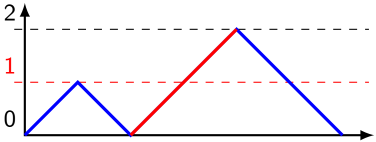


Figure: A dyck word of height 2.

$$O\left(\sqrt{n}\right)$$

Every k is not as simple as 1.

- $k = 1$:

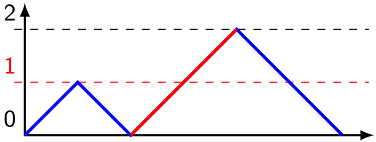


Figure: A dyck word of height 2.

$$O\left(\sqrt{n}\right)$$

- $k = 2$:

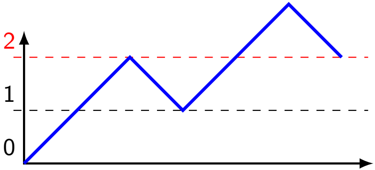


Figure: A substring of height 3.

Every k is not as simple as 1.

- $k = 1$:

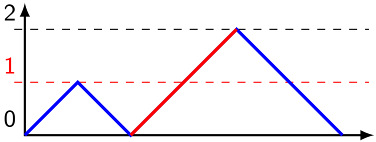


Figure: A dyck word of height 2.

$$O\left(\sqrt{n}\right)$$

- $k = 2$:

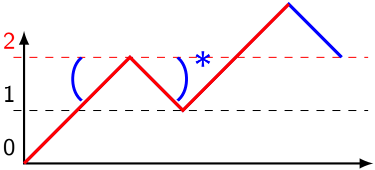


Figure: A substring of height 3.

Every k is not as simple as 1.

- $k = 1$:

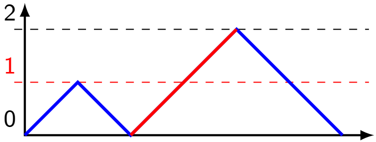


Figure: A dyck word of height 2.

$$O\left(\sqrt{n}\right)$$

- $k = 2$:

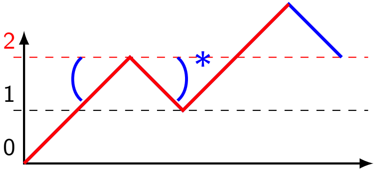


Figure: A substring of height 3.

$$O\left(\sqrt{n\log_2(n)}\right)$$

Small definition and intuition

- $\pm k$ **strings:**

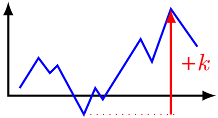


Figure: Representation of a $+k$ string.

Small definition and intuition

- **$\pm k$ strings:**

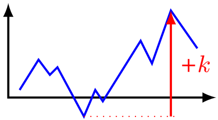


Figure: Representation of a $+k$ string.

- **Minimal $\pm k$ strings:**

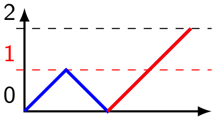


Figure: A non-minimal $+2$ string.

Small definition and intuition

- **$\pm k$ strings:**

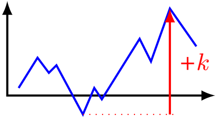


Figure: Representation of a $+k$ string.

- **Minimal $\pm k$ strings:**

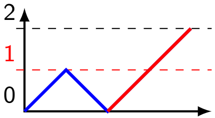


Figure: A non-minimal $+2$ string.

- **Minimal decomposition:**

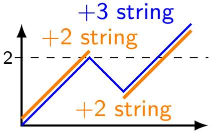


Figure: A +3 string decomposition.

A first inductive algorithm with a quantum query complexity of $\tilde{\Theta}\left(\sqrt{n}\right)$

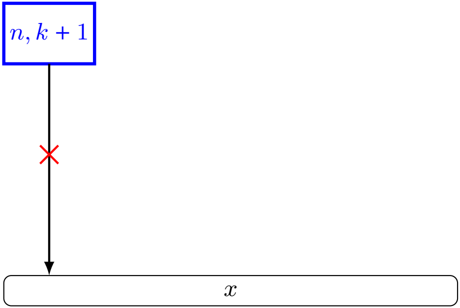


Figure: Schema to present the idea of the original algorithm. Not show the real computation of the QQC.

A first inductive algorithm with a quantum query complexity of $\tilde{\Theta}\left(\sqrt{n}\right)$

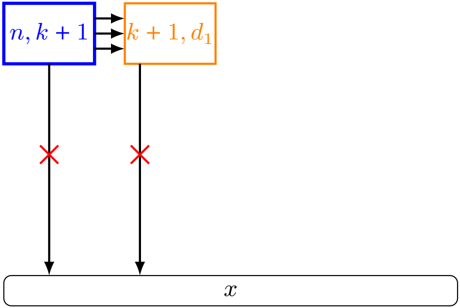


Figure: Schema to present the idea of the original algorithm. Not show the real computation of the QQC.

A first inductive algorithm with a quantum query complexity of $\tilde{\Theta}\left(\sqrt{n}\right)$

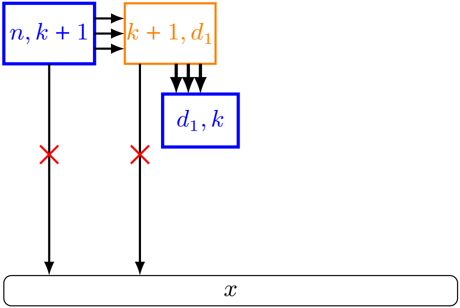


Figure: Schema to present the idea of the original algorithm. Not show the real computation of the QQC.

A first inductive algorithm with a quantum query complexity of $\tilde{\Theta}\left(\sqrt{n}\right)$

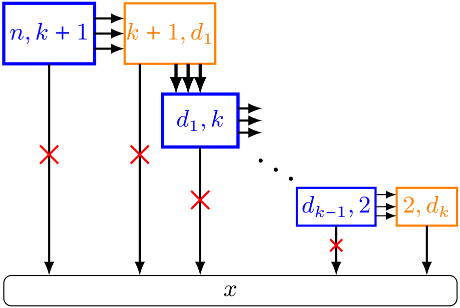


Figure: Schema to present the idea of the original algorithm. Not show the real computation of the QQC.

A first inductive algorithm with a quantum query complexity of $\tilde{\Theta}\left(\sqrt{n}\right)$

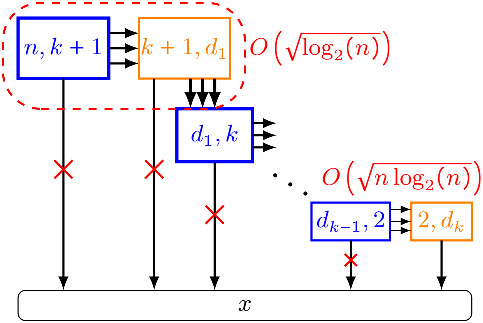


Figure: Schema to present the idea of the original algorithm. Not show the real computation of the QQC.

- **Original QQC:**

$$O\left(\sqrt{n}(\log_2(n))^{0.5k}\right)$$

A first inductive algorithm with a quantum query complexity of $\tilde{\Theta}\left(\sqrt{n}\right)$

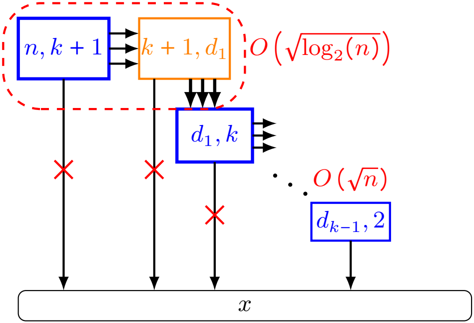


Figure: Schema to present the idea of the original algorithm. Not show the real computation of the QQC.

- **Original QQC:**

$$O\left(\sqrt{n}(\log_2(n))^{0.5k}\right)$$

- **Small update:**

$$O\left(\sqrt{n}(\log_2(n))^{0.5(k-1)}\right)$$

A first inductive algorithm with a quantum query complexity of $\tilde{\Theta}\left(\sqrt{n}\right)$

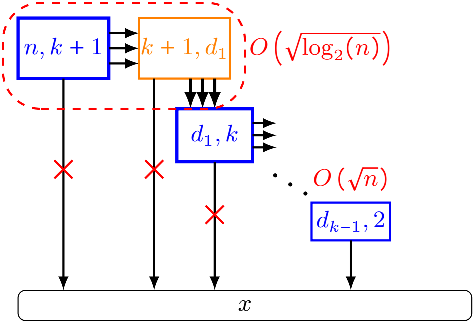


Figure: Schema to present the idea of the original algorithm. Not show the real computation of the QQC.

- **Original QQC:**

$$O\left(\sqrt{n}(\log_2(n))^{0.5k}\right)$$

- **Small update:**

$$O\left(\sqrt{n}(\log_2(n))^{0.5(k-1)}\right)$$

- **Bigger update:**

$$O\left(\sqrt{n}(\log_2(n))^{0.5(k-2)}\right)$$

A first inductive algorithm with a quantum query complexity of $\tilde{\Theta}\left(\sqrt{n}\right)$

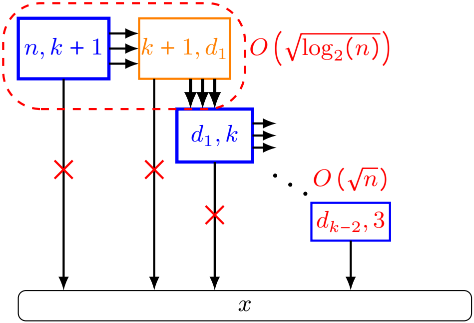


Figure: Schema to present the idea of the original algorithm. Not show the real computation of the QQC.

- **Original QQC:**

$$O\left(\sqrt{n}(\log_2(n))^{0.5k}\right)$$

- **Small update:**

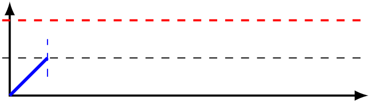
$$O\left(\sqrt{n}(\log_2(n))^{0.5(k-1)}\right)$$

- **Bigger update:**

$$O\left(\sqrt{n}(\log_2(n))^{0.5(k-2)}\right)$$

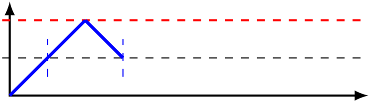
An easy and fast algorithm exists for k=2.

- **Second half:**



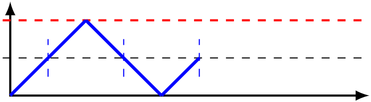
An easy and fast algorithm exists for k=2.

- **Second half:**



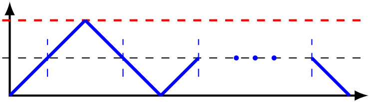
An easy and fast algorithm exists for k=2.

- **Second half:**



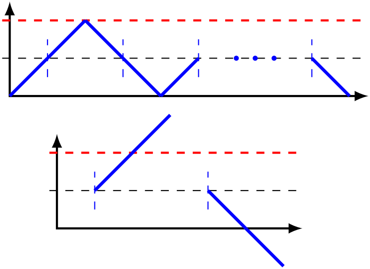
An easy and fast algorithm exists for k=2.

- **Second half:**



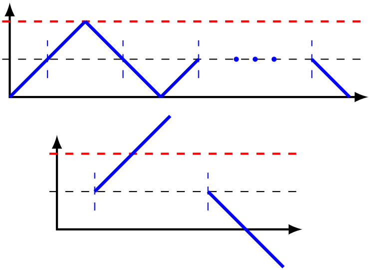
An easy and fast algorithm exists for k=2.

- **Second half:**

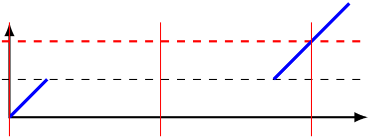


An easy and fast algorithm exists for k=2.

- **Second half:**

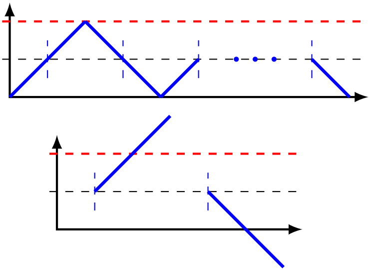


- **First half**:

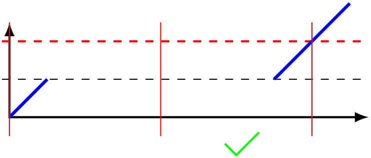


An easy and fast algorithm exists for k=2.

- **Second half:**

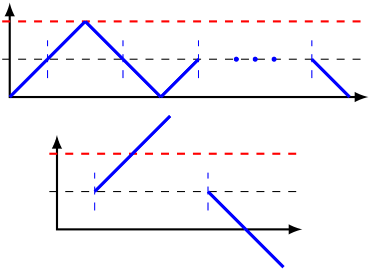


- **First half**:

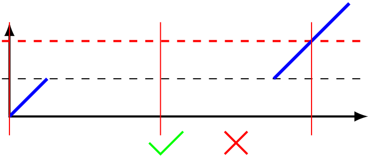


An easy and fast algorithm exists for k=2.

- **Second half:**

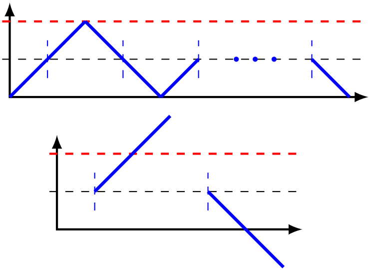


- **First half**:

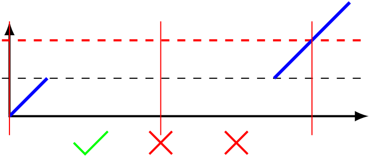


An easy and fast algorithm exists for k=2.

- **Second half:**

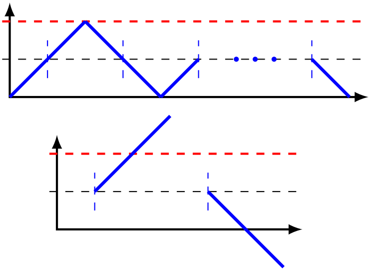


- **First half**:

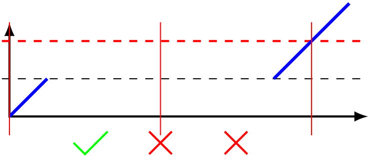


An easy and fast algorithm exists for k=2.

- **Second half:**



- **First half**:



$$O\left(\sqrt{n}\right)$$

Sommaire

1. Introduction
2. The progress to reduce the DYCK_k Quantum Query Complexity
3. New idea to get better quantum query complexity bounds
   - For lower bounds
   - For upper bounds:
   - Conclusion

The search for a new reduction.

- Tightness of $\text{Ex}_{2m}^{m|m+1}$'s one.
- New reduction:

$$\text{Ex}_{2m}^{m|m+1} \leq \textit{ new problem } \leq \textsc{Dyck}_k.$$

There is a lot of recursive calls:

for $1 \leq d_1 \leq \log_2(n)$ **do**
  for $1 \leq d_2 < d_1$ **do**
    $\ddots$
      for $1 \leq d_{k-1} < d_{k-2}$ **do**
        `Do smthg`

Figure: Currents algorithm behavior

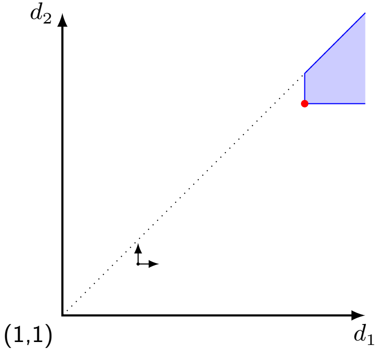


Figure: Graph for $k = 4$.

To conclude:

What has been done:

- Trichotomy
- Adversary methods
- Reduction methods
- New algorithm: $O\left(\sqrt{n}(\log_2(n))^{0.5k}\right) \rightarrow O\left(\sqrt{n}(\log_2(n))^{0.5(k-2)}\right)$

To conclude:

What has been done:

- Trichotomy
- Adversary methods
- Reduction methods
- New algorithm: $O\left(\sqrt{n}(\log_2(n))^{0.5k}\right) \to O\left(\sqrt{n}(\log_2(n))^{0.5(k-2)}\right)$

Possible idea to go further:

- Prove that new upper bound approach cannot work
- New algorithm
- General Adversary method

Scott Aaronson, Daniel Grier, and Luke Schaeffer.
A quantum query complexity trichotomy for regular languages, 2018.