Quantum programming and algorithms

Master of Computer Science, Minerve and ATHENA

Nicolas Ollinger, Ioan Todinca
2025-2026

lec 4

4. Grover's algorithm

[BBHT$_{1996}$] Boyer et al

The problem. We are given a function $f : \{0,1\}^n \to \{0,1\}$, as a black box (circuit). We aim to find, if it exists, a vector $x \in \{0,1\}^n$ such that $f(x) = 1$.

Grover's algorithm (1996) solves the problem in time $O(\sqrt{2^n})$ while any classical algorithm requires $\Omega(2^n)$ time.

It is a probabilistic algorithm: it finds the solution with a probability of at least $2/3$.

Standard amplification techniques can bring it as close to 1 as desired. ← exercise session 5

SAT (satisfiability) problem, in classical terms: even if f is a known Boolean function, we cannot do better than, roughly, 2^n time, under some complexity assumptions.

Grover would be one of the most useful algorithms, providing a (polynomial) speed-up for many classical algorithms. More details during the last lecture.

Grover's Algorithm

$\longrightarrow O(\sqrt{N})$ optimal algorithm.

Unsorted Search Problem

input: an oracle

$$|x\rangle \to \boxed{U_f} \to |x\rangle, \qquad |y\rangle \to \boxed{U_f} \to |y \oplus f(x)\rangle$$

strong hypothesis: there is one x st $f(x) = 1$.

output: find x!

// if we can solve this with $O(\log N)$, where N is the number of inputs, steps / calls to the oracle, it would provide a polynomial way to solve (quantically) $\boxed{\text{NP-complete problems}}$.

BQP

The quantum analogue of P.

A language $L \in BQP$ iff there exists a uniformly poly-time family of quantum circuits $\{Q_n : n \in \mathbb{N}\}$ s.t.

- Q_n acts on n qubits
- $\forall x \in L \quad P(Q_{|x|}(x) = 1) \geq \frac{2}{3}$
- $\forall x \notin L \quad P(Q_{|x|}(x) = 0) \geq \frac{2}{3}$

produced by a polynomial time program $f : n \mapsto Q_n$

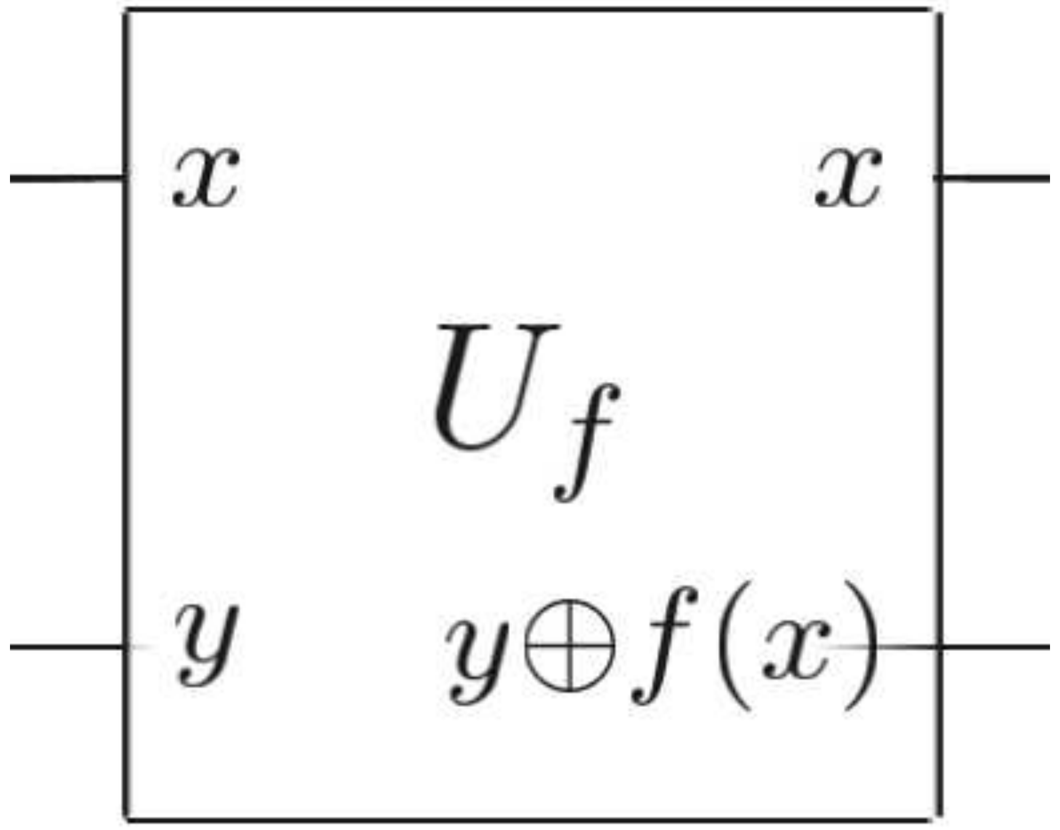
$$P \subseteq BQP$$

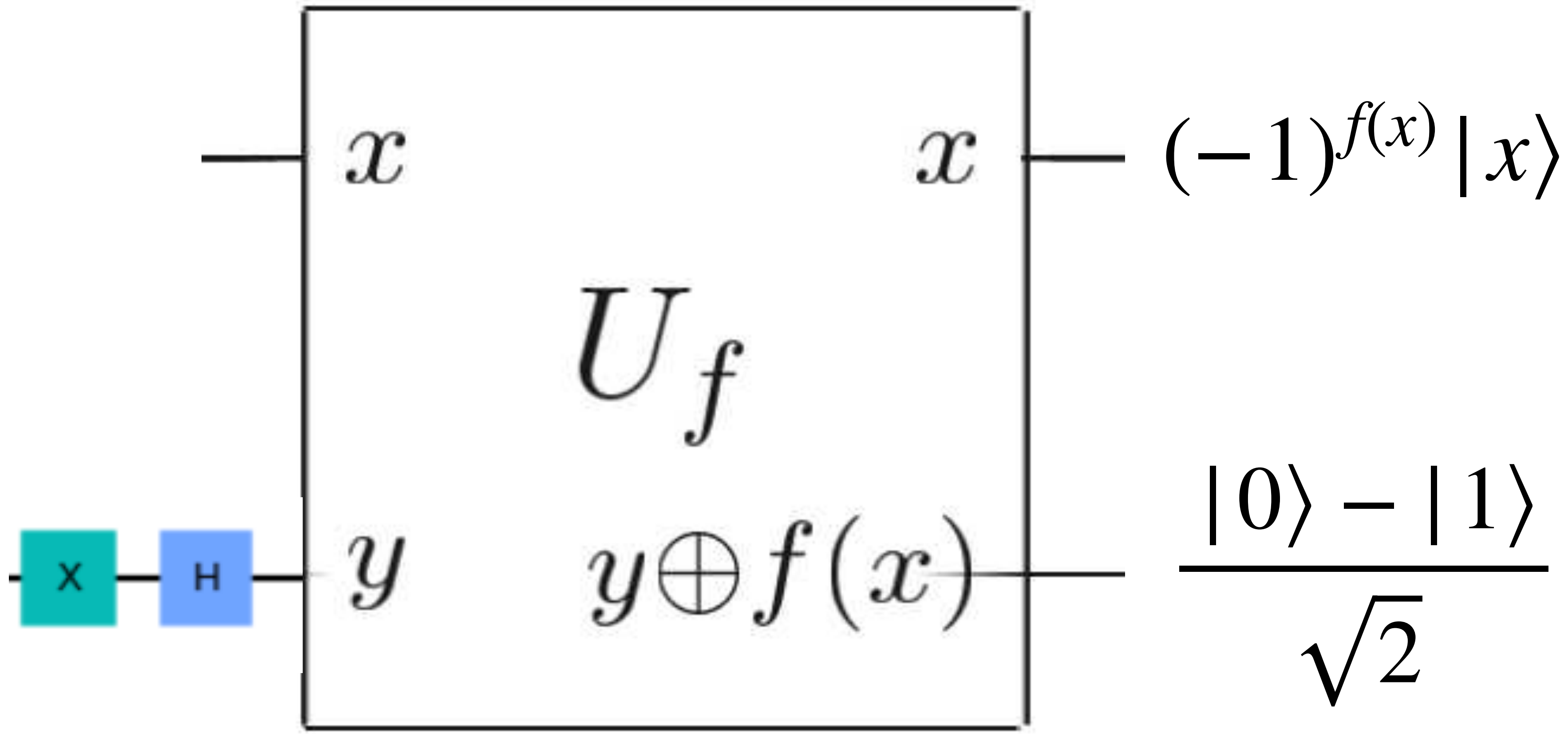
Grover gives a quadratic acceleration over brute-force when we have a polynomial checker.

oracle U_f

$$O(\sqrt{N})$$

4. Grover's algorithm - basic tools





Oracle Z_f

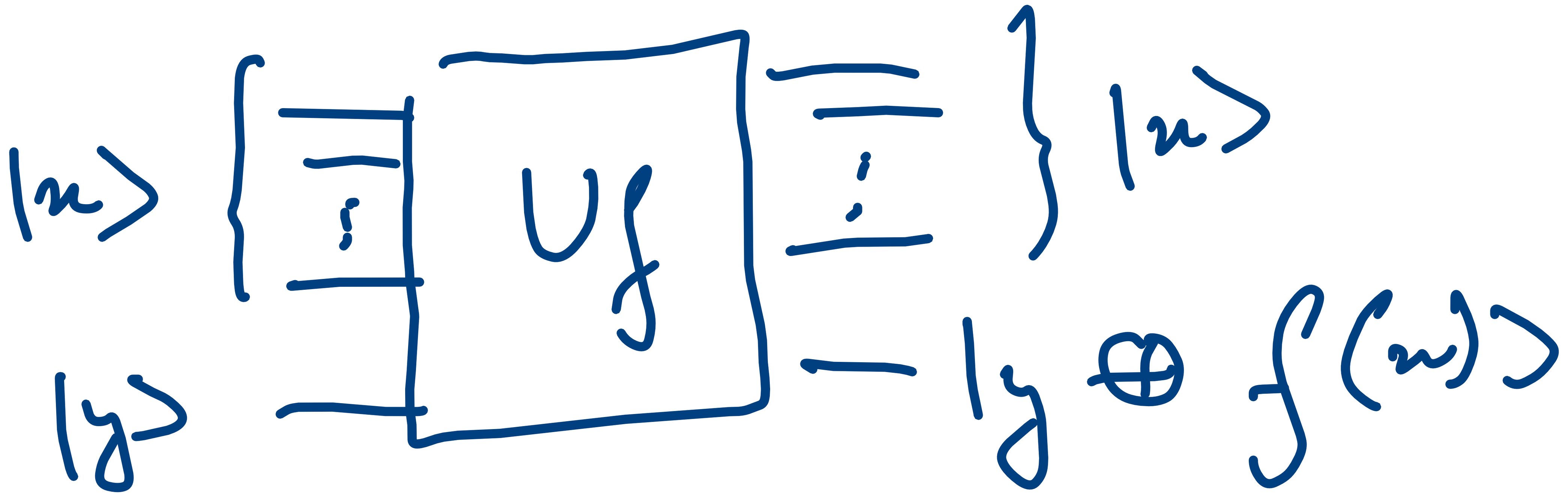
Reminder from previous lectures

- For any boolean function f, we can build $U_f : |x\rangle|y\rangle \mapsto |x\rangle|y \oplus f(x)\rangle$
- We denote $|-\rangle = \dfrac{|0\rangle - |1\rangle}{\sqrt{2}}$
- **By setting $y = |-\rangle$, we obtain as output** $(-1)^{f(x)}|x\rangle|-\rangle$
- **This new circuit is denoted Z_f.**

Oracle

$\downarrow$

Phase Shifting Oracle



with $|y\rangle = |-\rangle = \frac{1}{\sqrt{2}}\left(|0\rangle - |1\rangle\right)$

$$\boxed{U_f |x\rangle|-\rangle = (-1)^{f(x)} |x\rangle|-\rangle}$$

$$X \begin{cases} |0\rangle \mapsto |1\rangle \\ |1\rangle \mapsto |0\rangle \end{cases} \qquad X \begin{cases} |+\rangle \mapsto |+\rangle \\ |-\rangle \mapsto -|-\rangle \end{cases}$$

$$U_f = I - 2|x\rangle\langle x| \quad \text{with } f(x)=1$$

$$U_f = I - 2 \sum_{x:\, f(x)=1} |x\rangle\langle x|$$

multiple solutions

$|y\rangle\langle x|$ with $|x\rangle, |y\rangle$ in basis.

$$(|y\rangle\langle x|)|x\rangle = |y\rangle \underbrace{\langle x|x\rangle}_{1} = |y\rangle$$

$$(|y\rangle\langle x|)|z\rangle = |y\rangle \underbrace{\langle x|z\rangle}_{0} = 0$$

$x \perp z$

$$(I - 2|x\rangle\langle x|)|x\rangle = -|x\rangle$$

$$(I - 2|x\rangle\langle x|)|z\rangle = |z\rangle$$

$$A_0 = \{ x \mid f(x) = 0 \} \qquad A_1 = \{ x \mid f(x) = 1 \}$$

Let $\boxed{O = I - 2 \sum_{x \in A_1} |x\rangle\langle x|}$

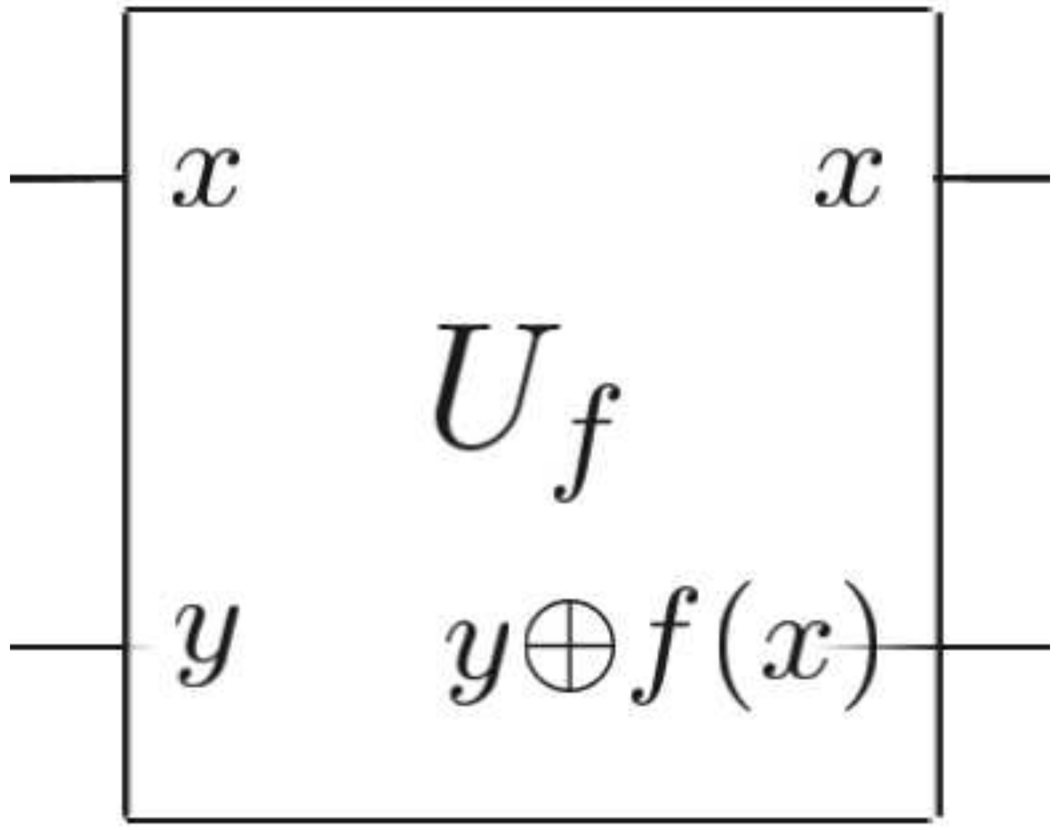
if $x \in A_0$, $O|x\rangle = I|x\rangle - 2 \sum_{y \in A_1} |y\rangle \underbrace{\langle y | x\rangle}_{0} = |x\rangle$

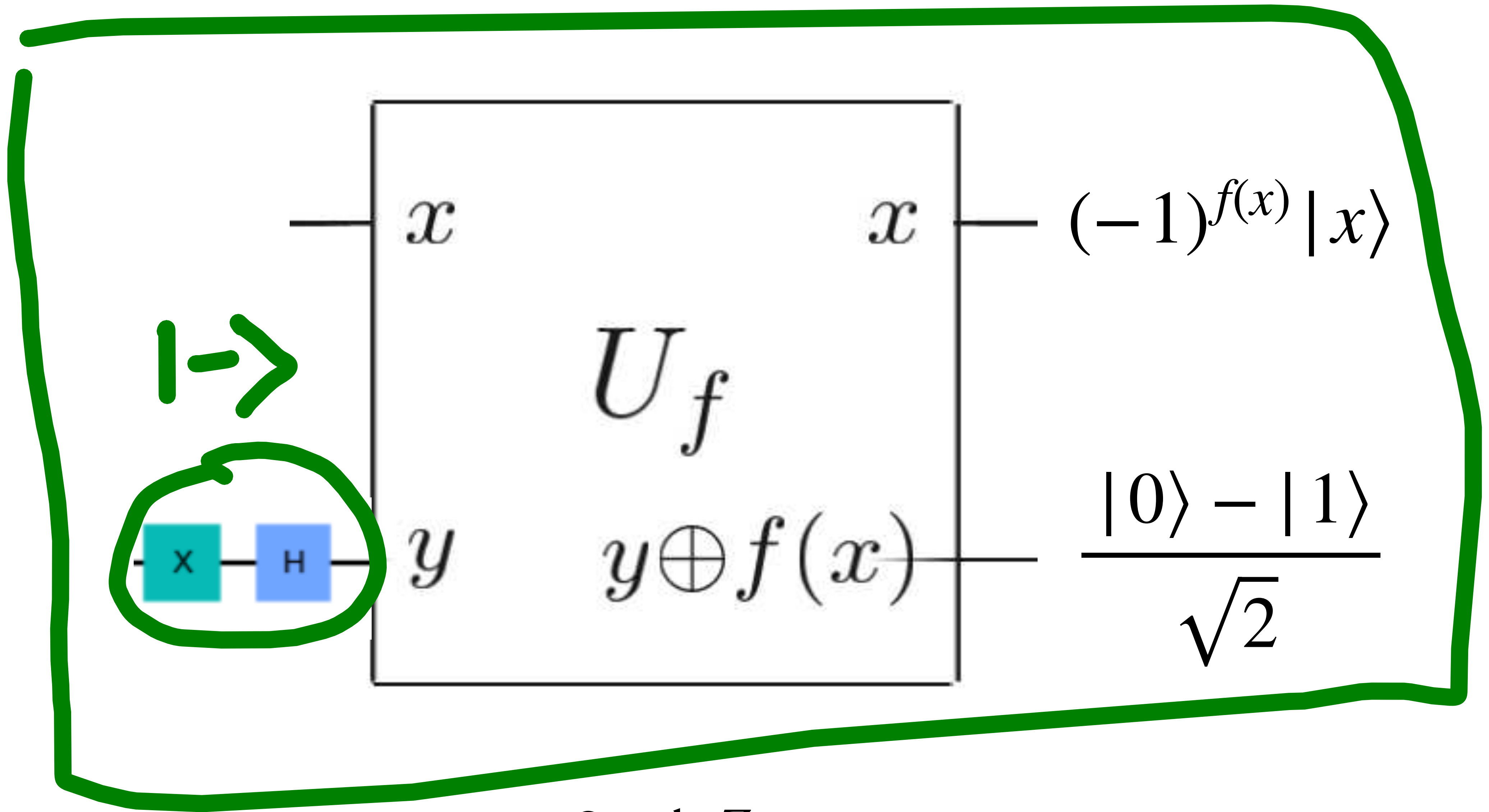
if $x \in A_1$, $O|x\rangle = I|x\rangle - 2|x\rangle = -|x\rangle$

Nielsen & Chuang (chap 6)

Amplitude Amplification

4. Grover's algorithm - basic tools





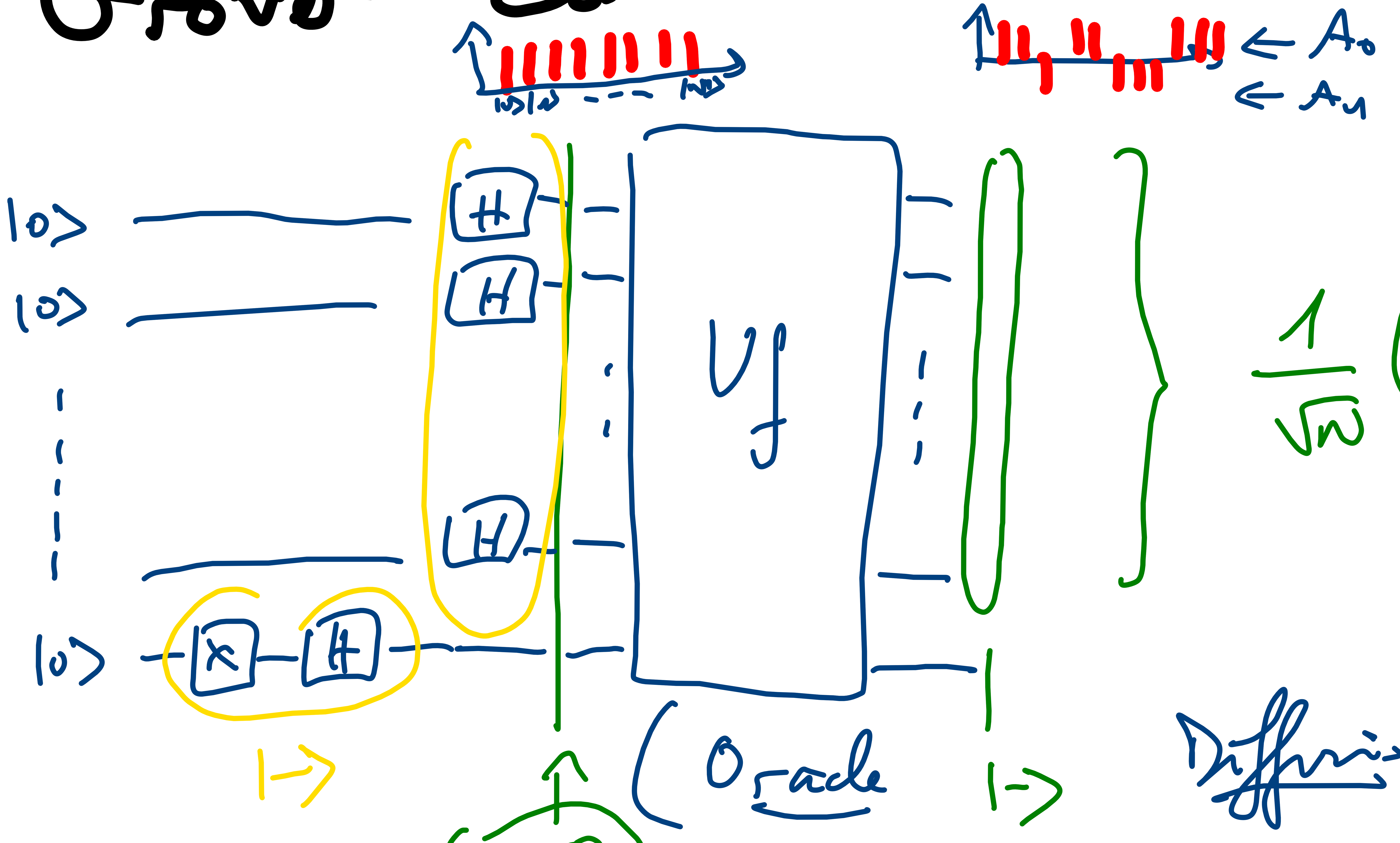
Oracle Z_f

Simplifying hypothesis. Assume that our function $f : \{0,1\}^n \to \{0,1\}$ is such that there exists a unique x_1 satisfying $f(x_1) = 1$

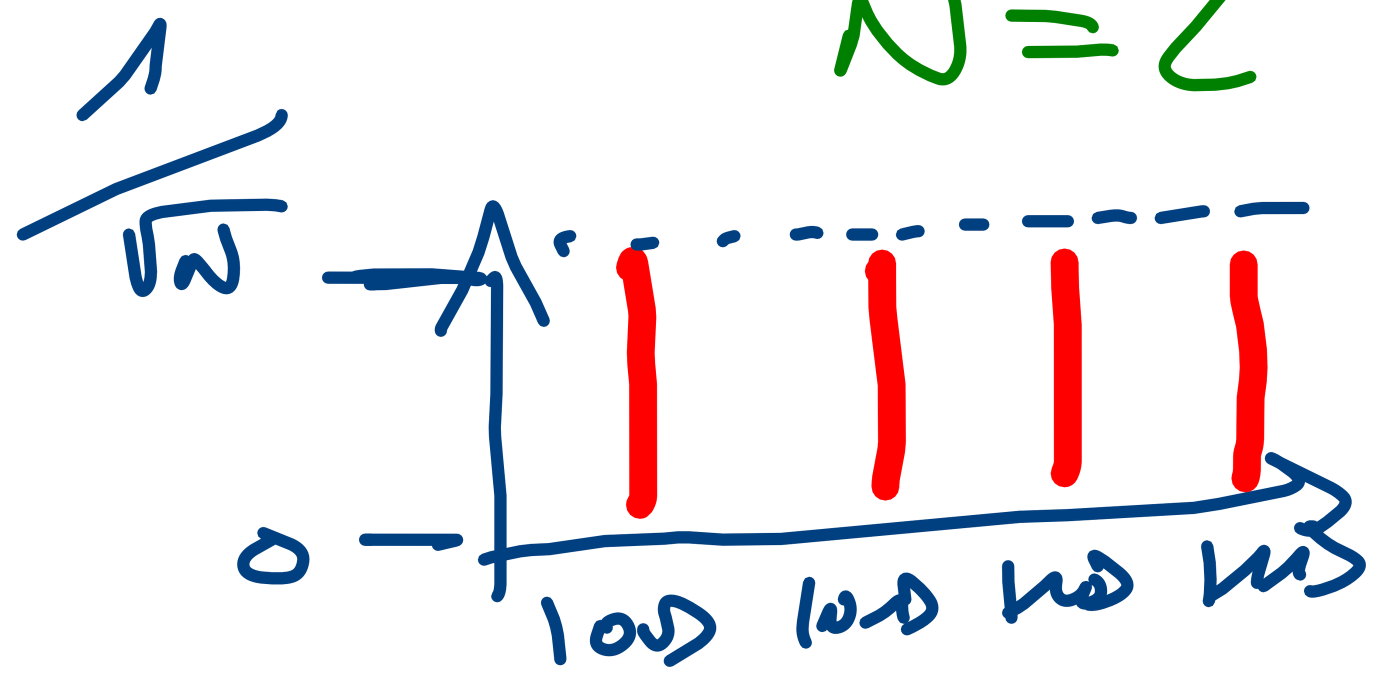
(We'll eventually solve the general case, don't worry.)

Exercise. What is the state of Z_f if we add an H gate on each of the first n input qubits?

Grover circuit



$N = 2^n$

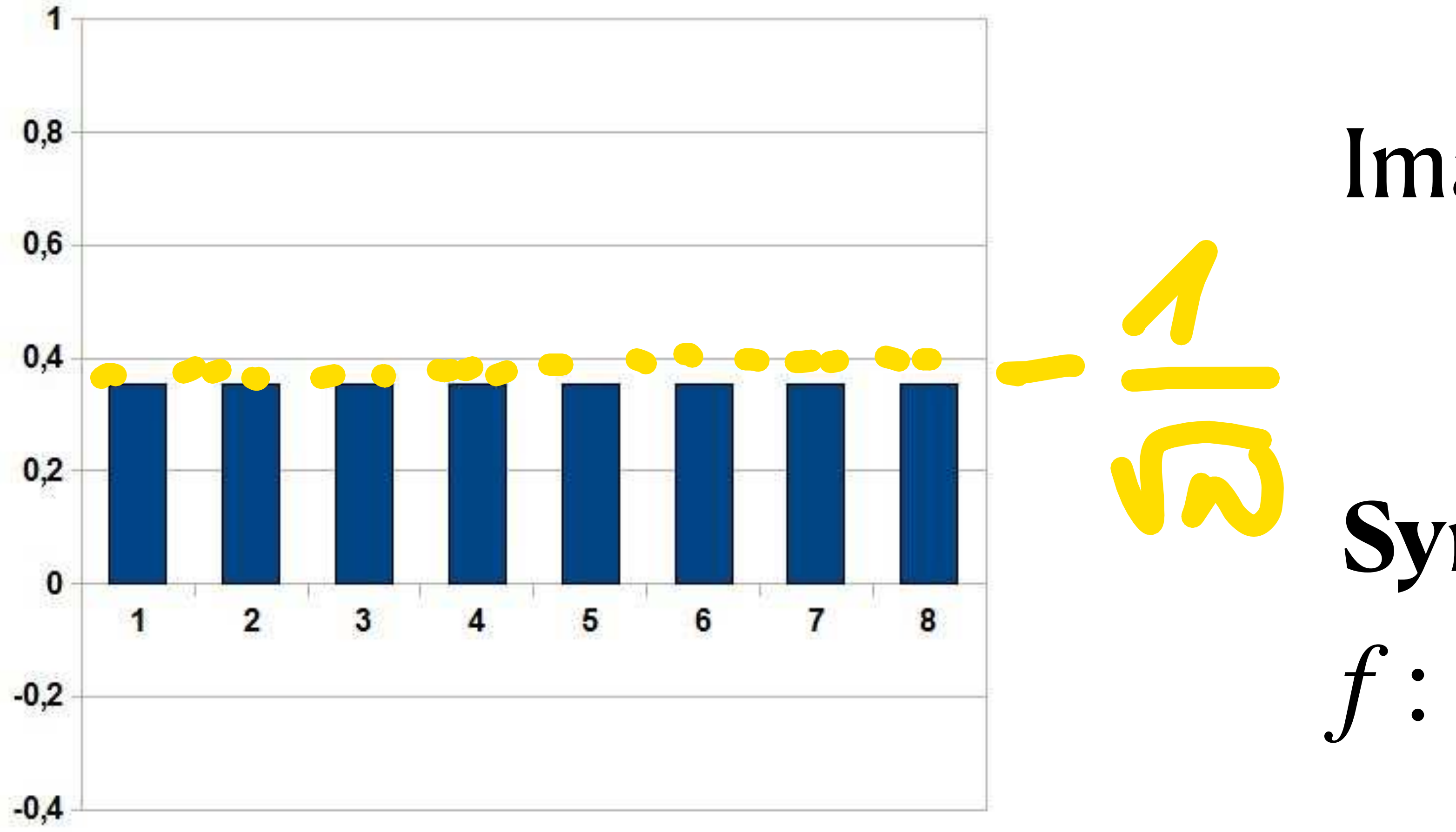


$$\underbrace{\frac{1}{\sqrt{N}}\left(\sum_{x\in A_0}|x\rangle - \sum_{x\in A_1}|x\rangle\right)}_{*\ \sum_{x=0}^{N-1}(-1)^{f(x)}|x\rangle}$$

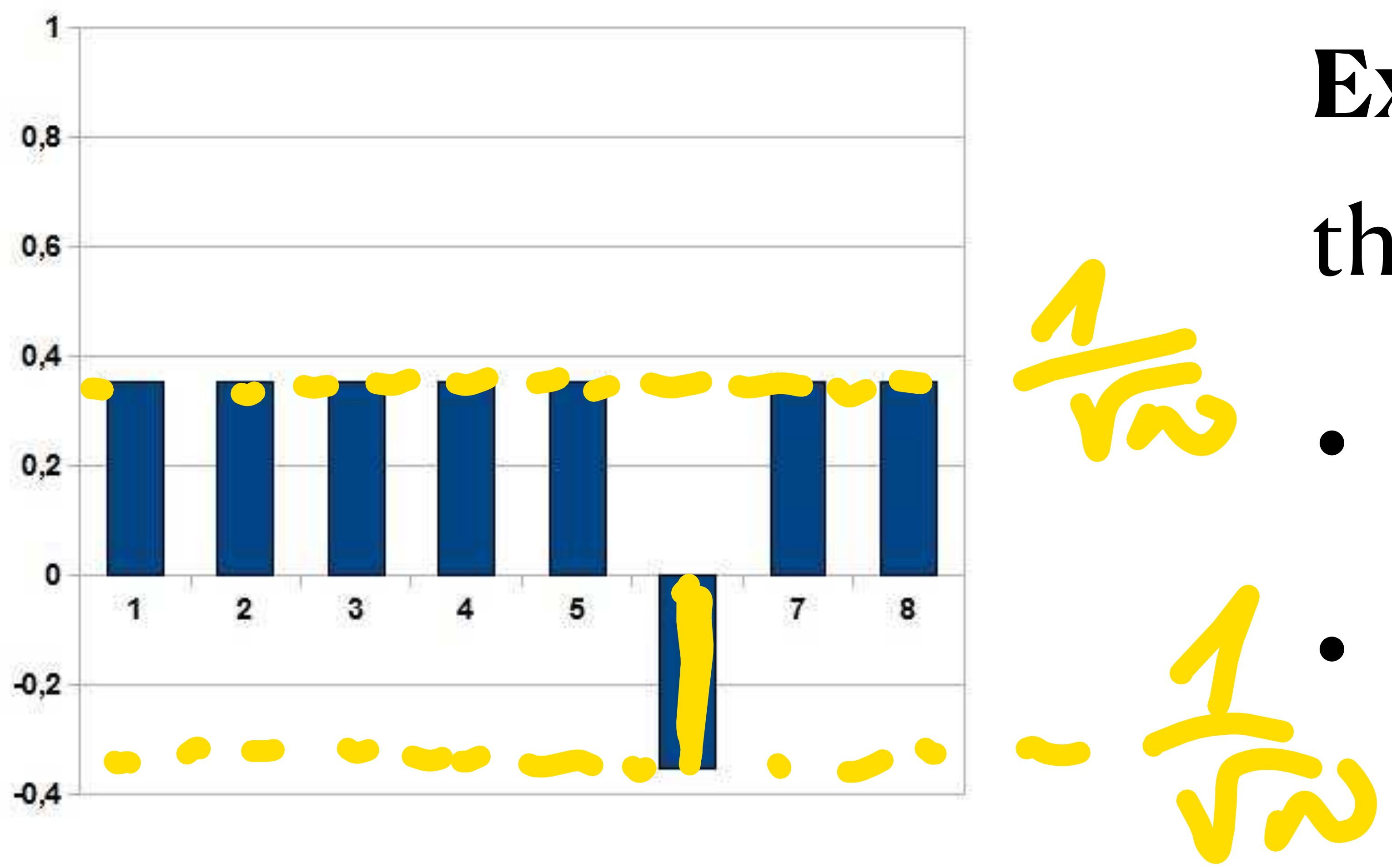
$|+\rangle^{\otimes n}|-\rangle$

$$|\psi\rangle = |+\rangle^{\otimes n} = \frac{1}{\sqrt{N}}\sum_{x=0}^{N-1}|x\rangle$$

4. Algorithme de Grover — première observation



Initial state after the H gates



Amplitudes of the output state

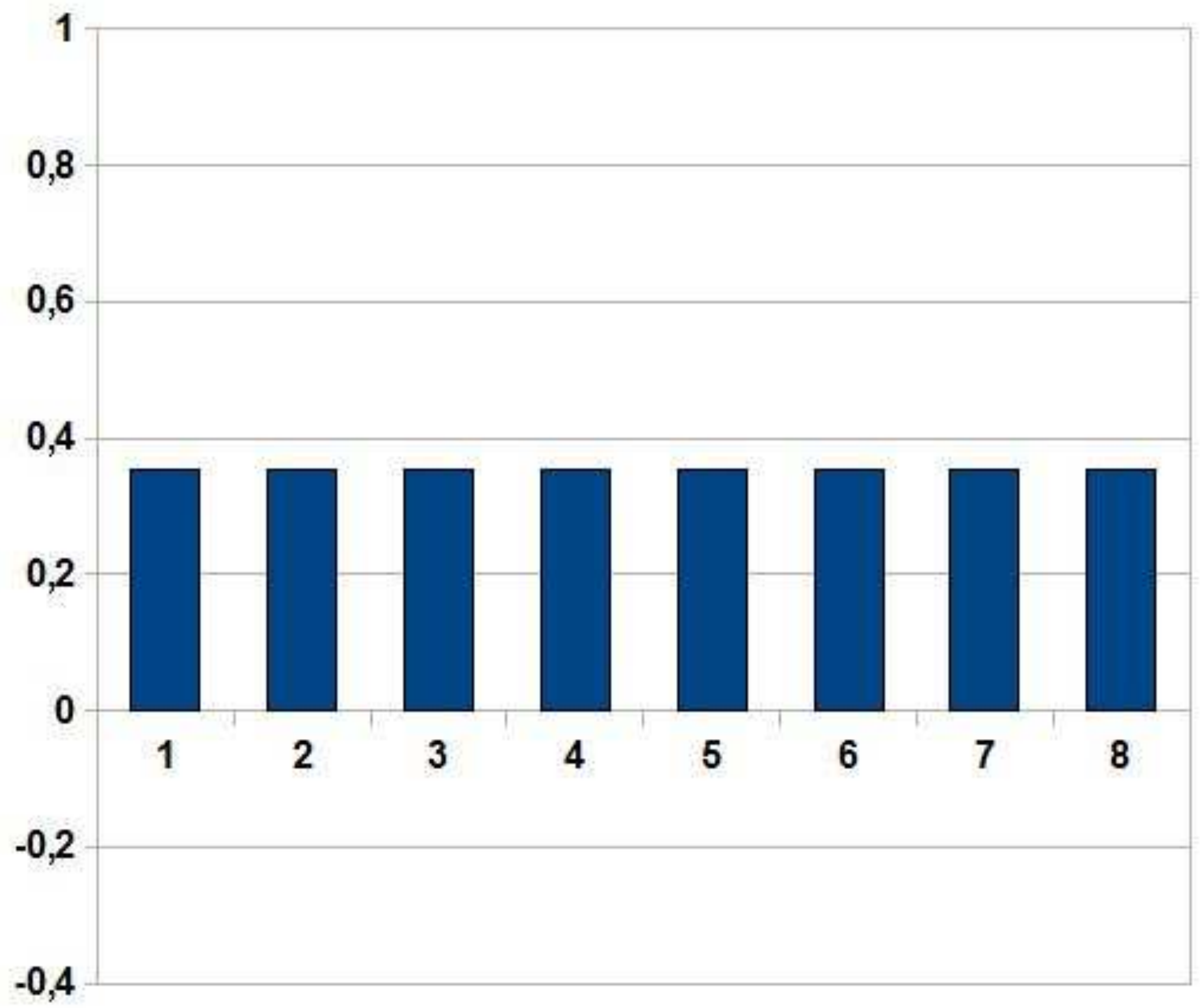
Images : https://fr.wikipedia.org/wiki/Algorithme_de_Grover

Symplifying hypothesis (reminder). For our function $f : \{0,1\}^n \to \{0,1\}$there exists a unique x_1 such that $f(x_1) = 1$.

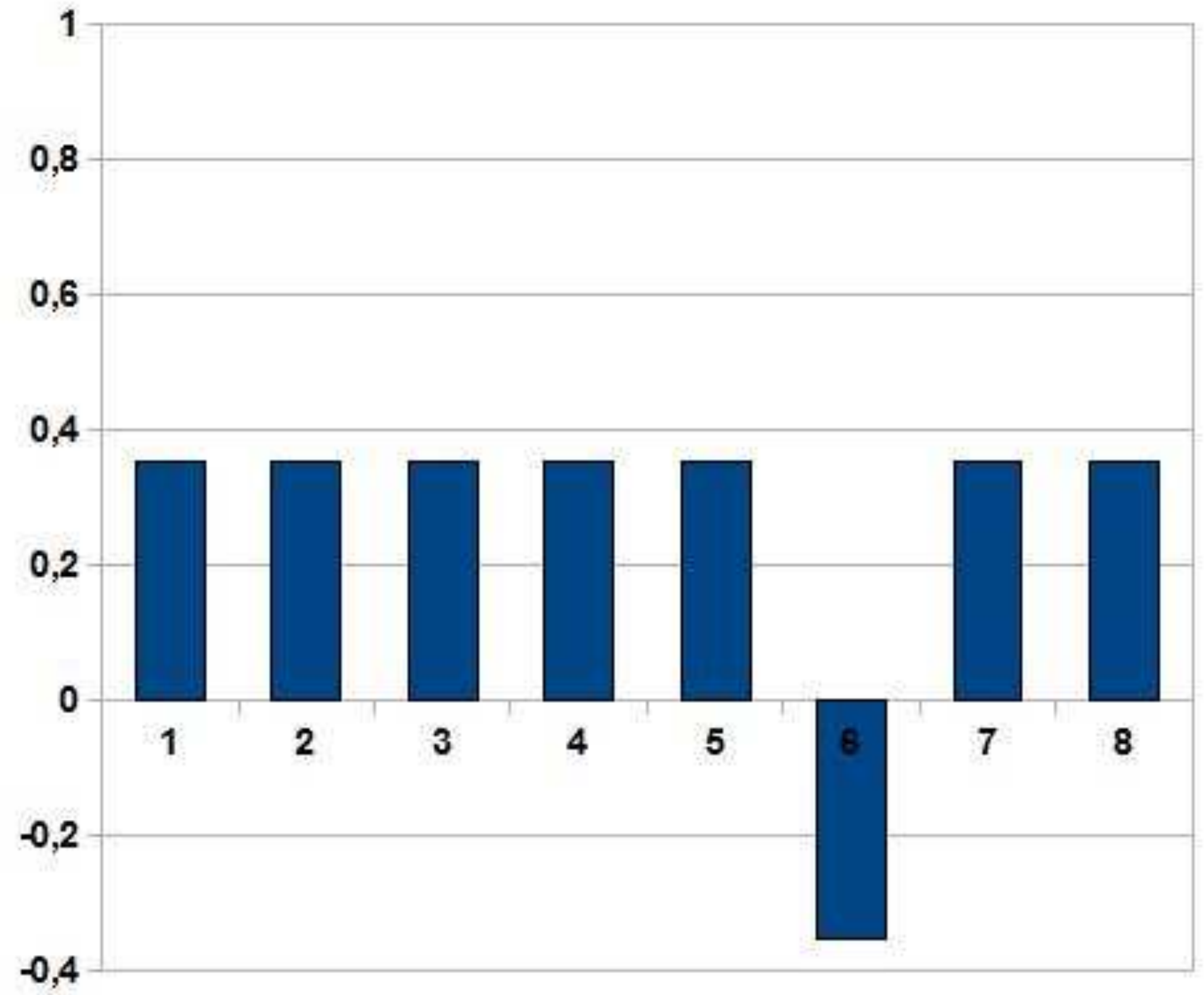
Exercise. What is the state of Z_f if we add an H gate on each of the first n input qubits?

- Up: the state right after the H gates (amplitudes)
- Down: output state, the amplitude of x_1 has changed its sign

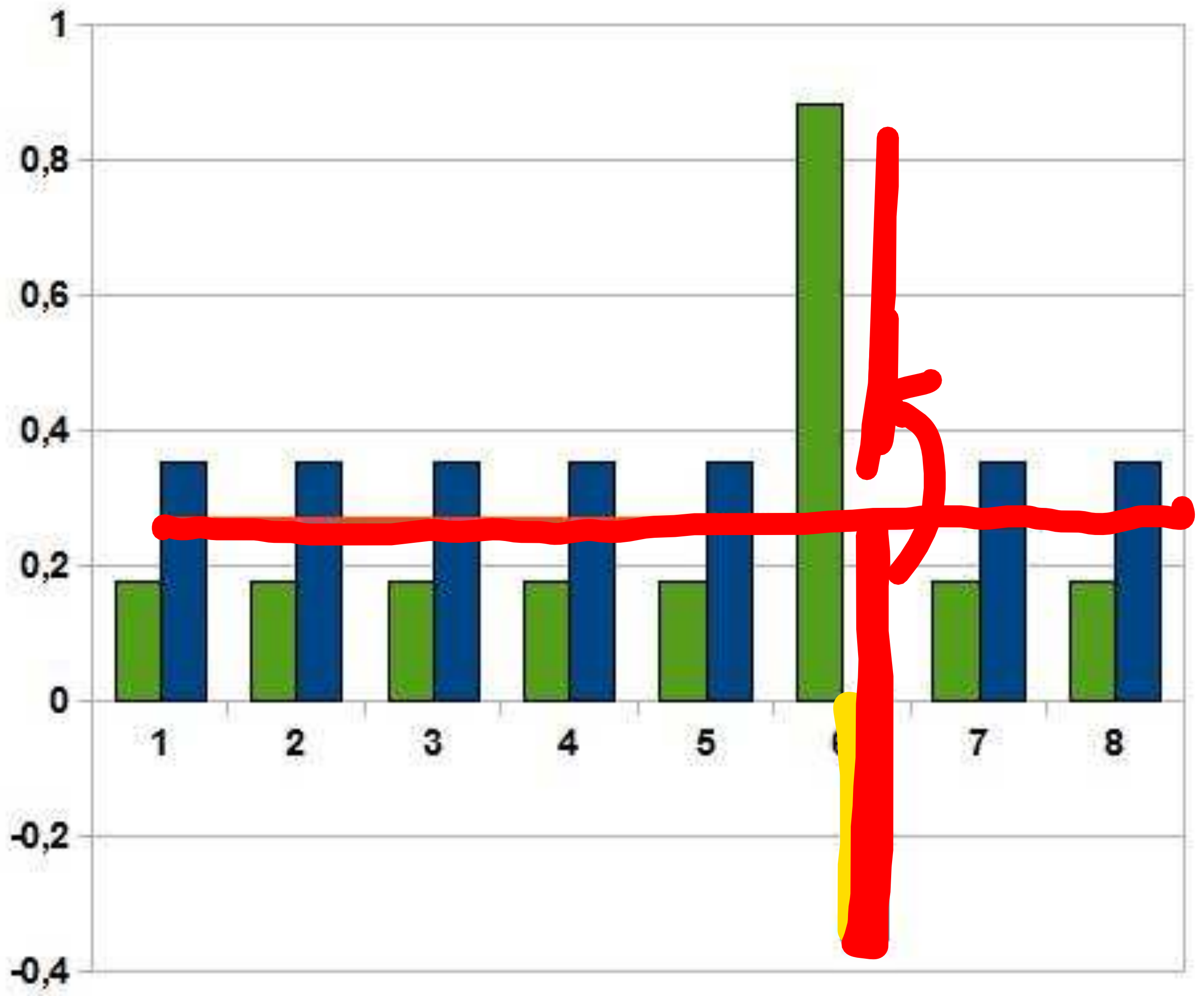
4. Grover's algorithm — symmetry w.r.t this average



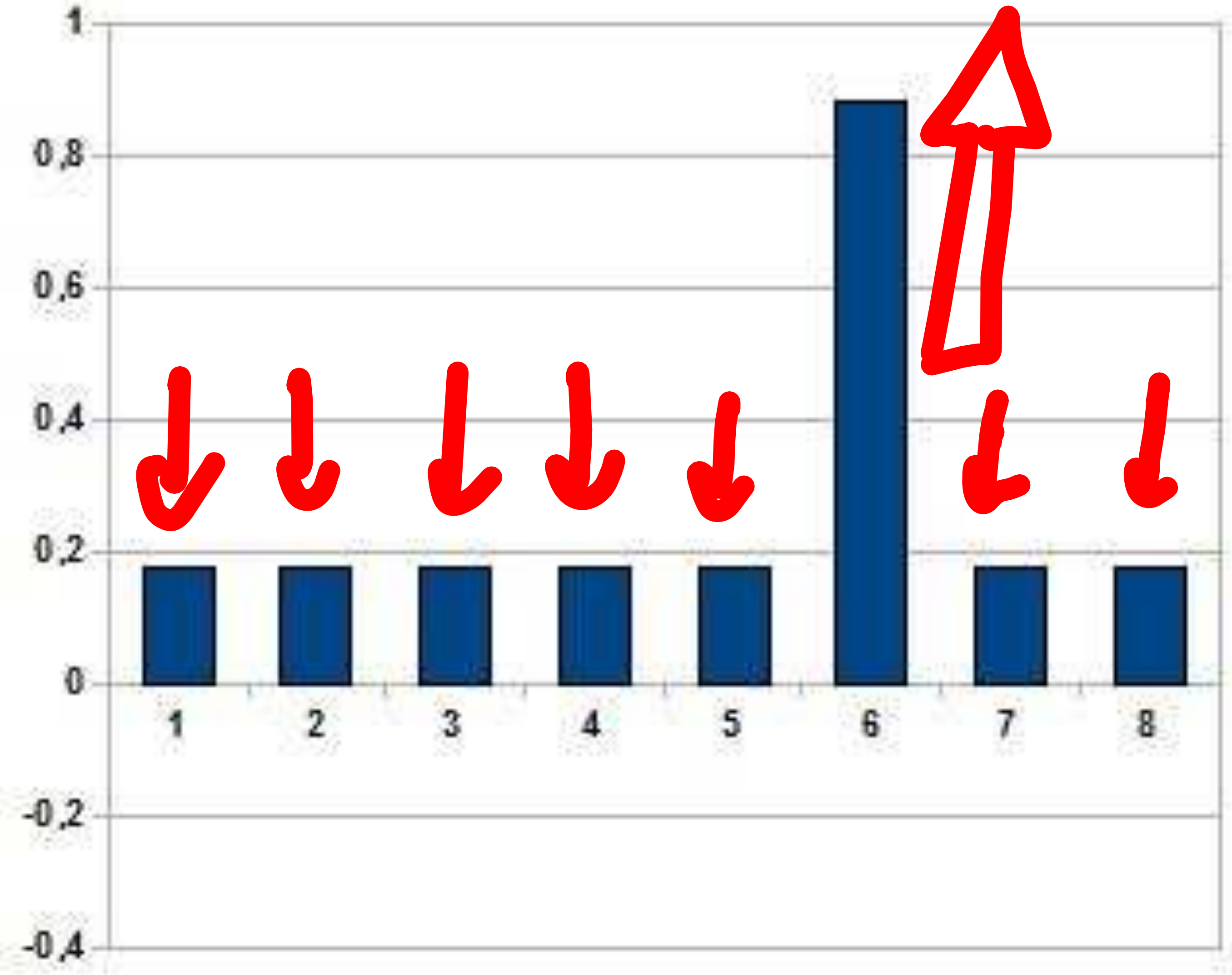
Initial state after the H gates



State after the Z_f circuit



Symmetry around the average



New state after all these steps

Grover's operator

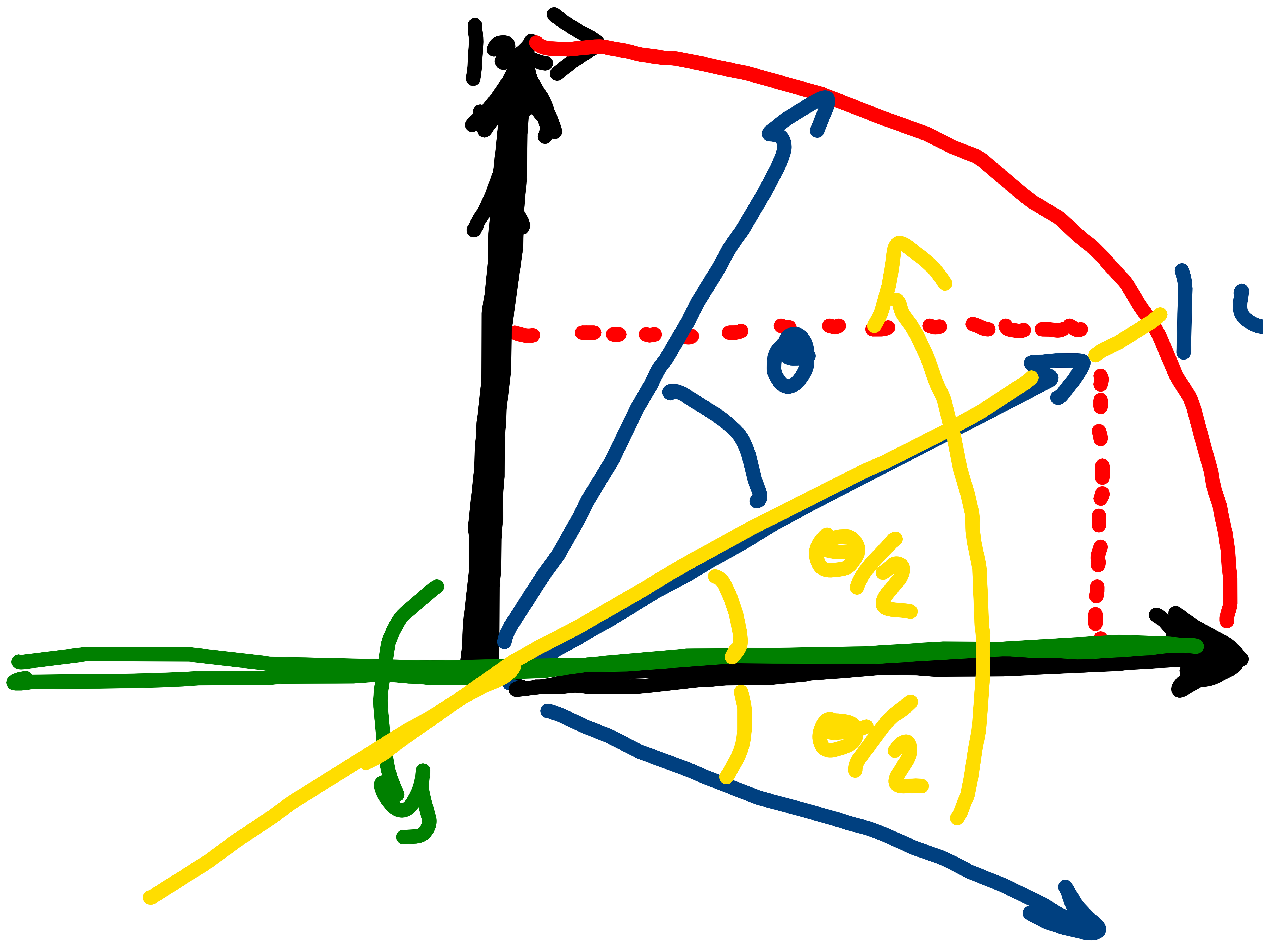
- Intuition: as if we compute the average of the amplitudes, and we apply a symmetry w.r.t this average
- We'll detail the implementation and the proofs

$|\psi\rangle : \frac{\theta}{2}$

$G|\psi\rangle : \frac{3\theta}{2}$

$\vdots$

$G^k|\psi\rangle : \frac{2k+1}{2}\theta$



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

$$\underbrace{\frac{1}{\sqrt{N-1}} \sum_{y \neq x} |y\rangle}_{|\alpha\rangle}$$

$$|\psi\rangle = \frac{1}{\sqrt{N}} |x\rangle + \sqrt{\frac{N-1}{N}}\, |\alpha\rangle$$

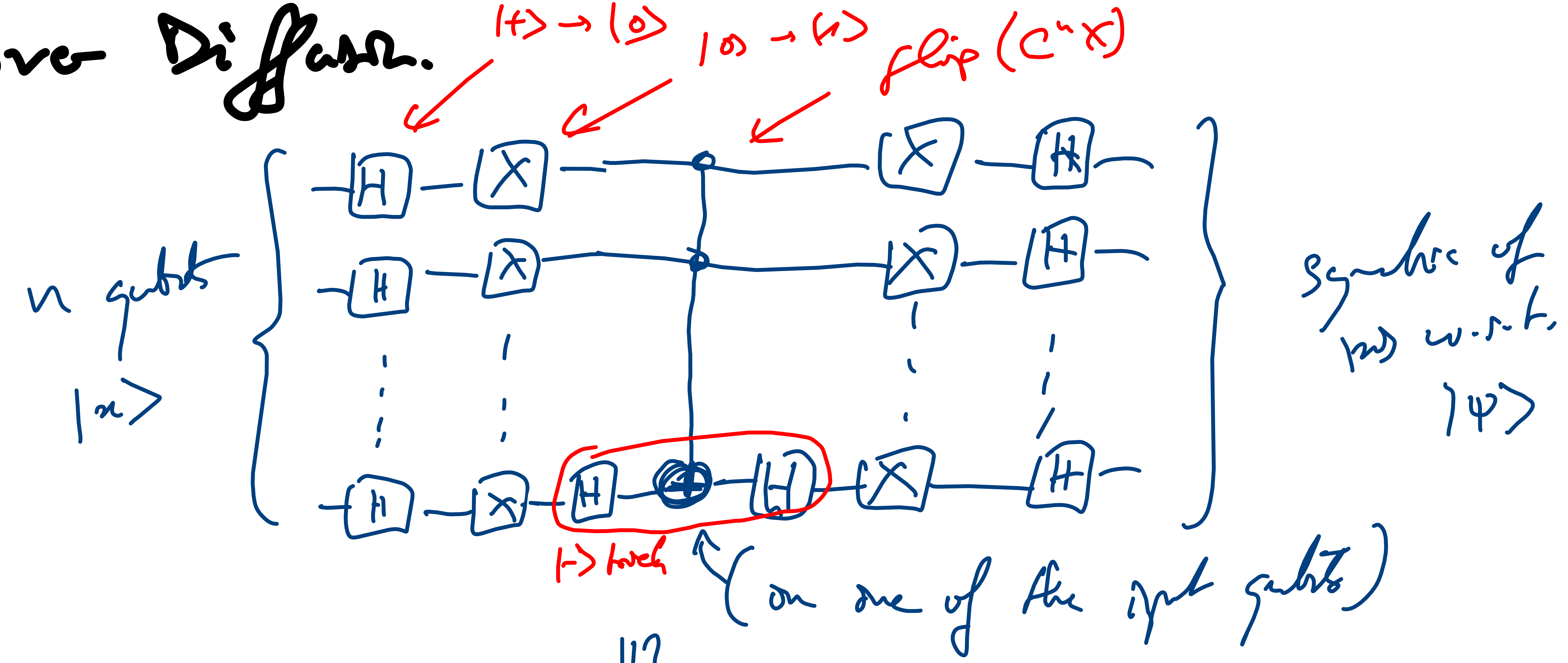
We are looking for an operator that captures symmetry w.r.t $|\psi\rangle$.

Ideas: Use Hadamard to do symmetry w.r.t $|0\rangle$

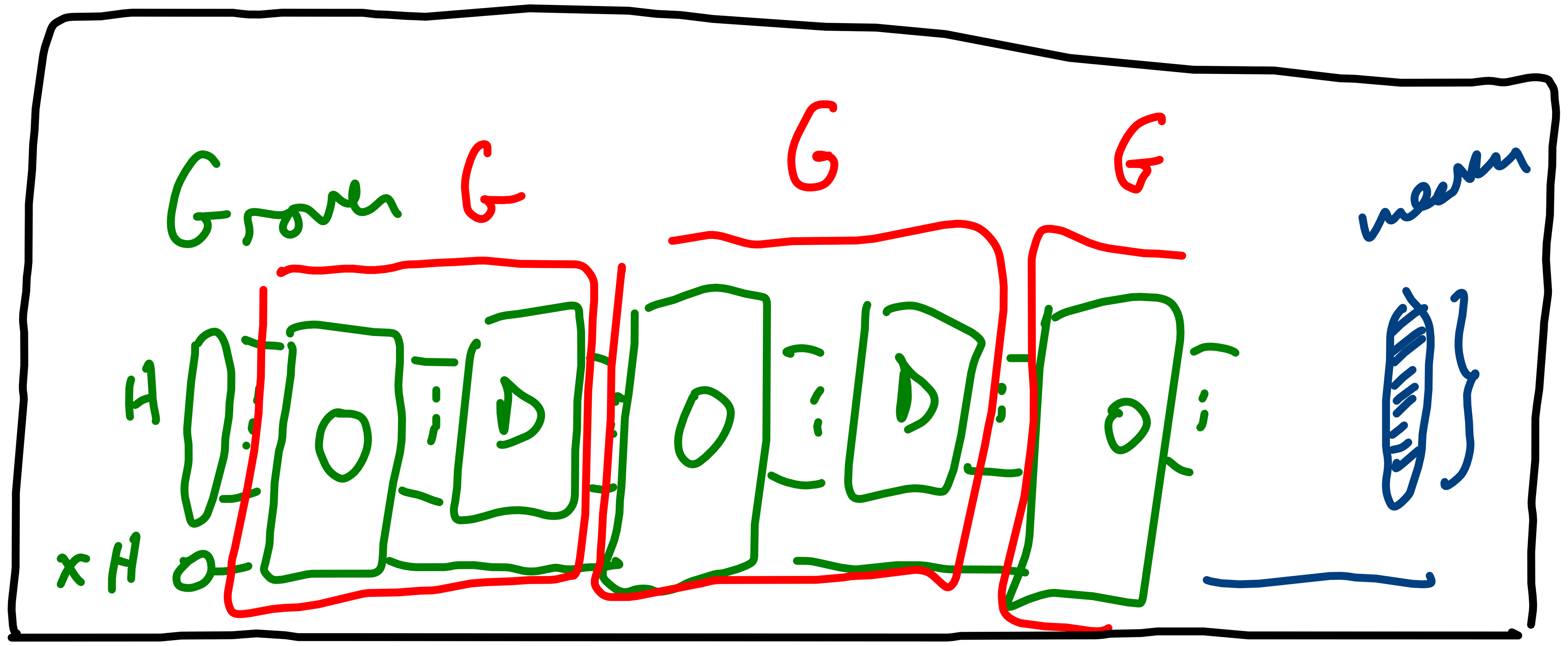
$$\boxed{I - 2|\psi\rangle\langle\psi|}$$

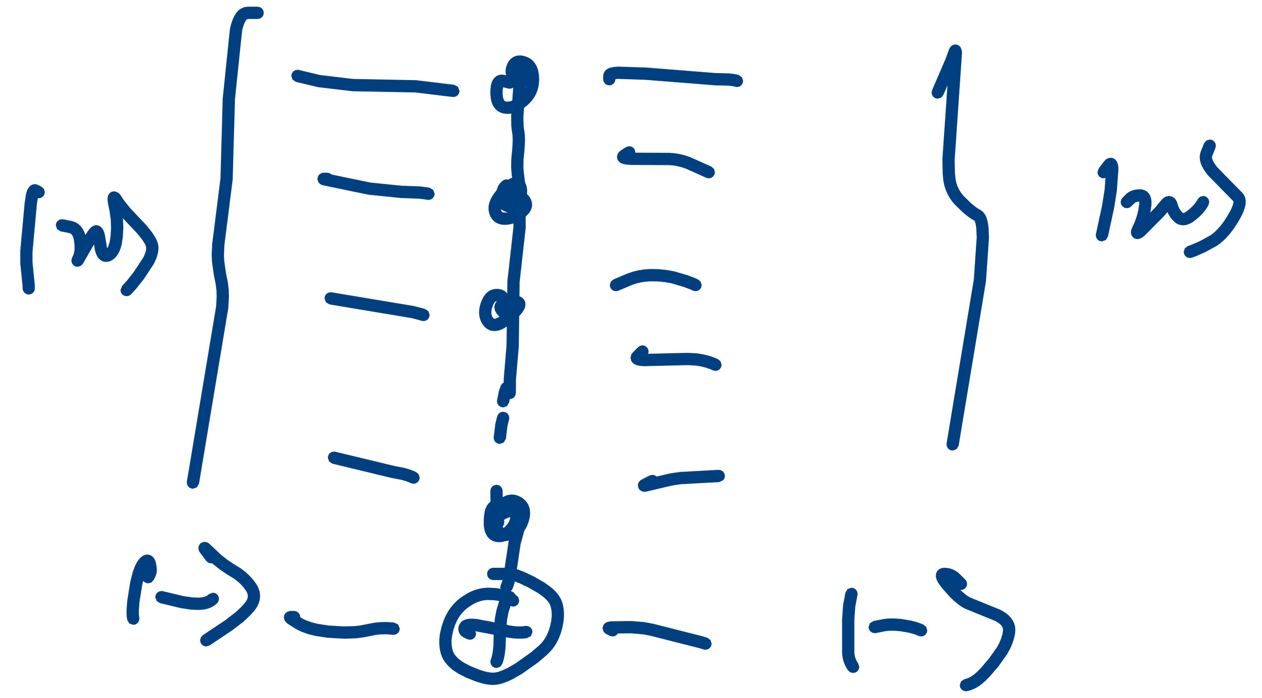
Ideaz: C^nX flips on a single value $|1\rangle^{\otimes n}$

Grover Diffusion.



≡?





~~phase amplitude~~

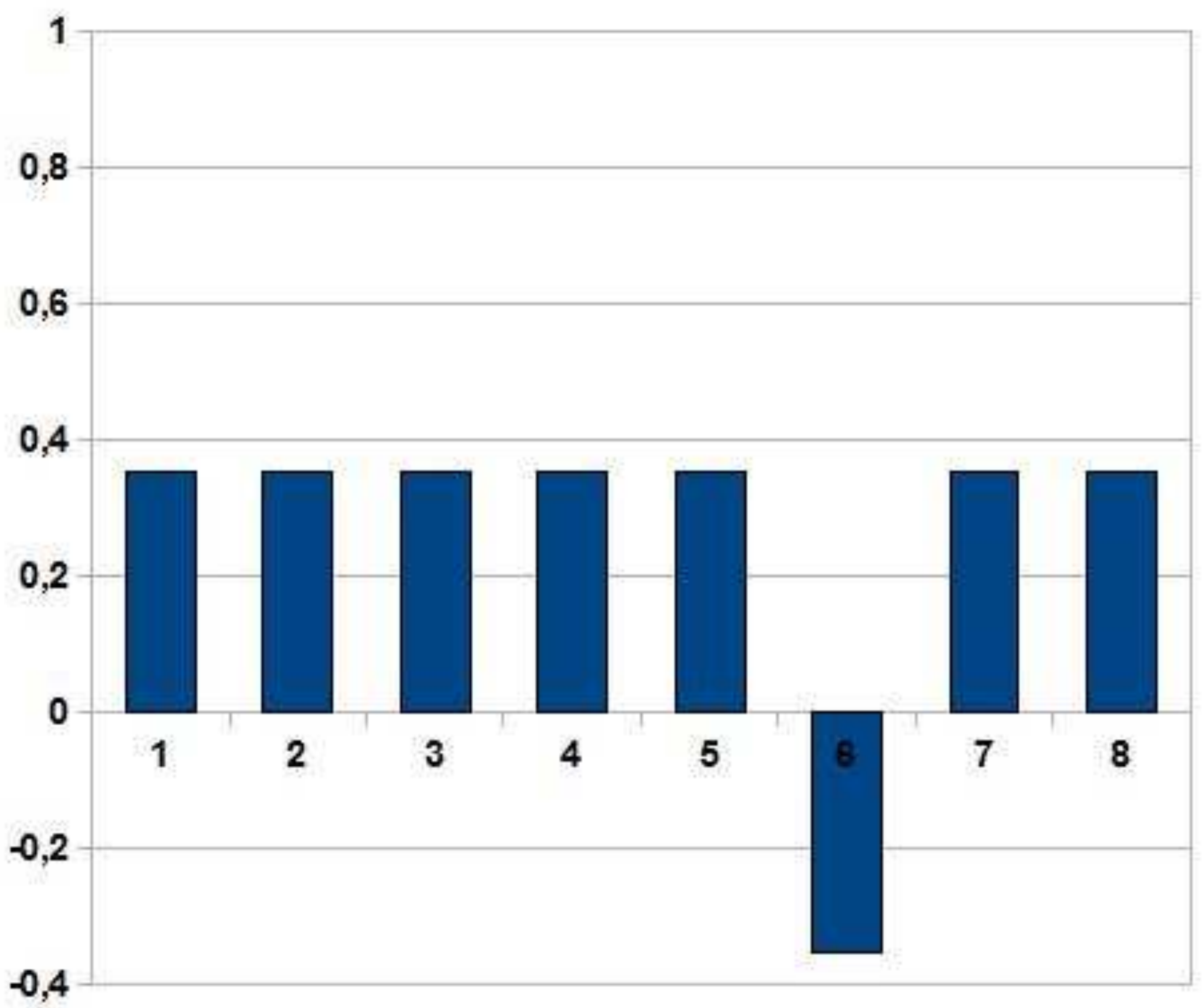
4. Grover's algorithm

— repeate

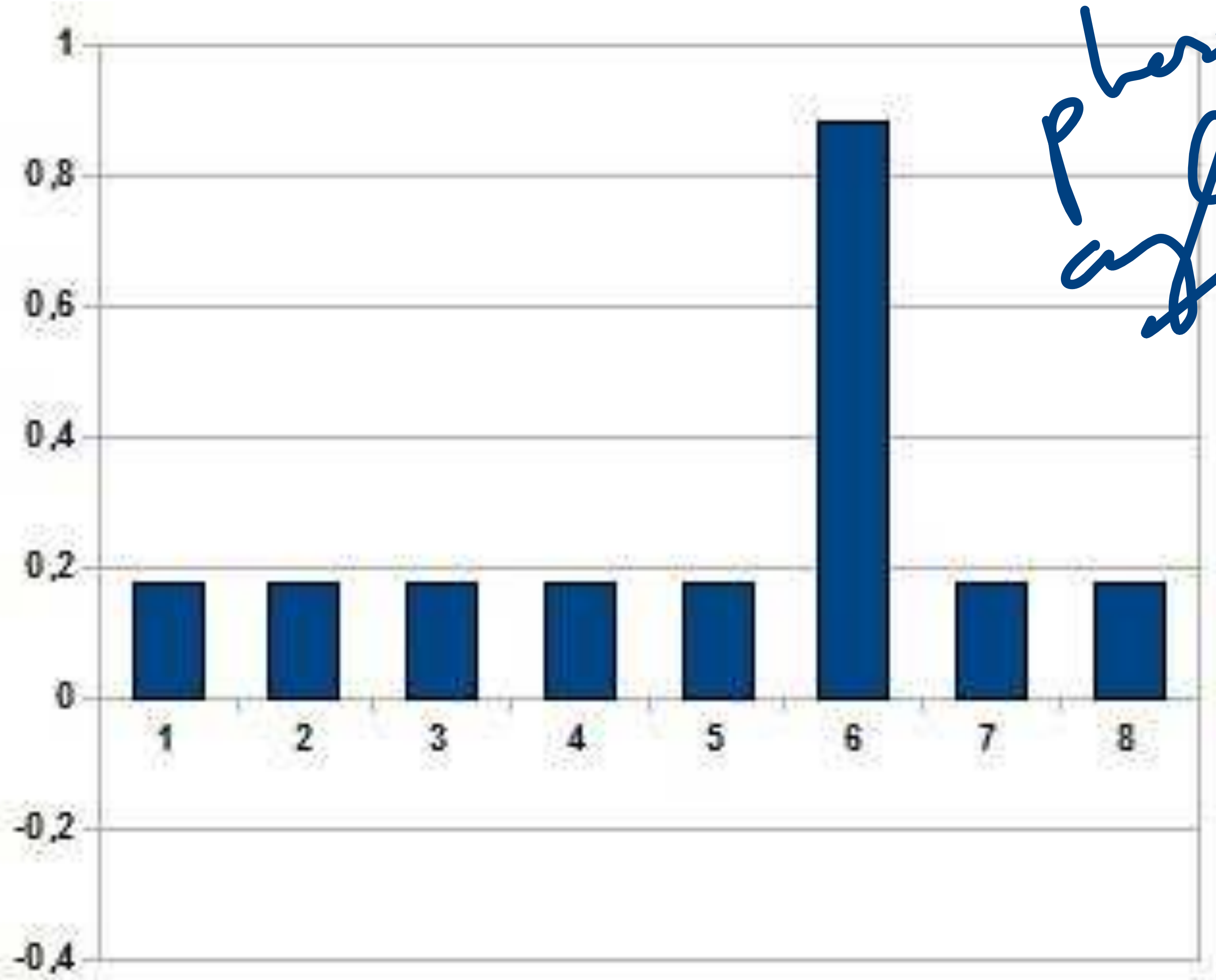
how many?

... a well-chose number of times

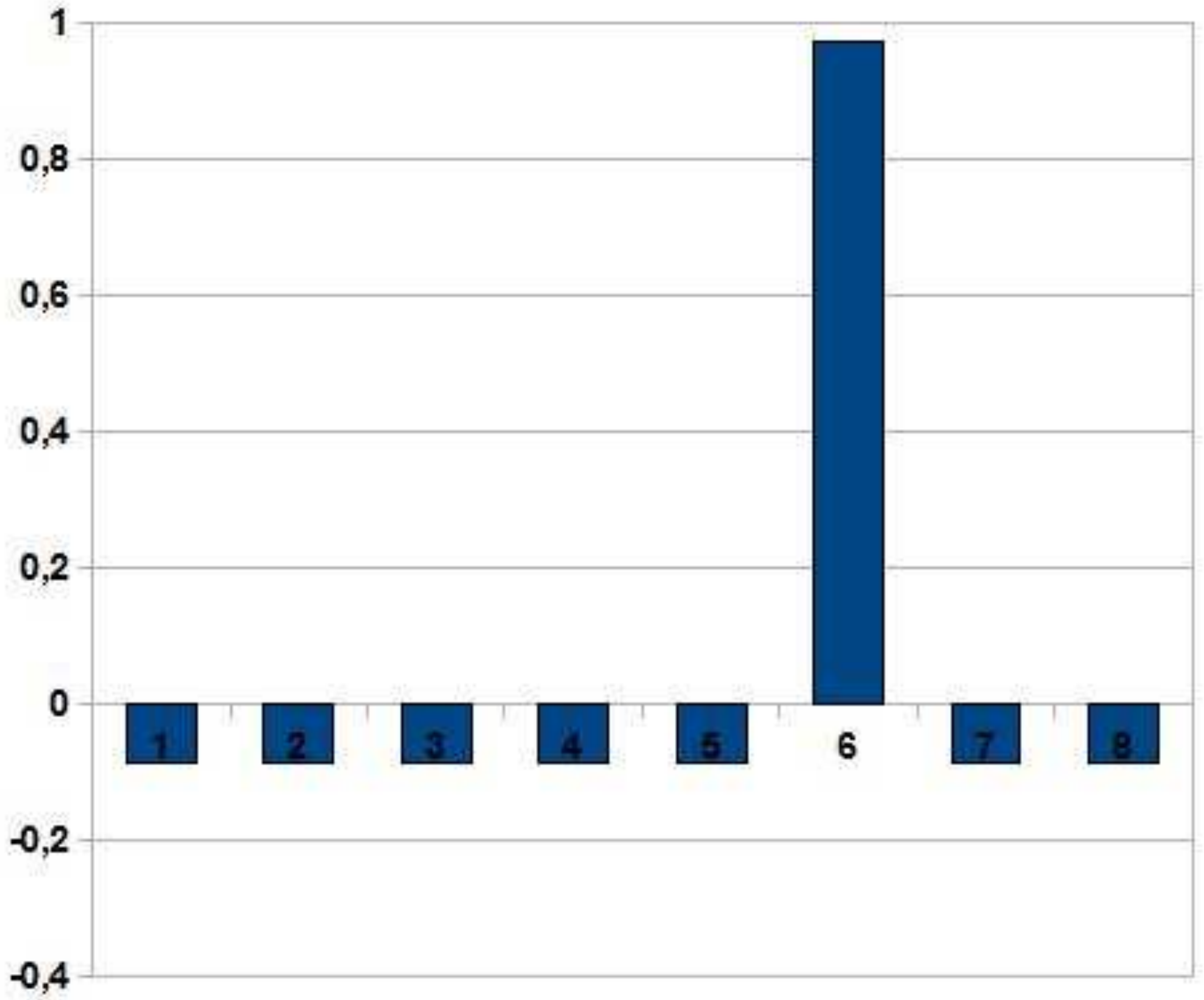
- In the example we measure x_1 with probability larger than 90 %



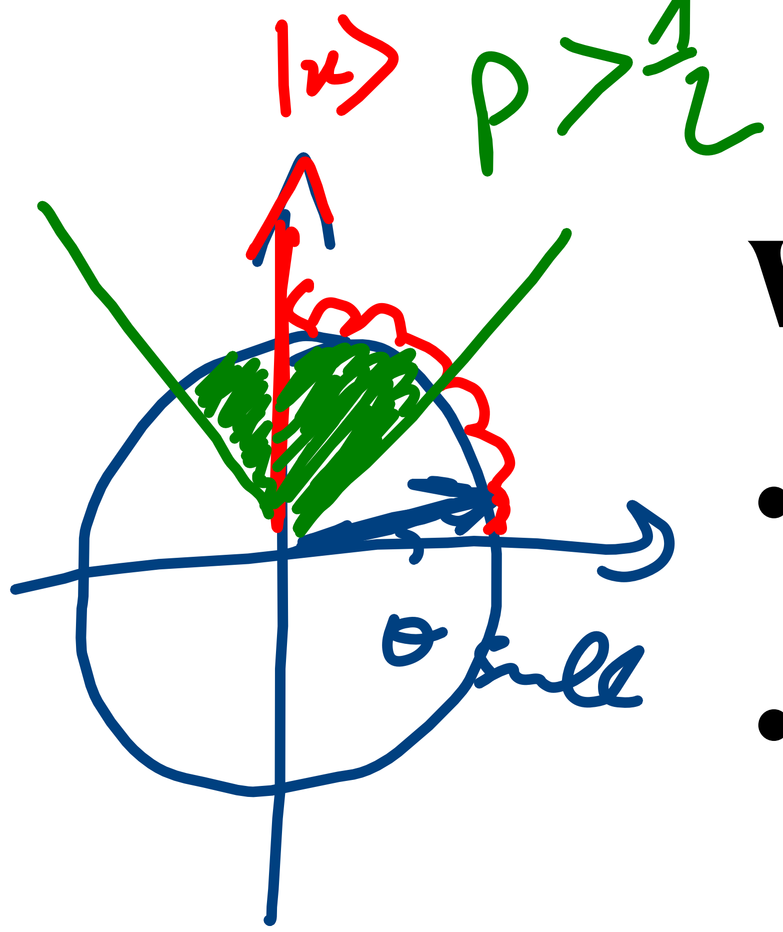
State after H gates and one Z_f



State after one Grover operator



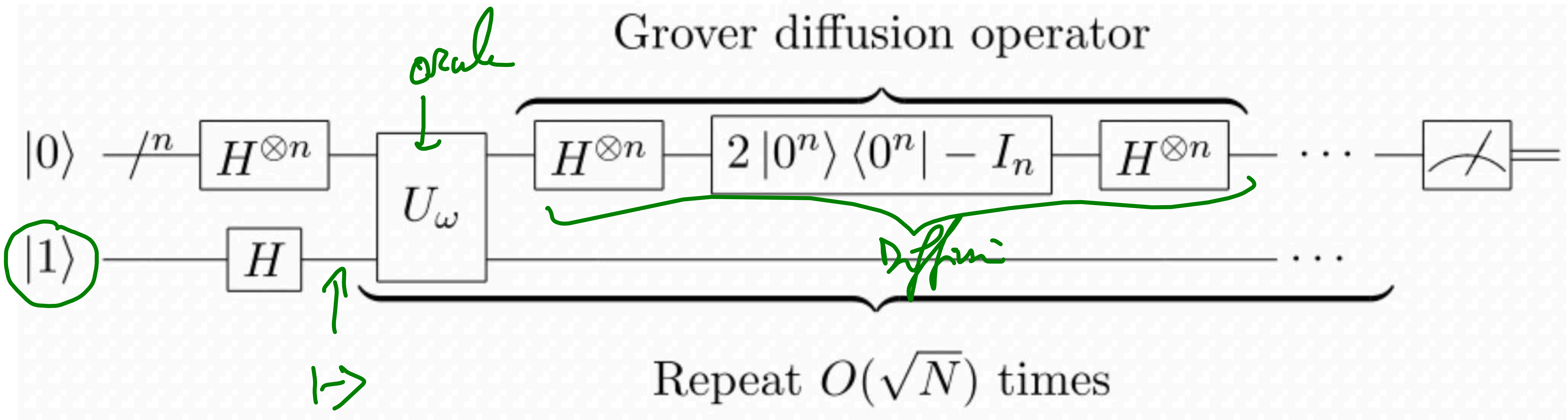
State after two Grover operator



We still need to:

- Detail the implementation
- Transform this small example into an actual proof

4. Grover's algorithm



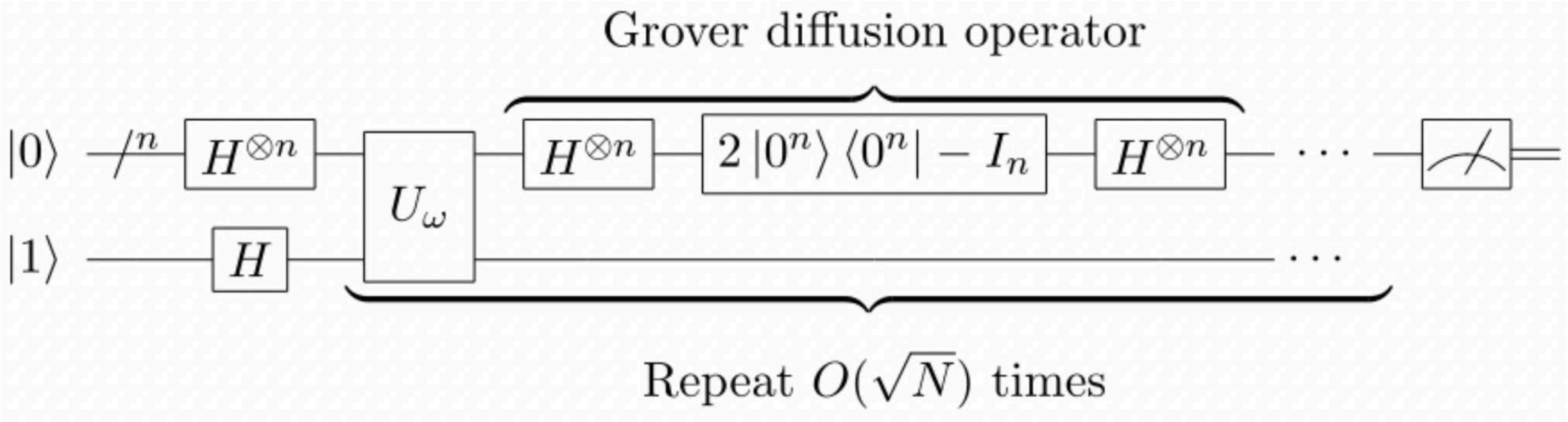
https://fr.wikipedia.org/wiki/Algorithme_de_Grover

General scheme. Here U_ω is just another notation for U_f.

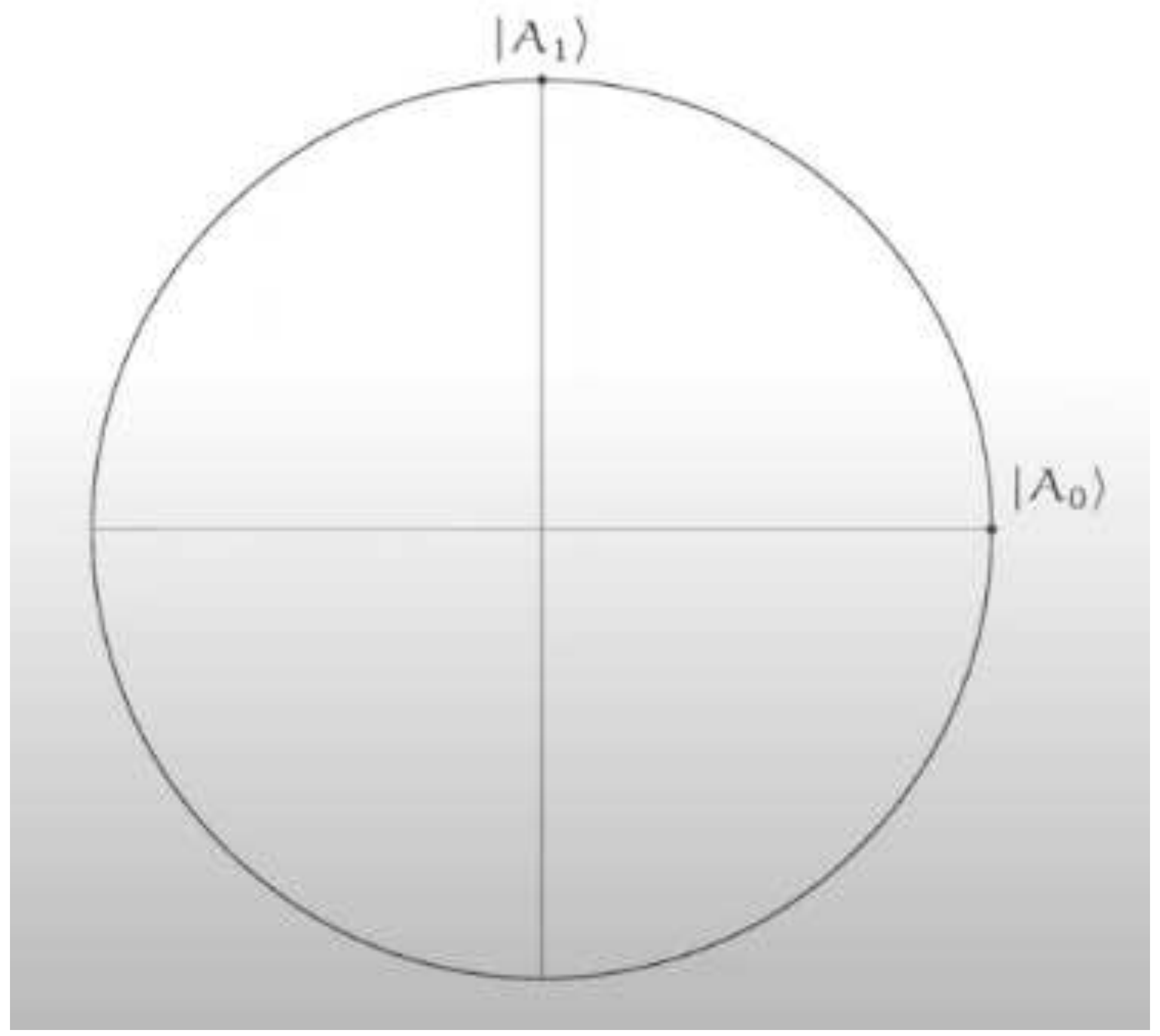
Grover's diffusion operator performs this "mirror around the average". Its implementation is fairly easy:

$$H^{\oplus n} Z_{OR} H^{\oplus n}$$

4. Grover's algorithm : the proof



https://fr.wikipedia.org/wiki/Algorithme_de_Grover



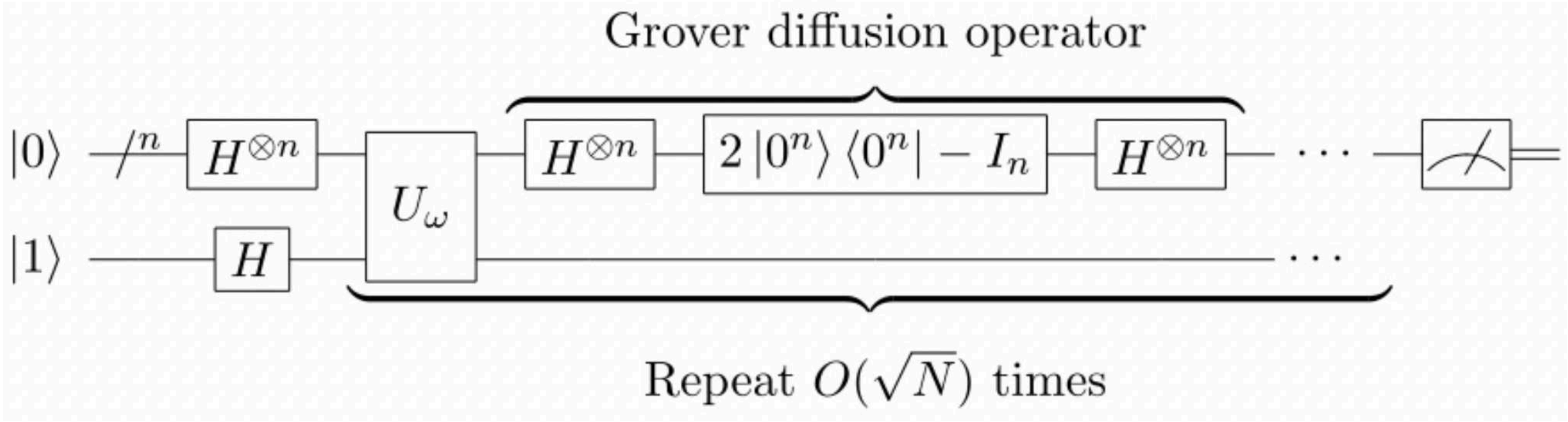
John Watrous, IBM, YouTube video

We denote $A_1 = \{x_1\}$ and
$A_0 = \{x \in \{0,1\}^n : f(x) = 0\}$

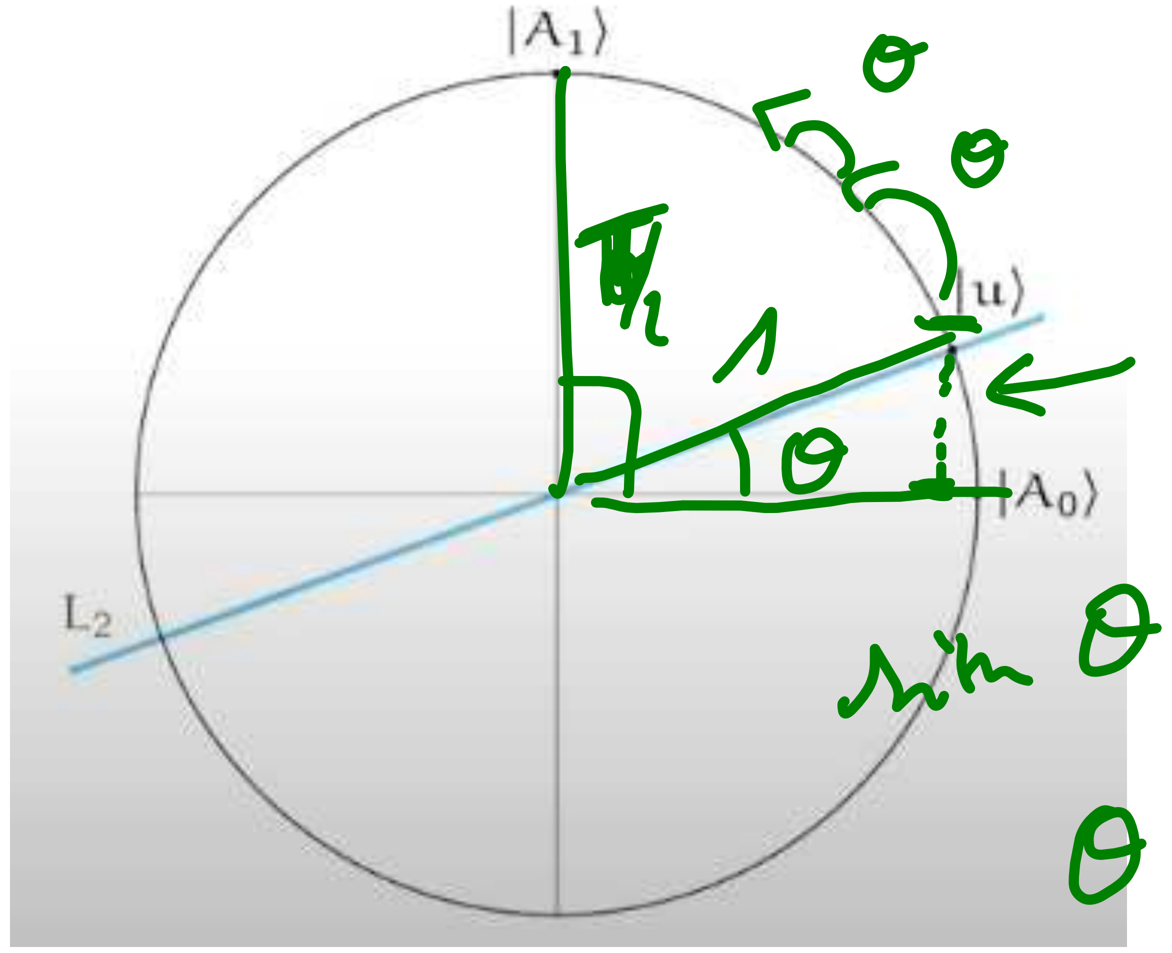
For any set of boolean vectors A of size n let
$$|A\rangle = \frac{1}{\sqrt{|A|}} \sum_{x \in A} |x\rangle.$$

Observe that vectors $|A_0\rangle$ and $|A_1\rangle$ are orthogonals.

4. Grover's algorithm : the proof



https://fr.wikipedia.org/wiki/Algorithme_de_Grover



θ θ $\pi/2$ 1 θ $\sqrt{1/N}$

$\sin\theta = \sqrt{1/N}$

$\theta \approx \sqrt{1/N}$

John Watrous, IBM

Let $|u\rangle = H^{\otimes n}|0^n\rangle$, the "uniform superposition" vector

$$\frac{2k+1}{2}\theta = \frac{\pi}{2}$$

with $\theta = \sqrt{1/N}$

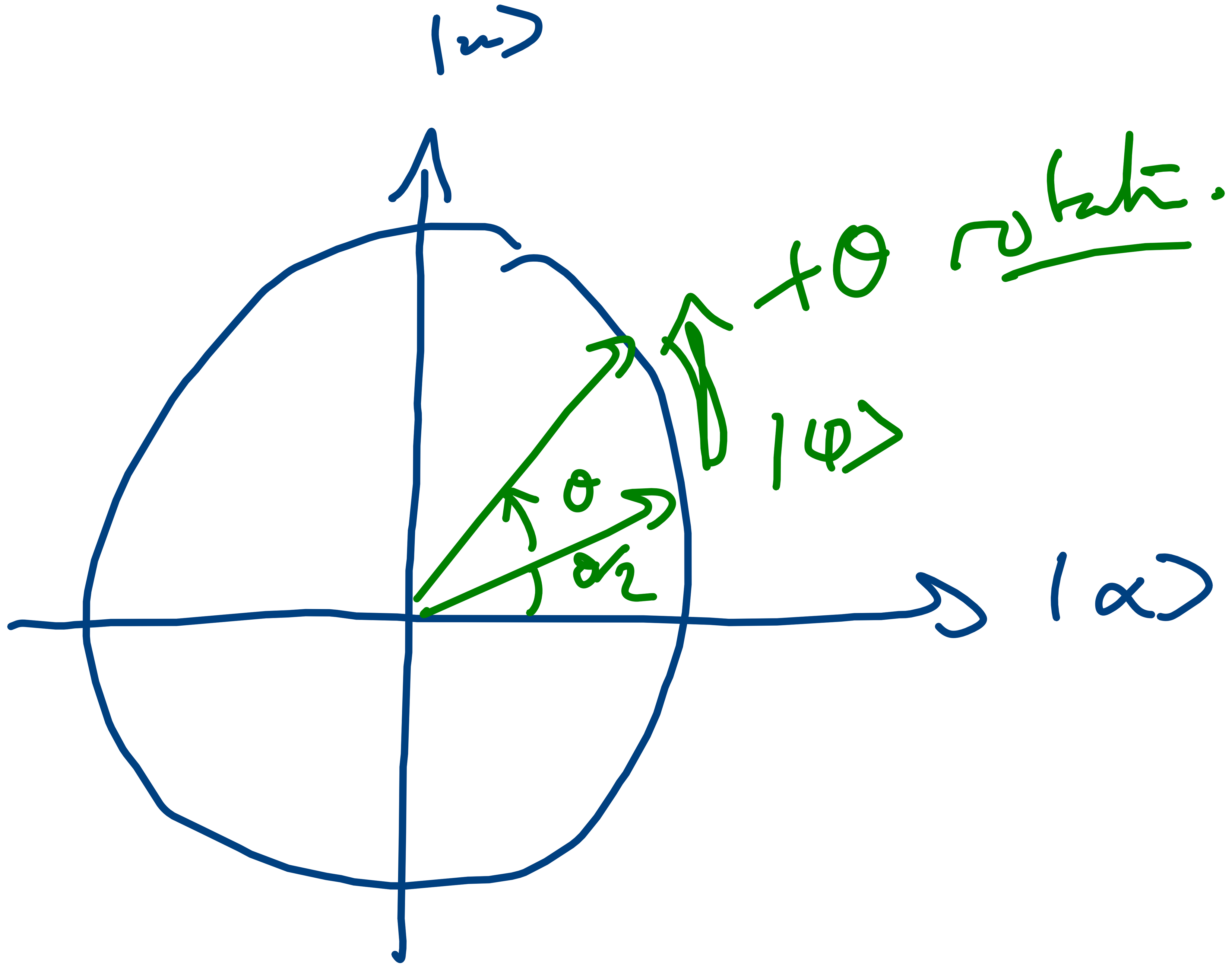
so $\boxed{k \sim \frac{\pi}{4}\sqrt{N}}$

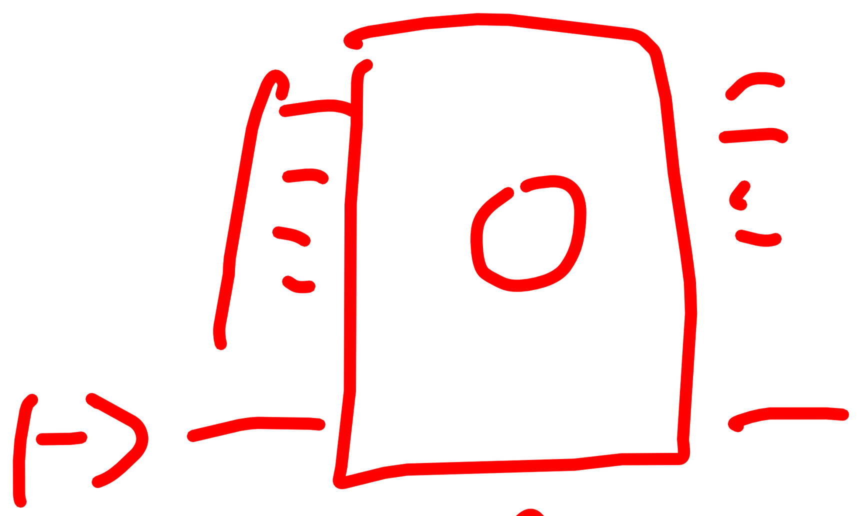
$$|u\rangle = \frac{1}{\sqrt{N}} \sum_{x\in\{0,1\}^n} |x\rangle$$

$$|u\rangle = \frac{1}{\sqrt{N}}\Big(\sum_{x_0\in A_0} |x_0\rangle + \sum_{x_1\in A_1} |x_1\rangle\Big)$$

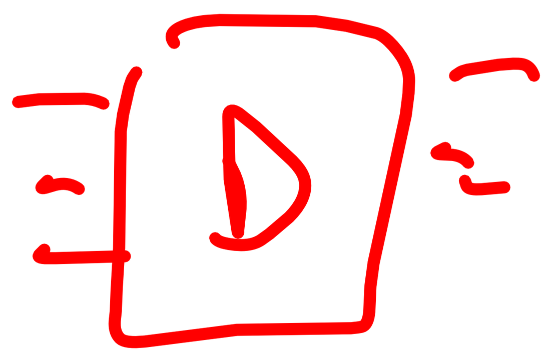
$$|u\rangle = \frac{\sqrt{|A_0|}}{\sqrt{N}}|A_0\rangle + \frac{\sqrt{|A_1|}}{\sqrt{N}}|A_1\rangle$$

$|u\rangle$ is a lainera combination of $|A_0\rangle$ and $|A_1\rangle$





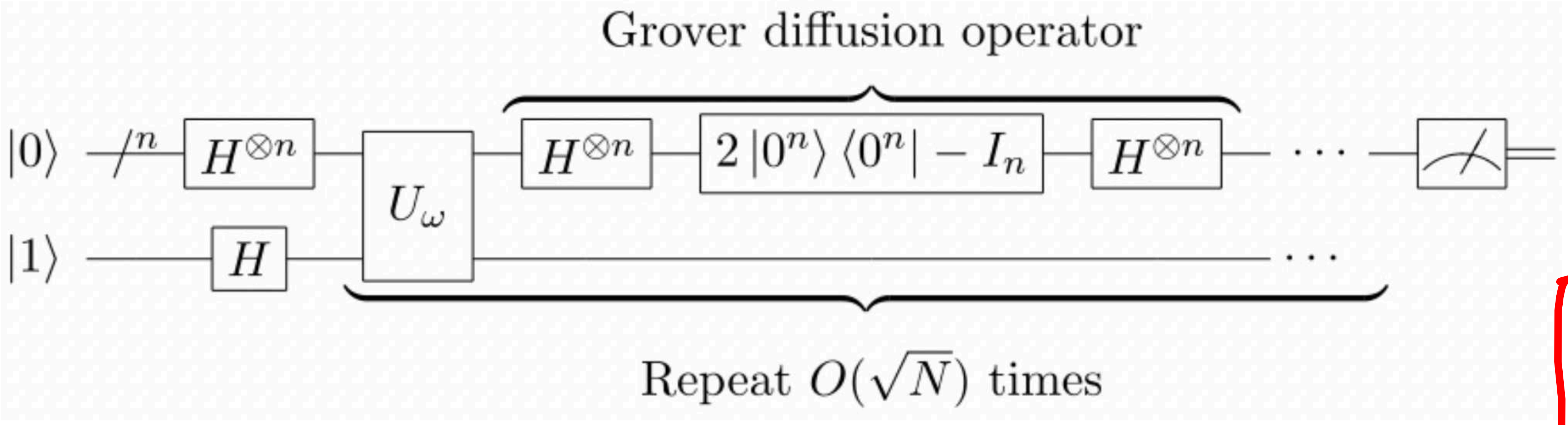
$I - 2|w\rangle\langle w|$



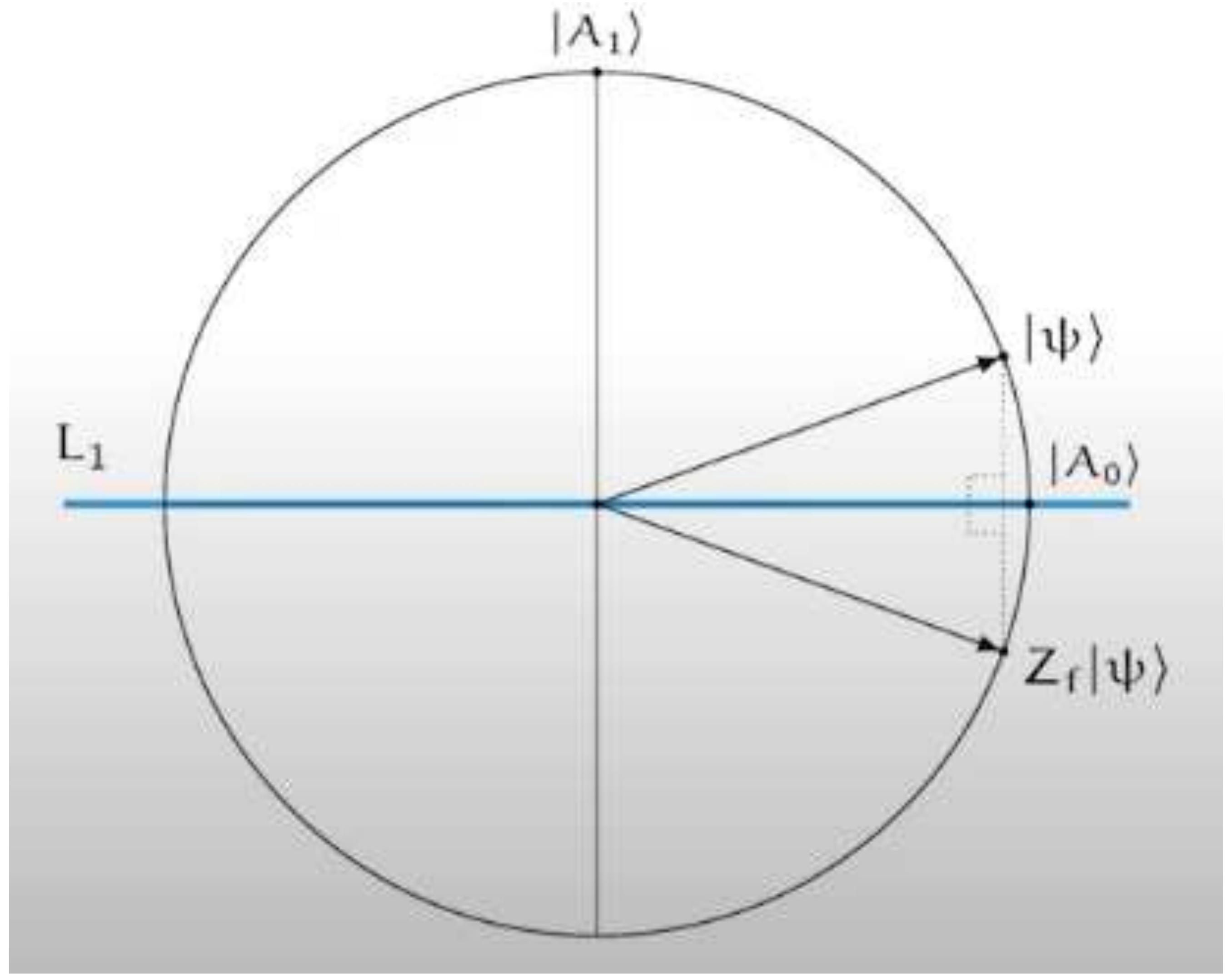
$2|\psi\rangle\langle\psi| - I$

rotate of θ

4. Grover's algorithm : the proof



https://fr.wikipedia.org/wiki/Algorithme_de_Grover



John Watrous, IBM

Understanding $Z_f|\psi\rangle$: symmetry around $|A_0\rangle$.

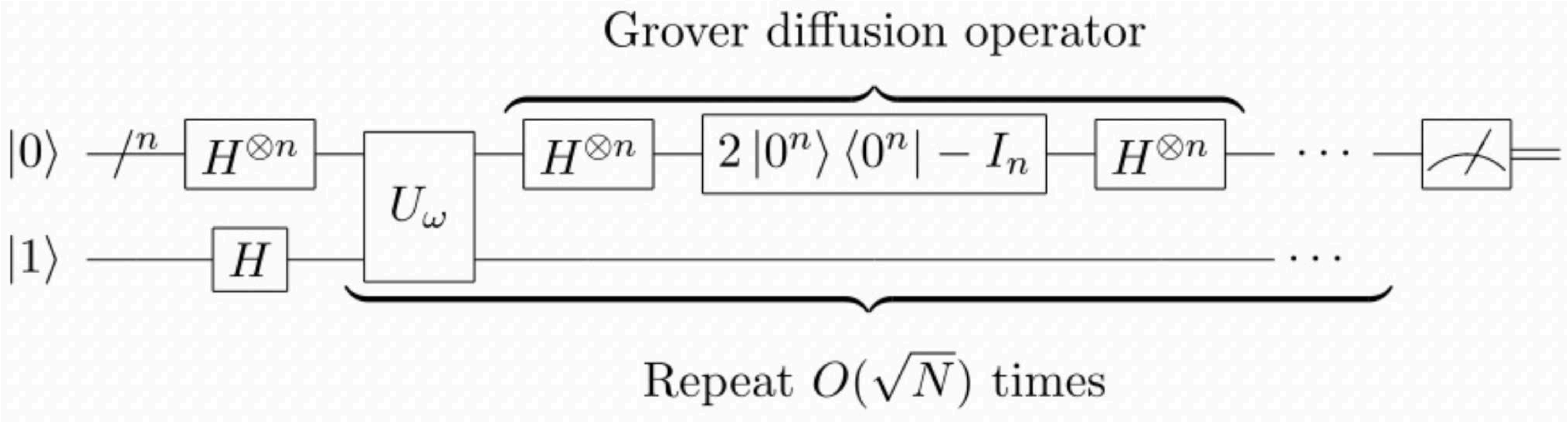
$$\boxed{|\psi\rangle = \sum_{x\in\{0,1\}^n} \alpha_x |x\rangle}$$

$$|\psi\rangle = \sum_{x_0\in A_0} \alpha_{x_0}|x_0\rangle + \sum_{x_1\in A_1} \alpha_{x_1}|x_1\rangle$$

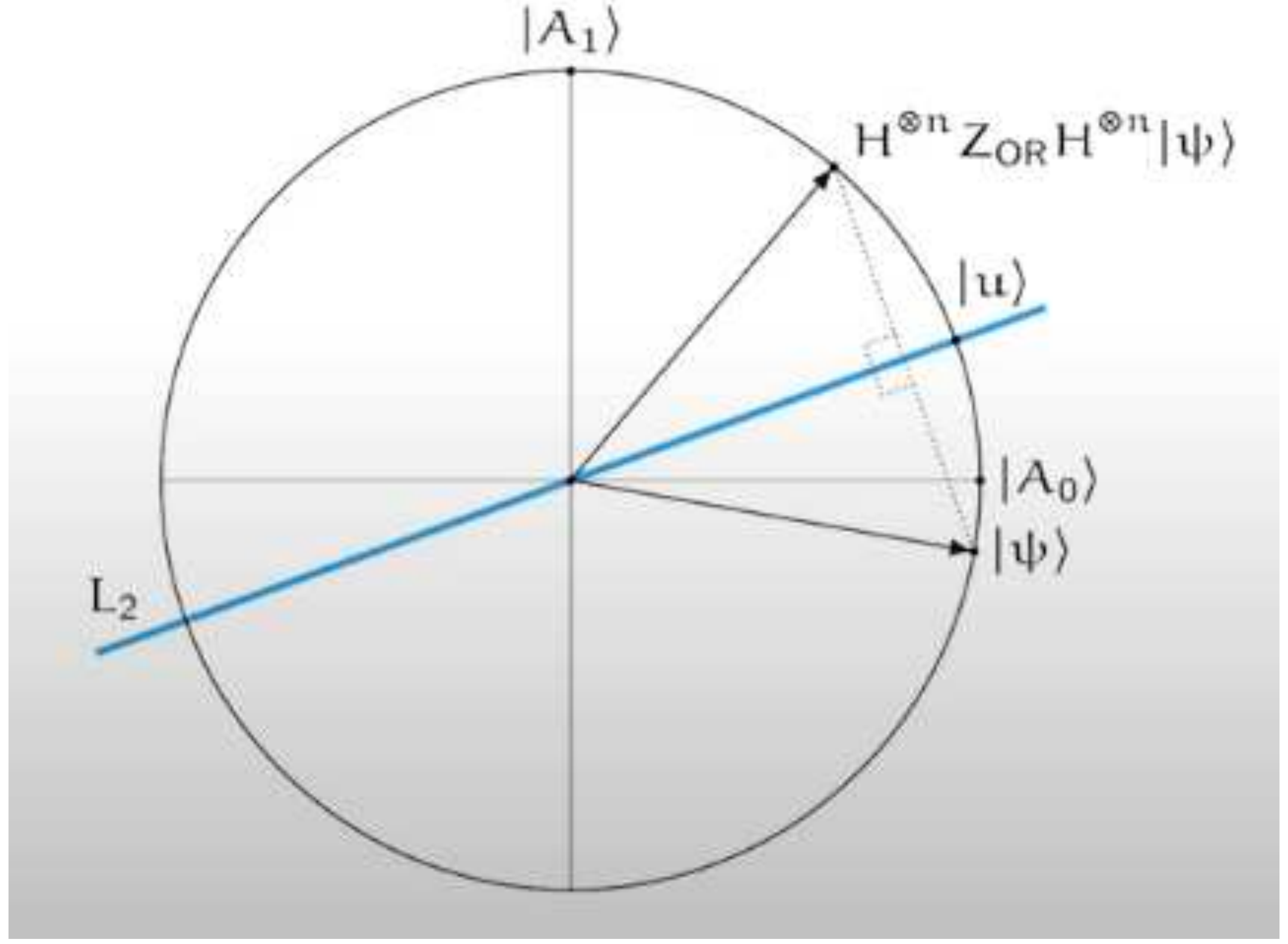
$$Z_f|\psi\rangle = \sum_{x_0\in A_0} \alpha_{x_0}(-1)^{\overset{0}{f(x_0)}}|x_0\rangle + \sum_{x_1\in A_1} \alpha_{x_1}(-1)^{\overset{1}{f(x_1)}}|x_1\rangle$$

$$Z_f|\psi\rangle = \sum_{x_0\in A_0} \alpha_{x_0}|x_0\rangle - \sum_{x_1\in A_1} \alpha_{x_1}|x_1\rangle$$

4. Grover's algorithm : the proof



https://fr.wikipedia.org/wiki/Algorithme_de_Grover



John Watrous, IBM

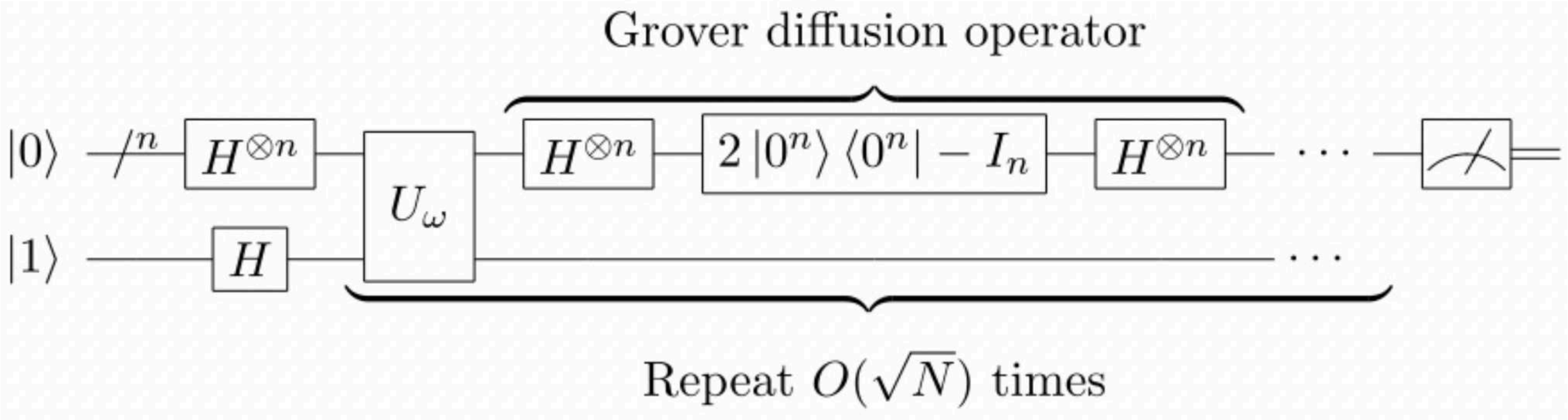
Understanding $H^{\oplus n} Z_{OR} H^{\oplus n}|\psi\rangle$: symmetry around $|u\rangle$.

$$Z_{OR} = 2|0^n\rangle\langle 0^n| - I$$

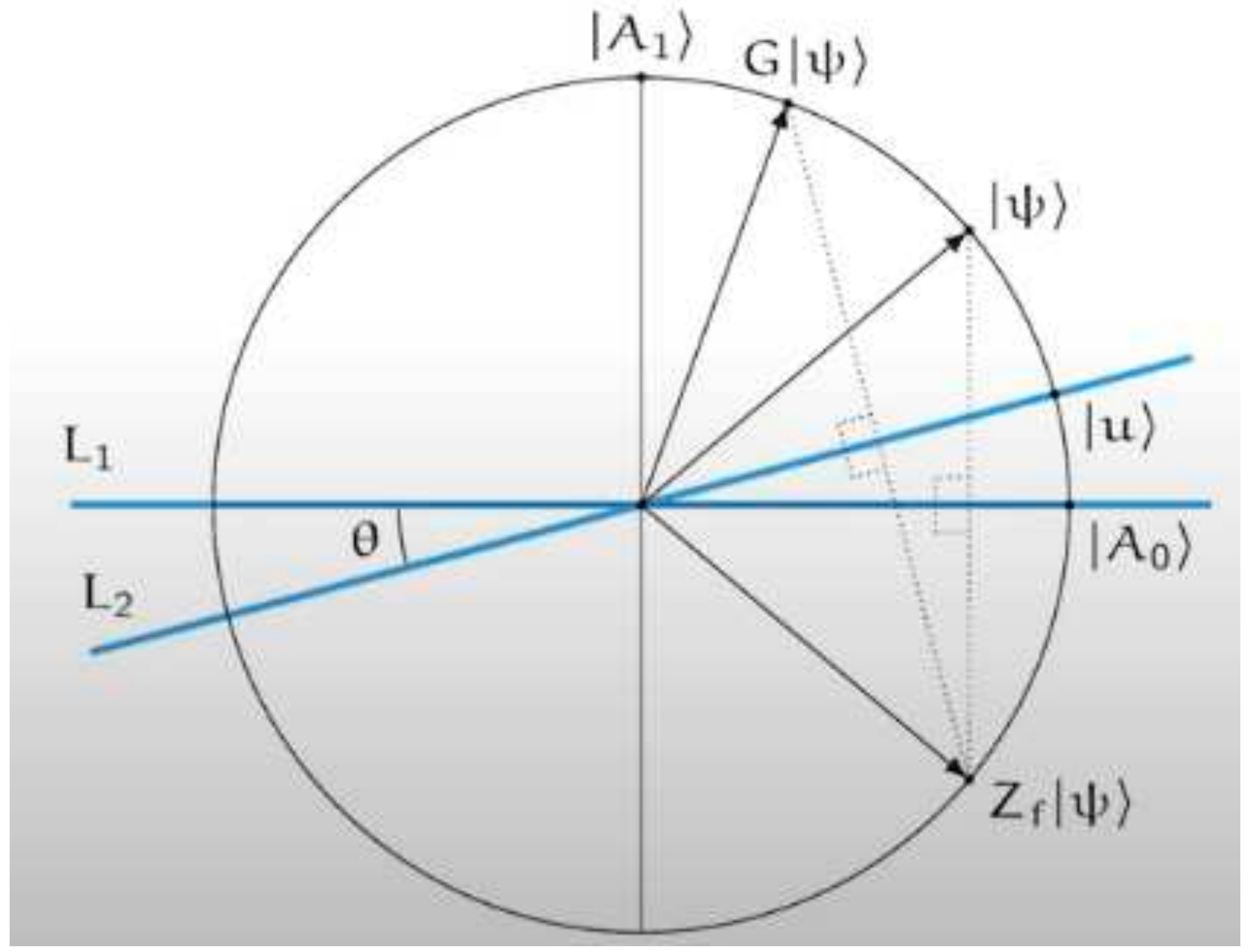
$$\begin{aligned} H^{\oplus n} Z_{OR} H^{\oplus n}|\psi\rangle &= H^{\oplus n}(2|0^n\rangle\langle 0^n| - I)H^{\oplus n} \\ &= 2H^{\oplus n}(|0^n\rangle\langle 0^n|)H^{\oplus n} - H^{\oplus n} I H^{\oplus n} \\ &= 2|u\rangle\langle u| - I \end{aligned}$$

We used the fact that $H^{\oplus n}|0^n\rangle = |u\rangle$

4. Grover's algorithm : the proof



https://fr.wikipedia.org/wiki/Algorithme_de_Grover



John Watrous, IBM

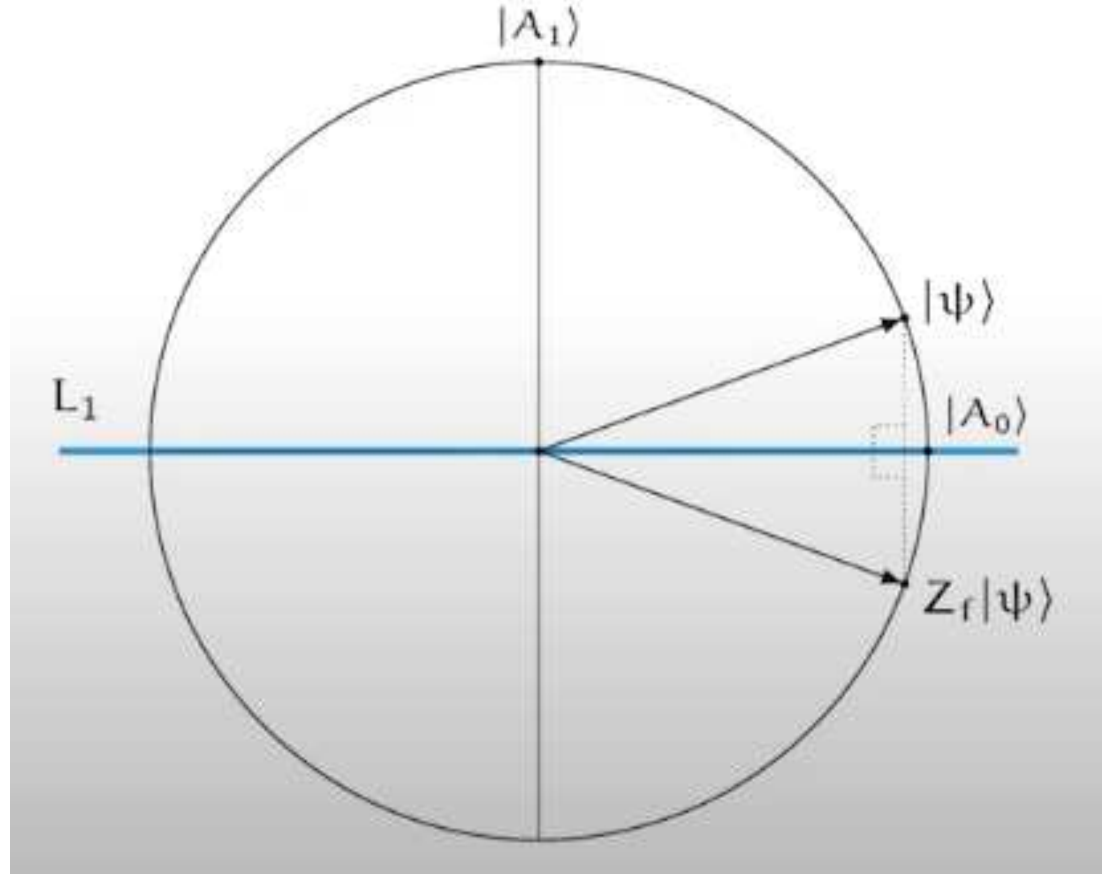
Understanding the Grover diffusion operator:

$(H^{\oplus n} Z_{OR} H^{\oplus n}) Z_f |\psi\rangle$

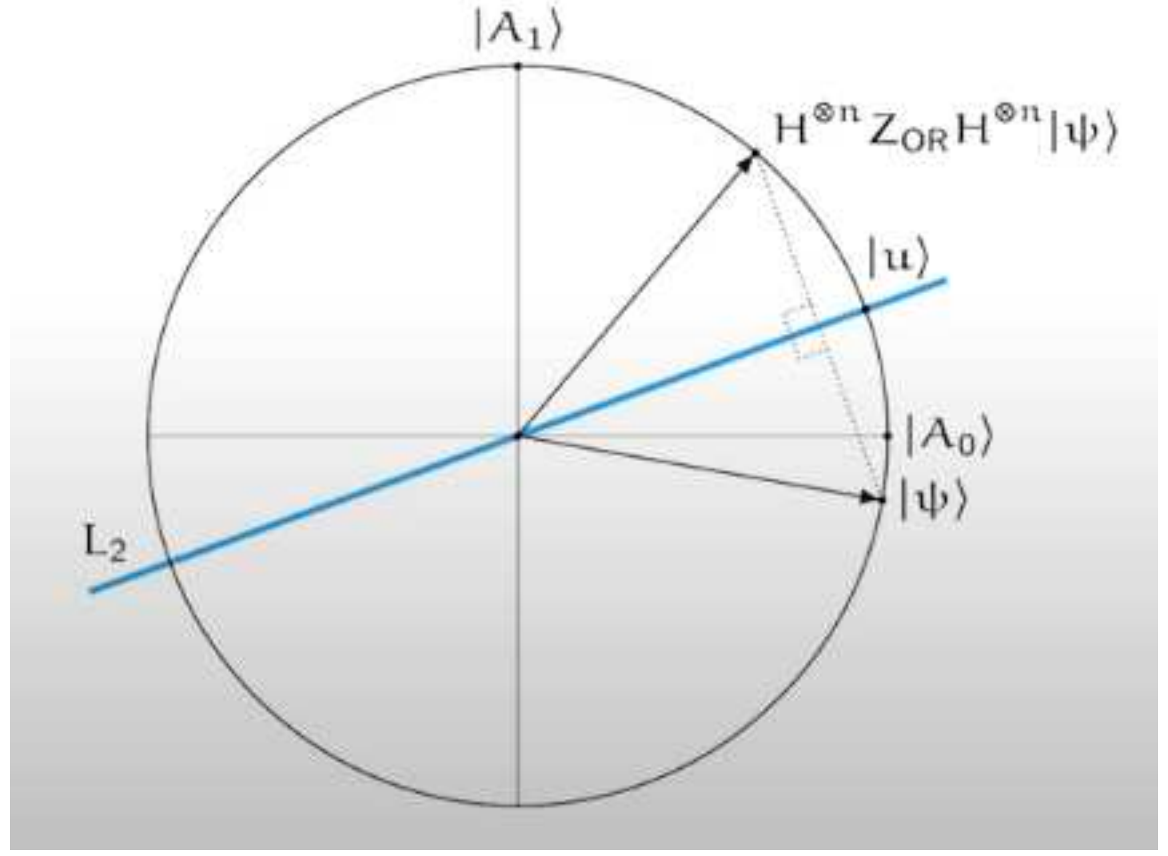
1. Symmetry around $|A_0\rangle$
2. Symmetry around $|u\rangle$

Equivalent to a rotation of vector $|\psi\rangle$ of angle 2θ, where θ is the angle between vectors $|u\rangle$ and $|A_0\rangle$

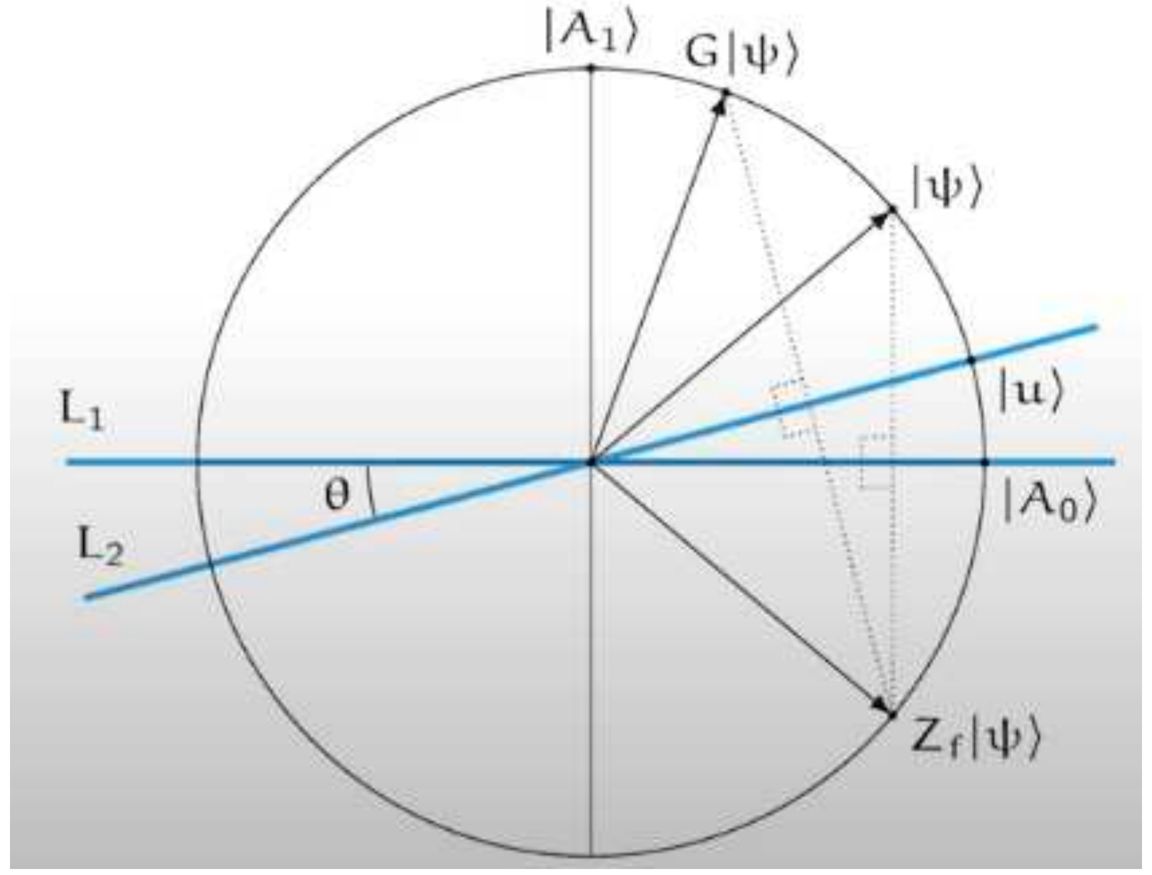
4. Grover's algorithm : the proof



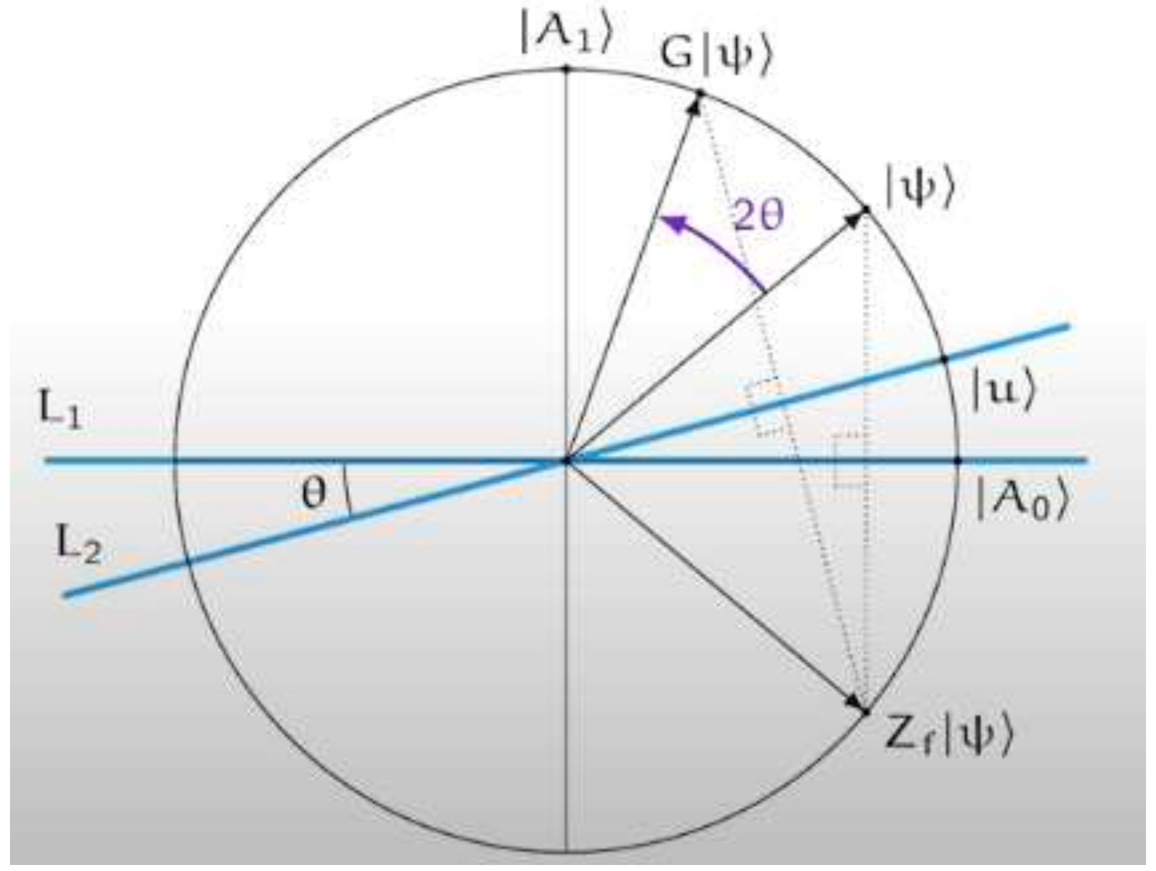
1. On applique Z_f



2. Puis $H^{\oplus n} Z_{OR} H^{\oplus n}$



3. La combinaison des deux produit...



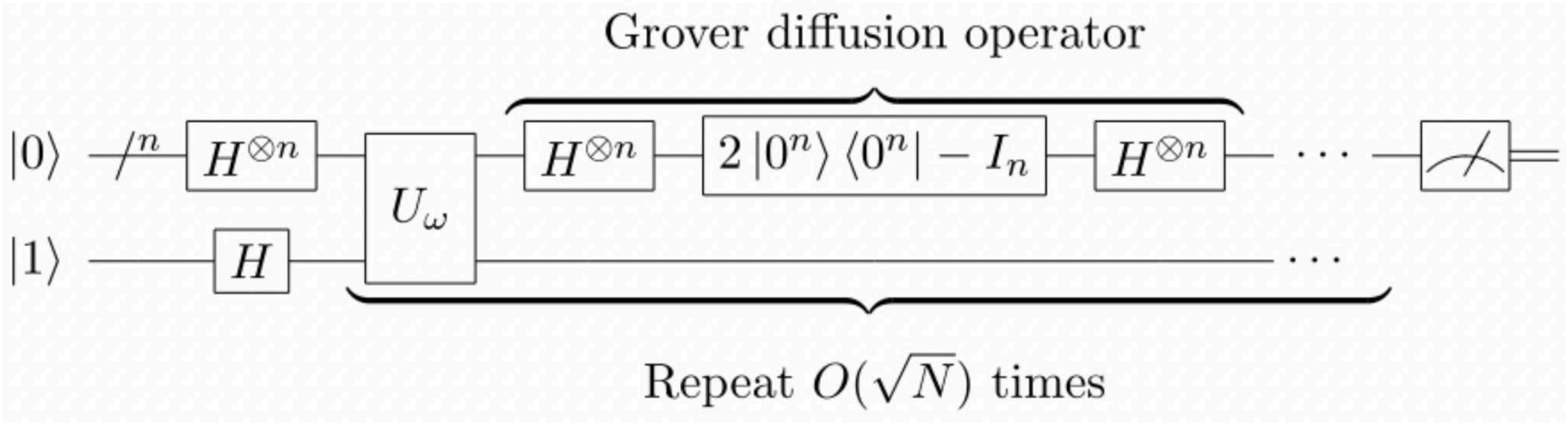
Une rotation d'angle 2θ

Understanding the Grover diffusion operator:
$(H^{\oplus n} Z_{OR} H^{\oplus n}) Z_f |\psi\rangle$

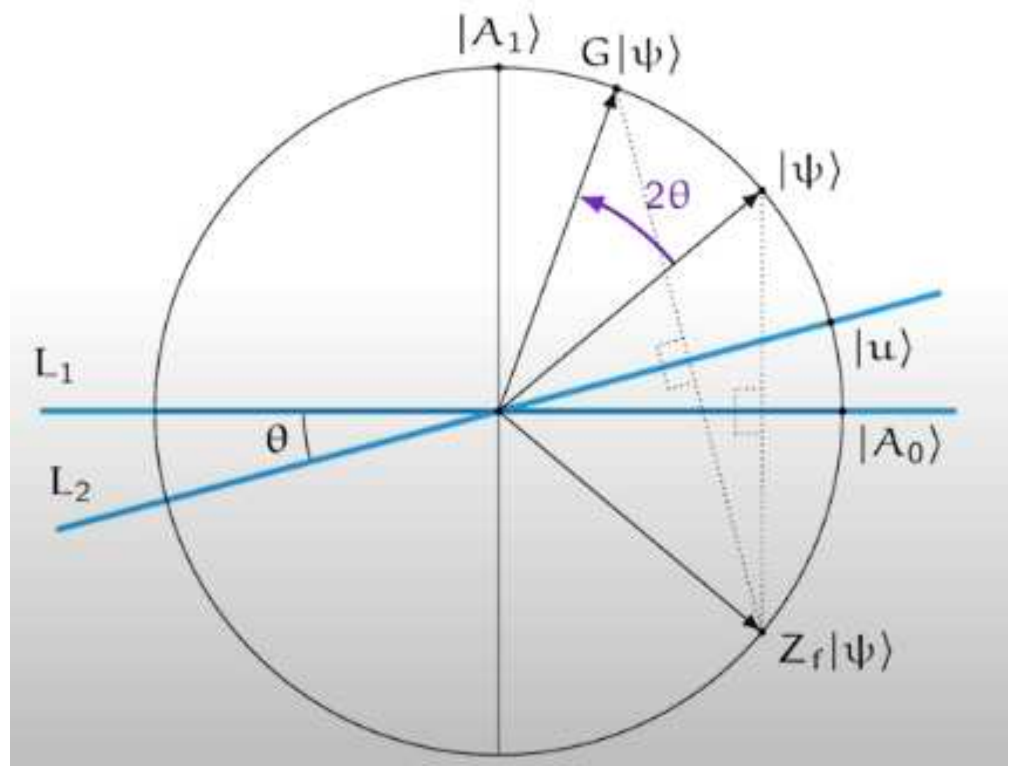
1. Symmetry around $|A_0\rangle$
2. Symmetry around $|u\rangle$

Equivalent to a rotation of vector $|\psi\rangle$ of angle 2θ, where θ is the angle between vectors $|u\rangle$ and $|A_0\rangle$

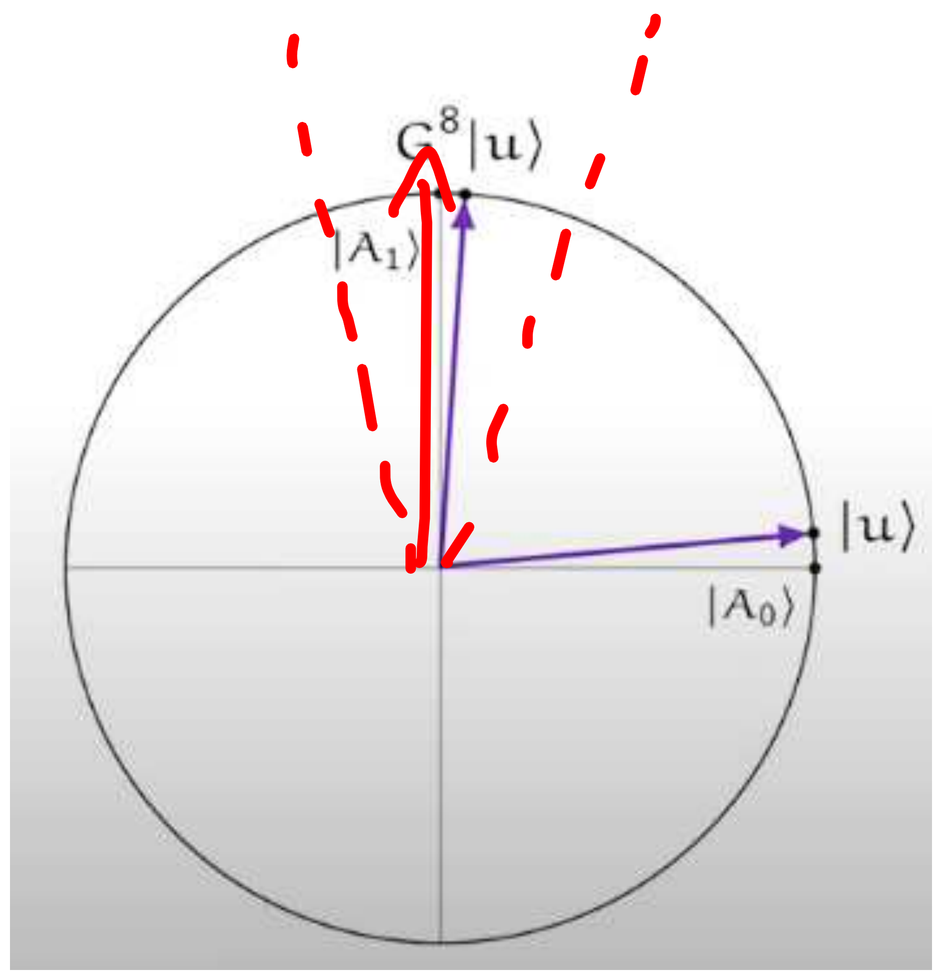
4. Grover's algorithm : choose the number of itérations



https://fr.wikipedia.org/wiki/Algorithme_de_Grover



Une rotation d'angle 2θ



$N = 128, t = 8$

1. We start from $\psi_0 = |u\rangle$, of angle θ with $|A_0\rangle$
2. After t itérations, the angle becomes $(2t+1)\theta\ |A_0\rangle$
3. We aim to measure A_1, thus we want the angle to become roughly 90°, or $\pi/2$

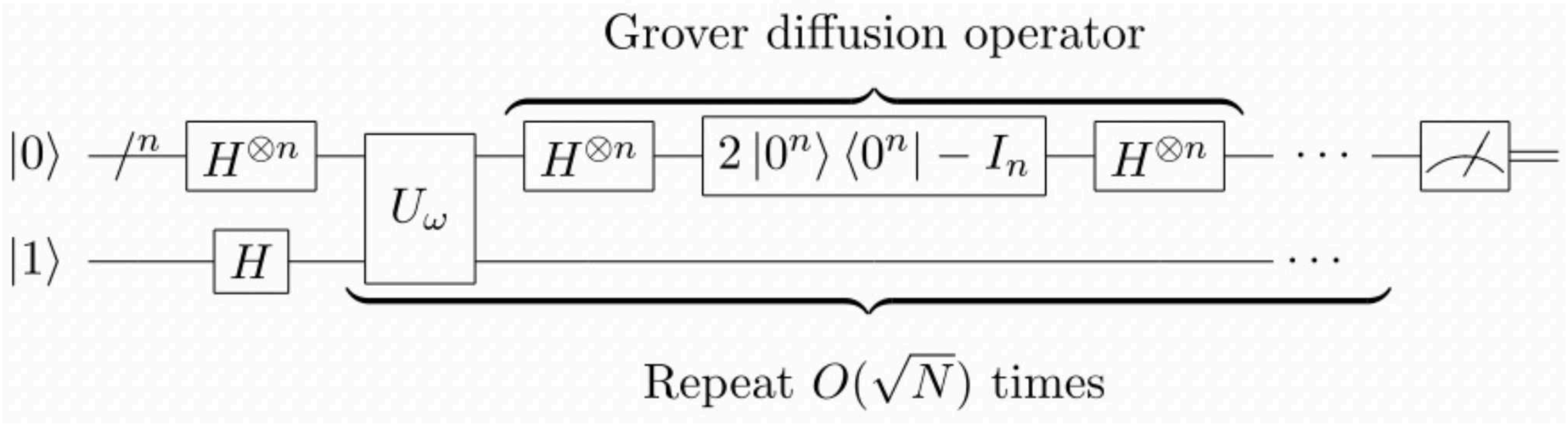
$$|u\rangle = \frac{1}{\sqrt{N}}|A_1| + \frac{\sqrt{N-1}}{\sqrt{N}}|A_0\rangle$$

$$|u\rangle = \sin(\theta)|A_1| + \cos(\theta)|A_0\rangle$$

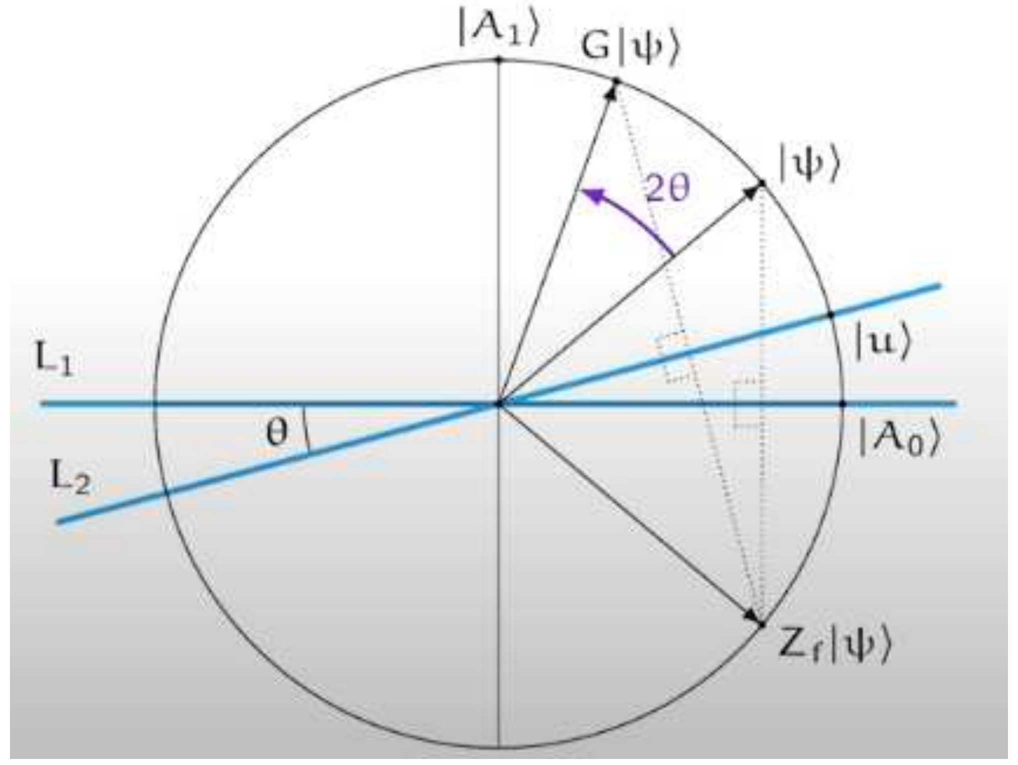
Thus $\sin(\theta) = \dfrac{1}{\sqrt{N}}$, and for large N, $\theta \sim \dfrac{1}{\sqrt{N}}$.

Therefore, we choose $\boxed{t = \lfloor \frac{\pi}{4}\sqrt{N} \rfloor}$ itérations

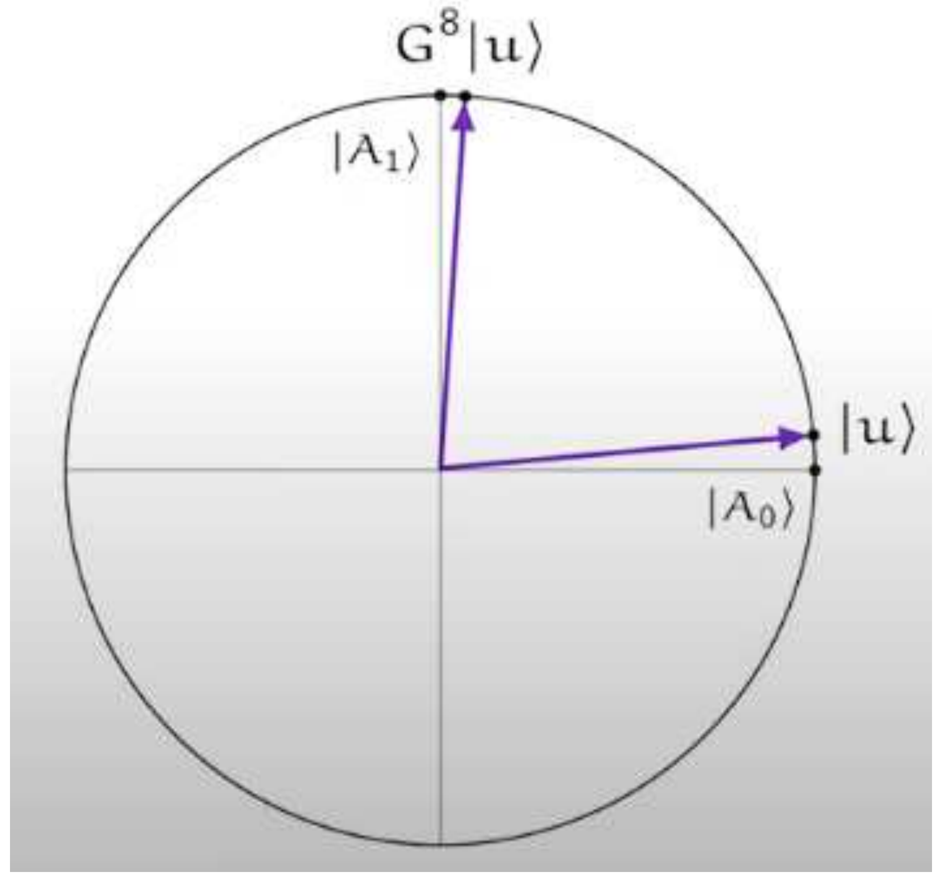
4. Grover's algorithm — the circuit



https://fr.wikipedia.org/wiki/Algorithme_de_Grover



Une rotation d'angle 2θ



$N = 128, t = 8$

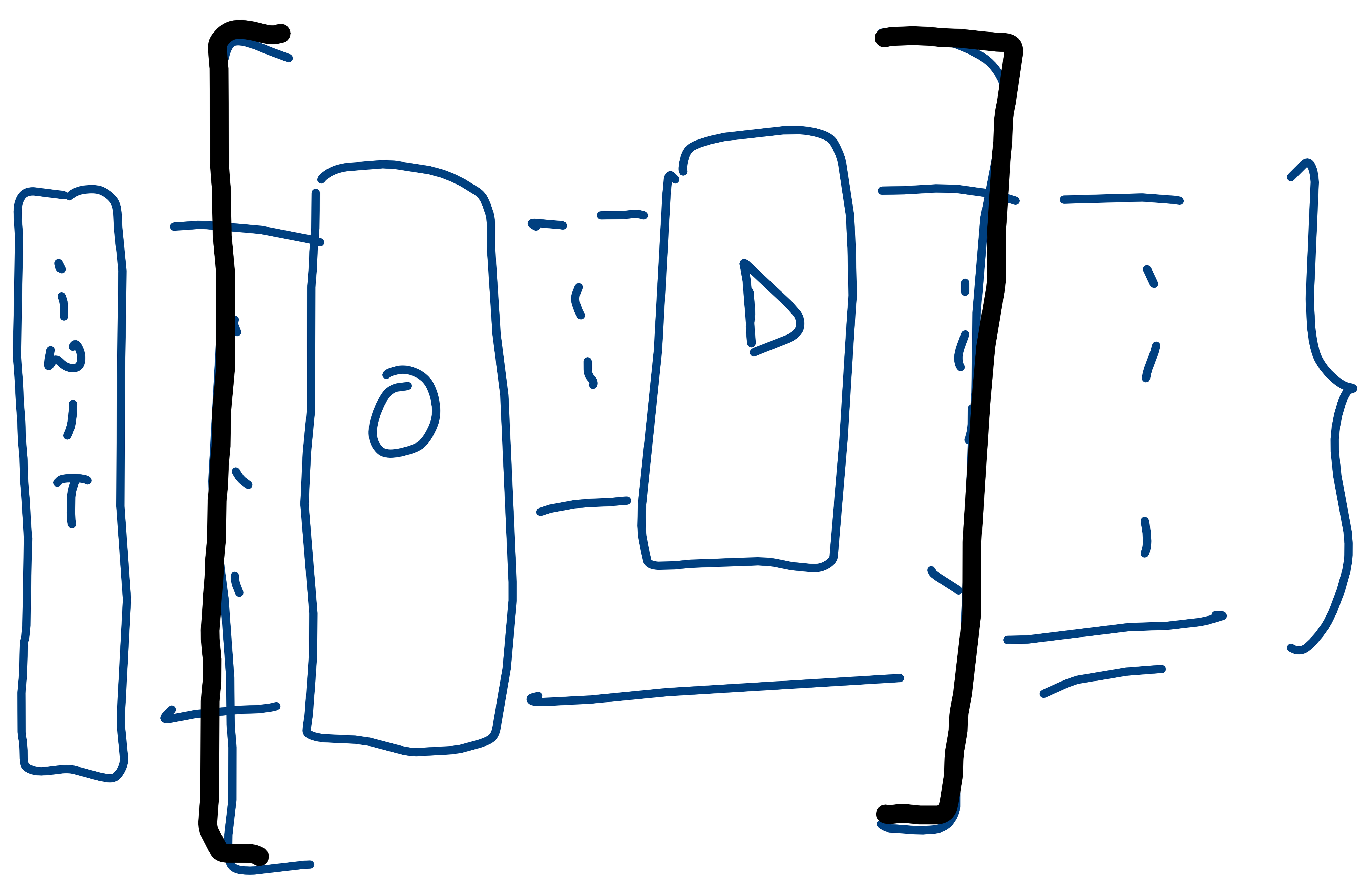
Grover's circuit, $n + 1$ qubits

1. Apply $H^{\oplus n}$: H gates on the first n qubits
2. Apply X and H on qubit $n + 1$
3. repeat $t = \lfloor \frac{\pi}{4}\sqrt{N} \rfloor$ times the Grover operator:
4. add Z_f
5. add $H^{\oplus n}$
6. add Z_{OR}
7. add $H^{\oplus n}$
8. measure the n first qubits

For the implementation of Z_f and Z_{OR} we use circuits U_f and U_{OR}, with the last qubit set to $|-\rangle$.

Circuit

depth



measure n qubits.

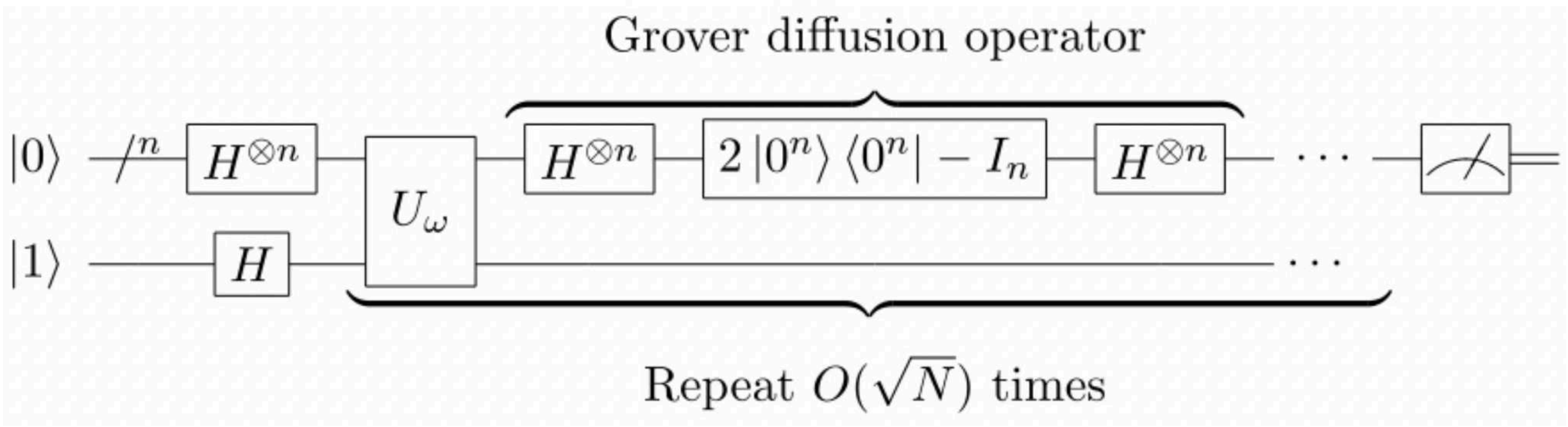
$\times \frac{\pi}{4}\sqrt{N}$ times

\# gates

depth

O is a polynomial circuit.

4. Grover's algorithm — the circuit



https://fr.wikipedia.org/wiki/Algorithme_de_Grover

Grover's circuit

1. Apply $H^{\oplus n}$ on the first n qubits, set the last one to $|-\rangle$
2. repeat $t = \lfloor \frac{\pi}{4}\sqrt{N} \rfloor$ times the Grover operator:
3. add Z_f then $H^{\oplus n}$ then Z_{OR} then $H^{\oplus n}$
4. Measure the first n qubits

Theorem. Grover's algorithm measures x_1 with probability at least $1 - \frac{1}{N}$, where $N = 2^n$.

Extensions

- If f has an arbitrary number of solutions, choose the number t of iterations uniformely at random in $\{1,\ldots,\pi\sqrt{N}/4\}$. Success probability $\geq 40\,\%$.
- One can do better, cf. [John Watrous, YouTube].
- Optimisation : $f:\{0,1\}^n \to \mathbb{N}$, compute x s.t. $f(x)$ is maximum: même complexité, [Dürr, Høyer '97].
- Many applications but $2^{n/2}$ gates! Why is this an issue?

Grover with M solutions

$\boxed{Q_1}$ What if $M > \frac{N}{2}$? Grover algorithm does not work BUT the classical probabilistic algo is OK.

Hypothesis $M \leq \frac{N}{2}$.

If M is known $O\left(\sqrt{N/M}\right)$ iterations is sufficient.

But what if the number of solutions is UNKNOWN ?

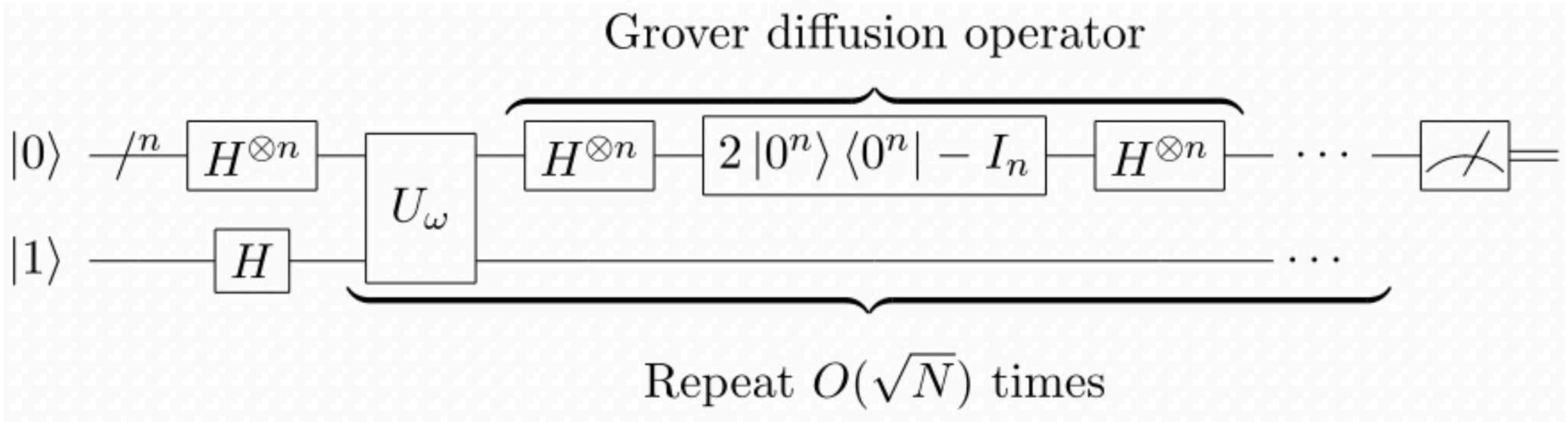
BBHT algorithm

expected number of rounds: $O\left(\sqrt{N/M}\right)$

$m \leftarrow 1$
while True:
    $j \leftarrow$ Random value from 0 to $m-1$
    use Grover with j rounds to get x
    if $f(x) = 1$: return x
    $m \leftarrow \min(\lambda m, \sqrt{N})$

with $\lambda \in (1, 4/3)$ (maybe 6/5)

4. Grover's algorithm



https://fr.wikipedia.org/wiki/Algorithme_de_Grover

Theorem. Grover's algorithm measures x_1 with probability at least $1 - \dfrac{1}{N}$, where $N = 2^n$.

Exercise 1.

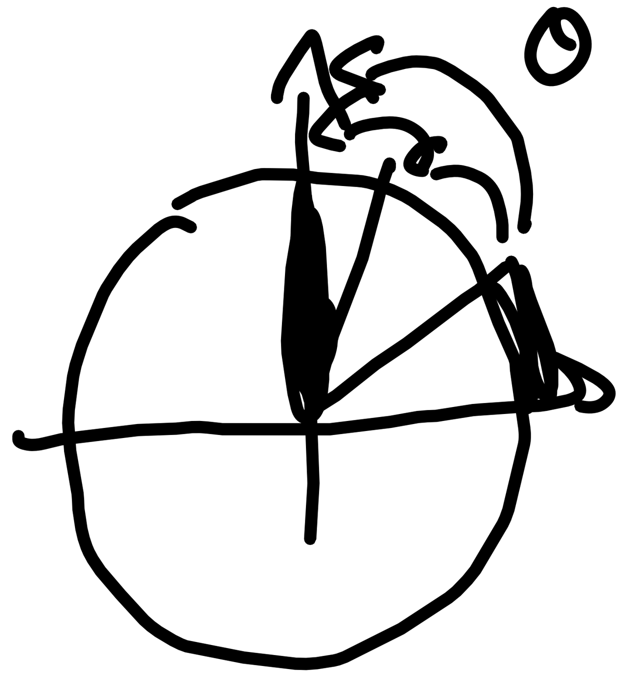
1. Describe completely Grover's algorithm, in the "simplified" case (unique solution).
2. Apply it to a function with 2 bits as input. Analyze the change in amplitudes after each step. What is the probability of finding the solution?
3. Same question for a 1-bit function.

Exercise 2. Now let us consider that the input function has no particular restrictions.

1. Recall the mixed classical/quantum algorithm, which finds a solution x_1 such that $f(x_1) = 1$ with probability $\geq 40\%$, if such a solution exists.
2. Modify the algorithm to obtain a solution with probability at least $1 - 1/N$. Specify its time complexity.

Grover with 2 qubits

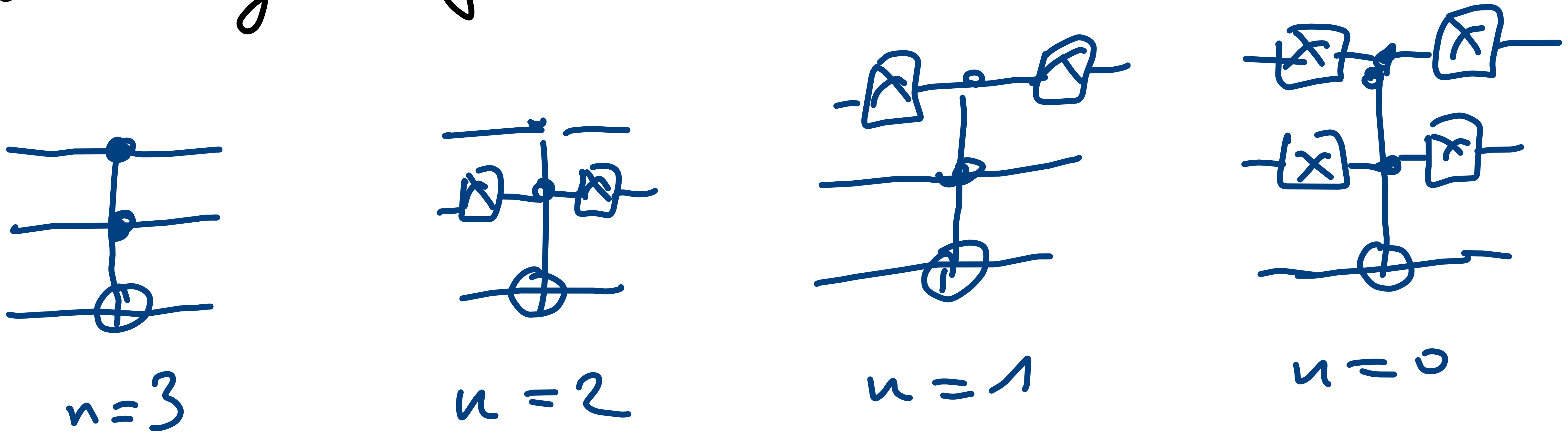
$\left\lfloor \frac{\pi}{4}\sqrt{4} \right\rfloor = \left\lfloor \frac{\pi}{2} \right\rfloor = 1$

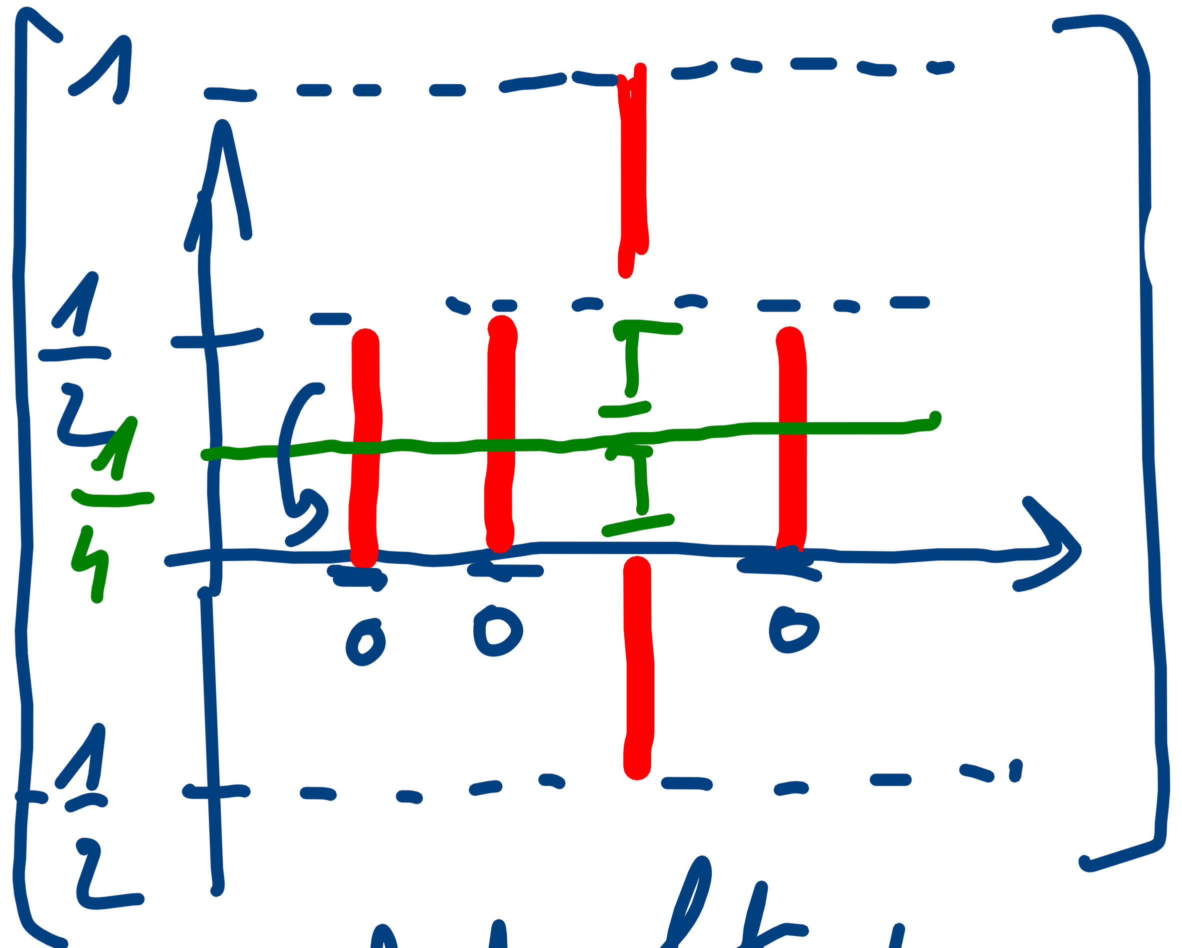


$\theta = \frac{\pi}{6}$

f with 2 input qubits $\quad f(x) = 1$ for exactly one value.

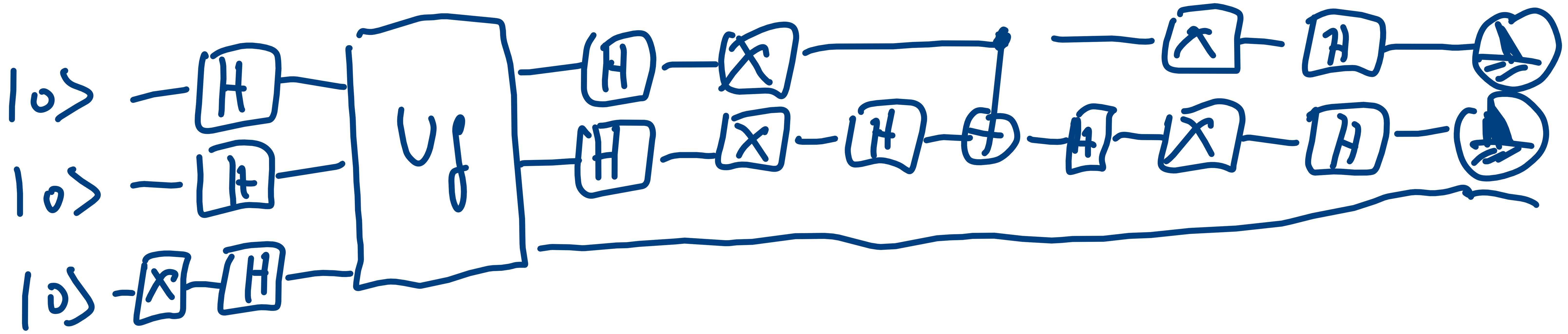
oracles for f:





(perfect solution with prob. 1!

In the classical case with 2 qubits we need 3 calls to f.

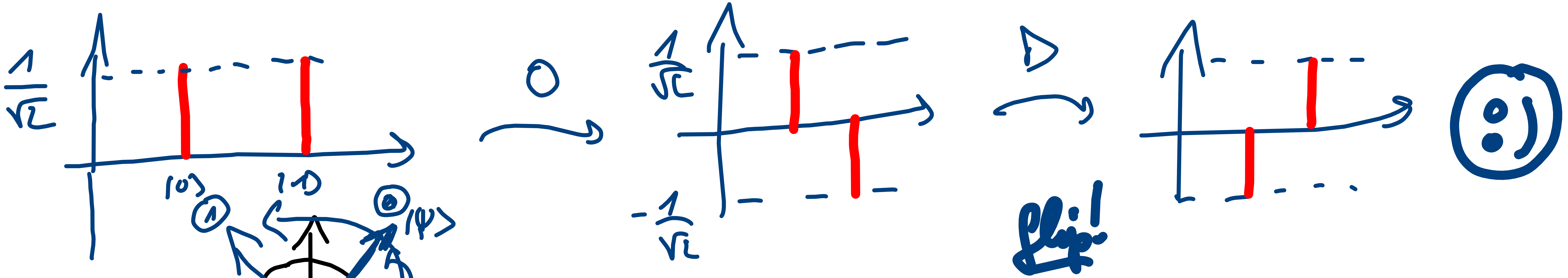


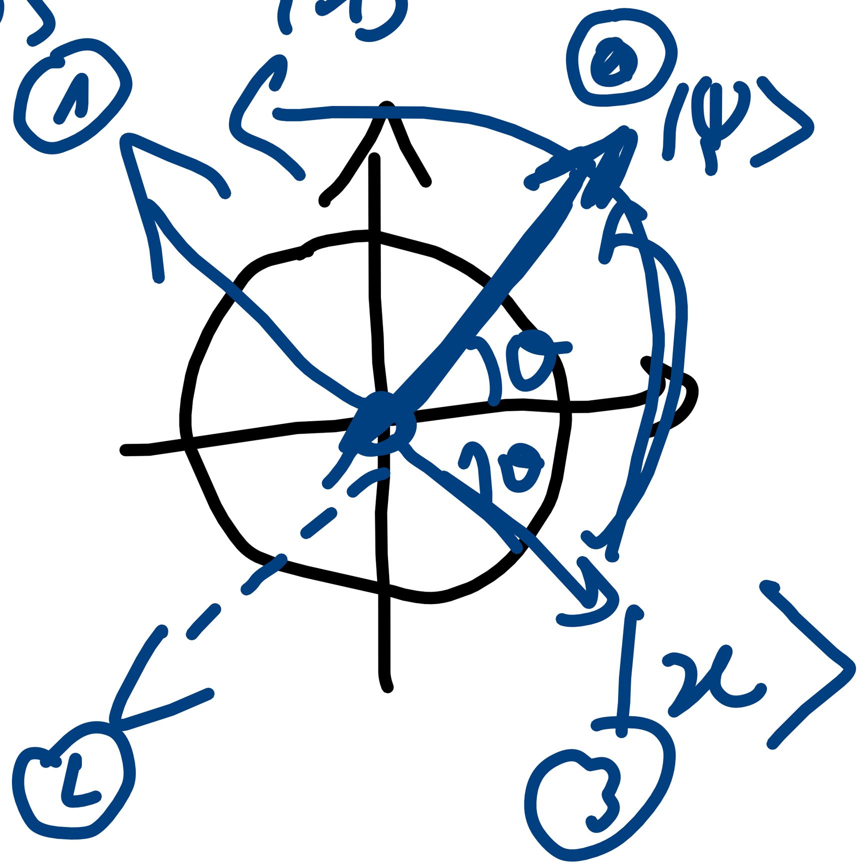
1 call to the oracle

Grover with 1 qubit

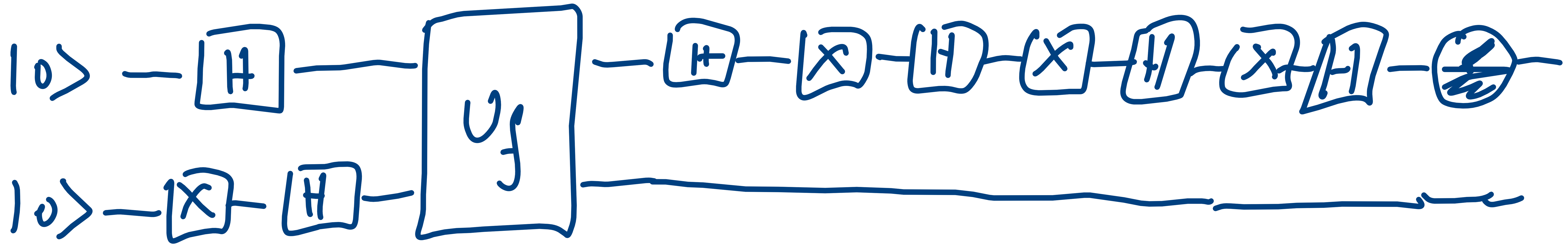
2 inputs $|0\rangle$ and $|1\rangle$

$$\left\lfloor \frac{\pi}{4}\sqrt{2} \right\rfloor = \frac{\pi}{2\sqrt{2}} = 1$$





$$\boxed{P = \frac{1}{2}}$$



C. Grover's algorithm in mixed algorithms

1. Introduction Consider the following problem from a crossword puzzle:

_ _ **r** _ **n** **h** _ (Solution - piranha)

You have an online dictionary with $1,000,000$ words in which the words are arranged alphabetically. You could program it to look for the solution to the puzzle so that it typically solves it after looking through $500,000$ words. It is very difficult to do much better than this. But that is: only if you limit yourself to a classical computer. A quantum computer can be in multiple states at the same time and, by proper design, can carry out multiple computations simultaneously. In case the above dictionary were available on a quantum computer, it would be possible to carry out the search in only about $1,000$ steps by using the quantum search algorithm.

Lov Grover, From Schrödinger's Equation to the Quantum Search Algorithm, ArXiV 2001.

Theorem. Grover's algorithm measures x_1 with probability at least $1 - \frac{1}{N}$, where $N = 2^n$.

Exercise. Let's imagine that we had to implement Grover's puzzle for real. We have discussed Grover's quantum circuit implementation at length.

1. What work would a computer scientist have to do if they only had Qiskit and a quantum computer at their disposal?
2. How could they encode a table of integers, or even Booleans? Even if it were inefficient?

Transpilation

$f \rightsquigarrow U_f$

C. Grover's algorithm in mixed algorithms

1. Introduction Consider the following problem from a crossword puzzle:

_ _ r _ n h _ (Solution - piranha)

You have an online dictionary with $1,000,000$ words in which the words are arranged alphabetically. You could program it to look for the solution to the puzzle so that it typically solves it after looking through $500,000$ words. It is very difficult to do much better than this. But that is: only if you limit yourself to a classical computer. A quantum computer can be in multiple states at the same time and, by proper design, can carry out multiple computations simultaneously. In case the above dictionary were available on a quantum computer, it would be possible to carry out the search in only about $1,000$ steps by using the quantum search algorithm.

Lov Grover, From Schrödinger's Equation to the Quantum Search Algorithm, ArXiV 2001.

Theorem. Grover's algorithm measures x_1 with probability at least $1 - \frac{1}{N}$, where $N = 2^n$.

Exercise. Let's imagine that we had to implement Grover's puzzle for real. We have discussed Grover's quantum circuit implementation at length.

1. What work would a computer scientist have to do if they only had Qiskit and a quantum computer at their disposal?
2. How could they encode a table of integers, or even Booleans? Even if it were inefficient?

Transpilation

$f \rightsquigarrow U_f$

PhysComp96
Full paper
Draft, 10 May 1996

Tight bounds on quantum searching

BBHT

Michel Boyer
Université de Montréal[*]
Gilles Brassard, FRSC[†]
Université de Montréal
Peter Høyer[‡]
Odense University[§]
Alain Tapp[¶]
Université de Montréal

We provide a tight analysis of Grover's recent algorithm for quantum database searching. We give a simple closed-form formula for the probability of success after any given number of iterations of the algorithm. This allows us to determine the number of iterations necessary to achieve almost certainty of finding the answer. Furthermore, we analyse the behaviour of the algorithm when the element to be found appears more than once in the table and we provide a new algorithm to find such an element even when the number of solutions is not known ahead of time. Using techniques from Shor's quantum factoring algorithm in addition to Grover's approach, we introduce a new technique for approximate quantum counting, which allows to estimate the number of solutions. Finally we provide a lower bound on the efficiency of any possible quantum database searching algorithm and we show that Grover's algorithm nearly comes within a factor 2 of being optimal in terms of the number of probes required in the table.

1 Introduction

Assume you have a large table $T[0\,..\,N-1]$ in which you would like to find some element x. More precisely, you wish to find an integer i such that $0 \le i < N$ and $T[i] = x$, provided such an i exists. This problem can obviously be solved in a time in $O(\log N)$ if the table is sorted, but no classical algorithm (deterministic or probabilistic) can succeed in the general case—when the elements of T are in an arbitrary order—with probability better than 1/2, say, without probing more than half the entries of T. Grover [4] has recently discovered an algorithm for the *quantum* computer that can solve this problem in expected time in $O(\sqrt{N}\,)$. He also remarked that a result in [1] implies that his algorithm is optimal, up to a multiplicative constant, among all possible quantum algorithms.

In this paper we provide a tight analysis of Grover's algorithm. In particular we give a simple closed-form formula for the probability of success after any given number of iterations. This allows us to determine the number of iterations necessary to achieve almost certainty of finding the answer, as well as an upper bound on the probability of failure. More significantly, we analyse the behaviour of the algorithm when the element to be found appears more than once in the table. An algorithm follows immediately to solve the problem in a time in $O(\sqrt{N/t}\,)$ when it is known that there are exactly t solutions. We also provide an algorithm capable of solving the problem in a time in $O(\sqrt{N/t}\,)$ even if the number t of solutions is not known in advance. Bringing ideas from Shor's quantum factorization algorithm [6] into Grover's algorithm, we sketch a new quantum algorithm capable of approximately *counting* the number of solutions. We also generalize Grover's algorithm in the case N is not a power of 2. Finally, we refine the argument of [1] to show that Grover's algorithm could not be improved to require much less than half the number of table lookups that it currently makes when a 50% probability of success is desired.

2 Finding a unique solution

Assume for now that there is a unique i_0 such that $T[i_0] = x$. For any real numbers k and ℓ such that $k^2 + (N-1)\ell^2 = 1$, define the state of a quantum register

$$|\Psi(k,\ell)\rangle = k|i_0\rangle + \sum_{i \ne i_0} \ell|i\rangle$$

where the sum is over all $i \ne i_0$ such that $0 \le i < N$. (We shall never need complex amplitudes in this paper, except in §7.)

[*]Département IRO, C.P. 6128, succursale centre–ville, Montréal, Canada H3C 3J7. {boyer,brassard,tappa}@iro.umontreal.ca
[†]Supported in part by NSERC and FCAR
[‡]Supported in part by the ESPRIT Long Term Research Programme of the EU under project number 20244 (ALCOM-IT).
[§]Department of Mathematics and Computer Science, Odense University, Campusvej 55, DK–5230 Odense M, Denmark. u2pi@imada.ou.dk
[¶]Supported in part by NSERC