Quantum Computing V
Grover's Search Algorithm

07.02.23
Jonas Frerick

Refs: - de Wolf's lecture notes [1907.09415] Ch. 7

- Preskill's lecture notes Ch. 6

- Quantum Computing – A Gentle Introduction Ch. 9 (Rieffel & Polak)

The problem:

For $N = 2^n$, we have some $x \in \{0,1\}^N$

Goal: Find i so that $x_i = 1$ (or output "no solution")

Alternative formulation:

For a given function $f: \{0,1\}^n \to \{0,1\}$, find $y \in \{0,1\}^n$ so that $f(y) = 1$

Classically, this corresponds to drawing from an urn with t marked elements.

t ← Hamming weight of the string x

Both formulations require a search for t marked objects in a "database" (to be precise: unordered database) with N entries.

$\Rightarrow$ classical runtime $= \mathcal{O}\left(\frac{N}{t}\right)^*$

* counting the evaluations of the oracle, usually an evaluation takes $\mathcal{O}(\text{polylog}(N))$

Now let's see how we can speed this up on a quantum computer:

<u>Ingredients:</u> (following the first description)

1.) oracle/phase query for the correct element:

$$O_{x,\pm}|i\rangle = (-1)^{x_i}|i\rangle$$

with the usual map between x and $|i\rangle$ via binary representation

2.) unitary transformation R_0:

$$R_0|i\rangle = \begin{cases} |i\rangle, & i = 0^n \\ -|i\rangle, & \text{else} \end{cases}$$

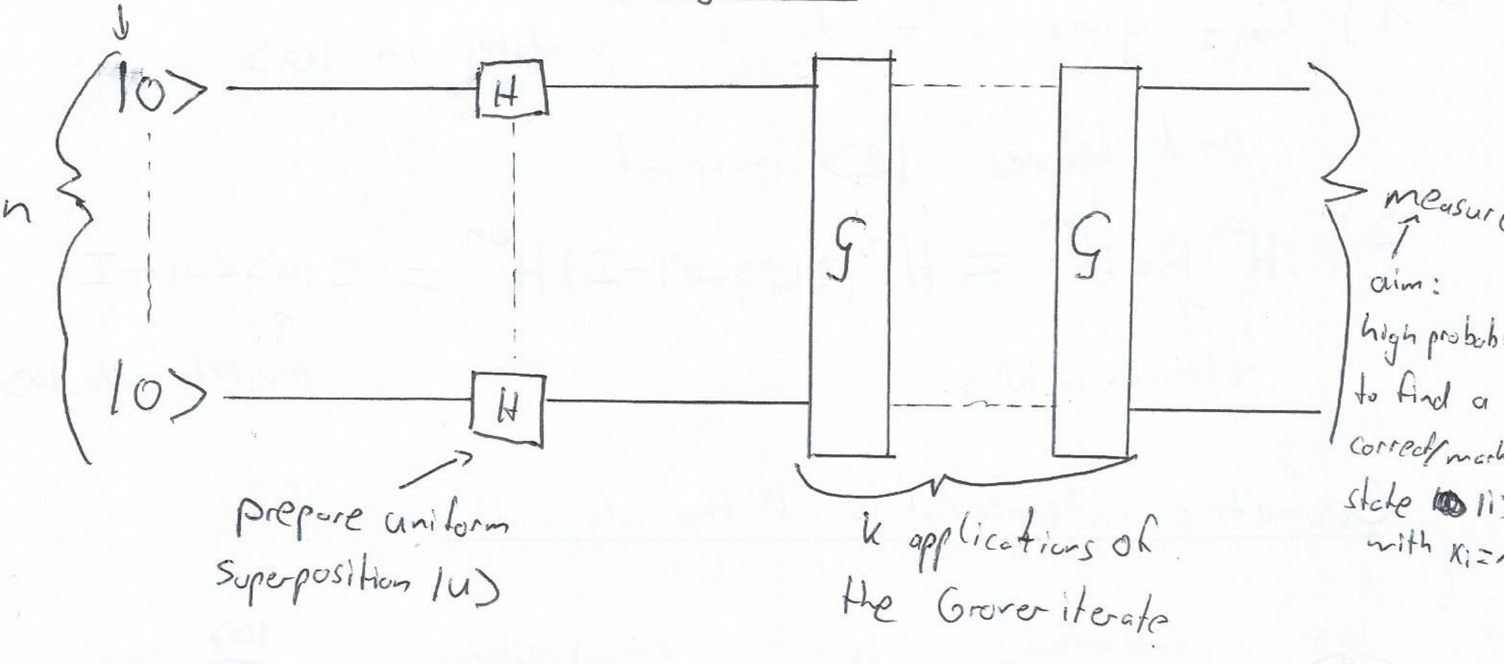
3.) Hadamard transformation

e.g. $H|0^n\rangle = \frac{1}{\sqrt{N}} \sum_i |i\rangle \equiv |u\rangle$

$\Rightarrow$ Define the <u>Grover iterate</u>

$$\boxed{G = H^{\otimes n} R_0 H^{\otimes n} O_{x,\pm}}$$

Grover's algorithm (1996)



Two questions:

(I) What is the iterate doing exactly?

(II) What value should k take?

(I) Write

$$|u\rangle = H^{\otimes n}|0^n\rangle = \sin(\theta)\,\underset{\text{"good"}}{|G\rangle} + \cos(\theta)\,\underset{\text{"bad"}}{|B\rangle}$$

$$\Rightarrow |G\rangle = \frac{1}{\sqrt{t}} \sum_{i:x_i=1} |i\rangle, \quad |B\rangle = \frac{1}{\sqrt{N-t}} \sum_{i:x_i=0} |i\rangle, \quad \theta = \arcsin\left(\sqrt{\frac{t}{N}}\right)$$

(Note: $|G\rangle$ and $|B\rangle$ are orthonormal)

We will show that G is the product of two reflections:

1.) through $|B\rangle$: $O_{x,\pm}$

2.) through $|u\rangle$: $H^{\otimes n} R_0 H^{\otimes n}$

Why? A reflection through a subspace V is a unitary A so that $Av = v \;\forall v \in V$ and $Aw = -w$ for $w \perp V$

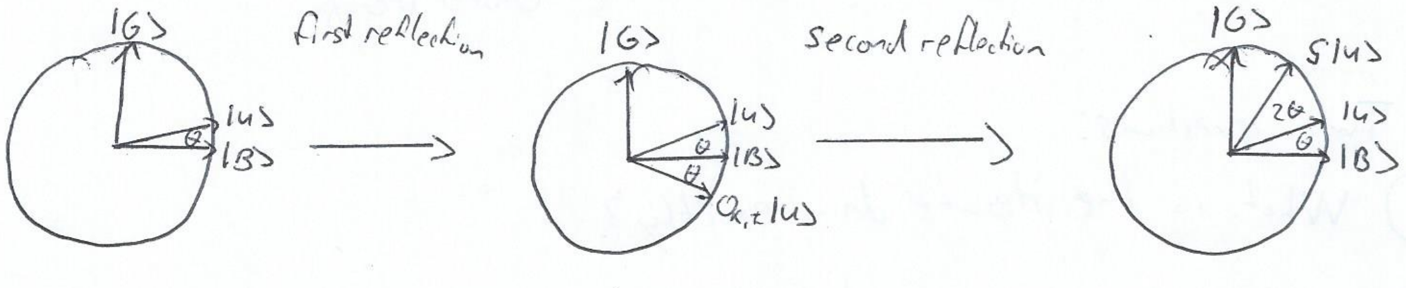
$$\Rightarrow A = 2\underset{\text{projector into subspace } V}{P_V} - I$$

$\Rightarrow$ 1.) $O_{x,\pm}$ gives a "$-$" to solutions/states in $|G\rangle$ and leaves $|B\rangle$ invariant

2.) $H^{\otimes n} R_0 H^{\otimes n} = H^{\otimes n}(2|0^n\rangle\langle 0^n| - I)H^{\otimes n} = 2|u\rangle\langle u| - I$

R_0: reflection in $|0^n\rangle$ — $2|u\rangle\langle u|$: projector into $|u\rangle$

Geometric interpretation of the algorithm



$\Rightarrow$ every application changes the angle by $+2\theta$!

$$\boxed{G^k|u\rangle = \sin((2k+1)\theta)|G\rangle + \cos((2k+1)\theta)|B\rangle}$$

Probability of finding a correct solution:

$$\boxed{P_k = \sin^2((2k+1)\theta)}$$

Now we can answer question II

We want $P_k = 1 \Rightarrow (2k+1)\theta = \frac{\pi}{2} \Leftrightarrow k = \frac{\pi}{4\theta} - \frac{1}{2}$

Let's choose $\tilde{k} = k + \varepsilon$ as the closest integer to k (i.e. $|\varepsilon| \leq \frac{1}{2}$)

$\Rightarrow P_{\tilde{k}} = \sin^2((2\tilde{k}+1)\theta) = \sin^2\left(\frac{\pi}{2} + 2\varepsilon\theta\right) = \cos^2(2|\varepsilon|\theta)$

$$\underset{\cos(x) \geq 1 - \frac{x^2}{2}}{\geq} 1 - 4\varepsilon^2\theta^2 \underset{|\varepsilon| \leq \frac{1}{2}}{\geq} 1 - \theta^2 = 1 - \arcsin\left(\sqrt{\tfrac{|G|}{N}}\right)^2 \underset{\arcsin(x) \geq x}{\geq} 1 - \frac{t}{N}$$

$\Rightarrow P_{\tilde{k}} = 1 - \delta$ with $\boxed{\delta \leq \frac{t}{N}}$ (usually) small error probability!

$$\tilde{k} \simeq \frac{\pi}{4\theta} - \frac{1}{2} \xrightarrow{\frac{t}{N} \ll 1} \frac{\pi}{4}\sqrt{\frac{N}{t}} = \boxed{\mathcal{O}\left(\sqrt{\frac{N}{t}}\right)}$$

$\Rightarrow$ Grover's search algorithm can speed up a "brute-force" search ~~spd~~ quadratically!

Algebraic argument

Write
$$|u_k\rangle = G^k |u\rangle = \overbrace{\sum_{i:x_i=1} a_k |i\rangle}^{\text{"}\sin(\theta_k)\,|G_k\rangle\text{"}} + \overbrace{\sum_{i:x_i=0} b_k |i\rangle}^{\text{"}\cos(\theta_k)\,|B_k\rangle\text{"}}$$

with $a_0 = b_0 = \frac{1}{\sqrt{N}}$

We can use that in the basis $|i\rangle$ we have

$$H^{\otimes n} R_0 H^{\otimes n} = \underbrace{\frac{2}{N} J}_{\hookrightarrow\, 2|u\rangle\langle u|} - I$$

uniform super-position

$$J = N\left\{\overbrace{\begin{pmatrix} 1 & \cdots & 1 \\ \vdots & \ddots & \vdots \\ 1 & \cdots & 1 \end{pmatrix}}^{N}\right.$$

$$\Rightarrow G|u_k\rangle = \sum_{i:x_i=1} a_{k+1} |i\rangle + \sum_{i:x_i=0} b_{k+1} |i\rangle$$

$$= \sum_{i:x_i=1} \left[(-1)\left(\frac{2t}{N} - 1\right) a_k + \frac{2(N-t)}{N} b_k\right] |i\rangle$$

$$+ \sum_{i:x_i=0} \left[\left(\frac{2N-2t}{N} - 1\right) b_k + (-1)\frac{2t}{N} a_k\right] |i\rangle$$

$$\Rightarrow a_{k+1} = \frac{N-2t}{N} a_k + \frac{2(N-t)}{N} b_k$$

$$b_{k+1} = -\frac{2t}{N} a_k + \frac{N-2t}{N} b_k$$

Interplay: Solving the recursion relation
(For an alternative and maybe more elegant approach you can check Wikipedia)

Define new variables $\alpha_k = a_k + C_1 b_k$

$\beta_k = a_k + C_2 b_k$

so that $\alpha_{k+1} \propto \alpha_k$ and $\beta_{k+1} \propto \beta_k$

$\Rightarrow$ We demand

$$\underbrace{a_{k+1} + C \, b_{k+1}}_{C_1 \text{ or } C_2} = \frac{N-2t(1+C)}{N} a_k + \frac{N(2+C)-2t(1+C)}{N} b_k$$

$$= \frac{N-2t(1+C)}{N}\left(a_k + \frac{N(2+C)-2t(1+C)}{N-2t(1+C)} b_k \right)$$

$$\overset{!}{=} \text{const.} \cdot \left(a_k + C b_k \right)$$

$$\Rightarrow \quad \cancel{NC} - \cancel{2tC} - 2tC^2 = 2N + \cancel{NC} - 2t - \cancel{2tC}$$

$$\Leftrightarrow \quad C_{1/2} = \pm i \sqrt{\frac{N-t}{t}} = \pm i \cot(\theta)$$

choose + for α_k

$$\Rightarrow \quad \alpha_{k+1} = \left[\frac{(N-2t)}{N} - 2i\sqrt{t}\sqrt{N-t} \right] \alpha_k \Rightarrow \alpha_k = \left[1 - 2\sin^2\theta - 2i\sin\theta\cos\theta \right]^k \alpha_0$$

$$\beta_{k+1} = \quad \dots \quad = \left[1 - 2\sin^2\theta + 2i\sin\theta\cos\theta \right]^k \beta_0$$

$$\alpha_0 = \frac{1}{\sqrt{N}}\left(1 + i\cot(\theta)\right), \quad \beta_0 = \frac{1}{\sqrt{N}}\left(1 - i\cot(\theta)\right)$$

$$\Rightarrow \quad \alpha_k = \frac{1}{\sqrt{N}} \left[\cos(2\theta) + i\sin(2\theta)\right]^k (1 + i\cot(\theta))$$

$$\beta_k = \frac{1}{\sqrt{N}} \left[\cos(2\theta) - i\sin(2\theta)\right]^k (1 - i\cot(\theta))$$

Let's go back to $a_k = \frac{1}{2}(\alpha_k + \beta_k)$

$$a_k \underset{\text{binomial theorem}}{=} \frac{1}{2\sqrt{N}} \left[\sum_{j=0}^{k} \binom{k}{j} \cos(2\theta)^{k-j} \sin(2\theta)^j \left(\overbrace{\left((-i)^j + i^j\right)}^{\text{picks out even numbers with alternating sign}} + i\cot(\theta) \overbrace{\left((-i)^j - i^j\right)}^{\text{picks out odd numbers with alternating sign}} \right)\right]$$

$$= \frac{1}{\sqrt{N}} \left[\sum_{l=0}^{\lfloor \frac{k}{2} \rfloor} (-1)^l \binom{k}{2l} \cos(2\theta)^{k-2l} \sin(2\theta)^{2l} + \cot(\theta) \sum_{l=0}^{\lfloor \frac{k-1}{2} \rfloor} (-1)^l \binom{k}{2l+1} \cos(2\theta)^{k-2l-1} \times \sin(2\theta)^{2l+1}\right]$$

$$= \frac{1}{\sqrt{N}} \left[\cos(2k\theta) + \cot(\theta)\sin(2k\theta)\right]$$

$$= \underbrace{\frac{1}{\sqrt{N}\sin\theta}}_{\frac{1}{\sqrt{t}}} \sin((2k+1)\theta) = \boxed{\frac{1}{\sqrt{t}} \sin((2k+1)\theta)}$$

$$\Rightarrow \quad P_k = \sum_{f(x_i)=1} |a_k|^2 = t\,|a_k|^2 = \underline{\underline{\sin^2((2k+1)\theta)}}$$

For completeness: $\boxed{b_k = \frac{1}{\sqrt{N-t}} \cos((2k+1)\theta)}$

Problems/comments/variations:

- if we know t **exactly**, we can assure to end up in exactly the right state
  - ↳ idea: add a qubit to guarantee $k = \frac{\pi}{4\theta} - \frac{1}{2}$ is integer

- t unknown: problem! (no stopping condition)
  - ↳ use exponentially increasing guesses for k (still only $\mathcal{O}(\sqrt{\frac{N}{t}})$ expected queries) [de Wolf]
  - ↳ no solution: check the outputs of the algorithm

    or

  - ↳ pick k from $\{0, 1, \dots, k_{max} (\simeq \lceil \frac{\pi}{4} \sqrt{N} \rceil)\}$ (uniformly) [Preskill]

- lower bound on t: $\tau \leq t$:

  run for $\sim 3\sqrt{\frac{N}{\tau}}$, this gives us a $\geq \frac{2}{3}$ chance that we find the right state:

  expected number of queries

  Markov's inequality: $P(t \geq 3\sqrt{\frac{N}{\tau}}) \leq \frac{1}{3}$

  probability that we have not found the correct state yet

- a variation can also be used to count t (approximately) by a combination of Grover's algorithm with a QFT (essentially using the periodicity of the success probability)
  - ↳ "quantum counting"

General amplitude amplification

Assume that we have a unitary A that prepares m qubits in a mixture of "good" and "bad" states:

$$A|0^m\rangle = \sqrt{p}\,|\psi_1\rangle + \sqrt{1-p}\,|\psi_0\rangle$$

$$\| \quad |u\rangle$$

orthonormal (e.g. the last qubit is $|0\rangle$ and $|1\rangle$ respectively)

Task: Increase weight of $|\psi_1\rangle$

Again, we need an R_G so that

$$R_G|\psi_1\rangle = -|\psi_1\rangle$$
$$R_G|\psi_0\rangle = |\psi_0\rangle$$

$\Rightarrow$ reflection through bad state (analogous to $O_{x,t}$)

we could use this via Z-gate on the last qubit

Similarly, we look at $A R_0 A^{-1}$:

$$A R_0 A^{-1}|u\rangle = (2A|0^m\rangle\langle 0^m|A^{-1} - AA^{-1})A|0^m\rangle$$
$$= 2A|0^m\rangle - A|0^m\rangle = \underline{|u\rangle}$$

for $|u^\perp\rangle$ with $\langle u|u^\perp\rangle = 0$:

$$A R_0 A^{-1}|u^\perp\rangle = (2|u\rangle\langle u| - I)|u^\perp\rangle = \underline{-|u^\perp\rangle}$$

$\Rightarrow$ <u>Amplitude Amplification Algorithm</u>

1.) Setup $|u\rangle = A|0^m\rangle$

2.) Repeat $k = \mathcal{O}\left(\frac{1}{\sqrt{p}}\right)$ times:

a.) apply R_G / reflect through $|\psi_0\rangle$

b.) apply $A R_0 A^{-1}$ / reflect through $|u\rangle$

We end up with $\sin((2k+1)\theta)|\psi_1\rangle + \cos((2k+1)\theta)|\psi_0\rangle$

$(\theta = \arcsin(\sqrt{p}))$

$\Rightarrow p_k = \sin^2((2k+1)\theta)$

We can choose the integer $\tilde{k}$ closest to $\frac{\pi}{4\theta} - \frac{1}{2} \xrightarrow{p \ll 1} \frac{\pi}{4\sqrt{p}}$

$\left(= \mathcal{O}\left(\frac{1}{\sqrt{p}}\right)\right)$

We see that Grover's algorithm is a special case of this with $A = H^{\otimes n}$ which gives a <u>uniform</u> distribution $p = \frac{t}{N}$

<u>Application:</u> (basically any classical algorithm with search component

take e.g. the satisfiability problem:

We have a boolean function $\phi(i_1, \dots, i_n)$

<u>Goal:</u> Find $i = i_1 \dots i_n$ that satisfies $\phi(i) = 1$.

Classically: brute force $\mathcal{O}(2^n)$

Quantum: Grover $\mathcal{O}(2^{n/2})$

Implementation: $N = 2^n$; $x^i = \phi(i)$, $i \in \{0,1\}^n$

(assume that we can calculate $\phi(i)$ in polynomial time)

Start with $|i,0,0\rangle \xmapsto{U_\phi} |i, \phi(i), w_i\rangle$

workspace for computation

$$\Rightarrow O_\phi = U_\phi^{-1} (I^N \otimes Z^1 \otimes I^1)\, U_\phi$$

Z-gate for 1 qubit $\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$

$$O_\phi |i,0,0\rangle = U_\phi^{-1} (I^N \otimes Z^1 \otimes I^1) |i, \phi(i), w_i\rangle$$

$$= U_\phi^{-1} \begin{cases} -|i,\phi(i), w_i\rangle , & \phi(i) = 1 \\ |i,\phi(i), w_i\rangle , & \phi(i) = 0 \end{cases} = (-1)^{\phi(i)} |i,0,0\rangle$$

$\xrightarrow{\text{effectively}}$

$$O_\phi |i\rangle = (-1)^{\phi(i)} |i\rangle$$

Now we can just run Grover's algorithm.

While the quadratic speed up is nice if we only have the brute force approach, sometimes clever classical algorithms will outperform a Grover search. □

Traveling salesperson problem

n cities; $\binom{n}{2}$ (weighted) connections, i.e. distances; $(n-1)!$ paths

↳ find the shortest path

$$\text{brute force: } \mathcal{O}(n!) \Rightarrow \text{Grover: } \mathcal{O}(\sqrt{n!}) \xrightarrow[\text{Stirling approx.}]{n\gg 1} \mathcal{O}\left(\left(\tfrac{n}{e}\right)^{\frac{n}{2}}\right)$$

Bellman – Held – Karp dynamic programming algorithm:

(Grover not applicable!)

$$\mathcal{O}(2^n) \ll \mathcal{O}\left(\left(\tfrac{n}{e}\right)^{\frac{n}{2}}\right) \quad \text{for } n \gg 1$$

$\Rightarrow$ Classical solution <u>much faster</u> for large n.

But there are other QC algorithms with polynomial speed up!

Proof that Grover's algorithm is optimal

We look for a general algorithm.

Let's call the desired result/state $|x\rangle$ (We assume there is only one solution.)

Any ~~[illegible]~~ algorithm with T applications of the oracle will look like

$$U(T,x)|0^n\rangle = |\psi_T^x\rangle = U_T \underset{\text{oracle}}{O_x} U_{T-1} O_x \cdots U_1 O_x U_0 |0^n\rangle$$

U_k: arbitrary unitary step between oracle applications

We also define $|\psi_T\rangle = U_T \cdots U_0 |0^n\rangle$ i.e. only applying the unitary intermediate steps and not the oracle

w.l.o.g. $\langle x | \psi_k^x \rangle \geq 0$

Now, we will look at

$$d_{kx} = \| |\psi_k^x\rangle - |\psi_k\rangle \|$$

$$\Rightarrow d_{k+1,x} = \| |\psi_{k+1}^x\rangle - |\psi_{k+1}\rangle \|$$

$$= \| U_{k+1} O_x |\psi_k^x\rangle - U_{k+1} |\psi_k\rangle \| = \| O_x |\psi_k^x\rangle - |\psi_k\rangle \|$$

$$= \| O_x (|\psi_k^x\rangle - |\psi_k\rangle) + \underbrace{(O_x - I)}_{I - 2|x\rangle\langle x|} |\psi_k\rangle \| \underbrace{\leq}_{\text{triangle inequality}} d_{kx} + 2|\langle x | \psi_k \rangle|$$

$$d_{0x} = 0 \Rightarrow d_{kx} \leq 2k |\langle x | \psi_k \rangle|$$

Define $D_k = \frac{1}{N} \sum_x d_{kx}^2 \leq \frac{4k^2}{N} \sum_x |\langle x | \psi_k \rangle|^2 \underbrace{=}_{|x\rangle \text{ is a basis}} \frac{4k^2}{N}$

Why do we sum over all x?

The algorithm should be able to find any state $|x\rangle$, so we average over all possible states.

Let's consider

$$d_{kx} = \| |\psi_k^x\rangle - |x\rangle \| \Rightarrow d_{kx}^2 = \| |\psi_k^x\rangle \|^2 - 2\langle x|\psi_k^x\rangle + \| |x\rangle \|^2$$

$$= 2 - 2\underbrace{\langle x|\psi_k^x\rangle}_{\geq \sqrt{p}}$$

$\geq \sqrt{p}$, the minimum overlap with the correct state (if p is the required probability to find $|x\rangle$)

Define $A_k = \frac{1}{N}\sum_x d_{kx}^2 \leq 2(1-\sqrt{p})$

Finally

$$c_{kx} = \| |x\rangle - |\psi_k\rangle \| \Rightarrow c_{kx}^2 = \| |x\rangle \|^2 - 2\,\mathrm{Re}\langle\psi_k|x\rangle + \| |\psi_k\rangle \|^2$$

$$= 2(1 - \mathrm{Re}\langle\psi_k|x\rangle)$$

$$\geq 2(1 - |\langle\psi_k|x\rangle|) \qquad |\mathrm{Re}(x)| \leq |x|$$

Define $C_k = \frac{1}{N}\sum_x c_{kx}^2 \geq 2 - \frac{2}{N}\sum_x |\langle\psi_k|x\rangle| \overset{\text{Cauchy-Schwarz}}{\geq} 2 - \frac{2}{\sqrt{N}}\sqrt{\sum_x |\langle\psi_k|x\rangle|^2}$

$$= 2\left(1 - \frac{1}{\sqrt{N}}\right) \geq 1$$

for $N \geq 4$

Furthermore:

$$d_{kx} = \| |\psi_k^x\rangle - |\psi_k\rangle \| = \| (|\psi_k^x\rangle - |x\rangle) + (|x\rangle - |\psi_k\rangle) \|$$

reverse triangle inequality $\rightarrow \geq a_{kx} - c_{kx}$

$$\Rightarrow D_k = \frac{1}{N} \sum_x d_{kx}^2 \geq \frac{1}{N} \sum_x a_{kx}^2 - \frac{2}{N} \sum_x a_{kx} c_{kx} + \frac{1}{N} \sum_x c_{kx}^2$$

Cauchy-Schwarz

$$\geq A_k - 2\sqrt{A_k C_k} + C_k = (-\sqrt{A_k} + \sqrt{C_k})^2$$

$$\Rightarrow \frac{4k^2}{N} \geq \left(1 - \sqrt{2}\sqrt{1-\sqrt{p}}\right)^2$$

$$\Rightarrow k \geq \sqrt{N} \; \frac{1 - \sqrt{2}\sqrt{1-\sqrt{p}}}{2} \; \hat{=} \; \mathcal{O}(\sqrt{N}) \text{ for reasonably high } p$$

Compare to Grover: $k \approx \frac{\pi}{4}\sqrt{N}$ $\Rightarrow$ within a factor of a few from the ideal result.

$\Rightarrow$ Grover gives us good (and almost ideal) speed up for brute force searches but not revolutionary decrease in run time.
Due to technical reasons, it is debated if this algorithm can be widely applied (unordered databases, evaluations for the oracle, state preparation / preparation of superposition