Quantum Algorithms
Lecture 14
Definition of Quantum Computation I

Zhejiang University

Computation by quantum circuits

Solving problems

Until now, we have been describing the work of a quantum computer. Now it is time to define when this work leads to the solution of problems that are interesting to us. The definition will resemble the definition of probabilistic computation.

Computing with probabilities

Consider a function $F\colon B^n \to B^m$. We examine a quantum circuit operating with n bits, $U = U_L \cdots U_2 U_1\colon B^{\otimes N} \to B^{\otimes N}$. Loosely speaking, this circuit computes F if, after having applied U to the initial state $|x, 0^{N-n}\rangle$ and “having looked” at the first m bits, we “see” $F(x)$ with high probability. (The remaining qubits can contain arbitrary garbage.)

Nature of probability

We only need to discuss the nature of that probability. The precise meaning of the words “having looked” and “see” is that a measurement of the values of the corresponding qubits is performed. Several different answers can be obtained as the result of this measurement, each with its own probability. Later (after some two lectures) this question will be considered in more details.

Computing

To give a definition of quantum computation of a function F, it suffices (without injecting physical explanations of this fact) to accept the following: the probability of getting a basis state x in the measurement of the state $|\psi\rangle = \sum_x c_x|x\rangle$ equals

$$P(|\psi\rangle, x)=|c_x|^2$$

We are interested in the probability that the computer will finish its work in a state of the form $(F(x), z)$, where z is arbitrary.

Definition

The circuit $U = U_L \cdots U_2 U_1$ computes F if for any x we have

$$\sum_z |\langle F(x), z|U|x, 0^{N-n}\rangle|^2 \geq 1 - \varepsilon$$

where ε is some fixed number smaller than $1/2$. (Note that $F(x)$ and x consist of different numbers of bits, although the total lengths of $(F(x), z)$ and $(x, 0^{N-n})$ must be equal to N.)

Error probability

Just as for probabilistic computation, the choice of ε is unimportant, inasmuch as it is possible to effect several copies of the circuit independently and to choose the result that is most frequently obtained. From the estimate of probabilistic amplification (Lecture 6) it follows that in order to decrease the probability of failure by a factor of a, we need to take $k = \Theta(\log a)$ copies of the circuit U.

Error probability

The choice of the most frequent result is realized by a classical circuit, using the majority function $MAJ(x1, \ldots, xk)$ (which takes value 1 when more than half of its arguments equal 1 and value 0 otherwise). The function $MAJ(x1, \ldots, xk)$ can be realized over a complete basis by a circuit of size $O(k)$. Therefore, the a-fold reduction of the error probability is achieved at the cost of increasing the circuit size by the factor $O(\log a)$.

MAJ quantumly

We can incorporate MAJ into quantum circuit. We may use the function $MAJ_{\oplus}$ realized by a reversible circuit, so that its input bits are the output qubits of k copies of the circuit U.

Probability of approximation

Suppose that each gate of the circuit U_k is approximated by $\widetilde{U}_k$ with precision δ. Prove that the resulting circuit $\widetilde{U} = \widetilde{U}_L \cdots \widetilde{U}_2 \widetilde{U}_1$ satisfies the inequality

$$\sum_z |\langle F(x), z | U | x, 0^{N-n} \rangle|^2 \geq 1 - \varepsilon$$

with ε replaced by $\tilde{\varepsilon} = \varepsilon + 2L\delta$.

Comparing effectiveness

Now that we have the definition of quantum computation, we can make a comparison of the effectiveness of classical and quantum computing. In the Introduction we mentioned three fundamental examples where quantum computation appears to be more effective than classical. We begin with the example where the greater effectiveness of quantum computation has been proved (although the increase in speed is only polynomial).

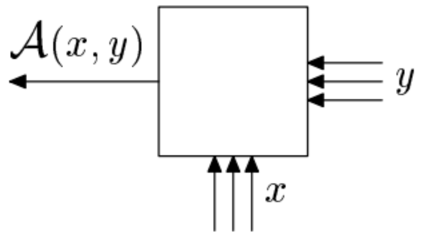
Quantum search: Grover's algorithm

Search with oracle

We will give a definition of a general search problem in classical and quantum formulations. It belongs to a class of computational problems with oracle, in which the input is given as a function (called a “black box”, or an oracle) rather than a binary word.

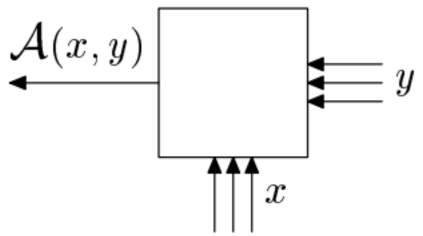
Search device

Suppose we have a device (see the diagram) that receives inputs x and y and determines the value of some predicate $A(x,y)$. We are interested in the predicate $F(x) = \exists y A(x,y)$. This resembles the definition of the class NP, but now the internal structure of the device calculating the predicate A is inaccessible to us. Under such conditions, it is not possible to complete the calculation faster than in $N = 2^n$ steps on a classical computer, where n is the number of bits in the binary word y.



Theorem

The problem can be formulated even without x: we need to compute the value of the "functional" $F(A) = \exists y A(y)$. If x is present, we can regard it as a part of the oracle, i.e., replace A with the predicate A_x such that $A_x(y) = A(x, y)$. Then $F(A_x) = F(x) = \exists y A(x, y)$.



Remark

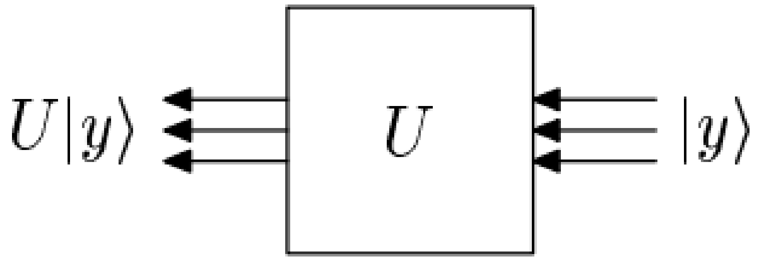
The version of the problem without x has another interpretation, which is quite rigorous (unlike the analogy with NP). Let us think of A as a bit string: y is the index of a bit, whereas $A(y)$ is the value of that bit. Then $F(A) = \bigvee_{y \in B^n} A(y)$, the OR function with $N = 2^n$ inputs.

Quadratic speedup

It turns out that a quantum computer can determine the value of $F(A)$ and even find a y for which A(y) is satisfied, in time $O(\sqrt{N})$. The lower bound $\Omega(\sqrt{N})$ has also been obtained, showing that in this situation quantum computers give only a quadratic speed-up in comparison with classical ones.

Quantum query

In the quantum formulation, the problem looks as follows. A quantum computer can query the oracle by sending y so that different values of y may form superpositions, and the oracle will return superpositions accordingly.



Quantum query

Interestingly enough, the oracle can encode the answer into a phase factor. Specifically, our oracle (or “black box”) is defined as an operator U acting by the rule

$$U|y\rangle = \begin{cases} |y\rangle & \text{if } \mathcal{A}(y) = 0, \\ -|y\rangle & \text{if } \mathcal{A}(y) = 1. \end{cases}$$

We assume that the computer can choose whether to query the oracle or not, which corresponds to applying the operator $\Lambda(U)$.

Search goal

The goal is to compute the value $F(A)$ and find an “answer” y for which $A(y)$ is satisfied. This should be done by a quantum circuit, using $\Lambda(U)$ as a gate (in addition to the standard basis).

Speedup – formally

The results that we have already mentioned are formulated as follows: there exist two constants C_1, C_2 such that there is a circuit of size $\leq C_1\sqrt{N}$, deciding the problem for an arbitrary predicate A; and, for an arbitrary circuit of size $\leq C_2\sqrt{N}$, there exists a predicate A for which the problem is not decided by this circuit (i.e., the circuit gives an incorrect answer with probability $> 1/3$).

Search for one element

We will construct a quantum circuit for a simplified version of the problem: we assume that the “answer” exists and is unique, and we denote it by y_0; we need to find y_0. The circuit will be described in terms of operator action on the basis vectors.

Two quantum operators

Consider two operators:

$$U = I - 2|y_0\rangle\langle y_0|$$

$$V = I - 2|\xi\rangle\langle\xi|$$

where $|\xi\rangle = \frac{1}{\sqrt{N}}\sum_y |y\rangle$

Operator U

The operator U is given to us (it is the oracle).

Example for 2 qubits with last element ($|11\rangle$) representing a solution.

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix}$$

Operator V

The operator V is represented by the matrix (recall that $N = 2^n$):

$$V = \begin{pmatrix} 1-\frac{2}{N} & \cdots & -\frac{2}{N} \\ \vdots & \ddots & \vdots \\ -\frac{2}{N} & \cdots & 1-\frac{2}{N} \end{pmatrix}$$

This operator is also known as diffusion transformation.

$$2\begin{pmatrix} \frac{1}{N} & \cdots & \frac{1}{N} \\ \vdots & \ddots & \vdots \\ \frac{1}{N} & \cdots & \frac{1}{N} \end{pmatrix} - I \qquad \begin{pmatrix} -0.5 & 0.5 & 0.5 & 0.5 \\ 0.5 & -0.5 & 0.5 & 0.5 \\ 0.5 & 0.5 & -0.5 & 0.5 \\ 0.5 & 0.5 & 0.5 & -0.5 \end{pmatrix}.$$

Realizing V

Let us realize V by a quantum circuit. We will proceed as follows: we transform $|\xi\rangle$ to $|0^n\rangle$ by some operator W, then apply the operator $I - 2|0^n\rangle\langle 0^n|$, and finally apply W^{-1}.

Realizing W

It is easy to construct an operator W that takes $|\xi\rangle$ to $|0^n\rangle$. The following will do: $W = H^{\otimes n}$, where H is the Hadamard gate from the standard basis. In fact $|\xi\rangle = \frac{1}{\sqrt{2^n}}(|0\rangle + |1\rangle)^{\otimes n}$, and H: $\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \rightarrow |0\rangle$.

$$H = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

Realizing V

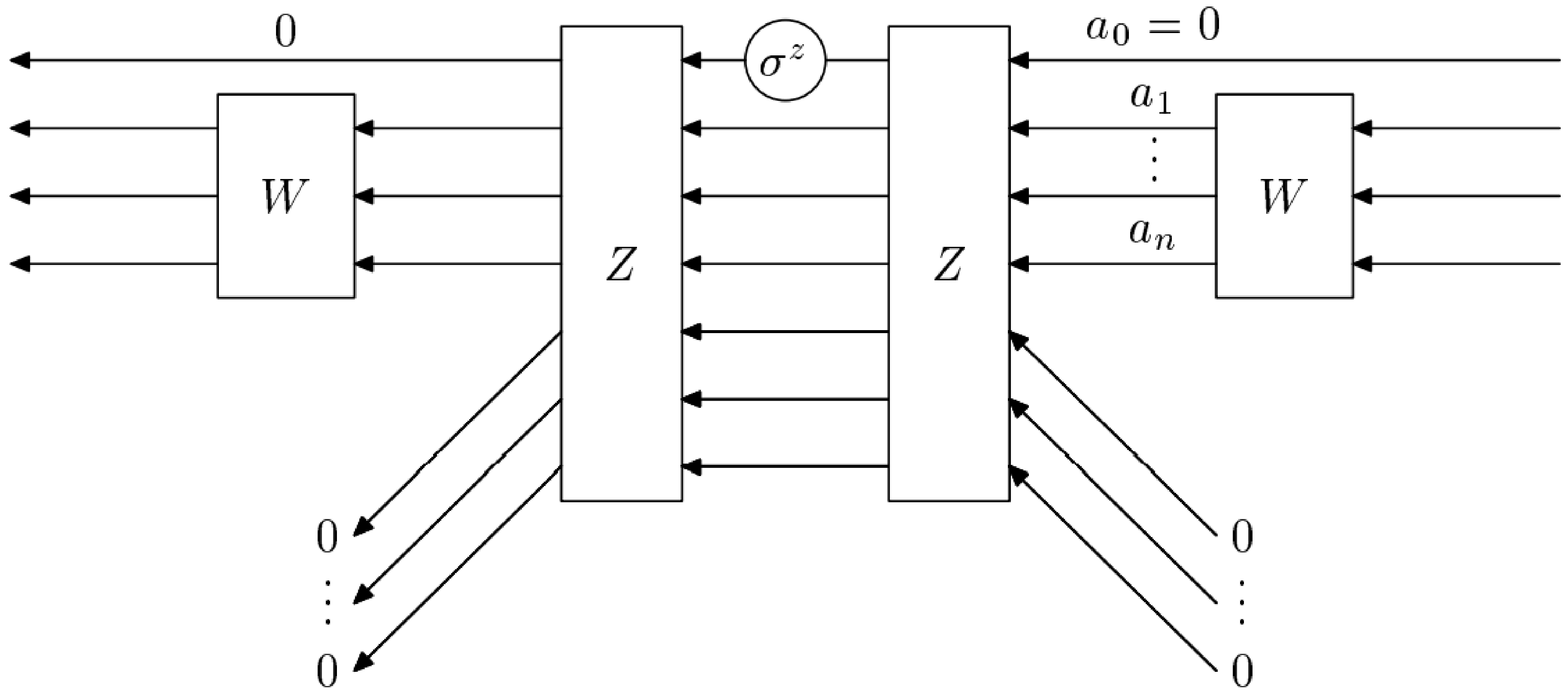
Now we have to implement the operator $I - 2|0^n\rangle\langle 0^n|$. We will use a reversible classical circuit that realizes the operator $Z\colon B^{n+1} \to B^{n+1}$,

$$Z|a_0, \ldots, a_n\rangle = |a_0 \oplus f(a_1, a_2, \ldots, a_n),\, a_1, \ldots, a_n\rangle;$$

$$f(a_1, \ldots, a_n) = \begin{cases} 1 & \text{if } a_1 = \cdots = a_n = 0, \\ 0 & \text{if } \exists\, j : a_j \neq 0. \end{cases}$$

(Up to a permutation of the arguments, $Z = \widehat{f_\oplus}$.) Since f has Boolean circuit complexity $O(n)$, Z can be realized by a reversible circuit of size $O(n)$.

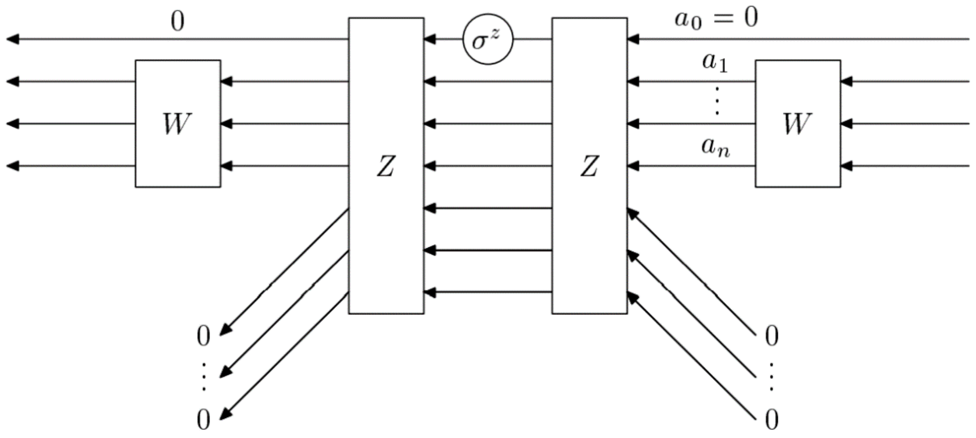
Realizing V



Realizing V

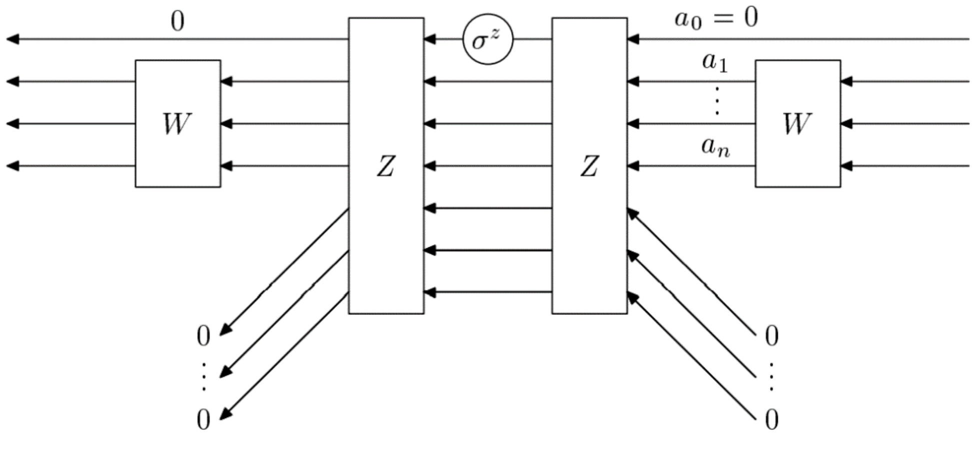
The central portion, incorporating Z, σ^z and Z, realizes the operator $I - 2|0^n\rangle\langle 0^n|$. In this circuit, the operator $\sigma^z = K^2$ (K from the standard basis) is used.

$$K = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix}$$



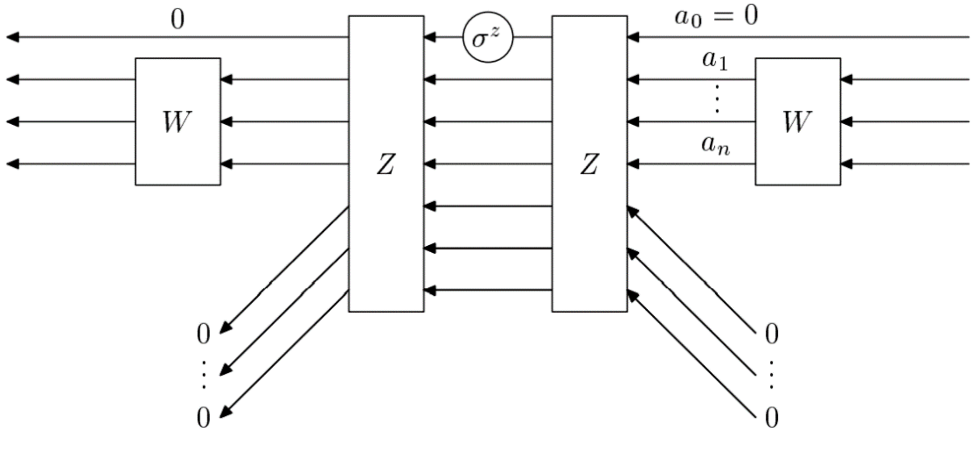
Realizing V

We note that W^2 and Z^2 act trivially (as the identity operator) on vectors with zero-valued borrowed qubits. Therefore the decisive role is played by the operator σ^z acting on an auxiliary qubit, which likewise returns to its initial value 0 in the end.



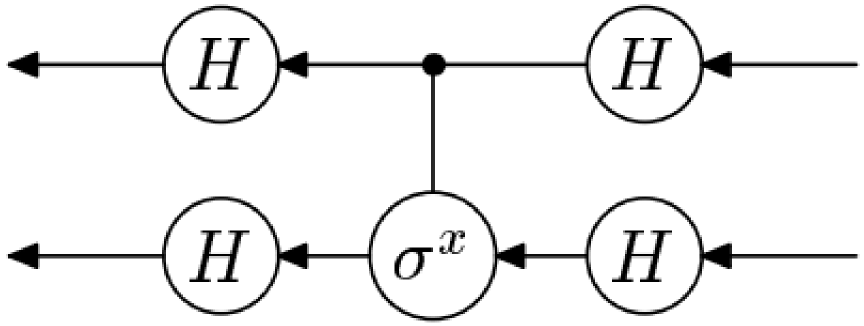
Realizing V

We must not be confused by the fact that although σ^z acts only on an $f_\oplus$-controlled qubit, the whole vector changes as a result. In general, the distinction between “reading” and “writing” in the quantum case is not absolute and depends on the choice of basis. Let us give a relevant example.



Reading vs writing

Let us find the matrix of $\Lambda(\sigma^x)\colon |a,b\rangle \to |a, a \oplus b\rangle$ relative to the basis $\frac{1}{\sqrt{2}}(|0\rangle \pm |1\rangle)$ for each of the qubits. In other words, we need to write the matrix of the operator $X = (H \otimes H)\Lambda(\sigma^x)(H \otimes H)$ relative to the classical basis. The circuit for this operator is shown here:



Reading vs writing

Using the equality $\Lambda(\sigma^x)|c,d\rangle = |c, c \oplus d\rangle$, we find the action of X on any basis vector:

$$\begin{aligned}
X|a,b\rangle &= \frac{1}{2}(H \otimes H)\Lambda(\sigma^x)\sum_{c,d}(-1)^{ac+bd}|c,d\rangle \\
&= \frac{1}{2}(H \otimes H)\sum_{c,d}(-1)^{ac+bd}|c, c \oplus d\rangle \\
&= \frac{1}{4}\sum_{a',b',c,d}(-1)^{a'c+b'(c+d)}(-1)^{ac+bd}|a',b'\rangle \\
&= \frac{1}{4}\sum_{a',b'} 2\delta_{b,b'} \cdot 2\delta_{a,\, a'\oplus b'}|a',b'\rangle = |a \oplus b, b\rangle.
\end{aligned}$$

Reading vs writing

Thus, in the basis $\frac{1}{\sqrt{2}}(|0\rangle \pm |1\rangle)$, the controlling and the controlled qubits have changed places. Which bit is “controlling” (is “read”) and which is “controlled” (is “modified”) depends on the choice of basis. Of course, such a situation goes against our classical intuition. It is hard to imagine that by passing to a different basis, a quantum printer suddenly becomes a quantum scanner.

Grover – first step

Let us return to the construction of a circuit for the general search problem. What follows is the main part of Grover’s algorithm. The oracle was $U = I - 2|y_0\rangle\langle y_0|$ given to us, and we have realized the operator $V = I - 2|\xi\rangle\langle\xi|$. We start computation with the vector $|\xi\rangle$, which can be obtained from $|0^n\rangle$ by applying the operator W.

Grover’s search

Now, with the aid of the operators U and V, we are going to transform $|\xi\rangle$ to the solution vector $|y_0\rangle$. For this, we will apply alternately the operators U and V:
$\cdots\ VUVU|\xi\rangle = (VU)^s|\xi\rangle$.

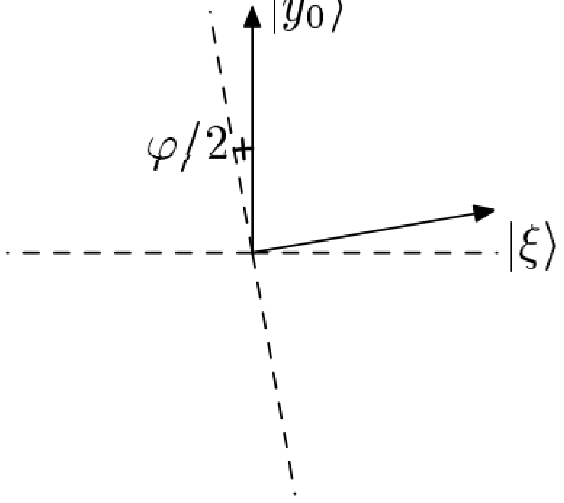
Analysis

$$(VU)^s|\xi\rangle$$

What do we get from this? Geometrically, both operators are reflections through hyperplanes. The subspace $L = C(|\xi\rangle, |y_0\rangle)$ is invariant under both operators, and thus, under VU. Since the initial vector $|\xi\rangle$ belongs to L, it suffices to consider the action of VU on this subspace.

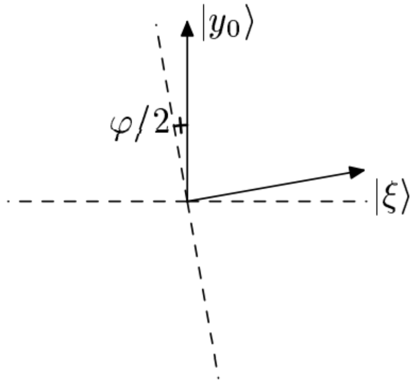
Analysis

The composition of two reflections with respect to two lines is a rotation by twice the angle between those lines. The angle is easy to calculate: $\langle\xi|y_0\rangle = \frac{1}{\sqrt{N}} = \sin\frac{\phi}{2}$, i.e., the lines are almost perpendicular. Therefore, we may write $VU = -R$, where R is the rotation by the small angle ϕ.



Analysis

But then $(VU)^s = (-1)^s R^s$, where R^s is the rotation through the angle $s\phi$. The sign does not interest us (phase factors do not affect probabilities). For large N, we have $\phi \approx 2/\sqrt{N}$. Then, after $s \approx (\pi/4)\sqrt{N}$ iterations, the initial vector is turned by an angle $s\phi \approx \pi/2$ and becomes close to the solution vector. This also indicates that the system ends up in the state $|y_0\rangle$ with probability close to one.

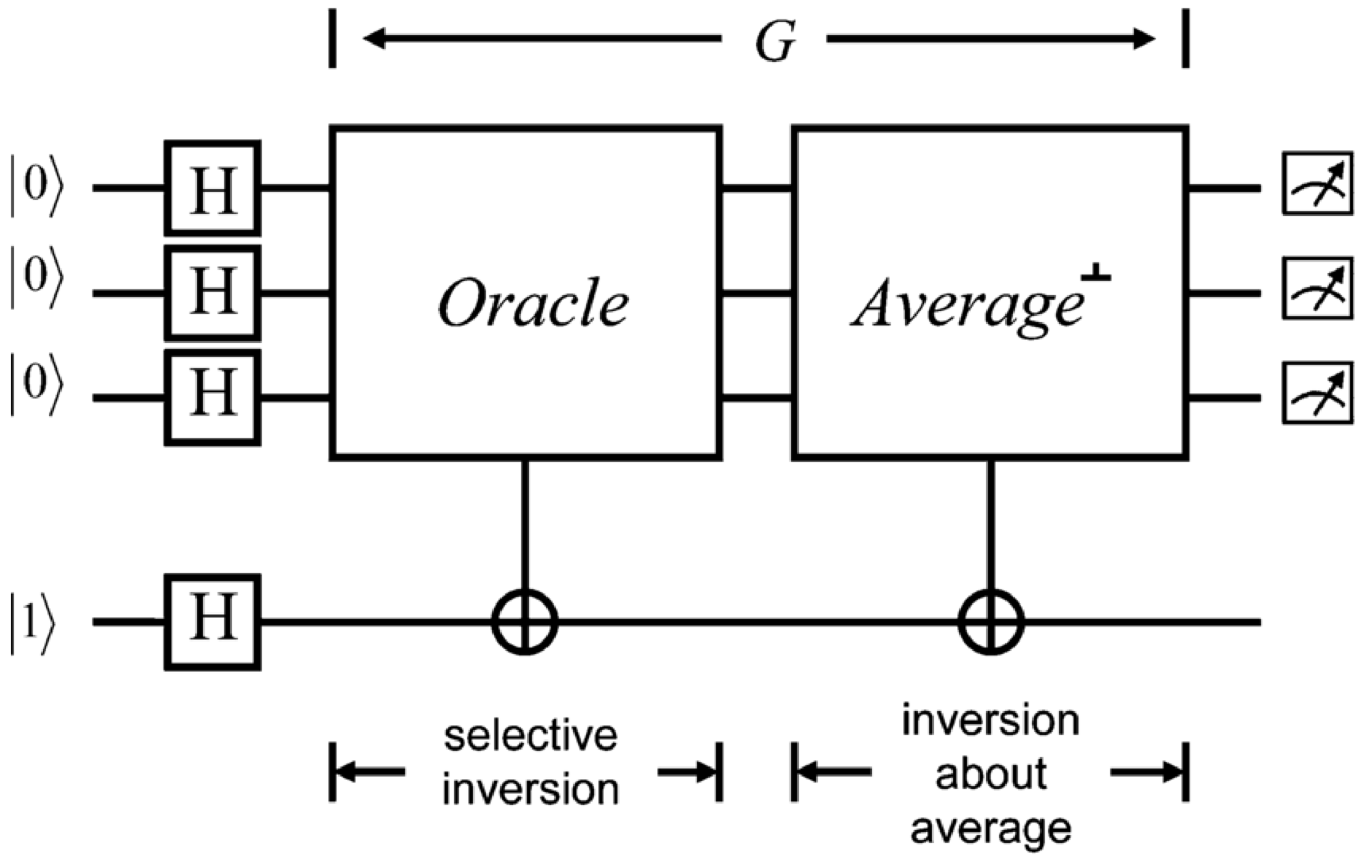


Analysis

To solve the search problem in the most general setting (when there may be several answers, or there may be none), additional technical devices are needed. Note that the number of steps for the rotation from the initial vector to some vector of the subspace spanned by the solution vectors is inversely proportional to the square root of the number of solutions.

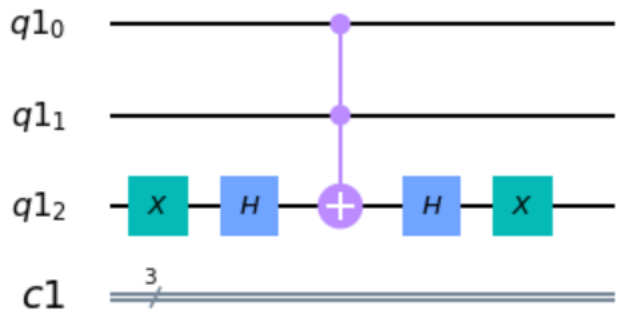
Grover’s search - implementation

Grover's search



Oracle query

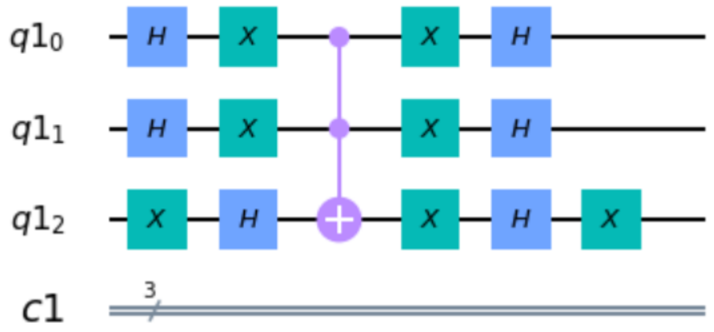
$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix}$$



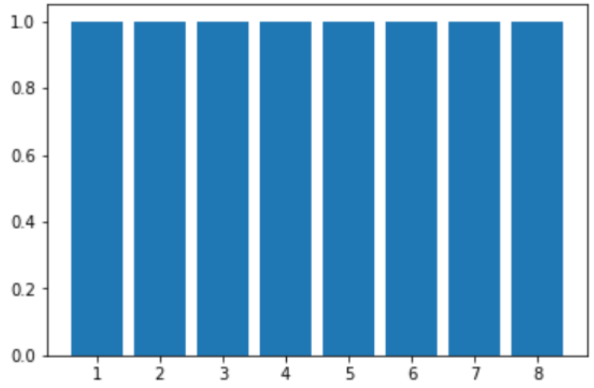
Diffusion (inversion)

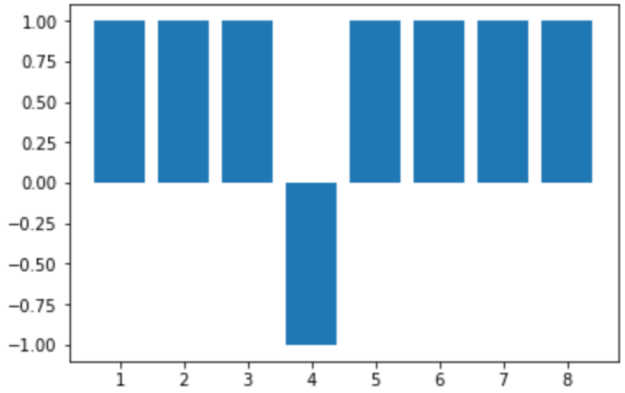
$$2\begin{pmatrix} \frac{1}{N} & \cdots & \frac{1}{N} \\ \vdots & \ddots & \vdots \\ \frac{1}{N} & \cdots & \frac{1}{N} \end{pmatrix} - I$$

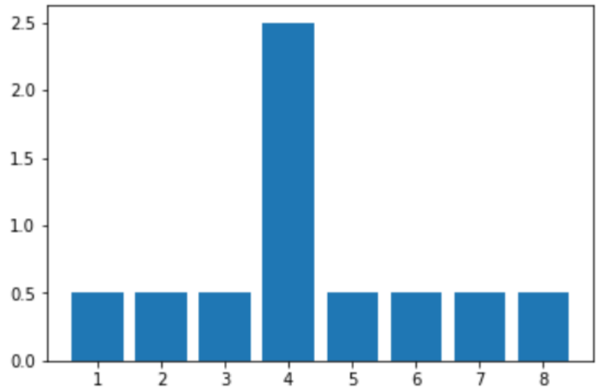
$$\begin{pmatrix} -0.5 & 0.5 & 0.5 & 0.5 \\ 0.5 & -0.5 & 0.5 & 0.5 \\ 0.5 & 0.5 & -0.5 & 0.5 \\ 0.5 & 0.5 & 0.5 & -0.5 \end{pmatrix}$$

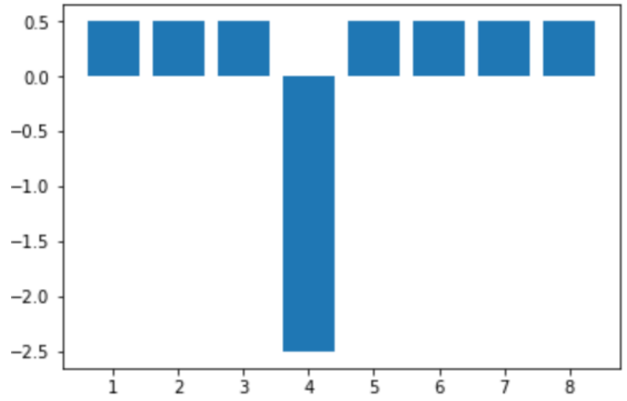


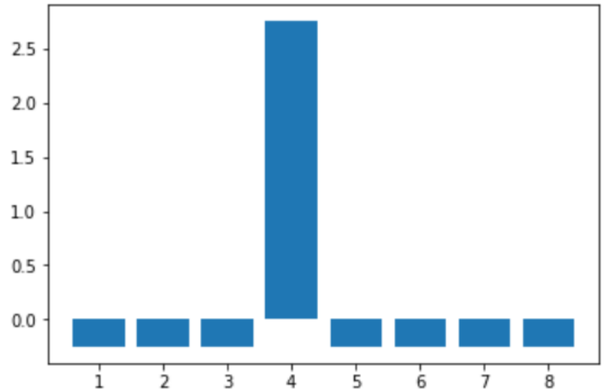
Grover’s search







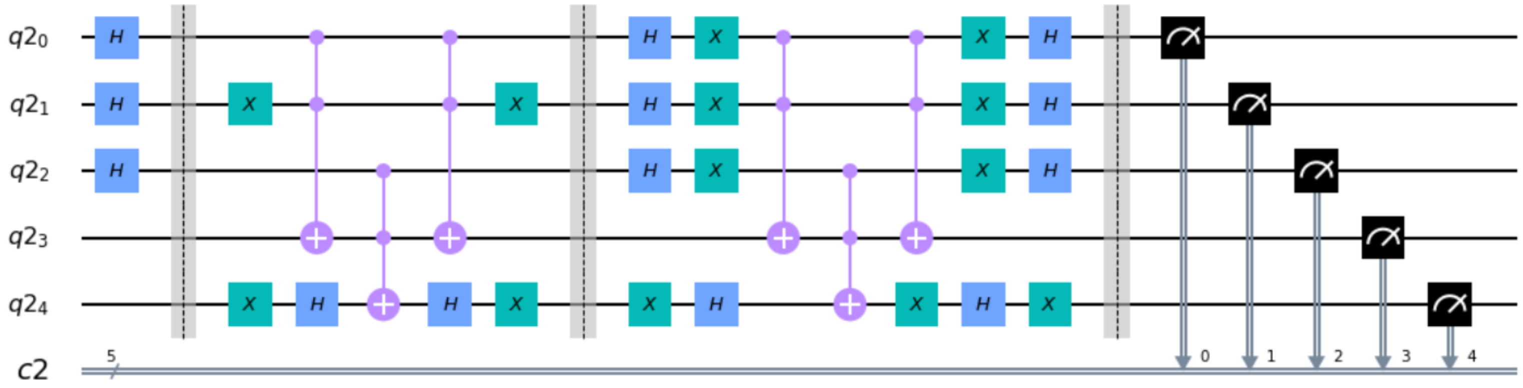


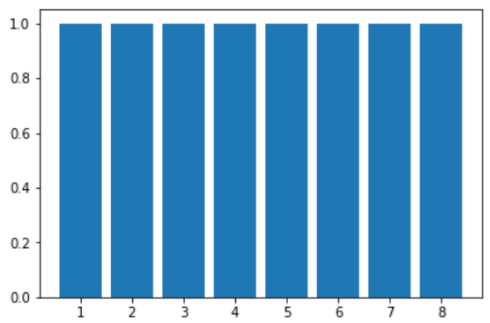


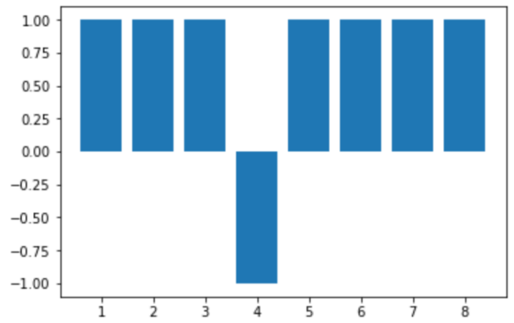
Grover’s search

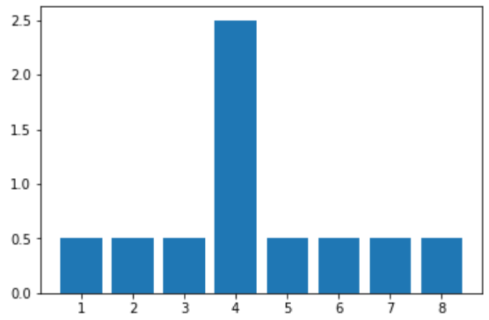
```
{'00001': 315, '00101': 7828, '00111': 307, '00010': 323, '00100': 304, '00000': 306, '00011': 304, '00110': 313}
```

```
Out[3]:
```









One qubit representation

The initial state is equal superposition:

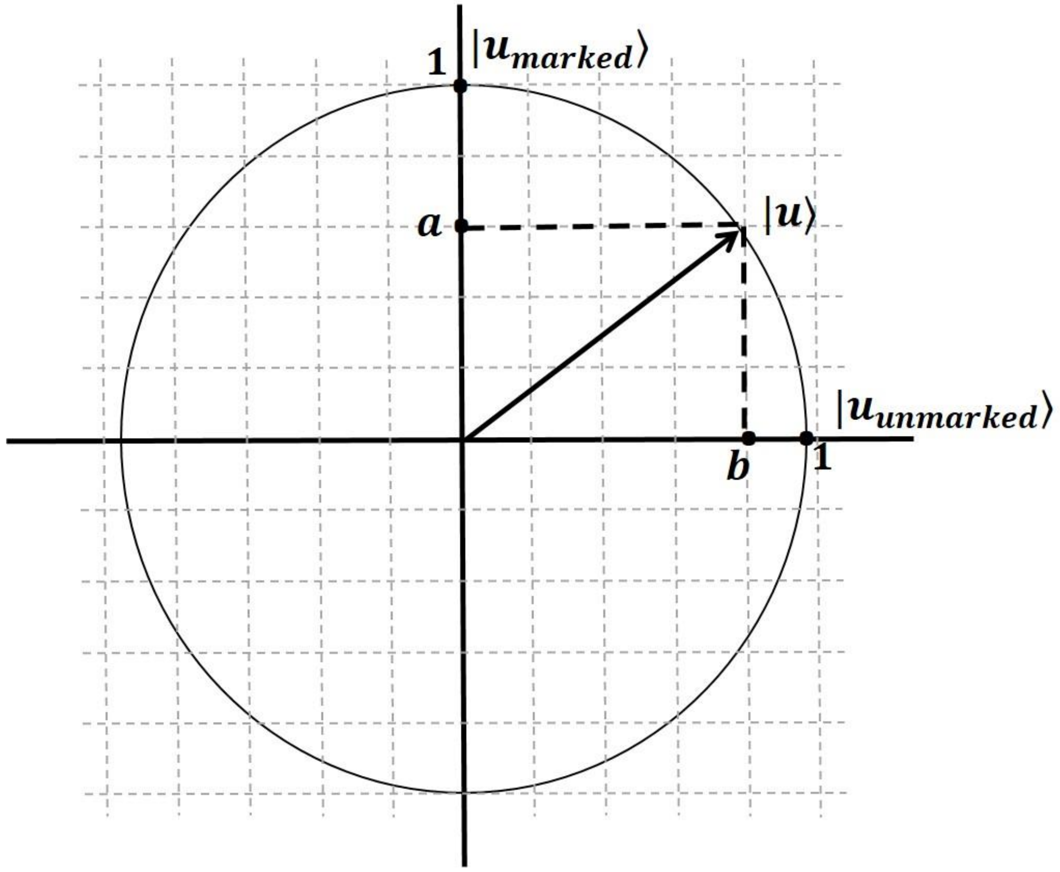
$|u >= \frac{1}{\sqrt{N}} \sum_{i=0}^{N-1} |i >$

We can group amplitudes of marked and unmarked elements to simplify analysis.

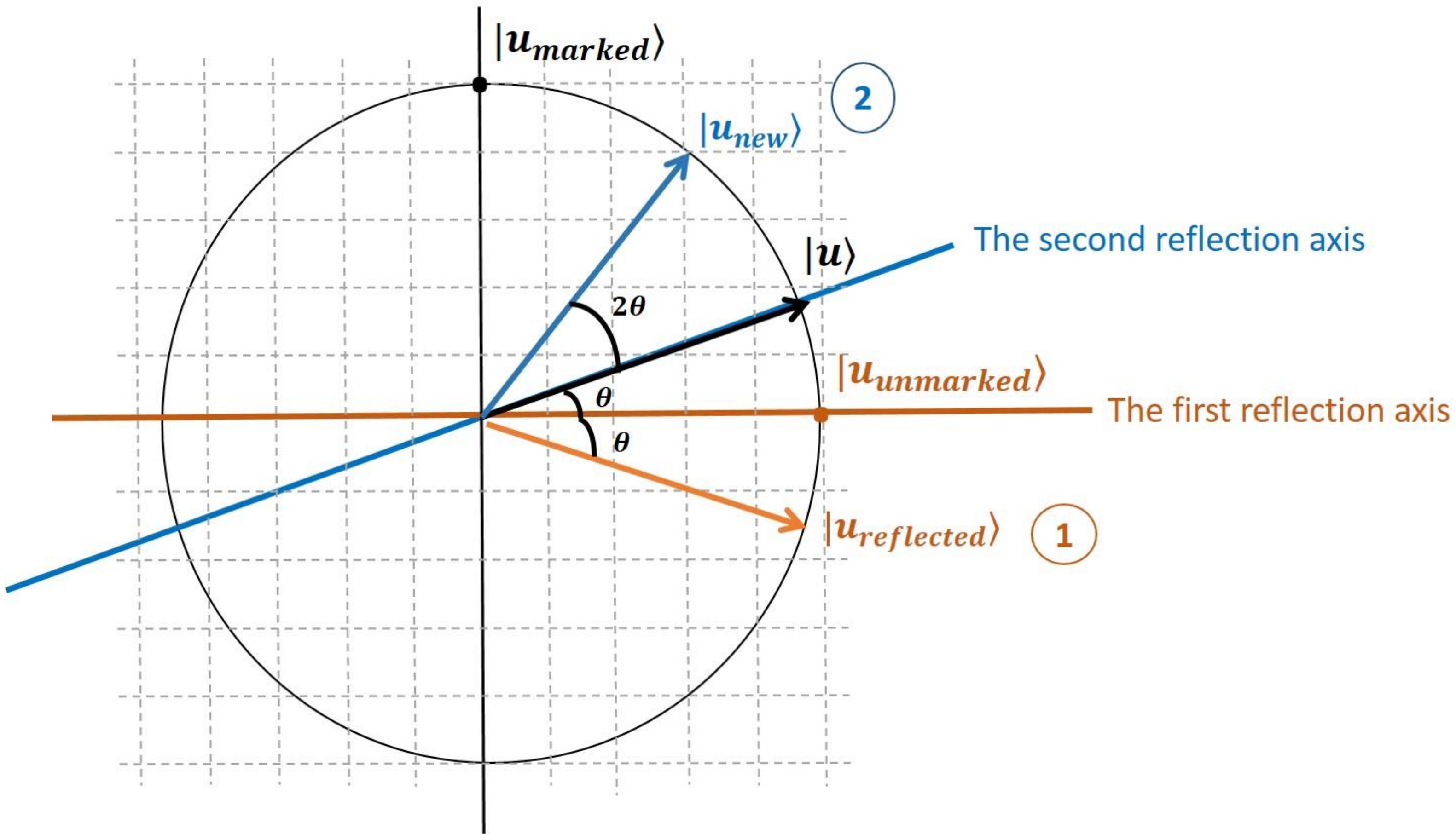
$|u >= a|u_{marked} > +b|u_{unmarked} >$

With 1 marked element $a = \frac{1}{\sqrt{N}}$.

One qubit representation



One qubit representation



Analysis – one element

For small θ: $\theta \approx sin\theta$.

We need to rotate by $\frac{\pi}{2}$.

$\theta \approx \frac{1}{\sqrt{N}}$, each iteration rotates by 2θ.

We need to perform $\frac{\pi/_2}{2/_{\sqrt{N}}} = \frac{\pi\sqrt{N}}{4}$ iterations.

Analysis – more elements

For one marked element we need to perform $\frac{\pi/2}{2/\sqrt{N}} = \frac{\pi\sqrt{N}}{4}$ iterations.

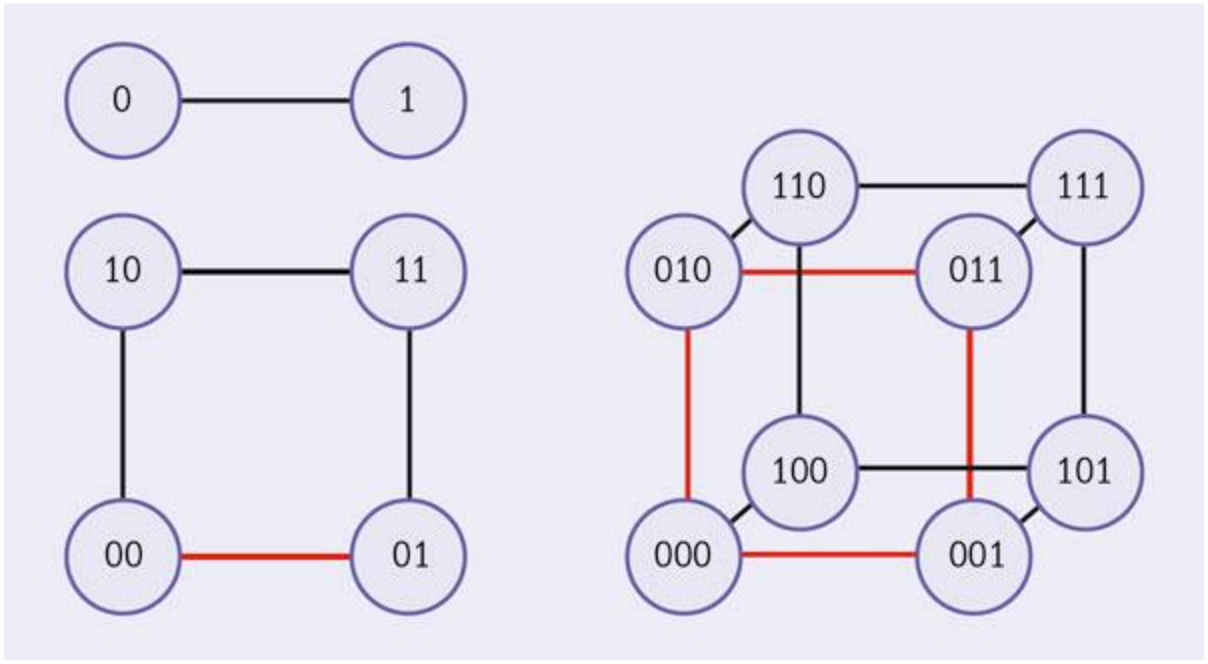
For k marked elements we need $\frac{\pi\sqrt{N}}{4\sqrt{k}}$ iterations.

For unknown number of marked elements try: $\frac{\pi\sqrt{N}}{4}, \frac{\pi\sqrt{N}}{4\sqrt{2}}, \frac{\pi\sqrt{N}}{4\sqrt{4}}, \frac{\pi\sqrt{N}}{4\sqrt{8}}, \frac{\pi\sqrt{N}}{4\sqrt{16}}, \dots$

Total number of iterations will be $O(\sqrt{N})$.

Space complexity

Each quantum state represents one index for one of the elements. Total number of qubits is n, when the total number of elements is $\mathrm{N} = 2^n$.

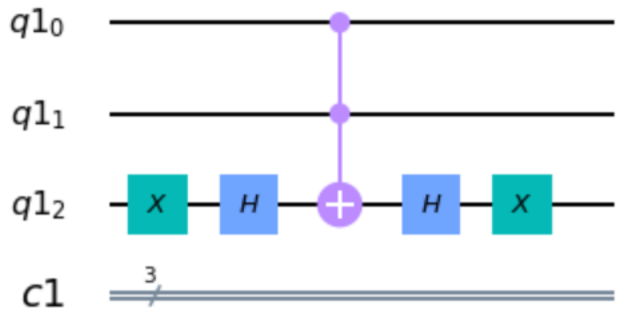


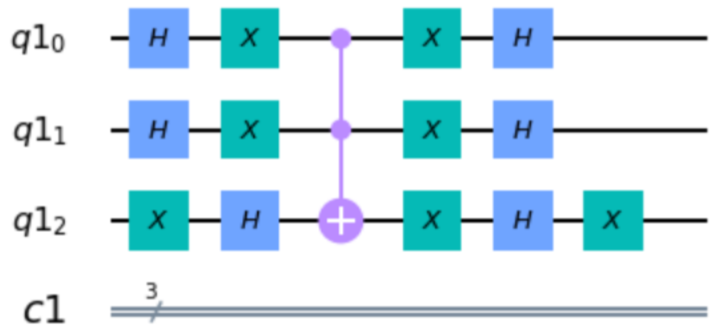
Time complexity

Total number of iterations is $O(\sqrt{N})$.

Each iteration has one query to the oracle and one diffusion operator.

Each mentioned operation has $const * n$ gates, $n \sim \log N$. Depth $\sim O(\sqrt{N})$, qubits $\sim \log N$, number of gates $\sim \sqrt{N \log N}$.





Time complexity

Grover cannot be made exact without losing the square-root speed-up.

Grover's search cannot be faster than $O(\sqrt{N})$.

Therefore, $O(\sqrt{N})$ time complexity is optimal.

Thank you for your attention!