Lecture 9: Unstructured search

Rajat Mittal

IIT Kanpur

We move to the next famous quantum algorithm, Grover's search algorithm. It gives a quantum way to search in an unstructured list/database. Grover's algorithm was discovered by Lov Grover in 1990's. Subsequently, many variations and other algorithms have been discovered on the basis of Grover's algorithm. We will see Grover's algorithm and some of its variations in this lecture.

As opposed to Shor's algorithm, it does not give an exponential speedup. Instead, its importance lies in the fact that the algorithm deals with a very general search problem. Since search problems are present in almost all areas (for instance, solving NP hard problems), this makes the algorithm useful in variety of applications.

1 Improved Elitzur-Vaidman

In the beginning of the course, we saw Elitzur-Vaidman as a way to find if a bomb is real without exploding it (Figure 1).

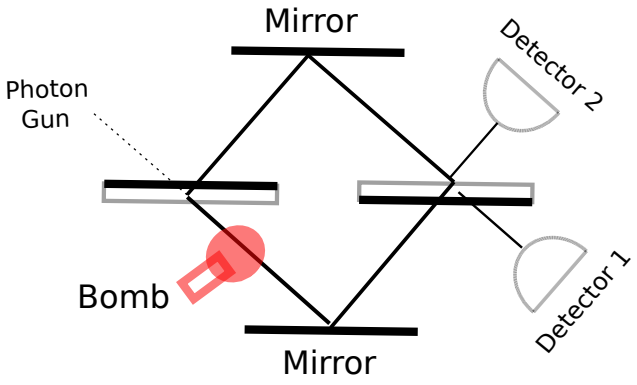
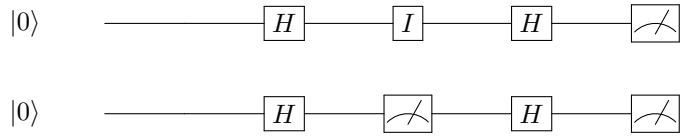


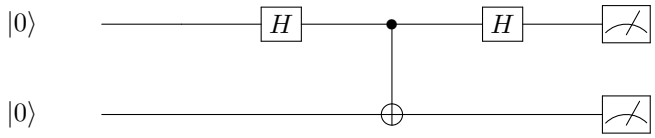
Fig. 1. Elitzur Vaidman bomb tester: how to test a bomb without exploding it.

The discussion on Mach-Zehnder showed that the beam splitters can be thought of as Hadamard gates and the bomb itself can be thought of as a measurement. In other words the circuit looks like (for bomb

being dud and real respectively) this.



From the principle of deferred measurement, we can move the intermediate measurement in the end for the bomb case.



Notice that the CNOT and measurement on the second register will be replaced by identity if the bomb is a dud.

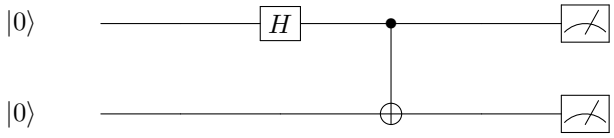
You can easily verify the properties of Elitzur-Vaidman bomb testing. If the controlled operation is identity, we will always obtain $|0\rangle$ (H is its own inverse). If the controlled operation is NOT, then there are two cases depending upon the state of the second qubit.

If second qubit is in $|1\rangle$, that means CNOT was applied and the bomb has blown. On the other hand if second qubit is $|0\rangle$, in the first qubit we get $|0\rangle$ and $|1\rangle$ with equal probability. If we get $|1\rangle$ in the first register, then we know that bomb is real and it has NOT exploded.

Exercise 1. Write the quantum state explicitly and check the above assertions.

The probability of not blowing the bomb and getting the correct answer is 1/4, you might not be happy about it. The task is to take this probability to close to 1, an improved Elitzur-Vaidman.

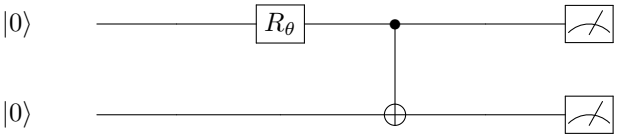
As a first step, let us analyze these circuits step by step. Till the first Hadamard, there is no difference. What about the difference in the states after the CNOT and measurement on the second register?



The probabilities can be easily computed,

$$\text{Dud} \to \begin{cases} 0 & \text{Measure 0 with probability } 1/2 \\ 1 & \text{Measure 1 with probability } 1/2 \end{cases} \quad \Big| \quad \text{Real} \to \begin{cases} 0 & \text{Measure 0 with probability } 1/2 \\ 1 & \text{Bomb explodes, second register in 1} \end{cases}$$

Already the probability of exploding is quite high. After the first Hadamard the state becomes $\frac{1}{\sqrt{2}}(|0\rangle+|1\rangle)$ and the $|1\rangle$ part destroys the bomb (with half the probability). The idea is to use operator R_θ which takes $|0\rangle$ to $(\cos\theta)|0\rangle + (\sin\theta)|1\rangle$ with a very small θ, that way the bomb goes off with very small probability (bomb going off means we measure the second register and got $|1\rangle$).

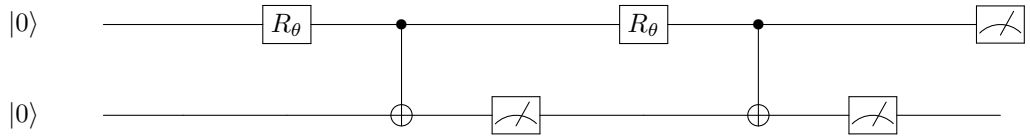


The probabilities will be,

$$\text{Dud} \to \begin{cases} 0 & \text{Measure 0 with probability } \ 1 \\ 1 & \text{Measure 1 with probability } \ \sin^2\theta \end{cases} \quad \Big| \quad \text{Real} \to \begin{cases} 0 & \text{Measure 0 with probability } \ 1 \\ 1 & \text{Bomb explodes with probability } \ \sin^2\theta \end{cases}$$

We are going to call this our basic circuit, even though it does not sound very helpful. First main idea/observation is that in the bomb case, measurement of second register resets the first register to $|0\rangle$. On the other hand, the state in the dud case is rotated to $(\cos\theta)|0\rangle + (\sin\theta)|1\rangle$. This means that the two cases are actually different.

If we perform the same circuit again, the bomb can explode with additional probability at most $\sin^2\theta$. Though, the state in the dud case is $(\cos 2\theta)|0\rangle + (\sin 2\theta)|1\rangle$. The two cases have started behaving differently.



The bomb will explode if we measure 1 in the second register in either of the two iterations of the basic circuit.

Exercise 2. Show that the bomb exploding probability is bounded by $2\sin^2\theta$.

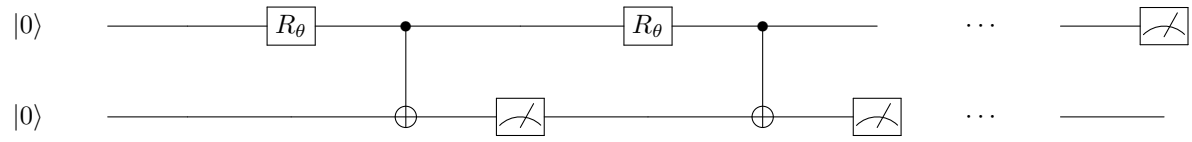
On the other hand, for the dud case, the state has changed to $(\cos 2\theta)|0\rangle + (\sin 2\theta)|1\rangle$ The probabilities become,

$$\text{Dud} \to \begin{cases} 0 & \text{Measure 0 with probability } 1 \\ 1 & \text{Measure 1 with probability } \sin^2 2\theta \end{cases} \quad \Big| \quad \text{Real} \to \begin{cases} 0 & \text{Measure 0 with probability } 1 \\ 1 & \text{Bomb explodes with probability } 2\sin^2\theta \end{cases}$$

If we repeat the basic circuit k times,

$$\text{Dud} \to \begin{cases} 0 & \text{Measure 0 with probability } 1 - \sin^2 k\theta \\ 1 & \text{Measure 1 with probability } \sin^2 k\theta \end{cases} \quad \Big| \quad \text{Real} \to \begin{cases} 0 & \text{Measure 0 with probability } 1 - k\sin^2\theta \\ 1 & \text{Bomb explodes with probability } k\sin^2\theta \end{cases}$$

Here comes the second main idea, $\sin^2 k\theta \approx k^2\theta^2$ reaches 1 much faster than $k\sin^2\theta \approx k\theta^2$. We will repeat the basic circuit $k = \frac{\pi}{2\theta}$ times to get $|1\rangle$ for the dud case.



Let us analyze all the cases in detail.

In case we don't have a bomb, the repeated rotation takes the first register to state $|1\rangle$ and then we can declare the bomb to be dud.

Exercise 3. Check that the above circuit will output 1 if the bomb is dud with probability 1.

In case we have the bomb, we might measure 0 in the first register (the first register gets reset every time to $|0\rangle$). Then we can declare that the bomb is real and has not exploded. The troublesome case is that the bomb is real and it explodes. The total probability of exploding is bounded by $k\sin^2\theta$. Since $\sin\theta \approx 1/k$, this probability goes to 0 with smaller θ.

Exercise 4. Can it happen that we measure $|1\rangle$ and the bomb is real?

The main analysis works because after k repetitions, probability of exploding increase much more slowly than the angle of rotation ($k\theta^2$ as compared to $k\theta$). So we can *quickly* go to $|1\rangle$ state in case of dud, keeping probability of exploding small. In one case the probability is amplified (explosion); in the other case the amplitude is amplified and we take the probability of that in the end (probability of measuring dud). The lesson is, probability grows much faster than amplitude.

Note 1. Similar idea will be used in Grover search.

Please refer to the following wiki article for details [2].

2 Search problem: finding an element in an unstructured database

The problem of *searching through an unstructured database* is ubiquitous and any improvement will help lot of applications. To take an example, consider that you want to find the roll number of a student through her name but the list is sorted on the basis of roll numbers (or unsorted, sorted on the basis of roll number does not help). We can apply binary search if the list is sorted; for an unstructured database, it seems that we should go through all the elements.

Before solving this problem on a quantum (or a classical) computer, let us formally define the unstructured database search problem.

Input: A list L with n elements, some of them are marked. L can be viewed as an element of $\{0,1\}^n$, where L_i is 1 iff i-th element is marked. We are also given an oracle to find whether an element is marked. Precisely, given an index of the list, the oracle tells you if the corresponding element is marked or not.

Output: Find a marked element using minimum number of queries to the oracle. To make it a decision problem, we can ask about the existence of a marked element.

We will assume two things for simplification. First, there is only one marked element. Second, we will assume that $n = 2^k$, implying that the index is a k bit string.

Exercise 5. Convince yourself that assuming $n = 2^k$ is not a significant assumption (since we are only interested in asymptotic complexity).

We can view list $L \in \{0,1\}^n$ as a truth table of a function on $\{0,1\}^k$ (because $n = 2^k$). Let $f : \{0,1\}^k \to \{0,1\}$ denote the function which tell us whether the index is marked or not (f is 1 if the index is marked and 0 otherwise). So, the action of the oracle is,

$$|x, b\rangle \to |x, b \oplus f(x)\rangle.$$

Here x is a k-bit string and b is a bit. This, like before, can be changed into an oracle,

$$O_f|x, b\rangle = (-1)^{f(x)\cdot b}|x, b\rangle.$$

Exercise 6. Do you remember how we did it? If not, go back and check.

Given such an oracle O_f, search problem is to find an x such that $f(x) = 1$. Like before, we will measure the complexity of our algorithm by the number of queries to oracle O_f. It can be shown that the time complexity is of similar order.

Let us take an example to show how the search problems is useful. Let x denote an assignment of variables for a 3-CNF formula C. It is *easy* to construct an oracle for f, where f indicates whether x is a satisfying assignment for C. Then, one approach to solve 3-SAT would be to search for a satisfying x. In general, this technique can be used for any NP-hard problem to search for a witness.

Importance of search problem: To solve a search problem, a classical algorithm needs to look through all the indices, and has to query oracle $\Theta(n)$ times. There is a formal way to prove that any randomized algorithm will take at least $\Omega(n)$ queries, we will cover it in a later lecture.

In contrast, Grover's algorithm finds the marked element in $O(\sqrt{n})$ queries. This might not seem like a big improvement (it is a quadratic advantage), but is important because of the usefulness of search. Many different problems in diverse areas have a brute force search algorithm. If we have the subroutine (oracle) to check the solution, we can improve the running time quadratically for all these problems on a quantum computer.

We already took one such example, NP-complete problems. Their solutions are easy to verify but difficult to find. The search for their solutions can be improved by a quadratic factor.

Exercise 7. Can it help in the brute force algorithm for factoring?

2.1 Randomized approach to searching

For a randomized algorithm, the trivial (and only possible) strategy seems to be to pick a random element and check if the element is marked.

Exercise 8. How many queries do we need to make such that the probability of acceptance is constant (remember that there is only one marked element)?

Even though this trivial randomized algorithm still take $\Theta(n)$ queries, we will see it in detail. It will provide us important clues on how to improve the search algorithm quantumly.

The randomized algorithm can be seen in steps/iterations. In every iteration, we toss a *coin* which has n possible outputs. We query the index given by the toss and check if it is marked. The algorithms stops as soon as we find the marked element. In this sense, if the element is not marked, the coin is tossed, and $1/n$ of the probability from there goes to the marked element and stays there (no toss once we get a marked element).

The following figure (Figure 2) captures the probabilities assigned to different indices at the end of the first and second iteration.

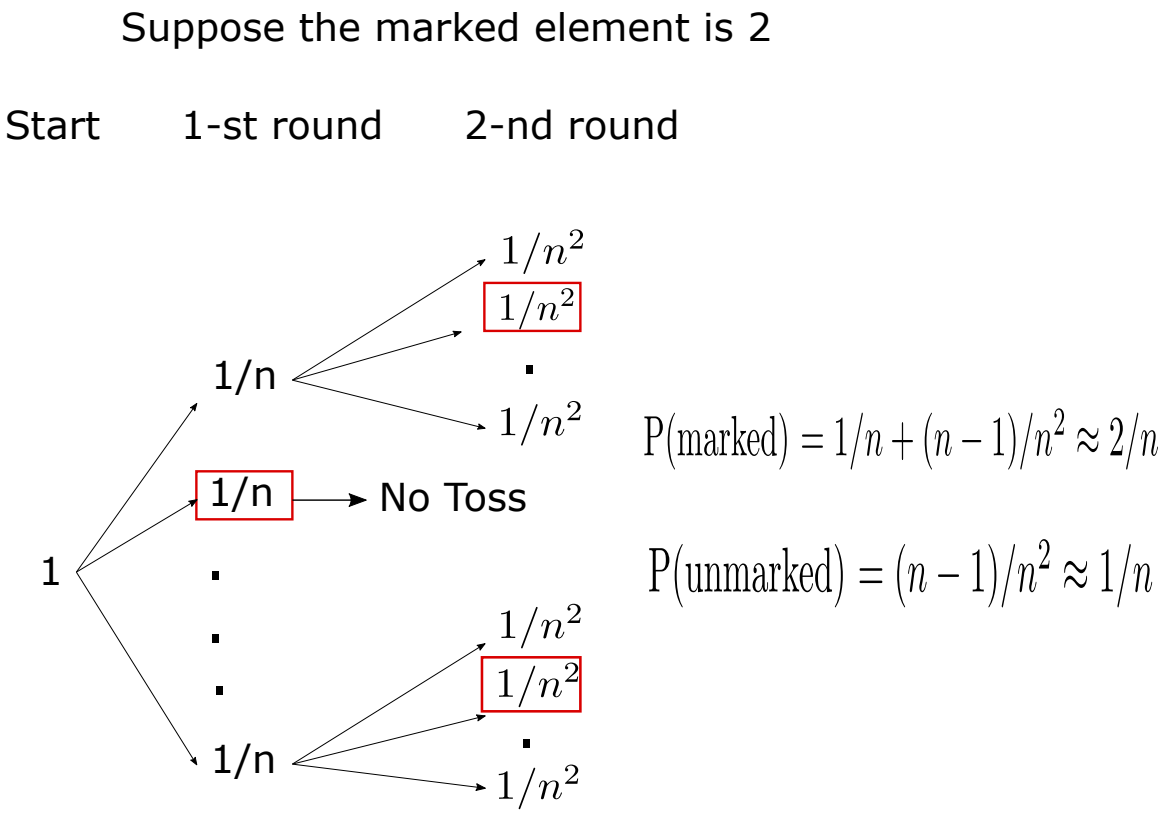


Fig. 2. Search problem

Since we stop after seeing the marked element. The probability of being in the marked element keeps increasing at the expense of small decrease in the probability of unmarked elements. After around $\Theta(1/n)$

iterations, the probability of being in the marked element becomes constant. In other words, we amplify the probability of being in the marked state till it becomes constant.

The change in probabilities can be plotted as a histogram (Figure 3).

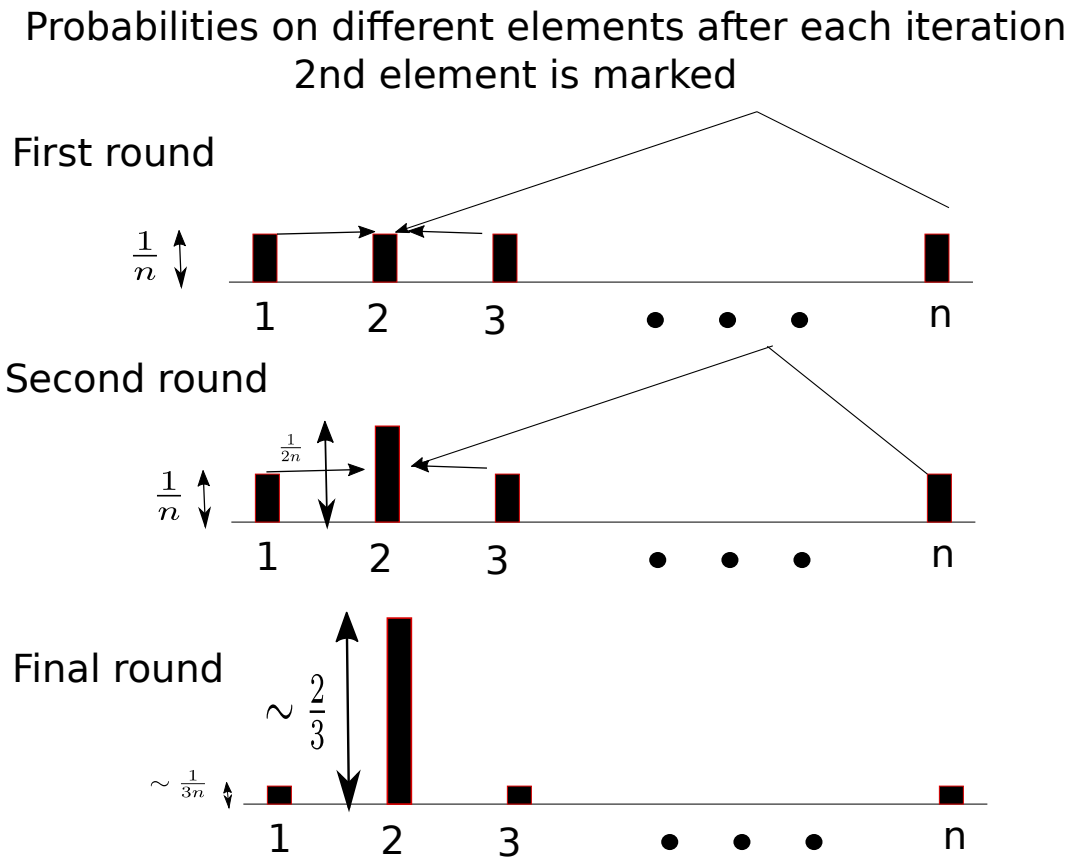


Fig. 3. Search problem

The same algorithm can be modified through our intuition from Elitzur-Vaidman. It is better to amplify amplitude rather than probability. Can we amplify the amplitude of being in the marked state in each iteration? This leads to the main idea behind Grover search.

2.2 Idea of Grover search

It will be helpful to name these three states. Since we assumed that there is only one marked element, say x^0, we are looking for the state,

$$|M\rangle = |x^0\rangle.$$

The remaining states (unmarked states, where we don't want to end up) has an equal superposition,

$$|U\rangle = \frac{1}{\sqrt{n-1}} \sum_{x \neq x^0} |x\rangle.$$

To start, we can prepare an equal superposition over all indices (how?), where each amplitude is $\sqrt{1/N}$,

$$|\psi\rangle = \frac{1}{\sqrt{n}} \sum_x |x\rangle.$$

Exercise 9. What is the probability that we get state $|x^0\rangle$, if we measure $|\psi\rangle$ in the standard basis?

If we measure right here, the probability of getting x^0 is $1/N$. We can repeat it; this would be the old randomized algorithm where we amplify probability. Instead, like Elitzur-Vaidman, we want to increase the amplitude without measuring.

We saw from last section that we would like to amplify amplitude on the marked state, $|x^0\rangle$. Comparing with the probability picture, we can draw a similar histogram for amplitude. Each entry of the histogram now represents amplitude on an index (instead of the probability).

After the first step of creating state $|\psi\rangle$, each amplitude will be $\sqrt{1/N}$. We would like to add around $\sqrt{1/N}$ amplitude to the marked state in each iteration, getting to constant amplitude in $O(\sqrt{N})$ iterations (Figure 4). If each iteration takes constant many queries, we get a search algorithm with $O(\sqrt{N})$ many queries!

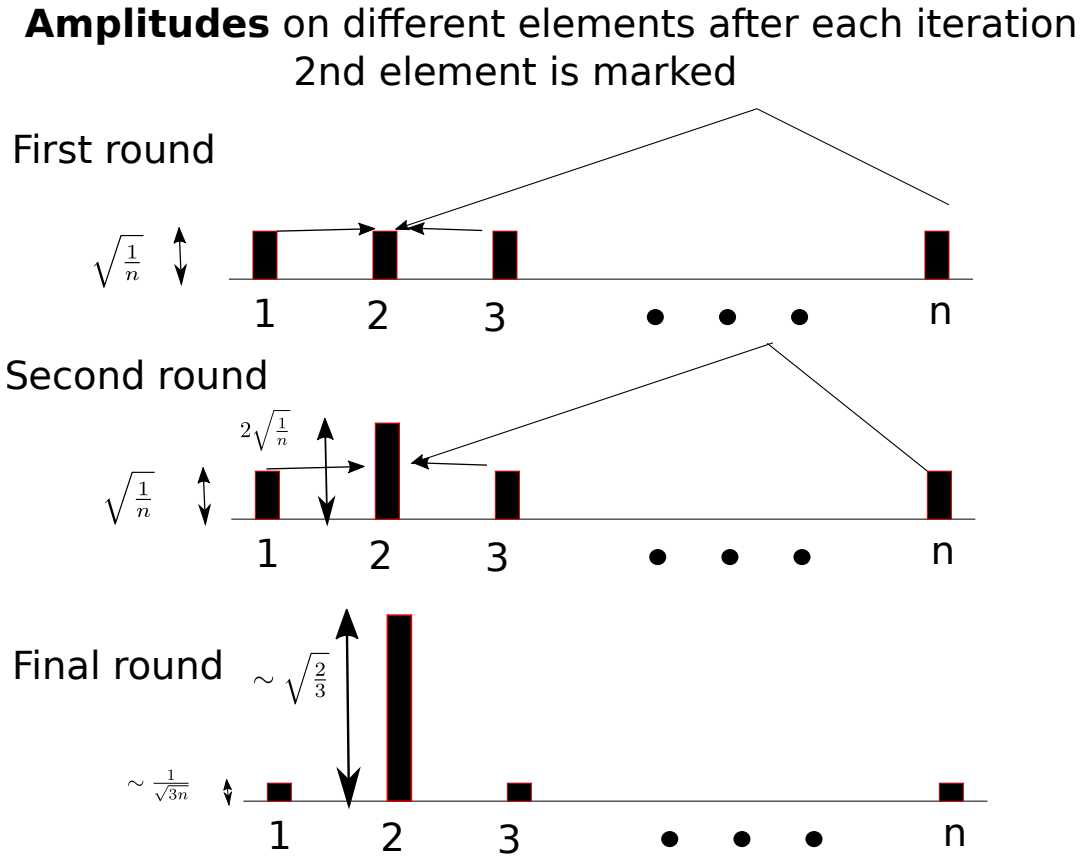


Fig. 4. Search problem

Note 2. This amplitude amplification for the dud case is done by rotation in Elitzur-Vaidman. There we needed to rotate in the standard basis. Unfortunately, here we would like to rotate in the basis of states $|M\rangle$ and $|U\rangle$. These states are not known to us, and are only accessible through the oracle.

What is the action of oracle on such a superposition. Since we get a phase of -1 on $|x^0\rangle$ and everything else remains the same, this is indeed a reflection (not rotation) on the amplitudes (Figure 5).

Note 3. If you want to view this in the state space and not a histogram, this is a reflection around the axis $|U\rangle$.

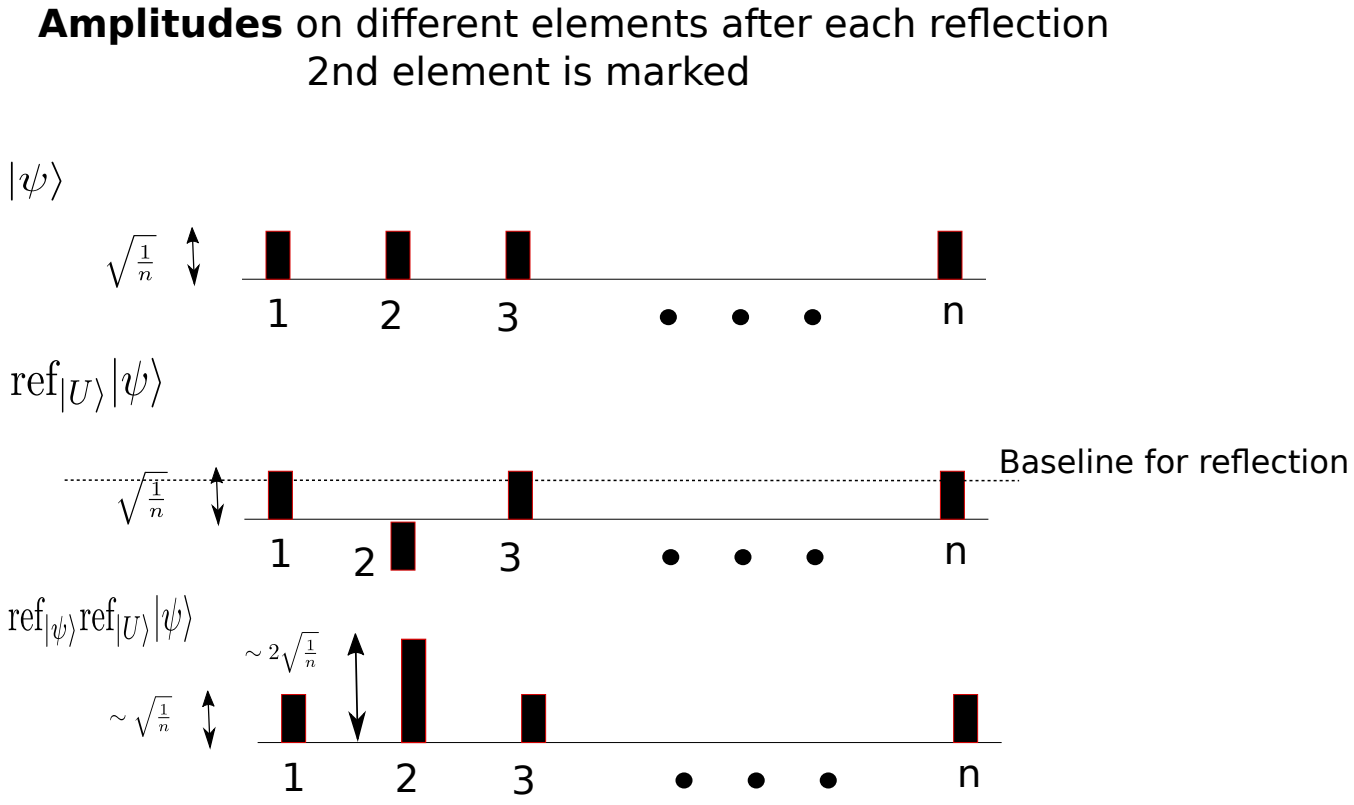


Fig. 5. Search problem

How do we get to rotation from this reflection. Looking at the histogram picture (Figure 5), if we can reflect around a slightly different baseline again, the amplitude on the marked state will be amplified. Indeed we will show in Section 2.3 that the product of two reflections is a rotation. For now, remember that the product of two reflections is a rotation by 2θ, where θ is the angle between the two axes of reflection.

Exercise 10. Try giving a picture proof of the above statement.

In other words, if we can perform another reflection where the axis make an angle around $\theta \sim \sqrt{1/N}$ with the state $|U\rangle$, we will be done. The state $|\psi\rangle$ is at an angle $\theta \approx \sin(\theta) = \sqrt{1/N}$ (Figure 6).

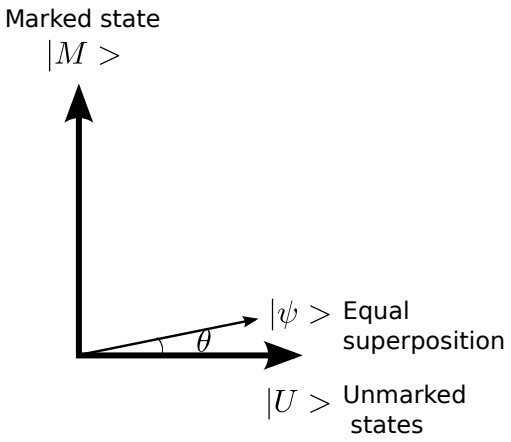


Fig. 6. Search problem

We should just reflect around $|\psi\rangle$. In the histogram picture, this would amount to reflection around the average of the amplitudes. You will prove in the assignment that reflection around $|\psi\rangle$ is actually the reflection around the average in the histogram picture (Exercise 35).

This gives us the complete algorithm for searching. We will make up one iteration by two reflections, one around $|U\rangle$ and one around $|\psi\rangle$. Since the angle between them is $\theta \sim \sqrt{1/N}$, we will rotate by angle 2θ. We would like to rotate by close to $\pi/2$ angle, and it will take $O(\sqrt{1/N})$ iterations (Figure 8).

We will see in the next section that rotation around $|\psi\rangle$ can be done without the oracle (not surprising, equal superposition does not depend upon the marked state). The exact implementation details are given in Section 3.

2.3 Product of two reflections

One of the central idea of Grover's algorithm is, product of two reflections is a rotation. Intuitively, you can verify that from Fig. 7. Let us prove it formally.

Say, there are two reflections, one about vector $|a\rangle$ and another about vector $|b\rangle$. Without loss of generality, assume that $|a\rangle = |0\rangle$ and $|b\rangle = \cos\theta|0\rangle + \sin\theta|1\rangle$. We will consider the plane spanned by $|a\rangle$ and $|b\rangle$.

The reflection around $|a\rangle$ is $2|0\rangle\langle 0| - I$, and hence equal to the matrix,

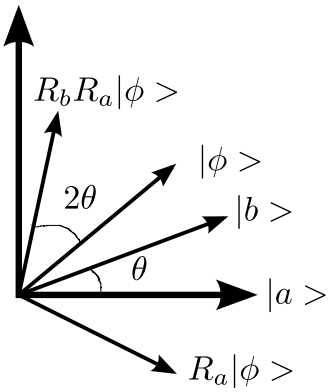
$$\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

Exercise 11. Show that the reflection around $|b\rangle$ is,

$$2|b\rangle\langle b| - I = \begin{pmatrix} 2\cos^2\theta - 1 & 2\cos\theta\sin\theta \\ 2\cos\theta\sin\theta & 2\sin^2\theta - 1 \end{pmatrix}$$

Then, the product of these two reflections is

$$R_{2\theta} := (2|b\rangle\langle b| - I)(2|a\rangle\langle a| - I) = \begin{pmatrix} \cos 2\theta & -\sin 2\theta \\ \sin 2\theta & \cos 2\theta \end{pmatrix}$$



R_a: Reflection around $|a>$

R_b: Reflection around $|b>$

Fig. 7. Search problem

Exercise 12. Find the eigenvalues and eigenvectors of this matrix.

This is the rotation matrix in the plane which rotates by angle 2θ. We conclude, the product of two reflections is a rotation by angle 2θ, where θ is the angle between the reflection axes.

3 Grover Search

As hinted above, the idea is to rotate the equal superposition vector multiple times and bring it as close to the marked state as possible. The rotation will be obtained by the product of two reflections. One reflection is around the equal superposition of unmarked states (performed by the oracle), and the other will be around the equal superposition $|\psi\rangle$.

One such rotation is known as a *Grover iteration*. By the previous section, it rotates any vector in the plane of $|U\rangle$ and $|M\rangle$ by an angle 2θ. We need to figure out two things.

- How to perform reflection around $|\psi\rangle$?
- How many times should we rotate?

The first question is easier to answer.

Exercise 13. How can we perform reflection around $|0\rangle$ state?

Since $|0\rangle$ is a constant state (in the sense that it does not depend on the input), we can easily recognize it and reflect around it. The reflection is equivalent (up to a global phase) to putting a phase of -1 if the state is $|0\rangle$, otherwise keep it unchanged.

Exercise 14. Can you think of a way to do it using controlled operations? It might be easier to think of the reflection around state $|11\cdots1\rangle$.

Notice that $|0\rangle = H^{\otimes k}|\psi\rangle$. So, the reflection around $|\psi\rangle$ will be performed by,

- Apply Hadamard transform to switch to $|0\rangle$ basis.
- Perform the reflection around $|0\rangle$.
- Apply Hadamard to come back to $|\psi\rangle$ basis.

The reflection around $|\psi\rangle$ will be written as,

$$H^{\otimes k}(2|0\rangle\langle 0| - I)H^{\otimes k} = 2|\psi\rangle\langle\psi| - I.$$

Combining the two reflections, the grover iteration is $G = H^{\otimes k}(2|0\rangle\langle 0| - I)H^{\otimes k}O_f$.

Now, we turn to the second problem. How many times should we rotate the initial vector $|\psi\rangle$? A Grover iteration G will rotate any vector by an angle 2θ in the plane of $|U\rangle$ and $|M\rangle$.

Exercise 15. What is θ here?

You can take the dot product of $|\psi\rangle$ and $|U\rangle$ to show that $\theta = \cos^{-1}\sqrt{\frac{n-1}{n}}$.

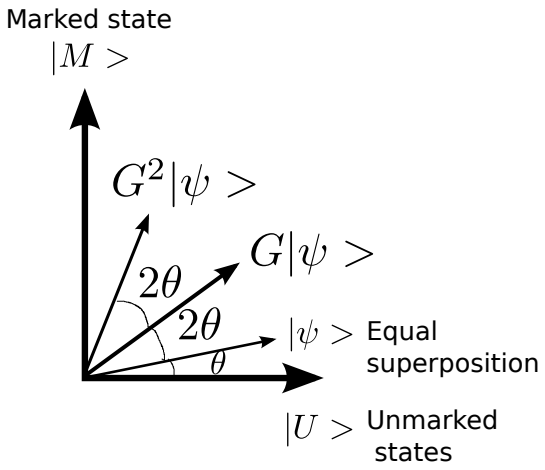


Fig. 8. Search problem

After l rotations the state $|\psi\rangle$ is at an angle $\theta + 2l\theta = (2l+1)\theta$ from the unmarked state $|U\rangle$. We want this angle to be as close to $\pi/2$ as possible. So, we need to apply Grover iteration,

$$l \approx \left(\frac{\pi}{2\theta} - 1\right)/2,$$

times.

Exercise 16. Why is the number of iterations approximately (and not exactly) equal to the expression above?

Since we can only apply Grover iteration integer number of times, l should be the nearest integer to $\left(\frac{\pi}{2\theta} - 1\right)/2$. To analyze the complexity of the algorithm, notice that $l = \Theta(\frac{1}{\theta})$. Since θ is small,

$$\frac{1}{\sqrt{n}} = \sin\theta \approx \theta.$$

Note 4. A simpler way is to ignore the initial angle (it is pretty small), then the number of iterations are approximately $\frac{\pi/2}{2\theta} = \frac{\pi}{4\theta}$.

Hence, the number of oracle calls required are $O(\sqrt{n})$ (one Grover iteration requires one call to oracle).

Exercise 17. We can do the reflection around $|\psi\rangle$ in poly-log(n) operations. Show that the time complexity of Grover's algorithm is also $\tilde{O}(\sqrt{n})$. Here, $\tilde{O}$ notation hides poly-log factors in the input size.

To conclude, Grover's algorithm is stated below.

- Apply Hadamard on $|0\rangle$ state to obtain the equal superposition over all indices, state $|\psi\rangle$.
- Apply Grover iteration $G = H^{\otimes k}(2|0\rangle\langle 0| - I)H^{\otimes k}O_f$ to the state $|\psi\rangle$ for l iterations, where $l = \lfloor\left(\frac{\pi}{2\cos^{-1}\sqrt{\frac{n-1}{n}}} - 1\right)/2\rceil$. ($\lfloor x\rceil$ is the closest integer to x.)
- Measure in the standard basis, outcome will be the marked state with high probability.

Let us see two very natural extensions of Grover's algorithm.

3.1 Amplitude amplification

An important generalization of Grover's algorithm is known as amplitude amplification. Let us first see what we *achieved* in Grover search. Given a searching space X (a set), say there is a marked/good subset X_1 and there is a unmarked/bad subset X_0. Given a marked element, we can recognize/verify it efficiently (an oracle is given for this task). We are interested in finding out an element in X_1.

If the set had no structure, this is the search problem and we get a quadratic speedup with Grover's algorithm. The classical algorithm is: we can find a market element with probability $\frac{|X_1|}{|X_0|+|X_1|}$ (pick it randomly), amplify this probability to constant by repeating this step.

Exercise 18. Suppose there are t marked elements in the set of n elements. How many times should we pick a random element to get constant probability of success?

Suppose there is some structure in these sets as opposed to unstructured search. Say, using this structure we are given an algorithm (probably classical) which finds a marked element with probability p. How can we increase the probability of success to a constant?

We have seen this multiple times before, a natural approach would be to apply the algorithm and check if the obtained element is marked or not. Repeating this procedure l times, the probability of success will be $1 - (1-p)^l$.

Exercise 19. Show that $l = \Theta(1/p)$ will make the above procedure succeed with constant probability?

The above exercise shows that we need to implement A around $1/p$ times. If

$$1/p \times (\text{time to run the procedure}) < \sqrt{n},$$

this algorithm will out-perform Grover's algorithm. Is there a way to utilize this new algorithm and get a quadratic speed up using quantum computing?

Exercise 20. Why is this problem useful?

What will be the quantum analog of the procedure we mentioned? The Hilbert space H is the space spanned by $|x\rangle$ for all $x \in X$. Say, the marked subspace is S_1, spanned by $|x\rangle$ for all $x \in X_1$ (similarly, we have S_0). We are given the ability to recognize marked elements. In other words, there is an oracle O_{X_1}, which puts the phase -1 on $|x\rangle$ if and only if the element is marked.

A quantum analog of the classical algorithm A (which succeeds with probability p), will move $|0\rangle$ to state $A|0\rangle$ which will have $\sqrt{p}$ overlap with the marked subspace. This ensures that the algorithm succeeds with probability p as specified.

Now we ask again, how many iteration of A are needed to boost the probability of success to a constant? In other words, can we amplify the amplitude (overlap with the marked subspace) to a constant? The crucial difference is: we are going to amplify amplitude instead of the probability.

It turns out that $O(\frac{1}{\sqrt{p}})$ iterations of A and its inverse A^{-1} will be enough to perform this *amplitude amplification*. This is quadratically faster than the classical approach. So, we get the same quadratic speedup even when there is a better than brute force algorithm for search.

Note 5. The inverse of algorithm A exists if it does not do any measurement.

You might have already guessed, Since this speedup is achieved by amplifying the amplitude of the state on the marked subspace, hence it is called *amplitude amplification*.

Steps for amplitude amplification: The strategy is very similar to Grover search; it can be seen as a generalization of Grover's algorithm. Suppose the state $A|0\rangle$ is,

$$|\psi\rangle := A|0\rangle = \cos\theta|b\rangle + \sin\theta|g\rangle.$$

Here $|g\rangle$ is the closest state to $|\psi\rangle$ in the marked/good subspace (similarly $|b\rangle$ in the unmarked/bad subspace).

Exercise 21. Show that we can always write $|\psi\rangle$ as such a state. What do we know about θ?

Remember that the state $A|0\rangle$ has overlap at least $\sqrt{p}$ with the marked subspace. Hence, θ is at least $sin^{-1}(\sqrt{p})$.

Note 6. In case of Grover search, we applied H instead of A to create equal superposition. Here we get to a state which has better overlap with the marked subspace in a single step. Another way to think about it is, A (and A^{-1}) allow us to rotate by a bigger θ.

We apply the same strategy as Grover, rotate the state $A|0\rangle$ to $|g\rangle$. Our algorithm will run in the plane spanned by $|b\rangle$ and $|g\rangle$. The new Grover iteration G will still be a product of two reflections: first rotation will be around $|\psi\rangle$, and the other one will be around $|b\rangle$.

The reflection around $|b\rangle$ is given by the oracle O_{X_1} (verify). The reflection around $|\psi\rangle$ is just $A(2|0\rangle\langle 0| - I)A^{-1}$.

Exercise 22. Show that $O(1/\sqrt{p})$ iteration of G will suffice to create a state with at least $2/3$ overlap with good subspace.

Note 7. Amplitude amplification assumes that A does not perform any measurements. Otherwise, we will not have A^{-1}.

Why is amplitude amplification useful? Grover search makes the brute force search algorithm faster. Amplitude amplification can make other kind of search algorithms quadratically better (under some restrictions).

We assumed that the number of marked elements was one in the previous section. What if the number of elements are more than 1? Amplitude amplification works when number of marked items are less than $\frac{n}{2}$.

If number of marked elements are m then your algorithm runs in $O(\sqrt{n/m})$ queries. Only the marked state $|M\rangle$ and the number of rotations change. The details are given as an assignment question.

Exercise 23. What if you have more than $n/2$ marked elements?

3.2 Number of marked items are not known

Let m be the number of marked elements in a search instance. Looking at Fig. 8 for Grover's algorithm, it is clear that the number of rotations should be approximately equal to

$$l = \lfloor \left(\frac{\pi}{2\cos^{-1}\sqrt{\frac{n-m}{n}}} - 1 \right) /2 \rceil.$$

In other words, rotating more (or less) than the required amount will take us away from the marked state. So, it seems that we need to know the number of marked items to execute Grover's algorithm.

It might seem a stretch that we already know the number of marked elements. In many search instances, there might not be a good estimate of number of marked elements. What could be done in such cases?

One way is to do a binary search on number of marked elements, i.e., we try values of $m = 1, 2, 4, 8, \cdots$. This will only incur an additional factor of $\log n$. It turns out, with help from our old friend, phase estimation, we can directly estimate the number of marked elements. Which matrix, and which eigenvector, should we use for phase estimation?

The answer is hidden in Grover iteration and its properties. Remember, we calculated the eigenvalues and eigenvectors of the rotation matrix,

$$R_{2\theta} = \begin{pmatrix} \cos 2\theta & -\sin 2\theta \\ \sin 2\theta & \cos 2\theta \end{pmatrix}$$

Grover iteration G is a matrix of this kind with $\cos\theta = \sqrt{\frac{n-m}{n}}$.

Exercise 24. Prove the above statement.

The matrix G has an eigenvector $\frac{1}{\sqrt{2}}|U\rangle + \frac{-i}{\sqrt{2}}|M\rangle$ with eigenvalue $e^{2i\theta}$. The other eigenvector is $\frac{1}{\sqrt{2}}|U\rangle + \frac{i}{\sqrt{2}}|M\rangle$ with eigenvalue $e^{-2i\theta}$.

Exercise 25. Can you find the number of marked elements now?

Since the eigenphase of Grover iteration is just a function of the number of marked solutions, estimating m is essentially a phase estimation on the Grover operator G.

Exercise 26. What is the relation between θ and number of marked elements?

The only other requirement for phase estimation is, we need an eigenvector of G to start the phase estimation algorithm.

Exercise 27. Show that $|\psi\rangle$ is a linear combination of the two eigenvectors mentioned above.

Hence, we can start with state $|\psi\rangle$ and apply phase estimation on G. We will either obtain 2θ or -2θ. In either case, it is easy to determine the number of marked elements.

How many oracle calls will this phase estimation take? We hope that it does not take more than $O(\sqrt{n})$ queries, the query complexity of Grover's algorithm without the estimation. From the discussion in last lecture about phase estimation, number of queries depend on the accuracy needed for the eigenphase 2θ.

The number of elements m should be estimated with an error at most $\sqrt{n}$ (which is equivalent to $k/2$ bits of accuracy for θ, $n = 2^k$). This is good enough for to figure out the number of rotations (for details, please refer to [1]).

Since we need an accuracy of $k/2$ bits on θ, that means, we need to apply phase estimation with $t = k/2 + f(\epsilon)$ qubits, where ϵ is the error probability of phase estimation. This will require only $O(\sqrt{n})$ calls to the oracle (we need controlled G^{2^t} operator).

Exercise 28. Construct a circuit for finding the number of marked elements.

This algorithm to estimate the number of marked elements is called *quantum counting*. Notice that the same idea will work to estimate $\sin\theta$ where

$$|\psi\rangle := A|0\rangle = \cos\theta|b\rangle + \sin\theta|g\rangle.$$

The corresponding algorithm is called *amplitude estimation*. Like in amplitude estimation, we will require A, A^{-1} and an oracle to give phase to "good" states.

4 Assignment

Exercise 29. Draw the circuit for the Grover iteration.

Exercise 30. Since we can only apply Grover iteration integer number of times, bound the maximum possible error.

Exercise 31. Run the Grover's algorithm for $m \leq \frac{n}{2}$ marked elements. What is the new marked state $|M\rangle$ and how many queries do we need to find a marked element?

Exercise 32. Find a quantum algorithm for search if we have more than $\frac{n}{2}$ marked elements using only constant number of queries?

Exercise 33. Why is amplitude amplification a generalization of Grover search? Carry out the details of amplitude amplification.

Exercise 34. Read section 6.2 from the book of Nielsen and Chuang [1].

Exercise 35. Show that the reflection around state $|\psi\rangle$ is same as reflecting amplitudes around the average in the histogram picture.

References

1. M. A. Nielsen and I. L. Chuang. *Quantum Computation and Quantum Information*. Cambridge, 2010.
2. Wikipedia. Elitzur-vaidman bomb tester, 2025. `https://en.wikipedia.org/wiki/Elitzur%E2%80%93Vaidman_bomb_tester`.