Lecture 11: Unstructured search

Rajat Mittal

IIT Kanpur

We move to the next famous quantum algorithm, Grover's search algorithm. As opposed to Shor's algorithm, it does not hope for 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.

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.

1 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 classical or quantum 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.

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

Given such an oracle O_f, search problems 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 will be shown later 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.

Classical approach: To solve a search problem, a classical algorithm needs to look through all the indices, and has to query oracle $\Theta(n)$ times. 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 2. How many queries do we need to make such that the probability of acceptance is constant (remember that there is only one marked element)?

There is a formal way to prove that any randomized algorithm will take at least $\Omega(n)$ queries, we won't cover it here. 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 3. Can it help in the brute force algorithm for factoring?

Idea of Grover search: What should be the quantum algorithm for searching an unstructured database? Remember the randomized algorithm, we were picking an element randomly. How could such a thing be done *quantumly*? Our states are vectors, each index (input to function f) will correspond to a basis state.

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 (bad 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 pick a random element from the list, like in other algorithms, we might want to create an equal superposition

$$|\psi\rangle = \frac{1}{\sqrt{n}} \sum_{x} |x\rangle,$$

apply the oracle and hope that we measure the marked element. We know that state $|\psi\rangle$ can be created using Hadamard transformation.

Exercise 4. What is the probability that we get state $|x^0\rangle$, if we measure $|\psi\rangle$ in the standard basis?

Clearly, the probability is very small, $\frac{1}{n}$. Instead, before measuring, we should get closer to state $|M\rangle$ (and away from $|U\rangle$). Notice that the states $|M\rangle$ and $|U\rangle$ are orthogonal to each other, and the plane spanned by them contains the equal superposition $|\psi\rangle$.

Exercise 5. Find the coefficients α, β, so that,

$$|\psi\rangle = \alpha|M\rangle + \beta|U\rangle.$$

Let us just worry about the plane spanned by $|U\rangle$ and $|M\rangle$, and see if we can move $|\psi\rangle$ closer to $|M\rangle$.

What does the phase oracle do? The oracle puts a phase of -1 on the marked state and leaves other states unchanged. Hence, the action of the oracle is $I - 2|x^0\rangle\langle x^0|$. This is equivalent to $2|U\rangle\langle U| - I$ in the plane spanned by $|M\rangle$ and $|U\rangle$.

The matrices of the form $2|\phi\rangle\langle\phi| - I$ are called the reflection matrices. That is because they reflect around the vector $|\phi\rangle$.

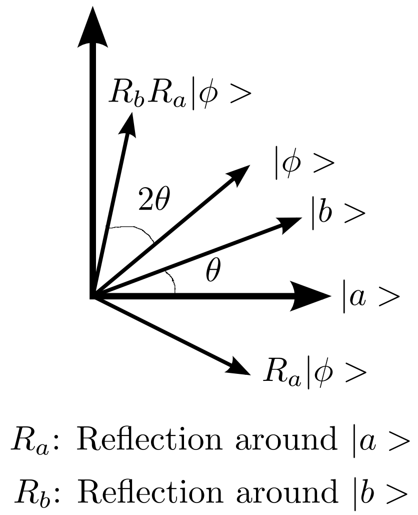


Fig. 1. Search problem

Exercise 6. Convince yourself that a reflection matrix performs a reflection in a plane.

So we have state $|\psi\rangle$ in the plane spanned by $|U\rangle$ and $|M\rangle$. We can reflect about the state $|U\rangle$ (the oracle), but we need to get closer to $|M\rangle$.

This will be achieved by using another reflection. Why will it work? We need to understand the properties of product of two reflections to answer that question.

Exercise 7. What should be the product of two reflections? Look at Fig. 1 for help.

1.1 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. 1. 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 8. 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

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

Exercise 9. 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.

Exercise 10. Can you guess the Grover's algorithm now?

2 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 11. 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 12. Can you think of a way to do it using controlled operations? It might be easier to think of reflection around state $|11\cdots 1\rangle$.

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 13. What is θ here?

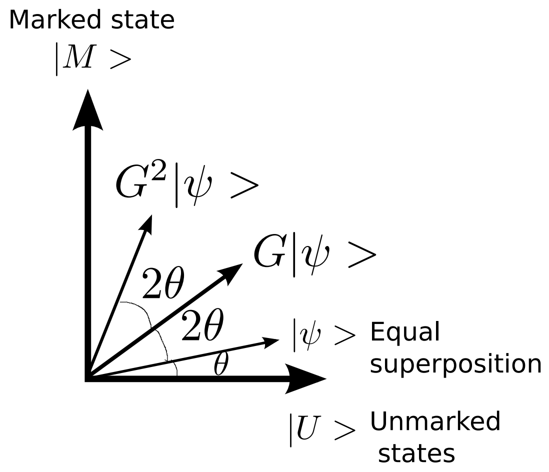
You can take the dot product of $|\psi\rangle$ and $|U\rangle$ to show that $\theta = \cos^{-1}\sqrt{\frac{n-1}{n}}$.

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



$$\cos\theta = \sqrt{\frac{n-1}{n}}$$

Fig. 2. Search problem

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,

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

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

Exercise 15. We can do the reflection around $|\psi\rangle$ in polylog(n) operations. Show that the time complexity of Grover's algorithm is also $\tilde{O}(\sqrt{n})$. Here, $\tilde{O}$ notation hides polylog 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.

2.1 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 16. 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. Using this structure, assume that 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 17. 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 18. 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 1. 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 19. 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 2. 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 20. 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 3. The 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 21. What if you have more than $n/2$ marked elements?

2.2 Number of marked items are not known

Let m be the number of marked elements in a search instance. Looking at Fig. 2 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? We will take help from our old friend, phase estimation. 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 22. 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 23. 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 24. 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 25. 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 $\sqrt{m}$. This is good enough for our purpose (to figure out the number of rotations). For details, please refer to [1].

Remember that $n = 2^k$, for $\sqrt{n}$ error, we need an accuracy of $k/2$ bits. 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 26. 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 marked states.

3 Assignment

Exercise 27. Draw the circuit for the Grover iteration.

Exercise 28. Since we can only apply Grover iteration integer number of times, bound the maximum possible error.

Exercise 29. 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 30. Find a quantum algorithm for search if we have more than $\frac{n}{2}$ marked elements using only constant number of queries?

Exercise 31. Why is amplitude amplification a generalization of Grover search? Carry out the details of amplitude amplification.

Exercise 32. Read section 6.2 from the book of Nielsen and Chuang [1].

References

1. M. A. Nielsen and I. L. Chuang. *Quantum Computation and Quantum Information.* Cambridge, 2010.