Unit 04: Grinding Gates in Quantum Computers

Tzu-Chieh Wei
C. N. Yang Institute for Theoretical Physics and Department of Physics and Astronomy, State University of New York at Stony Brook, Stony Brook, NY 11794-3840, USA

(Dated: September 4, 2024)

In this unit, we discuss the standard circuit-based quantum computation and introduce the Qiskit software developed by IBM.

Learning outcomes: (1) You'll be able to know elementary single-qubit and two-qubit quantum gates. (2) You'll be able to understand what quantum computation is and the specific quantum algorithm on searching. (3) You'll be able to get started in IBM Qiskit and run simple Jupyter notebooks of quantum circuits.

I. INTRODUCTION

We will now give an overview of quantum computation in terms of the standard circuit model. You have seen all the necessary ingredients before, and these are: (1) Initialization, (2) Gate operations and (3) Measurement/Readout. The schematic circuit is shown in Fig. 1.

In the initialization, we usually assume that all qubits are initialized to $|0\rangle$, unless stated otherwise. The specific layout of gate operations does not need to resemble that shown in the figure and it may depend on how an n-qubit unitary U is decomposed into a sequence of one-qubit and two-qubit gates. This second ingredient is the crux of quantum computation. To read out results, we need to perform measurement and it is usually assumed that each qubit is measured in the 0/1 or equivalently Z basis. If we need to measure in $+/-$ basis, we can apply a Hadamard gate H right before the 0/1 measurement.

Universal gate set. In classical computation, AND and NOT gates or just NAND gates are sufficient to build any Boolean functions. They are regarded as universal gate sets. In quantum computation, we also have the similar notion of universality. Given any unitary U acting on n-qubit, i.e., a unitary matrix of dimension $2^n \times 2^n$, it can be decomposed into a sequence of single-qubit and two-qubit gates. This is described in detail in Chapter 4 section 5 of Nielsen and Chuang. Essentially, the unitary U is decomposed into a sequence of the so-called two-level unitaries, each of which is a nontrivial action on a two-dimensional subspace, i.e., a multi-controlled-single-qubit unitary gates, i.e., the number of them is at most $2^{n-1}(2^n-1)$. Then as we shall see below in some examples of gate idenity, each multi-controlled gates can in turn be decomposed into single-qubit gates and CNOT gates. This means that the set of arbitrary single-qubit gates and the CNOT gate constitutes the corresponding universal set of gates in quantum computing. We note that the decomposition of U in the set is exact.

Exercise. What is the upper bound on the number of CNOT gates needed to implement an arbitrary U? Note that for a k-qubit controlled single-qubit gate (in total a $(k+1)$-qubit gate), one can use $2k-2$ Toffoli gates (see below) and single-qubit controlled single qubit gate (in total a 2-qubit gate).

You may have also heard people say that H, T and CNOT constitute a universal gate set, where we have seen H and CNOT, and T gate is

$$T = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{pmatrix},$$

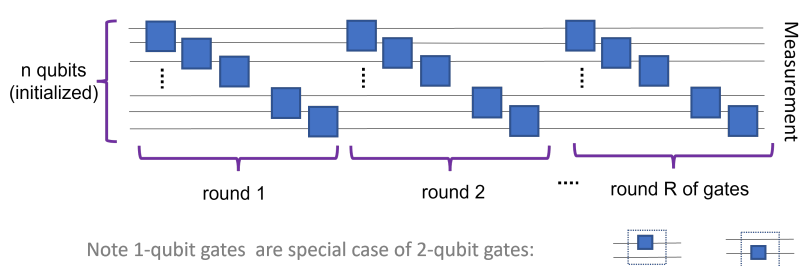
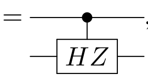


FIG. 1. Illustration of quantum computation in terms of the standard circuit model. We also note that one-qubit gates are a special case of two-qubit gates.

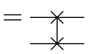
which can be regarded as a $\pi/4$ rotation about the z-axis: $T = e^{i\pi/8}e^{-i(\pi/4)(Z/2)}$. However, there are two fine-grained details. First, these three gates together *cannot* exactly constitute an arbitrary unitary U, but *can approximate* it as close as possible. This is the approximate universality. There are two other examples of such approximate universal gate sets.

- Example 2:

$$W = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{2}} \\ 0 & 0 & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{pmatrix} = \text{Controlled} - \text{HZ} =$$

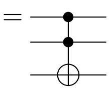


$$\text{SWAP} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} =$$



- Example 3:

$$H = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \quad \text{Toffoli} = \text{C}^2 - X \text{ (Control-Control-NOT)} =$$



The last two examples give other universal gates but achieve arbitrary gates in an approximate way. They also illustrate that universal quantum computation can be done with real-valued gates only. We note that the number i is essential in quantum mechanics [1, 2], though.

The exact universal set of gates includes all the one-qubit gates $u3(\theta, \phi, \lambda)$ and the CNOT gate (or equivalently C-Z gate for the latter); the specific form of u_3 is

$$u3(\theta,\phi,\lambda) = \begin{pmatrix} \cos(\theta/2) & -e^{i\lambda}\sin(\theta/2) \\ e^{i\phi}\sin(\theta/2) & e^{i\lambda+i\phi}\cos(\theta/2) \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\phi} \end{pmatrix}\begin{pmatrix} \cos(\theta/2) & -\sin(\theta/2) \\ \sin(\theta/2) & \cos(\theta/2) \end{pmatrix}\begin{pmatrix} 1 & 0 \\ 0 & e^{i\lambda} \end{pmatrix} = e^{i(\phi+\lambda)/2}e^{-i\phi Z/2}e^{-i\theta Y/2}e^{-i\lambda Z/2}, \tag{1}$$

which is a rotation using Euler angles: first around z by λ, then around y by θ, followed by rotation around z by ϕ.

Further comments on universality. It is recommended to read through Chapter 4.5 of Nielsen and Chuang on universal quantum gates. They illustrate that "two-level" unitary gates are universal, which basically shows that any $d \times d$ unitary matrix can be decomposed to a product at most $d(d-1)/2$ unitary gates, each of which is almost identity, except at a 2×2 block inside the matrix. This was done by a procedure similar to Gauss elimination, except that we have to use unitary matrices. Afterwards, one needs to show that any such two-level unitary gate can be decomposed as multi-qubit controlled NOT gates and multi-qubit controlled unitary gates (with the same 2×2 block). This establishes the exact universality of using CNOT gates and arbitrary one-qubit gates.

For a discrete set of universal gates, one needs to equip with the notion of approximation. For example, with H and T gates, one can generate a rotation by an angle that is an irrational multiple of 2π. This allows to approximate any single-qubit rotation gate as close as possible by using a longer sequence of H and T gates. The essential equation is

$$e^{-i\frac{\pi}{8}Z}e^{-i\frac{\pi}{8}X} = \cos^2\frac{\pi}{8}I - i\sin\frac{\pi}{8}\Big[\cos\frac{\pi}{8}(X+Z) + \sin\frac{\pi}{8}Y\Big],$$

i.e., a rotation about an axis $\vec{n} = (\cos\frac{\pi}{8}, \sin\frac{\pi}{8}, \cos\frac{\pi}{8})$ (un-normalized) by an angle θ with $\cos(\theta/2) = \cos^2\frac{\pi}{8}$. We thus need an efficient way to approximate a unitary using discrete gates. If it costs an exponential number of gates to reach the accuracy being ϵ, then the decomposition will not be useful. Naively, if we want to approximate an m gate circuit to accuracy ϵ, the number of gates from a fixed set required scales as $\Theta(m^2/\epsilon)$. Fortunately, there is a much better convergence according to the Solovay-Kitaev theorem, which uses $O\big(\log^c(1/\epsilon)\big)$ gates from the set to achieve the same accuracy ϵ (where $c \approx 2$ and later improved by Kuperberg to $c > 1.44042$). Recently there was an improvement claimed by H. F. Chau [3] that there is a way to use gates of the numer

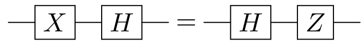
$$O\big(\log(1/\epsilon)\log\log(1/\epsilon)\log\log\log(1/\epsilon)\dots\big).$$

However, I have not verified this myself. Going through this can be assigned as a reading project.

In general, approximating arbitrary unitary gates is hard. This can be understood by a counting argument, by asking the alternative question that how many gates it takes to create an arbitrary n-qubit state. This can be done by using the ϵ-net picture; see Nielsen and Chuang 4.5.4.

II. GATE IDENTITIES AND MULTIQUBIT GATE DECOMPOSITION

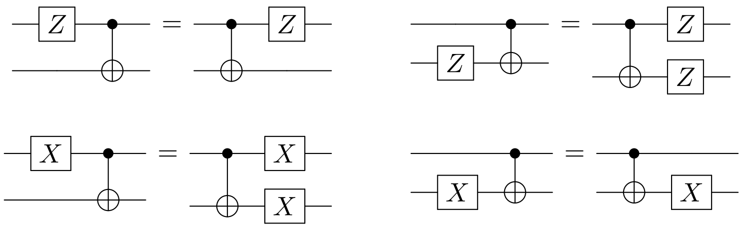
Gate identities. The first one we have seen earlier is the transformation between x and z bases via the Hadamard gate,



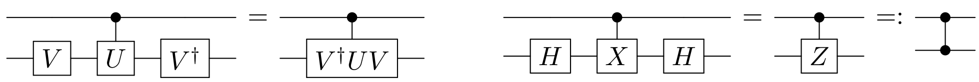
How do we transform between x and y bases? It is done via the phase gate,

$$S \equiv \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix}, \quad SXS^\dagger = Y.$$

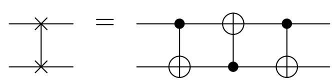
Given CNOT's importance as a two-qubit gate, we examine how it commutes with X and Z gates.



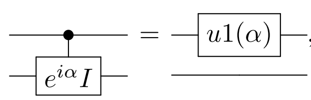
We can also conguate controlled gates by a single-qubit unitary at the target,



To build a swap gate between two qubits requires three CNOTs,

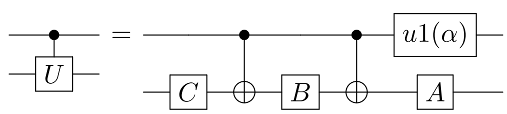


A controlled phase gate is similar to the phase kickback,



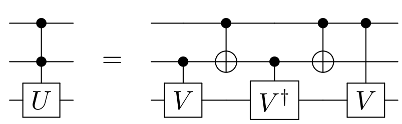
$$u1(\alpha) \equiv \begin{pmatrix} 1 & 0 \\ 0 & e^{i\alpha} \end{pmatrix}$$

Multiqubit gates from the standard set. There are a few useful identities for us to construct multiqubit gates. But first we look at the two-qubit gate,



(2)

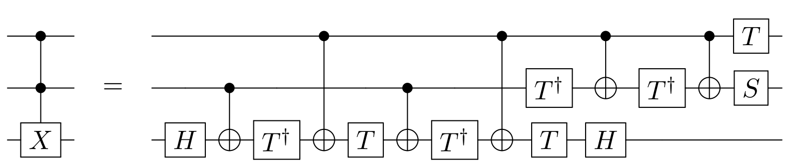
This is based on the single-qubit U being decomposed to $U = e^{i\alpha}AXBXC$, with $ABC = I$. The next is a three-qubit controlled gate,



where $V^2 = U$. For Toffoli gate: U=X, so can choose the following V and then the controlled-V,

$$V = \frac{1-i}{2}(I + iX) \Rightarrow V^2 = X.$$

This requires 8 CNOT gates. But there is a decomposition with 6 CNOT gates



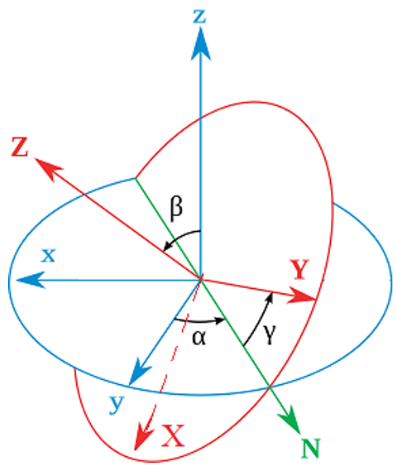


FIG. 2. Illustration of Euler rotations: how two sets of coordinate frames can be rotated to each other. The two x-y and X-Y planes intersect along the N axis. It is thus clear that the two axes z and Z are orthogonal to N. To align the two frames, one rotate y axis around z axis (by an angle α) so that y axis aligns with the N axis. Then one rotate z axis about N to align with the Z axis by an angle β. Finally, one rotates the y axis (now with N) around the Z axis by an angle γ to align with the Y axis. The x and X axes will then be aligned automatically.

This is left as an exercise. We note that there is a proof that there must be at least 5 two-qubit gates for the Toffoli gate [4].

In the book by Nielsen and Chuang, there is some discussion on n-qubit controlled unitary, but we will not cover it here.

Euler rotation. According to the steps in Fig. 2, the overall rotation is described by

$$U(\alpha,\beta,\gamma) = R_Z(\gamma)R_N(\beta)R_z(\alpha).$$

We will need the two identities

$$R_Z(\gamma) = R_N(\beta)R_z(\gamma)R_N(\beta)^{-1}, \quad R_N(\beta) = R_z(\alpha)R_y(\beta)R_z(\alpha)^{-1},$$

which simply come from the fact that a rotation by one axis can be done by first rotating the vector to a second axis, then rotating around this axis by the desired angle, and finally rotating from the second axis back to the firist. We then rewrite the above U with mixed axes to one with the fixed axes,

$$U(\alpha,\beta,\gamma) = R_z(\alpha)R_y(\beta)R_z(\gamma),$$

whose proof is left as an exercise.

We also include an arbitrary overall phase in the unitary group and for one qubit we have

$$e^{\imath\delta}U(\alpha,\beta,\gamma) = e^{i\delta}R_z(\alpha)R_y(\beta)R_z(\gamma).$$

The IBM's $u3$ gate (see above Eq. 1) is related to this via $\delta = \frac{\alpha+\gamma}{2},\ \beta=\theta,\ \alpha=\phi,\ \gamma=\lambda.$

Relating back to the decomposition of control-U in Eq. (2), for the above Euler decomposition, one can verify that $A = R_z(\alpha)R_y(\beta/2)$, $B = R_y(-\beta/2)R_z\big(-(\gamma+\alpha)/2\big)$ and $C = R_z\big((\gamma-\alpha)/2\big)$, which is left as an exercise (noting that $R_{y,z}(\theta)X = XR_{y,z}(-\theta)$).

III. IBM QISKIT: AN INTRODUCTION

A. Register for an account

Please go to `https://quantum-computing.ibm.com/` to register an account. After loginning to your account, you will see (1) IBM Quantum Composer and (2) IBM Quantum Lab among other things, as illustrated in Fig. 3.

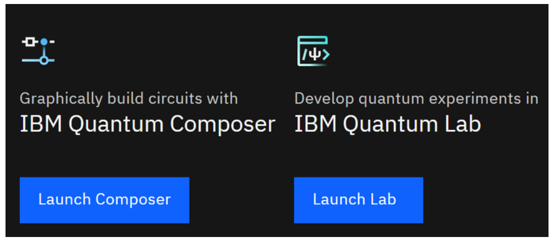


FIG. 3. Illustration of part of the view in a Qiskit account. You can access IBM Quantum at `https://quantum-computing.ibm.com/`. If you are new to IBM Quantum, you can create an IBMid to access it. Two key components to write and run codes are: (1) IBM Quantum Composer and (2) IBM Quantum Lab. [Update: IBM cancelled their Quantum Lab service on May 2024. In order to run Qiskit codes, you will need to install Qiskit on your machine or use other online platforms, such as Google's Colab.]

B. Qiskit gate set

Here, we list a gates that are available on IBM Qiskit [5]. (We note that some conventions have changed, e.g. gates u3 and u2, will be lumped into a single u gate and u1 gate is renamed as 'p' gate. This also affect their controlled versions.)
u gates.

- u3(angle1,angle2,angle3):

$$u3(\theta,\phi,\lambda) = \begin{pmatrix} \cos(\theta/2) & -e^{i\lambda}\sin(\theta/2) \\ e^{i\phi}\sin(\theta/2) & e^{i\lambda+i\phi}\cos(\theta/2) \end{pmatrix}$$

- u2(angle1,angle2):

$$u2(\phi,\lambda) = u3(\pi/2,\phi,\lambda) = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & -e^{i\lambda} \\ e^{i\phi} & e^{i(\phi+\lambda)} \end{pmatrix}$$

- u1(angle1):

$$u1(\lambda) = u3(0,0,\lambda) = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\lambda} \end{pmatrix}$$

Identity gate: iden = u0(angle),

$$u0(\delta) = u3(0,0,0)$$

Hadamard gate: h.
Phase gate and its inverse: s and sdg.
T gate and its inverse: t and tdg.
Rotations about X,Y,Z axes: rx(angle,qubit), ry(angle,qubit) and rz(angle,qubit).

There are multiple qubit gates.
Controlled-NOT gate: cx(control,target).
Controlled-Y and -Z gates: cy(control,target) and cz(control,target).
Controlled-Hadamard gate: ch(control,target).
Controlled-Rotation gates: crx(angle,control,target), cry(angle,control,target) and crz(angle,control,target).
Controlled-U1 gate: cu1(angle,control,target).
Controlled-U3 gate: cu3(angle1,angle2,angle3,control,target).
Swap gate: swap(qubit1,qubit2).
Toffoli gate: ccx(control1,control2,target).
Controlled swap gate (Fredkin gate): cswap(control,qubit2,qubit3).

The following are not unitary operations. **Measurement**: measure(qubit, classical bit).

Reset qubit to 0: reset(qubit).

Conditional operation (on classical outcome):

```
qc.measure(q, c)
qc.x(q[0]).c_if(c, 0)#apply X to q[0] if c is 0
```

C. Measurement

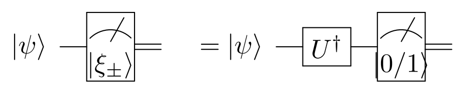
In IBM, Google or Rigetti, the measurement is done in 0/1 basis, e.g.

```
>>> qc.measure(q, c)
```

In order to measure in arbitrary basis defined by $|\xi(0)\rangle$ and $|\xi(1)\rangle$:

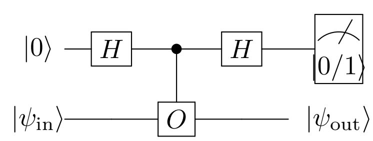
$$|\xi_+\rangle = U_\xi|0\rangle,\ |\xi_-\rangle = U_\xi|1\rangle,$$

we first apply inverse of U (i.e. $U^\dagger$) before measuring in 0/1 basis,



Note if one cares about the exact post-measurement state being in the $\xi\rangle$ basis, one should apply U after the measurement to undo the basis change $U^\dagger$ done earlier.

How to measure a single-qubit operator O (unitary and Hermitian, thus $O^2 = I$) and leave the output in the eigenstate? It can be done with the following circuit (a kind of Hadamard test),



(3)

Principle of deferred measurement. In principle, measurement can be moved to the end of the circuit; if measurement results are used to classically control some operation, it can be replaced by a corresponding controlled operation.

IV. CLIFFORD GATES AND GOTTESMAN-KNILL NO-GO THEOREM

Clifford gates UC are those that transform a Pauli product σ to another Pauli product operator σ' (up to ± 1):

$$U_C\,\sigma\,U_C^\dagger = \sigma'.$$

For one qubit, we have seen example gates that do this: $\{H, S\}$. Of course, Pauli gates X, Y, Z are also other examples, albeit trivial, as they commute or any commute with each other. In fact, the Pauli gates can be generated by $\{H, S\}$, e.g. $Z = S^2$, and $X = HS^2H$. It turns one-qubit Clifford group is generated by H and S, i.e. $\mathcal{C}_1 = \langle\langle H, S\rangle\rangle$. Moreover, for multiple qubit Clifford group, we need additionally CNOT gates.

Given that the transformation by the Clifford group can be identified by how Pauli products are transformed, quantum computation using only these gates (and simple measurements) can be efficiently simulated.

Theorem 10.7 [of Nielsen and Chuang]: (Gottesman–Knill theorem) Suppose a quantum computation is performed which involves only the following elements: state preparations in the computational basis, Hadamard gates, phase gates, controlled-NOT gates, Pauli gates, and measurements of observables in the Pauli group (which includes measurement in the computational basis as a special case), together with the possibility of classical control conditioned on the outcome of such measurements. Such a computation may be efficiently simulated on a classical computer.

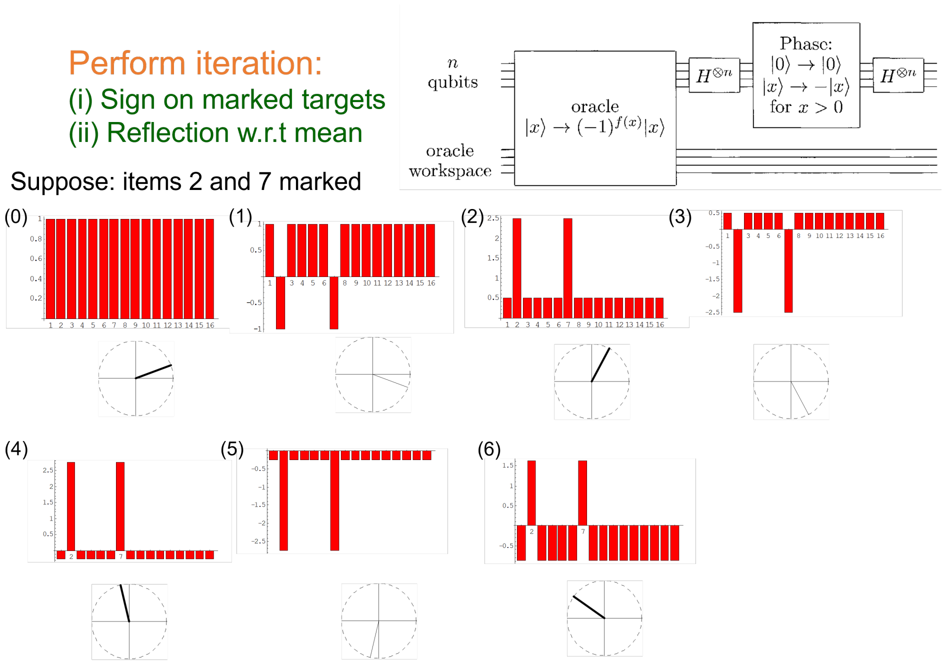


FIG. 4. Illustration of Grover's algorithm on 16 objects with 2 marked items.

V. GROVER'S SEARCH ALGORITHM

Let us refer to Fig. 4 for an illustration of Grover's search algorithm [6–8]. The algorithm invovles these following steps ,

(i) Flip the sign on marked targets (equivalent to reflection w.r.t. the unmarked "plane"):

$$\hat{O}_f = \sum_x (-1)^{f(x)} |x\rangle\langle x| = I - 2 \sum_{x\in\text{marked}} |x\rangle\langle x|.$$

(ii) Relfection w.r.t. to the mean:

$$U_s = 2|s\rangle\langle s| - I = H^{\otimes n}(2|0\ldots 0\rangle\langle 0\ldots 0| - I)H^{\otimes n},$$

where

$$|s\rangle = |++\cdots+\rangle = \frac{1}{\sqrt{N = 2^n}} \sum_{x=0}^{2^n-1} |x\rangle.$$

Under this,

$$|\alpha\rangle \equiv \sum_k \alpha_k |k\rangle \longrightarrow 2|s\rangle\langle s|\alpha\rangle - |\alpha\rangle,$$

in terms of the components,

$$\alpha_k \longrightarrow 2\frac{1}{N}\sum_j \alpha_j - \alpha_k = 2\langle\alpha\rangle - \alpha_k.$$

These two steps are reflections. One Grover iteration is a combination of the two and is thus a unitary operation that is equivalent to a rotation:

$$\hat{G} \equiv U_s \hat{O}_f,$$

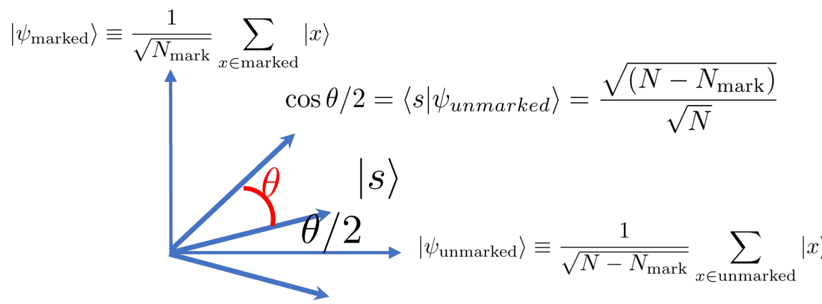


FIG. 5. Illustration of the geometric picture of the Grover's search algorithm.

with the angle θ satisfying

$$\sin\theta = 2\frac{\sqrt{N_{\text{mark}}(N-N_{\text{mark}})}}{N}.$$

This is illustrated in Fig. 5. The goal is to reach $\theta = \pi/2$ as close as possible, and the mumber of iterations to reach the angle $\pi/2$ is

$$N_{\text{iter}}\,\theta + \frac{\theta}{2} \approx \frac{\pi}{2}, \quad N_{\text{iter}} \approx \frac{\pi}{2\theta} - \frac{1}{2} \approx \left\lceil \frac{1}{4}\sqrt{\frac{N}{N_{\text{mark}}}} \right\rceil.$$

When there is just one marked item $N_{\text{mark}} = 1$,

$$N_{\text{iter}} \approx \left\lceil \frac{\sqrt{N}}{4} \right\rceil,$$

meaning that there is a speedup compared to classical search that needs to examine on average $N/2$ items.

Note that for $N = 4$ with only one marked item: $\theta = \pi/3$, one iteration reaches the target with probability 1.

When the number of marked items is unknown. How do we know when to stop? This requires the knowledge of the angle θ, which can be estimated using the quantum phase estimation algorithm, which we will discuss in a later lecture. Essentially, the Grover operator $\hat{G}$ has two eigenvalues $e^{\pm i\theta}$. If one can imprint the phase to a quantum register then we can read it out using the so-called quantum Fourier transform, which is part of the quantum phase estimation algorithm.

VI. GENERALIZATION: AMPLITUDE AMPLIFICATION

Amplitude amplication [9] generalizes the Grover's algorithm and we do not need to create an equal superposition of all objects. Let us recall the case for Grover.

$$\hat{G} \equiv U_s\,\hat{O}_f = H^{\otimes n}\,U_{|0\rangle^\perp}\,H^{\otimes n}\,\hat{O}_f,$$

where

$$\hat{O}_f = \sum_x (-1)^{f(x)}|x\rangle\langle x|,\ U_s = H^{\otimes n}(2|0\dots0\rangle\langle 0\dots 0| - I)H^{\otimes n} = H^{\otimes n}U_{|0\rangle^\perp}H^{\otimes n}.$$

The algorithm iteratively applies $\hat{G}$ to an initial state

$$|\psi_{\text{ini}}\rangle = H^{\otimes n}|0\dots0\rangle.$$

With such a picture, we can generalize the above form

$$\hat{G}_A = A\,U_{|0\rangle^\perp}\,A^{-1}\,\hat{O}_f = U_{|\psi\rangle^\perp}\,\hat{O}_f.$$

The initial state is generated by A,

$$|\psi\rangle = A|0\ldots0\rangle = \sin(\theta/2)|\psi_{\rm good}\rangle + \cos(\theta/2)|\psi_{\rm bad}\rangle,$$

where $|\psi_{\rm good}\rangle$ is a superposition of all marked items and $|\psi_{\rm bad}\rangle$ contains no marked item in the superposition. Note that $|00...0\rangle$ can contain extra work qubits. The action of $\hat{G}_A$ is a rotation of θ in the 2D space spanned by $\{|\psi_{\rm good}\rangle, |\psi_{\rm bad}\rangle\}$, or equivalently, spanned by

$$\{|\psi\rangle, |\bar{\psi}\rangle \equiv \cos(\theta/2)|\psi_{\rm good}\rangle - \sin(\theta/2)|\psi_{\rm bad}\rangle\}.$$

Then the action of $\hat{G}_A$ on $|\psi\rangle$ leads to

$$(\hat{G}_A)^k|\psi\rangle = \sin(k\,\theta + \theta/2)|\psi_{\rm good}\rangle + \cos(k\theta + \theta/2)|\psi_{\rm bad}\rangle.$$

The goal is to get as close to $k\theta + \theta/2 = \pi/2$ for some integer k.

Note that similar to Grover's search, the operator $\hat{G}_A$ also has eigenvalues $e^{\pm i\theta}$, which, in principle, can be calculated from the so-called quantum phase estimation (to be covered in a later lecture).

Hadamard test and generalization. This part combines ideas from Hadamard test, amplitude amplification and quantum phase estimation; see, e.g., Section II of Ref. [10] for a summary. We have seen amplitdue amplification, but we are not yet ready to discuss quantum phase estimation yet. Here, we will just illustrate the simplest scenario of the Hadamard test, shown in Eq. (3). Before the second Hadamard, the whole system is in

$$(|0\rangle|\psi_{\rm in}\rangle + |1\rangle O|\psi_{\rm in}\rangle)/\sqrt{2}. \tag{4}$$

Essentially, the Hadamard followed by measurement in 0/1 basis amounts to measuring in +/- basis. Denote the outcome by $(-1)^s$, where s is 0 or 1 (corresponding to the final Z measurement outcome), the second qubit becomes

$$|\psi_{\rm out}(s)\rangle = \frac{1}{2}(I + (-1)^s O)|\psi_{\rm in}\rangle, \tag{5}$$

which is a projection onto the +1 or -1 eigenstate of O (assuming $O^2 = I$). This is a useful technique to project a state into eigenstates of an observable that squares to identity.

We will discuss quantum phase estimation after we learn quantum Fourier transform in a later unit.

VII. CONCLUDING REMARKS

In this unit, we have discussed the standard circuit-based quantum computation and introduced the Qiskit software developed by IBM.

It is a good time to check whether you have achieved the following Learning Outcomes:

After this Unit, (1) You'll be able to know elementary single-qubit and two-qubit quantum gates. (2) You'll be able to understand what quantum computation is and the specific quantum algorithm on searching. (3) You'll be able to get started in IBM Qiskit and run simple Jupyter notebooks of quantum circuits.

Suggested reading: N&C chap 4; KLM chap 4; Qb 1.4; 2.4, 2.5. A. Barenco et al., Elementary gates for quantum computation [11]. We note that N&C chap 4.5 has useful discussions on efficiency in approximating arbitrary single-qubit gates (from a discrete set of gates) and mentions the Solovay-Kitaev theorem that uses $O(\log^c(1/\epsilon))$ gates to an accuracy ϵ with $c \approx 2$, which is faster than a naive estimate $\Theta(1/\epsilon)$; see also their Appendix 3. It also discusses a very important issue on efficiency of approximating arbitrary unitary gates, which is generically hard. This conclusion can be obtained by 'counting' the ϵ-net on a $(2^{n+1}-1)$ sphere in terms of $(2^{n+1}-2)$-sphere of radius ϵ, compared with the number of different states that m-gate crcuits with g different types of gates (each on at most f qubits) can generate.

---

[1] J. L. Miller, Does quantum mechanics need imaginary numbers?, Physics Today **75**, 14 (2022).
[2] M.-O. Renou, D. Trillo, M. Weilenmann, T. P. Le, A. Tavakoli, N. Gisin, A. Acín, and M. Navascués, Quantum theory based on real numbers can be experimentally falsified, Nature **600**, 625 (2021).

[3] H. Chau, Efficient fault-tolerant single qubit gate approximation and universal quantum computation without using the solovay-kitaev theorem, arXiv preprint arXiv:2406.04846 (2024).
[4] N. Yu, R. Duan, and M. Ying, Five two-qubit gates are necessary for implementing the toffoli gate, Physical Review A **88**, 010304 (2013).
[5] M. S. ANIS, Abby-Mitchell, H. Abraham, *et al.*, Qiskit: An open-source framework for quantum computing (2021).
[6] L. K. Grover, A fast quantum mechanical algorithm for database search, in *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing* (1996) pp. 212–219.
[7] L. K. Grover, Quantum mechanics helps in searching for a needle in a haystack, Phys. Rev. Lett. **79**, 325 (1997).
[8] L. K. Grover, Quantum computers can search rapidly by using almost any transformation, Phys. Rev. Lett. **80**, 4329 (1998).
[9] G. Brassard, P. Hoyer, M. Mosca, and A. Tapp, Quantum amplitude amplification and estimation, Contemporary Mathematics **305**, 53 (2002).
[10] J. Zhao, Y.-H. Zhang, C.-P. Shao, Y.-C. Wu, G.-C. Guo, and G.-P. Guo, Building quantum neural networks based on a swap test, Phys. Rev. A **100**, 012334 (2019).
[11] A. Barenco, C. H. Bennett, R. Cleve, D. P. DiVincenzo, N. Margolus, P. Shor, T. Sleator, J. A. Smolin, and H. Weinfurter, Elementary gates for quantum computation, Phys. Rev. A **52**, 3457 (1995).