Universal Quantum Gates

report for the course
3621510 Introduction to Quantum Computing, 5 Cp

written by
Juha-Matti Huusko
juha-matti.huusko@uef.fi
175783
University of Eastern Finland

submitted on
December 10, 2021

Abstract

A set of universal quantum gates is any set of gates which can express other unitary operations of the quantum computer.

We recall quantum gates and see that the so-called Clifford gates are not sufficient. However, a slightly larger set of gates is sufficient.

We see that any unitary operation can be expressed quite accurately quite fast with the mentioned universal gates.

We discuss how much accuracy is needed.

Division of work

All work was done by the only group member.

The report is quite short, since the group member has been sick for 2 weeks. Antibiotics lasted 2.12.-8.12.2021.

Contents

1 Definition of a set of Universal Quantum Gates

Universal Quantum Gates are defined in Wikipedia as:

"A set of universal quantum gates is **any set of gates** to which any operation possible on a quantum computer can be reduced, that is, any other unitary operation can be expressed as a finite sequence of gates from the set." [6]

Quantum gates can be represented with $2x2$ unitary matrices. The most common quantum gates are

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

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad Y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}, \quad Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix},$$

$$T = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{bmatrix}, \quad S = \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix} = T^2,$$

$$R_x(\theta) = \begin{bmatrix} \cos(\theta/2) & i\sin(\theta/2) \\ -i\sin(\theta/2) & \cos(\theta/2) \end{bmatrix}$$

The gates X, Y, Z can be represented by the name CNOT.

However, there is an infinite number of interesting quantum gates, for example,

$$R_x(\pi/2), R_2(\pi/3), R_3(\pi/4), \ldots$$

which can be needed in calculations. For example, a circuit for Quantum Fourier Transform is composed of H gates and the controlled version of

$$R = \begin{bmatrix} 1 & 0 \\ 0 & e^{2\pi i/2^m}. \end{bmatrix}$$

Here m can be any number. Depending on the situation, a different R is needed. [7]

It is not sustainable to manufacture a new gate whenever one is needed, because this would happen infinitely often.

How can we find a set of universal quantum gates which suffices for building quantum computers? Which gates are enough?

2 The set $\{CNOT, H, S\}$ is not universal

There is a somewhat indirect proof that the set $\{CNOT, H, S\}$ is not a set of universal quantum gates. The proof uses the Gottesman-Knill theorem.

Theorem 2.1 (Gottesman–Knill 1998)**.** *A quantum computer using*

- *Preparation of qubits in computational basis states,*
- *gates $\{CNOT, H, S\}$ (so-called Clifford gates)*
- *Measurements in the computational basis.*

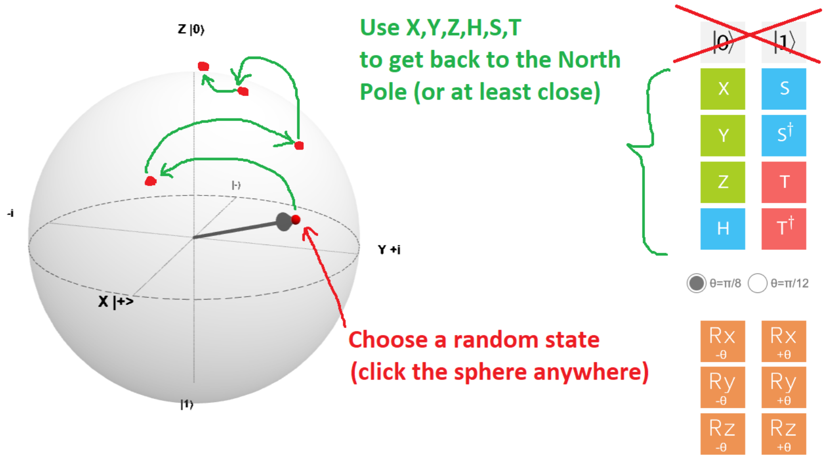
can be simulated efficiently on a classical computer.[5]

Now, not all quantum circuits can be simulated efficiently on a classical computer. (This was mentioned during the course.) Therefore, not all quantum circuits can be expressed by gates $\{CNOT, H, S\}$!

3 The set $\{CNOT, H, S, T\}$ is universal

The set $\{CNOT, H, S\}$ is not a set of universal quantum gates. However, with a minor addition T the set becomes a set of universal quantum gates. (The addition is very small, since $T^2 = S$.)

So, the set $\{CNOT, H, S, T\}$ is a set of universal quantum gates. Let's not look for a proof. Playing with the Bloch sphere for 2 minutes can convince us about the fact.



Visit the web page [2]. Click on a random place on the Bloch sphere. By using the buttons X, Y, Z, H, S, T, try to return to the North Pole. After playing for 2 minutes, it seems that this can be always done.

4 The set $CNOT, H, S, T\}$ is universal and can represent unitary operations quite fast

If we have a gate G, how many gates from $\{CNOT, H, S, T\}$ are needed to approximate it?

There is a theoretical result, which states that not very many gates are needed. The result contains an algorithm to find the approximation. In Wikipedia, the result is stated quite vaguely:

Theorem 4.1 (Solovay-Kitaev)**.** *If U is a set of universal gates, then any gate G can be approximated by a "fairly short" sequence of gates.[4]*

In [3], numerical bounds have been given as follows:

"The algorithm runs in $O(\log^{2.71}(1/\varepsilon))$ time, and produces as output a sequence of $O(\log^{3.97}(1/\varepsilon))$ quantum gates which is guaranteed to approximate the desired quantum gate to an accuracy within ε."

Let's consider a numerical example and apply these numerical bounds.

Example 4.1. Let G be some strange quantum gate. Let P be a product of universal gates. Let $G - P$ have absolute values of its elements less than 0.001. How many factors P usually has? 10?, 100?, 1000?

Now $\varepsilon = 0.001$ and $1/\varepsilon = 1000$. Hence

$$C \log^{3.97}(1000) \approx C(\log(1000))^4 = C3^4 = 81C$$

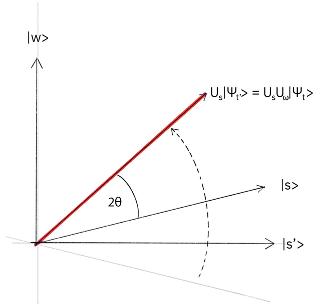
gates are needed.

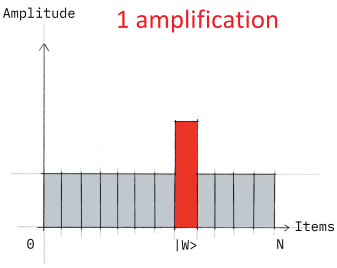
For $\varepsilon = 0.01$, only $C(\log 100)^4 = 2^4 = 16C$ gates are needed.

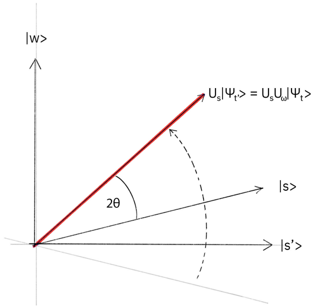
(Here C is the constant in the "big-Oh" $O(\log^{3.97}(1/\varepsilon))$.)

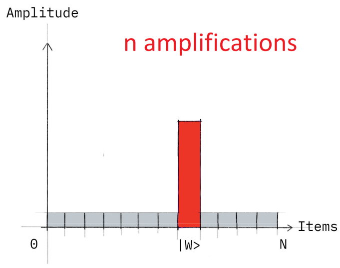
5 Perhaps no need to be so accurate

Grover's algorithm uses the the amplitude amplification trick to make the correct option to "stand up" and be visible.

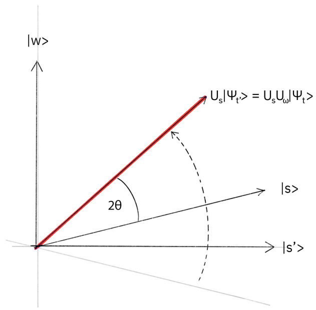


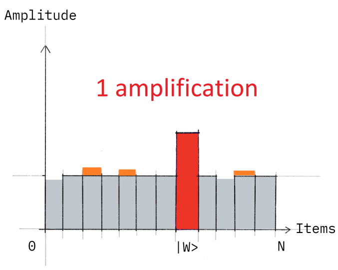


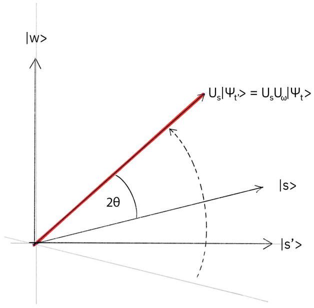




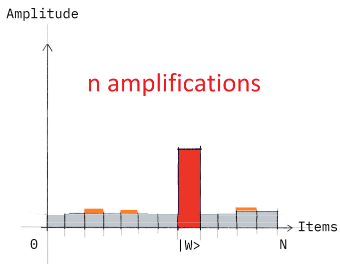
It is not clear what would happen, if there would be errors. Perhaps the errors would be negligible or they would accumulate. In case of negligible errors, we could obtain the following results for Grover's algorithm.







|w>
$U_s|\Psi_t\rangle = U_sU_\omega|\Psi_t\rangle$
|s>
2θ
|s'>



Amplitude
n amplifications
Items
0
|W>
N

6 A simple random search

It would have been good to try to implement the algorithm from citearxiv.

However, for this work, a simple random approach was tried with MATLAB. The needed codes are in the appendices.

The idea is to

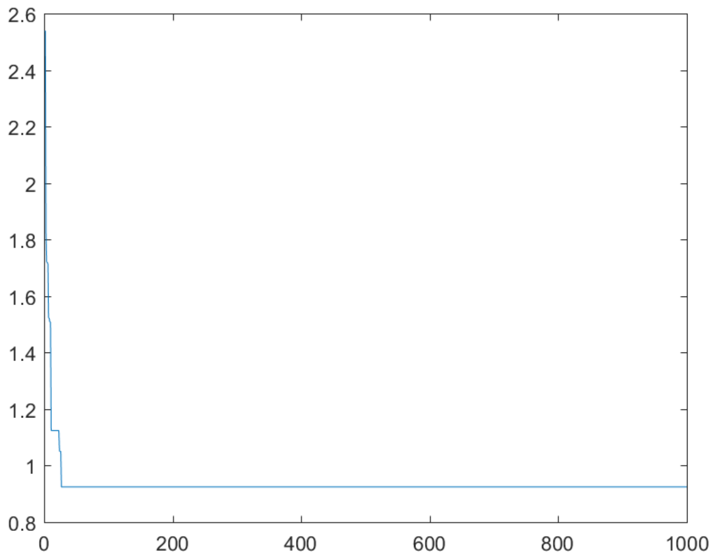
1. create a random unitary matrix

2. combine n random operations from the universal gate set

3. compare the new approximation to the previous one: choose the better one

4. repeatedly, choose a random n and redo steps 2 and 3

The convergence was not so clear as the following figure shows.



The figure is related to a situation, where

```
U =

  −0.6104 − 0.5097i   0.4653 + 0.3886i
  −0.4653 − 0.3886i  −0.6104 − 0.5097i


P =
```

```
−0.1768 − 0.1768i   0.1768 + 0.1768i
−0.1768 − 0.1768i  −0.1768 − 0.1768i
```

The corresponding sequence of gates is

$$33415415654334132353353364335466.$$

Here the gates are represented by numbers as $H-1$, $X-2$, $Y-3$, $Z-4$, $T-5$, $S-6$.

The code should be improved, for example, to remove "33". Two consequtive Y gates cancel each other.

7 Conclusions

The main finding was that the Bloch sphere [2] is a strong tool. It can convince us about facts which have been found true during the past 25 years – that it is possible to approximate any unitary operation with a finite set of quantum gates.

The approximation was seeked by using a simple random approach. The approach gave some insight but was not satisfactory. Perhaps it would have been possible to try to implement the algorithm from [3].

References

[1] https://quantum-computing.ibm.com/composer/docs/iqx/guide/grovers-algorithm (visited 10.12.2021)

[2] https://sami.andberg.net/bloch/bloch.html (visited 10.12.2021)

[3] https://arxiv.org/abs/quant-ph/0505030 (visited 10.12.2021)

[4] https://en.wikipedia.org/wiki/Solovay%E2%80%93Kitaev_theorem (visited 10.12.2021)

[5] https://en.wikipedia.org/wiki/Gottesman%E2%80%93Knill_theorem (visited 10.12.2021)

[6] https://en.wikipedia.org/wiki/Quantum_logic_gate#Universal_quantum_gates (visited 10.12.2021)

[7] https://en.wikipedia.org/wiki/Quantum_Fourier_transform (visited 10.12.2021)

8 Appendices

The following are MATLAB codes. The functions "randomuni" and "uni" should be saved to the MATLAB working directory. Then the loop code should be used to generate a random unitary matrix via "randomuni" and to try to represent it via the universal gates by function "uni".

8.1 Code: Random unitary matrix

```
function [Us] = randomuni()
%Produces a random unitary matrix

%Let's make a real rotation matrix
t=2*pi*rand(1);
U=[cos(t),-sin(t);sin(t),cos(t)];

%Let's multiply by a unimodular complex number
s=2*pi*rand(1);
Us=exp(1i*s)*U;
end
```

8.2 Code: Representation with universal gates

```
function [P,seq] = uni(U,A,sequence,n)
%Represents a unitary operation by using 2x2 matrices H,X,Y,Z,T,S

%A set of universal quantum gates is defined by
H=0.5*[1,1;1,-1];
X=[0,1;1,0];
Y=1i*[0,-1;1,0];
Z=[1,0;0,-1];
T=[1,0;0,exp(1i*pi/4)];
S=T^2;

gates{1}=H;
gates{2}=X;
gates{3}=Y;
gates{4}=Z;
gates{5}=T;
gates{6}=S;

vnew=1+round(5*rand(n,1));
Pnew=eye(2);
for k=1:n
    Pnew=Pnew*gates{vnew(k)};
end

normOld=norm(U-A,'fro');
normNew=norm(U-A*Pnew,'fro');

if normOld<=normNew
    P=A;
    seq=sequence;
else
    P=A*Pnew;
    seq=[sequence;vnew];
end

end
```

8.3 Code: Loop for seeking a good representation

```
N=1000;
U=randomuni();
P=eye(2);
seq=[];
for k=1:N
[P,seq] = uni(U,P,seq,1+round(5*rand(1)));
x(k)=norm(P-U,'fro');
end
plot([1:N],x)
```