Name: **Huan Q. Bui**
Course: **8.370 - QC**
Problem set: **#6**
Due: Wednesday, Nov 2, 2022
Collaborators/References: discussions with Prof. Shor, Shira Asa-El, and Christina Yu

1. Pauli group, Clifford group

(a) The Clifford group is

$$\mathcal{C} \equiv \{\text{unitaries } U : UgU^\dagger \in \mathcal{P}\, \forall g \in \mathcal{P}\}$$

where $\mathcal{P}$ is the Pauli group. Suppose $A, B \in \mathcal{C}$. Then for any $g \in \mathcal{P}$:

$$ABg(AB)^\dagger = A\underbrace{(BgB^\dagger)}_{\in\mathcal{P}}A^\dagger = Ag'A^\dagger \in \mathcal{P}.$$

So, $AB \in \mathcal{C}$. Now let $g \in \mathcal{P}$ be given and $A \in \mathcal{C}$. The map $g \to AgA^\dagger$ is a bijection between $\mathcal{P}$ and itself is bijective because it is injective and $\mathcal{P}$ is a finite group. To prove injectivity is easy: if $Ag_1A^\dagger = Ag_2A^\dagger$ then $g_1 = g_2$ by left-multiplying and right-multiplying by $A^\dagger$ and A respectively. So, we have that $A^\dagger gA = A^\dagger Ag'A^\dagger A = g' \in \mathcal{P}$ for some $g' \in \mathcal{P}$. So, every element of $\mathcal{C}$ has an inverse. The identity element is simply the identity matrix. Associativity is inherited from associativity of matrix multiplication. So, $\mathcal{C}$ is a group.

(b) The Pauli group for 1 qubit is given by

$$\mathcal{P}_1 = \{\pm I, \pm iI \pm X, \pm iX, \pm Y, \pm iY, \pm Z, \pm iZ\}.$$

The Pauli group for n qubits is simply generated, via the tensor product, by the operators in $\mathcal{G}_1$, which are generated by the Pauli matrices X, Y, Z. H is unitary. So, it remains to check that H is in $\mathcal{C}_1$. Using the fact that $H^\dagger = H^{-1} = H$, we can simply check that $HXH, HYH, HZH \in \mathcal{P}_2$:

$$\begin{aligned} HXH &= Z \in \mathcal{P}_1 \\ HZH &= X \in \mathcal{P}_1 \\ HYH &= -Y \in \mathcal{P}_1 \end{aligned}$$

And we're done!

(c) To check that the $CNOT$ gate is in $\mathcal{C}$, we check that it is in $\mathcal{C}_2$. $CNOT$ is unitary, so we check that $CNOT\, g\, CNOT^\dagger$ is in $\mathcal{P}_2$ for all generators g of $\mathcal{P}_2$. The generators of $\mathcal{P}_2$ are once again X, Y, Z tensored with the identity matrix either on the first or second qubit. This means there are six cases:

$$\begin{aligned} CNOT\, X \otimes \mathcal{I}\, CNOT^\dagger &= X \otimes X \in \mathcal{P}_2 \\ CNOT\, Y \otimes \mathcal{I}\, CNOT^\dagger &= Y \otimes X \in \mathcal{P}_2 \\ CNOT\, Z \otimes \mathcal{I}\, CNOT^\dagger &= Z \otimes \mathcal{I} \in \mathcal{P}_2 \\ CNOT\, \mathcal{I} \otimes X\, CNOT^\dagger &= \mathcal{I} \otimes X \in \mathcal{P}_2 \\ CNOT\, \mathcal{I} \otimes Y\, CNOT^\dagger &= Z \otimes Y \in \mathcal{P}_2 \\ CNOT\, \mathcal{I} \otimes Z\, CNOT^\dagger &= Z \otimes Z \in \mathcal{P}_2 \qquad \checkmark \end{aligned}$$

Here, the $CNOT$ gate in consideration is one where the first qubit is the control. However, since the other CNOT gate only differs on on which qubits it uses as control and target, we only need to check one of the two CNOTs.

(d) We want to check that $T \notin \mathcal{C}_1$. Consider $g = \mathrm{X} \in \mathcal{P}_1$.

$$TXT^\dagger = \begin{pmatrix} 0 & e^{-i\pi 4} \\ e^{i\pi/4} & 0 \end{pmatrix} \notin \mathcal{P}_1.$$

So T is not in the Clifford group.

2. Gotta erase workbits in Simon's algorithm! (see edited problem)

It's not completely obvious to me that the oracle, in practice, requires extra qubits to implement. But okay. Following the information given in the problem, we have that after applying the Hadamard transform, followed by the circuit, followed by the CNOT's (which copies $f(x)$ into the fourth register), followed by the inverse function, the quantum register takes the form

$$\sum_{x=0}^{2^L-1} |x\rangle\, |0\rangle\, |0\rangle\, |f(x)\rangle\,.$$

If we applying the Hadamard transform now we will get Simon's algorithm. However, suppose now that we do not apply the reverse function. Then the state of the system before the second Hadamard transform is

$$\sum_{x=0}^{2^L-1} |x\rangle\, |f(x)\rangle\, \left|\text{junk}_x\right\rangle |f(x)\rangle$$

After the Hadamard transform (on the first register, of course), this becomes

$$\sum_{k=0}^{2^L-1} \sum_{x=0}^{2^L-1} (-1)^{x\cdot k}\, |k\rangle\, |f(x)\rangle\, \left|\text{junk}_x\right\rangle |f(x)\rangle$$

Here I have dropped all normalization since it is not relevant to this problem. Now, we measure the first register, so we look for the amplitudes of the terms $|k\rangle\, |f(x)\rangle \left|\text{junk}_x\right\rangle |f(x)\rangle$. Notice that unlike in the usual Simon's algorithm where the amplitude of the term $|k\rangle\, |f(j)\rangle$ has two contributions due to j and $j \oplus c$, here the contributions to the *probability* of seeing $|k\rangle$ is in general the sum of the squares of the amplitudes of $|k\rangle\, |f(x)\rangle \left|\text{junk}_x\right\rangle |f(x)\rangle$. We no longer get the right kind of interference: since the junk terms can be different for the same $|k\rangle$, we may not add the amplitudes before squaring. This means that we can no longer deduce properties of k in relation to c based on our observations of k after the measurement.

3. Simon's algorithm

(a) Recall Simon's algorithm. After the second application of the Hadamard transform to the first register, the state of the system is

$$\frac{1}{2^n} \sum_{k=0}^{2^n-1} \left(\sum_{j=0}^{2^n-1} (-1)^{j\cdot k} \right) |k\rangle\, |f(j)\rangle\,.$$

Now we want to compute the probability of seeing a pair of values $|k\rangle\, |f(j)\rangle$. This is equal to the square of the amplitude on this state. How many values of j produce $f(j)$? Suppose there are $2L$ different j's for which $f(j)$ gives the same output. Then they must be $\{j_1, j_2, \ldots, j_L, j_1 \oplus c, j_2 \oplus c, \ldots, j_L \oplus c\}$ since $\oplus$ is symmetric. This means that the amplitude for $|k\rangle\, |f(j)\rangle$ has the form

$$\frac{1}{2^n} \sum_{i=1}^{L} \left((-1)^{j_1\cdot k} + (-1)^{(j_1 \oplus c)\cdot k} \right) = \frac{1}{2^n} (1 + (-1)^{c\cdot k}) \sum_{i=1}^{L} (-1)^{j_i\cdot k}$$

This amplitude is nonzero only if $c \cdot k = 0 \mod 2$, so **YES** the quantum part of Simon's algorithm still always return a binary string with $c \cdot k = 0 \mod 2$.

(b) The probability of measuring a state with $f = 1$ is

$$p = \Pr(f = 1) = \frac{1}{2^{2n}} \sum_{k=0}^{2^n-1} \left|(1 + (-1)^{c\cdot k})(-1)^{d\cdot k}\right|^2 = \frac{1}{2^{2n}} \sum_{k=0}^{2^n-1} \left|(1 + (-1)^{c\cdot k})\right|^2 = \frac{1}{2^{n-1}} \quad \forall c \neq 0.$$

The last equality is due to the fact that $c \cdot k = 0$ for half the values of k between 0 and $2^n - 1$, and so the numerator is $2^2 \times (2^n/2) = 2 \times 2^n$.

Now, let the number of failures before the first success be $k - 1$. The probability for getting $k - 1$ failures and a success on the kth try is:

$$\Pr(X = k) = (1 - p)^{k-1} p.$$

The expected value for the random variable X, the number of measurements before seeing $f = 1$, is

$$E[X] = \sum_{k=0}^{\infty} k \Pr(X = k) = \frac{1}{p}.$$

So the answer is that we are expected to measure $\boxed{2^{n-1}}$ times.

The amplitude of the pair $|k\rangle |f(j)\rangle$ in the registers is

$$\frac{1}{2^n}(1 + (-1)^{c\cdot k}) \sum_{i=1}^{L} (-1)^{j_i \cdot k}$$

Here, $L = 1$ for $j = d$ and $L = 2^{n-1} - 1$ otherwise. This amplitude is nonzero only if $c \cdot k = 0 \mod 2$. The probability associated with finding a k that is perpendicular to c is thus:

$$\frac{1}{2^{2n-2}} \left| \sum_{i=1}^{2^{n-1}-1} (-1)^{j_i \cdot k} \right|^2 + \frac{1}{2^{2n-2}}$$

Notice that the sum inside the absolute square has an odd number of terms, so it is nonzero. As a result, if we find any $|k\rangle |f(j)\rangle$ after our measurement, then that k string has the property $k \cdot c = 0$. The probability for this is also on the order of $1/2^{2n-2}$. Therefore, following the same argument as that given in the lecture notes, the number of times we have to run the algorithm is $O(n)$. I think the answer is $O(n)$ as opposed to $O(n^2)$ as written the lecture notes. The oracle calls f only once in the circuit, independent of the register size, and we run the circuit $O(n)$ times.

On a classical computer, we will essentially have to evaluate f until we get **two** outputs that have value 1. Suppose we decided to evaluate all inputs in some order. The worst thing that could happen is that $f = 1$ only for the last two outputs. Even if we find $f = 1$ on the very first evaluation, we may still have to evaluate to the last input to find $f = 1$ again. So, to guarantee success we will have to evaluate $O(2^n)$ times.

4. Partial transpose

(a) Suppose M is separable, i.e.,

$$M = \sum_i \lambda_i |v_i\rangle \langle v_i| \otimes |w_i\rangle \langle w_i|$$

where λ_i's are positive. Then the partial transpose of M according to the definition in the problem is

$$pt(M) = \sum_i \lambda_i (|v_i\rangle \langle v_i|)^\top \otimes |w_i\rangle \langle w_i| .$$

Since $\Pi_i = |v_i\rangle \langle v_i|$ are orthogonal projections, the matrices $(|v_i\rangle \langle v_i|)^\top$ are also orthogonal projections. We may very well consider the transposition as a unitary change of basis in the first qubit and write

$$pt(M) = \sum_i \lambda_i \left|v_i'\right\rangle \left\langle v_i'\right| \otimes |w_i\rangle \langle w_i| .$$

It is clear that the spectrum of $pt(M)$ is exactly the same as that of M in this case, so $pt(M)$ must also be positive. However, we could also be explicit: Let $x = \sum_{i,j} c_{ij} \left|v_i'\right\rangle \left|w_j\right\rangle$. Then

$$\langle x|\ pt(M)\ |x\rangle = \sum_i \lambda_i |c_{ii}|^2 \geq 0.$$

And we're done.

(b) The density matrix for $|\psi\rangle = (1/\sqrt{2})(|00\rangle + |11\rangle)$ is

$$\rho = |\psi\rangle \langle\psi| = \frac{1}{2}\begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 \end{pmatrix}$$

From here we find

$$pt(\rho) = \frac{1}{2}\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

The eigenvalues of this matrix are $-1/2, 1/2, 1/2, 1/2$, so $pt(\rho)$ is not non-negative which implies that ρ is not separable in view of Part (a).

5. Teleporting a qutrit directly

Instead of embedding the qutrit in a set of qubits of higher dimensions, we can teleport qutrits directly. Let $\omega = e^{2\pi i/3}$, the cube root of unity. Define

$$P = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \omega & 0 \\ 0 & 0 & \omega^2 \end{pmatrix} \qquad \text{and} \qquad T = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}.$$

The analogue of the EPR pair is the state

$$|EPR_3\rangle = \frac{1}{\sqrt{3}}(|00\rangle + |11\rangle + |22\rangle)$$

and the analogue of the Pauli matrices are the nine matries $P^a T^b$, with $0 \leq a, b < 3$. We teleport the qutrit as follows:

Alice has some qutrit in state $|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle + \gamma\,|2\rangle$. She now measures her two qutrits (the qutrit in state $|\psi\rangle$ and her half of the EPR_3 pair) in the EPR-pair basis which consists of nine states in the form

$$(\mathcal{I} \otimes P^a T^b)(|00\rangle + |11\rangle + |22\rangle).$$

Alice will then send her measurement result (one of the nine possible ones) to Bob, and Bob will apply one of nine unitaries to his qutrit to obtain ψ. In particular, if Alice sees a state associated with P^aT^b, then Bob applies P^aT^b to his qutrit. Below we will show explicitly.

Here we show the nine-element basis for Alice's measurement, the resulting state on Bob's qutrit, and Bob's corrective unitary for each case:

$$\begin{aligned}
&(1/\sqrt{3})(\mathcal{I}\otimes P^0T^0)(|00\rangle+|11\rangle+|22\rangle)\longrightarrow|B\rangle=(1/\sqrt{3})\mathcal{I}(\alpha|0\rangle+\beta|1\rangle+\gamma|2\rangle)\longrightarrow P^0T^0\\
&(1/\sqrt{3})(\mathcal{I}\otimes P^1T^0)(|00\rangle+|11\rangle+|22\rangle)\longrightarrow|B\rangle=(1/\sqrt{3})P^2(\alpha|0\rangle+\beta|1\rangle+\gamma|2\rangle)\longrightarrow P^1T^0\\
&(1/\sqrt{3})(\mathcal{I}\otimes P^2T^0)(|00\rangle+|11\rangle+|22\rangle)\longrightarrow|B\rangle=(1/\sqrt{3})P^1(\alpha|0\rangle+\beta|1\rangle+\gamma|2\rangle)\longrightarrow P^2T^0\\
&(1/\sqrt{3})(\mathcal{I}\otimes P^0T^1)(|00\rangle+|11\rangle+|22\rangle)\longrightarrow|B\rangle=(1/\sqrt{3})T^2(\alpha|0\rangle+\beta|1\rangle+\gamma|2\rangle)\longrightarrow P^0T^1\\
&(1/\sqrt{3})(\mathcal{I}\otimes P^1T^1)(|00\rangle+|11\rangle+|22\rangle)\longrightarrow|B\rangle=(1/\sqrt{3})P^2T^2(\alpha|0\rangle+\beta|1\rangle+\gamma|2\rangle)\longrightarrow P^1T^1\\
&(1/\sqrt{3})(\mathcal{I}\otimes P^2T^1)(|00\rangle+|11\rangle+|22\rangle)\longrightarrow|B\rangle=(1/\sqrt{3})PT^2(\alpha|0\rangle+\beta|1\rangle+\gamma|2\rangle)\longrightarrow P^2T^1\\
&(1/\sqrt{3})(\mathcal{I}\otimes P^0T^2)(|00\rangle+|11\rangle+|22\rangle)\longrightarrow|B\rangle=(1/\sqrt{3})T(\alpha|0\rangle+\beta|1\rangle+\gamma|2\rangle)\longrightarrow P^0T^2\\
&(1/\sqrt{3})(\mathcal{I}\otimes P^1T^2)(|00\rangle+|11\rangle+|22\rangle)\longrightarrow|B\rangle=(1/\sqrt{3})P^2T(\alpha|0\rangle+\beta|1\rangle+\gamma|2\rangle)\longrightarrow P^1T^2\\
&(1/\sqrt{3})(\mathcal{I}\otimes P^2T^2)(|00\rangle+|11\rangle+|22\rangle)\longrightarrow|B\rangle=(1/\sqrt{3})PT(\alpha|0\rangle+\beta|1\rangle+\gamma|2\rangle)\longrightarrow P^2T^2.
\end{aligned}$$

All the calculations are done using Mathematica. I have also checked that the basis Alice uses is indeed an orthonormal basis. To do this, I formed a matrix U from the EPR states and verify that $U^\dagger U=\mathcal{I}$.

Mathematica code:

```
In[64]:= (*qutrit*)

In[49]:= EPR = 1/Sqrt[3]*{1, 0, 0, 0, 1, 0, 0, 0, 1};

In[15]:= Id3 = IdentityMatrix[3];

In[16]:= w = Exp[2*Pi*I/3];

In[17]:= P = {{1, 0, 0}, {0, w, 0}, {0, 0, w^2}};

In[18]:= T = {{0, 1, 0}, {0, 0, 1}, {1, 0, 0}};

(*EPR basis*)

In[50]:= EPR1 = EPR;

In[51]:= EPR2 = KroneckerProduct[Id3, P] . EPR;

In[52]:= EPR3 = KroneckerProduct[Id3, P . P] . EPR;

In[53]:= EPR4 = KroneckerProduct[Id3, T] . EPR;

In[54]:= EPR5 = KroneckerProduct[Id3, P . T] . EPR;

In[55]:= EPR6 = KroneckerProduct[Id3, P . P . T] . EPR;

In[56]:= EPR7 = KroneckerProduct[Id3, T . T] . EPR;

In[57]:= EPR8 = KroneckerProduct[Id3, P . T . T] . EPR;

In[58]:= EPR9 = KroneckerProduct[Id3, P . P . T . T] . EPR;

In[28]:= (*check ONB*)

In[62]:= (*form a matrix U using the EPRs and compute U.ConjugateTranpose[U]*)

In[63]:= U = {EPR1, EPR2, EPR3, EPR4, EPR5, EPR6, EPR7, EPR8, EPR9};

In[67]:= U . ConjugateTranspose[U] // FullSimplify

Out[67]= {{1, 0, 0, 0, 0, 0, 0, 0, 0},
          {0, 1, 0, 0, 0, 0, 0, 0, 0},
          {0, 0, 1, 0, 0, 0, 0, 0, 0},
           {0, 0, 0, 1, 0, 0, 0, 0, 0},
           {0, 0, 0, 0, 1, 0, 0, 0, 0},
           {0, 0, 0, 0, 0, 1, 0, 0, 0},
           {0, 0, 0, 0, 0, 0, 1, 0, 0},
           {0, 0, 0, 0, 0, 0, 0, 1, 0},
           {0, 0, 0, 0, 0, 0, 0, 0, 1}}

In[80]:= (*full state with 3 qutrits*)
State = Flatten[KroneckerProduct[{a, b, c}, EPR]];

In[81]:= (*Alice measures. We keep only the first 3 entries since only those are relevant*)

In[82]:= Take[P1 . State, 3]

Out[82]= {a/(3 Sqrt[3]), b/(3 Sqrt[3]), c/(3 Sqrt[3])}

In[83]:= Take[P2 . State, 3]

Out[83]= {a/(3 Sqrt[3]), (b E^(-((2 I \[Pi])/3)))/(3 Sqrt[3]), (c E^((2 I \[Pi])/3))/(3 Sqrt[3])}

In[84]:= Take[P3 . State, 3]

Out[84]= {a/(3 Sqrt[3]), (b E^((2 I \[Pi])/3))/(3 Sqrt[3]), (c E^(-((2 I \[Pi])/3)))/(3 Sqrt[3])}

In[85]:= Take[P4 . State, 3]

Out[85]= {b/(3 Sqrt[3]), c/(3 Sqrt[3]), a/(3 Sqrt[3])}

In[86]:= Take[P5 . State, 3]

Out[86]= {b/(3 Sqrt[3]), (c E^(-((2 I \[Pi])/3)))/(3 Sqrt[3]), (a E^((2 I \[Pi])/3))/(3 Sqrt[3])}

In[87]:= Take[P6 . State, 3]

Out[87]= {b/(3 Sqrt[3]), (c E^((2 I \[Pi])/3))/(3 Sqrt[3]), (a E^(-((2 I \[Pi])/3)))/(3 Sqrt[3])}

In[88]:= Take[P7 . State, 3]

Out[88]= {c/(3 Sqrt[3]), a/(3 Sqrt[3]), b/(3 Sqrt[3])}

In[89]:= Take[P8 . State, 3]

Out[89]= {c/(3 Sqrt[3]), (a E^(-((2 I \[Pi])/3)))/(3 Sqrt[3]), (b E^((2 I \[Pi])/3))/(3 Sqrt[3])}

In[90]:= Take[P9 . State, 3]

Out[90]= {c/(3 Sqrt[3]), (a E^((2 I \[Pi])/3))/(3 Sqrt[3]), (b E^(-((2 I \[Pi])/3)))/(3 Sqrt[3])}
```