CIS 492/593 Lab 6 Simon's Algorithm Fall 2023

In this hands-on lab, you will study and practice:

- the Simon's algorithm
- the Oracle construction method for the Simon's algorithm

Login to a workstation and open a terminal (either middle click the terminal icon on the left side bar or press CTRL-ALT-T). In the terminal window, type

```
cd QC
jupyter notebook
```

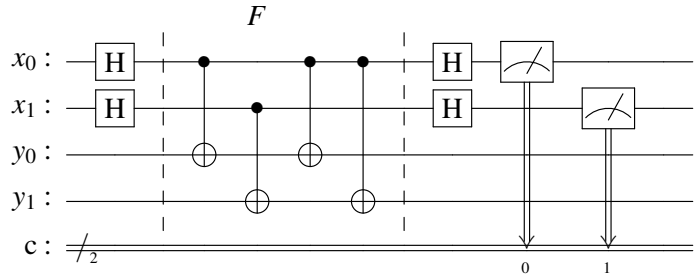
Inside the browser, click "New" and choose "Python 3 (ipykernel)". For each experiment, you need to put the experiment number as a comment (e.g. `# Experiment 1`) in the top of the code.

Experiment 1: Simon's Algorithm with a 2-bit secret string

- In the empty cell, type
  ```
  %load ~cis492s/pub/Simon.py
  ```
  and click "Run" to load a program which will give us a 2-bit string whose dot product with the secret string s is 0.
- Study the code. Note that the `Oracle()` function builds and returns the circuit for the oracle function F. We also use the method `compose()` in Qiskit to combine two quantum circuits together. That is, we add the oracle circuit `Fc` to the circuit `qc`.
- Click the cell containing the code and then click "Run" to execute it. Record the output string of the top 2-qubit x.
- Repeat above step to query the oracle until you get two different strings. How many times have you queried the oracle?
- Ignore the output string '00' which wouldn't give us any information. Use the other output string to calculate its dot product with the secret string s (i.e. $s_0 s_1$) and the result should be 0. Hence, you can get one equation regarding s_0 and/or s_1. Based on the equation, figure out the bit pattern of the secret string s.

Experiment 2: Simon's Algorithm with another 2-bit secret string

- Click the cell which contains the code done in Experiment 1. Click the "Edit" button and select "Copy Cells". Click "Edit" again and choose "Paste Cells Below".
- Modify the code in the `Oracle()` function to construct the circuit of F (i.e. between two barrier lines) , like below:

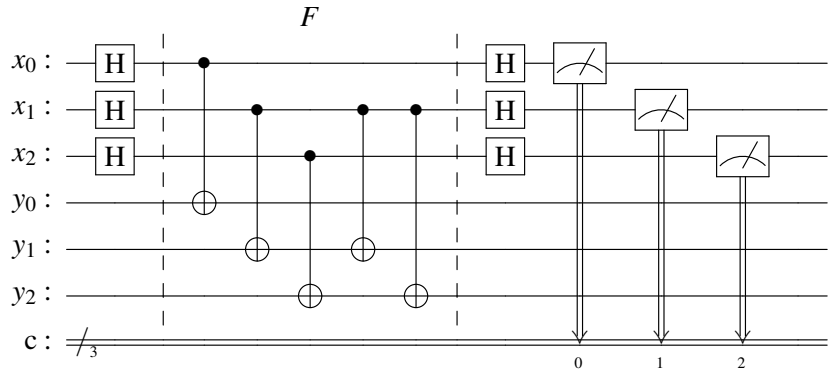


- Click "Run" to execute the code. Record the output string of the top 2-qubit x.
- Repeat above step to query the oracle until you get two different strings. How many times have you queried the oracle?

- Use the output string which is not '00. We know that this string's dot product with the secret string s (i.e. $s_0 s_1$ should be 0. Hence, you can obtain one equation related to s_0 and/or s_1. Based on the equation, derive the bit pattern of the secret string s. Note that the secret string cannot be '00'.

Experiment 3: Simon's Algorithm with a 3-bit secret string

- Click the cell which contains the code done in Experiment 1. Click the "Edit" button and select "Copy Cells". Click "Edit" again and choose "Paste Cells Below".
- Change the line of the code 'n = 2' to be 'n = 3'.
- Modify the code in the `Oracle()` function to construct the circuit of F (i.e. between two barrier lines) , like below:



- Click "Run" to execute the code. Record the output string of the top 3-qubit x.
- Repeat above step to query the oracle until you get three different strings. How many times have you queried the oracle?
- Use the output strings except the string '000'. We know that each of these strings has a dot product of 0 with the secret string s (i.e. $s_0 s_1 s_2$). Hence, you can obtain two (or three) equations related to s_0, s_1, and/or s_2. Based on the equations, figure out the bit pattern of the secret string s. Note that the secret string cannot be '000'.

Experiment 4: The Oracle Construction method for Simon's Algorithm

In this experiment, you will learn how to construct the Oracle circuit for a given string s. There are two steps in the construction method:

Step I: Apply the CNOT gates from qubits of the first register (i.e. x) to qubits of the second register (i.e. y). So the content of each qubit in the first register, which is the classical information encoded as either a $|0\rangle$ or a $|0\rangle$, will be copied to the corresponding qubit in the second register. In fact, this can be simply implemented by using one line of code:

```
fc.cx(x,y)
```

instead of using a loop :

```
for i in range n:
    fc.cx(x[i],y[i])
```

Step II: Because the bits in the string s cannot be all 0, find the least index k such that $s_k = 1$. Next, apply the CNOT gates from the qubit x_k to any qubit y_i if $s_i = 1$.
For example, if $s = 110$, the least index k is 0 because $s_0 = 1$. Therefore, we apply two CNOT gates from x_0 to y_0 and from x_0 to y_1.

In fact, the Oracle circuits in Experiment 2 and Experiment 3 are also based on the two-step procedure.

- Click the cell which contains the code done in Experiment 3. Click the "Edit" button and select "Copy Cells". Click "Edit" again and choose "Paste Cells Below".
- Assume s is '110'. Replace the line

```
n = 3
```

  with

```
s = '110'
n = len(s)
```

- Modify the code in the `Oracle()` function to construct the circuit of F for $s = 110$ using the two-step procedure described above.
- Increase the parameter shots from 1 to 1000 when calling `aer_sim.run()` because we want to collect all possible strings whose dot product with s is 0.
- To verify that those output strings which have a dot product of 0 with s, add the code below at the end of the program.

```
# Calculate the dot product of the results
def bdotz(s, a):
    accum = 0
    for i in range(len(s)):
        accum += int(s[i]) * int(a[i])
    return (accum % 2)

for a in answer:
    print( '{}.{} = {} (mod 2)'.format(s, a, bdotz(s,a)) )
```

- Click "Run" to execute the code and record the result. Use the first two output strings which are not '000' to figure out the string s.

Experiment 5: The Oracle Construction method for Simon's Algorithm – II

- Click the cell which contains the code done in Experiment 4. Click the "Edit" button and select "Copy Cells". Click "Edit" again and choose "Paste Cells Below".
- Change the string s from '110' to '01011'.
- Modify the code in the `Oracle()` function to construct the circuit of F for $s = 01011$ using the two-step procedure.
- Click "Run" to execute the code and record the result. Use the four linear independent output strings which are not '000' to figure out the string s.

Click "File" and choose "Save as". Type "lab6" in the entry box and click "Save" to save your work today into the file `lab6.ipynb`.
To turn in your file, use CTRL-ALT-T to open a terminal and type

```
ssh grail
```

and type your password to login to the server grail. Then, type

```
cd QC
turnin -c cis492s -p lab6 lab6.ipynb
```

to electronically turn in your file `lab6.ipynb`.

Shutdown the jupyter notebook and logout the workstation.
Hand in your lab report before your leave.