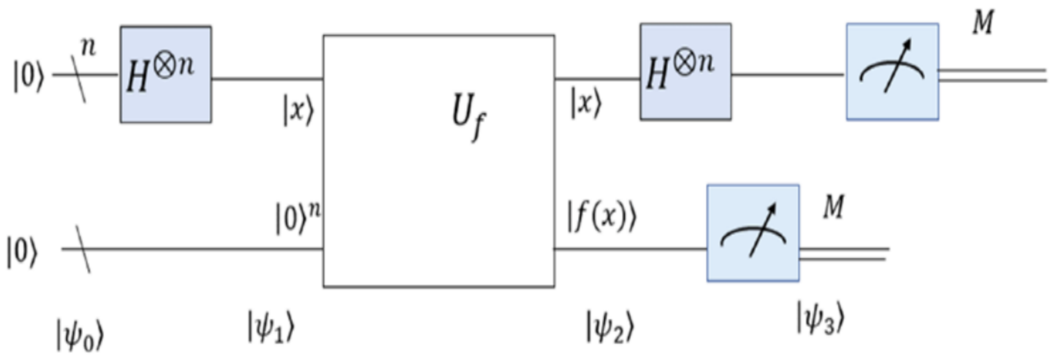
严格推导出Simon算法中每一步量子门作用后系统状态的变化过程，特别详细地写出对后n位和前n位量子比特分别测量后系统状态的变化。

Simon's algorithm



$$
\begin{aligned}
|\psi_0\rangle &= |0\rangle^{\otimes n}|0\rangle^{\otimes n} \\
|\psi_1\rangle &= |H\rangle^{\otimes n}|0\rangle^{\otimes n}|0\rangle^{\otimes n} \\
&= [\frac{1}{\sqrt{2}}(|0\rangle+|1\rangle)]^{\otimes n}|0\rangle^{\otimes n} \\
&\xrightarrow{x=x_{n-1}2^{n-1}+\cdots+x_0 2^0} \frac{1}{\sqrt{2^n}}\sum_{x=0}^{2^{n-1}}|x\rangle|0\rangle^{\otimes n} \\
|\psi_2\rangle &= U_f|\psi_1\rangle = \frac{1}{\sqrt{2^n}}\sum_{x=0}^{2^{n-1}}|x\rangle|f(x)\rangle \\
&= \frac{1}{\sqrt{2^n}}\sum_{x\in\{0,1\}^n}|x,f(x)\rangle \\
|\psi_3\rangle &= (H^{\otimes n}\otimes I)|\psi_2\rangle \\
&= \frac{1}{\sqrt{2^n}}\sum_{x\in\{0,1\}^n}H^{\otimes n}|x\rangle\otimes|f(x)\rangle \\
\because H^{\otimes n}|y\rangle &= \frac{1}{\sqrt{2^n}}\sum_{i=0}^{2^n-1}(-1)^{\langle i,j\rangle}|x_i\rangle \\
&= \frac{1}{\sqrt{2^n}}\sum_{x\in\{0,1\}^n}(-1)^{\langle x,y\rangle}|x\rangle \\
\therefore |\psi_3\rangle &= \frac{1}{\sqrt{2^n}}\sum_{x\in\{0,1\}^n}(\frac{1}{\sqrt{2^n}}\sum_{z\in\{0,1\}^n}(-1)^{\langle z,x\rangle}|z\rangle)\otimes|f(x)\rangle \\
&= \frac{1}{2^n}\sum_{x\in\{0,1\}^n}\sum_{z\in\{0,1\}^n}(-1)^{\langle z,x\rangle}|z,f(x)\rangle
\end{aligned}
$$

- 若 f 是双射函数，则输出值 $f(x)$ 只与特定输入 x 有关，若测量 $|f(x)\rangle$ ，随后记观察到 z 的概率为 $p(z)$ ，则有

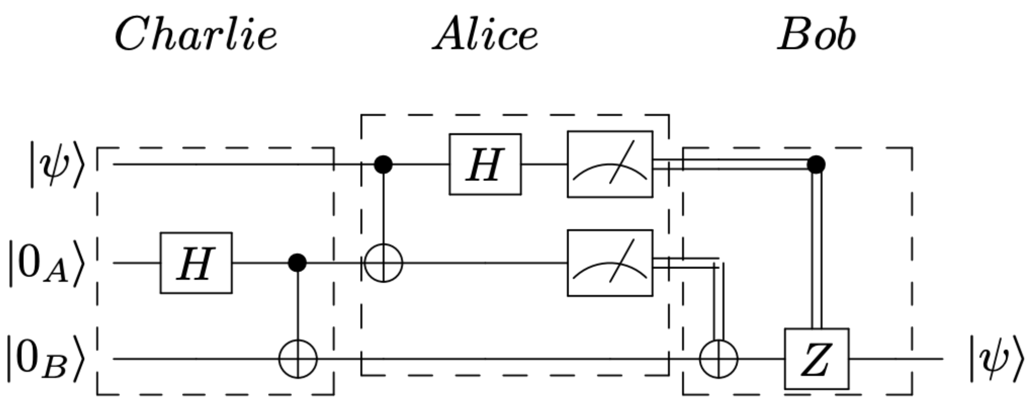
$$
\begin{aligned}
|\psi_3\rangle &= \frac{1}{2^n}\sum_{z\in\{0,1\}^n}|z\rangle\otimes(\sum_{x\in\{0,1\}^n}(-1)^{\langle z,x\rangle}|f(x)\rangle)\\
\therefore p(z) &= \frac{|\sum_{x\in\{0,1\}^n}(-1)^{\langle z,x\rangle}|f(x)\rangle|^2}{\sum_{z\in\{0,1\}^n}|\sum_{x\in\{0,1\}^n}(-1)^{\langle z,x\rangle}|f(x)\rangle|^2}\\
\because \langle f(x),f(y)\rangle &= \begin{cases}1, & x=y,\\ 0, & x\neq y.\end{cases}\\
\therefore |\sum_{x\in\{0,1\}^n}(-1)^{\langle z,x\rangle}|f(x)\rangle|^2 &= \sum_{x\in\{0,1\}^n}(-1)^{\langle z,x\rangle}(-1)^{\langle z,x\rangle}\langle f(x),f(x)\rangle = 2^n\\
\therefore p(z) &= \frac{2^n}{2^{2n}} = \frac{1}{2^n}
\end{aligned}
$$

- 若 f 是二对一函数，记 $c\in\{0,1\}^n$ 为使得 $f(x)=f(y)\Leftrightarrow y=x\oplus c$ 成立的二进制串，则有 $c\neq 0$ 且

$$
\begin{aligned}
|\psi_3\rangle &= \frac{1}{2^n}\sum_{z\in\{0,1\}^n}|z\rangle\otimes(\sum_{x\in\{0,1\}^n}\frac{(-1)^{\langle z,x\rangle}(1+(-1)^{\langle x,c\rangle})}{2}|f(x)\rangle)\\
p(z) &= \frac{|\sum_{x\in\{0,1\}^n}\frac{(-1)^{\langle z,x\rangle}(1+(-1)^{\langle x,c\rangle})}{2}|f(x)\rangle|^2}{\sum_{z\in\{0,1\}^n}|\sum_{x\in\{0,1\}^n}\frac{(-1)^{\langle z,x\rangle}(1+(-1)^{\langle x,c\rangle})}{2}|f(x)\rangle|^2}\\
\because \langle f(x),f(y)\rangle &= \begin{cases}1, & x=y\vee x=y\oplus c\\ 0, & \text{else.}\end{cases}\\
\therefore &|\sum_{x\in\{0,1\}^n}\frac{(-1)^{\langle z,x\rangle}(1+(-1)^{\langle x,c\rangle})}{2}|f(x)\rangle|^2\\
&= \sum_{x\in\{0,1\}^n}\sum_{y\in\{0,1\}^n}\frac{(-1)^{\langle z,c\rangle}(1+(-1)^{\langle z,c\rangle})^2}{4}\langle f(x),f(y)\rangle\\
&= \sum_{x\in\{0,1\}^n}\frac{(-1)^{\langle z,c\rangle}(1+(-1)^{\langle z,c\rangle})^2}{2} = \begin{cases}2^n, & \langle z,c\rangle=0\\ 0, & \langle z,c\rangle=1\end{cases}\\
\therefore p(z) &= \begin{cases}\frac{1}{2^{n-1}}, & \langle z,c\rangle=0\\ 0, & \langle z,c\rangle=1\end{cases}
\end{aligned}
$$

此时可通过求多次测量并求解线性方程组来得到 c

基本量子编程：选择一种开源的量子编程工具，编写程序实现至少一个简单量子算法（例如我们在第一章中介绍过的算法），把你的量子程序上传到大夏学堂，下次上课时分享。



模拟量子隐形传态：

```python
from pyquil.api import WavefunctionSimulator
from netQuil import Agent, QConnect, CConnect, Simulation
from pyquil import Program
from pyquil.gates import *


class Charlie(Agent):
    '''
    Charlie sends Bell pairs to Alice and Bob
    '''

    def run(self):
        # Create bell state pair
        p = self.program
        p += H(0)
        p += CNOT(0, 1)

        self.qsend(alice.name, [0])
        self.qsend(bob.name, [1])


class Alice(Agent):
    '''
    Alice projects her state on her bell state pair from Charlie
    '''

    def run(self):
        p = self.program

        # Define Alice's Qubits
        phi = self.qubits[0]
        qubitsCharlie = self.qrecv(charlie.name)
        a = qubitsCharlie[0]

        # Entangle Ancilla and Phi
        p += CNOT(phi, a)
        p += H(phi)

        # Measure Ancilla and Phi
        p += MEASURE(a, ro[0])
        p += MEASURE(phi, ro[1])


class Bob(Agent):
    '''
    Bob recreates Alice's state based on her measurements
    '''

    def run(self):
        p = self.program

        # Define Bob's qubits
        qubitsCharlie = self.qrecv(charlie.name)
        b = qubitsCharlie[0]
```

```python
        # Prepare State Based on Measurements
        p.if_then(ro[0], X(b))
        p.if_then(ro[1], Z(b))


p = Program()
p += H(2)

# Create Classical Memory
ro = p.declare('ro', 'BIT', 3)

# Create Alice, Bob, and Charlie. Give Alice qubit 2 (phi). Give Charlie qubits
[0,1] (bell state pairs).
alice = Alice(p, qubits=[2], name='alice')
bob = Bob(p, name='bob')
charlie = Charlie(p, qubits=[0, 1], name='charlie')

# Connect agents to distribute qubits and report results
QConnect(alice, bob, charlie)
CConnect(alice, bob)

# Run simulation
Simulation(alice, bob, charlie).run()
wfn = WavefunctionSimulator().wavefunction((p))
print(wfn)
```