浙江大学

《量子计算理论基础与软件系统》
实验报告

实验题目：	Lab1 Quantum Circuit Simulation
姓　　名：	王晓宇
学　　号：	3220104364
电子邮箱：	3220104364@zju.edu.cn
联系电话：	19550222634
授课教师：	卢丽强/尹建伟
助　　教：	储天尧

2024 年 9 月 28 日

Lab1 Quantum Circuit Simulation

Q1

根据 `qubit_simulator` 中的源代码分析 `qubit-simulator` 的基本原理、结构及运行流程。

基本原理

QubitSimulator 通过使用线性代数和矩阵运算来模拟量子计算中的量子比特和量子门操作。量子比特的状态由一个复数向量表示，量子门操作通过矩阵乘法作用于这个状态向量。支持多种常见的量子门操作，并能够测量量子态。

结构

QubitSimulator 类包含以下主要部分：

- 初始化模块

  `__init__`：初始化模拟器，设置量子比特数量，初始化状态向量和电路列表

- 验证模块

  `_validate_qubit_index` 方法：验证量子比特索引是否在有效范围内
  在`Gates类`中我们有`_validate_gate` 方法：验证量子门是否是酉矩阵

- 量子门操作：

  `_apply_gate` 方法：应用给定的量子门到目标量子比特，支持受控门操作
  `h`、`t`、`x`、`cx`、`u`、`cu` 方法：分别应用 Hadamard 门、π/8 门、Not 门、CNot 门、通用 U 门和受控 U 门

- 测量和运行：

  `measure` 方法：测量量子态，返回测量结果的List
  `run` 方法：运行模拟并返回测量结果的Dictionary

- 重置和绘图：

  `reset` 方法：重置模拟器到初始状态
  `plot_wavefunction` 方法：调用matplotlib.pyplot库,绘制波函数的振幅和相位

- 字符串表示和内存大小：

  `__str__` 方法：返回电路的字符串表示,在终端中可以看到
  `__getsize__` 方法：返回实例的总内存大小

运行流程

1. **初始化：**
   创建 QubitSimulator 实例时，指定量子比特数量，初始化状态向量为全零并将第一个元素设置为 1，表示初始态为 |0⟩
2. **应用量子门：**

   调用量子门方法（如 h、t、x 等）时，首先验证量子比特索引，然后构造操作矩阵并应用到状态向量上，更新电路列表
3. **测量量子态：**

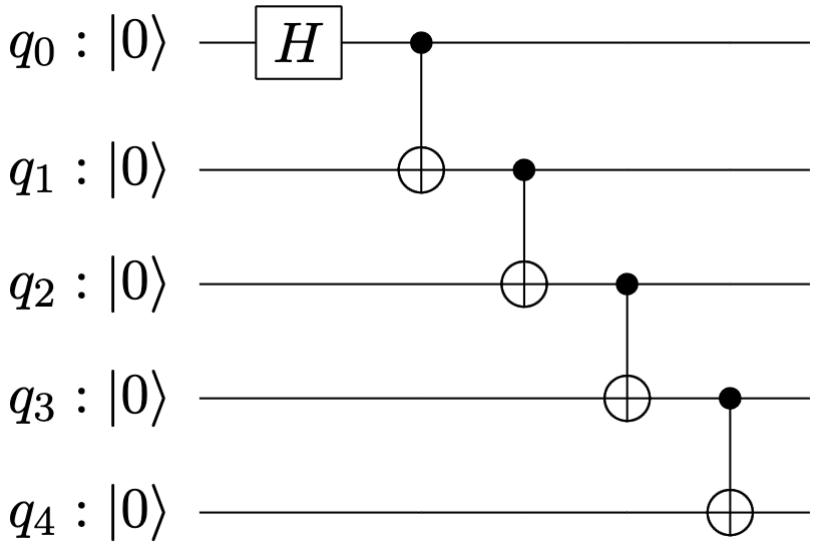
   调用 measure 方法时，计算每个量子态的概率，根据测量次数生成测量结果列表。
4. **运行模拟：**
   调用 run 方法时，执行多次测量并统计结果，返回测量结果的字典

5. 重置和绘图：
   调用 reset 方法时，重置模拟器到初始状态
   调用 plot_wavefunction 方法时，绘制当前量子态的振幅和相位
   调用 **str** 方法时，返回电路的字符串表示

通过上述流程，QubitSimulator 能够模拟量子计算中的基本操作，提供量子态的测量和可视化功能

Q2

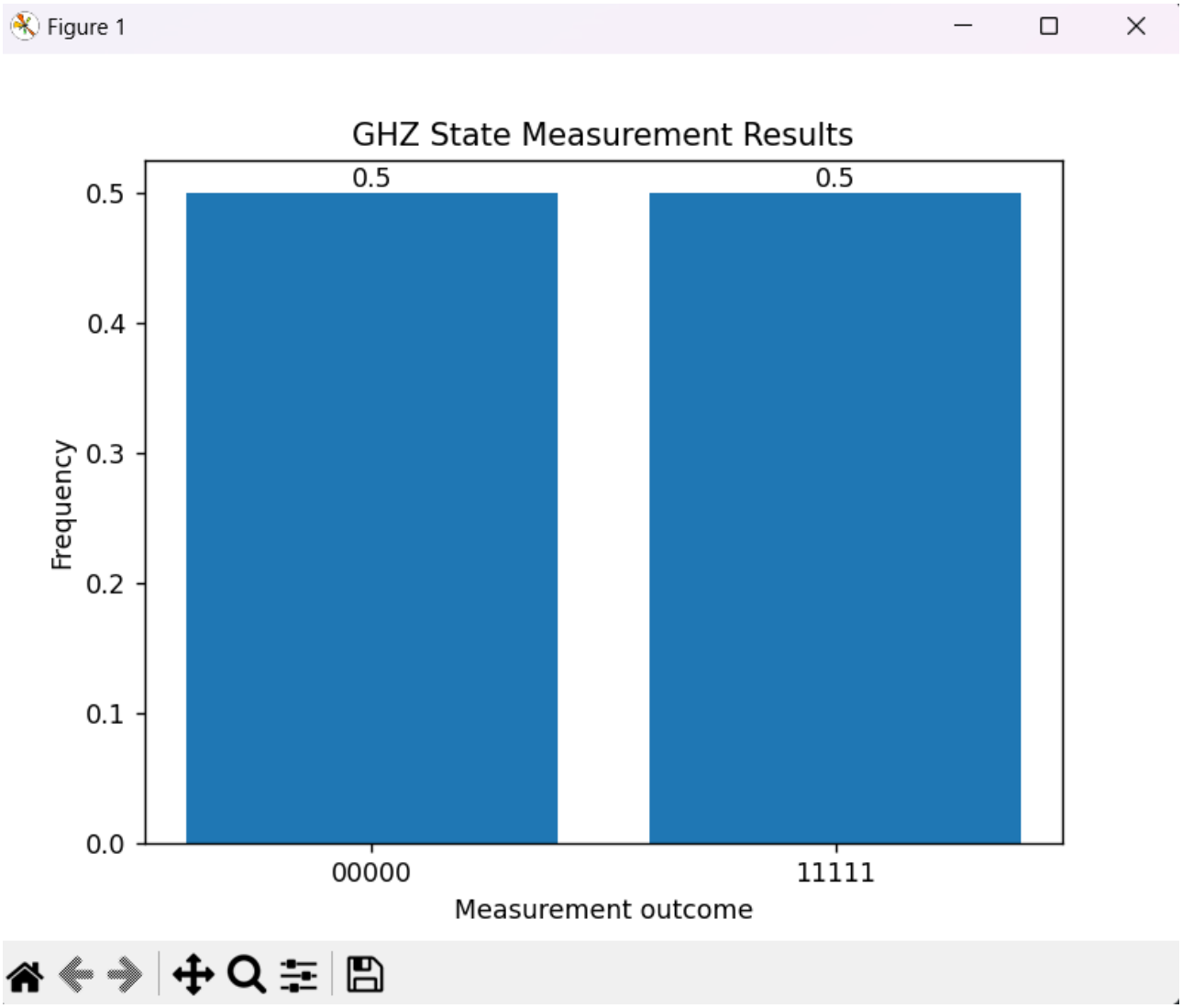
使用 `qubit-simulator` 构造如下图所示的 5-qubit GHZ 电路，并模拟运行，画出结果概率分布直方图。



贴个源码

```python
from qubit_simulator import QubitSimulator
import matplotlib.pyplot as plt


def plot_histogram(result):
    labels, counts = zip(*result.items())
    total_counts = sum(counts)
    frequencies = [float(count) / total_counts for count in counts]

    bar = plt.bar(labels, frequencies)
    plt.xlabel('Measurement outcome')
    plt.ylabel('Frequency')
    plt.title('GHZ State Measurement Results')
```

```
    plt.bar_label(bar, label_type='edge')
    plt.show()

circ = QubitSimulator(5)

circ.h(0)
for i in range(4):
    circ.cx(i, i + 1)
print(circ.__str__())

result = circ.run(shots=1000)

plot_histogram(result)

```



Q3

下段代码构造了一个量子电路，运行这段代码，并尝试调整代码中 `n_qubits` 的值，然后观察量子电路模拟的运行时间。绘制量子电路模拟运行时间与量子电路比特数的关系图，并分析 `qubit-simulator` 进行量子电路模拟的复杂度。

```
import random
import time
from qubit_simulator import QubitSimulator

def apply_circuit(circuit, n):
    circuit.h(n - 1)
    for qubit in range(n - 1):
        circuit.cu(qubit, qubit + 1, random.random() * 3.14, random.random() * 3.14, random.random() * 3.14)

n_qubits = 5 # change this value (<=16)
simulator = QubitSimulator(n_qubits)

t = time.time()
apply_circuit(simulator, n_qubits)
print(time.time() - t)

simulator.run(shots=1000)
```

这里对源代码做一些改进，使得可以自动生成折线图与运行时间图：

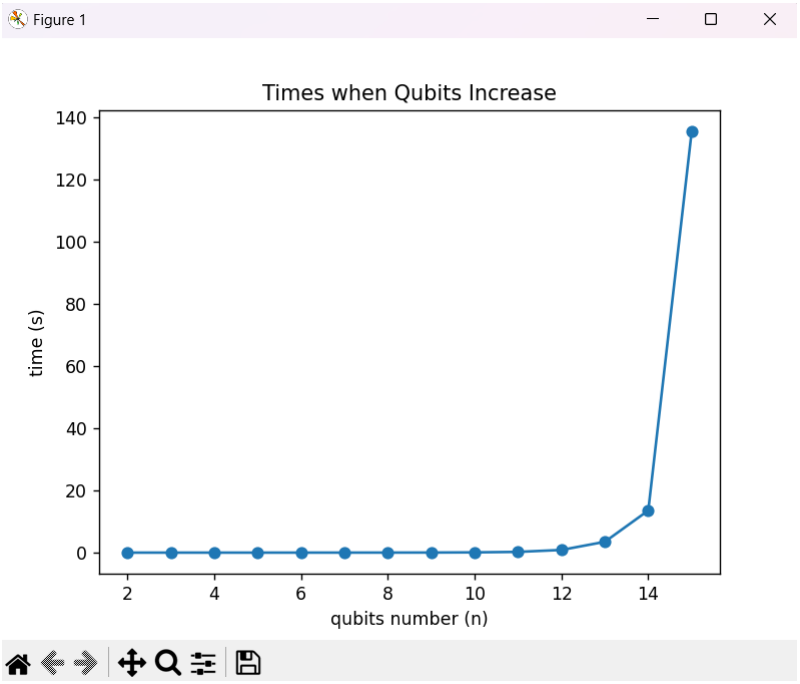
```
import random
import time
from qubit_simulator import QubitSimulator
import matplotlib.pyplot as plt
```

```
import time


def apply_circuit(circuit, n):
    circuit.h(n - 1)
    for qubit in range(n - 1):
        circuit.cu(qubit, qubit + 1, random.random() * 3.14,
random.random() * 3.14, random.random() * 3.14)

# n_qubits = 5 # change this value (<=16)
n_qubits_list = list(range(2, 16))
times = []
for n_qubits in n_qubits_list:
    simulator = QubitSimulator(n_qubits)
    t = time.time()
    apply_circuit(simulator, n_qubits)
    simulator.run(shots=1000)
    elapsed_time = time.time() - t
    times.append(elapsed_time)
    print(f"n_qubits: {n_qubits}, time: {elapsed_time}")
    del simulator
    time.sleep(1)

plt.plot(n_qubits_list, times, marker='o')

plt.title('Times when Qubits Increase')
plt.xlabel('qubits number (n)')
plt.ylabel('time (s)')

plt.show()
```

但是我的电脑只能跑得动2-15个bits OrzOrz

产生的结果如下：

```
n_qubits: 2, time: 0.0
n_qubits: 3, time: 0.0019202232360839844
n_qubits: 4, time: 0.0012431144714355469
n_qubits: 5, time: 0.0
n_qubits: 6, time: 0.0035810470581054688
n_qubits: 7, time: 0.007758617401123047
n_qubits: 8, time: 0.014197587966918945
n_qubits: 9, time: 0.027012109756469727
n_qubits: 10, time: 0.08073234558105469
n_qubits: 11, time: 0.23883342742919922
n_qubits: 12, time: 0.8604371547698975
n_qubits: 13, time: 3.4824724197387695
n_qubits: 14, time: 13.53893494606018
n_qubits: 15, time: 135.37829208374023
```



随着量子比特数量的增加，模拟运行时间呈指数增长。这是因为量子比特数量的增加会导致状态空间的指数级扩展，从而增加了计算复杂度和所需的计算资源

从输出结果可以看出，量子电路模拟运行时间随着量子比特数量的增加呈现出指数增长的趋势，与矩阵运算的空间大小有关，由于矩阵乘法为$2^n * 2^n = 4^n$,因此复杂度随空间大小变化应为$O(4^n)$,由图示结果也不难看出基本吻合。