

量子计算理论基础与软件系统 实验报告

Lab 1 Quantum Circuit Simulation

27.10.2025

redacted

1. qubit_simulator 基本原理

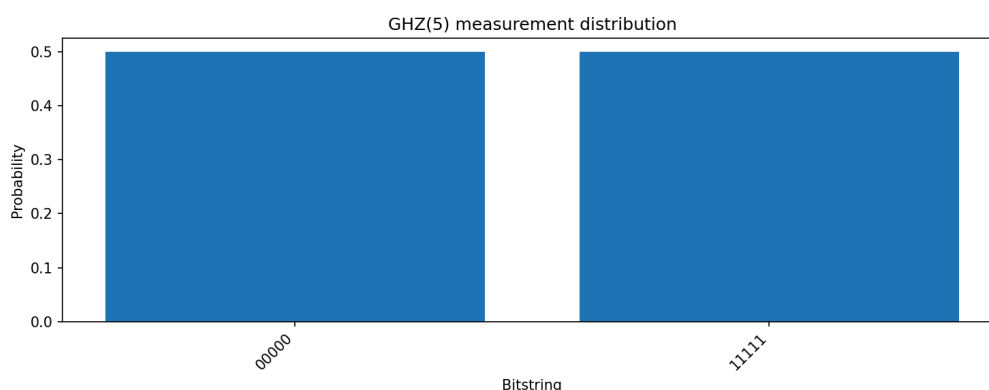
gates.py 规定了一个门类，其中包含多种被预定义的量子门，例如 T 门，H 门，泡利 X 门，通用 U 门等，最后定义一个函数检测门的合法性。

qubit_simulator.py 定义了一个有着一定数量量子比特的量子计算机 QubitSimulator 类，用一个长度为 2^n 的复向量 (state_vector) 表示量子态 $|\psi\rangle \in \mathbb{C}^{2^n}$ ，还有一个 circuit 属性用于存储量子计算机的电路。各种门通过 _apply_gate 函数施加于当前的量子态上，受控门通过构造“条件翻转目标位”的全局矩阵来实现：当控制位为 1 时，在目标位对应的 2×2 子空间中应用单比特门。run 函数通过 measure 函数对量子态进行测量，默认在计算基下测量 ($p_i = |a_i|^2$)，并返回测量结果。

2. GHZ(5) 模拟运行

根据图像，只需要对 qubit 0 施加 H，然后按链式依次施加 CNOT(0→1, 1→2, 2→3, 3→4) 即可得到 GHZ(5) 态。核心代码如下：

```
1 simulator = QubitSimulator(5)
2 simulator.h(0)
3 for i in range(4):
4     simulator.cx(i, i + 1)
5 counts = simulator.run(shots=9999)
6 print("GHZ counts:", counts)
7 plot_histogram(counts, "GHZ(5) measurement distribution", "ghz_5_hist.png")
```



对应命令行输出的量子比特状态：

```
1 $ python main.py
2 GHZ counts: {'00000': 4999, '11111': 5000}
```

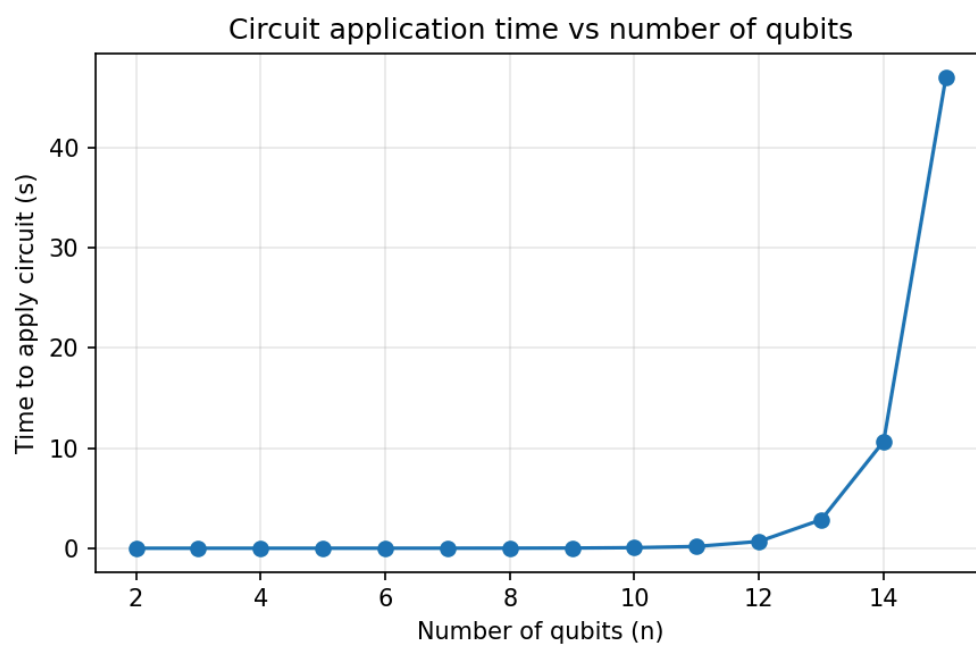
3. 复杂度分析

这个电路先对最高位施加 H，然后依序对相邻比特施加受控 U（随机角度），经过尝试，n=16 时会导致内存不够用，于是遍历 n 从 2 到 15，记录每次运行时间并绘制折线图。

```
1 def apply_circuit(circuit, n):
2     circuit.h(n - 1)
3     for qubit in range(n - 1):
4         circuit.cu(qubit, qubit + 1, random.random() * 3.14,
5                    random.random() * 3.14, random.random() * 3.14)
6
7 ns = list(range(2, 16)) # number of qubits, iterated
8 times = [] # record times
9 for n in ns:
10     sim = QubitSimulator(n)
11     t = time.time()
12     apply_circuit(sim, n)
13     elapsed = time.time() - t
14     times.append(elapsed)
15     print(f'n={n}, time={elapsed:.6f}s')
16     sim.run(shots=1000)
```

终端输出时间如下：

```
1 $ python test.py
2 n=2, time=0.000580s
3 n=3, time=0.000604s
4 n=4, time=0.000581s
5 n=5, time=0.000689s
6 n=6, time=0.001581s
7 n=7, time=0.004001s
8 n=8, time=0.008008s
9 n=9, time=0.019244s
10 n=10, time=0.056722s
11 n=11, time=0.180028s
12 n=12, time=0.679046s
13 n=13, time=2.852152s
14 n=14, time=10.674568s
15 n=15, time=47.061871s
```



可见时间复杂度随着量子比特数的增加呈指数增长趋势（由于量子门在 apply 时需要使用 Kronecker 张量积进行拓展，于是时间复杂度为 $O(2^n)$ ），这与量子态向量的存储需求（存储长度为 2^n 的矢量）是一致的。