Intro to Cirq
Cirq vs Qiskit

Nitya

Agenda

- Create states and circuits
- Add gates to perform computation
- Simulate
- Alternative representations of qubits

Creating qubits and circuits

Cirq

- s = cirq.NamedQubit('source’)
- qlist = cirq.LineQubit.range(4)
- Circuit = cirq.Circuit()

Qiskit

- qreg = QuantumRegister(4)
- creg = ClassicalRegister(4)
- Circuit = QuantumCircuit(qreg,creg)
- Alternatively,

  Circuit1 = QuantumCircuit(4,4)

Display qubits (and circuits)

Cirq

- print(cirq.LineQubit(1))
- print(qlist[1])
- For named qubits, print(s)
- print(Circuit)

Qiskit

- Circuit.draw(‘mpl’)
- Circuit1.draw()

Task 1
1.1:

- Create a quantum circuit with 10 qubits.

Add gates to circuit

Cirq

- **circuit.append(GATE(qubit))**
- **X, Z, H, CX, CZ, SWAP, CCX etc.**
- Circuit.append(X(qlist[2]))
- Circuit.append(CX(qlist[2], qlist[0]))
- Circuit.append(CCX(qlist[0], qlist[1], qlist[2]))

Qiskit

- **circuit.gate(qubit)**
- **x, z, h, cx, cz, swap, ccx, etc.**
- Circuit.x(qreg[2])
- Circuit.cx(qreg[2],qreg[0])
- Circuit.ccx(0,1,2)

Task 1
1.2:

- Apply H gate to qubit 0
- Apply nine $CNOT$ gates where qubit 0 is the control and qubit i is the target for i=1⋯9 .
- Draw your circuit.

Fancy gate operations
Apply multiple gates at once:

Cirq

- Use ‘on_each’
- circuit.append(H.on_each(*qlist)
)

Qiskit

- Just apply the gate on the quantum register
- circuit.h(qreg)

https://realpython.com/python-kwargs-and-args/

Fancy gate operations
Controlled version of a gate

Cirq

- CCCH = H(qlist[2]).controlled_by(qlist[0], qlist[1],qlist[3])
- circuit2.append(CCCH)

Qiskit

- Define custom gate:

CCCH = HGate().control(3)

- Specify control and target qubits

circuit.append(CCCH,[0,1,2,3])

Task 2

- Create a quantum circuit with 10 qubits.

- Apply H gates to all qubits.
- Apply X gate to qubit 0 controlled by qubits 1-9
- Apply H gates to all qubits.
- Draw your circuit.

FYI fancier gate operations DIY version of gates

Cirq

- ROOTX = X**0.5

circuit2.append(ROOTX(qlist[1]))

https://quantumcomputing.stackexchange.com/questions/17494/what-is-the-square-root-of-the-not-gate
https://quantumcomputing.stackexchange.com/questions/15381/square-root-of-pauli-operators-is-there-a-common-convention-to-define-them-uniq?rq=1

Qiskit

ROOTX = XGate().power(exponent=0.5)

- circuit2.append(ROOTX,[1])

Step 1: Measure

Cirq

- Circuit.append(measure(*qList, key = 'result')

Qiskit

- Circuit.measure(qreg,creg)

https://www.w3schools.com/python/python_dictionaries.asp

1.1: Measure all qubits

Cirq

- result = samples.measurements["result"]

Qiskit

- Circuit.measure(qreg,creg)

1.2 Measure specific qubits

Cirq

- circuit.append(measure(qlist[0], key='result'))

Qiskit

- circuit.measure(qreg[0],creg[0])

Step 2: Simulate

Cirq

- Sim = cirq.Simulator()
- Samples = sim.run(circuit, repetitions = 1000)
- Display results

Print(samples.histogram(key = ‘result’)

Qiskit

Backend = Aer.get_backend(‘qasm_simulator)

Job = execute(circuit,backend,shots = 1000)

Result = job.get_result()

Counts = results.get_counts()

Print(counts)

Task 3

- Implement the circuit in Task 1. Measure all the qubits and simulate your circuit for 1000 times.

State representation (no measurement)

Cirq

- circuit.append(H(qlist[0]))

- S = cirq.Simulator()
- results=s.simulate(circuit)
- print(results)

Qiskit

circuit.h(qreg[0])

vsimulator = Aer.get_backend('statevector_simulator')

job = execute(circuit,vsimulator,shots=1)

state = job.result().get_statevector()

print(state)

Task 4

- Create a quantum circuit with 4 qubits. Apply Hadamard gate to each qubit and CZ gate to qubits 0 and 1. Use the simulator without measuring the circuit. Check the entries with negative sign.

Unitary matrix representation

Cirq

- Use ‘dot unitary’
- (cirq.unitary(CX))
- cirq.unitary(circuit)

Qiskit

- Use ‘dot to_matrix’
- (CXGate().to_matrix())
- usimulator = Aer.get_backend('unitary_simulator')

job = execute(circuit,usimulator,shots=1)

matrix = job.result().get_unitary()

print('Unitary matrix representation of H operator on 2 qubits.')

print(matrix)

Summary

✓Creating qubits

✓Creating circuits

✓Adding different gates

✓Simulating circuits

✓Making measurements

✓Looking at statevectors and unitaries