DECOMPOSING A MULTI-CONTROLLED TOFFOLI GATE

Sasanka Dowarah, Saeed Rahmanian Koshkaki, Michael H. Kolodrubetz

May 30, 2022

The problem

We want to decompose a MCX gate with 14 control qubits and a maximum of 5 ancilla qubits using a minimal number of single and double qubits CX gates.

Solution

Our approach will be to write the MCX gates with 14 control qubits in terms of other MCX gates with fewer control qubits. Which then using the function `transpile` of qiskit can be decomposed into single and double qubit gates. This can be done in several ways, however, in order to achieve the minimum circuit depth using 5 ancilla qubits in our circuit, the optimal solution is to use ten 3-qubit gates and one 4-qubit gates as shown in figure 1. After writing the circuit using MCX gates with 3 and 4 controls, using the function `transpile`, these two kinds of gates are efficiently decomposed into a sequence of one and two-qubit gates. These decomposition result in a circuit of depth 130 with 198 U gates and 176 CX gates.

The 5 ancilla qubits are prepared in the state $|0\rangle$.

The code

Imports the necessary qiskit functions.

```
from qiskit import QuantumRegister, ClassicalRegister,
    QuantumCircuit, transpile
```

We set up the circuit with 14 control qubits and 5 ancilla qubits, and one target qubit.

```
qr = QuantumRegister(14, 'c') # 14 control qubits.
target_qubit = QuantumRegister(1,'t') # target qubit.
anc = QuantumRegister(5, 'ancilla') # ancilla qubits.
qc = QuantumCircuit(qr,target_qubit, anc)
```

Then we create ten 3-qubit gates and one 4-qubit gate.

```
# 3-qubit gates.
qc.mcx(qr[0:3],anc[0])
qc.mcx(qr[3:6],anc[1])
qc.mcx(qr[6:9],anc[2])
qc.mcx(qr[9:12],anc[3])
qc.mcx([qr[12],qr[13],anc[0]],anc[4])

# 4-qubit gate
qc.mcx(anc[1:5],target_qubit[0])


# Apply the 3-qubit gates in reverse to return controls and
    ancillas to original state.
qc.mcx([qr[12],qr[13],anc[0]],anc[4])
qc.mcx(qr[9:12],anc[3])
qc.mcx(qr[6:9],anc[2])
qc.mcx(qr[3:6],anc[1])
qc.mcx(qr[0:3],anc[0])

qc.draw('mpl')
```

Using the function `transpile` of qiskit, we then decompose the above circuit into single and double qubit gates. We have chosen the basis set as {U,CX}, as it gives the minimum circuit depth.

```
circuit_decomposition= transpile(qc, basis_gates=['u','cx'],
    optimization_level = 3)

print("Circuit depth =", circuit_decomposition.depth())

print("Number of 1 and 2 qubit gates =",circuit_decomposition.
    count_ops())
```

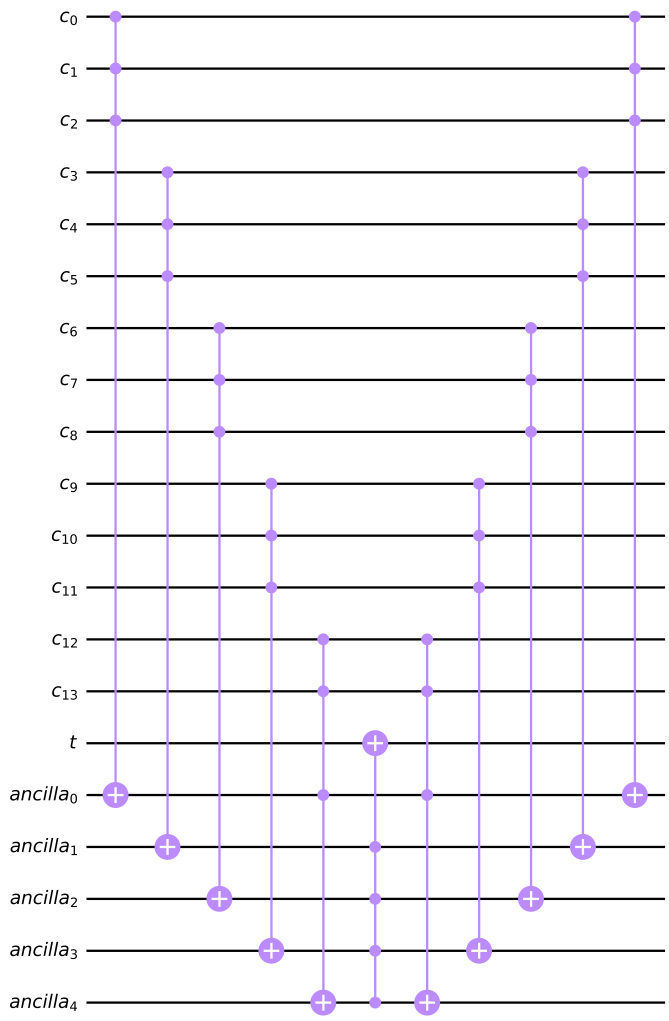

Figure 1: The MCX gate with 14 control qubits and 5 auxiliary qubits is split into ten 3-qubit gates and one 4-qubit gates.