**Quantum Circuit Optimization with SPIRAL: A First Look**

### Quantum State - Linear Algebra

**QC is State Linear Algebra**

- **Superposition:** qubit states are unit-norm complex vectors on the Bloch Sphere.
- **States with respect to a certain basis denote the “square root probability”** that the value is read when measured in that basis.

\[
\begin{bmatrix}
\frac{1}{\sqrt{2}} \left| 0 \right> + \frac{1}{\sqrt{2}} \left| 1 \right>
\end{bmatrix}
\]

- For a qubit, the coherence of the system is 100% for the both states without measurement.

**Quantum State – Linear Algebra**

- Qubits are unit-norm, 2D complex vectors (4D space).
- Qubits can be “measured” in any orthogonal basis. Outcome probability is the magnitude squared of the projection onto measured basis.

\[
q = \begin{bmatrix} a \\ b \end{bmatrix}, \quad |q| = 1
\]

- Qubits are non-adjacent qubits.

**Qubit Connectivity**

- Qubits must be physically adjacent to interact.

### Problem Definition

**Qubit Topology**

- In a qubit operation on 2-qubit basis – QC $G_{2}^{2}$ – the probability in the basis is the probability of successful connectivity measurements.

\[
\begin{bmatrix}
|00\rangle, |01\rangle, |10\rangle, |11\rangle
\end{bmatrix}
\]

- Non-trivial to find practical usage on large circuits.

**Quantum Fourier Transform (QFT)**

- **Shor’s Algorithm**

\[
\begin{bmatrix}
|00\rangle
\end{bmatrix} \rightarrow \begin{bmatrix}
|00\rangle
\end{bmatrix}
\]

**Quantum Algorithm Search**

- Many existing solvers are QAOA Optimizers

\[
|\psi\rangle = \begin{bmatrix}
|0\rangle \\ |1\rangle
\end{bmatrix}
\]

- The true problem: search over the circuit space for a given matrix.

### Our Approach: SPIRAL

**SPIRAL Quantum Compiler**

- **Transaction Algorithms:**

\[
\begin{bmatrix}
|0\rangle \\ |1\rangle \\ |2\rangle \\ |3\rangle \\ |4\rangle \\ |5\rangle \\ |6\rangle \\ |7\rangle
\end{bmatrix}
\]

- **Quantum Fourier Transform**

\[
\begin{bmatrix}
|0\rangle \\ |1\rangle \\ |2\rangle \\ |3\rangle \\ |4\rangle \\ |5\rangle \\ |6\rangle \\ |7\rangle
\end{bmatrix}
\]

- **Optimization**

\[
|\psi\rangle = \begin{bmatrix}
|0\rangle \\ |1\rangle
\end{bmatrix}
\]

**Rewrite Rules**

- Perform direct or conditional substitutions to collapse gates and simplify circuit description.

**Transform Algorithms:**

- **Rewriting Rules:**

\[
\begin{bmatrix}
|0\rangle \\ |1\rangle \\ |2\rangle \\ |3\rangle \\ |4\rangle \\ |5\rangle \\ |6\rangle \\ |7\rangle
\end{bmatrix}
\]

**First Results**

- We tested SPIRAL against IBM’s Qiskit optimizer with Cost(|r| + |CNOT|) ($|CNOT|$.)
- Executed final QASM code on IBM’s Tenerife and Bogota devices.

**SPIRAL System Overview**

- **Generative, Abstraction-Based Compiler:**

\[
\begin{bmatrix}
|0\rangle \\ |1\rangle \\ |2\rangle \\ |3\rangle \\ |4\rangle \\ |5\rangle \\ |6\rangle \\ |7\rangle
\end{bmatrix}
\]

- **The Embed Operation**

\[
\begin{bmatrix}
|0\rangle \\ |1\rangle \\ |2\rangle \\ |3\rangle \\ |4\rangle \\ |5\rangle \\ |6\rangle \\ |7\rangle
\end{bmatrix}
\]

**Rewrite Rules**

- Perform direct or conditional substitutions to collapse gates and simplify circuit description.

**Transform Algorithms:**

- **Rewriting Rules:**

\[
\begin{bmatrix}
|0\rangle \\ |1\rangle \\ |2\rangle \\ |3\rangle \\ |4\rangle \\ |5\rangle \\ |6\rangle \\ |7\rangle
\end{bmatrix}
\]

**Optimization**

\[
|\psi\rangle = \begin{bmatrix}
|0\rangle \\ |1\rangle \\ |2\rangle \\ |3\rangle \\ |4\rangle \\ |5\rangle \\ |6\rangle \\ |7\rangle
\end{bmatrix}
\]

**Multi-Qubit Computation**

- Parallel gate applications are tensor products

\[
U = \begin{bmatrix}
\rho & \sigma
\end{bmatrix}
\]

**First Results**

- We tested SPIRAL against IBM’s Qiskit optimizer with Cost(|r| + |CNOT|) ($|CNOT|$.)
- Executed final QASM code on IBM’s Tenerife and Bogota devices.

**SPIRAL System Overview**

- **Generative, Abstraction-Based Compiler:**

\[
\begin{bmatrix}
|0\rangle \\ |1\rangle \\ |2\rangle \\ |3\rangle \\ |4\rangle \\ |5\rangle \\ |6\rangle \\ |7\rangle
\end{bmatrix}
\]

- **The Embed Operation**

\[
\begin{bmatrix}
|0\rangle \\ |1\rangle \\ |2\rangle \\ |3\rangle \\ |4\rangle \\ |5\rangle \\ |6\rangle \\ |7\rangle
\end{bmatrix}
\]

**Rewrite Rules**

- Perform direct or conditional substitutions to collapse gates and simplify circuit description.

**Transform Algorithms:**

- **Rewriting Rules:**

\[
\begin{bmatrix}
|0\rangle \\ |1\rangle \\ |2\rangle \\ |3\rangle \\ |4\rangle \\ |5\rangle \\ |6\rangle \\ |7\rangle
\end{bmatrix}
\]

**Optimization**

\[
|\psi\rangle = \begin{bmatrix}
|0\rangle \\ |1\rangle \\ |2\rangle \\ |3\rangle \\ |4\rangle \\ |5\rangle \\ |6\rangle \\ |7\rangle
\end{bmatrix}
\]

**Multi-Qubit Computation**

- Parallel gate applications are tensor products

\[
U = \begin{bmatrix}
\rho & \sigma
\end{bmatrix}
\]

**First Results**

- We tested SPIRAL against IBM’s Qiskit optimizer with Cost(|r| + |CNOT|) ($|CNOT|$.)
- Executed final QASM code on IBM’s Tenerife and Bogota devices.