Global CNOT Synthesis with Holes

Ewan Murphy and Aleks Kissinger

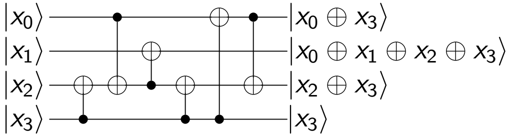
Quantum Physics and Logic (QPL) Conference

2023

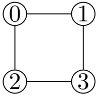
Limitations of NISQ Computers

Connectivity Constraints

Circuit



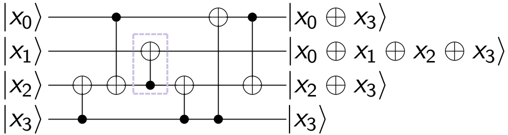
Constraint Topology



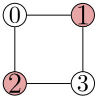
Limitations of NISQ Computers

Connectivity Constraints

Circuit



Constraint Topology

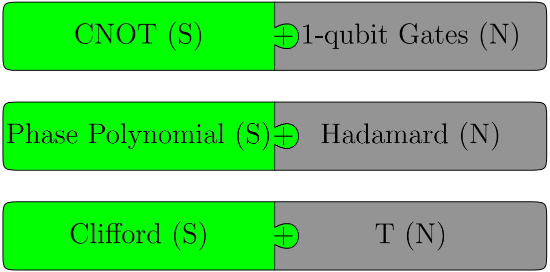


Circuit Synthesis

- Compilation method to overcome NISQ computers shortcommings
- Usually works for circuits made from a specific gate set
- Generates improved circuit, e.g with fewer gates or respecting connectivity constraints, from an efficent representation of the original circuit

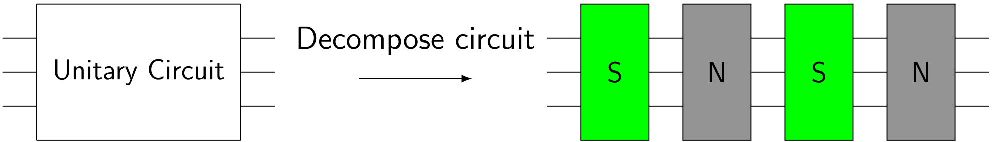
Unitary Decomposition

Gate Sets



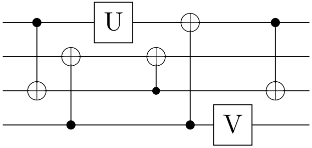
Unitary Decomposition

Slicing

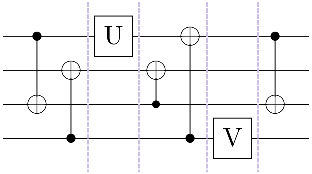


Unitary Decomposition

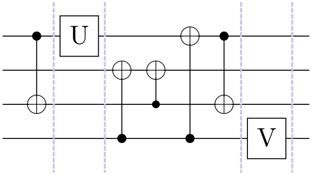
Slicing



(a) General Circuit



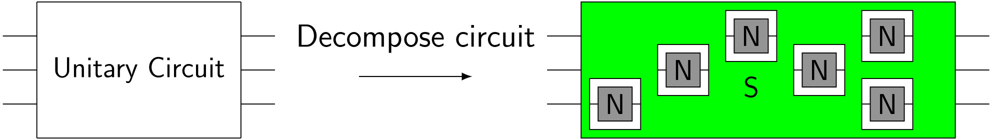
(b) Naïve slicing



(c) Alternative slicing

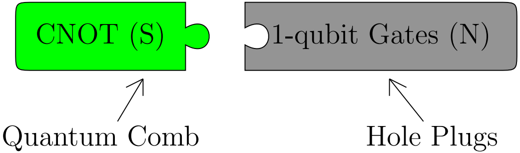
Unitary Decomposition

Quantum Combs

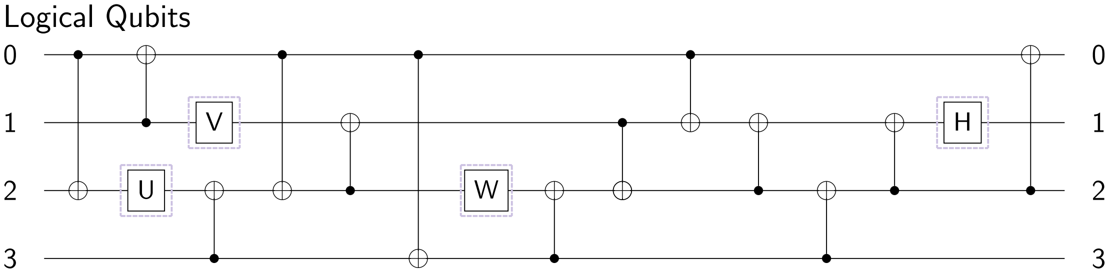


The problem now becomes how to synthesise a quantum comb

CNOT Synthesis

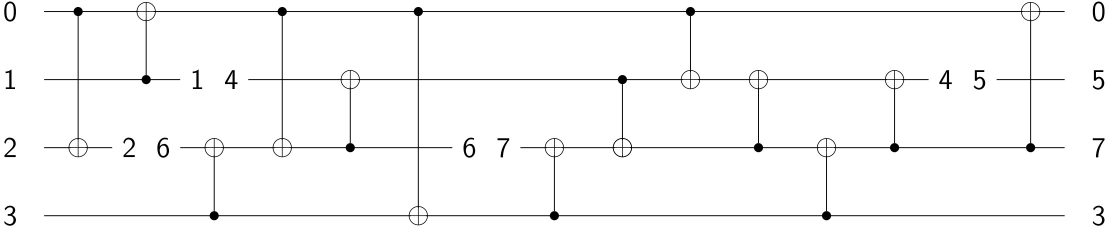


CNOT Combs



CNOT Combs

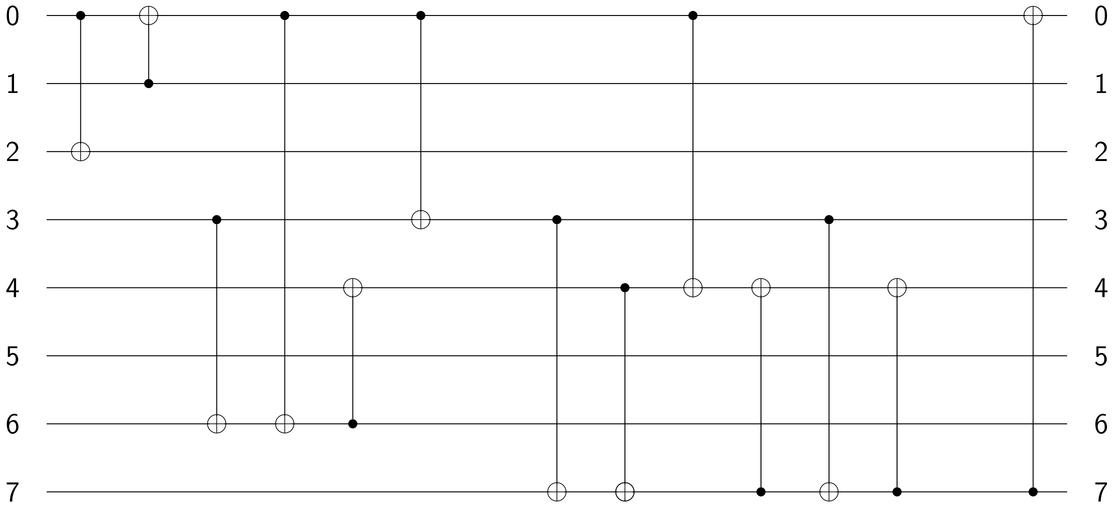
Temporal Qubits



$\mathcal{H} = \{(1,4),(2,6),(6,7),(4,5)\}$

$p :: \{(1,4) \mapsto V, (4,5) \mapsto H, (2,6) \mapsto U, (6,7) \mapsto W\}$

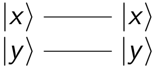
CNOT Combs



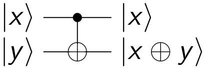
RowCol - Synthesis Algorithm for CNOT Circuits

Circuit Representation

Identity Gate



CNOT Gate



CNOT(c, t) corresponds to R(c, t)

Identity Parity Matrix

$$\begin{array}{c} \\ x' \\ y' \end{array} \begin{array}{c} \begin{array}{cc} x & y \end{array} \\ \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \end{array}$$

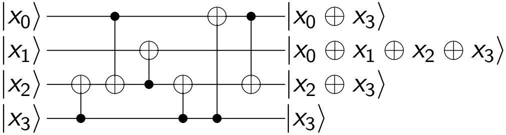
CNOT Parity Matrix

$$\begin{array}{c} \\ x' \\ y' \end{array} \begin{array}{c} \begin{array}{cc} x & y \end{array} \\ \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} \end{array}$$

RowCol - Synthesis Algorithm for CNOT Circuits

Circuit Representation

CNOT Circuit



CNOT(c, t) corresponds to R(c, t)

Circuit Parity Matrix

$$
\begin{array}{c} \\ x_0' \\ x_1' \\ x_2' \\ x_3' \end{array}
\begin{array}{c} \begin{array}{cccc} x_0 & x_1 & x_2 & x_3 \end{array} \\ \begin{pmatrix} 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix} \end{array}
$$

RowCol - Synthesis Algorithm for CNOT Circuits

Algorithm

- RowCol reduces a parity matrix to the identity by eliminating the row and column for each qubit.
- RowCol can synthesise to constrained architecures:
  - Qubit being eliminated has to correspond to non-cutting vertex.
  - This is done using Steiner trees.

Eliminate Row 1 and Column 1:

$$\begin{pmatrix} 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix} \xrightarrow{R_1 := R_0 + R_1} \begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix} \xrightarrow{R_0 := R_3 + R_0} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Eliminate Column 2:

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix} \xrightarrow{R_1 := R_2 + R_1} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix} \xrightarrow{R_1 := R_3 + R_1} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

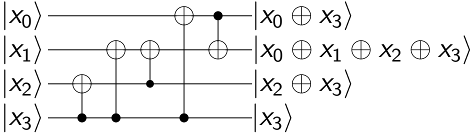
Eliminate Column 3:

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix} \xrightarrow{R_2 := R_3 + R_2} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

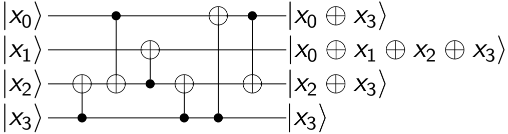
We have reached the identity matrix meaning our synthesis process is over.

Row Operations: $R(0,1)$, $R(3,0)$, $R(2,1)$, $R(3,1)$, $R(3,2)$.

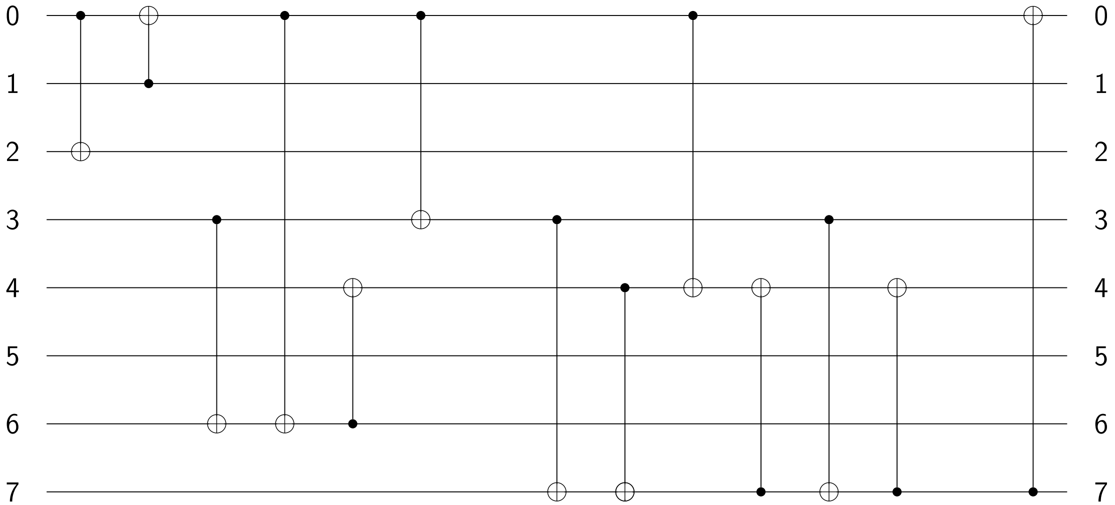
New Circuit



Old Circuit



CombSynth - Synthesis Algorithm for CNOT combs



CombSynth - Synthesis Algorithm for CNOT combs

$$
\begin{array}{c c}
 & \begin{array}{cccccccc} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{array} \\
\begin{array}{c} 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \end{array} &
\begin{pmatrix}
0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\
1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\
1 & 1 & 0 & 1 & 1 & 0 & 1 & 1
\end{pmatrix}
\end{array}
$$

$$\mathcal{H} = \{(1,4),(2,6),(6,7),(4,5)\}$$

$$t(0) = 0, t(1) = 5, t(2) = 7, t(3) = 3$$

CombSynth - Synthesis Algorithm for CNOT combs

	0	1	2	3	4	5	6	7
0	0	0	0	1	1	0	1	1
1	0	1	0	0	0	0	0	0
2	1	0	1	0	0	0	0	0
3	1	1	0	1	0	0	0	0
4	1	1	0	0	1	0	1	0
5	0	0	0	0	0	1	0	0
6	1	1	0	1	0	0	1	0
7	1	1	0	1	1	0	1	1

Generate sub-matrix →

	0	1	2	3	4	5	6	7
0	0	0	0	1	1	0	1	1
5	0	0	0	0	0	1	0	0
7	1	1	0	1	1	0	1	1
3	1	1	0	1	0	0	0	0

A temporal qubit can be extracted if its row and column in the full parity matrix can be eliminated by only row operations on the submatrix.

CombSynth - Synthesis Algorithm for CNOT combs

Initial sub-matrix	Eliminated sub-matrix	Row operations required for reduction
$\begin{array}{c} \\ 0 \\ 5 \\ 7 \\ 3 \end{array}\begin{array}{c} \begin{array}{cccccccc} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{array} \\ \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}$	$\begin{array}{c} \\ 0 \\ 5 \\ 7 \\ 3 \end{array}\begin{array}{c} \begin{array}{cccccccc} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{array} \\ \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}$	N/A

$$\mathcal{H} = \{(1,4),(2,6),(6,7),(4,5)\}$$

$$t(0) = 0, t(1) = 5, t(2) = 7, t(3) = 3$$

CombSynth - Synthesis Algorithm for CNOT combs

Initial sub-matrix	Eliminated sub-matrix	Row operations required for reduction
$\begin{array}{c} \\ 0 \\ 4 \\ 7 \\ 3 \end{array} \begin{array}{c} \begin{array}{cccccccc} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{array} \\ \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}$	$\begin{array}{c} \\ 0 \\ 4 \\ 7 \\ 3 \end{array} \begin{array}{c} \begin{array}{cccccccc} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{array} \\ \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}$	$R(7,0), R(3,7),$ $R(4,7), R(0,7)$

$$\mathcal{H} = \{(1,4),(2,6),(6,7)\}$$

$$t(0) = 0,\ t(1) = 4,\ t(2) = 7,\ t(3) = 3$$

CombSynth - Synthesis Algorithm for CNOT combs

Initial sub-matrix	Eliminated sub-matrix	Row operations required for reduction
$\begin{array}{c} \\ 0 \\ 4 \\ 6 \\ 3 \end{array}\begin{array}{c} \begin{array}{cccccccc} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{array} \\ \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}$	$\begin{array}{c} \\ 0 \\ 4 \\ 6 \\ 3 \end{array}\begin{array}{c} \begin{array}{cccccccc} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{array} \\ \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}$	$R(6,4), R(3,6)$

$$\mathcal{H} = \{(1,4),(2,6)\}$$

$$t(0) = 0, t(1) = 4, t(2) = 6, t(3) = 3$$

CombSynth - Synthesis Algorithm for CNOT combs

Initial sub-matrix	Eliminated sub-matrix	Row operations required for reduction
$\begin{array}{c} \\ 0 \\ 4 \\ 2 \\ 3 \end{array}\begin{array}{c} \begin{array}{cccccccc} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{array} \\ \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}$	$\begin{array}{c} \\ 0 \\ 4 \\ 2 \\ 3 \end{array}\begin{array}{c} \begin{array}{cccccccc} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{array} \\ \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}$	$R(3,4), R(0,3)$

$$\mathcal{H} = \{(1,4)\}$$

$$t(0) = 0, t(1) = 2, t(2) = 7, t(3) = 3$$

CombSynth - Synthesis Algorithm for CNOT combs

Initial sub-matrix	Eliminated sub-matrix	Row operations required for reduction
$\begin{array}{c} \\ 0 \\ 4 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{cccccccc} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{array} \\ \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}$	$\begin{array}{c} \\ 0 \\ 4 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{cccccccc} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{array} \\ \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}$	$R(0, 4)$

$$\mathcal{H} = \{(1, 4)\}$$

$$t(0) = 0, t(1) = 4, t(2) = 2, t(3) = 3$$

CombSynth - Synthesis Algorithm for CNOT combs

Initial sub-matrix	Eliminated sub-matrix	Row operations required for reduction
$\begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \end{array}\begin{array}{c} \begin{array}{cccccccc} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{array} \\ \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}$	$\begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \end{array}\begin{array}{c} \begin{array}{cccccccc} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{array} \\ \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}$	$R(0,2), R(1,0)$

$$\mathcal{H} = \{\}$$

$$t(0) = 0, t(1) = 1, t(2) = 2, t(3) = 3$$

CombSynth - Synthesis Algorithm for CNOT combs

Initial sub-matrix	Eliminated sub-matrix	Row operations required for reduction
$\begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \end{array}\begin{array}{c} \begin{matrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{matrix} \\ \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}$	$\begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \end{array}\begin{array}{c} \begin{matrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{matrix} \\ \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}$	$R(1,2)$

$$\mathcal{H} = \{\}$$

$$t(0) = 0, t(1) = 1, t(2) = 2, t(3) = 3$$

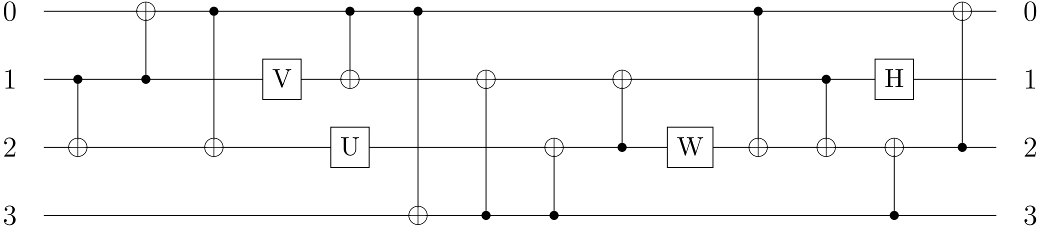
CombSynth - Synthesis Algorithm for CNOT combs

Row Operations: $R(7,0)$,$R(3,7)$,$R(4,7)$,$R(0,7)$,$R(6,4)$,$R(3,6)$,$R(3,4)$,$R(0,3)$, $R(0,4)$,$R(0,2)$,$R(1,0)$,$R(1,2)$

Holes: $\mathcal{H} = \{(1,4),(2,6),(6,7),(4,5)\}$

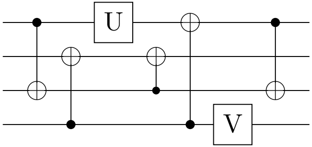
Plugs: $p :: \{(1,4) \mapsto V, (4,5) \mapsto H, (2,6) \mapsto U, (6,7) \mapsto W\}$

CombSynth - Synthesis Algorithm for CNOT combs

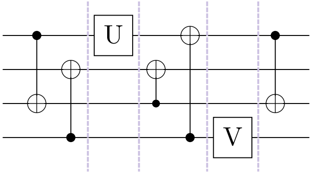


Computational Experiments

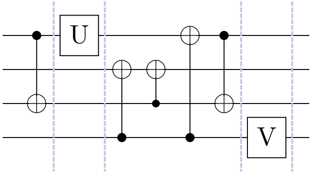
Slicing



(a) General Circuit



(b) Naïve slicing



(c) Alternative slicing

Computational Experiments

Experimental Parameters

Architectures

- 9q-square
- 16-square
- regetti_16q_aspen
- bm_qx5
- ibm_q20_tokyo

CNOT Count

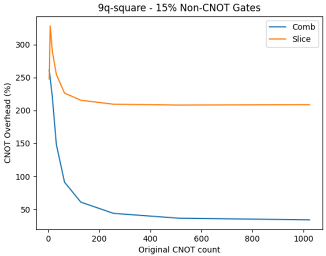
- 4
- 8
- 16
- 32
- 64
- 128
- 256
- 512
- 1024

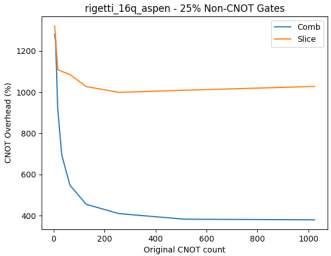
Non-CNOT Proportion

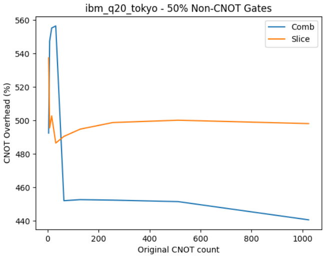
- 5%
- 15%
- 25%
- 50%

Computational Experiments

Graph Selection







Computational Experiments

Results

Architectures	5% Non-CNOT Gates	
	Comb	Slice
9q-square	-43.1%	80.79%
16q-square	13.11%	344.5%
regetti_16q_aspen	47.31%	555.4%
bm_qx5	32.27%	461.9%
ibm_q20_tokyo	33.17%	393.1%

Computational Experiments

Results

Architectures	15% Non-CNOT Gates	
	Comb	Slice
9q-square	34.12%	208.7%
16q-square	154.4%	511.1%
regetti_16q_aspen	231.2%	893.1%
bm_qx5	197.2%	698.8%
ibm_q20_tokyo	183.6%	481.0%

Computational Experiments

Results

Architectures	25% Non-CNOT Gates	
	Comb	Slice
9q-square	91.93%	255.9%
16q-square	263.9%	564.6%
regetti_ 16q_aspen	379.6%	1027%
bm_qx5	322.4%	783.6%
ibm_q20_tokyo	289.3%	500.2%

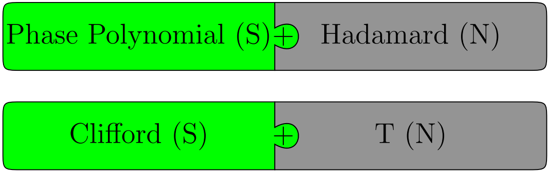
Architectures	50% Non-CNOT Gates	
	Comb	Slice
9q-square	182.1%	306.8%
16q-square	437.0%	606.9%
regetti_16q_aspen	614.9%	1119%
bm_qx5	527.8%	837.7%
ibm_q20_tokyo	440.7%	498.2%

Summary and Conclusion

- Proposed using quantum combs as an alternative to slicing for generalising circuit synthesis
- Introduced CombSynth, a synthesis algorithm for CNOT combs based on RowCol
- Tested CombSynth against RowCol with slicing and found it performed better on a wide variety of experimental parameters.

Future Work

Other Gate Sets



Future Work

Compare with literature

Investigate how synthesis with quantum combs relates to compilation methods that don't use slicing:

- ZX Circuit Extraction : Duncan, Kissinger, Perdrix and van de Wetering 2019
- Lazy Synthesis : Martiel and Goubault de Brugière 2020