Optimal Synthesis of Linear Reversible Circuits

Ketan Patel, Igor Markov, and John Hayes

University of Michigan
Electrical Engineering & Computer Science

Outline

- Motivation
- Background
- Lower Bound
- Synthesis Algorithm
- Application: [Quantum] Stabilizer Circuits
- Future work

Motivation

Reversible Computation

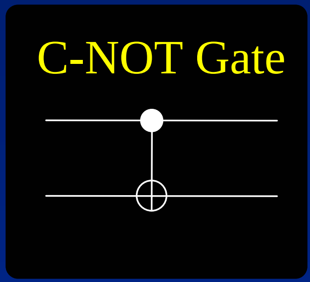
- "Energy-free" computation
- Reversible applications: cryptography, DSP, etc.

Quantum Computation

- Application to important quantum circuits: stabilizer circuits

C-NOT Gate

Input	Output
00	00
01	01
10	11
11	10



First input (**control**) passes through unchanged

Second input (**target**) inverted if control=1

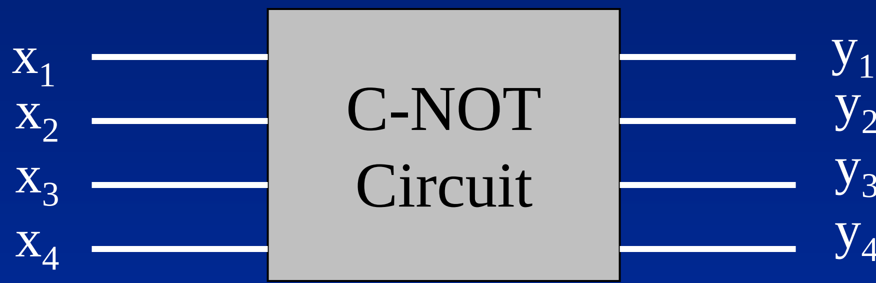
C-NOT is **linear** (under bitwise XOR operation ⊕)

i.e.

$$f(x_1 \oplus x_2) = f(x_1) \oplus f(x_2) \quad \text{for all} \quad x_1, x_2 \in \{0,1\}^n$$

C-NOT Circuits

Compute linear transformation over $\{0,1\}^n$



$|0000\rangle \rightarrow |0000\rangle$

Mapping of $|0001\rangle, |0010\rangle, |0100\rangle, |1000\rangle$ determine full mapping

Example:

$$f(|1100\rangle) = f(|1000\rangle) \oplus f(|0100\rangle)$$

Matrix Representation

Can use matrix representation

Example:

$$\left\{\begin{array}{l} |1\,0\,0\rangle \to |0\,0\,1\rangle \\ |0\,1\,0\rangle \to |0\,1\,1\rangle \\ |0\,0\,1\rangle \to |1\,0\,1\rangle \end{array}\right\} \Leftrightarrow \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \end{bmatrix}$$

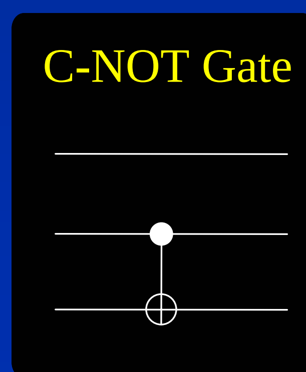
$$|1\,1\,0\rangle \to \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \end{bmatrix} \bullet \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix}$$

C-NOT action corresponds to multiplication by elementary matrix

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix} \Leftrightarrow$$

row (target) →

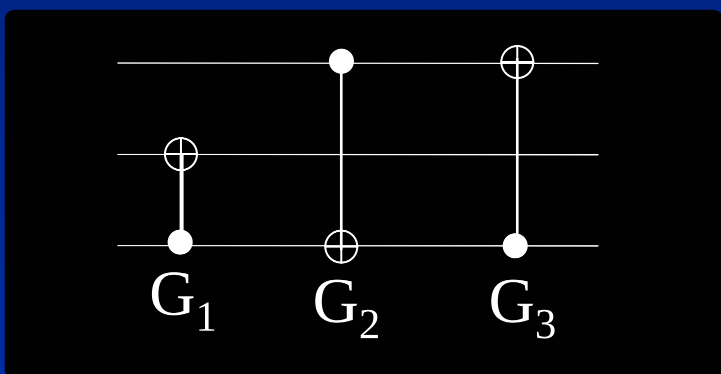
↑ column (control)



Matrix Representation (cont.)

Concatenation of C-NOT's corresponds to matrix multiplication

Example:



$$\Rightarrow \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \end{bmatrix} = \underbrace{\begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{G_3} \bullet \underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}}_{G_2} \bullet \underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix}}_{G_1}$$

Matrix row reduction $\Rightarrow$ decomposition into elementary matrices (recall Gaussian elimination)

C-NOT Synthesis

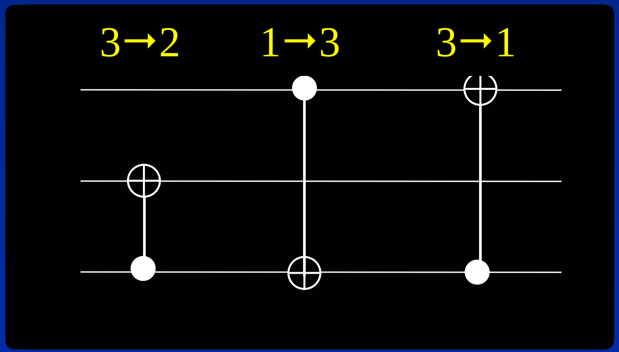
C-NOT Synthesis ⟺ binary matrix row reduction

# of gates = # of row operations

Example:

$$\begin{bmatrix}0&0&1\\0&1&1\\1&0&1\end{bmatrix} \overset{3\to1}{\Rightarrow} \begin{bmatrix}1&0&0\\0&1&1\\1&0&1\end{bmatrix} \overset{1\to3}{\Rightarrow} \begin{bmatrix}1&0&0\\0&1&1\\0&0&1\end{bmatrix} \overset{3\to2}{\Rightarrow} \begin{bmatrix}1&0&0\\0&1&0\\0&0&1\end{bmatrix}$$

$$\Rightarrow \begin{bmatrix}0&0&1\\0&1&1\\1&0&1\end{bmatrix} = \begin{bmatrix}1&0&1\\0&1&0\\0&0&1\end{bmatrix} \bullet \begin{bmatrix}1&0&0\\0&1&0\\1&0&1\end{bmatrix} \bullet \begin{bmatrix}1&0&0\\0&1&1\\0&0&1\end{bmatrix}$$



Gaussian Elimination Requires $O(n^2)$ row ops (gates)

Lower Bound

Number of *n*-wire linear reversible transformations

$$\prod_{i=0}^{n-1}(2^n - 2^i)$$

If all transformations require ≤ *d* gates,
then the number of *d*-gate circuits must be no smaller

$$\prod_{i=0}^{n-1}(2^n - 2^i) \le (n(n-1)+1)^d$$

$\Rightarrow$ $c\, n^2/\log n \le d$ for some $c>0$

Conclusion: need at least $c\, n^2/\log n$ gates

Our Synthesis Algorithm (intuition)

$$n \left\{ \begin{bmatrix} 1 & 0 & \cdots \\ 1 & 1 & \cdots \\ 0 & 1 & \cdots \\ 1 & 0 & \cdots \\ 0 & 1 & \cdots \\ \vdots & \vdots & \vdots \\ 1 & 0 & \cdots \\ 1 & 1 & \cdots \\ 0 & 0 & \cdots \\ 0 & 1 & \cdots \end{bmatrix} \right.$$

Assume n is large

- Use ≤ n row ops to eliminate duplicate sub-rows
- Leave no more than 3 non-zero sub-rows

  relatively few operations necessary to clear

Reduces number of ops by factor ~2

Synthesis Algorithm

- Group columns into sections of size $\leq \alpha \log n$ with $\alpha < 1$
- Eliminate duplicate sub-rows in each column section

  # row ops = $O(n) \bullet$ # sections = $O(n^2/\log n)$
- Place 1s on the diagonal

  # row ops = $O(n)$
- Eliminate (relatively few) remaining off-diagonal 1s

  # row ops = $O(2^{\alpha \log n}) \bullet$ # sections

  $= O(n^{\alpha}) \bullet n/(\alpha \log n) = O(n^{1+\alpha}/\log n)$

Total # of row operations: $O(n^2/\log n)$

Execution Time

Execution time dominated by row ops

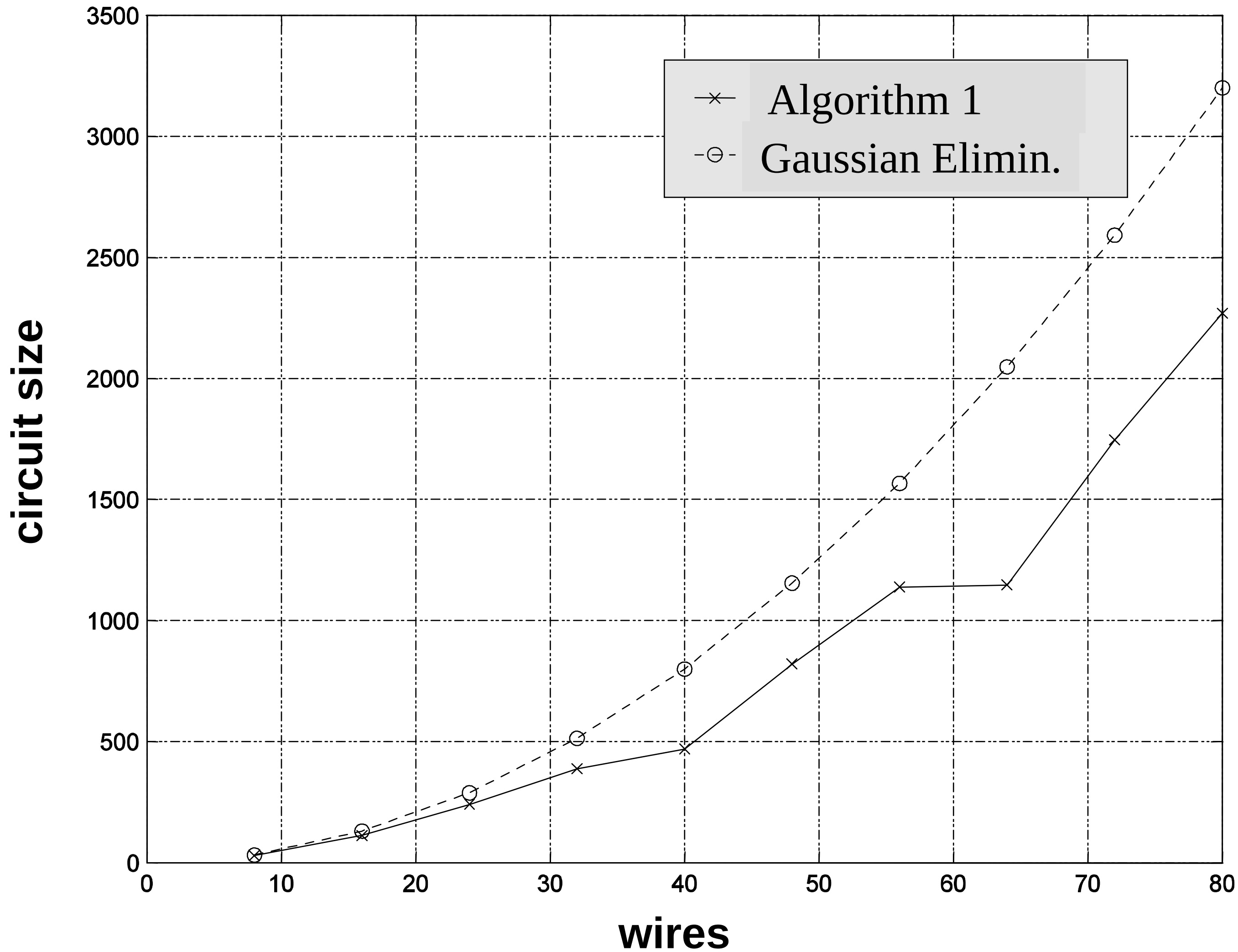
$\Rightarrow$ $O(n^3/\log n)$

Execution time for Gaussian Elimination

$\Rightarrow$ $O(n^3)$

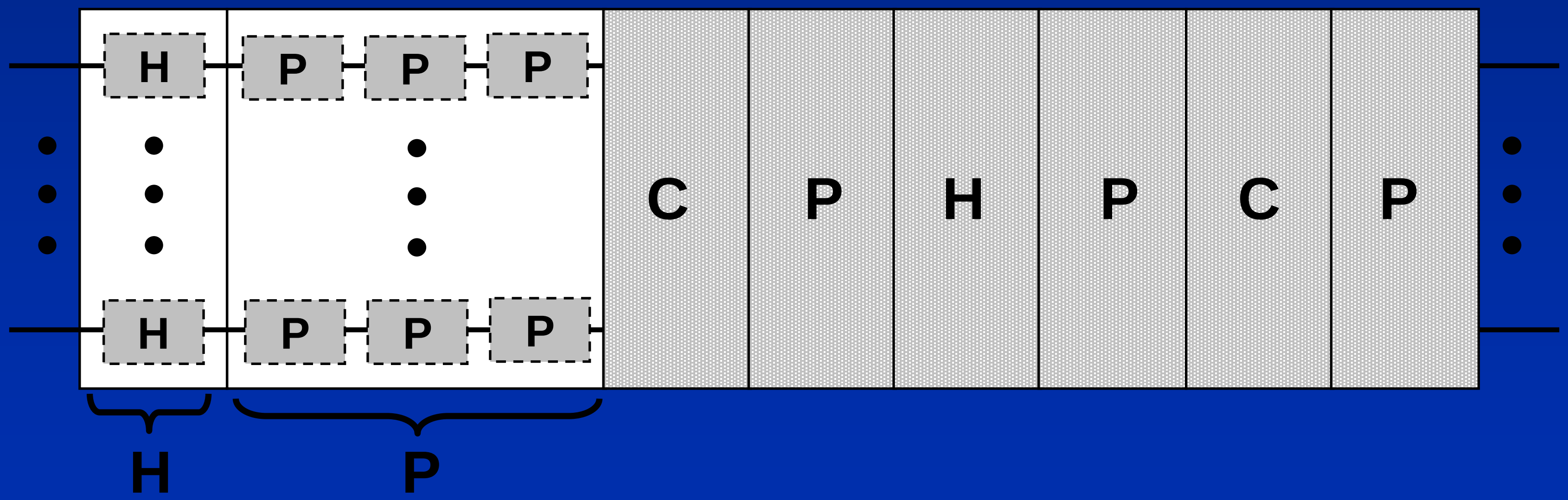
...but how soon do the asymptotics kick in?

Empirical Results



Application: Stabilizer Circuits

- Important class of quantum circuits
- Composed of Hadamard, Phase, and C-NOT gates
- Can use blocks of H, P & C gates to synthesize (Aaronson & Gottesman)



Circuit size dominated by size of C-blocks

Related Topics

- Optimal column partitioning
- Synthesis of general reversible circuits
  - T-C-N-T decomposition
- Applications to synthesis of quantum circuits
  - E.g., via the Gottesman-Knill theorem
- Reducing circuit depth rather than size

Thank you