Symbolic synthesis of Clifford circuits and beyond...

Matt Amy\textsuperscript{1}, Owen Bennett-Gibbs\textsuperscript{2}, Neil Julien Ross\textsuperscript{3}

\textsuperscript{1}Simon Fraser University
\textsuperscript{2}McGill University
\textsuperscript{2}Dalhousie University

Quantum Physics and Logic
Oxford, June 30, 2022
Extracting circuits from things

Circuit extraction from ZX/ZH diagrams (without gflow) is hard!

---

Extracting circuits from things

Circuit extraction from ZX/ZH diagrams (without gflow) is hard!

What if we extract from the sum-over-paths instead?

Kissinger & van de Wetering, Reducing T-count with the ZX-calculus.
The sum-over-paths representation

A path sum is a symbolic expression of a linear operator \( \Psi : \mathbb{C}^{2^m} \rightarrow \mathbb{C}^{2^n} \) as a sum indexed by binary variables:

\[
\Psi |x\rangle = |\Psi(x)\rangle = \mathcal{N} \sum_{y \in \mathbb{Z}_2^k} e^{2\pi i P(x,y)} f(x, y) |f(x, y)\rangle,
\]
A **path sum** is a symbolic expression of a linear operator \( \Psi : \mathbb{C}^{2^m} \rightarrow \mathbb{C}^{2^n} \) as a sum indexed by binary variables:

\[
\Psi |x\rangle = |\Psi(x)\rangle = \mathcal{N} \sum_{y \in \mathbb{Z}_2^k} e^{2\pi i P(x,y)} |f(x, y)\rangle,
\]

\( \mathcal{N} \in \mathbb{C} \setminus \{0\} \) is a normalization factor,
A path sum is a symbolic expression of a linear operator \( \Psi : \mathbb{C}^{2^m} \to \mathbb{C}^{2^n} \) as a sum indexed by binary variables:

\[
\Psi |x\rangle = |\Psi(x)\rangle = \mathcal{N} \sum_{y \in \mathbb{Z}_2^k} e^{2\pi i P(x, y)} |f(x, y)\rangle,
\]

- \( \mathcal{N} \in \mathbb{C} \setminus \{0\} \) is a normalization factor,
- \( P : \mathbb{Z}_2^m \times \mathbb{Z}_2^k \to \mathbb{R} \) is a real-valued multilinear polynomial, and
The sum-over-paths representation

A **path sum** is a symbolic expression of a linear operator \( \Psi : \mathbb{C}^{2^m} \to \mathbb{C}^{2^n} \) as a sum indexed by binary variables:

\[
\Psi |x\rangle = |\Psi(x)\rangle = \mathcal{N} \sum_{y \in \mathbb{Z}_2^k} e^{2\pi i P(x, y)} |f(x, y)\rangle,
\]

- \( \mathcal{N} \in \mathbb{C} \setminus \{0\} \) is a normalization factor,
- \( P : \mathbb{Z}_2^m \times \mathbb{Z}_2^k \to \mathbb{R} \) is a real-valued multilinear polynomial, and
- \( f : \mathbb{Z}_2^m \times \mathbb{Z}_2^k \to \mathbb{Z}_2^n \) is system of Boolean-valued multilinear polynomials.
A **path sum** is a symbolic expression of a linear operator $\Psi : \mathbb{C}^{2^m} \rightarrow \mathbb{C}^{2^n}$ as a sum indexed by binary variables:

$$
\Psi |x\rangle = |\Psi(x)\rangle = \mathcal{N} \sum_{y \in \mathbb{Z}_2^k} e^{2\pi i P(x,y)} |f(x, y)\rangle,
$$

- $\mathcal{N} \in \mathbb{C} \setminus \{0\}$ is a normalization factor,
- $P : \mathbb{Z}_2^m \times \mathbb{Z}_2^k \rightarrow \mathbb{R}$ is a real-valued multilinear polynomial, and
- $f : \mathbb{Z}_2^m \times \mathbb{Z}_2^k \rightarrow \mathbb{Z}_2^n$ is a system of Boolean-valued multilinear polynomials.
Examples

Phase & reversible gates:

\[ S|x\rangle = i^x|x\rangle, \quad T|x\rangle = \omega^x|x\rangle \text{ where } \omega = e^{2\pi i/8} \]

\[ CNOT|x\rangle|y\rangle = |x\rangle|x \oplus y\rangle \]
Examples

Phase & reversible gates:

\[ S|x\rangle = i^x |x\rangle, \quad T|x\rangle = \omega^x |x\rangle \text{ where } \omega = e^{2\pi i/8} \]

\[ CNOT|x\rangle|y\rangle = |x\rangle|x \oplus y\rangle \]

Branching gates:

\[ H|x\rangle = \frac{1}{\sqrt{2}} \sum_y (-1)^{xy} |y\rangle \]
Examples

Phase & reversible gates:

\[ S|x\rangle = i^x|x\rangle, \quad T|x\rangle = \omega^x|x\rangle \] where \( \omega = e^{2\pi i/8} \)

\[ CNOT|x\rangle|y\rangle = |x\rangle|x \oplus y\rangle \]

Branching gates:

\[ H|x\rangle = \frac{1}{\sqrt{2}} \sum_y (-1)^{xy}|y\rangle \]

Cups & caps:

\[ \subset = \sum_y |y\rangle|y\rangle, \quad \supset |x_1\rangle|x_2\rangle = \frac{1}{2} \sum_y (-1)^{y(x_1+x_2)} \]
Examples

Phase & reversible gates:

\[ S|x\rangle = i^x|x\rangle, \quad T|x\rangle = \omega^x|x\rangle \text{ where } \omega = e^{2\pi i / 8} \]

\[ CNOT|x\rangle|y\rangle = |x\rangle|x \oplus y\rangle \]

Branching gates:

\[ H|x\rangle = \frac{1}{\sqrt{2}} \sum_y (-1)^{xy} |y\rangle \]

Cups & caps:

\[ \subset = \sum_y |y\rangle|y\rangle, \quad \supset |x_1\rangle|x_2\rangle = \frac{1}{2} \sum_y (-1)^{y(x_1+x_2)} \]

Gate composition:

\[ TH|x\rangle = \frac{1}{\sqrt{2}} \sum_y (-1)^{xy} (T|y\rangle) = \frac{1}{\sqrt{2}} \sum_y (-1)^{xy} \omega^y |y\rangle \]
Recent work on formalizing the sum-over-paths:

- Re-writing system and compositional model for circuits\(^1\)
- Connections to graphical calculi\(^2,3\) \(ZH \equiv SOP\)
- Dagger compact structure\(^3\)
- Complete re-write rules for \(\zeta^k_2\) phases\(^4\)

---

\(^1\) Amy, Towards large-scale functional verification of universal quantum circuits. QPL 2018


\(^3\) Vilmart, The Structure of Sum-Over-Paths, its Consequences, and Completeness for Clifford. FoSSaCs 2021.

Equational reasoning

Let $\Psi$ be a path sum, $f$ a Boolean polynomial, and assume $y \notin FV(\Psi)$ and $x, y \notin FV(f)$. Then the following equations hold.

\[
\sum_y |\Psi\rangle = 2|\Psi\rangle \quad \text{[E]}
\]

\[
\sum_{x,y} (-1)^{y(x+f)}|\Psi(x)\rangle = 2|\Psi(f)\rangle \quad \text{[I]}
\]

\[
\sum_y i^y (-1)^{yf} |\Psi\rangle = \omega\sqrt{2}(-i)^f |\Psi\rangle \quad \text{[U]}
\]

\[
\sum_y |\Psi(y)\rangle = \sum_y |\Psi(y+f)\rangle \quad \text{[V]}
\]

**Proposition**

*The [E], [I], and [U] rules are complete for Stabilizer operations*
Simplifications in SOP-land

The original **phase polynomial optimization**\(^5\) extracted a circuit from the simplified sum-over-paths

---

Simplifications in SOP-land

The original phase polynomial optimization\(^5\) extracted a circuit from the simplified sum-over-paths

More sophisticated re-writing can go further, but like in ZX-land circuit extraction gets harder!

\[ |x_1 x_2 x_3 \rangle \mapsto |x_1 (x_2 \oplus x_1 \cdot x_3) x_3 \rangle \]

Synthesis of the sum-over-paths

The synthesis problem

*Given a promise that a path sum \( |\psi\rangle \) represents a unitary transformation, synthesize/extract a circuit implementing \( \psi \) over some gate set \( G \).*
Synthesis of the sum-over-paths

The synthesis problem

Given a promise that a path sum $|\Psi\rangle$ represents a unitary transformation, synthesize/extract a circuit implementing $\Psi$ over some gate set $G$

In this work:

▶ Hardness of checking the unitarity condition
▶ Synthesis of Clifford path sums
▶ Synthesis of general path sums
The UNITARY problem
The UNITARY problem

Given a sum-over-paths, does it represent a unitary transformation?
The UNITARY problem

Given a sum-over-paths, does it represent a unitary transformation?

Theorem

The UNITARY problem is coNP-hard.
The UNITARY problem

Given a sum-over-paths, does it represent a unitary transformation?

Theorem

The UNITARY problem is coNP-hard.

Proof sketch:
Reduce TAUT \( \{ = \text{propositional tautologies}\} \) to UNITARY by constructing, for a propositional formula \( \varphi \), a path sum

\[
|x\rangle \mapsto \varphi(x)|x\rangle
\]
The Tseytin transformation

For a Boolean *polynomial* $p$, there exists a direct encoding

$$p(x) = 2^{-1} \sum_{y \in \mathbb{Z}_2} (-1)^y (1+p(x))$$

However, writing a propositional formula $\varphi$ as a polynomial, denoted $\overline{\varphi}$, may use exponential overhead.

We can instead use the **Tseytin transformation** to write $\varphi$ as an equi-satisfiable conjunction of constant-size clauses

$$\mathcal{T}(x_1 \land (x_2 \lor (\neg x_3))) = (z_1 \leftrightarrow \neg x_3) \land (z_2 \leftrightarrow (x_2 \lor z_1)) \land (z_3 \leftrightarrow (x_1 \land z_2)) \land z_3$$
A polynomial size encoding

Given $T(\varphi) = \land_i (z_i \leftrightarrow \varphi_i)$ encode $T(\varphi)$ inductively as:

- $\Psi_{z\leftrightarrow\varphi} = 2^{-1} \sum_{y \in \mathbb{Z}_2} (-1)^y (z + \overline{\varphi})$
- $\Psi_{c_1 \land c_2} = \Psi_{c_1} \cdot \Psi_{c_2}$

By globally summing over all $z_i$ we obtain a polynomial-size encoding over only the free variables of $\varphi$:

$$\varphi(x) = 2^{-(k+1)} \sum_{y} \sum_{y \in \mathbb{Z}_2^k} \sum_{z \in \mathbb{Z}_2^k} (-1)^y (1+z_1+\sum_i y_i(z_i+\overline{\varphi_i}(x)))$$

Corollary The UNITARY problem is coNP-hard
A polynomial size encoding

Given \( T(\varphi) = \land_i(z_i \leftrightarrow \varphi_i) \) encode \( T(\varphi) \) inductively as:

\[
\begin{align*}
\Psi_{z \leftrightarrow \varphi} &= 2^{-1} \sum_{y \in \mathbb{Z}_2} (-1)^y (z + \overline{\varphi}) \\
\Psi_{c_1 \land c_2} &= \Psi_{c_1} \cdot \Psi_{c_2}
\end{align*}
\]

By globally summing over all \( z_i \) we obtain a polynomial-size encoding over only the free variables of \( \varphi \):

\[
\varphi(x) = 2^{-(k+1)} \sum_{y} \sum_{y \in \mathbb{Z}_2^k} \sum_{z \in \mathbb{Z}_2^k} (-1)^y (1 + z_1 + \sum_i y_i (z_i + \overline{\varphi}_i(x)))
\]

Corollary

*The UNITARY problem is coNP-hard*
Synthesizing Clifford circuits
Clifford path sums

\[ H : |x\rangle \mapsto \sqrt{2}^{-1} \sum_y (-1)^{xy} |y\rangle \]
\[ S : |x\rangle \mapsto i^x |x\rangle \]
\[ CZ : |x\rangle |y\rangle \mapsto (-1)^{xy} |x\rangle |y\rangle \]

Path sums over Clifford gates have the form

\[ |x\rangle \mapsto \frac{1}{\sqrt{2^m}} \sum_{y \in \mathbb{Z}_2^m} i^{L(x,y)} (-1)^{Q(x,y)} |f(x, y)\rangle \]

where

- \( L \) is linear,
- \( Q \) is pure quadratic, and
- \( f \) is affine.
Proposition (Affine normal form)

Any Clifford path sum can be re-written up to a permutation as

$$|x⟩ \mapsto \frac{ω^I}{\sqrt{2^k}} \sum_{y \in \mathbb{Z}_2^k} i^{L(x,y)} (-1)^{Q(x,y)} |y⟩ \otimes |f(x, y)⟩$$

in polynomial time using the equations [E], [I], and [U].

Works by eliminating variables from the sum until a minimal spanning set for the affine subspace is obtained.
Proposition (Affine normal form)

Any Clifford path sum can be re-written up to a permutation as

$$|x\rangle \mapsto \frac{\omega^I}{\sqrt{2^k}} \sum_{y \in \mathbb{Z}_2^k} i^{L(x,y)}(-1)^{Q(x,y)}|y\rangle \otimes |f(x, y)\rangle$$

in polynomial time using the equations [E], [I], and [U].

Works by eliminating variables from the sum until a minimal spanning set for the affine subspace is obtained.

Note: can be made unique with a tweak from Tommy & Miriam’s work presented earlier this week.
Decomposition into linear operators

Decomposing $L$, $Q$, and $f$ into functions on inputs $x$, affine basis variables $y$, and $x$-$y$ cross terms, the affine normal form factors into the following sequence of operators:

$$
|x\rangle \mapsto \omega^i i^{L_x(x)} (-1)^{Q_x(x)} |x\rangle \quad \{S, CZ\}
$$

$$
|x\rangle \mapsto |R(x)\rangle |f_x(x)\rangle \quad \{\text{CNOT}\}
$$

$$
|R(x)\rangle |f_x(x)\rangle \mapsto \frac{1}{\sqrt{2^k}} \sum_{y \in \mathbb{Z}_2^k} (-1)^{\sum_i y_i R_i(x)} |y\rangle |f_x(x)\rangle \quad \{H\}
$$

$$
|y\rangle |f_x(x)\rangle \mapsto |y\rangle |f_x(x) + f_y(y) + b\rangle \quad \{X, \text{CNOT}\}
$$

$$
|y\rangle |f(x, y)\rangle \mapsto i^{L_y(y)} (-1)^{Q_y(y)} |y\rangle |f(x, y)\rangle \quad \{S, CZ\}
$$
A simple, constructive proof of the Bruhat decomposition

Any Clifford operator can be written in a 9 stage circuit

\[ S \cdot CZ \cdot X \cdot \text{CNOT} \cdot H \cdot \text{CNOT} \cdot X \cdot CZ \cdot S \]

\[ 6^{\text{Maslov, Roetteler, Shorter stabilizer circuits via Bruhat decomposition and quantum circuit transformations. IEEE TIT 2018.}} \]
A simple, constructive proof of the Bruhat decomposition

Any Clifford operator can be written in a 9 stage circuit

\[ S \cdot CZ \cdot X \cdot CNOT \cdot H \cdot CNOT \cdot X \cdot CZ \cdot S \]

Theorem

Any Clifford operator can be synthesized in polynomial time over \{CNOT, X, CZ, S, H\} as an 8 stage circuit of the form

\[ S \cdot CZ \cdot CNOT \cdot H \cdot CNOT \cdot X \cdot CZ \cdot S \]

\[ S^L_x(x) \quad \cdots \quad (-1)^Q_x(x) \quad U \quad H \quad \cdots \quad S^L_y(y) \]

Corollary

An affine normal form

\[ |x⟩ \mapsto \frac{ω^l}{\sqrt{2^k}} \sum_{y \in \mathbb{Z}_2^k} i^{L(x,y)} (-1)^{Q(x,y)} |y⟩ \otimes |f(x, y)⟩ \]

can be implemented with Clifford gates and ancillas initialized in the \(|0⟩\) state if and only if

\[ \text{rank}(\{ R_i \} \cup \{(f_x)_i\}) = n \]
Synthesizing general circuits
Synthesizing more general circuits

Can we synthesize non-Clifford operators?
Can we synthesize non-Clifford operators?

By inverting the sum-over-paths, we can view gates as reduction rules, e.g.,

\[ \Lambda_k(X) : |x\rangle|y \oplus \prod_i x_i \rangle \mapsto |x\rangle|y\rangle \]

\[ \Lambda_k(\theta) : e^{2\pi i \theta} \prod_i x_i |x\rangle \mapsto |x\rangle \]

\[ H : \frac{1}{\sqrt{2}} \sum_{x' \in \mathbb{Z}_2} (-1)^{xx'} |x'\rangle \mapsto |x\rangle \]

and synthesize by reducing to the identity!
Example: QFT\textsubscript{3} derivation

Derive the circuit by applying re-write rules!

\[ QFT_3 |x_1x_2x_3\rangle = \frac{1}{\sqrt{2^3}} \sum_{y_1,y_2,y_3} \omega^{x_3y_3} j^{x_3y_2+x_2y_3} (-1)^{x_3y_1+x_2y_2+x_1y_3} |y_1y_2y_3\rangle \]
Example: QFT\(_3\) derivation

Derive the circuit by applying re-write rules!

\[
QFT_3 |x_1x_2x_3\rangle = \frac{1}{\sqrt{2^3}} \sum_{y_1,y_2,y_3} \omega^{x_3y_3} i^{x_3y_2 + x_2y_3} (-1)^{x_3y_1 + x_2y_2 + x_1y_3} |y_1y_2y_3\rangle
\]

\[
\underbrace{H_1}_{\frac{1}{\sqrt{2^2}}} \rightarrow \frac{1}{\sqrt{2^2}} \sum_{y_1,y_2} \omega^{x_3y_3} i^{x_3y_2 + x_2y_3} (-1)^{x_2y_2 + x_1y_3} |x_3y_2y_3\rangle
\]
Example: QFT$_3$ derivation

Derive the circuit by applying re-write rules!

$$QFT_3 |x_1 x_2 x_3\rangle = \frac{1}{\sqrt{2^3}} \sum_{y_1, y_2, y_3} \omega^{x_3 y_3} i^{x_3 y_2 + x_2 y_3} (-1)^{x_3 y_1 + x_2 y_2 + x_1 y_3} |y_1 y_2 y_3\rangle$$

$$H_1 \xrightarrow{1} \frac{1}{\sqrt{2^2}} \sum_{y_1, y_2} \omega^{x_3 y_3} i^{x_3 y_2 + x_2 y_3} (-1)^{x_2 y_2 + x_1 y_3} |x_3 y_2 y_3\rangle$$

$$cS^\dagger_{1,2} cT^\dagger_{1,3} \xrightarrow{1} \frac{1}{\sqrt{2^2}} \sum_{y_2, y_3} i^{x_2 y_3} (-1)^{x_2 y_2 + x_1 y_3} |x_3 y_2 y_3\rangle$$
Example: QFT$_3$ derivation

Derive the circuit by applying re-write rules!

$$QFT_3 |x_1x_2x_3⟩ = \frac{1}{\sqrt{2^3}} \sum_{y_1,y_2,y_3} \omega^{x_3y_3} j^{x_3y_2 + x_2y_3} (-1)^{x_3y_1 + x_2y_2 + x_1y_3} |y_1y_2y_3⟩$$

$$H_1 \rightarrow \frac{1}{\sqrt{2^2}} \sum_{y_1,y_2} \omega^{x_3y_3} j^{x_3y_2 + x_2y_3} (-1)^{x_2y_2 + x_1y_3} |x_3y_2y_3⟩$$

$$cS_{1,2} cT_{1,3} \rightarrow \frac{1}{\sqrt{2^2}} \sum_{y_2,y_3} j^{x_2y_3} (-1)^{x_2y_2 + x_1y_3} |x_3y_2y_3⟩$$

$$H_2 \rightarrow \frac{1}{\sqrt{2}} \sum_{y_3} j^{x_2y_3} (-1)^{x_1y_3} |x_3x_2y_3⟩$$
Example: $\text{QFT}_3$ derivation

Derive the circuit by applying re-write rules!

$$QFT_3 |x_1 x_2 x_3\rangle = \frac{1}{\sqrt{2^3}} \sum_{y_1, y_2, y_3} \omega^{x_3 y_3} i^{x_3 y_2 + x_2 y_3} (-1)^{x_3 y_1 + x_2 y_2 + x_1 y_3} |y_1 y_2 y_3\rangle$$

$H_1 \rightarrow \frac{1}{\sqrt{2^2}} \sum_{y_1, y_2} \omega^{x_3 y_3} i^{x_3 y_2 + x_2 y_3} (-1)^{x_2 y_2 + x_1 y_3} |x_3 y_2 y_3\rangle$

$cS_{1,2}^\dagger cT_{1,3}^\dagger \rightarrow \frac{1}{\sqrt{2^2}} \sum_{y_2, y_3} i^{x_2 y_3} (-1)^{x_2 y_2 + x_1 y_3} |x_3 y_2 y_3\rangle$

$H_2 \rightarrow \frac{1}{\sqrt{2}} \sum_{y_3} i^{x_2 y_3} (-1)^{x_1 y_3} |x_3 x_2 y_3\rangle$

$cS_{2,3}^\dagger \rightarrow \frac{1}{\sqrt{2}} \sum_{y_3} (-1)^{x_1 y_3} |x_3 x_2 y_3\rangle$
Example: \( \text{QFT}_3 \) derivation

Derive the circuit by applying re-write rules!

\[
\text{QFT}_3 |x_1 x_2 x_3 \rangle = \frac{1}{\sqrt{2^3}} \sum_{y_1, y_2, y_3} \omega^{x_3 y_3} i^{x_3 y_2 + x_2 y_3} (-1)^{x_3 y_1 + x_2 y_2 + x_1 y_3} |y_1 y_2 y_3 \rangle
\]

\[
H_1 \rightarrow \frac{1}{\sqrt{2^2}} \sum_{y_1, y_2} \omega^{x_3 y_3} i^{x_3 y_2 + x_2 y_3} (-1)^{x_2 y_2 + x_1 y_3} |x_3 y_2 y_3 \rangle
\]

\[
cS_{1,2}^{\dagger} cT_{1,3}^{\dagger} \rightarrow \frac{1}{\sqrt{2^2}} \sum_{y_2, y_3} i^{x_2 y_3} (-1)^{x_2 y_2 + x_1 y_3} |x_3 y_2 y_3 \rangle
\]

\[
H_2 \rightarrow \frac{1}{\sqrt{2}} \sum_{y_3} i^{x_2 y_3} (-1)^{x_1 y_3} |x_3 x_2 y_3 \rangle
\]

\[
cS_{2,3}^{\dagger} \rightarrow \frac{1}{\sqrt{2}} \sum_{y_3} (-1)^{x_1 y_3} |x_3 x_2 y_3 \rangle
\]

\[
H_3 \rightarrow |x_3 x_2 x_1 \rangle
\]
YEAH, WELL, THAT'S JUST LIKE ONE CIRCUIT, MAN
A **generalized permutation** is a permutation matrix times a (unitary) diagonal matrix.

**Proposition**

*Any unitary $U$ can be written as a series of alternating stages of $H$ gates and generalized permutations $G$*

$$U = G_1 H_1 G_2 H_2 G_3 \cdots H_n G_n$$

---

7 Kliuchnikov, Maslov, Mosca, *Fast and efficient exact synthesis of single qubit unitaries generated by Clifford and $T$ gates*. QIC 2013.

Hadamards and permutations

A **generalized permutation** is a permutation matrix times a (unitary) diagonal matrix.

**Proposition**

*Any unitary $U$ can be written as a series of alternating stages of $H$ gates and generalized permutations $G$*

$$U = G_1 H_1 G_2 H_2 G_3 \cdots H_n G_n$$

The **Number-Theoretic™** method\(^7\)\(^8\) synthesizes unitary matrices by applying a generalized permutation so it can be **reduced** by a Hadamard gate

\(^7\)Kliuchnikov, Maslov, Mosca, *Fast and efficient exact synthesis of single qubit unitaries generated by Clifford and T gates*. QIC 2013.

Symbolic exact synthesis

Recall: A path sum $\Psi$ is (Hadamard) **reducible** if we can apply the rule

$$H : \frac{1}{\sqrt{2}} \sum_{x' \in \mathbb{Z}_2} (-1)^{xx'} |x'\rangle \mapsto |x\rangle$$
Symbolic exact synthesis

Recall: A path sum $\Psi$ is (Hadamard) **reducible** if we can apply the rule

$$H : \frac{1}{\sqrt{2}} \sum_{x' \in \mathbb{Z}_2} (-1)^{xx'} |x'\rangle \mapsto |x\rangle$$

Algorithm:

1. Simplify $|\Psi\rangle$ using $[E]$, $[I]$, $[U]$
2. If $\exists$ generalized permutation $G$ s.t. $G^\dagger |\Psi\rangle$ is reducible,
   2.1 $|\Psi\rangle \leftarrow (H \otimes I_{n-1}) G^\dagger |\Psi\rangle$
   2.2 Go to 1.
3. If path variables remain or $\Psi$ is non-unitary, fail

*How do we find $G$? Does there always exist such a $G$?*
A heuristic for generalized permutations

1. Apply affine simplifications to the output state $|f(x, y)\rangle$
2. Apply non-linear simplifications to the phase $e^{2\pi iP(x, y)}$
3. Apply non-linear simplifications to the output state $|f'(x, y)\rangle$
4. Apply non-linear simplifications to the phase $e^{2\pi iP'(x, y)}$
5. **If the path sum is still irreducible, find simultaneous variable substitutions to make it reducible**
   - Heuristic: degree reduction

See the paper for more details!
Re-synthesizing random Clifford and Clifford+$T$ circuits:

<table>
<thead>
<tr>
<th>$n$</th>
<th># gates</th>
<th># circuits</th>
<th>avg. time (s)</th>
<th>avg. change</th>
<th>success</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>500</td>
<td>1000</td>
<td>0.137</td>
<td>+19.2%</td>
<td>–</td>
</tr>
<tr>
<td>20</td>
<td>1000</td>
<td>1000</td>
<td>0.481</td>
<td>-12.9%</td>
<td>–</td>
</tr>
<tr>
<td>50</td>
<td>500</td>
<td>1000</td>
<td>0.264</td>
<td>+90.7%</td>
<td>–</td>
</tr>
<tr>
<td>50</td>
<td>1000</td>
<td>1000</td>
<td>1.518</td>
<td>+129.1%</td>
<td>–</td>
</tr>
</tbody>
</table>

Clifford+$T$ heuristic fails more as gate density increases
Decompiling from a low-level gate set (e.g. Clifford+$T$) to $H+$ generalized permutations often reveals high-level structure!
QFT, synthesized

Compiled with Feynman: https://github.com/meamy/feynman
In this talk...

- Unitarity testing is coNP-hard
- Normal forms & extraction of an 8-stage Clifford circuit
- Partial heuristic for general circuit extraction

Future work

- Use completions of path sum re-writing to prove completeness of our synthesis framework
- Come up with complete procedure for finding a reducing generalized permutation
- Explore use in peephole re-synthesis
Thank you!