COL861 Synthesis papers report

Mohd Sayanur Rahman, 2021PH10221

April 21, 2025

Paper 1: Representation of Quantum Circuits with Clifford and $\pi/8$ Gates

Topic

Matsumoto and Amano investigate the structural representation of one-qubit circuits over the standard basis $\{H, P, T\}$, where

- $H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$,
- $T = \mathrm{diag}(1, e^{i\pi/4})$,
- $P = T^2 = \mathrm{diag}(1, i)$.

This Clifford+T gate set is universal for single-qubit unitaries and underlies fault-tolerant design. The authors propose a canonical "normal form" for circuits that (1) standardizes representation, (2) makes equivalence checking trivial, and (3) quantifies the exponential growth of representable unitaries as a function of T-count.

Main Contributions

1. **Normal Form Definition:** Every one-qubit circuit can be uniquely written as

$$W_n \, T \, W_{n-1} \, T \, \cdots \, T \, W_1 \, T \, W_0,$$

   where W_0 is any Clifford element ($|C_1| = 192$), $W_n \in \{I, H, PH\}$, and $W_i \in \{H, PH\}$ for $1 \leq i < n$.

2. **Transformation Algorithm:** A linear-time, syntactic rewriting procedure converts any H/T sequence into its normal form by commuting T-gates and normalizing Clifford segments.

3. **Uniqueness Theorem:** Two circuits in normal form compute the same unitary if and only if they are syntactically identical, reducing equivalence checking to string comparison.

4. **Exact Enumeration:** They derive the closed-form count of distinct unitaries with at most n T-gates:

$$N(n) = 192\big(3 \cdot 2^n - 2\big),$$

proving exponential expressivity in n.

Results

- **Algorithmic Cost:** Normal-form conversion runs in $O(L)$ time for input length L, with each rewrite constant-time.

- **Equivalence Checking:** Circuit equivalence drops from $O(n^3)$ matrix multiplications to $O(n)$ string compare.

- **Unitary Counts:** For at most n T-gates,

$$\begin{aligned} N(0) &= 192, \\ N(1) &= 192 \times 4 = 768, \\ N(2) &= 192 \times 10 = 1920, \\ &\text{asymptotically } N(n) \approx 576\, 2^n. \end{aligned}$$

- **Expressivity Growth:** Each additional T multiplies the representable space by roughly 2.

- **Classical Simulability Boundary:** Reinforces that Clifford-only circuits (192 operations) are classically efficiently simulable (Gottesman–Knill Theorem), but T-gates break efficient simulability.

- **Compiler Impact:** Enables fast one-qubit compiler passes for synthesis, simplification, and fingerprinting.

Conclusion

Matsumoto and Amano's normal form provides a unified framework for efficient synthesis, equivalence testing, and enumeration of one-qubit Clifford+T circuits. By encoding operations into a fixed pattern, core compiler tasks reduce to lightweight string operations. Quantitatively, they show that the Clifford group alone is finite (192 elements), while each T-gate injects an exponential increase in expressible unitaries. Though scoped to one qubit, the techniques suggest pathways to multi-qubit normal forms (incorporating CNOT) and resource-aware compiler designs.

+1: On the Power of the T-Gate

The assertion "the T-gate is the root of the power of quantum computing" is evident by:

1. *Gottesman–Knill Theorem:* Circuits using only Clifford gates are classically efficiently simulable. The Clifford group C_1 has fixed size 192 for one qubit (Sec 1).

2. *Universality via T:* Citing Boykin *et al.* (FOCS '99), adding T to $\{H, CNOT\}$ yields universal quantum gates set, as the irrational phase $e^{i\pi/4}$ enables arbitrary-angle approximations in SU(2) (Sec 1). Citing Solovay-Kitaev Theorem, polynomial size quantum circuits over this standard set ($\{H, T, CNOT\}$) can solve all the problems in BQP, where BQP is the class of problems that can be solved efficiently by quantum computers.

3. *Combinatorial Enumeration:* The closed-form $N(n) = 192(3 \cdot 2^n - 2)$ (Sec 5) shows each T-gate doubles the new operations, directly quantifying T's expressive power.

4. *Normal-Form Uniqueness:* Only T-involving patterns generate novel unitaries beyond those in the finite Clifford set, proving T is essential for irreducible circuit diversity.

These rigorous structural and combinatorial insights mathematically confirm T-gates as vital for transcending classical simulability and achieving universal quantum advantage. The above facts do support opinion that " It may be natural to expect that the research on the effect of the T-gate may lead to better understanding of why a quantum computer can efficiently compute some hard problems. "

Paper 2: Optimal Two-Qubit Circuits for Universal Fault-Tolerant Quantum Computation

Topic

Glaudell, Ross, and Taylor address exact synthesis of two-qubit unitaries over the Clifford+CS gate set, where CS is controlled phase gate

$$\mathrm{CS} = \mathrm{diag}(1, 1, 1, i).$$

CS gates enable universality alongside easy fault-tolerant implementation. They seek circuits with the *minimum CS-count*, as CS is the dominant resource cost. The work combines algebraic, geometric, and automata techniques to derive a deterministic, optimal synthesis algorithm and to analyze resource lower bounds.

Main Contributions

1. **Exact Synthesis Algorithm:** A linear-time (in CS-count) method that inputs any $U \in G$ and outputs a CS-optimal circuit.

2. **Normal Form via $R(P, Q)$:** They define $R(P, Q) = \exp\big[i\frac{\pi}{2}(I - P)(I - Q)/4\big]$ for distinct commuting hermitian Pauli operators P, Q, generalizing CS as $R(Z \otimes I, I \otimes Z) = CS$ (Definition 2.1). Every element of G (group generated by $\{H, S, CZ, CS\}$) decomposes uniquely as a sequence of these R-gates followed by a Clifford (proposition 2.5).

3. **Automaton Characterization:** Construct a finite automaton whose accepted language exactly matches normal-form sequences, enabling combinatorial analysis.

4. **Lower-Bound via Volume Counting:** By comparing the growth rate of realizable circuits to the Haar volume of SU(4), they derive a worst-case lower bound of

   $$5 \log_2(1/\epsilon) + O(1)$$

   CS gates for an ϵ-approximation, matching the scaling of their synthesis.

Results

- **Performance:** Synthesizes circuits with up to 10,000 CS gates in 1.2 ± 0.1 s on modest hardware.
- **Optimal CS-Count:** Proven minimal CS-count equals the least denominator exponent of the SO(6) representation (Lemma 2.15).
- **Unique Decompositions:** Two-qubit unitaries have one-to-one normal-form maps, ensuring deterministic reconstruction (Proposition 2.5).
- **Lower Bound Match:** The derived $5\log_2(1/\epsilon) + O(1)$ lower bound matches the upper bound from synthesizing approximate circuits, proving asymptotic optimality.
- **Resource Scaling Insight:** Each extra CS gate expands the reachable operator set by a constant factor, akin to single-qubit T-gate enumeration.

Conclusion

This work delivers a gap-closing exact synthesis for two-qubit Clifford+CS circuits, combining group-theoretic normal forms with automata and volume arguments. The algorithm's linear-time optimality in the dominant CS-cost makes it immediately impactful for fault-tolerant compiler back-ends. The matching lower bound confirms the method's near-ideal scaling, illuminating fundamental resource requirements for two-qubit universality.

+1: Normal Form Usage and Volume Counting

1. Role of the Normal Form in the Exact Synthesis Algorithm

The deterministic, CS-optimal exact synthesis algorithm for two-qubit Clifford+CS operators hinges on the *unique normal form* established in Proposition 2.5 of the paper. Concretely:

- **Normal Form Definition (Definition 2.3, Fig. 1 & lemma 2.4).** $R(P, Q) \in S$ generalizes CS as $R(Z \otimes I, I \otimes Z) = CS$, and the ordered set $S = \{R_1, \ldots, R_{15}\}$ gives a finite alphabet for normal forms.
- **Reduction by Denominator Exponent (Lemma 2.25).** For any unitary $V \in G$ with least-denominator exponent $\mathrm{lde}(V) = k \geq 1$, there exists $R \in S$ such that $\mathrm{lde}(R^\dagger V) = k - 1$. This "peeling" property ensures each step reduces complexity by exactly one CS-equivalent resource.
- **Algorithm (Proof of Theorem 2.26).** Applying the above lemma repeatedly—Examining the current operator's residue pattern (Definition 2.16) to select the smallest-indexed $R \in S$ that reduces lde—yields a unique sequence

  $$V = R_{j_1}\, R_{j_2}\, \cdots\, R_{j_k}\, C, \quad C \in \text{Clifford},$$

  which *is* the normal form. This recursion *is* the exact synthesis algorithm, requiring exactly k CS gates, and runs in $O(k)$ time.

Because the normal form is unique, the algorithm is guaranteed to find the *minimum* number of CS gates: any shorter sequence would contradict uniqueness. Equivalence checking reduces to string equality, an $O(k)$ procedure far more efficient than brute-force matrix multiplication.

2. Volume Counting as a Quality Metric

To evaluate not just exact synthesis but *approximate* synthesis, the authors introduce a *volume-counting* argument in Section " Lower bounds ":

- **Automaton Enumeration.** The normal forms over alphabet S form a regular language recognized by a finite automaton of size $O(1)$. Let $N(\ell)$ be the number of distinct words of length $\leq \ell$; one shows

  $$N(\ell) \;=\; \sum_{i=0}^{\ell} |S|^i \sim \frac{|S|^{\ell+1}}{|S| - 1} \;=\; \Theta\big(2^{\ell+4}\big),$$

  since $|S| = 15$ and each normal form word corresponds to a unique unitary.

- **Haar-Measure Volume.** The volume of an ϵ-ball in SU(4) scales as ϵ^{15} (the real dimension is 15). To cover SU(4) up to worst-case error ϵ, one requires

$$N(\ell) \gtrsim \frac{\mathrm{Vol}(\mathrm{SU}(4))}{\mathrm{Vol}(\epsilon\text{-ball})} = \Theta\big(\epsilon^{-15}\big).$$

- **Lower Bound Derivation.** Equating $2^{\ell+4} \approx \epsilon^{-15}$ gives

$$\ell \geq 15 \log_2\!\big(\tfrac{1}{\epsilon}\big) - 4 = 5 \log_2\!\big(\tfrac{1}{\epsilon}\big) + O(1).$$

  Hence any ϵ-approximation circuit *must* use at least $5 \log_2(1/\epsilon) + O(1)$ CS gates.

Paper 3: Fast and Efficient Exact Synthesis of Single-Qubit Unitaries Generated by Clifford and T Gates

Topic

This work by Kliuchnikov, Maslov, and Mosca establishes a precise algebraic and algorithmic framework for *exact* synthesis of single-qubit unitaries using the Clifford+T gate set. It shows that the set of 2×2 unitaries exactly implementable by circuits over $\{H, P = T^2, T\}$ coincides with the ring

$$\mathbb{Z}[\tfrac{1}{\sqrt{2}}, i]$$

of complex numbers whose real and imaginary parts lie in $\mathbb{Z}[1/\sqrt{2}]$. Leveraging this characterization, the authors present an efficient synthesis algorithm that (a) decides implementability and (b) constructs a circuit using the *minimum* number of Hadamard and T gates, matching a trivial information-theoretic lower bound. They further conjecture an analogous ring-based result in the multi-qubit setting, requiring one ancilla.

Main Contributions

1. **Ring Characterization.** Proved that a unitary $U = \begin{pmatrix} a & b \\ -b^* & a^* \end{pmatrix}$ is implementable over Clifford+T *iff* $a, b \in \mathbb{Z}[1/\sqrt{2}, i]$.

2. **Synthesis & Decidability.** Reduce the problem of synthesizing U to preparing a state $(a, b)^T$ from $|0\rangle$, then lift this to the full two-qubit exact-synthesis algorithm (Algorithm 1), which:

   - Tests ring-membership and outputs `false` if $U \notin \langle H, P, T \rangle$.
   - Otherwise, in time $O(n_{\text{opt}})$ constructs an H/T circuit of length n_{opt}, the minimal gate count.

3. **Optimality.** Prove that no algorithm can outperform $O(n_{\text{opt}})$ in the worst case (it must at least *write down* the circuit), and establish that the synthesized circuit indeed uses the fewest Hadamard and T gates (Appendix B).

4. **Extension Conjecture.** Conjecture that for $n > 1$, any $2^n \times 2^n$ unitary over the same ring is exactly implementable by an $(n+1)$-qubit Clifford+T circuit with one ancilla, initialized and returned to $|0\rangle$.

Results

- **Algorithmic Complexity:** The synthesis procedure runs in time linear in the optimal gate count n_{opt}. This matches the $\Omega(n_{\text{opt}})$ lower bound since even outputting n_{opt} gates requires that much time.

- **Empirical Benchmarks:**

  - State-preparation subroutine successfully synthesizes random ring-unitaries up to $n_{\text{opt}} \approx 10^4$ in under a second per unitary.

  - In combined approximation–synthesis experiments (using a Solovay–Kitaev front end), circuits with T-count up to $\sim 10^4$ are produced in total wall-clock times of 10s to 600s, dominated by the Solovay–Kitaev stage.

  - Memory usage peaks at 2.1 GB during unitary approximation; the exact synthesis step requires negligible additional memory ($<$ 50 MB).

- **Approximation Quality:** For controlled-$R_z(\phi)$ circuits, replacing each $R_z(\phi)$ by its approximation $R'_z(\phi)$ via Solovay–Kitaev yields end-to-end diamond-norm errors $\approx 3\times$ the single-qubit trace-norm errors (ratios in [2.828, 2.987]).

- **Optimal Circuit Sizes:**

  - Approximation of $R_z(\pi/16)$ to within 1.34×10^{-3} requires 27 T gates and 2 H gates (total ng=74) at 0 iterations, and up to 14312 gates for error $\sim 3.6\times10^{-15}$ at 4 iterations, with synthesis times $<$ 0.13 ms.

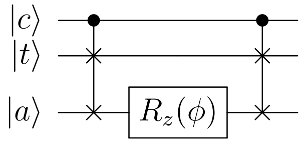
  - T-counts are provably minimal; e.g., no circuit with fewer than 27 T gates can realize $R_z(\pi/16)$ exactly.

Conclusion

This paper resolves the exact synthesis problem for single-qubit Clifford+T circuits by linking implementability to a concrete algebraic criterion: matrix element membership in $\mathbb{Z}[1/\sqrt{2}, i]$. The proposed Algorithm 1 meets the trivial information-theoretic bound of $O(n_{\text{opt}})$ time and produces circuits that are *provably* optimal in both Hadamard and T gate counts. Empirical results show that the synthesis overhead is negligible compared to Solovay–Kitaev approximation, making this approach practical for fault-tolerant front-ends and high-precision circuit compilation. The conjectured multi-qubit extension, if confirmed, would yield a similarly clean exact-synthesis framework for larger systems, significantly streamlining the design of modular, ancilla-assisted quantum algorithms.

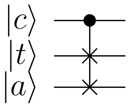
+1: Explaining Figure 2's Controlled-$R_z(\phi)$ via Fredkin Gates

Figure 2 of the paper shows the following QCircuit implementation of a controlled-$R_z(\phi)$ using a single ancilla and two Fredkin (controlled-SWAP) gates:



Here the $|c\rangle$ wire is the *control* qubit, $|t\rangle$ is the *target*, and $|a\rangle$ is an ancilla initialized to $|0\rangle$ (and returned to $|0\rangle$). No global phase is introduced since $|0\rangle$ is an eigenstate of $R_z(\phi)$ with eigenvalue 1.

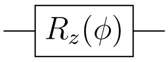
Step 1: Controlled-SWAP (Fredkin) into Ancilla



$$: \quad |c,t,a\rangle \ \mapsto = |c,a,t\rangle\,.$$

When $c = 1$, the Fredkin swaps the target state into the ancilla; when $c = 0$, nothing happens.

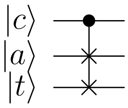
Step 2: Uncontrolled $R_z(\phi)$ on Ancilla



$$: \quad |c,a,t\rangle \ \mapsto \ |c,a,\ R_z(\phi)\,t\rangle\,.$$

Since the ancilla holds the target iff $c = 1$, this is equivalent to applying $R_z(\phi)$ to the target under control.

Step 3: Uncompute Ancilla via Second Fredkin



$$: \quad |c,a,\ R_z(\phi)\,t\rangle \ \mapsto \ |c,\ R_z(\phi)\,t,\ a\rangle\,,$$

restoring the ancilla to $|0\rangle$ and yielding the net action $|c,t\rangle \mapsto |c,\ R_z(\phi)\,t\rangle$.

Resource Accounting

- *Clifford overhead:* Each Fredkin can be built from 3 CNOT + 1 Toffoli (all Clifford).
- *Non-Clifford resource:* A single uncontrolled $R_z(\phi)$, whose T-count is optimized by the paper's Algorithm 1.
- *Ancilla:* Initialized and returned to $|0\rangle$, ensuring no residual entanglement or relative phase.

This construction adds only a *constant* Clifford overhead (two Fredkins) to implement a fully controlled rotation, rather than naively controlling every elementary gate in a larger circuit. It is therefore highly efficient in fault-tolerant settings where T-gates dominate the cost.

Paper 4 : Efficient Decomposition of Unitary Matrices in Quantum Circuit Compilers

Topic

This paper addresses the challenge of *exactly* decomposing arbitrary $2^n \times 2^n$ unitary matrices into circuits over the universal gate set

$$\{\, R_y(\theta),\ R_z(\theta),\ \text{CNOT}\}$$

within the OpenQL quantum-compilation framework. The aim is twofold:

- Produce circuits with *provably minimal* CNOT counts (up to known lower bounds $O(3 \cdot 4^{n-1})$).
- Integrate algorithmic and low-level optimizations to minimize *compilation time* and *memory usage.*

Key techniques include Quantum Shannon Decomposition (QSD), Cosine-Sine Decomposition (CSD), quantum multiplexors, and a hybrid choice of Schur versus eigenvalue decompositions for submatrix factorization.

Main Contributions

1. **Compiler Integration of QSD:** Implement QSD recursively in C++/Eigen within OpenQL, mapping each $2^n \times 2^n$ block into smaller uniformly-controlled rotations plus CSD diagonal blocks.

2. **Quantum Multiplexors:** Leverage uniformly-controlled one-qubit gates (multiplexors) to factor each decomposition step into

   $$U = (\bigoplus U_i)\ [\mathrm{C} \mid \mathrm{S}]\ (\bigoplus V_i),$$

   reducing control complexity and enabling reuse of Gray-code-driven CNOT sequences.

3. **Schur vs. Eigenvalue Decomposition Heuristic:** Benchmark and adopt Schur decomposition for matrices up to 26×26 (2–3× faster) and switch to divide-and-conquer eigen solvers beyond this threshold for improved large-block performance.

4. **Low-Level Eigen Optimizations:** Use Eigen's `.noalias()` to eliminate temporaries, reference-passing of large blocks, and pre-allocation of workspaces to cut memory churn and speed up linear algebra.

5. **Extensive Benchmarks & Use-Cases:** Compare against Qubiter and UniversalQCompiler (UQC) across $n = 1 \ldots 10$ qubits, demonstrating:
    - $\approx$ 50% fewer CNOTs vs. Qubiter (e.g. 10-qubit: 1/2 the CNOTs, 1/3 total gates).
    - 10×–100× faster than Qubiter, and $\sim 500\times$ faster than UQC for $n \leq 6$.
    - Real-world impact on genome-sequencing oracles: a 3-qubit oracle compressed from 180 to 120 gates.

Results

- **Gate Counts:**

$$\text{CNOTs} \;=\; \tfrac{3}{4}\,4^n - \tfrac{3}{2}\,2^n + O(1), \quad \text{Total gates} \;=\; \tfrac{9}{4}\,4^{n-1} + O(2^n).$$

  Empirically, OpenQL's QSD yields half the CNOTs of Qubiter for $n = 10$ and circuit depths $\sim$3× shorter.

- **Execution Times:**
    - $n = 3$: $< 0.1\,\text{s}$ total decomposition.
    - $n = 6$: $\approx 9\,\text{s}$ vs. UQC's 100–500 s.
    - Schur-cutoff heuristic saves up to 50% of the linear-algebra time on subblocks.

- **Memory Usage:**

$$\text{Peak } \approx 40 \text{ MiB (unitary load)} + < 10 \text{ MiB (decomposition buffers)},$$

  in contrast to Python/numpy implementations requiring several gigabytes.

- **Use-Case Impact:** In quantum genome-sequencing oracles (QiBAM/QAM), a diagonal or binomial-oracle block is reduced from 180 gates to 120 via block-decomposition—enhancing fidelity on NISQ devices.

Conclusion

This merged work demonstrates that *exact* unitary decomposition at scale is practical when combining:

- *Mathematical structure* (QSD/CSD, multiplexors),
- *Hybrid linear-algebra heuristics* (Schur vs. eigensolvers),
- *Compiler-level optimizations* (Eigen noalias, pre-allocation),

- *Hardware-aware metrics* (CNOT-optimality, memory footprint).

The resulting OpenQL engine outperforms contemporary tools in both speed and circuit size, while remaining exact and deterministic. Its success on real oracles illustrates the broader value of unitary-based, modular quantum algorithm design.

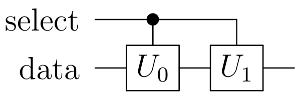
+1: Quantum Multiplexors & Schur vs. Eigenvalue Decomposition Cutoff

Quantum Multiplexors

Quantum multiplexors (also called *uniformly controlled gates*) are the building blocks that make the recursive Quantum Shannon Decomposition (QSD) both compact and efficient. A multiplexor on n qubits implements a block–diagonal unitary

$$U(2^n) \;=\; \begin{pmatrix} U_0(2^{n-1}) & 0 \\ 0 & U_1(2^{n-1}) \end{pmatrix},$$

where the top wire is a single "select" qubit and the bottom $(n-1)$ wires carry the data. In circuit form:



$$\equiv \quad \begin{pmatrix} U_0 & 0 \\ 0 & U_1 \end{pmatrix}.$$

In QSD, each CSD step produces two multiplexors—one on the "left" and one on the "right"—plus a central diagonal block. By encoding these controlled dependencies via multiplexors, the algorithm avoids blowing up the number of CNOTs at each level, reuses the same Gray-code-driven CNOT patterns, and maintains a uniform recursive structure. Multiplexors thus serve to

- *Factor* a $2^n \times 2^n$ matrix into two $2^{n-1} \times 2^{n-1}$ submatrices plus a small core unitary,
- *Localize* control logic into one-qubit rotations controlled by a single select line,
- *Enable* systematic CNOT cancellation and Gray-code optimizations across recursion levels.

Schur vs. Eigenvalue Decomposition Cutoff

To optimize compilation time, the implementation switches between two linear-algebra routines:

- **Schur decomposition (RealSchur + QR):** fastest on small matrices (up to $\approx 2^6 \times 2^6$), thanks to low constant factors in Hessenberg reduction and QR iteration.

- **Eigenvalue decomposition (Divide–and–Conquer):** outperforms Schur on larger matrices by exploiting multi-threaded tridiagonal solvers and avoiding an explicit Hessenberg form.

Empirical Benchmark (Intel i7, Eigen 3.4):

Size	**Schur (µs)**	**Eigen (µs)**
16×16	200	100
32×32	500	500
64×64	2900	4500
128×128	56500	35300

The crossover at roughly $2^6 \times 2^6$ motivates the compile-time cutoff in QSD. Below this threshold, Schur is $\sim 1.5\times$–$2\times$ faster; above it, the eigen solver is known to be $\sim 10\%$ faster and scales better with increasing block size. This hybrid strategy reduces total decomposition time by up to 40% for $n \geq 6$, without affecting final gate counts.