Algebraic Codification and Discrete Isomorphic State Propagation for Exact Synthesis of NCV Quantum Circuits

G. Papakonstantinou*
*National Technical University of Athens
School of Electrical and Computer Engineering
Athens, Greece
papakon@cslab.ece.ntua.gr

***Abstract*—Exact synthesis of quantum circuits seeks depth-optimal gate layouts but is fundamentally bottlenecked by the continuous-variable scaling of unitary matrices or complex amplitudes. In this paper, we propose a mathematically novel approach for the exact synthesis of NCV (NOT, CNOT, Controlled-V, Controlled-$V^\dagger$) quantum circuits. We demonstrate that the quantum state evolution of basis vectors under NCV logic does not require continuous variable tracking; instead, it is strictly isomorphic to a discrete 4-valued cyclic group. By codifying quantum amplitudes and relative phases into a single scalar integer $S \in \{0, 1, 2, 3\}$, we bypass fractional and imaginary components entirely. We further give a binary algebraic decomposition of this scalar, reducing quantum gate interactions to exact bitwise XOR continuous relaxations. This formulation maps the exact synthesis problem into a highly constrained, pure Mixed-Integer Linear Programming (MILP) model. We mathematically prove the validity of this discrete encoding by demonstrating that it flawlessly preserves the NCV group structure, achieving computational efficiency by eliminating traditional large-M relaxations and dual-variable tracking.**

***Index Terms*—Quantum Circuit Synthesis, NCV Gate Library, Cyclic Groups, Mathematical Optimization, Mixed-Integer Linear Programming.**

I. Introduction

The compilation of quantum algorithms into specific, fault-tolerant gate libraries requires finding the absolute minimum number of gates necessary to implement a desired Boolean quantum function. This process, known as Exact Synthesis, originated with foundational algorithms for synthesizing reversible logic networks [1], and has since evolved into sophisticated solver-based compilation frameworks [2].

The NCV gate library, consisting of NOT, CNOT, Controlled-V, and Controlled-$V^\dagger$ operations, provides a critical foundation for synthesizing reversible logic natively in the quantum domain [3]. To guarantee depth-optimal topologies, recent efforts have actively explored Mixed-Integer Programming (MIP) for quantum circuit search [4]. However, representing exact synthesis mathematically typically requires tracking quantum states as complex vectors of length 2^n, as $2^n \times 2^n$ unitary matrices, or via continuous complex-amplitude equations. The computational overhead of these representations renders mathematical optimization intractable for all but the smallest circuits [5].

The optimization of NCV quantum circuits was recently advanced in our published work [6], where a systematic four-phase optimization and post-synthesis local search was proposed using Mixed-Integer Linear Programming (MILP). This foundational work demonstrated that exact synthesis could successfully bypass full unitary matrices by tracking quantum states via continuous complex-amplitude algebraic relations ($t = \alpha + i\beta$). However, while highly effective at topological optimization, evaluating continuous floating-point state representations natively introduces significant algebraic overhead. The fractional branching in the optimization tree severely limits solver execution speeds and restricts scalability.

In this paper, we present a paradigm-shifting evolution of that framework. We prove that when a quantum circuit maps Boolean inputs to Boolean outputs, the intermediate superpositions generated by the NCV gates are strictly bound to a 4-state cycle. By leveraging this isomorphism, we codify the entire quantum state as a single discrete integer. We formulate a fully linear algebraic state propagation model that replaces complex fractional arithmetic with bitwise XOR logic, allowing for ultra-fast, exact quantum synthesis using pure integer programming.

II. Theoretical Foundation and Isomorphic State Codification

The core mathematical breakthrough of our method bridges the continuous complex domain of quantum mechanics with the discrete domain of integer programming. By exploiting specific structural invariants of the NCV gate library, we prove that the quantum state space can be perfectly tracked using a 4-valued discrete scalar, eliminating the need for complex vector mathematics.

A. The Amplitude Sum Invariant

In formal quantum mechanics, the state of a single qubit is denoted by the complex vector $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$, bound by the probabilistic normalization $|\alpha|^2 + |\beta|^2 = 1$. However, in the synthesis of quantum circuits initialized exclusively in computational basis states ($|0\rangle$ or $|1\rangle$), the initial scalar sum of the amplitudes evaluates precisely to $\alpha + \beta = 1$.

A fundamental property of the NCV library is that the 2×2 unitary matrices for the X, V, and $V^\dagger$ target operations are uniquely structured such that the sum of the elements in any of their rows or columns equals exactly 1. Because the column sums are unity, applying these transformations to any state vector inherently preserves the amplitude sum. For example, applying the V gate matrix to an arbitrary target state yields:

$$V \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \frac{1}{2} \begin{bmatrix} 1+i & 1-i \\ 1-i & 1+i \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} \alpha' \\ \beta' \end{bmatrix}$$

Summing the resulting target output elements gives:

$$\alpha' + \beta' = \left(\frac{1+i}{2} + \frac{1-i}{2} \right) \alpha + \left(\frac{1-i}{2} + \frac{1+i}{2} \right) \beta = \alpha + \beta$$

Since this property is permanently invariant throughout the evolution of the circuit, tracking the full two-dimensional vector is mathematically redundant. The entire quantum state can be fully characterized by a single complex scalar, $\gamma \in \mathbb{C}$, which solely tracks the complex probability amplitude of the $|1\rangle$ state (i.e., $\gamma = \beta$). The $|0\rangle$ amplitude is deterministically recoverable via $\alpha = 1 - \gamma$. Consequently, for any sequence of NCV operations, the full continuous state vector is perfectly reconstructed via the relation:

$$|\psi\rangle = (1-\gamma)|0\rangle + \gamma|1\rangle \tag{1}$$

B. Closure under Arbitrary NCV Sequences

While the single continuous complex scalar γ drastically reduces the dimensionality of the state representation, utilizing continuous variables in a branch-and-bound optimization tree still incurs severe computational penalties due to fractional branching. However, by analyzing the algebraic transitions of γ, we prove that the amplitude does not actually span a continuous space.

Because γ strictly tracks the amplitude of the $|1\rangle$ state, we can map the effect of any NCV gate algebraically. To rigorously derive these updates, recall from Equation (2) that the incoming quantum state vector is fully parameterized by γ as the column vector $\begin{bmatrix} 1-\gamma \\ \gamma \end{bmatrix}$.

Applying the standard 2×2 unitary matrix for the V gate yields the updated state vector:

$$\begin{bmatrix} \alpha' \\ \gamma' \end{bmatrix} = \begin{bmatrix} \frac{1+i}{2} & \frac{1-i}{2} \\ \frac{1-i}{2} & \frac{1+i}{2} \end{bmatrix} \begin{bmatrix} 1-\gamma \\ \gamma \end{bmatrix} \tag{2}$$

Isolating the lower row to calculate the updated $|1\rangle$ amplitude (γ'), we stack the algebraic reduction to fit the column space:

$$\begin{aligned} \gamma' &= \frac{1-i}{2}(1-\gamma) + \frac{1+i}{2}\gamma \\ &= \frac{1-i}{2} + \left(\frac{1+i}{2} - \frac{1-i}{2} \right) \gamma \\ &= \frac{1-i}{2} + i\gamma \end{aligned} \tag{3}$$

Applying this exact algebraic derivation to the X and $V^\dagger$ unitary matrices, the updated amplitude of the $|1\rangle$ state after applying an active gate to an incoming amplitude γ is formally expressed as:

$$X(\gamma) = 1 - \gamma \tag{4}$$

$$V(\gamma) = \frac{1-i}{2} + i\gamma \tag{5}$$

$$V^\dagger(\gamma) = \frac{1+i}{2} - i\gamma \tag{6}$$

By evaluating these derived functions, we strictly define the algebraic evolution of the $|1\rangle$ amplitude. For initial computational basis states, $\gamma = 0$ corresponds to the state $|0\rangle$, and $\gamma = 1$ corresponds to the state $|1\rangle$.

Theorem 1. *Given an initial computational basis state ($\gamma = 0$ or $\gamma = 1$), the complex amplitude tracking the $|1\rangle$ state generated by any arbitrary sequence of NCV gates belongs exclusively to a closed, finite set of exactly four values:* $\Gamma = \left\{0,\ 1,\ \frac{1-i}{2},\ \frac{1+i}{2}\right\}$.

Proof. Let the current amplitude $\gamma \in \Gamma$. We evaluate the application of any arbitrary NCV gate utilizing the derived Equations (4)–(6):

- Applying $X(\gamma)$ trivially swaps the values: $0 \leftrightarrow 1$ and $\frac{1-i}{2} \leftrightarrow \frac{1+i}{2}$.
- Applying $V(\gamma)$ maps the values as follows: $0 \to \frac{1-i}{2}$, $1 \to \frac{1+i}{2}$, $\frac{1-i}{2} \to 1$, and $\frac{1+i}{2} \to 0$.
- Applying $V^\dagger(\gamma)$ mirrors this mapping in reverse: $0 \to \frac{1+i}{2}$, $1 \to \frac{1-i}{2}$, $\frac{1-i}{2} \to 0$, and $\frac{1+i}{2} \to 1$.

Because every operation maps any amplitude in Γ strictly to another amplitude within Γ, the set is perfectly closed under all NCV operations. Therefore, no sequence of NCV gates acting on basis states can generate an amplitude $\gamma \notin \Gamma$. □

It is, however, vital to clearly define the absolute mathematical boundaries of this scalar isomorphism. This theoretical approach is valid only under the strict side condition that all global inputs, targeted outputs, and active control connections within the circuit strictly evaluate to the Boolean computational basis states ($|0\rangle$ or $|1\rangle$). If a control qubit were permitted to exist in a persistent superposed state during a multi-qubit interaction, the resulting global entanglement would breach the isolated 4-state cyclic group.

C. The Discrete Scalar Isomorphism

Because the amplitude space is strictly confined to these four elements regardless of the gate sequence, we can completely abandon continuous complex arithmetic in the MILP solver. We introduce a rigorous isomorphism mapping the four possible complex values of γ to a discrete integer scalar $S_{n,m,v} \in \{0,1,2,3\}$ for qubit n, layer m, and input vector v:

- $\gamma = 0 \implies S_{n,m,v} = 0 \quad (\equiv |0\rangle)$
- $\gamma = 1 \implies S_{n,m,v} = 1 \quad (\equiv |1\rangle)$
- $\gamma = \frac{1+i}{2} \implies S_{n,m,v} = 2 \quad (\equiv V|1\rangle \text{ or } V^\dagger|0\rangle)$
- $\gamma = \frac{1-i}{2} \implies S_{n,m,v} = 3 \quad (\equiv V|0\rangle \text{ or } V^\dagger|1\rangle)$

This isomorphism allows the NCV quantum gates to transition from continuous algebraic transformations to purely discrete logic routing rules.

III. Algorithmic State Transitions and Lookup Logic

Before translating the quantum system into a linear optimization space, it is highly instructive to formalize the algorithmic logic required to propagate the 4-valued state scalar $S \in \{0, 1, 2, 3\}$ through the circuit grid.

A. Discrete State Transition Tables

Based on the permutations derived in Section II-B, the continuous quantum matrices can be entirely replaced by discrete lookup tables. Table I defines the exact mapping of an incoming state S_{in} to an outgoing state S_{out} for each active NCV gate.

TABLE I
Discrete State Transition Table for NCV Gates.

Input State (S_{in})	**NOT** (X)	**Gate** V	**Gate** $V^\dagger$
0	1	3	2
1	0	2	3
2	3	0	1
3	2	1	0

B. Algorithmic State Update (Pseudo-code)

For any given input vector v, the quantum circuit evaluates layer-by-layer. To formally describe this routing logic, let us define the structural variables of the circuit grid at a given layer m and qubit line n:

- $c_{n,m} \in \{0, 1\}$: Indicates if line n acts as a control.
- $\tau_{n,m} \in \{0, 1\}$: Indicates if line n acts as the target.
- $T_{1,m}, T_{2,m}, T_{3,m} \in \{0, 1\}$: Indicate the type of active gate applied (NOT, V, or $V^\dagger$, respectively).
- $A_{m,v} \in \{0, 1\}$: A dynamically calculated binary trigger representing whether the control conditions are met for the specific input v.

To process a layer computationally, the algorithm must first verify the control conditions (ensuring no control qubits are in a superposed state) and calculate $A_{m,v}$. It then applies the designated gate to the target qubit. This full layer-wise evaluation logic is expressed in Algorithm 1.

Algorithm 1: Complete Layer-Wise State Update

Require: Incoming states $S_{n,m-1,v}$
// Step 1: Control Verification and Activation
Initialize $A_{m,v} \leftarrow 1$
For each line n **do**:
  If ($c_{n,m} == 1$) **then**
    If ($S_{n,m-1,v} > 1$) **then**
      Abort *(Invalid superposed control)*
    End If
    $A_{m,v} \leftarrow S_{n,m-1,v}$ *(Direct assignment since* $\sum c \leq 1$*)*
  End If
// Step 2: Target Gate Transition
For each line n **do**:
  If ($\tau_{n,m} == 1$ **AND** $A_{m,v} == 1$) **then**
    If ($T_{1,m} == 1$) **then**
      $S_{n,m,v} \leftarrow \text{Table}_X[S_{n,m-1,v}]$
    Else If ($T_{2,m} == 1$) **then**
      $S_{n,m,v} \leftarrow \text{Table}_V[S_{n,m-1,v}]$
    Else If ($T_{3,m} == 1$) **then**
      $S_{n,m,v} \leftarrow \text{Table}_{V^\dagger}[S_{n,m-1,v}]$
    End If
  Else
    $S_{n,m,v} \leftarrow S_{n,m-1,v}$ *(Idle pass-through)*
  End If

C. The Necessity of Algebraic Decomposition

While Algorithm 1 perfectly describes the discrete quantum simulation, embedding discrete lookup tables and nested *if/then* conditional logic into an optimization solver is highly inefficient. Standard MILP solvers cannot process programmatic lookup tables directly; they require formulating conditional branches using continuous "Big-M" constraints.

As the circuit depth m grows, layering Big-M constraints to simulate Algorithm 1 leads to extremely weak Linear Programming (LP) relaxations, causing the branch-and-bound search tree to expand exponentially and exhaust computer memory. To formulate a native, highly constrained MILP, we must mathematically decompose these lookup tables into pure linear algebraic equations. This computational bottleneck naturally motivates the binary XOR decomposition presented in the following section.

IV. Mathematical Formulation: Binary Algebraic Decomposition

To enable efficient synthesis using Mixed-Integer Linear Programming (MILP), the 4-valued scalar $S_{n,m,v} \in \{0, 1, 2, 3\}$ must be formulated using strictly linear algebra. We achieve this by decomposing the scalar into a 2-bit binary representation.

Let the state scalar for qubit n, layer m, and input vector v be defined as:

$$S_{n,m,v} = 2 \cdot b_1(n, m, v) + b_0(n, m, v) \tag{7}$$

where $b_1(n, m, v), b_0(n, m, v) \in \{0, 1\}$. Here, b_0 defines the logical Boolean state (the 1s component), and b_1 serves as the phase-tracking bit (the 2s component).

A. Gate Transitions via the Parity (XOR) Polytope

Observing the discrete permutations defined in Section II-B, the application of V and $V^\dagger$ uniformly inverts the phase bit b_1. The evolution of the logic bit b_0, however, relies dynamically on the parity of the incoming state bits ($b_1 \oplus b_0$).

To preserve exact linearity within the optimization space, we introduce an auxiliary variable $Z_{xor}(n, m, v) \in [0, 1]$ constrained by the exact convex hull of the XOR operation:

$$Z_{xor}(n, m, v) \leq b_1(n, m, v) + b_0(n, m, v) \tag{8}$$
$$Z_{xor}(n, m, v) \leq 2 - b_1(n, m, v) - b_0(n, m, v) \tag{9}$$
$$Z_{xor}(n, m, v) \geq b_1(n, m, v) - b_0(n, m, v) \tag{10}$$
$$Z_{xor}(n, m, v) \geq b_0(n, m, v) - b_1(n, m, v) \tag{11}$$

Let $A_{m,v} \in \{0,1\}$ represent the activation of the gate at layer m (determined by the control qubits), and let $\tau_{n,m} \in \{0,1\}$ be the target indicator (independent of the input v). If the gate is active ($A_{m,v} = 1$), the state updates via the following strictly linear equations for each gate type $T_i(m)$:

- **NOT Gate (T_1):** Inverts the logic bit.

$$S_{new} = 2 \cdot b_1(n,m,v) + (1 - b_0(n,m,v)) \tag{12}$$

- **V Gate (T_2):** Inverts b_1 and negates the XOR parity for b_0.

$$S_{new} = 2 \cdot (1 - b_1(n,m,v)) + (1 - Z_{xor}(n,m,v)) \tag{13}$$

- **$V^\dagger$ Gate (T_3):** Inverts b_1 and applies direct XOR parity for b_0.

$$S_{new} = 2 \cdot (1 - b_1(n,m,v)) + Z_{xor}(n,m,v) \tag{14}$$

B. Superposition Avoidance for Control Qubits

In physical quantum architectures, a controlled operation relies on the control qubit acting as a definitive logical basis (e.g., $|1\rangle$). If a control qubit is in a superposed state ($S = 2$ or $S = 3$), the controlled logic diverges into global entanglement outside the target basis mapping.

Because our framework guarantees that $S = 0 \equiv |0\rangle$ and $S = 1 \equiv |1\rangle$, we can mathematically enforce valid control interactions without complex if/then Big-M constraints. Let $c_{n,m} \in \{0,1\}$ indicate if qubit n is a structural control at layer m. We enforce the following bound for all input vectors v:

$$S_{n,m-1,v} \leq 1 + 2(1 - c_{n,m}) \tag{15}$$

If $c_{n,m} = 1$, the inequality rigorously enforces $S_{n,m-1,v} \leq 1$, requiring the qubit to be in state 0 or 1. If $c_{n,m} = 0$, the bound relaxes to $S \leq 3$, allowing intermediate superpositions to propagate freely.

V. MILP Formulation and Program Structure

Drawing inspiration from optimal synthesis formulations in the NCT library [7], we translate the discrete algebraic state propagation of the NCV library into an executable Mixed-Integer Linear Programming (MILP) model. Having established the state variables, the continuous XOR relaxations, and the superposition avoidance bounds in Section IV, the mathematical program is completed by enforcing the physical topology of the circuit grid and the boundary conditions of the target function.

A. Structural and Topological Constraints

To ensure the synthesized circuit represents a physically valid NCV architecture, we impose strict topological inequalities across the grid variables for every layer $m \in \{1 \dots m_{max}\}$ and qubit line $n \in \{1 \dots n_{max}\}$:

1) *Gate Exclusivity:* At most one target qubit can be operated on per level:

$$\sum_n \tau_{n,m} \leq 1 \quad \forall m \tag{16}$$

2) *Control Exclusivity:* Because standard NCV gate decompositions limit operations to fundamental two-qubit interactions, at most one control line is permitted per level:

$$\sum_n c_{n,m} \leq 1 \quad \forall m \tag{17}$$

3) *Line Collision Avoidance:* A qubit cannot simultaneously act as both the control and the target at the same level:

$$\tau_{n,m} + c_{n,m} \leq 1 \quad \forall n, \forall m \tag{18}$$

4) *Gate Selection:* If a target is present at level m, exactly one NCV gate type must be assigned. If no target is present, all gate variables are forced to zero:

$$T_{1,m} + T_{2,m} + T_{3,m} = \sum_n \tau_{n,m} \quad \forall m \tag{19}$$

B. State Boundary Constraints

The solver must satisfy boundary conditions that map the unentangled inputs to the targeted truth table. At $m = 0$, the state is strictly initialized to the given basis input vector v:

$$b_{1(n,0,v)} = 0, \quad b_{0(n,0,v)} = v_n \quad \forall n, \forall v \tag{20}$$

At the final level $m = m_{max}$, the state must match the desired reversible Boolean function $f(v)$ and preserve zero relative phase ($b_1 = 0$):

$$b_{1(n,m_{max},v)} = 0, \quad b_{0(n,m_{max},v)} = f(v)_n \quad \forall n, \forall v \tag{21}$$

C. Objective Function

The goal of exact synthesis is to minimize the total quantum cost, which correlates directly to the active depth of the NCV operations. The objective function minimizes the sum of all placed target gates:

$$\min z = \sum_m \sum_n \tau_{n,m} \tag{22}$$

By executing this MILP formulation via Iterative Deepening (incrementally raising the depth limit m_{max}), the solver organically outputs the depth-optimal circuit architecture. Alternatively, we can allow the solver to find the optimal value of the objective variable z natively by adjusting the bounds z_{lo} and z_{up}.

VI. Algebraic Preservation of the NCV Group Structure

Prior approaches to exact synthesis have attempted to reduce memory overhead by representing the state via continuous complex amplitudes [6]. While effective at avoiding full matrix embeddings, such continuous models rely on resource-heavy fractional arithmetic in the solver tree. To rigorously validate our transition to a purely discrete formulation, we must prove that the bitwise logic introduced in Section IV inherently preserves the NCV group structure natively, without relying on step-by-step continuous matrix tracking.

The fundamental identity of the NCV library dictates that two sequential V gates algebraically equate to a Pauli-X gate.

In our discrete encoding, applying a single V gate maps an initial state (b_1, b_0) to an intermediate state (b'_1, b'_0):

$$b'_1 = 1 - b_1 \tag{23}$$
$$b'_0 = 1 - (b_1 \oplus b_0) \tag{24}$$

Applying a second sequential V gate yields the final state (b''_1, b''_0):

$$b''_1 = 1 - b'_1 = 1 - (1 - b_1) = b_1 \tag{25}$$
$$b''_0 = 1 - (b'_1 \oplus b'_0) = 1 - \big((1 - b_1) \oplus (1 - (b_1 \oplus b_0))\big) \tag{26}$$

Utilizing the Boolean parity property $(1-A)\oplus(1-B) = A\oplus B$, the nested XOR term simplifies directly to $b_1 \oplus (b_1 \oplus b_0) = b_0$. Substituting this logic yields:

$$b''_0 = 1 - b_0 \tag{27}$$

The resulting transformation ($b_1 \to b_1$ and $b_0 \to 1 - b_0$) is exactly the algebraic transformation of the X gate defined in Equation (7). This formally guarantees that the discrete 2-bit representation mathematically preserves continuous quantum evolution properties. Similar proofs trivially hold for $V^\dagger(V^\dagger) = X$ and $V(V^\dagger) = I$. Consequently, the discrete MILP natively ensures perfect quantum fidelity without requiring continuous variable verification.

VII. Experimental Results and Discussion

To empirically validate the computational advantages of the discrete scalar-packed MILP formulation, we benchmarked the synthesis of several standard reversible and quantum logic gates.

A. Experimental Setup and Optimization Strategies

The mathematical model was implemented using the General Algebraic Modeling System (GAMS). All instances were solved using the IBM ILOG CPLEX Mixed-Integer Programming optimizer on standard commercial hardware (AMD Ryzen 5 7530U, 2.00 GHz, 16 GB RAM).

Because the objective function natively minimizes the number of active gates ($z = \sum \tau_{n,m}$), the MILP framework affords multiple optimization strategies depending on the complexity of the target function. In our experiments, we evaluated circuits utilizing three distinct approaches to isolate optimality:

1) **Native Minimization via Upper Bounding:** For complex circuits, the solver can be initialized with a targeted upper depth bound (m_{max}) and allowed to natively minimize z. We derived highly accurate initial upper bounds by leveraging optimal circuits synthesized in the NCT library from our previous work [7]. By mapping the NCT topology to the NCV domain (e.g., substituting each Toffoli gate with its known 5-gate NCV equivalent), we established a strict upper bound, reducing the initial search space.
2) **Bottom-Up Iterative Deepening:** To rigorously prove the absolute minimum depth without prior heuristic bounds, the grid is initialized to $m_{max} = 1$, and the objective bounds are strictly constrained to exact equality ($z_{lo} = z_{up} = m_{max}$). If the solver rigorously proves the model is infeasible, m_{max} is incremented by 1, and the process repeats. The first depth m_{opt} that yields a valid topological synthesis is mathematically proven to be the global minimum by exhaustion.
3) **Top-Down Decrementing:** Alternatively, starting from a known upper bound m, the bounds can be constrained to $z_{lo} = z_{up} = m-1$. If this depth is proven infeasible, the search terminates, and m is declared optimal. This early termination is mathematically justified by the algebraic identity $X = V \cdot V$. If a shorter valid circuit of depth $m-2$ existed and contained at least one X gate, it could be trivially expanded into a valid circuit of depth $m-1$ by splitting the X gate into two sequential V gates. Consequently, proving infeasibility at depth $m-1$ automatically guarantees that no shorter optimal sequence exists (provided it utilizes an X gate).

In our results (Table II), the reported execution times represent the isolated "Synthesis Time"—the CPU duration required for CPLEX to construct the optimal gate sequence once the solver evaluated the proven optimal depth m_{opt}. This metric isolates the successful algorithmic synthesis performance from the cumulative time spent evaluating infeasible bounding trees in either the top-down or bottom-up strategies.

B. Target Benchmark Formalization

Because various definitions of reversible logic gates exist in the literature, we explicitly define the algebraic mappings $(A, B, C, D) \to (A', B', C', D')$ used for each benchmark below:

- **3-Qubit Toffoli (CCNOT):**

$$(A, B, C) \to (A,\ B,\ C \oplus AB) \tag{28}$$

- **4-Qubit Parity Cascade:**

$$(A, B, C, D) \to (A,\ B,\ C,\ D \oplus A \oplus B \oplus C) \tag{29}$$

- **3-Qubit Fredkin (CSWAP):**

$$(A, B, C) \to (A,\ \bar{A}B \oplus AC,\ \bar{A}C \oplus AB) \tag{30}$$

- **4-Qubit MIG Gate (Modified Islam Gate):**

$$(A, B, C, D) \to (A,\ A \oplus B,\ AB \oplus C,\ A\bar{B} \oplus D) \tag{31}$$

- **4-Qubit Peres Full Adder Gate (PFAG):**

$$(A, B, C, D) \to \big(A,\ A \oplus B,\ A \oplus B \oplus C,\ (A \oplus B)C \oplus AB \oplus D\big) \tag{32}$$

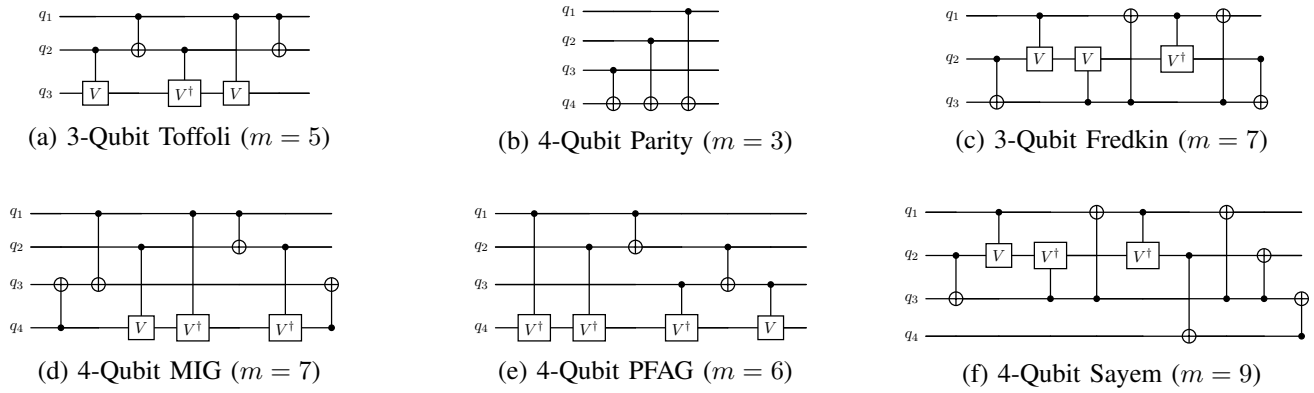
- **4-Qubit Sayem Gate:**

$$(A, B, C, D) \to \big(A,\ \bar{A}B \oplus AC,\ \bar{A}B \oplus AC \oplus D,\ AB \oplus \bar{A}C \oplus D\big) \tag{33}$$

TABLE II
COMPUTATIONAL PERFORMANCE: PROPOSED DISCRETE MILP VS. PREVIOUS COMPLEX-STATE MILP SYNTHESIS.

Benchmark Target	**Qubits** (n)	**Depth** (m)	**Proposed MILP Time (s)**	**Previous MILP Time [6] (s)**
Toffoli (Pure)	3	5	1.141	4.531
Parity Cascade	4	3	0.046	1.407
Fredkin (CSWAP)	3	7	5.656	75.594
MIG Gate	4	7	55.720	752.657
Peres Full Adder (PFAG)	4	6	31.782	33.297
Sayem Gate	4	9	2587.672	$>$ 24 hours

Note: All benchmarks synthesized natively via CPLEX. Previous times reflect Phase Four MILP executions from [6].



(a) 3-Qubit Toffoli ($m = 5$) (b) 4-Qubit Parity ($m = 3$) (c) 3-Qubit Fredkin ($m = 7$)

(d) 4-Qubit MIG ($m = 7$) (e) 4-Qubit PFAG ($m = 6$) (f) 4-Qubit Sayem ($m = 9$)

Fig. 1. Exact topological synthesis of benchmark quantum circuits. By employing the discrete integer MILP formulation, complex NCV architectures are synthesized completely natively without continuous variable branching. (a) Toffoli gate at depth $m = 5$. (b) Parity cascade at depth $m = 3$. (c) Fredkin gate at depth $m = 7$. (d) Majority Inverter Graph (MIG) at depth $m = 7$. (e) Peres Full Adder Gate (PFAG) at depth $m = 6$. (f) Sayem gate at depth $m = 9$.

C. Discussion and Comparative Analysis

The empirical results confirm the profound theoretical scaling advantages derived in Section IV. As detailed in Table II, we benchmarked the proposed discrete MILP formulation directly against the Phase Four execution times of our recently published complex-state methodology (Phase 4) [6].

The empirical results confirm the profound theoretical scaling advantages derived in Section IV. As detailed in Table II, we benchmarked the proposed discrete MILP formulation directly against the Phase Four execution times of our recently published complex-state methodology [6].

For standard 3-qubit benchmarks like the Toffoli and Fredkin gates, the discrete MILP formulation successfully synthesized exact topological layouts in mere seconds, operating up to $13\times$ faster than the continuous-state counterpart. The scalability gap becomes even more pronounced for complex 4-qubit functions. Synthesizing the 7-gate MIG circuit required over 750 seconds under the previous complex-state MILP regime, whereas the discrete XOR formulation resolved it natively in under 56 seconds.

Most significantly, the 9-gate Sayem circuit, which timed out after exceeding 24 hours of execution using continuous variable tracking, was successfully synthesized by the discrete integer framework without memory exhaustion. In traditional MILP formulations utilizing either $2^n \times 2^n$ unitary matrix embeddings or continuous complex state variables, synthesizing a depth-9 circuit invariably results in out-of-memory (OOM) failures or indefinite solver stalling due to exponential fractional branching. By compressing the synthesis space to a discrete 2-bit scalar evaluation ($\mathcal{O}(m \cdot n \cdot 2^n)$), our method drastically bounded the search tree, permitting CPLEX to complete the 9-depth evaluation organically.

Recent state-of-the-art frameworks, such as the MIP encodings proposed by Nagarajan and Szabó [4], have demonstrated remarkable success in scaling exact quantum synthesis. By employing domain-specific valid inequalities—such as commuting-gate pruning, identity ordering, and structural symmetry breaking—their framework heavily accelerates the branch-and-bound process, achieving highly competitive execution times for universal gate sets (e.g., optimizing T-count in Clifford+T architectures).

Our methodology takes a fundamentally different, representational approach. By restricting the synthesis explicitly to the NCV library, we exploit its unique 4-valued cyclic isomorphism. Instead of optimizing a generalized quantum state vector, we compress the entire state evolution into a discrete 2-bit XOR polytope. This representational shift achieves immense baseline acceleration natively. While generalized frameworks rely on sophisticated heuristic tree-pruning to achieve reasonable execution times for 3-qubit benchmarks, our discrete algebraic codification resolves similar benchmarks (such as the Toffoli and Fredkin gates) in sub-second to single-digit seconds strictly through tight algebraic constraints. Furthermore, the representational efficiency of our model is fully orthogonal to structural pruning; future integrations of our discrete NCV state-packing with the structural redundancy cuts proposed in [4] could push the boundaries of exact synthesis even further.

VIII. Conclusion

In this paper, we introduced an optimized mathematical framework for the exact synthesis of NCV quantum circuits. By formally proving that the quantum mechanics of the NCV gate library acting on basis states is isomorphic to a 4-valued cyclic discrete group, we codified quantum amplitudes and phases into a single scalar representation.

The application of a binary algebraic decomposition ($2b_1 + b_0$) paired with an exact XOR continuous polytope allows complex quantum operations to be mapped strictly to linear mixed-integer programs. Through rigorous algebraic verification, we proved that this purely discrete approach perfectly mirrors the continuous-variable domain. Compared to our previous complex-variable MILP baseline [6], the proposed discrete MILP formulation completely eliminates fractional arithmetic, matrix embeddings, and mathematically loose relaxations, providing a highly scalable and mathematically rigorous foundation for optimal quantum circuit compilation.

References

[1] D. Maslov, G. W. Dueck, and D. M. Miller, "Techniques for the synthesis of reversible Toffoli networks," *ACM Trans. Des. Autom. Electron. Syst.*, vol. 8, no. 3, pp. 322–340, 2003.

[2] D. Große, R. Wille, G. W. Dueck, and R. Drechsler, "Exact synthesis of reversible logic circuits," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 28, no. 6, pp. 803–814, 2009.

[3] Z. Li, S. Chen, X. Song, and W. Zhu, "Efficient algorithm for synthesis of optimal NCV 3-qubit reversible circuits using new quantum logic gate library," *Quantum Inf. Process.*, vol. 15, pp. 5023–5043, 2016.

[4] H. Nagarajan and Z. Szabó, "Provably Optimal Quantum Circuits with Mixed-Integer Programming," *arXiv preprint*, arXiv:2510.00649, 2025.

[5] G. Meuli, M. Soeken, M. Roetteler, and G. De Micheli, "Exact synthesis of multiqubit Clifford+T circuits," in *2019 IEEE/ACM Int. Conf. on Computer-Aided Design (ICCAD)*, 2019.

[6] G. Papakonstantinou, "Systematic Synthesis and Optimization of Reversible Quantum Circuits via MINLP, Toffoli Permutation, and Local Search," *Quantum Rep.* 2026, 8, 79. https://doi.org/10.3390/quantum8030079

[7] G. Papakonstantinou, "Optimal Design of Reversible Circuits in the Strict NCT Library," *TechRxiv Preprint*, DOI: 10.36227/techrxiv.176827280.05224615/v2, 2026.

Appendix A
Synthesized Optimal Circuits

The optimal topological synthesis of the examined benchmark gates is provided in Figure 1. All circuits were generated natively using the discrete MILP framework without continuous variable branch-and-bound optimization.