Towards a Hierarchical Quantum Circuit Language

WILLIAM SCHOBER, Università della Svizzera italiana, Switzerland
SCOTT WESLEY, Dalhousie University, Canada

Many circuit-based quantum programming languages contain limited functionality for handling programs with quantum branching events. In this paper we introduce a hierarchical circuit language with two new connectives, one for generalized controlled operations and one for sub-circuit exponentiation, to reason formally about such programs. We construct the hierarchical circuit language using a novel category-theoretic construction, demonstrate that its compositional properties are non-trivial, show a circuit identity which may be useful for circuit optimization by a compiler, demonstrate how generalized controlled operations describe quantum control flow, and then contrast generalized controlled operations with Pauli rotations.

ACM Reference Format:
William Schober and Scott Wesley. 2025. Towards a Hierarchical Quantum Circuit Language . 1, 1 (March 2025), 8 pages. https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

As quantum computing hardware continues to advance, so too does its software. While quantum circuits remain the de facto standard language for low-level quantum programming, they are cumbersome when designing large-scale programs with tens of thousands of gates. Higher-level programming abstractions of quantum circuits are clearly on the horizon. In particular, quantum control flow [19] has proven to be challenging to implement since the control qubits in a quantum conditional statement may be in superposition, and can therefore become entangled with the data register [21]. A number of existing programming languages contain classical conditional statements [7, 18], and some allow features like uncomputation [4, 9] or coherent control [2]. Recently, Heunen et al. introduced a language with a quantum branching primitive for constructing controlled unitaries [11]. In this paper we introduce a new hierarchical circuit language which generalizes several existing notions of controlled sub-circuits (e.g. the `if-let` statement of [11]) and unifies this with the sub-circuit exponentiation found in OpenQASM 3 [7]. The language builds upon a novel category-theoretic construction. We expect that the circuit abstractions it provides will prove useful for control flow and higher-level compiler optimization.

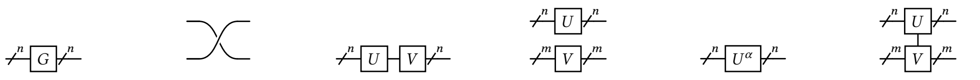
In Sec. 2 we formally introduce the hierarchical circuit language, as a universal algebraic (see, [1]) extension of PROP categories (see, [3, 6, 13]). We describe the inductive construction that is used to generate the hierarchical language via closures over an increasing family of nested sub-languages, and then provide its semantics and some of its non-trivial compositional properties. Then we give a concrete example of a gate set for hierarchical circuits that is universal for quantum computation. In Sec. 3 we outline some properties of hierarchical circuits which may be useful for control flow or higher-level circuit optimization. We contrast this new abstraction with Pauli rotations, a common gate abstraction whose semantics also involve exponentiation.

2 Hierarchical Quantum Circuits

The hierarchical language introduced in this section builds upon the PROP formalism for quantum circuits. Recall that in the PROP formalism, a family of quantum circuits is defined with respect to a gate set $\mathcal{G}$. Each gate $G \in \mathcal{G}$ has an associated signature $G : n \to n$ which states that the gate G

Authors' Contact Information: William Schober, Università della Svizzera italiana, Lugano, Ticino, Switzerland; Scott Wesley, Dalhousie University, Halifax, Nova Scotia, Canada.

2025. ACM XXXX-XXXX/2025/3-ART
https://doi.org/XXXXXXX.XXXXXXX



(a) $G : n \longrightarrow n$. (b) $\beta : 2 \longrightarrow 2$. (c) $V \circ U$. (d) $U \boxtimes V$. (e) $U \uparrow \alpha$. (f) $U \odot V$.

Fig. 1. The graphical language for hierarchical quantum circuits.

has n input wires and n output wires. Gates can be composed sequentially and in parallel, using the sequential composition operator $(\circ)$ and the parallel composition operator $(\boxtimes)$. Given $n \in \mathbb{N}$, we write $\mathrm{Prop}(\mathcal{G})(n)$ for the collection of all circuits expressible over $\mathcal{G}$ using the empty wire which is denoted 1, the wire crossing (β), sequential composition, and parallel composition. Formally, $\mathrm{Prop}(\mathcal{G})$ is the free PROP category generated by $\mathcal{G}$ which corresponds to the least fixed-point of the following inductive definition where $(\circ)$ and $(\boxtimes)$ are viewed as syntactic connectives.

- $1 \in \mathrm{Prop}(\mathcal{G})(1)$ and $\beta \in \mathrm{Prop}(\mathcal{G})(2)$.
- If $G \in \mathcal{G}$ and $G : n \longrightarrow n$ then $G \in \mathrm{Prop}(\mathcal{G})(n)$.
- If $U \in \mathrm{Prop}(\mathcal{G})(n)$ and $V \in \mathrm{Prop}(\mathcal{G})(n)$, then $V \circ U \in \mathrm{Prop}(\mathcal{G})(n)$.
- If $U \in \mathrm{Prop}(\mathcal{G})(n)$ and $V \in \mathrm{Prop}(\mathcal{G})(m)$ then $U \boxtimes V \in \mathrm{Prop}(\mathcal{G})(n+m)$.

The hierarchical circuit language introduces two new connectives, $(\uparrow)$ and $(\odot)$, which correspond to sub-circuit exponentiation and generalized controls, respectively. The $(\uparrow)$ connective takes a circuit $U : n \longrightarrow n$ together with a real number $\alpha \in \mathbb{R}$, and then return a circuit $(U \uparrow \alpha) : n \longrightarrow n$ with the intended interpretation that U is exponentiated by α. The $(\odot)$ connective takes two circuits $U : n \longrightarrow n$ and $V : m \longrightarrow m$, and then return a new circuit $U \odot V : (n+m) \longrightarrow (n+m)$ with the intended interpretation that V is controlled by U. The graphical language is depicted in Fig. 1.

Each circuit in the hierarchical language has an associated level, which captures the nesting depth of these new connectives. The level of a circuit is important, both from the point-of-view of compilation and relational theories. To encode the notion of levels, the hierarchical circuit language is defined inductively, starting from a primitive gate set $\mathcal{G}$ and the language of level zero circuits, denoted $C_0 = \mathrm{Prop}(\mathcal{G})$. The first step is to introduce powers of each circuit. Formally, for each collection of circuits C define the set $\mathrm{Exp}(C)$ of circuit exponentials as follows.

$$\mathrm{Exp}(C) = \{(U \uparrow \alpha) : n \longrightarrow n \mid U \in C(n) \text{ and } \alpha \in \mathbb{R}\}$$

Then $\mathcal{G}_0^{exp} = \mathrm{Exp}(\mathcal{G}) \cup \mathcal{G}$ and $C_0^{exp} = \mathrm{Prop}\left(\mathcal{G}_0^{exp}\right)$. The second step is to close the gate set under generalized controls. Formally, $\mathrm{Ctrl}(C)$ is the smallest gate set such that if U is in $(C \cup \mathrm{Ctrl}(C))(n)$ and V is in $(C \cup \mathrm{Ctrl}(C))(m)$, then $U \odot V$ is in $\mathrm{Ctrl}(C)(n+m)$. Then $\mathcal{G}_1 = \mathrm{Ctrl}\left(\mathcal{G}_0^{exp}\right) \cup \mathcal{G}_0^{exp}$ and $C_1 = \mathrm{Prop}\left(\mathcal{G}_1\right)$. The construction then proceeds as follows for $n \geq 1$.

- $\mathcal{G}_n^{exp} = \mathrm{Exp}\left(C_n\right) \cup \mathcal{G}_n$ and $C_n^{exp} = \mathrm{Prop}\left(\mathcal{G}_n^{exp}\right)$.
- $\mathcal{G}_{n+1} = \mathrm{Ctrl}\left(\mathcal{G}_n^{exp}\right) \cup \mathcal{G}_n^{exp}$ and $C_{n+1} = \mathrm{Prop}\left(\mathcal{G}_{n+1}\right)$.

This gives rise to the following inclusion of sub-languages.

$$\begin{tikzcd}
\mathcal{G} \arrow[r, hook] \arrow[d, hook] & \mathcal{G}_0^{exp} \arrow[r, hook] \arrow[d, hook] & \mathcal{G}_1 \arrow[r, hook] \arrow[d, hook] & \mathcal{G}_1^{exp} \arrow[r, hook] \arrow[d, hook] & \mathcal{G}_2 \arrow[r, hook] \arrow[d, hook] & \mathcal{G}_2^{exp} \arrow[r, hook] \arrow[d, hook] & \mathcal{G}_3 \arrow[r, hook] \arrow[d, hook] & \cdots \\
C_0 \arrow[r, hook] & C_0^{exp} \arrow[r, hook] & C_1 \arrow[r, hook] & C_1^{exp} \arrow[r, hook] & C_2 \arrow[r, hook] & C_2^{exp} \arrow[r, hook] & C_3 \arrow[r, hook] & \cdots
\end{tikzcd}$$

The *hierarchical circuit language* obtained from $\mathcal{G}$ is then defined to be $\mathrm{LvProp}(\mathcal{G}) = \bigcup_{n=1}^{\infty} C_n$. Given a circuit $U \in \mathrm{LvProp}(\mathcal{G}_0)(n)$, the *level* U denoted $\mathrm{lv}(U)$ is defined to be the least integer ℓ such that $U \in C_\ell(n)$. If $\mathrm{lv}(U) = \ell$, then U contains a gate built from ℓ layers of nested circuits.

The matrix semantics for a circuit language $C = \mathrm{Prop}(\mathcal{G})$ are defined by a mapping F from $\mathcal{G}$ to unitary matrices such that if $G : n \longrightarrow n$ then $F(G)$ is a $(2^n) \times (2^n)$ matrix. The mapping F then extends inductively to a function from circuits in C to matrices as follows.

- If $G \in \mathcal{G}$, then $[\![G]\!] = F(G)$.

- If $U \in C(n)$ and $V \in C(n)$, then $[\![V \circ U]\!] = [\![V]\!][\![U]\!]$.
- If $U \in C(n)$ and $V \in C(m)$, then $[\![U \boxtimes V]\!] = [\![U]\!] \otimes [\![V]\!]$.

The remainder of this section will explore how $[\![-]\!]$ extends to $\mathrm{LvProp}(\mathcal{G})$.

2.1 The semantics of sub-circuit exponentiation

Recall that $\exp(M) := \sum_{n=0}^{\infty} \frac{M^n}{n!}$. For each unitary matrix U, there exists a unique skew-Hermitian matrix M with spectrum in $i(-\pi, \pi]$ such that $\exp(M) = U$. This defines a branch cut $\mathrm{Log}(-)$ of $\exp^{-1}(-)$ when restricted to unitary matrices. If U is a unitary matrix, and $\alpha \in \mathbb{R}$, then U^α is defined to be $\exp(\mathrm{Log}(U)\alpha)$ as in [7]. Consequently, $[\![U \uparrow \alpha]\!] := [\![U]\!]^\alpha = \exp(\mathrm{Log}(U)\alpha)$.

Unfortunately, $(\uparrow)$ is not an action by $(\mathbb{R}, \cdot\,)$ with respect to the semantics of $[\![-]\!]$. This means that there exists a $U \in C(n)$ and $\alpha, \beta \in \mathbb{R}$ such that $[\![(U \uparrow \alpha) \uparrow \beta]\!] \neq [\![U \uparrow (\alpha\beta)]\!]$. In terms of compilation, this means that rewriting a term of the form $(U \uparrow \alpha) \uparrow \beta$ to $U \uparrow (\alpha\beta)$ is not a sound optimization. For example, consider the case where $[\![U]\!] = Z$. Since Z is diagonal, then the following equations hold.

$$M = \mathrm{Log}[\![U]\!] = \begin{bmatrix} \mathrm{Log}(1) & 0 \\ 0 & \mathrm{Log}(-1) \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & i\pi \end{bmatrix} \qquad [\![U \uparrow \alpha]\!] = \exp(M\alpha) = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\alpha\pi} \end{bmatrix}$$

It can then be asked whether $[\![(U \uparrow \alpha) \uparrow \beta]\!] = [\![U \uparrow (\alpha\beta)]\!]$. This requires solving for $\mathrm{Log}[\![U \uparrow \alpha]\!]$.

- If $\alpha \in (-1, 1]$, then $\alpha\pi \in (-\pi, \pi]$, and consequently $\mathrm{Log}[\![U \uparrow \alpha]\!] = \alpha M$. It then follows by definition that $[\![(U \uparrow \alpha) \uparrow \beta]\!] = \exp(M(\alpha\beta)) = [\![U \uparrow (\alpha\beta)]\!]$.
- If $\alpha \notin (-1, 1]$, then $\mathrm{Log}[\![U \uparrow \alpha]\!] = \overline{\alpha} M$ where $\overline{\alpha}$ is the remainder of α modulo π. Then $[\![(U \uparrow \alpha) \uparrow \beta]\!] = [\![U \uparrow (\alpha\beta)]\!]$ if and only if $\exp(M(\overline{\alpha}\beta)) = \exp(M(\alpha\beta))$, which means that $\overline{\alpha}\beta$ and $\alpha\beta$ differ by an integer multiple of 2π. It follows by a short calculation that this happens if and only if β is a rational number whose denominator divides $(\alpha - \overline{\alpha})$.

This generalizes to diagonal matrices with one or more eigenvalues λ outside $\{0, 1\}$, by replacing α with $\lambda\alpha$ in each step of the argument. The argument then generalizes to non-diagonal Hermitian matrices M by first diagonalizing M as $P^\dagger DP$, and then using the fact that $\exp(iM\alpha) = P^\dagger \exp(iD\alpha)P$. In Theorem 2.1, the most general case is handled. Since the restriction to $(-1, 1]$ does define a monoid action, then exponentiation can be used to define the roots of circuits and their inverses.

Theorem 2.1. *Let $U \in C(n)$ with $\{\lambda_j\}_{j=1}^{2^n}$ the eigenvalues of $\mathrm{Log}([\![U]\!])$. For each $\alpha \in \mathbb{R}$ and $\beta \in \mathbb{R}$, $[\![(U \uparrow \alpha) \uparrow \beta]\!] = [\![U \uparrow (\alpha\beta)]\!]$ if and only if for each $j \in [2^n]$ either $\lambda_j\alpha \in i(-\pi, \pi]$ or $\beta \in \mathbb{Q}$ with reduced denominator dividing $\lfloor \lambda_j\alpha \,/\, (2i\pi) \rfloor$. This means that $(-1, 1]$ is the largest sub-monoid of $(\mathbb{R}, \cdot)$ such that $(\uparrow)$ restricts to an action with respect to the semantics of $[\![-]\!]$.*

2.2 The semantics of generalized controls

Let $U \in C(n)$ and $V \in C(m)$. Then define $[\![U \odot V]\!] := \exp(M_U \otimes M_V / (i\pi))$ where $M_U = \mathrm{Log}[\![U]\!]$ and $M_V = \mathrm{Log}[\![V]\!]$. Since the spectra of M_U and M_V are in $i(-\pi, \pi]$, then the spectrum of $M_U \otimes M_V / (i\pi)$ is in $i(-\pi, \pi]$. This means that the $\odot$-product is associative with respect to $[\![-]\!]$. An n-fold $\odot$-product therefore takes the form $\left[\!\!\left[\bigodot_{j=1}^n U_j \right]\!\!\right] = \exp\left(\left(\bigotimes_{j=1}^n \mathrm{Log}[\![U_j]\!] \right) / (i\pi)^{n-1} \right) = \exp\left(i\pi \bigotimes_{j=1}^n \frac{\mathrm{Log}[\![U_j]\!]}{i\pi} \right)$.

If there exists some circuit $E \in C(0)$ such that $[\![E]\!] = -1$, then the $\odot$-product is also unital and can be related to sub-circuit exponentiation. Since $\mathrm{Log}([\![E]\!]) = \mathrm{Log}(-1) = \mathrm{Log}(e^{i\pi}) = i\pi$, then $[\![E \odot U]\!] = [\![U]\!] = [\![U \odot E]\!]$ for each circuit U in C. Moreover, if $\alpha \in (-1, 1]$, then $\mathrm{Log}([\![E \uparrow \alpha]\!]) = \alpha\pi$. It follows that $[\![(E \uparrow \alpha) \odot U]\!] = [\![U \uparrow \alpha]\!]$ on the domain where $(\uparrow)$ is a monoid action.

2.3 A concrete example

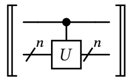
A surprising result is that the following hierarchical gate set is universal for quantum computation.

$$\mathcal{G} = \left\{ \;\bullet\;,\; -\!\!\bullet\!\!-,\; -\!\!\oplus\!\!- \right\} \qquad \text{where} \qquad [\![\bullet]\!] = -1 \qquad [\![-\!\!\bullet\!\!-]\!] = Z \qquad [\![-\!\!\oplus\!\!-]\!] = X.$$

This can be combined with sub-circuit exponentiation to recover global phase gates and rotations.

$$[\![\bullet^{\alpha}]\!] = e^{i\pi\alpha} \qquad \left[\!\!\left[-\!\bullet^{\alpha}\!- \right]\!\!\right] = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\pi\alpha} \end{bmatrix} \qquad \left[\!\!\left[-\!\oplus^{\alpha}\!- \right]\!\!\right] = \frac{1}{2}\begin{bmatrix} 1+e^{i\pi\alpha} & 1-e^{i\pi\alpha} \\ 1-e^{i\pi\alpha} & 1+e^{i\pi\alpha} \end{bmatrix}$$

Since the image of $[\![-]\!]$ contains all global phase gates, Z-rotations, and X-rotations, then LvProp($\mathcal{G}$) is universal for single-qubit quantum computation by [15]. The $\odot$-product strictly generalizes the notion of controlled sub-circuits $C(U)$ since,



$$= [\![-\!\bullet\!- \odot U]\!] = \exp\left(\frac{\mathrm{Log}(Z) \otimes \mathrm{Log}[\![U]\!]}{i\pi}\right) = \exp\left(\begin{bmatrix} 0 & 0 \\ 0 & \mathrm{Log}[\![U]\!] \end{bmatrix}\right) = \begin{bmatrix} I & 0 \\ 0 & [\![U]\!] \end{bmatrix}.$$

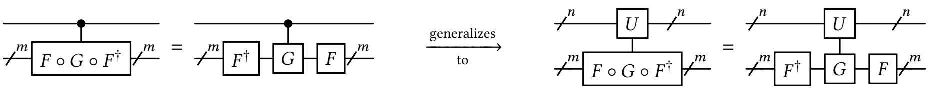
By setting $U = -\!\oplus\!-$, we find that the image of $[\![-]\!]$ contains the controlled-not gate. Therefore, LvProp($\mathcal{G}$) is universal for quantum computation by [15].

3 Applications to higher-level circuit optimization and control flow

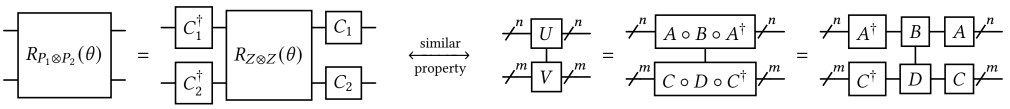
Hierarchical quantum circuits provide an abstract representation of standard quantum circuits with more structure, in the form of the ($\odot$) and ($\uparrow$) connectives. This additional structure may expose higher-level optimizations which could be hard to find after compilation to a hardware-specific low-level gate set. Additionally the ($\odot$) connective serves as a primitive for quantum control flow. In this section we discuss a property of hierarchical quantum circuit which may be useful for circuit optimization, and show how it relates to an existing primitive for quantum control flow. We then contrast hierarchical quantum circuits with Pauli rotations [10, 12], a similar but distinct notion.

3.1 A useful property of hierarchical circuits

A well-known circuit identity which has appeared in recent work on high-level programming language design [9], low-level circuit optimization [16], and circuit equational theories [8] is that $C(F \circ G \circ F^{\dagger}) = (I \boxtimes F) \circ C(G) \circ (I \boxtimes F^{\dagger})$ where F and G are unitary circuits and ($\dagger$) is complex conjugation. This equation generalizes to the $\odot$-product where the black control is replaced by any choice of unitary circuit U. The original circuit identity corresponds to the special case $U = -\!\bullet\!-$.



Since ($\odot$) is symmetric up to wire crossing, this property holds on both wire bundles simultaneously, meaning $(A \circ B \circ A^{\dagger}) \odot (C \circ D \circ C^{\dagger}) = (A \boxtimes C) \circ (B \odot D) \circ (A^{\dagger} \boxtimes C^{\dagger})$ for any unitary circuits A, B, C, and D. A similar property is also shared by Pauli rotations [12], which are introduced formally in the next section. Pauli rotations have the the property $R_{CPC^{\dagger}}(\theta) = CR_P(\theta)C^{\dagger}$ for any Clifford circuit C. In particular, if P_1 and P_2 are not already diagonal, then $R_{P_1 \otimes P_2}(\theta)$ can be diagonalized component-wise via single-qubit Cliffords satisfying $P_j = C_j Z C_j^{\dagger}$. Likewise, a generalized controlled operation $U \odot V$ can be diagonalized component-wise via unitary circuits of the correct arities (n and m respectively).



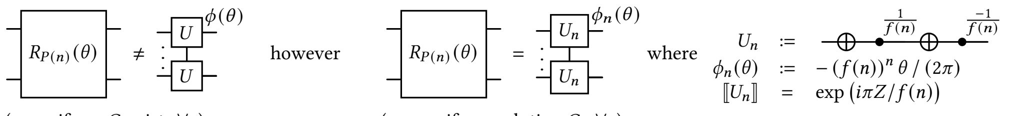
Given a choice of normal form for each $\odot$-free circuit, this yields a normal form for $U \odot V$ via the choice of diagonalization $B \odot D$. These normal forms can be seen as an exponentially long sequence of normal terms from [11]. More concretely, assume that U and V have eigenvectors $\{u_j\}_{j=1}^{2^n}$ and $\{v_j\}_{j=1}^{2^m}$ with respective eigenvalues $\{e^{i\pi\theta_j}\}_{j=1}^{2^n}$ and $\{e^{i\pi\rho_j}\}_{j=1}^{2^m}$. In the language of [11], the normal form of $U \odot V$ will have a term "`if let` $\left|u_j \otimes v_k\right\rangle$ `then Ph`$(\pi\theta_j\rho_k)$;" for each $j \in [2^n]$ and $k \in [2^m]$. This means $(\odot)$ is a primitive that describes quantum control flow by packaging together branching statements over the eigenbases of U and V.

3.2 Simulating Pauli rotations with hierarchical circuits

Pauli rotations [10, 12], sometimes called Pauli gadgets [17], are defined as $R_P(\theta) = \exp(-i\theta P/2)$ where $\theta \in \mathbb{R}$ and $P = \bigotimes_{j=1}^{n} P_j$ is an n-fold tensor product of Pauli matrices $P_j \in \{\pm I, \pm X, \pm Y, \pm Z\}$. Pauli rotations are widely used in circuit optimization [20], benchmarking [14], and chemistry applications [17].

Pauli rotations bear some semblance to the $(\odot)$ and $(\uparrow)$ connectives of hierarchical circuits, since all three notions have semantics of the form $\exp(-)$. In particular, if $P(n) = Z^{\otimes n}$, then it may appear possible that $R_{P(n)}(\theta) = [\![G(n,\theta)]\!]$ for some *control tower* $G(n,\theta) := \left(U^{\odot n}\right) \uparrow \phi(\theta)$ composed of n stacked copies of a fixed diagonal one-qubit circuit U, raised to an overall power $\phi(\theta)$ which encodes the rotation angle θ. However, no such $G(n,\theta)$ exists, as the sub-circuit U would need to satisfy $\mathrm{Log}[\![U]\!] = i\pi Z$, which violates the assumption that $\mathrm{Log}[\![U]\!]$ has eigenvalues in $i(-\pi, \pi]$. Nor does there exist a choice of k towers $G_1(n,\theta)$ through to $G_k(n,\theta)$ such that the circuit $U = G_k(n,\theta) \circ \cdots \circ G_1(n,\theta)$ of constant depth k satisfies $[\![U]\!] = R_{P(n)}(\theta)$ for all n and θ. To simulate a Pauli rotation using a circuit of control towers, the circuit depth k must depend on n. This is also the case when simulating a Pauli rotation using the Clifford+T gate set; each Pauli rotation must be compiled into a subcircuit whose depth depends on n.

However, the expressiveness of hierarchical circuits does allow for a Pauli rotation to be expressed as a single *non-uniform* control tower $G_f(n,\theta)$ where U_n and ϕ_n are allowed to depend on n via a function $f : \mathbb{N} \to (1, \infty)$, and $\mathrm{lv}(U_n) = 1$. The function f serves to rescale the eigenvalues of $\mathrm{Log}[\![U_n]\!]$ to lie in $i(-\pi, \pi]$, which can then be corrected for in ϕ_n.



$$\begin{aligned} U_n &:= \\ \phi_n(\theta) &:= -\,(f(n))^n\,\theta\,/\,(2\pi) \\ [\![U_n]\!] &= \exp\left(i\pi Z/f(n)\right) \end{aligned}$$

(no uniform G exists $\forall n$) (non-uniform solution G_f $\forall n$)

Two interesting cases are when $f(n)$ is constant and when $f(n)$ is the n-th root of a constant. For example, if $f(n) = 2$, then $\phi_n(\theta) = 2^n\theta/(2\pi)$ depends explicitly on the value of n, but $[\![U_n]\!] = iZ$ does not. In contrast, if $f(n) = \sqrt[n]{2}$, then $[\![U_n]\!] = \exp\left(i\pi Z/\sqrt[n]{2}\right) = \cos\left(\pi/\sqrt[n]{2}\right)I + i\sin\left(\pi/\sqrt[n]{2}\right)Z$ depends explicitly on the value of n, but $\phi_n(\theta) = -\theta/\pi$ does not.

4 Future Work

In this paper, we introduced a hierarchical quantum circuit language for reasoning about controls and exponentiation. It was shown that generalized controls are associative, whereas sub-circuit exponentiation is only a monoid action under special assumptions. A component-wise diagonalization was then introduced which generalizes that found in [8, 9], and is shared by Pauli rotations. This property also leads to a normal form which can be expressed in terms of the normal terms found in [11], showing that generalized controls are indeed an abstraction of quantum control flow. It was then shown how these new primitives differ from Pauli rotations, and how Pauli rotations can be compiled to $O(1)$-depth hierarchical circuits, provided that a non-uniform decomposition is allowed. In future work, we would like to explore the interaction of generalized controls with sub-circuit exponentiation, in the hopes of obtaining a equational theory which generalizes [8]. We are also interested to extend hierarchical circuits to non-unitary circuits involving measurement, initialization, and discarding as in [2, 5].

5 Proof that no constant-depth circuit of uniform control towers can simulate a Pauli rotation for all n

Let U_j be an arbitrary diagonal one-qubit circuit so that $[\![U_j]\!] = \mathrm{diag}(e^{i\pi\lambda_{j,0}}, e^{i\pi\lambda_{j,1}})$ and $\mathrm{Log}[\![U_j]\!]/(i\pi) = \mathrm{diag}(\lambda_{j,0}, \lambda_{j,1})$ with $\lambda_{j,0}, \lambda_{j,1} \in (-1, 1]$. Let $G_j(n, \theta) := \left(U_j^{\odot n}\right)\uparrow\phi_j(\theta)$ be the *uniform* control tower constructed from n stacked copies of U_j, with a power given by a real function $\phi_j(\theta)$, as above. Note that neither U_j nor ϕ_j can depend on n, since G_j is uniform. Consider the circuit $U = G_k \circ ... \circ G_1$ comprised of k control towers G_j composed in sequence, with $j \in [k]$, and $k \in \mathbb{N}$. The depth of U is the constant k.

Theorem 5.1. *There does not exist a constant depth k circuit U comprised of uniform control towers G_j that satisfies $[\![U]\!] = R_{P(n)}(\theta)$ for all n and θ.*

Proof. Assume there exists a U satisfying $[\![U]\!] = R_{P(n)}(\theta)$ for all n. U has semantic interpretation

$$[\![U]\!] = [\![G_k \circ ... \circ G_1]\!] = \prod_{j=1}^{k} [\![G_j]\!] = \prod_{j=1}^{k} [\![\left(U_j^{\odot n}\right)\uparrow\phi_j(\theta)]\!] \tag{1}$$

$$= \prod_{j=1}^{k} \exp(\phi_j(\theta)\,\mathrm{Log}(\exp(i\pi \bigotimes_{l=1}^{n} \frac{\mathrm{Log}[\![U_j]\!]}{i\pi}))) \tag{2}$$

$$= \prod_{j=1}^{k} \exp(i\pi\phi_j(\theta) \bigotimes_{l=1}^{n} \frac{\mathrm{Log}[\![U_j]\!]}{i\pi}) \tag{3}$$

Since U_j is diagonal for all j, it follows that $\bigotimes_{l=1}^{n} \mathrm{Log}[\![U_j]\!]/i\pi$ is also diagonal for all j. Therefore $\bigotimes_{l=1}^{n} \mathrm{Log}[\![U_j]\!]/i\pi$ commutes with $\bigotimes_{l=1}^{n} \mathrm{Log}[\![U_{j'}]\!]/i\pi$ for all j, j', and the Baker-Campbell-Hausdorff identity gives us

$$= \exp(i\pi \sum_{j=1}^{k} \phi_j(\theta) \bigotimes_{l=1}^{n} \frac{\mathrm{Log}[\![U_j]\!]}{i\pi}) \tag{4}$$

Now let $\frac{\mathrm{Log}[\![U_j]\!]}{i\pi} =: \begin{bmatrix} \lambda_{j,0} & 0 \\ 0 & \lambda_{j,1} \end{bmatrix} = \lambda_{j,0} |0\rangle \langle 0| + \lambda_{j,1} |1\rangle \langle 1|$.

$$= \exp(i\pi \sum_{j=1}^{k} \phi_j(\theta) \sum_{x\in\{0,1\}^n} \lambda_{j,0}^{n-|x|}\lambda_{j,1}^{|x|} |x\rangle \langle x|) \tag{5}$$

$$= \exp(\sum_{x\in\{0,1\}^n} i\pi \sum_{j=1}^{k} \phi_j(\theta)\lambda_{j,0}^{n-|x|}\lambda_{j,1}^{|x|} |x\rangle \langle x|) \tag{6}$$

$$= \sum_{x\in\{0,1\}^n} \left(e^{i\pi \sum_{j=1}^{k} \phi_j(\theta)\lambda_{j,0}^{n-|x|}\lambda_{j,1}^{|x|}}\right) |x\rangle \langle x| \tag{7}$$

A Pauli rotation has the form

$$R_{P(n)}(\theta) = \exp(\frac{-i\theta}{2} \bigotimes_{l=1}^{n} Z) = \exp(\sum_{x\in\{0,1\}^n} \frac{-i\theta}{2}(-1)^{|x|} |x\rangle \langle x|) \tag{8}$$

$$= \sum_{x\in\{0,1\}^n} \left(e^{\frac{-i\theta}{2}(-1)^{|x|}}\right) |x\rangle \langle x| \tag{9}$$

Since $[\![U]\!] = R_{P(n)}(\theta)$, and both matrices are now written in an explicitly diagonal form, they must be equal in every entry x. Now consider $n \longrightarrow \infty$. Since $\lambda_{j,0}$ and $\lambda_{j,1}$ were defined via the principal logarithm, we have $\lambda_{j,0}, \lambda_{j,1} \in (-1, 1]$. If $\lambda_{j,0} = \lambda_{j,1} = 1$ then $[\![G_j]\!] = e^{i\pi\phi_j(\theta)}I$ is just a global phase gate, and we pick another $j' \neq j$ that isn't a global phase gate. Note that there must be at least one G_j that isn't proportional to I, since if $G_j \propto I$ for all j, then $U \propto I$, which violates the assumption that $[\![U]\!] = R_{P(n)}(\theta)$. At least one of $\lambda_{j,0}, \lambda_{j,1} \neq 1$, let's say $\lambda_{j,1} \neq 1$ without loss of generality.

Consider the bottom-right entry where x is the all-1 string. Then $|x| = n$, and so $i\pi \sum_{j=1}^{k} \phi_j(\theta)\lambda_{j,0}^{n-|x|}\lambda_{j,1}^{|x|} \longrightarrow 0$ as $n \longrightarrow \infty$. But then

$$e^{i\pi \sum_{j=1}^{k} \phi_j(\theta)\lambda_{j,0}^{n-|x|}\lambda_{j,1}^{|x|}} = e^0 \neq e^{\frac{-i\theta}{2}(-1)^{|x|}} \tag{10}$$

unless $\theta = 0$. But then $[\![U]\!]$ cannot be equal to $R_{P(n)}(\theta)$ for all θ and n since they differ in this entry when $n \longrightarrow \infty$ and $\theta \neq 0$, a contradiction. If instead $\lambda_{j,0} \neq 1$, then consider the top-left entry where x is the all-0 string. Then $|x| = 0$, and so $i\pi \sum_{j=1}^{k} \phi_j(\theta)\lambda_{j,0}^{n-|x|}\lambda_{j,1}^{|x|} \longrightarrow 0$ as $n \longrightarrow \infty$, and again

$$[\![U]\!] = e^{i\pi \sum_{j=1}^{k} \phi_j(\theta)\lambda_{j,0}^{n-|x|}\lambda_{j,1}^{|x|}} = e^0 \neq e^{\frac{-i\theta}{2}(-1)^{|x|}} = R_{P(n)}(\theta) \tag{11}$$

unless $\theta = 0$, again a contradiction. In all cases we have $[\![U]\!] \neq R_{P(n)}(\theta)$ for at least some choices of n, θ, contradicting our premise.

□

References

[1] Franz Baader and Tobias Nipkow. 1998. *Term Rewriting and All That.* Cambridge University Press. doi:10.1017/CBO9781139172752

[2] Kathleen Barsse, Romain Péchoux, and Simon Perdrix. 2025. A Quantum Programming Language for Coherent Control. arXiv:2507.10466 [cs.LO] https://arxiv.org/abs/2507.10466

[3] Ville Bergholm and Jacob D Biamonte. 2011. Categorical quantum circuits. *Journal of Physics A: Mathematical and Theoretical* 44, 24 (May 2011), 245304. doi:10.1088/1751-8113/44/24/245304

[4] Benjamin Bichsel, Maximilian Baader, Timon Gehr, and Martin Vechev. 2020. Silq: a high-level quantum language with safe uncomputation and intuitive semantics. In *Proceedings of the 41st ACM SIGPLAN Conference on Programming Language Design and Implementation* (London, UK) *(PLDI 2020).* Association for Computing Machinery, New York, NY, USA, 286–300. doi:10.1145/3385412.3386007

[5] Alexandre Clément, Noé Delorme, and Simon Perdrix. 2024. Minimal Equational Theories for Quantum Circuits. arXiv:2311.07476 [quant-ph] https://arxiv.org/abs/2311.07476

[6] Alexandre Clément, Nicolas Heurtel, Shane Mansfield, Simon Perdrix, and Benoît Valiron. 2023. A Complete Equational Theory for Quantum Circuits. In *2023 38th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS).* IEEE, 1–13. doi:10.1109/lics56636.2023.10175801

[7] Andrew Cross, Ali Javadi-Abhari, Thomas Alexander, Niel De Beaudrap, Lev S. Bishop, Steven Heidel, Colm A. Ryan, Prasahnt Sivarajah, John Smolin, Jay M. Gambetta, and Blake R. Johnson. 2022. OpenQASM 3: A Broader and Deeper Quantum Assembly Language. *ACM Transactions on Quantum Computing* 3, 3 (Sept. 2022), 1–50. doi:10.1145/3505636

[8] Noé Delorme and Simon Perdrix. 2025. Diagrammatic Reasoning with Control as a Constructor, Applications to Quantum Circuits. arXiv:2508.21756 [quant-ph] https://arxiv.org/abs/2508.21756

[9] Peng Fu, Kohei Kishida, Neil J. Ross, and Peter Selinger. 2025. Proto-Quipper with Reversing and Control. *Electronic Proceedings in Theoretical Computer Science* 426 (Aug. 2025), 1–22. doi:10.4204/eptcs.426.1

[10] David Gosset, Vadym Kliuchnikov, Michele Mosca, and Vincent Russo. 2014. An algorithm for the T-count. *Quantum Info. Comput.* 14, 15–16 (Nov. 2014), 1261–1276.

[11] Chris Heunen, Louis Lemonnier, Christopher McNally, and Alex Rice. 2025. Quantum circuits are just a phase. arXiv:2507.11676 [cs.PL] https://arxiv.org/abs/2507.11676

[12] Daniel Litinski. 2019. A Game of Surface Codes: Large-Scale Quantum Computing with Lattice Surgery. *Quantum* 3 (March 2019), 128. doi:10.22331/q-2019-03-05-128

[13] Saunders MacLane. 1965. Categorical Algebra. *Bull. Amer. Math. Soc.* 70, 1 (Jan. 1965), 40–106.

[14] Yusei Mori, Hideaki Hakoshima, Kyohei Sudo, Toshio Mori, Kosuke Mitarai, and Keisuke Fujii. 2025. Quantum circuit unoptimization. *Physical Review Research* 7, 2 (May 2025). doi:10.1103/physrevresearch.7.023139

[15] Michael A. Nielsen and Isaac L. Chuang. 2011. *Quantum Computation and Quantum Information.* Cambridge University Press.

[16] Maxime Remaud and Vivien Vandaele. 2025. *Ancilla-Free Quantum Adder with Sublinear Depth.* Springer Nature Switzerland, 137–154. doi:10.1007/978-3-031-97063-4_11

[17] Seyon Sivarajah, Silas Dilkes, Alexander Cowtan, Will Simmons, Alec Edgington, and Ross Duncan. 2020. t|ket⟩: a retargetable compiler for NISQ devices. *Quantum Science and Technology* 6, 1 (Nov. 2020), 014003. doi:10.1088/2058-9565/ab8e92

[18] Krysta Svore, Alan Geller, Matthias Troyer, John Azariah, Christopher Granade, Bettina Heim, Vadym Kliuchnikov, Mariia Mykhailova, Andres Paz, and Martin Roetteler. 2018. Q#: Enabling Scalable Quantum Computing and Development with a High-level DSL. In *Proceedings of the Real World Domain Specific Languages Workshop 2018 (RWDSL2018).* ACM, 1–10. doi:10.1145/3183895.3183901

[19] Benoît Valiron. 2022. Semantics of quantum programming languages: Classical control, quantum control. *Journal of Logical and Algebraic Methods in Programming* 128 (2022), 100790. doi:10.1016/j.jlamp.2022.100790

[20] Vivien Vandaele. 2025. Lower T-count with faster algorithms. arXiv:2407.08695 [quant-ph] https://arxiv.org/abs/2407.08695

[21] Charles Yuan, Agnes Villanyi, and Michael Carbin. 2024. Quantum Control Machine: The Limits of Control Flow in Quantum Programming. *Proceedings of the ACM on Programming Languages* 8, OOPSLA1 (April 2024), 1–28. doi:10.1145/3649811