Translating Symmetric Pattern-Matching and Quantum Control into Executable Circuits on Ket

Flávio Borin Júnior and Juliana Kaizer Vizzotto

***Abstract*— High-level quantum programming languages provide expressive constructs such as symmetric pattern-matching and quantum control, but connecting these abstractions to concrete circuit representations is challenging. This work proposes a semantics-preserving translation from a reversible quantum language into quantum circuits representable in the Ket framework for quantum programming in Python. Categorical semantics serves as an intermediate layer, ensuring unitarity, reversibility, and compositionality. The intended result is a method that unites high-level abstractions with the practical possibility of compilation and execution in Ket.**

***Keywords*— Quantum Programming Language, Categorical Semantics, Quantum Circuits.**

I. Introduction

Quantum control is a central concept in quantum programming languages and semantics. It allows to formalizing how quantum operations can be composed and controlled within higher-level languages. Since the Peter Selinger's seminal work [1] on the semantics of quantum programming languages, quantum control provides a structured means to represent conditional quantum operations and the manipulation of quantum data while preserving unitarity and reversibility.

A typed reversible programming language was previously presented by Sabry [2]. More specificaly, the work proposes a typed reversible programming language extended to a quantum domain with linear combinations of terms, enabling quantum loops and recursion in a meaningful way. The work connects classical reversible computation principles with quantum control structures, providing an operational semantics inspired by algebraic lambda calculus, and exemplifies expressive unitary operations modeling.

The Ket platform [4] is a modern quantum programming framework designed to provide hybrid quantum-classical execution of quantum algorithms, and access to quantum hardware backends. We target Ket because it provides a high-level yet efficient abstraction for representing quantum circuits, supports modular circuit construction, and symbolic manipulation, making it an ideal environment to demonstrate the practical implementation of high-level quantum programming languages.

In this context, this extended abstract is an initial work that proposes an approach to translate the high-level quantum programming constructs introduced in [2] into quantum circuits that are compatible with the Ket framework.

Flávio Borin Júnior, UFSM, Santa Maria-RS, e-mail: fbjunior@inf.ufsm.br; Juliana Kaizer Vizzotto, Departamento de Linguagens e Sistemas de Computação, UFSM, Santa Maria-RS, e-mail: juvizzotto@inf.ufsm.br.

II. Metodology

The reversible quantum programming language considered builds on the foundations established that classical reversible computations are extended to encompass quantum superpositions and control flows with a strong semantic basis.

Our approach, inspired in the work [3], uses categorical semantics as an intermediate, mathematically rigorous representation between the syntactic structures of the high-level reversible quantum language and the concrete quantum circuits targeted for execution.

Following Abramsky and Coecke's categorical quantum mechanics framework [5], programs are interpreted as morphisms in a suitable category enriched with tensor products (to represent qubit composition) and biproducts (to express classical control and superpositions). This abstracts away low-level linear algebra into compositional structures amenable to formal reasoning.

The proposed translation is semantics-preserving, ensuring that the original quantum program's meaning is respected. Moreover, it is both modular and compositional, allowing program fragments to be translated incrementally while supporting structured reasoning about correctness and enabling systematic optimizations.

A. Syntax to Categorical Morphisms

The syntactic constructs such as symmetric pattern-matching, recursion (fixpoints), and linear combinations correspond to specific morphisms and their compositions in the chosen category. For instance, pattern-matching translates to morphisms that implement controlled decompositions of quantum states, while fixpoints correspond to categorical fixed points or traces over morphisms.

As an example, consider the controlled-NOT gate. Pattern-matching on the control qubit corresponds to splitting the tensor product over its first component, allowing different operations to be applied to the second qubit: the identity morphism for $|0\rangle$ and the Pauli-X morphism for $|1\rangle$. Recombining the branches produces a single morphism that preserves the monoidal composition and captures quantum control categorically.

$$\text{cnot} : \mathbb{B} \otimes \mathbb{B} \leftrightarrow \mathbb{B} \otimes \mathbb{B} = \begin{pmatrix} |0,x\rangle \leftrightarrow |0,x\rangle \\ |1,0\rangle \leftrightarrow |1,1\rangle \\ |1,1\rangle \leftrightarrow |1,0\rangle \end{pmatrix}$$

B. From Categorical Semantics to Circuits

The categorical morphisms have concrete realizations as quantum circuits in standard models like FHilb (finite-dimensional Hilbert spaces and linear maps). Utilizing this

correspondence, morphisms can be algorithmically compiled into circuit components—e.g., unitary gates and control structures—that Ket can represent and execute.

III. Mapping Symmetric Pattern Matching Syntax to Categorical Semantics

This section presents the categorical interpretation of the reversible language's typing judgments, establishing a formal bridge between the syntactic constructs and their semantic counterparts in a dagger symmetric monoidal category.

A. Typing Contexts

A *typing context* Δ is a finite set of variable-type assignments,

$$\Delta = \{x_1 : A_1, x_2 : A_2, \ldots, x_n : A_n\}$$

used to track variables and their associated types under a linear type discipline. Due to reversibility constraints, variables in Δ are treated linearly, meaning each variable is used exactly once.

Categorically, the context Δ is interpreted as the tensor product of the interpretations of its component types,

$$\Delta = A_1 \otimes A_2 \otimes \cdots \otimes A_n,$$

where each A_i is an object in a dagger symmetric monoidal category $\mathcal{C}$, often instantiated as the category of finite-dimensional Hilbert spaces.

B. Typing Judgments

Typing judgments of the form

$$\Delta \vdash_v v : A$$

express that value v has type A under context Δ. Categorically, these correspond to morphisms

$$\Delta \vdash_v v : A : \Delta \to A,$$

interpreting values as structured embeddings or constructors.

More central to reversible computation, typing judgments on terms

$$\Delta \vdash t : A \leftrightarrow B$$

denote reversible transformations between types A and B under context Δ. Such terms are interpreted by invertible morphisms in $\mathcal{C}$:

$$\Delta \vdash t : A \leftrightarrow B : \Delta \otimes A \to \Delta \otimes B.$$

These morphisms preserve the linear structure, reflecting that transformations are unitary and reversible.

C. Summary

Syntax	**Categorical Semantics**
Typing Context Δ	Tensor product $\Delta = \bigotimes_i A_i$
Value typing $\Delta \vdash_v v : A$	Morphism $\Delta \to A$
Term typing $\Delta \vdash t : A \leftrightarrow B$	Isomorphism $\Delta \otimes A \to \Delta \otimes B$

The idea in this interpretation is to formalize reversible computations as morphisms in $\mathcal{C}$, ensuring that program semantics are captured as unitary and invertible transformations amenable to both reasoning and quantum circuit realization.

IV. Mapping Categorical Semantics of Value Typing to Quantum Circuits

The categorical semantics of a value typing judgment

$$\Delta \vdash_v v : A,$$

is interpreted as a morphism in a dagger symmetric monoidal category,

$$\Delta \vdash_v v : A : \Delta \to A,$$

where Δ represents the tensor product of quantum registers corresponding to the typing context, and A the object representing the type of v.

In quantum circuit terms, this morphism corresponds to a *state preparation circuit*, which transforms the input qubits associated with Δ into the quantum state encoding the value v of type A.

For example, consider preparing the Boolean value $|1\rangle$ on a single qubit from the initial state $|0\rangle$. The corresponding quantum circuit is a single Pauli-X gate:

More complex values correspond to unitary circuits that prepare multi-qubit states via compositions of reversible gates, ensuring the operation is unitary and invertible in accordance with the reversible language semantics.

This mapping guarantees that value typing morphisms in the categorical semantics framework have concrete and physically realizable interpretations as quantum circuits.

V. Final Discussion

This proposal outlines a research direction that aims to connect high-level abstractions for quantum programming, grounded in categorical semantics, with their concrete realization as circuits. The central goal is to develop a semantics-preserving translation in which constructs such as pattern-matching and general recursion are systematically mapped to categorical morphisms and subsequently compiled into standard circuit models. Achieving this would highlight both the expressive power of categorical methods for reasoning about quantum computation and the practical feasibility of running the resulting programs through Ket. By pursuing this approach, we expect to open new opportunities for rigorous yet executable models of quantum programming, and contribute to the research area of semantics-driven compilation techniques.

References

[1] P. Selinger, "Towards a quantum programming language," *Mathematical Structures in Computer Science*, v. 14, n. 4, pp. 527–586, August 2004.

[2] A. Sabry, B. Valiron, and J. K. Vizzotto, "From Symmetric Pattern-Matching to Quantum Control," in *Foundations of Software Science and Computation Structures*, C. Baier and U. Dal Lago, Eds., Cham: Springer International Publishing, pp. 348–364, 2018.

[3] J. Grattage and T. Altenkirch, "A compiler for a functional quantum programming language," Manuscript, 2005.

[4] E. C. R. da Rosa e R. de Santiago, "Ket Quantum Programming," *J. Emerg. Technol. Comput. Syst.*, v. 18, n. 1, pp. 12:1–12:25, Out. 2021. Association for Computing Machinery.

[5] S. Abramsky and B. Coecke, "Categorical quantum mechanics," arXiv:0808.1023, 2008.