Graphical Language with Delayed Trace: Picturing Quantum Computing with Finite Memory

– Extended Abstract –

Full version accepted at LICS2021, available on arXiv https://arxiv.org/abs/2102.03133

Titouan Carette, Marc de Visme, Simon Perdrix
Université de Lorraine, CNRS, Inria, LORIA F 54000 Nancy, France

Abstract—**Graphical languages, like quantum circuits or ZX-calculus, have been successfully designed to represent (memoryless) quantum computations acting on a finite number of qubits. Meanwhile, delayed traces have been used as a graphical way to represent finite-memory computations on streams, in a classical setting (cartesian data types). We merge those two approaches and describe a general construction that extends any graphical language, equipped with a notion of discarding, to a graphical language of finite memory computations. In order to handle cases like the ZX-calculus, which is complete for post-selected quantum mechanics, we extend the delayed trace formalism beyond the causal case, refining the notion of causality for stream transformers. We design a stream semantics based on stateful morphism sequences and, under some assumptions, show universality and completeness results. Finally, we investigate the links of our framework with previous works on cartesian data types, signal flow graphs, and quantum channels with memories.**

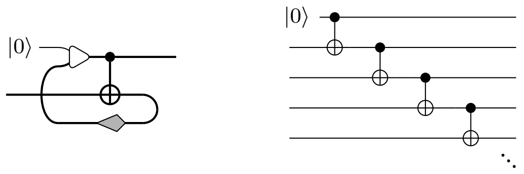


Fig. 1: **Left:** *A cascade of CNots on a stream of qubits. At the first tick, a CNot is applied: the control qubit is in the $|0\rangle$ state, the target qubit is the first qubit of the input stream. The control qubit is then output and the target qubit stored in the memory. At the second tick the stored qubit becomes the control qubit and the second qubit of the input stream becomes the target qubit and so on.* **Right:** *An informal unfolded version of the cascade of CNots.*

I. Introduction

Motivations. Several graphical languages have been successfully developed for representing finite dimension quantum processes. The quantum circuits and the ZX-calculus are the main examples of such graphical languages. The ZX-calculus is equipped with a complete equational theory [1], [2], that allows, among other applications, to perform circuit optimization [3], [4], and to design fault tolerant computations [5], [6]. These graphical languages have been designed for finite-dimension quantum mechanics: each wire represents a finite system – generally a qubit – as a consequence, a finite diagram can only represent a finite dimension quantum evolution. Notice that using the scalable construction [7], one can represent finite registers, with the possibility to split and merge registers. This construction makes the representation more compact but it remains a representation of finite dimension quantum computations. There is a fundamental reason for this restriction: finite dimension Hilbert spaces, contrary to infinite dimension ones, form a compact closed category, and the compact closure is the cornerstone of graphical languages like the ZX-calculus.

To go beyond finite registers, we explore in this paper the design of graphical languages for quantum stream transformations, *i.e.*, computations taking (infinite) sequences of quantum inputs to (infinite) sequences of quantum outputs. Intuitively a transformation acting on a stream of qubits, inputs a qubit and outputs a qubit at each clock tick. In order to allow interactions between systems input at distinct clock ticks, a memory mechanism is required to store some data across the ticks. Such a quantum transformation is called a *quantum channel with memory* in [8].

We choose to graphically represent the memory mechanism using *delayed traces*, *i.e.*, feedback loops that store qubits from clock tick to the next. The example consisting in applying a CNot gate on consecutive qubits of a stream is given in Fig. 1.

Delayed traces have been studied [9] as a construction which can be applied to any cartesian category. We explore the extension of this construction to the quantum case, since a quantum graphical language, from a categorical point of view, form a category which is symmetric monoidal but not cartesian.

Depicting finite memory quantum computations on streams does not provide a universal model of quantum computation. It is however an interesting fragment to explore, strictly more expressive than memory-free languages designed for finite registers, and an intermediate scale model potentially easier to implement using quantum technologies available in the short term, than a universal quantum computer.

Contributions. We introduce a general construction that extends any (not necessarily quantum) graphical language $\mathcal{G}$ equipped with a discarding map, to a graphical language

$\mathcal{G}^\omega$ with finite memory acting on streams. The construction consists in adding a delayed trace to model the memory, as well as stream constructors and destructors.

A key property of the construction is that the delay only commutes with causal transformations. Indeed, applying a non causal transformation before storing a system can produce some side effect on the output at tick k which would occur only at tick $k+1$ if the transformation is applied later on. Moreover, the infinite nature of the computation requires the introduction of a coinduction principle, to show for instance that storing a system in a memory forever and never using it, is equivalent to discard this system right away.

We introduce the *finite approximations* of a $\mathcal{G}^\omega$-diagram D as a sequence of $\mathcal{G}$-diagrams: the k^{th} diagram of the sequence represents the behavior of D from the initial to the k^{th} tick. The semantics of a $\mathcal{G}^\omega$-diagram is then defined as the sequence of interpretations of its finite approximations.

We show *universality* of the language by characterizing finite approximations that can be obtained by our language as being exactly those that are monotone and regular. Finally, we also show the *completeness* of the language, up to some additional assumptions which are satisfied in the quantum case.

II. Finite Memory Quantum Computing

In this section, we review various notions of quantum computing and motivate by examples the kind of computations the language presented in the next section is designed to represent.

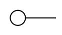
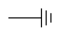
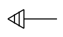
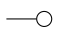
A. Completely Positives Maps

We use the density matrix formalism of finite dimensional quantum mechanics over qubits, see [16] for a more complete presentation. We have a symmetric monoidal category $\mathbf{CPM}_2$ of generalized quantum processes over qubits. The objects are the sets of linear operators of the form $\mathcal{M}_{2^n\times 2^n}(\mathbb{C})$ representing systems of n qubits. The morphisms are the linear maps that are completely positive. Among those maps only the trace preserving ones correspond to actual physical transformations, we write this subcategory $\mathbf{CPTP}_2$. The state of a n-qubit system is a **density matrix** *i.e.* a map $\mathbb{C}\to\mathcal{M}_{2^n\times 2^n}(\mathbb{C})$, which is positive semi-definite Hermitian and has unit trace.

Let the **discard** map be $\top \overset{\text{def}}{=} \rho\mapsto \mathrm{Tr}(\rho)$, which corresponds to measuring a qubit and forgetting the result. A quantum evolution f is **causal** if $\top\circ f = \top$, intuitively causal evolutions are side-effect free.

B. Quantum Gates

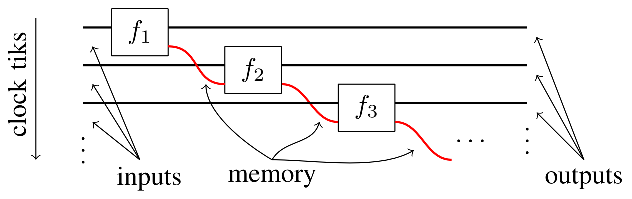
We represent the maps of $\mathbf{CPTP}_2$ as gates in circuits. The composition corresponds to plugging gates and the tensor product to putting them side by side. Note that usually quantum circuits cannot represent $\mathbf{CPM}_2$ in full generality, other graphical language like the ZX-calculus have been designed for this. We only present a few gates, taken both from the quantum circuits and ZX-calculus, that we will use in examples.

	$=$	$\lvert 0\rangle\langle 0\rvert$		$=$	$\rho\mapsto Tr(\rho)$
	$=$	$\frac{1}{2}\begin{pmatrix}1&0\\0&1\end{pmatrix}$		$=$	$\rho\mapsto\langle 0\rvert\rho\lvert 0\rangle$

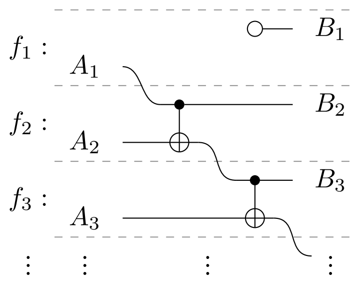
We are now ready to provide concrete examples of quantum computation with memory.

C. Quantum Computation with Memory

In order to go beyond quantum computation acting on a finite register of qubits, it is natural to consider streams of qubits: we consider a global clock, such that at each clock tick some qubits are input. For allowing interactions across clock ticks, like applying a CNot on two qubits input at distinct clock ticks, a memory mechanics is required to store a qubit and intuitively wait for another qubit to be available. Quantum channels with memory, introduced by Kretschmann and Werner [8], can be informally depicted as follows:[1]



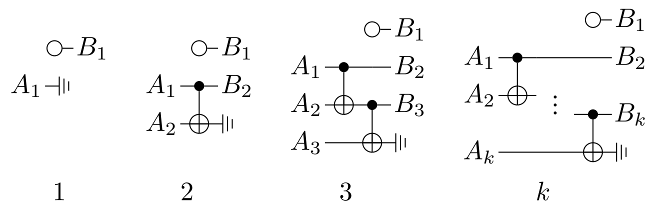
Thus the behavior of the computer at clock tick $k>0$ is a quantum process $f_k : A_k\otimes M_{k-1}\to B_k\otimes M_k$, with $M_0\overset{\text{def}}{=}\mathbb{C}$. Following the terminology for such processes in the classical case [9], we call such collection of processes a **stateful morphism sequence**. We give the example of a cascade of CNots gates (see Fig. 1). At first clock tick the memory is initialized with $\lvert 0\rangle\langle 0\rvert$. At each clock tick, a CNot is applied, the control qubit being the memory qubit and the target being the input qubit. Finally, the memory qubit is output and the input qubit is stored in the memory. The corresponding stateful morphism sequence is:



In practice, one cannot access the whole infinite computation at once, but only what has been computed up to some clock tick k. To stop the computation of a stateful morphism at clock tick k, we discard the memory system M_k and obtain, by plugging the memories, a process $\bigotimes_{i=1}^{k} A_i\to\bigotimes_{i=1}^{k} B_i$ called

[1] In [8], the authors mainly consider the case of a clock without initialization, *i.e.*, clock ticks in $\mathbb{Z}$ rather than $\mathbb{N}_{\geq 1}$

the **finite approximation** at clock tick k. For the cascade of CNot the sequence of finite approximations is:

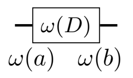
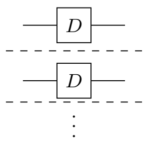
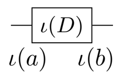
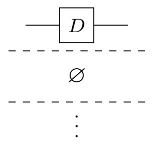


A stateful morphism sequence leads to a unique sequence of finite approximations. However, two different stateful morphism sequences can have the same sequence of finite approximations, they are then said **observationally equivalent**.

III. Our Graphical Language

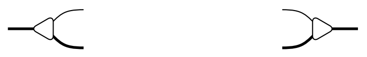
A. The Stream Type

We start with a graphical language $\mathcal{G}$, and we define a graphical language $\mathcal{G}^\omega$ describing computations of $\mathcal{G}$ with finite memory. We first add a temporal information; for every diagram $D \in \mathcal{G}(a, b)$, we have in $\mathcal{G}^\omega$ the two following diagrams:

Diagram	Stateful morphism sequence	Diagram	Stateful morphism sequence
			

Additionally, we have $\lozenge^n\omega(D) \in \mathcal{G}^\omega(\lozenge^n\omega(a), \lozenge^n\omega(b))$ and $\lozenge^n\iota(D) \in \mathcal{G}^\omega(\lozenge^n\iota(a), \lozenge^n\iota(b))$ (for $n > 0$) that do nothing for n ticks and then behave as $\omega(D)$ or $\iota(D)$ starting from the $(n+1)$-th tick of the clock rather than from the first one.

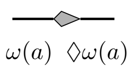
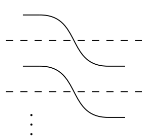
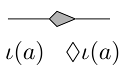
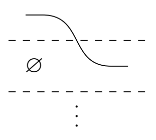
To link the types of the form $\omega(-)$ and $\iota(-)$, we have in $\mathcal{G}^\omega$ a natural isomorphism unfolding the first tick of a stream: $\lozenge^n\omega(a) \cong \lozenge^n\iota(a) \otimes \lozenge^{n+1}\omega(a)$. This isomorphism is graphically represented by



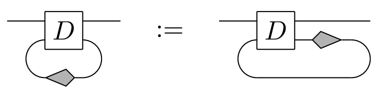
stream derivative **stream initialization**

B. The Delayed Trace

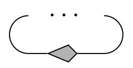
For every type $a \in \mathcal{G}^\omega$, we introduce a delay generator in $\mathcal{G}^\omega(a, \lozenge a)$ to represent memory cells storing information for one tick. For examples we have:

Diagram	Stateful morphism sequence	Diagram	Stateful morphism sequence
			

In the quantum case, using the compact closure of the ZX-calculus, we are able to define a delayed trace as follows:



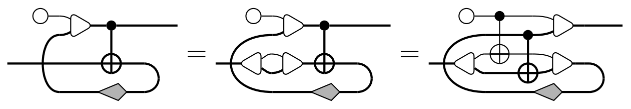
When the category $\mathcal{G}$ is not compact closed (or traced), we need to manually introduce the delayed trace as a generator:



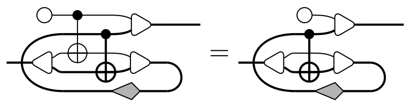
We detail in our paper all the equations that this trace-like primitive must satisfies, and its stateful morphism sequence.

C. Rewriting Rules in Action

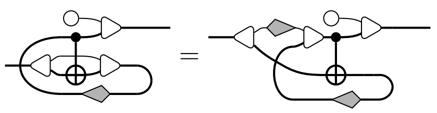
The rewriting rules of $\mathcal{G}^\omega$ are of four kinds. Firstly, the stream derivative and stream initialization form a natural isomorphism, which allow us to rewrite the cascade of CNot:



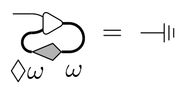
Then, we have the rewriting rules of our initial language $\mathcal{G}$:



Thirdly, we have the trace-like axioms that the delayed trace must satisfies, which allow us to complete the unfolding of the initial value $|0\rangle\langle 0|$ from the cascade of CNot:



Lastly, there is a co-induction rule which allows one to prove that storing a system in a memory forever and never using it, is equivalent to discard this system right away:



D. Soundness, Completeness and Universality

In our paper, we formalize how a semantics for our initial language $[\![-]\!] : \mathcal{G} \to \mathcal{C}$ can be used to build a semantics for our new graphical language $(\!|-|\!) : \mathcal{G}^\omega \to \mathrm{FinApp}(\mathcal{C})$, where $\mathrm{FinApp}(\mathcal{C})$ is the category of finite approximations satisfying the properties of monotonicity – which corresponds to the fact that one cannot change the past, and regularity – which corresponds to the fact that we use a finite diagram to represent an infinite sequence. Assuming $[\![-]\!]$ is itself sound, complete and universal, and furthermore that $\mathcal{C}$ satisfies some conditions, we then prove the following results:

Theorem 1 (Soundness)**.** *Given two diagrams $D, D' \in \mathcal{G}^\omega$, if D can be rewritten into D' then $(\!|D|\!) = (\!|D'|\!)$.*

Theorem 2 (Completeness)**.** *Given two diagrams $D, D' \in \mathcal{G}^\omega$, if $(\!|D|\!) = (\!|D'|\!)$ then D can be rewritten into D'.*

Theorem 3 (Universality)**.** *The functor $(\!|-|\!)$ is full.*

References

[1] R. Vilmart, "A near-optimal axiomatisation of ZX-calculus for pure qubit quantum mechanics," in *Proceedings of the 34th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, 2019. [Online]. Available: https://arxiv.org/abs/1812.09114

[2] T. Carette, E. Jeandel, S. Perdrix, and R. Vilmart, "Completeness of graphical languages for mixed states quantum mechanics," in *International Colloquium on Automata, Languages, and Programming (ICALP'19)*, 2019.

[3] R. Duncan, A. Kissinger, S. Perdrix, and J. van de Wetering, "Graph-theoretic Simplification of Quantum Circuits with the ZX-calculus," *Quantum*, vol. 4, p. 279, Jun. 2020. [Online]. Available: https://doi.org/10.22331/q-2020-06-04-279

[4] A. Kissinger and J. van de Wetering, "Reducing the number of non-Clifford gates in quantum circuits," *Physical Review A*, vol. 102, no. 2, p. 022406, 2020.

[5] M. Hanks, M. P. Estarellas, W. J. Munro, and K. Nemoto, "Effective compression of quantum braided circuits aided by ZX-calculus," *Phys. Rev. X*, vol. 10, p. 041030, Nov 2020. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevX.10.041030

[6] N. de Beaudrap and D. Horsman, "The ZX calculus is a language for surface code lattice surgery," *Quantum*, vol. 4, p. 218, Jan. 2020. [Online]. Available: https://doi.org/10.22331/q-2020-01-09-218

[7] T. Carette, D. Horsman, and S. Perdrix, "SZX-calculus: Scalable graphical quantum reasoning," in *MFCS 2019-44th International Symposium on Mathematical Foundations of Computer Science*, vol. 138, 2019, pp. 55–1.

[8] D. Kretschmann and R. F. Werner, "Quantum channels with memory," *Physical Review A*, vol. 72, no. 6, p. 062323, 2005.

[9] D. Sprunger and S.-y. Katsumata, "Differentiable causal computations via delayed trace," in *2019 34th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*. IEEE, 2019, pp. 1–12.

[10] A. Kissinger and D. Quick, "A First-order Logic for String Diagrams," in *6th Conference on Algebra and Coalgebra in Computer Science (CALCO 2015)*, ser. Leibniz International Proceedings in Informatics (LIPIcs), L. S. Moss and P. Sobocinski, Eds., vol. 35. Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2015, pp. 171–189. [Online]. Available: http://drops.dagstuhl.de/opus/volltexte/2015/5533

[11] S. Abramsky and C. Heunen, "H*-algebras and nonunital frobenius algebras: First steps in infinite-dimensional categorical quantum mechanics," in *Clifford Lectures, AMS Proceedings of Symposia in Applied Mathematics*, vol. 71, 2012.

[12] B. Coecke and C. Heunen, "Pictures of complete positivity in arbitrary dimension," *Information and Computation*, vol. 250, pp. 50–58, 2016.

[13] S. Gogioso and F. Genovese, "Infinite-dimensional categorical quantum mechanics," in Proceedings 13th International Conference on *Quantum Physics and Logic,* Glasgow, Scotland, 6-10 June 2016, ser. Electronic Proceedings in Theoretical Computer Science, R. Duncan and C. Heunen, Eds., vol. 236. Open Publishing Association, 2017, pp. 51–69.

[14] B. Schumacher and R. F. Werner, "Reversible quantum cellular automata," *arXiv preprint quant-ph/0405174*, 2004.

[15] P. Arrighi, V. Nesme, and R. Werner, "Unitarity plus causality implies localizability," *Journal of Computer and System Sciences*, vol. 77, no. 2, pp. 372–378, 2011.

[16] M. A. Nielsen and I. Chuang, "Quantum computation and quantum information," 2002.

[17] W. F. Stinespring, "Positive functions on C*-algebras," *Proceedings of the American Mathematical Society*, vol. 6, no. 2, pp. 211–216, 1955.

[18] A. Kissinger and S. Uijlen, "A categorical semantics for causal structure," in *2017 32nd Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*. IEEE, 2017, pp. 1–12.

[19] M. Backens, "The ZX-calculus is complete for stabilizer quantum mechanics," *New Journal of Physics*, vol. 16, no. 9, p. 093021, sep 2014. [Online]. Available: https://doi.org/10.1088\%2F1367-2630\%2F16\%2F9\%2F093021

[20] E. Jeandel, S. Perdrix, and R. Vilmart, "A complete axiomatisation of the ZX-calculus for Clifford+T quantum mechanics," in *Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*. ACM, 2018, pp. 559–568.

[21] A. Hadzihasanovic, K. F. Ng, and Q. Wang, "Two complete axiomatisations of pure-state qubit quantum computing," in *Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science*, ser. LICS '18. New York, NY, USA: ACM, 2018, pp. 502–511. [Online]. Available: http://doi.acm.org/10.1145/3209108.3209128

[22] F. Bonchi, P. Sobociński, and F. Zanasi, "A categorical semantics of signal flow graphs," in *International Conference on Concurrency Theory*. Springer, 2014, pp. 435–450.

[23] D. R. Ghica, A. Jung, and A. Lopez, "Diagrammatic semantics for digital circuits," in *26th EACSL Annual Conference on Computer Science Logic (CSL 2017)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2017.

[24] M. Román, "Comb diagrams for discrete-time feedback," *arXiv preprint arXiv:2003.06214*, 2020.

[25] B. Coecke and R. Duncan, "Interacting quantum observables: categorical algebra and diagrammatics," *New Journal of Physics*, vol. 13, no. 4, p. 043016, 2011.

[26] E. Jeandel, S. Perdrix, and R. Vilmart, "Diagrammatic reasoning beyond clifford+ t quantum mechanics," in *Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*. ACM, 2018, pp. 569–578.

[27] F. Zanasi, "Interacting hopf algebras: the theory of linear systems," *arXiv preprint arXiv:1805.03032*, 2018.