NExt ApplicationS of Quantum Computing

D5.6: Diagrammatical languages to represent Hamiltonians and how to compute the exponential of Hamiltonians

Document Properties

Contract Number	951821
Contractual Deadline	30-06-2023
Dissemination Level	Public
Nature	Report
Editors	Emmanuel Jeandel, UL
Authors	Emmanuel Jeandel, UL
Reviewers	Vedran Dunjko, U. Leiden Vicente Moret Bonillo , U. Coruña
Date	14-06-2023
Keywords	QAOA, Hamiltonians, ZX-Calculus
Status	Under Review
Release	0.2

This project has received funding from the European Union's Horizon 2020 research and innovation programme under Grant Agreement No. 951821

History of Changes

Release	Date	Author, Organisation	Description of Changes
0.1	31/05/2023	E Jeandel, UL	First Version
0.2	14/06/2023	E.Jeandel, UL	Reviewers comments taken into account

Table of Contents

1. Executive Summary 4

2. Introduction 5

3. Definitions 6
3.1. Diagrammatic Languages . . . 6
3.2. The ZX-Calculus . . . 7
3.2.1. Definitions . . . 7
3.2.2. Examples . . . 8

4. A language to represent Hamiltonians 10
4.1. A Pro for Ising Hamiltonians . . . 10
4.1.1. Discussion . . . 10
4.1.2. Definition . . . 10
4.1.3. Example . . . 11
4.2. A Prop for Ising Hamiltonians . . . 11
4.2.1. Definition . . . 11
4.2.2. Definition . . . 12
4.2.3. Examples . . . 13
4.2.4. Notes . . . 14

5. Representing Hamiltonians and their exponential in the ZX-Calculus 15
5.1. Computing the exponential . . . 15
5.2. Hamiltonians in the ZX-Calculus . . . 17
5.3. An example . . . 18

6. Conclusions 19

List of Tables 20

Bibliography 21

A. Appendix 22
A.1. Compatibility of the two compositions . . . 22

1.Executive Summary

This deliverable provides a graphical representation of Ising Hamiltonians, with the explicit goal of using this representation to analyse later the QAOA algorithm. This analysis entails computations with both the Hamiltonian and its exponential.

To reach this goal, a language should be provided where both a matrix and its exponential can be described using the same diagrammatic formalism.

This deliverable achieves this goal by first building a specific graphical language, called UH_D, to represent Hermitian matrices, then provide an algorithm to compute from this representation a representation in the ZX-calculus of both this matrix and its exponential.

Translations from the Hamiltonian to its exponential are streamlined as much as possible by use of monoidal functors. This implies the exponential is constructed brick by brick by putting together representation of each part of the Hamiltonian.

An example of such a translation is provided.

2.Introduction

The ZX-Calculus, introduced in (Coecke & Duncan, 2011), is a graphical language that represents quantum evolutions by *diagrams*. Each diagram corresponds to a linear transformation between Hilbert spaces. Compared to other graphical languages for quantum computation, e.g. circuits, the ZX-calculus comes with a set of rules, such that two diagrams that represent the same evolution can be transformed from one to the other using these rules. This implies in particular that it is possible to reason entirely diagrammatically about quantum computing using the ZX-Calculus.

This has been applied successfully in various contexts, from measurement-based quantum computing (MBQC) (Duncan & Perdrix, 2010; Kissinger & van de Wetering, 2019) to circuit optimization (Cowtan et al., 2020; de Beaudrap et al., 2020; Duncan et al., 2020; Kissinger & van de Wetering, 2020), and error-correction codes (Garvie & Duncan, 2018).

In this deliverable we set the stage for an application of the ZX-Calculus to the Quantum Approximate Optimization Algorithm (QAOA) (Farhi et al., 2014).

QAOA is a general method to find approximate solutions to combinatorial problems. Given a function $f : \{0,1\}^n \to \mathbb{R}$ we want to optimize, we represent the function by a diagonal Hamiltonian[1] H s.t. $\langle x| H |x\rangle = f(x)$. In this way, the problem becomes finding the smallest (or largest) eigenvalue of H. A circuit $C(\alpha_1, \ldots \alpha_k)$ is then built from $e^{i\alpha_1 H} \ldots e^{i\alpha_k H}$ and other generic Hamiltonians and optimized in such a way that the output of C is closest to the optimal eigenvector.

The analysis of such an algorithm relies therefore in computing $\langle \psi| H |\psi\rangle$ for a vector $|\psi\rangle$ obtained from a circuit using exponentials in H. To understand diagrammatically such a circuit, we need a language to represent both a Hamiltonian and its exponential simultaneously.

We provide such a construction in this deliverable.

More precisely, we provide in the next section a language, called UH_D, that can represent Diagonal Hamiltonians. We then give in the next section an algorithm, starting from the diagram, to represent both the matrix and its exponential in the ZX-Calculus. An example is given showing how the algorithm works.

Related works

The recent article (Carette et al., 2023) describes a language to represent Hermitian-preserving superoperators. The maps $0 \to n$ of this language (i.e. diagrams with no inputs) are therefore representations of Hamiltonians of size 2^n. This language uses the ZW-Calculus as a foundation instead of the ZX-Calculus, and it is not clear in full generality how to use this language to compute the exponential of an Hamiltonian, even it if it is diagonal.

Up to our knowledge, the only article trying to compute diagrammatically the exponential of an Hamiltonian is (Shaikh et al., 2022). However their construction relies heavily on the Cayley-Hamilton theorem and therefore works only if we know the characteristic polynomial of the Hamiltonian, or its eigenvalues. We don't know of any context in which their construction can therefore be used.

In this work, we will obtain directly diagrams of the ZX-calculus for an Ising Hamiltonian matrix from its representation. Previous works (Jeandel et al., 2022; Wang & Yeung, 2022) have suggested how to *differentiate* diagrams. This can be used to obtain a diagram from an Hamiltonian matrix H by first writing a diagram for its exponential $exp(tH)$ (which is sometimes easier) and then differentiate it as a function of t. While this approach is fundamentally different, it leads to diagrams that are very similar to those we obtain here.

[1] In our context, an Hamiltonian is the same as an Hermitian matrix

3. Definitions

3.1. Diagrammatic Languages

In this section, we introduce graphical (diagrammatical) languages.

A prop is one of the categorical ways to represent circuits. Morphisms in a prop (may) represent circuits: we have two composition laws on circuits: parallel ($\otimes$) and sequential ($\circ$) compositions that should satisfy some obvious properties.

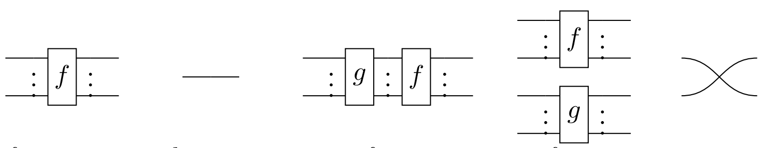
Definition 1 (prop)**.** *A **prop** $\boldsymbol{P}$ is the data, for each pair $(n, m) \in \mathbb{N}^2$ of nonnegative integers of a set $\boldsymbol{P}[n, m]$, called the set of morphisms. An element $f \in \boldsymbol{P}[n, m]$ is usually written $f : n \to m$. A prop should contain the following operators and constants:*

- *A **composition** $\circ : \boldsymbol{P}[b, c] \times \boldsymbol{P}[a, b] \to \boldsymbol{P}[a, c]$ satisfying: $(f \circ g) \circ h = f \circ (g \circ h)$.*
- *A **tensor product** $\otimes : \boldsymbol{P}[a, b] \times \boldsymbol{P}[c, d] \to \boldsymbol{P}[a + c, b + d]$, satisfying: $(f \otimes g) \otimes h = f \otimes (g \otimes h)$ and $(f \circ g) \otimes (h \circ k) = (f \otimes h) \circ (g \otimes k)$.*
- *An **empty morphism** $1 : 0 \to 0$ such that $f \otimes 1 = 1 \otimes f = f$ for all $f : a \to b$.*
- *An **identity** $id : 1 \to 1$ such that $f \circ id^{\otimes a} = id^{\otimes b} \circ f = f$ for all $f : a \to b$. With the convention $id^{\otimes 0} = 1$.*
- *A **symmetry** $\sigma : 2 \to 2$ satisfying: $\sigma^2 = id^{\otimes 2}$ and such that, $\sigma_a \circ (f \otimes id) = (id \otimes f) \circ \sigma_b$, for all $f : a \to b$, where $\sigma_{n+1} = (1^{\otimes n} \otimes \sigma) \circ (1 \otimes \sigma_n)$, with $\sigma_0 = id$.*

A prop functor F (i.e. a morphism of props) from P to Q is a set of maps, again denoted F, from $P[n, m]$ to $Q[n, m]$ such that $F(f \circ g) = F(f) \circ F(g), F(f \otimes g) = F(f) \otimes F(g)$, and $F(id_P) = id_Q, F(0_P) = 0_Q, F(\sigma_P) = \sigma_Q$.

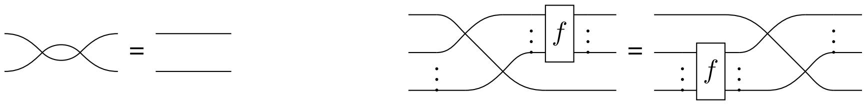
In the language of categories (Mac Lane, 1971), a prop is a small strict symmetric monoidal category whose monoid of object is spanned by a unique object.

Props admit a nice diagrammatical representation that gives a topological interpretation to the axioms (Selinger, 2010). A morphism $f : n \to m$ is represented as a box with n inputs and m outputs, with inputs on the left and outputs on the right. Composition is represented by plugging the boxes, and the tensor product by drawing the boxes side by side. The identity is represented by a single wire, the empty morphism by an empty diagram and the symmetry by wire crossing:



$f : n \to m$ $\quad id : 1 \to 1$ $\quad f \circ g$ $\quad f \otimes g$ $\quad \sigma : 2 \to 2$

These notations are well chosen, as the equations defining a prop correspond to equalities of diagrams that are obvious to the eye (Joyal & Street, 1991; Selinger, 2010), so we can equivalently work with equations or with diagrams. To give one example, the symmetry axioms express that the boxes can move through wires:



An important example of a prop is the Circuit prop, i.e. the props whose morphisms are unitaries: $P[n, n]$ is the set of unitaries of size 2^n, $\circ$ is the composition of unitaries, $\otimes$ is the tensor product of unitaries. Diagrams in this prop are exactly what is usually called a quantum circuit, without measures or wires initialized to a particular value.

Diagrammatical languages, i.e. props, are usually either defined explicitely, or implicitely by giving a set of generators and relations, s.t. the prop is everything one can built from the generators, with diagrams corresponding to the relations supposted to be equal. In this deliverable, we will not focus too much on the relations.

To simplify the discussion, we will not make a distinction between diagrams and the matrices they represent. Usually these might be considered two different props, that are isomorphic (or that we are trying to make isomorphic).

3.2. The ZX-Calculus

3.2.1. Definitions

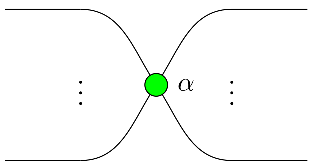
The ZX-Calculus (Coecke & Duncan, 2011; van de Wetering, 2020) is a set of generators and relations that capture the Qubit prop, that is the prop of matrices of size $2^n \times 2^m$:

- $P[n, m]$ is the set of matrices of size $2^m \times 2^n$.
- The sequential composition $\circ$ is the composition of matrices
- The spatial composition $\otimes$ is the tensor product of matrices

The generators are as follows:

- The green node $R_Z^{(n,m)}(\alpha) : n \to m$ that corresponds to the matrix $\begin{pmatrix} 1 & 0 & \cdots & 0 & 0 \\ 0 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & \cdots & 0 & e^{i\alpha} \end{pmatrix}$.

  It is represented by the following diagram, with the α omitted if it is 0.

  In the special case $m = n = 1$, this corresponds to the matrix $\begin{pmatrix} 1 & 0 \\ 0 & e^{i\alpha} \end{pmatrix}$, which is a phase shift gate.

  If $n = 0$, this diagram does not have any input. It represents a matrix of size $2^m \times 2^0$, i.e. a column vector of size 2^m. An important special case is when $n = 0, m = 1, \alpha = 0$, i.e. when the diagram has no input, one output and angle 0. In this case, it corresponds to the matrix $\begin{pmatrix} 1 \\ -1 \end{pmatrix}$, i.e. a scaled version of the vector $|+\rangle = \frac{|0\rangle+|1\rangle}{2}$. Similary, the node with parameters $n = 0, m = 1, \alpha = \pi$ corresponds to a scaled version of the vector $|-\rangle$.

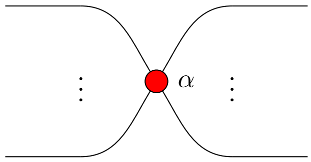
  If $m = n = 0$, the diagram has no input or output, it represents the *scalar* (i.e. a 1×1 vector) $(1 + e^{i\alpha})$

- The Hadamard matrix $h : 1 \to 1$ that represents the matrix $\frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$ and represented by the diagram:

- The red node $R_X^{(n,m)}(\alpha) : n \to m$ that represents the same matrix as the green node, but conjugated by the Hadamard matrix. Formally, it represents the matrix:

$$\left(\frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}\right)^{\otimes m} \begin{pmatrix} 1 & 0 & \cdots & 0 & 0 \\ 0 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & \cdots & 0 & e^{i\alpha} \end{pmatrix} \left(\frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}\right)^{\otimes n}$$

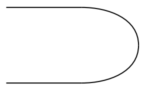
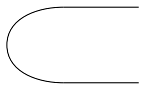
  It is represented pictorially by



In the special case when $n = m = 1$ i.e. a diagram with one input, and one output, the matrix is $\frac{1}{2}\begin{pmatrix} 1+e^{i\alpha} & 1-e^{i\alpha} \\ 1-e^{i\alpha} & 1+e^{i\alpha} \end{pmatrix}$. They can be interpreted as a rotation along the x-axis on the Bloch sphere. When $\alpha = 0$ it corresponds to the matrix $\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$, i.e. the X gate.

While green nodes with zero input and one output, and $\alpha \in \{0, \pi\}$ represent scaled version of $|+\rangle$ and $|-\rangle$, red nodes with the same parameters correspond to scaled version of $|0\rangle$ and $|1\rangle$ respectively.

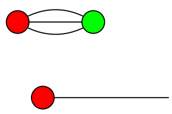
Indeed $R_X^{(0,1)}(0) = \begin{pmatrix} \sqrt{2} \\ 0 \end{pmatrix} = \sqrt{2}\,|0\rangle$.

- The cup matrix: $2 \to 0$ that represents the row vector $\begin{pmatrix} 1 & 0 & 0 & 1 \end{pmatrix}$ and is represented by 
- The cap matrix: $0 \to 2$ that represents the column vector $\begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix}$ and is represented by 

3.2.2. Examples

It is well known that every complex matrix can be represented by a diagram of the ZX-calculus (a ZX-diagram).

The vector $|0\rangle$. For instance, the vector $|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$ can be represented by the diagram:



To understand this diagram, note that it consists in two different diagrams side by side. We should therefore compute to which matrix the two differents parts refers to, and do their tensor product to obtain the matrix corresponding to the diagram

As explained above, we already know that a red node with 0 inputs and 1 output represents the vector $\sqrt{2}\,|0\rangle$, so we focus now on the upper part of the diagram.

The upper part of the diagram consists in a red node with 0 inputs and 3 ouputs composed with a green node with 3 inputs and 0 outputs. We therefore need to understand to which diagrams these two nodes correspond, and then do their matrix product. Note that the result is a diagram with 0 inputs and 0 outputs, and therefore corresponds to a matrix of size $2^0 \times 2^0$, i.e. a scalar.

The red node with 0 inputs and 3 outputs represents the column vector $1/\sqrt{2}\begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix}$ and the green node with 3 inputs and 0 outputs represents the row vector $\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$. Their matrix product is:

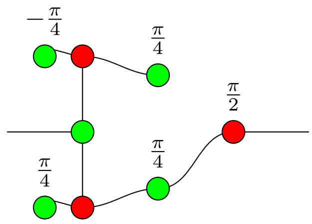
$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix} = \frac{1}{\sqrt{2}}$$

The diagram on the top therefore represents the scalar $\frac{1}{\sqrt{2}}$, and the diagram on the bottom represents the vector $\sqrt{2}\,|0\rangle$. The two diagrams are put side by side, which means we should do their tensor product to obtain the result:

$$\frac{1}{\sqrt{2}} \otimes \sqrt{2}\,|0\rangle = |0\rangle$$

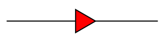
In what follows, to simplify the exposition, we will only represent diagrams upto scalar, i.e. we will say that the red node with 0 inputs and 1 output is the vector $|0\rangle$ while it is truly the vector $\sqrt{2}\,|0\rangle$.

The matrix $\begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$. Another important example is the matrix $\begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$ can be represented by the diagram

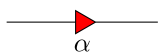


It is a tedious exercise to show that this diagram indeed corresponds to the intended matrix. Notice that this diagram contains *vertical* wires. It is intended to represent that the green node can be applied before the two red nodes (by representing it slightly to the left), or after the two red nodes, and that both will give the same matrix.

This matrix plays a central role in many applications, and will be represented by a triangle in what follows:



We will represent more generally the matrix $\begin{pmatrix} 1 & \alpha \\ 0 & 1 \end{pmatrix}$ by the picture:



A diagram for this matrix is not easy to produce in general for arbitrary α in the ZX-Calculus. If α is an integer, it can be obtained by composing the triangle with itself n times. Otherwise, the best way to produce this diagram is to use the *algebraic ZX-Calculus* (Wang & Yeung, 2021) where the triangle can be understood as a new generator.

The ZX-Calculus comes with a complete equational theory, i.e. a set of rules on how to rewrite diagrams s.t. two diagrams that represents the same matrix are equal up to local applications of the rules. It is not the purpose of this deliverable to reason about the equational theory, and it will therefore not be presented here. See (Vilmart, 2019) for one such axiomatisation.

4. A language to represent Hamiltonians

In this section, we introduce a diagrammatic language that will be able to represent diagonal Hamiltonians. For presentation purpose, we will mostly focus on representing *Ising* Hamiltonians, i.e. Hamiltonians of the form $\sum_{i,j} \alpha_i Z_i \otimes Z_j + \sum_i \beta_i Z_i$. Most Hamiltonians that appear in combinatorial problems, and in QAOA applications, are indeed Ising Hamiltonians. The generalization of the language to arbitrary diagonal Hamiltonians is straightforward, but adds heavier notation.

4.1. A Pro for Ising Hamiltonians

4.1.1. Discussion

Before introducing the language, it is important to say that Hamiltonians do not enjoy the same closure properties as other classes of matrices. In particular, the product of two Hermitian matrices is not an Hermitian matrix.

This is a problem because usually diagrammatic languages assume the composition $\circ$ is the matrix product, or related to the matrix product.

To obtain a diagrammatic language, we have to do something different. We will indeed suppose that the composition $\circ$ is the *sum* of the matrices. This is relevant, because sums of Hermitian matrices are again Hermitian. Changing the composition $\circ$ to the matrix sum, also means we need to have a non-obvious parallel composition. Indeed the tensor product does not commute with the sum. The good solution is to use the *tensor sum*:

$$A \oplus B = A \otimes I_M + I_N \otimes B$$

where I_M and I_N are the identity matrices of the same size as A and B respectively

There is also a more fundamental problem, which is that a Hermitian matrix with two columns permuted is not a Hermitian matrix anymore. This means that it will be hard to define the swap generator σ (the diagram that corresponds to permuting two wires) in a diagrammatic language for Hamiltonians.

In this section, we will therefore merely obtain a *pro* and not a prop: diagrams that we can compose sequentially or in parallel, but without having the possibility of permuting two wires.

This problem will be solved in the next section.

4.1.2. Definition

The previous discussion leads to the following definition:

Definition 2. *The pro of Hamiltonian matrices is the pro P where:*

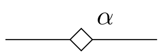
- *$P[n,n]$ is the set of all Hermitian matrices of dimension 2^n. $P[n,m]$ is empty*[1] *for $n \neq m$.*
- *The sequential composition of $M \in P[n,n]$ and $N \in P[n,n]$ is $N + M$*
- *The parallel composition of $M \in P[n,n]$ and $N \in P[m,m]$ is $N \oplus M$.*
- *The identity id is the 0 matrix.*

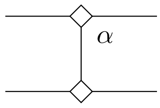
We stress again that we merely obtain a pro: we do not have a way to permute wires.

Now that we have the definition of the Pro, we can give generators and represent Hamiltonian matrices in it:

[1] The fact we need all our morphisms to have the same number of inputs and outputs is similar to what happens for quantum circuits.

Definition 3. *We distinguish two sets of generators in the previous pro:*

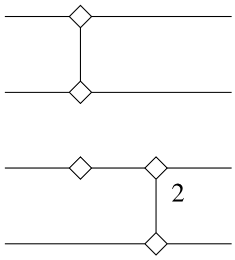
- *The* $1 \to 1$ *matrix* $\begin{pmatrix} \alpha & 0 \\ 0 & -\alpha \end{pmatrix}$ *represented by* 

. α is equal to 1 *if not represented.*
- *The* $2 \to 2$ *matrix* $\begin{pmatrix} \alpha & 0 & 0 & 0 \\ 0 & -\alpha & 0 & 0 \\ 0 & 0 & -\alpha & 0 \\ 0 & 0 & 0 & \alpha \end{pmatrix}$ *represented by* 

. α is equal to 1 *if not represented.*

4.1.3. Example

Now we have everything to represent Hamiltonians. For instance, the Hamiltonian $H = Z_1 Z_2 + Z_3 + 2Z_3 Z_4$ can be represented by:



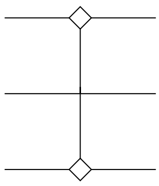
Notice that descriptions of Hamiltonian implicitely hide the identity matrices. Indeed, the Hamiltonian $H = Z_1 Z_2 + Z_3 + 2Z_3 Z_4$ should be written more carefully as $H = Z \otimes Z \otimes I \otimes I + I \otimes I \otimes Z \otimes I + 2I \otimes I \otimes Z \otimes Z$. This writing makes the appearance of the tensor sum in our diagrams more natural.

4.2. A Prop for Ising Hamiltonians

4.2.1. Definition

As explained in the previous section, what we have obtained is merely a pro: We did not provide any generator that is able to permute wires.

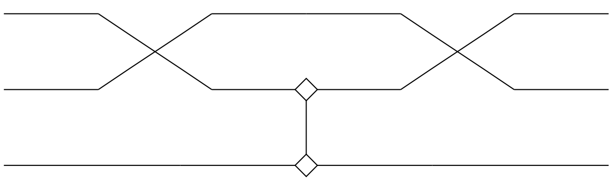
In fact, our previous language cannot even express all diagonal Hamiltonians. The problem will be apparent when trying to represent the Hamiltonian $Z_1 Z_3 = Z \otimes I \otimes Z$. An obvious representation would be:



However such a representation is impossible: Our generator can only be applied to two consecutive wires.

There are two ways to circumvent the problem: The first is to add generators that correspond to operators $Z_i Z_j$ applied on nonconsecutive wires (a family of generators indexed by the distance between the two wires). This first solution is very cumbersome, especially once one wants to deal with general diagonal Hamiltonians and not only Ising Hamiltonians.

The second solution is to indeed find a way to permute wires, so the following diagram makes sense:



While the diagram might look clean, there are two problems semantically. The first problem is that most permutations are not Hermitian matrices, which means that our language will be able to express more than just Hermitian matrices. The second problem is that our composition operator is the matrix sum, and in the previous diagram we certainly are trying to do the product of the three matrices, and not their sum.

Some changes are therefore needed the previous diagram to make sense.

For this, we introduce a new prop, called the *Unitary-Hermitian* prop. In this prop, a diagram represents intuitively an Hermitian matrix on which some Unitary matrix acts, i.e. a pair (U, H) of matrices. We will only be interested in the "H" part of this pair, but the diagram really represents both simultaneously.

Diagrammatically, the best way to think about a diagram $n \to n$ is to represent a matrix of dimension 2^{n+1}, that is with one more wire, of the form

$$\begin{pmatrix} U & UH \\ 0 & U \end{pmatrix} = (I \otimes U) \begin{pmatrix} I & H \\ 0 & I \end{pmatrix}$$

where U is any unitary matrix and H is a Hermitian matrix. The additional wire specifies which part of the matrix we are interested in: if it is initialized with $|0\rangle$ and postselected on $|0\rangle$, we recover the unitary part U. If it is initialized with $|1\rangle$ and postselected on $|0\rangle$ we recover the Hermitian part (skewed by U), the matrix UH. We can think of this additional (not represented) wire as some kind of control wire.

The sequential composition of two such matrices is obvious. For the parallel composition of two diagrams M and N, we need to be careful : the hidden wire should be shared by M and N.

4.2.2. Definition

This leads to the following definition:

Definition 4. *The Unitary-Hermitian prop (UH) is the prop P where:*

- *$P[n, n]$ are matrices of size 2^{n+1} of the form*

$$\begin{pmatrix} U & UH \\ 0 & U \end{pmatrix}$$

  where U is unitary and H Hermitian. $P[n, m]$ is empty for $n \neq m$.

- *The sequential composition of $M \in P[n, n]$ and $N \in P[n, n]$ is $M \circ N = MN$*

- *The parallel composition of $M = \begin{pmatrix} U & UH \\ 0 & U \end{pmatrix}$ and $N = \begin{pmatrix} U' & U'H' \\ 0 & U' \end{pmatrix}$ is*

$$M \boxtimes N = \begin{pmatrix} U \otimes U' & (U \otimes U')(H \oplus H') \\ 0 & U \otimes U' \end{pmatrix}$$

This formal definition of the parallel composition indeed corresponds to the intuition given above.

There are a few things to prove for the definition to make sense. First that the sequential composition is indeed in $P[n, n]$. Indeed:

$$\begin{pmatrix} U & UH \\ 0 & U \end{pmatrix} \begin{pmatrix} V & VK \\ 0 & V \end{pmatrix} = \begin{pmatrix} UV & UVK + UHV \\ & UV \end{pmatrix} = \begin{pmatrix} UV & UV(K + V^\dagger HV) \\ & UV \end{pmatrix}$$

and $(K + V^\dagger HV)$ is indeed Hermitian.

It is also nontrivial that the two compositions are compatible, i.e. that $(A \boxtimes B)(C \boxtimes D) = (AC \boxtimes BD)$. This is proven in Appendix A.1

We reiterate that in this prop, diagrams represents pairs (U, H) of matrices, but we will think of them as representing only the second matrix, with U as a byproduct. If we want to represent only Ising matrices, this prop is too large: we don't need all matrices U and we don't need all Hamiltonians. We will consider the subprop:

Definition 5. *The* diagonal *subprop UH_D is the subprop of UH consisting of all matrices*

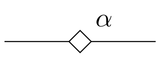
$$M = \begin{pmatrix} U & UH \\ 0 & U \end{pmatrix}$$

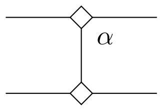
where H is diagonal Hermitian and U is unitary and has exactly one nonzero coefficient per row and column.

(In pratice, we could require that U is a permutation matrix, but we can accomodate a slightly bigger prop).

In this prop, we can now represent Ising Hamiltonians, using permutations if necessary, by using the same two generators as before.

Proposition 1. *Any Ising Hamiltonian can be represented in UH_0 using the following two generators, and possibly permutation of the wires:*

- *The $1 \to 1$ matrix* $\begin{pmatrix} 1 & 0 & \alpha & 0 \\ 0 & 1 & 0 & -\alpha \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$ *represented by* 

. α is equal to 1 if not represented.
- *The $2 \to 2$ matrix* $\begin{pmatrix} 1 & 0 & 0 & 0 & \alpha & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & -\alpha & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & -\alpha & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & \alpha \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$ *represented by* 

. α is equal to 1 if not represented.

More precisely the diagram will represent the matrix $\begin{pmatrix} I & H \\ 0 & I \end{pmatrix}$.

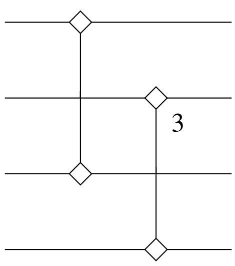
Notice that these generators are the same as in Definition 3, except they represent different matrices: If a generator represented a matrix H in the previous pro, it now represents the matrix $\begin{pmatrix} I & H \\ 0 & I \end{pmatrix}$.

Proof. Let $H = \sum_i \alpha_i Z_i + \sum_{i,j} \beta_i Z_i Z_j$. It is enough to represent each term independently by a diagram , and then sum them by putting them one after the other.

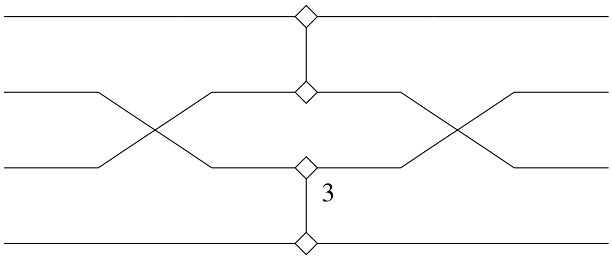
The Hamiltonian $\alpha_i Z_i$ can be obtained by using the first generator on the ith wire. The Hamiltonian $\beta_{i,j} Z_i Z_j$ can be obtained by first exchanging the wire j and the wire $i+1$, using permutations, then using the second generators on wires i and $i+1$, then exchanging againt the wire j and the wire $i+1$. See the example below for a picture. $\square$

4.2.3. Examples

The Hamiltonian $H = Z_1 Z_3 + 3 Z_2 Z_4$ can be represented as:



or, without using notations:



More accurately, this diagram represents the matrix $\begin{pmatrix} I & H \\ 0 & I \end{pmatrix}$.

4.2.4. Notes

We note a few things about our new diagram representation:

- Our composition is now the "normal" composition rather than the sum of matrices.
- There is a prop functor F from the pro of the previous section to the prop UH sending H to $\begin{pmatrix} I & H \\ 0 & I \end{pmatrix}$. This is indeed a functor, i.e. $F(A + B) = F(A) \circ F(B)$, and $F(A \oplus B) = F(A) \boxtimes F(B)$.
- There is a prop functor F from the prop of circuits to the prop UH sending U to $\begin{pmatrix} U & 0 \\ 0 & U \end{pmatrix}$. This functor is also monoidal, i.e. $F(A \otimes B) = F(A) \boxtimes F(B)$. There is a reverse functor G from the prop UH to the prop of circuits that maps $\begin{pmatrix} U & UH \\ 0 & U \end{pmatrix}$ to U, i.e. that forgets abouts H.

5.Representing Hamiltonians and their exponential in the ZX-Calculus

We will now explain how we can use the representation of the previous section to represent Hamiltonians and compute their exponential in the ZX-Calculus.

5.1. Computing the exponential

First, we explain how to compute the exponential of a diagram given in the UH_D language. The resulting diagram is a diagram of the ZX-Calculus. Fix a real number t. We denote by $\exp_t(H)$ the map e^{itH}.

The main theorem is as follows. It implies that we can build the exponential brick by brick.

Theorem 1. *The map F from the prop UH_D to the prop $Qubit$ that sends $M = \begin{pmatrix} U & UH \\ 0 & U \end{pmatrix}$ to $U \exp_t(H)$ is a prop functor. In particular, $F(MN) = F(M)F(N)$ and $F(M \boxtimes N) = F(M) \otimes F(N)$.*

The fact that F is a prop functor also expresses that permuting wires in the prop UH_D also corresponds to permuting wires in the prop $Qubit$, i.e. the image of σ is σ.

Proof. Let $M = \begin{pmatrix} U & UH \\ 0 & U \end{pmatrix}$ and $N = \begin{pmatrix} V & VK \\ 0 & V \end{pmatrix}$. Then

$MN = \begin{pmatrix} UV & UVK + UHV \\ 0 & UV \end{pmatrix} = \begin{pmatrix} UV & UV(K + V^\star HV) \\ 0 & UV \end{pmatrix}$

Therefore

$$F(MN) = UV \exp_t(K + V^\star HV)$$

Notice that K and $V^\star HV$ are both diagonal matrices, so their exponential commute, and we can write:

$$F(MN) = UV \exp_t(V^\star HV) \exp_t(K) = UVV^\star \exp_t(H)V \exp_t(K) = U \exp_t(H)V \exp_t(K) = F(M)F(N)$$

Now

$$M \boxtimes N = \begin{pmatrix} U \otimes V & (U \otimes V)(H \oplus K) \\ 0 & U \otimes V \end{pmatrix}$$

so that $F(M \boxtimes N) = U \otimes V \exp_t(H \oplus K)$.

Now $\exp_t(H \oplus K) = \exp_t(H \otimes I + I \otimes K) = \exp_t(H \otimes I) \exp_t(I \otimes K)$ as the two matrices commute and

$$\exp_t(H \otimes I) = \sum_n \frac{(it)^n (H \otimes I)^n}{n!} = \sum_n \frac{(itH)^n \otimes I}{n!} = \sum_n \frac{(itH)^n}{n!} \otimes I = \exp_t(H) \otimes I$$

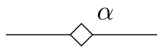
Therefore $F(M \boxtimes N) = (U \otimes V)(\exp_t(H) \otimes I)(I \otimes \exp_t(K)) = (U \exp_t(H) \otimes V \exp_t(K)) = F(M) \otimes F(N)$

□

The consequence of the theorem is that it is enough to define the exponentials on the generators to obtain the exponential on any diagram. As explained in Proposition 1, Ising Hamiltonians are built using 2 generators presented in section 4 and the swap. It is therefore enough to give the image of the three generators by F to know how to compute the exponential of any Ising Hamiltonian.

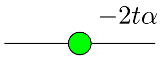
We already know that permuting wires in UH_D corresponds to permuting wires in $Qubit$. It remains to find representations for the image of the two generators of UH_D:

- Consider the generator

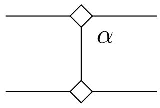


The exponential of the matrix $\begin{pmatrix} \alpha & 0 \\ 0 & -\alpha \end{pmatrix}$ is the matrix $\begin{pmatrix} e^{it\alpha} & 0 \\ 0 & e^{-it\alpha} \end{pmatrix}$

Upto a scalar this is exactly the matrix related to the green node of parameter $-2t\alpha$. In other words, it is:

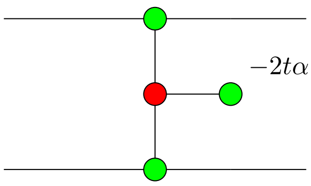


- Consider the diagram



The exponential of the matrix $\begin{pmatrix} \alpha & 0 & 0 & 0 \\ 0 & -\alpha & 0 & 0 \\ 0 & 0 & -\alpha & 0 \\ 0 & 0 & 0 & \alpha \end{pmatrix}$ is the matrix $\begin{pmatrix} e^{it\alpha} & 0 & 0 & 0 \\ 0 & e^{-it\alpha} & 0 & 0 \\ 0 & 0 & e^{-it\alpha} & 0 \\ 0 & 0 & 0 & e^{it\alpha} \end{pmatrix}$

Up to a scalar, this can be done with the following ZX-diagram:



We therefore obtain the following correspondence table

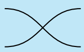
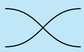
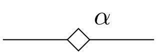
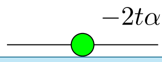
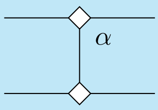
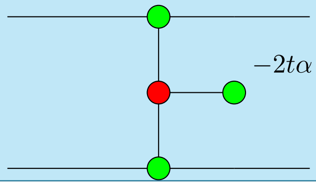
H	$exp_t(H)$
	
	
	

***Table 1**: How to transform generators of the UH_D calculus into representations of their exponentials*

We then obtain the following algorithm to compute the exponential of the diagram:

- See the diagram as parallel and sequential compositions of the generators
- Replace each generator by their exponential using the table
- Compose them using the parallel and sequential compositions of the ZX-Calculus

5.2. Hamiltonians in the ZX-Calculus

Given an Hamiltonian H, the QAOA algorithm produces a circuit involving its exponential $exp_t(H)$ that computes a state $|\psi\rangle$. To analyse the results of the circuit, one needs to compute $\langle\psi|\, H\, |\psi\rangle$. To analyse this diagrammatically, we need to be able to represent both the matrix H and its exponential in the same formalism. This is not the case for now, as the matrix is represented in the language UH_D and its exponential in the ZX-Calculus.

However, we can also use the language UH_D to represent the matrix in the ZX-Calculus itself.

Indeed, we explained in Section 4 that the intuitive understanding of diagrams of the UH_D are diagrams with some kind of hidden additional wire. To produce terms of the ZX-Calculus, we will make this additional wire explicit.

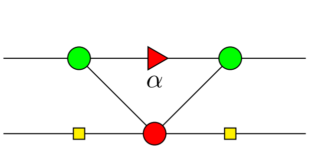
The two generators of UH_D correspond to the two matrices:

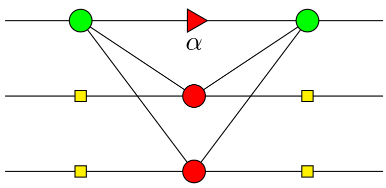
$$\begin{pmatrix} 1 & 0 & \alpha & 0 \\ 0 & 1 & 0 & -\alpha \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

and

$$\begin{pmatrix} 1 & 0 & 0 & 0 & \alpha & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & -\alpha & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & -\alpha & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & \alpha \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

They can be represented in the ZX-Calculus by the following two diagrams





In these two diagrams, the upper wire represents the hidden (control) wire.

To obtain a ZX-diagram, we just have to use this correspondence, keeping in mind that the control wire should be common among all generators, which implies that we should first “linearize” the diagram so that there are never two generators in parallel. We will then obtain a diagram that represents the matrix $\begin{pmatrix} I & H \\ 0 & I \end{pmatrix}$, and we will need to initialize the control wire with $|1\rangle$ and postselect it on $|0\rangle$.

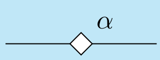
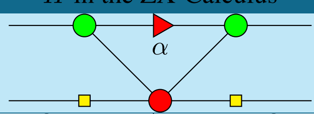
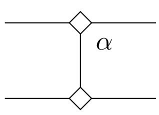
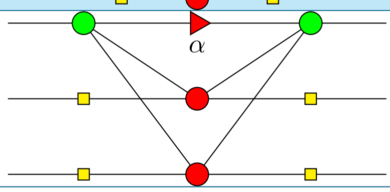
H in the UH_D calculus	H in the ZX-Calculus
	
	

Table 2: How to transform diagrams of the UH_C calculus into diagrams of the ZX-calculus. The wire on top should be common among all generators.

We then obtain the following algorithm to transform a diagram of the UH_D calculus into a diagram of the ZX-calculus.:

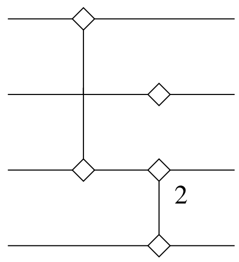
- See the diagram as parallel and sequential compositions of the generators, in such a way that there is never two generators in parallel
- Replace each generator by their equivalent in the ZX-calculus using the table, and adding a wire. The top wire should be common among all generators
- Compose them using the parallel and sequential compositions of the ZX-Calculus
- Initialize the top wire with $|1\rangle$ and postselect it on $|0\rangle$ by putting a red node $0 \to 1$ with parameter π on the left (which corresponds to $|1\rangle$, and a red node $1 \to 0$ with parameter π on the right (which corresponds to $\langle 0|$).

5.3. An example

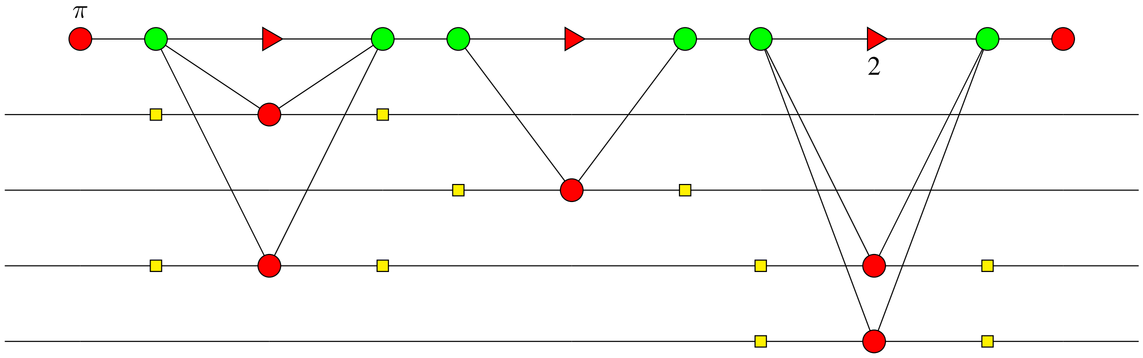
To put everything together, we give an example of an Hamiltonian in the UH_D calculus, how it is represented in the ZX-calculus, and how its exponential is represented.

Our example is the Hamiltonian $H = Z_1Z_3 + Z_2 + 2Z_3Z_4$.

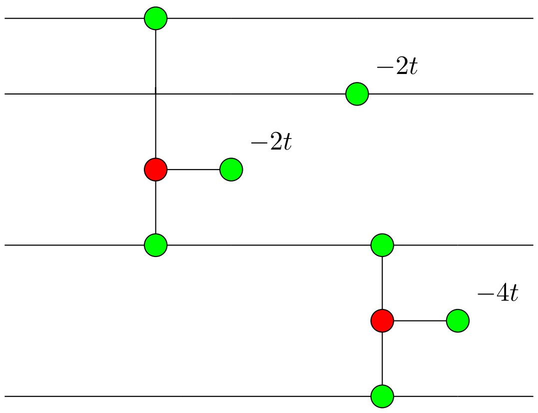
In the UH_D calculus, this can be represented by:



Using the table of section 5.2, we obtain the following diagram for the same Hamiltonian in the ZX-Calculus.



And its exponential can be found using the table of section 5.1



6.Conclusions

This deliverable provides a way to represent Hamiltonians in a new language, called UH_D, and an algorithm to obtain representations of the same matrix and its exponential in the ZX-Calculus. Subsequent work in the context of the NEASQC project consists in using this representation to analyse formally some particular instances of QAOA.

List of Tables

Table 1.: How to transform generators of the UH_D calculus into representations of their exponentials . . 16
Table 2.: How to transform diagrams of the UH_C calculus into diagrams of the ZX-calculus. The wire on top should be common among all generators. 17

Bibliography

Carette, T., Hoffreumon, T., Larroque, É., & Vilmart, R. (2023). Complete graphical language for hermiticity-preserving superoperators. *Proceedings of the 38th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*.

Coecke, B., & Duncan, R. (2011). Interacting quantum observables: Categorical algebra and diagrammatics. *New Journal of Physics*, *13*(4), 043016. https://doi.org/10.1088/1367-2630/13/4/043016

Cowtan, A., Dilkes, S., Duncan, R., Simmons, W., & Sivarajah, S. (2020). Phase gadget synthesis for shallow circuits. *Electronic Proceedings in Theoretical Computer Science*, *318*, 213–228. https://doi.org/10.4204/eptcs.318.13

de Beaudrap, N., Bian, X., & Wang, Q. (2020). Techniques to reduce π/4-parity-phase circuits, motivated by the ZX calculus. *Electronic Proceedings in Theoretical Computer Science*, *318*, 131–149. https://doi.org/10.4204/eptcs.318.9

Duncan, R., Kissinger, A., Perdrix, S., & van de Wetering, J. (2020). Graph-theoretic simplification of quantum circuits with the ZX-calculus. *Quantum*, *4*, 279. https://doi.org/10.22331/q-2020-06-04-279

Duncan, R., & Perdrix, S. (2010). Rewriting measurement-based quantum computations with generalised flow. In S. Abramsky, C. Gavoille, C. Kirchner, F. Meyer auf der Heide, & P. G. Spirakis (Eds.), *Automata, languages and programming* (pp. 285–296). Springer Berlin Heidelberg.

Farhi, E., Goldstone, J., & Gutmann, S. (2014). A quantum approximate optimization algorithm. https://doi.org/10.48550/ARXIV.1411.4028

Garvie, L., & Duncan, R. (2018). Verifying the smallest interesting colour code with quantomatic. *Electronic Proceedings in Theoretical Computer Science*, *266*, 147–163. https://doi.org/10.4204/eptcs.266.10

Jeandel, E., Perdrix, S., & Veshchezerova, M. (2022). Addition and differentisdation of zx-diagrams. *International Conference on Formal Structures for Computation and Deduction (FSCD)*. https://doi.org/10.4230/LIPIcs.FSCD.2022.13

Joyal, A., & Street, R. (1991). The geometry of tensor calculus, I. *Advances in mathematics*, *88*(1), 55–112.

Kissinger, A., & van de Wetering, J. (2019). Universal MBQC with generalised parity-phase interactions and pauli measurements. *Quantum*, *3*, 134. https://doi.org/10.22331/q-2019-04-26-134

Kissinger, A., & van de Wetering, J. (2020). Reducing the number of non-clifford gates in quantum circuits. *Physical Review A*, *102*(2). https://doi.org/10.1103/physreva.102.022406

Mac Lane, S. (1971). *Categories for the working mathematician*. Springer.

Selinger, P. (2010). A survey of graphical languages for monoidal categories. In *New structures for physics* (pp. 289–355). Springer.

Shaikh, R. A., Wang, Q., & Yeung, R. (2022). *How to sum and exponentiate hamiltonians in zxw calculus* [arXiv:2212.04462].

van de Wetering, J. (2020). Zx-calculus for the working quantum computer scientist.

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

Wang, Q., & Yeung, R. (2021). Representing and implementing matrices using algebraic zx-calculus. https://doi.org/10.48550/ARXIV.2110.06898

Wang, Q., & Yeung, R. (2022). Differentiating and integrating zx diagrams.

A.Appendix

A.1. Compatibility of the two compositions

In this section, we use the following facts about $\oplus$:

- $(M+N)\oplus(O+P)=(M\oplus O)+(N\oplus P)$
- $(U\otimes V)(M\oplus N)(U^\dagger\otimes V^\dagger)=(UMU^\dagger)\oplus(VNV^\dagger)$ if U and V are unitary.

Now let $A=\begin{pmatrix}U & UH\\ 0 & U\end{pmatrix}$, $B=\begin{pmatrix}U' & U'H'\\ 0 & U'\end{pmatrix}$, $C=\begin{pmatrix}V & VK\\ 0 & V\end{pmatrix}$ and $D=\begin{pmatrix}V' & V'K'\\ 0 & V'\end{pmatrix}$

Then

$$A\boxtimes B=\begin{pmatrix}U\otimes U' & (U\otimes U')(H\oplus H')\\ 0 & U\otimes U'\end{pmatrix}$$

$$C\boxtimes D=\begin{pmatrix}V\otimes V' & (V\otimes V')(K\oplus K')\\ 0 & V\otimes V'\end{pmatrix}$$

and therefore

$$\begin{aligned}(A\boxtimes B)(C\boxtimes D) &= \begin{pmatrix}UV\otimes U'V' & (UV\otimes U'V')(K\oplus K')+(U\otimes U')(H\oplus H')(V\otimes V')\\ 0 & UV\otimes U'V'\end{pmatrix}\\ &= \begin{pmatrix}UV\otimes U'V' & (UV\otimes U'V')\left[K\oplus K'+(V^\dagger\otimes V'^\dagger)(H\oplus H')(V\otimes V')\right]\\ 0 & UV\otimes U'V'\end{pmatrix}\\ &= \begin{pmatrix}UV\otimes U'V' & (UV\otimes U'V')\left(K\oplus K'+V^\dagger HV\oplus V'^\dagger H'V'\right)\\ 0 & UV\otimes U'V'\end{pmatrix}\end{aligned}$$

On the other hand

$$AC=\begin{pmatrix}UV & UVK+UHV\\ 0 & UV\end{pmatrix}=\begin{pmatrix}UV & UV(K+V^\dagger HV)\\ 0 & UV\end{pmatrix}$$

and

$$BD=\begin{pmatrix}U'V' & U'V'(K'+V'^\dagger H'V')\\ 0 & U'V'\end{pmatrix}$$

so that

$$\begin{aligned}AC\boxtimes BD &= \begin{pmatrix}UV\otimes UV' & (UV\otimes UV')((K+V^\dagger HV)\oplus(K'+V'^\dagger H'V'))\\ 0 & UV\otimes U'V'\end{pmatrix}\\ &= (A\boxtimes B)(C\boxtimes D)\end{aligned}$$