ZH: A Complete Graphical Calculus for Quantum Computations Involving Classical Non-linearity

Miriam Backens
Department of Computer Science
University of Oxford
miriam.backens@cs.ox.ac.uk

Aleks Kissinger
Institute for Computing and Information Sciences
Radboud University
aleks@cs.ru.nl

We present a new graphical calculus that is sound and complete for a universal family of quantum circuits, which can be seen as the natural string-diagrammatic extension of the approximately (real-valued) universal family of Hadamard+CCZ circuits. The diagrammatic language is generated by two kinds of nodes: the so-called 'spider' associated with the computational basis, as well as a new arity-N generalisation of the Hadamard gate, which satisfies a variation of the spider fusion law. Unlike previous graphical calculi, this admits compact encodings of non-linear classical functions. For example, the AND gate can be depicted as a diagram of just 2 generators, compared to ~ 25 in the ZX-calculus. Consequently, N-controlled gates, hypergraph states, Hadamard+Toffoli circuits, and diagonal circuits at arbitrary levels of the Clifford hierarchy also enjoy encodings with low constant overhead. This suggests that this calculus will be significantly more convenient for reasoning about the interplay between classical non-linear behaviour (e.g. in an oracle) and purely quantum operations. After presenting the calculus, we will prove it is sound and complete for universal quantum computation by demonstrating the reduction of any diagram to an easily describable normal form.

1 Introduction

Graphical calculi enable reasoning about quantum computation in an intuitive yet rigorous way. Calculi based on string diagrams are more flexible than circuit-style languages, allowing the description of states and measurement projections as well as unitary operations in one unified framework. Their rigour is ensured by the category-theoretical underpinnings [25]. The best-known of these graphical languages is the ZX-calculus, which was first introduced 10 years ago [6, 7]. It is built around the interactions of two complementary bases, the computational basis and the Hadamard basis, which are graphically represented by so-called *spiders*. A related formalism is the ZW-calculus [14], which is built around the interactions of generators related to the two different types of three-qubit entangled states: GHZ states and W states.

Here, we introduce a new graphical language called the *ZH-calculus*, which roughly follows this analogy with multipartite entanglement:

ZX-calculus : *ZH-calculus* :: *graph states* : *hypergraph states*

Graph states are the basic resource for the one-way model of measurement-based quantum computation [22], and have been studied extensively using the ZX-calculus [7, 10, 11, 12]. Hypergraph states were introduced in [24] as a generalisation of graph states, and have recently gathered some interest due, for example, to the role they play in quantum search algorithms [23], exponentially-growing Bell violations [13], and universal measurement-based quantum computation [20].

Like the ZX- and ZW-calculi, the ZH-calculus includes a family of "Z-spiders" associated with the computational basis. However, its point of departure is the inclusion of "H-boxes", which are n-ary generalisations of the Hadamard gate satisfying a variation of the spider fusion law, much like the

Giulio Chiribella, Peter Selinger (Eds.): 15th International Conference on Quantum Physics and Logic (QPL 2018).
EPTCS 287, 2019, pp. 23–42, doi:10.4204/EPTCS.287.2

one satisfied by W-spiders in the ZW-calculus.[1] Whereas Hadamard gates are used to introduce edges between 2 vertices in a graph state, H-boxes can introduce hyperedges between n vertices in a hypergraph state. Seen from another perspective, H-boxes are closely related to both n-ary AND gates in classical logic and to the Hadamard-CCZ family of quantum circuits. As a result, Boolean circuits can be encoded in the ZH-calculus with low constant overhead. In particular, the linear maps corresponding to classical AND and NOT gates can be depicted as follows in terms of the ZH calculus:

While the unitary NOT gate has a simple expression in the ZX-calculus, a typical encoding of an AND gate requires 25 basic generators and non-Clifford phases (cf. [8], §12.1.3.1). Similarly, multiply-controlled phase gates also have very succinct representations, indicating that the ZH-calculus may be useful for analysing Hadamard-CCZ circuits (a.k.a. Hadamard-Toffoli circuits [26, 1], cf. forthcoming [5] for connection to ZH), as well as diagonal operators at any level of the Clifford hierarchy [9].

Our main theorem is the ZH-calculus is complete with respect to its standard interpretation as matrices. That is, if two ZH-diagrams describe the same matrices, then they are provably equal using the rules of the ZH-calculus. Perhaps one of the most appealing features of the calculus is the simplicity of this completeness proof. The core of the proof (section 3) fits on 4 pages, where only especially intricate lemmas—which appear in Appendix B—were done within the proof assistant Quantomatic [19]. This is due to two main factors. The first is the extensive use of *!-box notation* [18], which gives an elegant way to reason about diagrams which have arbitrarily-large fan-out-type behaviour. The second is a unique normal form for the ZH-calculus, which expresses any matrix as a Schur product – i.e. entrywise product – of elementary matrices with the property that all but one entry of each matrix is 1. This multiplicative construction contrasts with the additive construction of the normal form in the ZW-calculus [14], which arises as a sum of elementary matrices with the property that all but one entry of each matrix is 0. For example the normal form of the diagram corresponding to the matrix $\left(\begin{smallmatrix} a & b \\ c & d \end{smallmatrix}\right)$ is effectively constructed as follows, where the left-hand side represents the approach in the ZH-calculus with $*$ denoting entrywise multiplication, and the right-hand side represents the approach in the ZW-calculus:

$$\begin{pmatrix} a & 1 \\ 1 & 1 \end{pmatrix} * \begin{pmatrix} 1 & b \\ 1 & 1 \end{pmatrix} * \begin{pmatrix} 1 & 1 \\ c & 1 \end{pmatrix} * \begin{pmatrix} 1 & 1 \\ 1 & d \end{pmatrix} = \begin{pmatrix} a & b \\ c & d \end{pmatrix} = \begin{pmatrix} a & 0 \\ 0 & 0 \end{pmatrix} + \begin{pmatrix} 0 & b \\ 0 & 0 \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ c & 0 \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ 0 & d \end{pmatrix}.$$

Unlike the completeness proofs for universal versions of the ZX-calculus [16, 21], which make use of the ZW-completeness proof via suitable translations between the two respective languages, our proofs of soundness, completeness, and universality are self-contained and don't rely on encoding into another calculus.

The paper is structured as follows. The generators and relations of the ZH-calculus are introduced in Section 2 and the calculus is proved to be universal and sound. The completeness proof is given in Section 3. In section 4 we survey two potential applications and comment on future work. Omitted proofs and a link to the Quantomatic project used to prove Lemmas B.1 and B.3 are given in the appendix.

2 Definition of the ZH-calculus

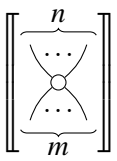
The ZH-calculus is a graphical language expressing operations as *string diagrams*. These are diagrams consisting of dots or boxes, connected by wires. Wires are also allowed to have one or two "dangling"

[1] Despite satisfying a similar variation of the spider fusion rule, this generalisation of the Hadamard node is different from that employed in the original version of the Y-calculus [17, Definition 2 of Version 1].

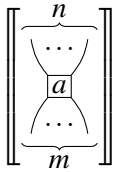
ends, which are not connected to a dot or box: these represent inputs of the diagram if oriented towards the bottom, outputs of the diagram if oriented to the top.

2.1 The generators and their interpretation

The diagrams of the ZH-calculus are generated by *Z-spiders*, which are represented as white dots, and *H-boxes*, which are represented as white boxes labelled with a complex number a. These generators are interpreted as follows, where $[\![\cdot]\!]$ denotes the map from diagrams to matrices.



$:= |0\rangle^{\otimes n}\langle 0|^{\otimes m} + |1\rangle^{\otimes n}\langle 1|^{\otimes m}$

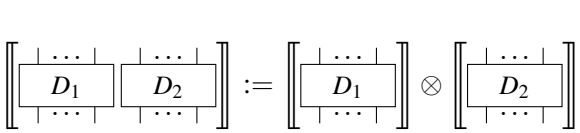


$:= \sum a^{i_1 \dots i_m j_1 \dots j_n} |j_1 \dots j_n\rangle\langle i_1 \dots i_m|$

The sum in the second equation is over all $i_1, \dots, i_m, j_1, \dots, j_n \in \{0,1\}$, i.e. an H-box represents a matrix all but one of whose entries are equal to 1. A label of -1 is usually left out and the box is then drawn smaller, e.g.  $:=$  . Straight and curved wires have the following interpretations:

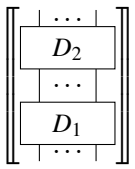
$$[\![\,|\,]\!] := |0\rangle\langle 0| + |1\rangle\langle 1| \qquad [\![\smile]\!] := |00\rangle + |11\rangle \qquad [\![\frown]\!] := \langle 00| + \langle 11|.$$

The juxtaposition of two diagrams is interpreted as the tensor product of the corresponding matrices and the sequential composition of two diagrams is interpreted as the matrix product of the corresponding matrices:

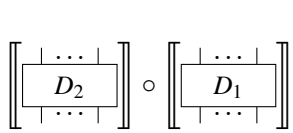


$:=$

$\otimes$

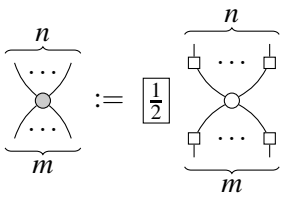


$:=$

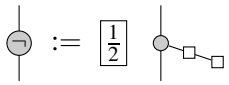


$\circ$

The statements of the relations of the ZH-calculus will be simplified by introducing two derived generators, called *grey spiders* and NOT, respectively.



(1)

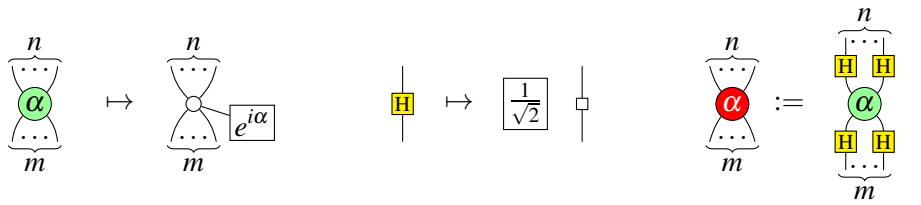


(2)

With these definitions,  acts on computational basis states as XOR and  acts as NOT:

$$\left[\!\!\left[\,\right]\!\!\right] = |0\rangle\langle 00| + |0\rangle\langle 11| + |1\rangle\langle 01| + |1\rangle\langle 10| \qquad \left[\!\!\left[\,\right]\!\!\right] = |0\rangle\langle 1| + |1\rangle\langle 0|.$$

There is an evident encoding of the generators of the ZX-calculus into ZH given by the following translation:



Since it is well-known that the ZX-calculus is universal for representing arbitrary linear maps $\mathbb{C}^{2^m} \to \mathbb{C}^{2^n}$, the following is immediate:

Proposition 2.1. Any linear map $\mathbb{C}^{2^m} \to \mathbb{C}^{2^n}$ can be expressed using the generators of the ZH-calculus.

We will also give a normal form in Section 3 which directly implies universality of the ZH-calculus, without going via ZX.

2.2 The relations, and soundness

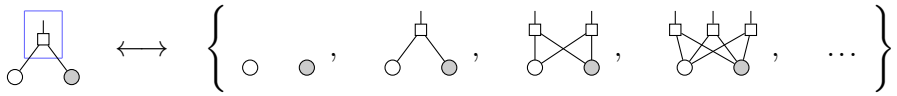
The rules of the ZH-calculus are given in Figure 1. We furthermore add one meta rule, often stated as "only topology matters". That is, two diagrams are considered equivalent if one can be deformed into the other. Furthermore, the Z-spiders and H-boxes are assumed to be *symmetric* and *undirected*: i.e. two inputs or outputs of the same generator can be swapped without changing the interpretation, and an input can be "bent" around to become an output, or conversely. Graphically:

Proposition 2.2. The ZH-calculus is sound.

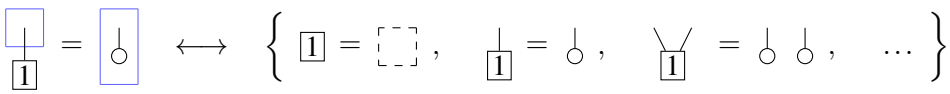
Proof. It is straightforward to check the symmetry properties for each generator and all of the rules in Figure 1 by concrete calculation. Soundness of the meta rule "only the topology matters" follows by considering the string diagrams as morphisms in a compact closed category [25]. □

2.3 !-box notation

Many of the calculations in this paper are greatly simplified by the use of *!-box notation* [18]. A !-box (pronounced "bang box") in a string diagram represents a part of the diagram that is able to fan out arbitrarily. That is, the contents of a !-box, along with any wires into or out of the !-box, can be copied n times for any non-negative integer n. For example, the !-box diagram below represents the following family of (concrete) string diagrams, one for each n:



All of the resulting string diagrams are well-defined because all of our generators can have arbitrary arities. We can also use !-boxes in diagram equations, as long as each !-box on the LHS has a corresponding !-box on the RHS, and the inputs/outputs in each !-box match. Such a rule represents a family of equations where each *pair* of corresponding !-boxes is replicated n times:



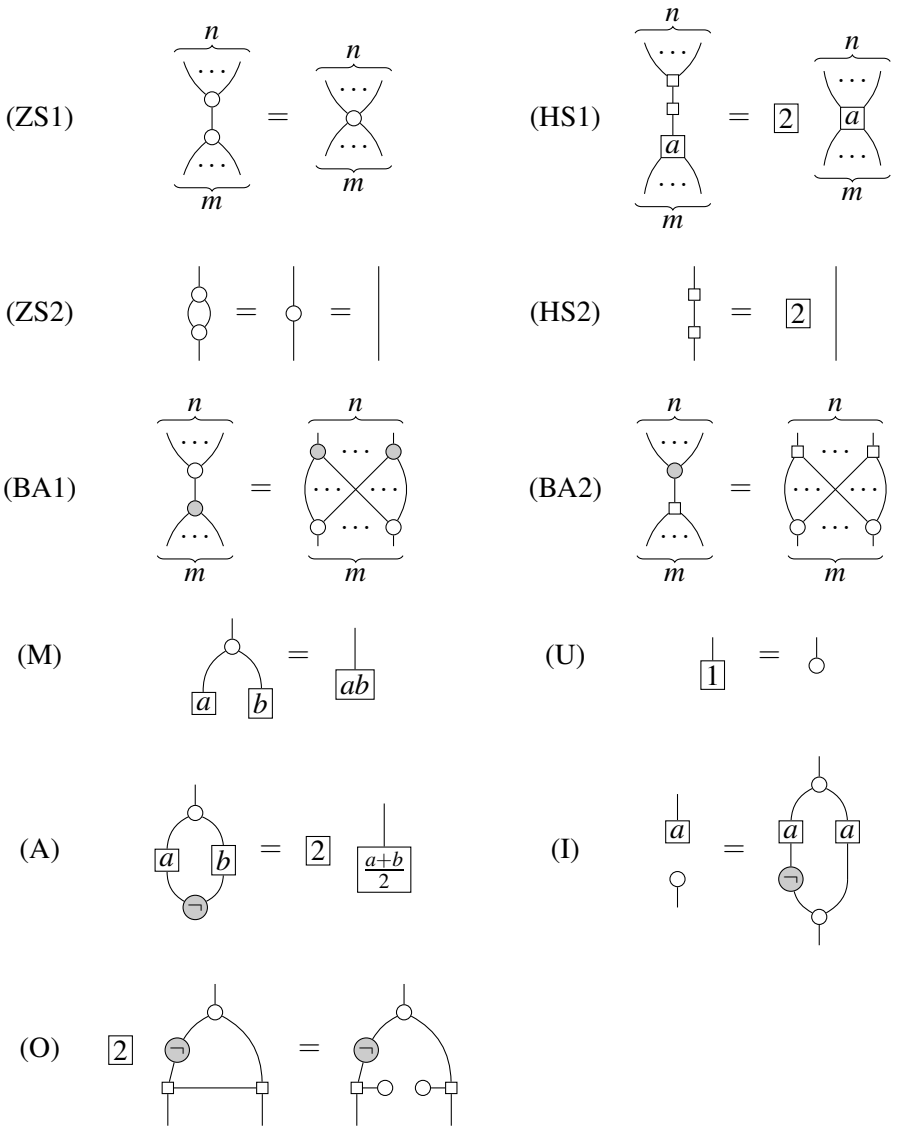
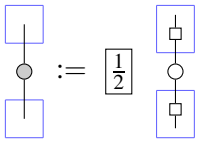


Figure 1: The rules of the ZH-calculus. Throughout, m,n are nonnegative integers and a,b are arbitrary complex numbers. The right-hand sides of both *bialgebra* rules (BA1) and (BA2) are complete bipartite graphs on $(m+n)$ vertices, with an additional input or output for each vertex. The horizontal edges in equation (O) are well-defined because only the topology matters and we do not need to distinguish between inputs and outputs of generators. n.b. the rules (M), (A), (U), (I), and (O) are pronounced *multiply*, *average*, *unit*, *intro*, and *ortho*, respectively.

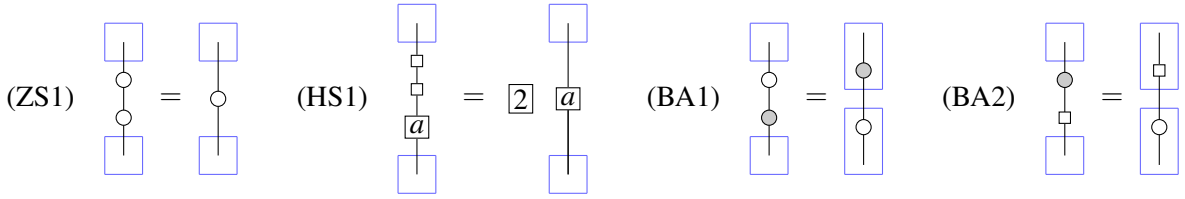
Note the dashed box on the right-hand side of the first equation denotes an empty diagram.

With this notation, the definition of grey spiders (1) becomes



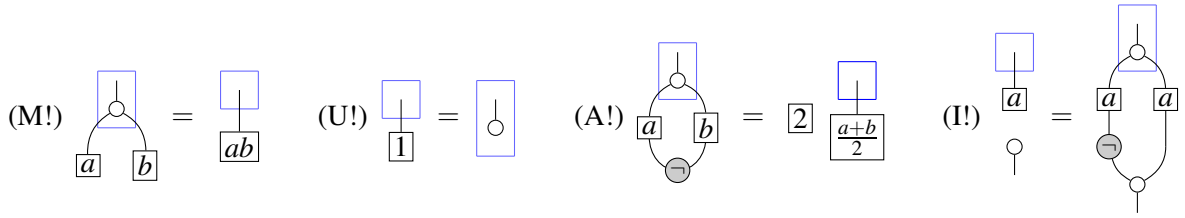
(3)

Additionally, the rules (ZS1), (HS1), (BA1), and (BA2) from Figure 1 become



Using the rules in this form makes it straightforward to prove !-box generalisations of the rules (M), (U), (A), and (I).

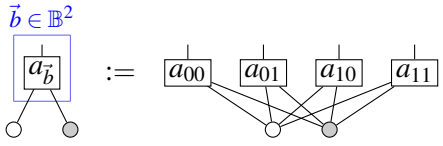
Lemma 2.3. The ZH-calculus satisfies the following rules:



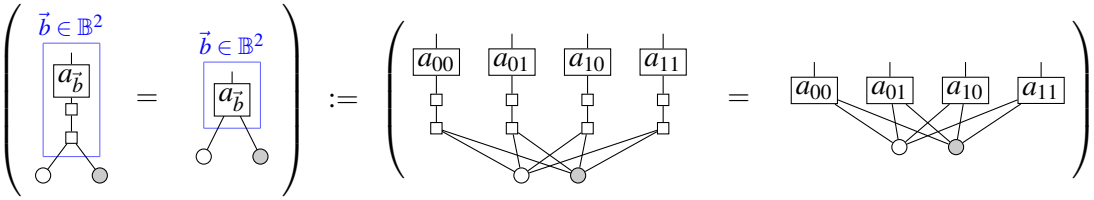
This lemma is proved in Appendix A. At this point, it is worth highlighting the special cases of (M!) and (U!) where the !-box is expanded 0 times:

These rules enable us to multiply scalars at will, and in particular eliminate scalars by multiplying by the inverse. From hence forth, we will use this fact without further comment during our proofs.

In this paper, we use a mild, but very useful, extension of the usual !-box notation, which allows !-boxes to be indexed by a the elements of a finite set. For example, indexing over the finite set $\mathbb{B}^2 := \{00, 01, 10, 11\}$, we can write expressions such as:

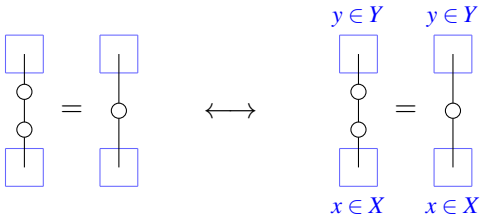


This extends to equations in the obvious way:



where we require corresponding !-boxes on the LHS and RHS to be indexed by the *same* finite set. Note that inputs and outputs of a copy associated with the index $x \in X$ on the LHS are matched with inputs and outputs of the *same* copy on the RHS.

We recover the behaviour of normal, un-labelled !-boxes by interpreting a !-box without a label as being indexed by an *arbitrary* finite set, e.g.

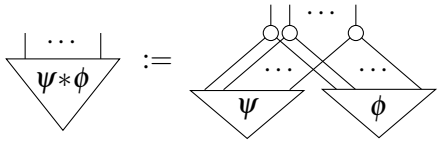


(for any finite sets X and Y)

3 Completeness

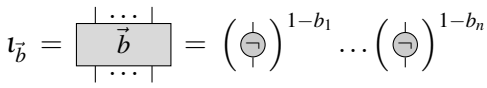
We show that the ZH-calculus is complete by demonstrating the existence of a unique normal form for ZH-diagrams. It is first worth noting that, because we can turn inputs into outputs arbitrarily (cf. the beginning of section 2.2), it suffices to consider diagrams which have only outputs. We call these *states*. Concretely, these are interpreted as column vectors (i.e. kets).

For states ψ, ϕ, let $\psi * \phi$ be the *Schur product* of ψ and ϕ obtained by plugging the i-th output of ψ and ϕ into  , for each i:



It follows from (ZS1) that $*$ is associative and commutative, so we can write k-fold Schur products $\psi_1 * \psi_2 * \ldots * \psi_k$ without ambiguity. For any finite set J with $|J| = k$, let $\prod_{j \in J} \psi_j$ be the k-fold Schur product.

Let $\mathbb{B}^n$ be the set of all n-bitstrings. For any $\vec{b} := b_1 \ldots b_n \in \mathbb{B}^n$, define the *indexing map* $\iota_{\vec{b}}$ as follows:



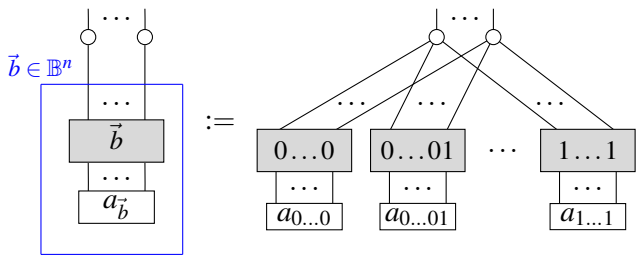
(4)

Then normal forms are given by the following 2^n-fold Schur products:

$$\prod_{\vec{b} \in \mathbb{B}^n} \left(\iota_{\vec{b}} \circ H_n(a_{\vec{b}})\right) \tag{5}$$

where $H_n(a_{\vec{b}})$ is the arity-n H-box (considered as a state) labelled by an arbitrary complex number $a_{\vec{b}}$.

A normal form diagram can be seen as a collection of n spiders, fanning out to 2^n H-boxes, each with a distinct configuration of NOT's corresponding to the 2^n bitstrings in $\mathbb{B}^n$. Diagrammatically, normal forms are:



Theorem 3.1. Normal forms are unique. In particular:

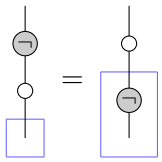
$$\left[\!\!\left[\prod_{\vec{b}\in\mathbb{B}^n} \left(\iota_{\vec{b}} \circ H_n(a_{\vec{b}})\right) \right]\!\!\right] = \sum_{\vec{b}\in\mathbb{B}^n} a_{\vec{b}} |\vec{b}\rangle. \tag{6}$$

Proof. The map $\iota_{\vec{b}}$ is a permutation that acts on computational basis elements as $|\vec{c}\rangle \mapsto |\vec{c} \oplus \vec{b} \oplus \vec{1}\rangle$. In particular, it sends the basis element $|\vec{1}\rangle$ to $\vec{b}$. Hence $\iota_{\vec{b}} \circ H_n(a_{\vec{b}})$ is a vector with $a_{\vec{b}}$ in the $\vec{b}$-th component and 1 everywhere else. The Schur product of all such vectors indeed gives the RHS of (6). □

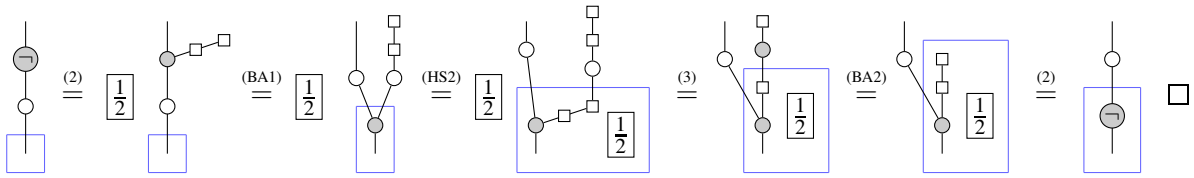
Since equation (6) gives us a means of constructing any vector in $\mathbb{C}^{2^n}$, Theorem 3.1 can also be seen as a proof of universality of the ZH calculus, independent of the encoding into ZX we gave in section 2.1.

We now prove 2 lemmas which will assist in manipulating normal forms:

Lemma 3.2. The NOT operator copies through white spiders:

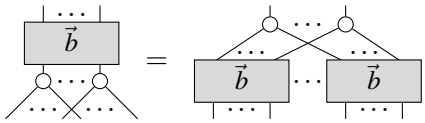


Proof. Starting from the left-hand side,



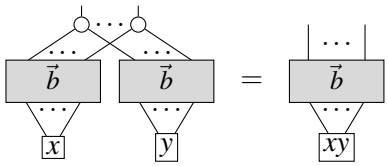
□

Lemma 3.3. The $\iota_{\vec{b}}$ operator copies through white spiders, i.e. for any $\vec{b} \in \mathbb{B}^n$:



Proof. This follows immediately from Lemma 3.2 via the definition of $\iota_{\vec{b}}$ (4). □

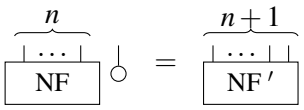
Lemma 3.4. The ZH-calculus enables the computation of the Schur product of two maps of the form $\iota_{\vec{b}} \circ H_n(x)$ and $\iota_{\vec{b}} \circ H_n(y)$ for any $\vec{b} \in \mathbb{B}^n$ and $x, y \in \mathbb{C}$:



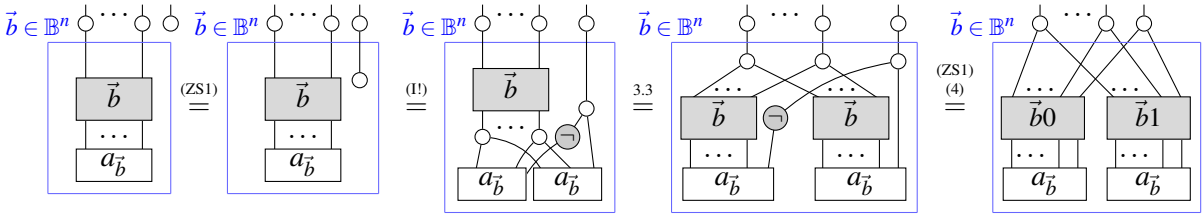
Proof. Apply Lemma 3.3, followed by (M!). □

We will now show that normal form diagrams, when combined in various ways, can also be put into normal form. Let  denote an arbitrary normal-form diagram. It is straightforward to see that permuting the outputs of a normal-form diagram merely interchanges the bits in the coefficients $a_{\vec{b}}$. Hence, normal forms are preserved under permutations of outputs. Furthermore:

Proposition 3.5. A diagram consisting of a normal form diagram juxtaposed with $\circ$ can be brought into normal form using the rules of the ZH-calculus:

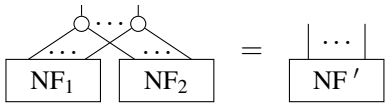


Proof. Starting from the left-hand side, which we expand using the indexed !-box notation,



The last diagram is a normal form diagram with $n+1$ outputs, i.e. the desired result. □

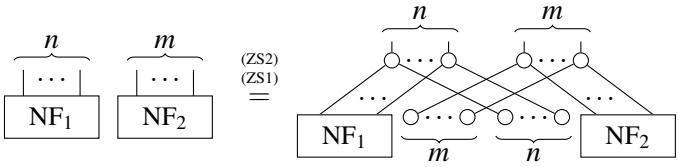
Proposition 3.6. The Schur product of two normal form diagrams can be brought into normal form using the rules of the ZH-calculus.



Proof. This follows from (ZS1) and Lemma 3.4. □

Corollary 3.7. The tensor product of two normal form diagrams can be brought into normal form using the rules of the ZH-calculus.

Proof. A tensor product can be expressed as

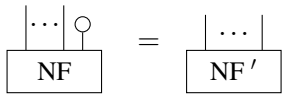


The diagram NF_1 and the leftmost m copies of $\circ$ can be combined into one normal-form diagram with $(n+m)$ outputs by successive applications of Proposition 3.5. Similarly, the rightmost n copies of $\circ$ and NF_2 can be combined into one normal-form diagram with $(n+m)$ outputs. The desired result then follows by Proposition 3.6. □

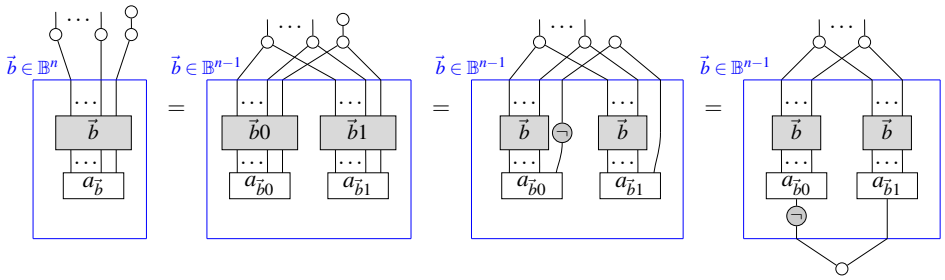
Remark 3.8. Note that a single scalar H-box is a normal form diagram. Corollary 3.7 thus implies that a diagram consisting of a normal form diagram juxtaposed with a scalar H-box can be brought into normal form. In the following proofs, we will therefore ignore scalars for simplicity: they can be added back in and then incorporated to the normal form without problems.

We are now ready to prove the most difficult case, which is contraction. The majority of the work goes into proving Lemma B.3, which we call the Disconnect Lemma. It uses the (O) rule to disconnect the 2^n-legged $\circ$-spider arising from a contraction of a normal form into 2^{n-1} separate cups. It was proven with the help of the graphical proof assistant Quantomatic. Details and full proof are given in Appendix B.

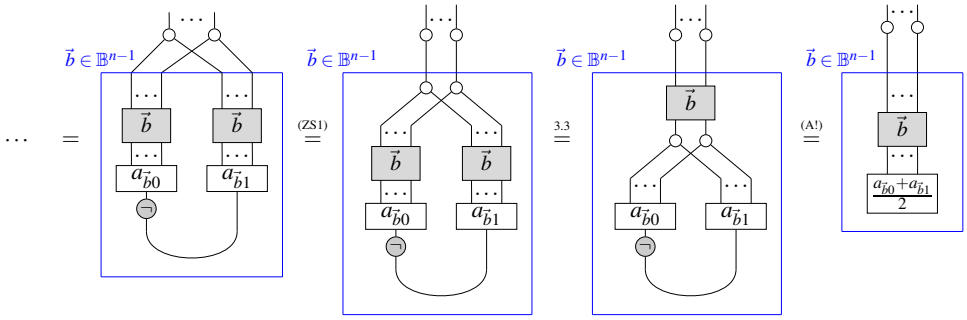
Proposition 3.9. The diagram resulting from applying ⏚ to an output of a normal form diagram can be brought into normal form:



Proof. Starting from an arbitrary normal form, with a ⏚ plugged into the right most output, we have:



Then, we can apply Lemma B.3:

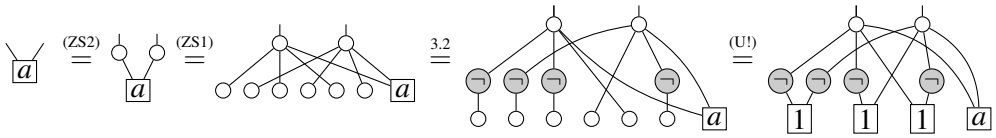


The final diagram is in normal form, which completes the proof. □

Our strategy will now be to show that any diagram can be decomposed into H-boxes, combined via the operations of extension, convolution, and contraction. This will give us a completeness proof, thanks to the following proposition.

Lemma 3.10. Any H-box can be brought into normal form using the rules of the ZH-calculus.

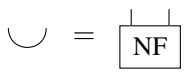
Proof. The matrix of an H-box $H_n(a)$ has 1's in every entry but the very last one. Hence, to bring an H-box into normal form, we just need to introduce 'dummy' 1's for every other matrix entry. We demonstrate the principle using a binary H-box but the argument is analogous for any other arity:



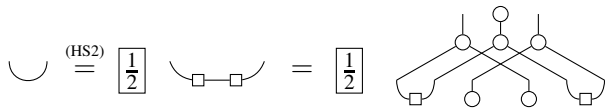
□

To simplify the decomposition of diagrams into H-boxes, we prove a few corollaries.

Corollary 3.11. The diagram of a single cup can be brought into normal form:

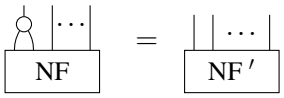


Proof. We can rewrite the cup as a pair of H-boxes using (HS2). This can then be written in terms of extension, convolution, and contraction as follows:



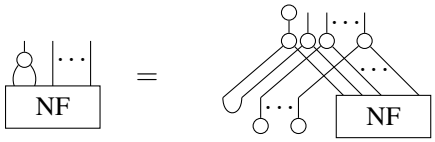
Hence, we can apply Lemma 3.10 and Propositions 3.5, 3.6, and 3.9 to get a normal form. □

Corollary 3.12. The diagram resulting from applying  to a pair of outputs of a normal form diagram can be brought into normal form.



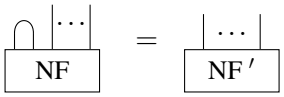
(7)

Proof. Applying a  to a pair of outputs has the same result as convolving with a cup, then contracting one of the outputs. That is, we can decompose (7) as follows:



then apply Corollary 3.11 and Propositions 3.5, 3.6, and 3.9. □

Corollary 3.13. Applying a cap to a normal form diagram results in another normal form diagram:



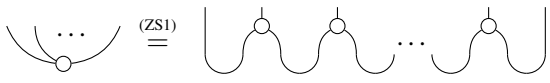
Proof. Since the cap can be decomposed as  , the result follows immediately from Corollary 3.12 and Proposition 3.9. □

Thanks to Corollaries 3.7 and 3.13, we are able to turn any diagram of normal forms into a normal form. It only remains to show that the generators of the ZH-calculus can themselves be made into normal forms. We have already shown the result for H-boxes, so we only need the following.

Lemma 3.14. Any Z-spider can be brought into normal form using the rules of the ZH-calculus.

Proof. We can turn  into an H-box using (U) and then bring it into normal form via Lemma 3.10. By (ZS1),  , which can be brought into normal form using (U), Lemma 3.10, and Corollaries 3.7 and 3.13. This covers the cases of Z-spiders with 0 or 1 incident wires.

We can decompose any Z-spider with $n \geq 2$ incident wires as a tensor product of $(n-1)$ cups, with each cup  -ed with its neighbours:



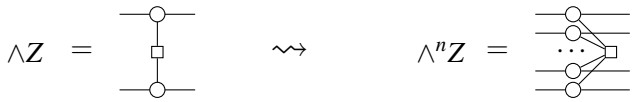
If $n = 2$, no  are needed and the equality is by (ZS2) instead of (ZS1). In either case, the diagram can be brought into normal form by applying Corollaries 3.7, 3.11, and 3.12. □

Theorem 3.15. The ZH-calculus is complete: for any ZH diagrams D_1 and D_2, if $[\![D_1]\!] = [\![D_2]\!]$ then D_1 is convertible into D_2 using the rules of the ZH-calculus.

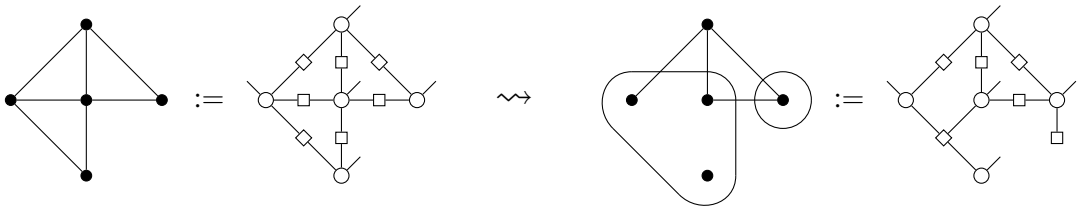
Proof. By Theorem 3.1, it suffices to show that any ZH diagram can be brought into normal form. Lemmas 3.10 and 3.14 suffice to turn any generator into normal form. Corollary 3.7 lets us turn any tensor product of generators into a normal form and Corollary 3.13 lets us normalise any arbitrary wiring. □

4 Applications and future work

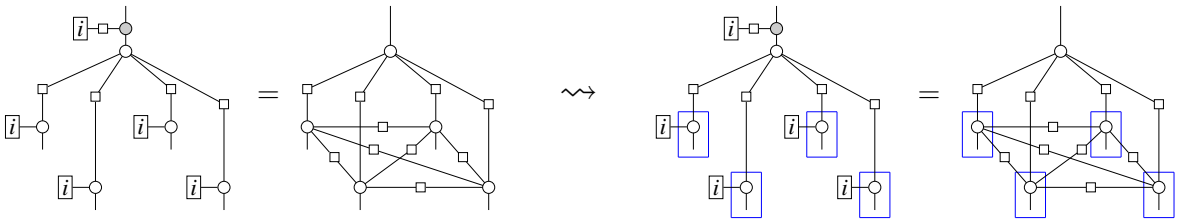
We will now briefly survey some of the potential applications for the ZH-calculus. We begin with the simple observation that n-ary H-boxes let us generalise the usual string diagrammatic description the controlled-Z gate (as in e.g. the ZX calculus) to an n-controlled-Z gate:



Using the decomposition of controlled-Z gates above, a representation of graph states as ZX-diagrams was given in [10], which in turn gave a fully diagrammatic derivation of the local complementation law for graph states [10] and a new procedure for extracting circuits from certain computations in the one-way model [11]. Passing from $\wedge Z$ to $\wedge^n Z$ gives an analogous representation for *hypergraph states*:



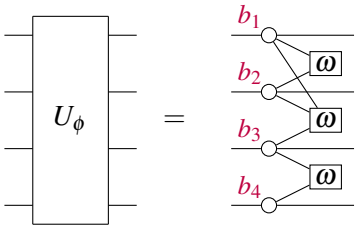
Indeed this was the original motivation for considering H-boxes of arbitrary arity. Using a method analogous to proofs in Appendix A, we can routinely introduce !-boxes to known rules involving graph states (e.g. local complementation and feed-forward rules) to generalise them to hypergraph states. For example, introducing !-boxes to the local complementation rule enables complementing hyperedges of arbitrary arity overlapping on a single vertex:



This potentially gives a powerful new language and set of techniques for working with hypergraph states. Exploring these techniques, and the relationship to known rules for manipulating hypergraph states is a topic of future work.

In another direction, if we consider diagrams whose H-boxes are labelled by a fixed root of unity $\omega := \exp(i\pi/2^m)$, we obtain an encoding for unitary gates described by arbitrary *phase polynomials* [2], i.e. gates of the form $U_\phi|\vec{b}\rangle = \omega^{\phi(\vec{b})}|\vec{b}\rangle$ for some polynomial $\phi(\vec{b})$ over n boolean variables. These have a simple graphical representation, where Z-spiders represent variables and ω-labelled H-boxes represent

terms in the phase polynomial. For example:



where $\phi(\vec{b}) = b_1b_2 + b_1b_2b_3 + b_3b_4$

One can then straightforwardly show basic properties of these unitaries (e.g. composition, commutation, and replacement of non-linear AND terms by linear XOR terms) using the rules of the ZH-calculus. The phase polynomial formalism for $m = 2$ has been used extensively in studying optimisation problems for Clifford+T circuits [3, 2, 4, 15], and it was recently shown that all diagonal gates in the Clifford hierarchy are of the form U_ϕ, where the level of the hierarchy depends on m and the degree of ϕ [9]. Gaining access to this phase polynomial structure diagrammatically could therefore yield new methods for quantum circuit optimisation and/or fault tolerant computation through automated diagram rewriting in tools like Quantomatic.

Acknowledgements. The authors would like to thank Simon Perdrix and Mariami Gachechiladze for the fruitful conversations in which the foundations of the ZH-calculus were developed. We are also grateful to Niel de Beaudrap for interesting discussions about applications of the ZH-calculus and to Sal Wolffs for careful reading of our proofs (and pointing out a major omission in Corollary 3.12).

The research leading to these results has received funding from the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013) ERC grant agreement no. 334828 (Backens) and 320571 (Kissinger). The paper reflects only the authors' views and not the views of the ERC or the European Commission. The European Union is not liable for any use that may be made of the information contained therein.

References

[1] Dorit Aharonov (2003): *A simple proof that Toffoli and Hadamard are quantum universal*. arXiv:quant-ph/0301040.

[2] M. Amy, D. Maslov & M. Mosca (2014): *Polynomial-Time T-Depth Optimization of Clifford+T Circuits Via Matroid Partitioning*. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 33(10), pp. 1476–1489, doi:10.1109/TCAD.2014.2341953.

[3] M. Amy, D. Maslov, M. Mosca & M. Roetteler (2013): *A Meet-in-the-Middle Algorithm for Fast Synthesis of Depth-Optimal Quantum Circuits*. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 32(6), pp. 818–830, doi:10.1109/TCAD.2013.2244643.

[4] M. Amy & M. Mosca (2016): *T-count optimization and Reed-Muller codes*. arXiv:1601.07363.

[5] Niel de Beaudrap: *A toy theory of tensor networks for exact quantum algorithms*. To appear.

[6] B. Coecke & R. Duncan (2008): *Interacting quantum observables*. In: *Proceedings of the 37th International Colloquium on Automata, Languages and Programming (ICALP)*, Lecture Notes in Computer Science, doi:10.1007/978-3-540-70583-3_25.

[7] B. Coecke & R. Duncan (2011): *Interacting quantum observables: categorical algebra and diagrammatics*. *New Journal of Physics* 13, p. 043016, doi:10.1088/1367-2630/13/4/043016. arXiv:quant-ph/09064725.

[8] B. Coecke & A. Kissinger (2017): *Picturing Quantum Processes: A First Course in Quantum Theory and Diagrammatic Reasoning*. Cambridge University Press, doi:10.1017/9781316219317.

[9] Shawn X. Cui, Daniel Gottesman & Anirudh Krishna (2017): *Diagonal gates in the Clifford hierarchy*. *Phys. Rev. A* 95, p. 012329, doi:10.1103/PhysRevA.95.012329.

[10] R. Duncan & S. Perdrix (2009): *Graph states and the necessity of Euler decomposition*. *Mathematical Theory and Computational Practice*, pp. 167–177, doi:10.1007/978-3-642-03073-4_18.

[11] R. Duncan & S. Perdrix (2010): *Rewriting measurement-based quantum computations with generalised flow*. In: *Proceedings of ICALP*, Lecture Notes in Computer Science, Springer, pp. 285–296, doi:10.1007/978-3-642-14162-1_24.

[12] Ross Duncan (2013): *A graphical approach to measurement-based quantum computing*. In Chris Heunen, Mehrnoosh Sadrzadeh & Edward Grefenstette, editors: *Quantum Physics and Linguistics*, OUP, doi:10.1093/acprof:oso/9780199646296.003.0003. arXiv:1203.6242 [quant-ph].

[13] Mariami Gachechiladze, Costantino Budroni & Otfried Gühne (2016): *Extreme violation of local realism in quantum hypergraph states*. *Physical Review Letters* 116(7), p. 070401, doi:10.1103/PhysRevLett.116.070401.

[14] Amar Hadzihasanovic: *The algebra of entanglement and the geometry of composition*. PhD thesis, University of Oxford. arXiv:1709.08086.

[15] Luke E Heyfron & Earl T Campbell (2018): *An efficient quantum compiler that reduces T count*. *Quantum Science and Technology* 4(1), p. 015004, doi:10.1088/2058-9565/aad604.

[16] Emmanuel Jeandel, Simon Perdrix & Renaud Vilmart (2018): *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 '18, ACM, New York, NY, USA, pp. 559–568, doi:10.1145/3209108.3209131.

[17] Emmanuel Jeandel, Simon Perdrix & Renaud Vilmart (2018): *Y-Calculus: A Language for Real Matrices Derived from the ZX-Calculus*. *Electronic Proceedings in Theoretical Computer Science* 266, pp. 23–57, doi:10.4204/EPTCS.266.2. Version 1 at arXiv:1702.00934v1.

[18] Aleks Kissinger, Alex Merry & Matvey Soloviev (2014): *Pattern graph rewrite systems*. *Electronic Proceedings in Theoretical Computer Science* 143, pp. 54–66, doi:10.4204/EPTCS.143.5.

[19] Aleks Kissinger & Vladimir Zamdzhiev (2015): *Quantomatic: A proof assistant for diagrammatic reasoning*. In: *International Conference on Automated Deduction*, Springer, pp. 326–336, doi:10.1007/978-3-319-21401-6_22.

[20] Jacob Miller & Akimasa Miyake (2016): *Hierarchy of universal entanglement in 2D measurement-based quantum computation*. *Npj Quantum Information* 12(16036), doi:10.1038/npjqi.2016.36.

[21] Kang Feng Ng & Quanlong Wang (2017): *A universal completion of the ZX-calculus*. arXiv:1706.09877.

[22] R. Raussendorf, D.E. Browne & H.J. Briegel (2003): *Measurement-based quantum computation on cluster states*. *Physical Review A* 68(2), p. 22312, doi:10.1103/PhysRevA.68.022312.

[23] M Rossi, D Bruß & C Macchiavello (2014): *Hypergraph states in Grover's quantum search algorithm*. *Physica Scripta* 2014(T160), p. 014036, doi:10.1088/0031-8949/2014/T160/014036.

[24] Matteo Rossi, M Huber, D Bruß & C Macchiavello (2013): *Quantum hypergraph states*. *New Journal of Physics* 15(11), p. 113022, doi:10.1088/1367-2630/15/11/113022.

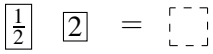
[25] P. Selinger (2007): *Dagger compact closed categories and completely positive maps*. *Electronic Notes in Theoretical Computer Science* 170, pp. 139–163, doi:10.1016/j.entcs.2006.12.018.

[26] Yaoyun Shi (2003): *Both Toffoli and controlled-NOT Need Little Help to Do Universal Quantum Computing*. *Quantum Info. Comput.* 3(1), pp. 84–92.

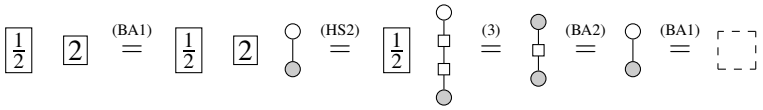
A Proofs of the !-box versions of basic rules

The following Lemma will be useful for the later results.

Lemma A.1. The scalar H-box with phase 2 and the scalar H-box with phase $\frac{1}{2}$ are inverses of each other.

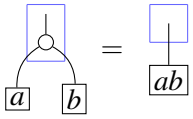


Proof. Starting from the left-hand side, we have

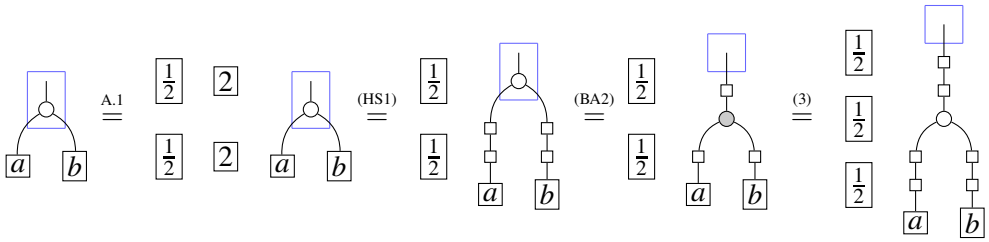


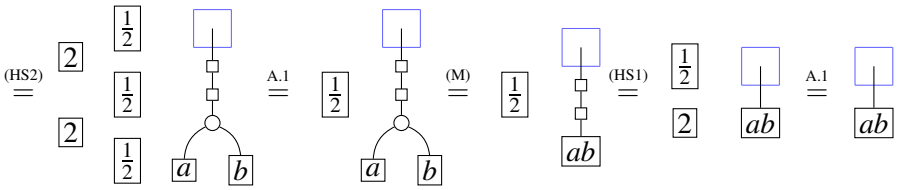
□

Lemma A.2. The following !-box generalisation of (M) holds in the ZH-calculus.



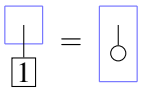
Proof. Starting from the left-hand side, we have



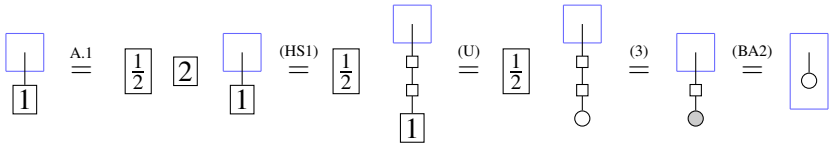


□

Lemma A.3. The phase-1 H-box of any arity decomposes:



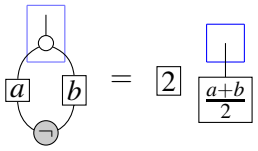
Proof. Again, starting from the left-hand side,



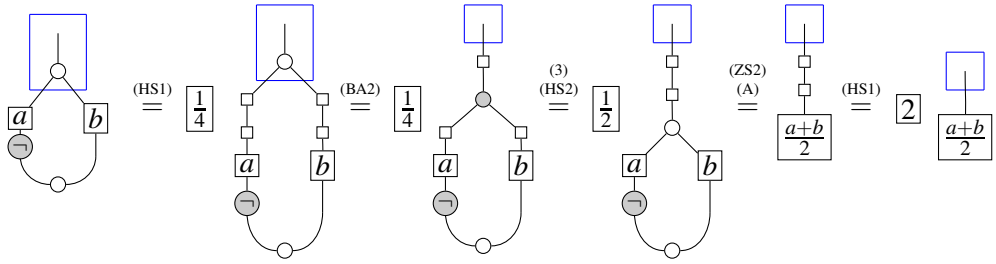
□

As we noted at the end of Section 2.3, Lemmas A.2 and A.3 allow us to combine and cancel scalars at will. We will do this for the remainder of the proofs.

Lemma A.4. The following !-box generalisation of (A) holds in the ZH-calculus.

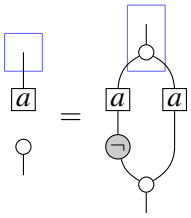


Proof. Starting from the left-hand side,

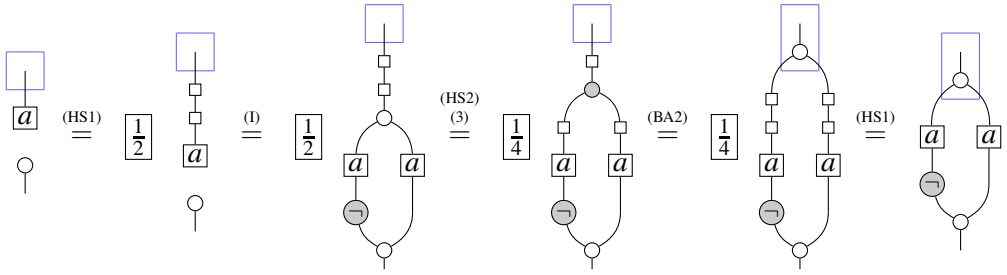


□

Lemma A.5. The introduction rule (I) holds for H boxes with arbitrarily many inputs:



Proof. As usual, we start from the left-hand side:



□

Combining Lemmas A.2–A.5 gives a proof of Lemma 2.3.

B Proof of the disconnect lemma

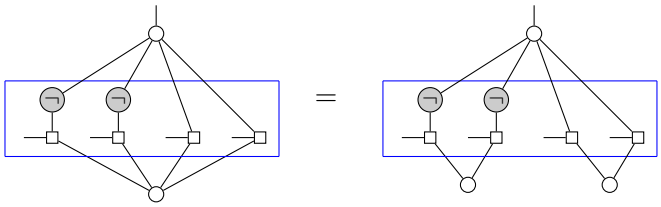
The proof of the disconnect lemma was produced with the aid of Quantomatic. In particular, Lemmas B.1 and B.3 are exported from the Quantomatic derivations `disconnect-4` and `gen-disconnect-4` in the `zh` project, available here:

https://github.com/Quantomatic/sample-projects/tree/qpl2018

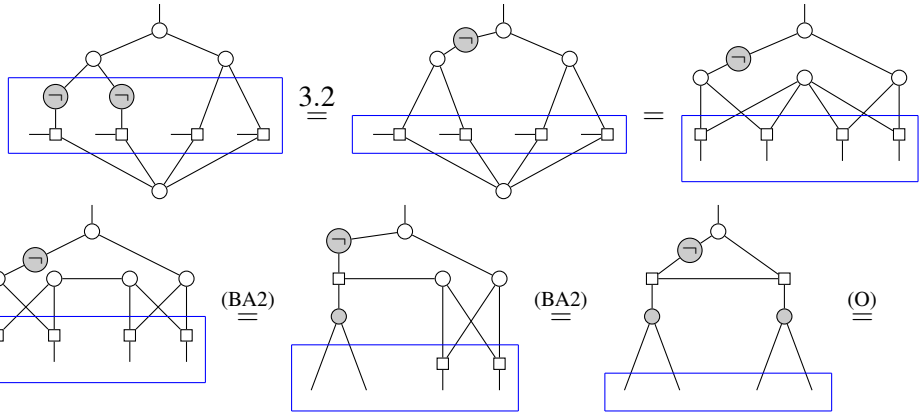
It should be viewable in the latest version of Quantomatic. However, for the sake of posterity, the version of `Quantomatic.jar` used to make this project has also been uploaded here:

https://github.com/Quantomatic/sample-projects/releases/tag/qpl2018

Lemma B.1.

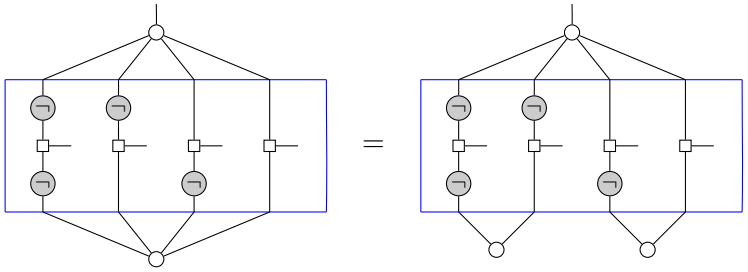


Proof.

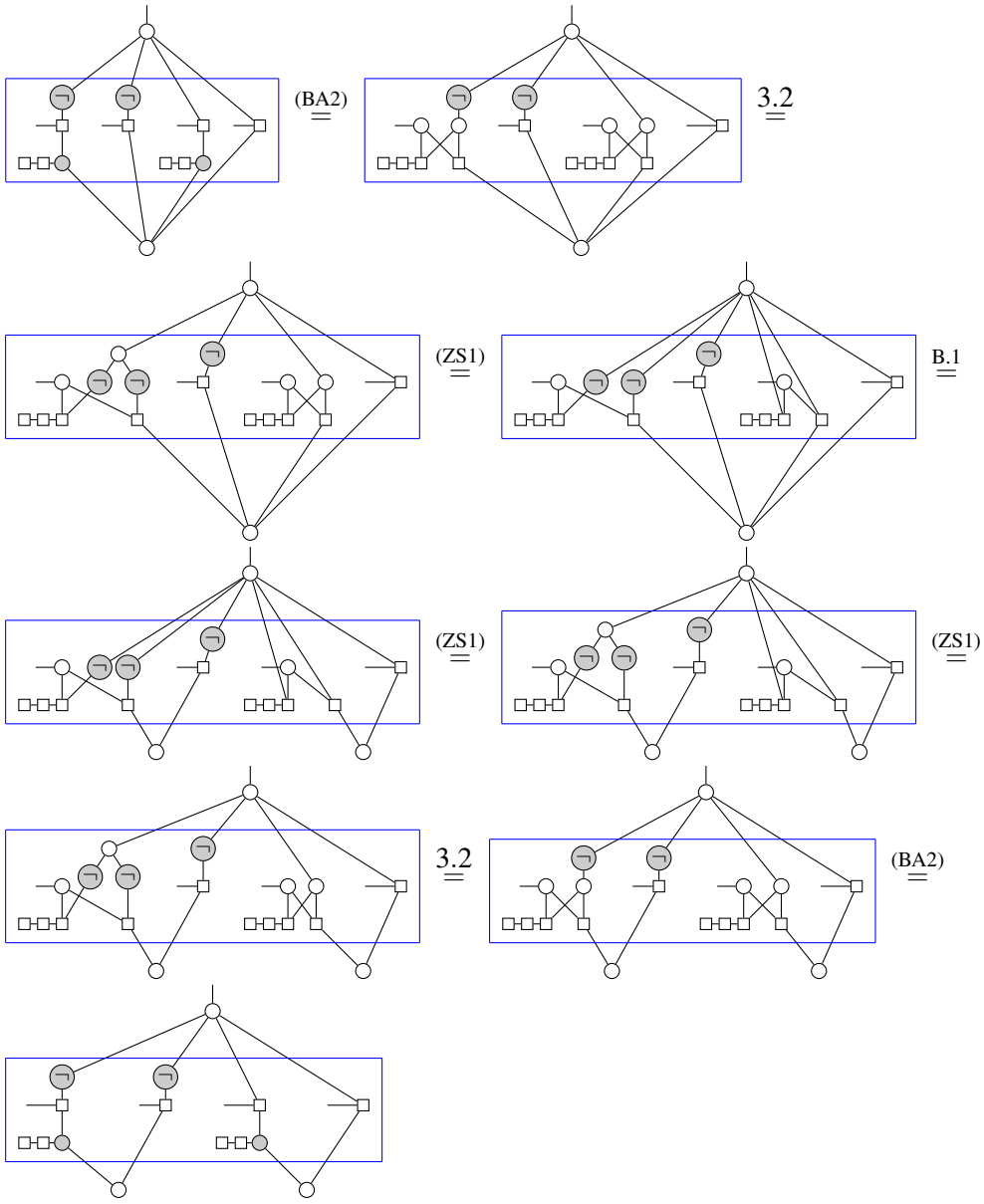


□

Lemma B.2.

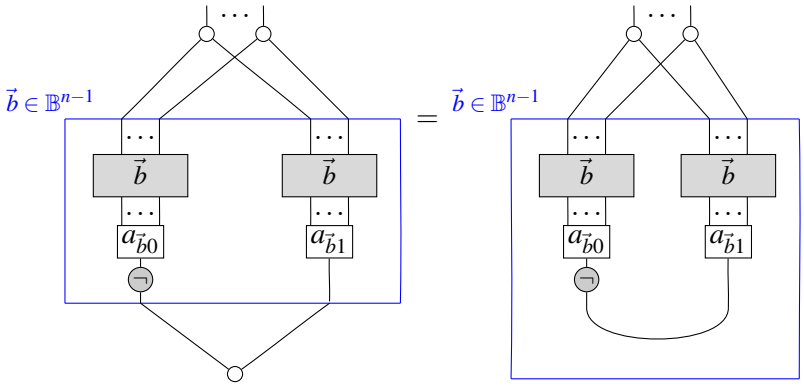


Proof. We begin by decomposing the bottom two nodes on the left-hand side using (2). Then:



This yields the desired result after reintroducing the ⊖ nodes via (2). □

Lemma B.3.

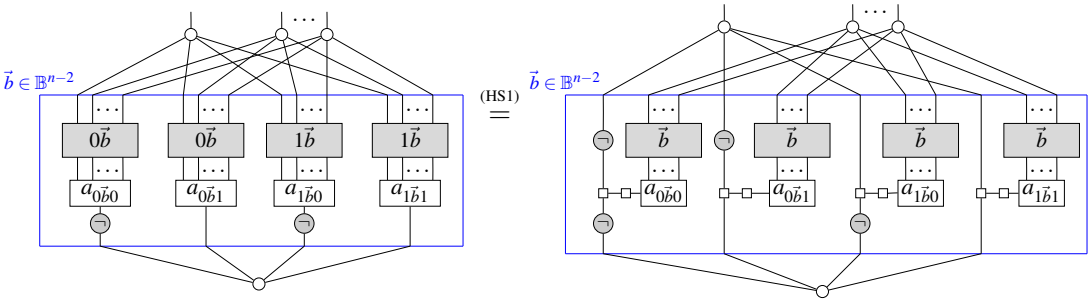


(8)

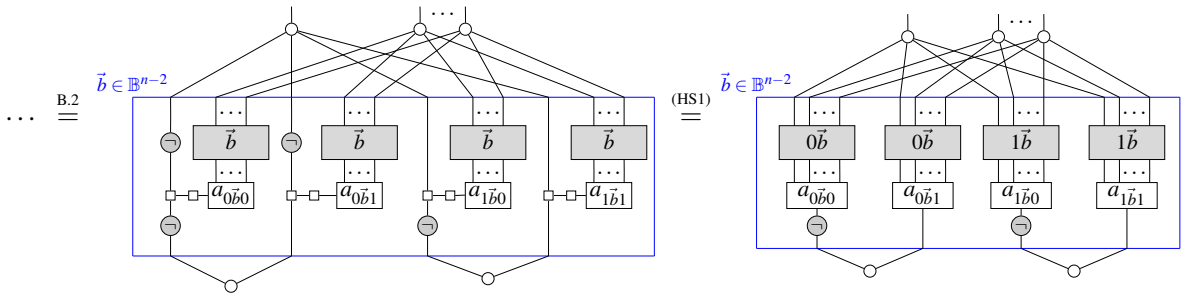
Proof. First, note that we can split the set $\mathbb{B}^{n-1}$ into two pieces, based on whether the most significant bit is 0 or 1:

$$\mathbb{B}^{n-1} = \{0\vec{b} \mid \mathbb{B}^{n-2}\} \uplus \{1\vec{b} \mid \mathbb{B}^{n-2}\}$$

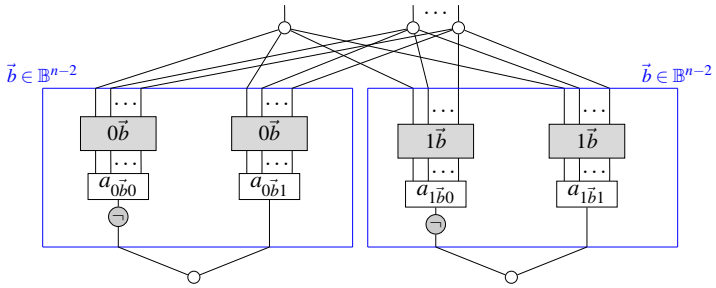
Hence, the LHS can be equivalently written as:



We can then apply Lemma B.2 to split the bottom spider in two:



We can then split the !-box into two parts, indexed over the same set to obtain:



We now have two copies of a graph which is very similar to the LHS of (8), but for one fewer bit. We can thus repeat the process above to split each of the two spiders using the second bit of the bitstring, then the third, and so on, until $\circ$-spiders only connect pairs of H-spiders that disagree on the least significant bit. Replacing 2-legged $\circ$-spiders with cups, we obtain the RHS of (8). $\square$