Duncan, Ross and Lucas, Maxime (2014) Verifying the Steane code with Quantomatic. Electronic Proceedings in Theoretical Computer Science, 171. pp. 33-49. ISSN 2075-2180 , http://dx.doi.org/10.4204/EPTCS.171.4

Verifying the Steane code with Quantomatic

Ross Duncan
University of Strathclyde
Glasgow, UK
ross.duncan@strath.ac.uk

Maxime Lucas
Université Libre de Bruxelles
Brussels, Belgium
mlucas@ulb.ac.be

In this paper we give a partially mechanized proof of the correctness of Steane's 7-qubit error correcting code, using the tool Quantomatic. To the best of our knowledge, this represents the largest and most complicated verification task yet carried out using Quantomatic.

1 Introduction

Even more so than their classical equivalents, quantum information processing technologies are susceptible to noise and other errors, which can destroy stored data and render computations meaningless. For this reason, quantum error-correcting codes are seen as a crucial ingredient of any practical quantum information scheme.

Conceptually, error-correcting codes are simple: the space of possible messages is embedded in a larger code-space. If an error occurs on the encoded message, it will (hopefully) be mapped to an element of the code-space that does not correspond to any message. By careful choice of the code-space, and if the error rate is not too high, it should be possible to recover the original message.

For example, consider the classical repetition code. Here a bit is mapped to a code-space consisting of three bits:

$$0 \mapsto 000 \qquad\qquad 1 \mapsto 111$$

If a single bit of the codeword is flipped, then the original encoded bit can be recovered by taking the majority of the three bits. Notice that this code only protects against the specific error model where only one bit can be flipped: if two (or more) bits are flipped then the code cannot correct the error and the encoded bit will be flipped.

The classical case is simpler than the quantum case[1] for two reasons: the only possible errors are bit flips; and, it is possible to measure a classical bit without changing its value. However neither of these problems is fatal. Although the errors which may afflict a qubit might be any unitary map, it suffices to coherently correct Pauli X and Z errors, since these generate all the others. To detect the errors in the first place, quantum codes typically entangle the code-word with an ancilla and measure the ancilla. For the full details see, for example, [7], and references therein.

While the simple repetition code described above can be adapted for the quantum case, at a cost of 9 qubits for each encoded qubit, we will consider a 7-qubit code due to Andrew Steane [8]. Like the repetition code, the Steane code can only correct single qubit errors[2]. Practically speaking the code consists of three circuits: an encoding circuit, a decoding circuit (adjoint to the encoder), and an error correcting circuit that tests the code-space for errors and corrects them.

[1] In fact, we consider the simplest possible case for both: errors affect only a single bit/qubit. This is not terribly realistic.

[2] This is not exactly true. As we shall see later, the code can correct a single Z error and/or a single X error, but they do not have to affect the same qubit.

M.J. Hoban, B. Coecke (Eds.): 10th International
Workshop on Quantum Physics and Logic (QPL 2013)
EPTCS 171, 2014, pp. 33–49, doi:10.4204/EPTCS.171.4

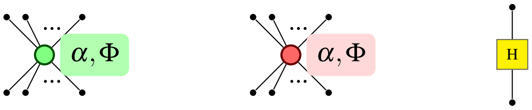


Figure 1: Permitted interior vertices

In the rest of the paper, the three circuits mentioned above will be translated into the graphical formalism of the ZX-calculus [1]. Using the formalism, as implemented in the automated rewriting system *Quantomatic* [2], the circuits are simplified. We then, again by rewriting, derive the error syndrome conditions for the correction circuit, and verify that it does indeed correct the claimed errors. (It is remarkable how simple and clear this graphical proof is, compared to the usual textbook approach.) Finally, we show that the combined circuit—encoder-corrector-decoder—rewrites to the identity on one qubit, proving it does not introduce any errors itself as a consequence of its design.

2 The ZX-calculus

The ZX-calculus is a formal language for reasoning about quantum computational systems. It comprises a graphical syntax and a set of axioms presented as rewrite rules. It has a standard semantics in Hilbert spaces, and can represent any quantum circuit. We briefly review the ZX-calculus here; for full detalis consult [1]; here we make use of the ZX-calculus extended with conditional vertices, as elaborated in [5, 6].

Definition 2.1. Let S be some set of variables. A *diagram over S* is an open graph whose interior vertices are restricted to the following types:

- Z-vertices, labelled by an angle $\alpha \in [0, 2\pi)$ and a boolean formula Φ with variables from S; these are shown as (light) green circles,
- X vertices, labelled by an angle $\alpha \in [0, 2\pi)$ and a boolean formula Φ with variables from S; these are shown as (dark) red circles,
- H (or Hadamard) vertices, restricted to degree 2; shown as squares.

The allowed vertices are shown in Figure 1.

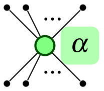
If an X or Z vertex is labelled by $\alpha = 0$ then the label is omitted. If the labelling formula Φ is not constant then the vertex is called *conditional*; otherwise it is unconditional. We omit the formulae from unconditional vertices.

For each S, the diagrams over S form a symmetric monoidal category (in fact compact closed) in the obvious way: the objects of the category are natural numbers and an arrow $g : m \to n$ is a diagram with m input and n outputs. The tensor product is juxtaposition, and composition $g \circ f$ is defined by identifying the output vertices of f with the input vertices of g. For more details see [4, 3]. Denote this category $\mathbb{D}(S)$; we denote the category $\mathbb{D}(\emptyset)$ of unconditional diagrams by $\mathbb{D}$. Note that the components shown in Figure 1 are the generators of $\mathbb{D}(S)$.

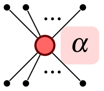
Definition 2.2. Let $v : S \to \{0,1\}$ be any boolean function; it naturally assigns a truth value $v(\Phi)$ to any formula Φ over S. We define a *valuation* functor $\hat{v} : \mathbb{D}(S) \to \mathbb{D}$ by relabelling the Z and X vertices as follows:

$$\alpha, \Phi \mapsto \begin{cases} \alpha & \text{if } v(\Phi) = 1 \\ 0 & \text{otherwise} \end{cases}$$

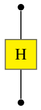
Definition 2.3. Let $\llbracket\cdot\rrbracket : \mathbb{D} \to \mathbf{fdHilb}$ be a traced monoidal functor; define its action on objects by $\llbracket n \rrbracket = \mathbb{C}^{2^n}$; define its action on the generators as:

$\llbracket$  $\rrbracket$

$$= \left\{ \begin{array}{lcl} |0\rangle^{\otimes m} & \mapsto & |0\rangle^{\otimes n} \\ |1\rangle^{\otimes m} & \mapsto & e^{i\alpha}|1\rangle^{\otimes n} \end{array} \right.$$

$\llbracket$  $\rrbracket$

$$= \left\{ \begin{array}{lcl} |+\rangle^{\otimes m} & \mapsto & |+\rangle^{\otimes n} \\ |-\rangle^{\otimes m} & \mapsto & e^{i\alpha}|-\rangle^{\otimes n} \end{array} \right.$$

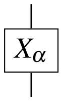
$\llbracket$  $\rrbracket$

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

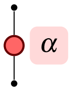
Definition 2.4. The denotation of a diagram D over variables S is a superoperator constructed by summing over all the valuations of S:

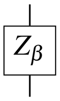
$$\rho \mapsto \sum_{v \in 2^S} \llbracket \hat{v}(D) \rrbracket \rho \llbracket \hat{v}(D) \rrbracket^\dagger .$$

Example 2.5 (Unitary gates)**.** Any quantum circuit can be written in the ZX-calculus. This is most easily seen via the translation of a universal gate set:

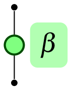


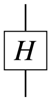
$$= \begin{pmatrix} \cos\frac{\alpha}{2} & -i\sin\frac{\alpha}{2} \\ -i\sin\frac{\alpha}{2} & \cos\frac{\alpha}{2} \end{pmatrix} =$$

$\llbracket$  $\rrbracket$

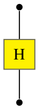


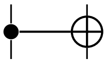
$$= \begin{pmatrix} 1 & 0 \\ 0 & e^{i\beta} \end{pmatrix} =$$

$\llbracket$  $\rrbracket$

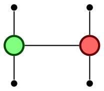


$$= \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} =$$

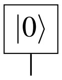
$\llbracket$  $\rrbracket$



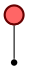
$$= \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}. =$$

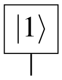
$\llbracket$  $\rrbracket$

Of course, one easily incorporates state preparations as well. For example:

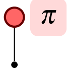


$$= \begin{pmatrix} 1 \\ 0 \end{pmatrix} =$$

$\llbracket$  $\rrbracket$

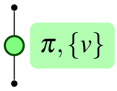


$$= \begin{pmatrix} 0 \\ 1 \end{pmatrix} =$$

$\llbracket$  $\rrbracket$

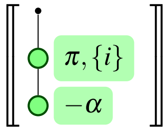
Notice that these components only require *uncondtional* diagrams.

Example 2.6 (Measurements and classical control)**.** A classically controlled operation is encoded using a conditional operator. For example, the following diagram represents a Pauli Z which is turned on by the classical data stored in the variable v.

$\llbracket$  $\rrbracket$

$$= \{\, \rho \mapsto \mathbb{1}\rho\mathbb{1} + Z\rho Z \,\}$$

We represent measurements by combining a conditional Pauli and a projection. Let $|\alpha_\pm\rangle = (|0\rangle \pm e^{i\alpha}|1\rangle)/\sqrt{2}$, then



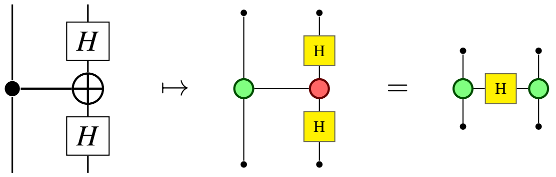
$$= \left\{ \rho \mapsto \langle\alpha_+|(\mathbb{1}\rho\mathbb{1} + Z\rho Z)|\alpha_+\rangle \;=\; \langle\alpha_+|\rho|\alpha_+\rangle + \langle\alpha_-|\rho|\alpha_-\rangle \right\}$$

is the measurement in the basis $|\alpha_\pm\rangle$. Notice that here the variable i represents the *outcome* of the measurement; hence to properly encode a measurement it should always be a *fresh variable*.

Axioms. The ZX-calculus calculus is also an equational theory; its axioms are shown in Figure 2. Two key properties are:

- *Soundness:* If $D = D'$ via the equational rules, then $[\![D]\!] = [\![D']\!]$.
- *Colour Duality*: If $D_1 = D_2$ is a derivable equation in the ZX-calculus then $D'_1 = D'_2$ is also derivable, where D'_i is obtained from D_i by exchanging the colours red and green.

Example 2.7 (The $\wedge Z$-gate)**.** The $\wedge Z$-gate can be obtained by using a Hadamard gate to transform the target bit of a $\wedge X$ gate.

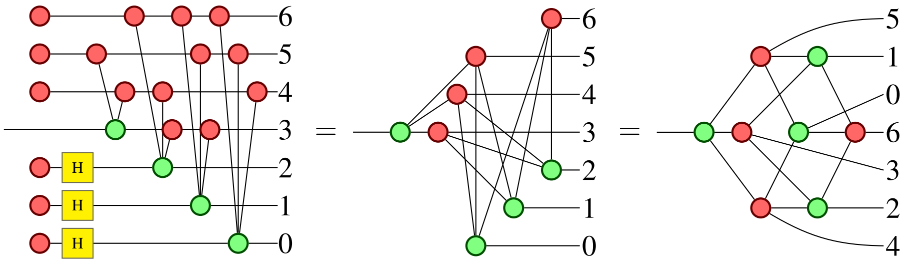


3 Translation and Simplification

In this section we describe how the encoding and correcting circuits of the Steane code can be reprsented in ZX-calculus, and perform some simplifications. We will not at any point give a traditional *definition* of the code other than these circuits; the discussion of Section 4 will demonstrate quite clearly how it works. **NB.** From this point onwards, we adopt the convention that diagrams are oriented with the inputs to the *left* side and outputs to the *right*.

The encoder The encoder maps a single input qubit to the 7-qubit code-space via the circuit shown in Figure 3 (a). Note that qubit 3 is the input. Using the translation described in Example 2.5 we obtain the ZX-calculus term shown in Figure 3 (b).

The encoding circuit is simple enough that Quantomatic can reduce it to its minimal form without human intervention. The resulting simplification is shown below; the final equation is merely a rearrangment of the vertices to show the structure more clearly.



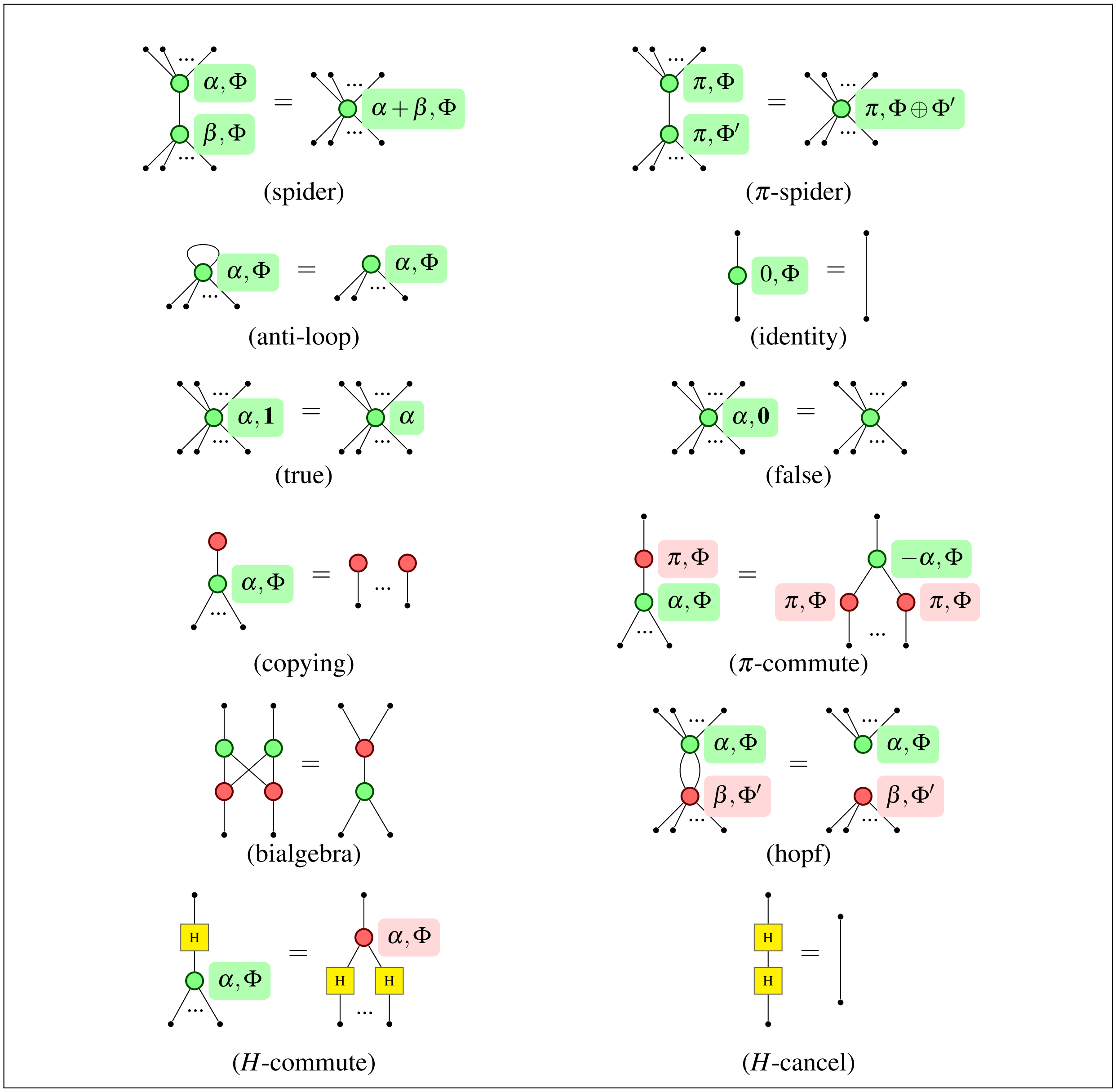


Figure 2: Equational rules for the ZX-calculus. We present the rules for the Z subsystem; to obtain the complete set of rules exchange the colours in the rules shown above.

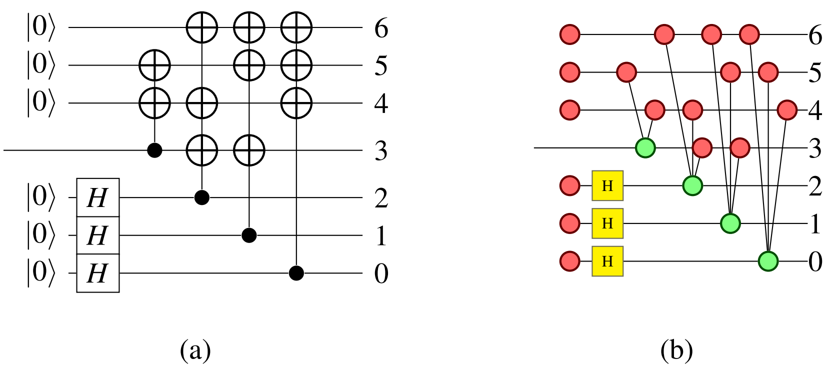


Figure 3: The encoder (a) as a circuit; (b) in the ZX-calculus.

The decoding circuit is simply the encoder in reverse.

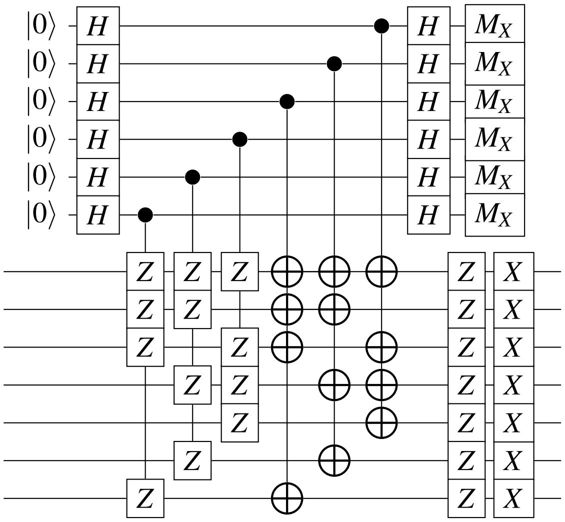


Figure 4: The circuit to correct errors. Note that the conditions to activate the Pauli X and Z corrections are not shown.

The error corrector An encoder is not very useful if we cannot detect and correct errors on the encoded data. The circuit for carrying out this function is shown in Figure 4. Notice that the circuit naturally splits into two parts. The *error-detecting* part introduces a 6-qubit ancilla, entangles it with the input, and then measures the ancillae. The resulting measurements are called the *error syndrome*. The *error-correcting* part comprises the Pauli operations at the end, which are applied *conditionally*, depending on the value of the error syndrome. One crucial detail has been omitted from this picture: we do not show the conditions which control the error-correcting part. We will derive these conditions in the next section.

We translate the circuit into the ZX-calculus as shown in Figure 5. The variables $a, b, \ldots, f$ indicate the outcomes of the six syndrome measurments, while the variables x_i and z_i indicate whether a Pauli X or Z correction need be applied on qubit i of the codeword.

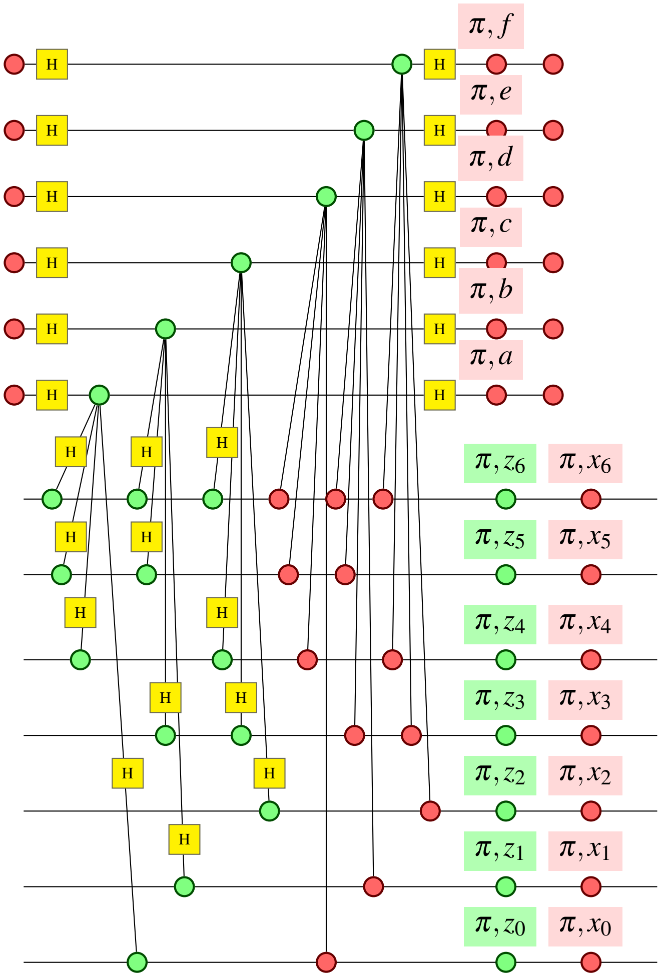
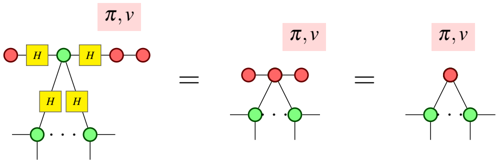


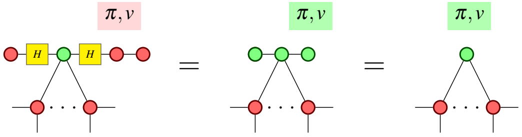
Figure 5: The correcting circuit in the ZX-calculus

This diagram can be substantially simplified; to do so we rely on some easy lemmas:

Lemma 3.1.



Lemma 3.2.



By three applications of each lemma, the correcting circuit rewrites to the diagram in Figure 6. Notice that this is not a minimal form, however it is well adapted to the analysis of the next section.

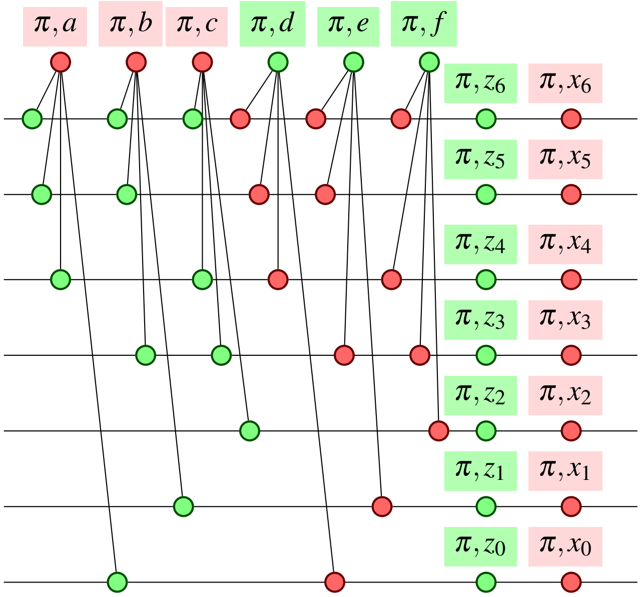


Figure 6: The simplified correcting circuit

4 Verification

In this section we will demonstrate the correctness of the Steane error-correcting code. There are two parts to this argument. Firstly we show that the correcting circuit can remove single qubit Pauli errors; and secondly, we show that the composition of encoder, corrector, and decoder will reduce to the identity for one qubit.

Consider again the error-correcting circuit in Figure 6. The Pauli Z on qubit i must be activated exactly when there is a Z-error on qubit i: therefore the variable z_i must be true exactly in this case. However the only available information are the syndrome measurements $a, b, \dots, f$. Therefore the first step is to derive the value of the z_i and x_i in terms of these variables.

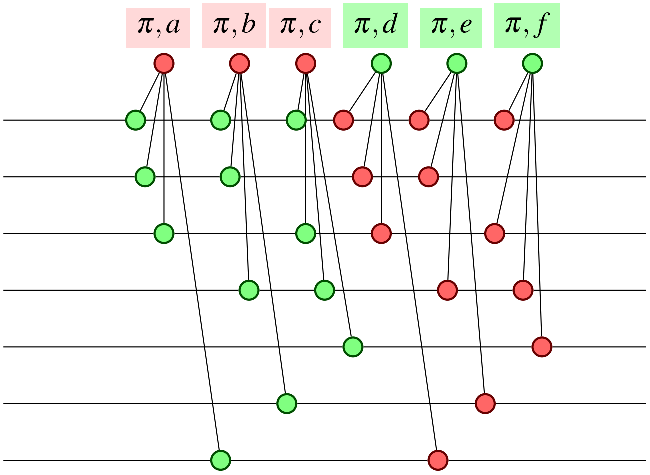
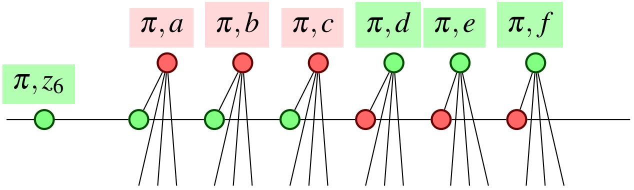


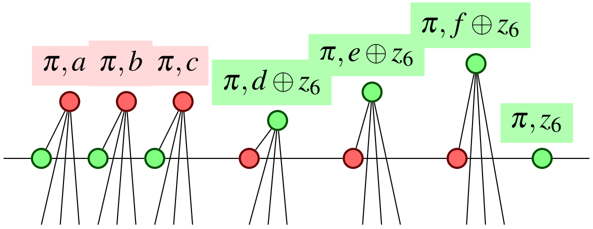
Figure 7: The error-detecting circuit

Note the colour-symmetry of the error-detection sub-circuit (Figure 7). From this we can immediately see that the first three measurements are resonsible for detecting X errors, while the remaing three are responsible for Z errors. The variables a, b, c then form the X-syndrome, and d, e, f the Z-syndrome. Viewing the each sub-syndrome as a bit-string, the eight possible values correspond to the 7 qubits of the codeword, and the possibility “no error”.

As a typical example, we derive the value of z_6, indicating a Z-error on qubit 6. Ignoring the other qubits of the codeword we have the following situation:



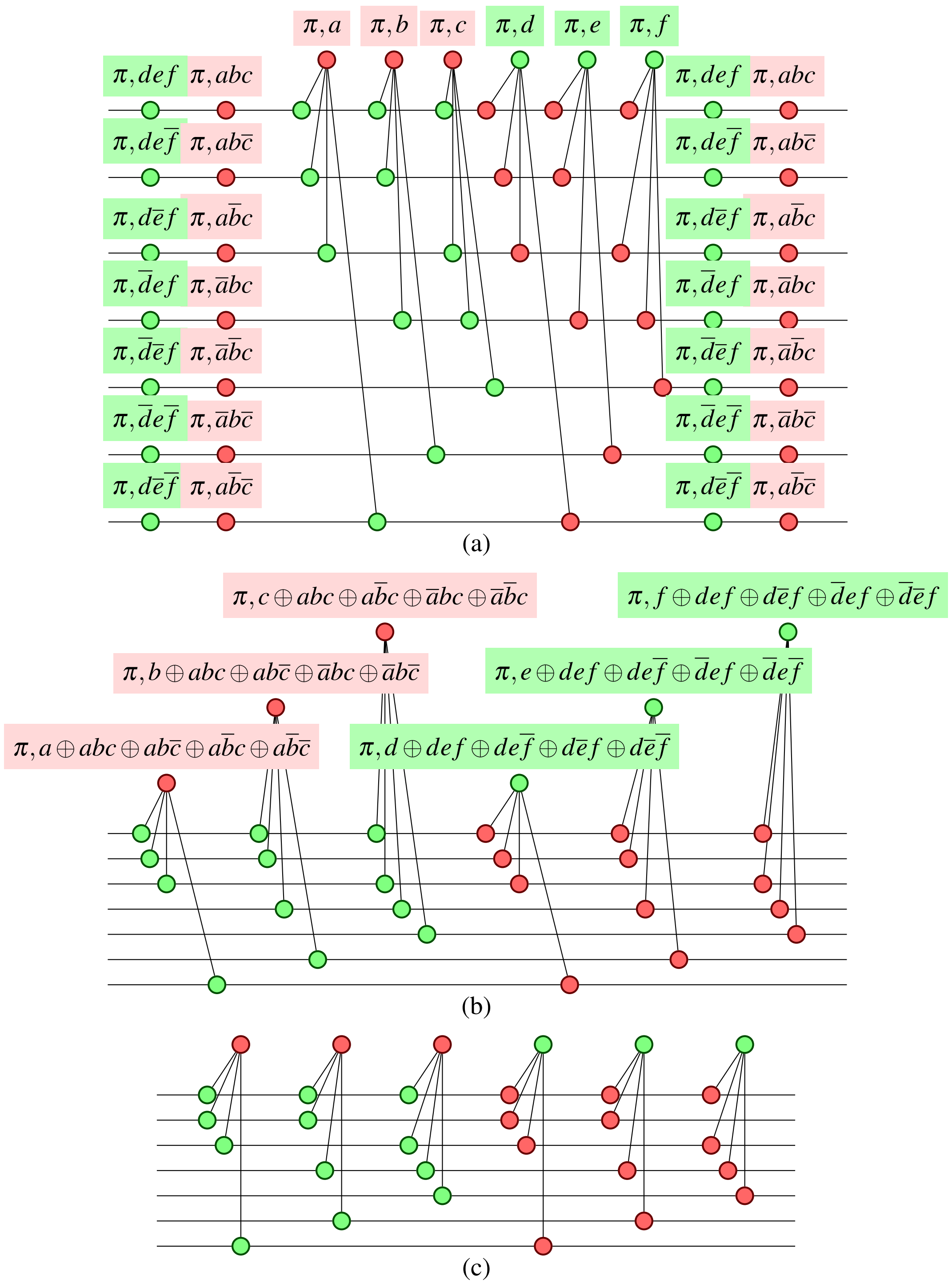
By repeatedly applying the α-commute, copying, and spider rules, this can be rewritten as follows:



Now it is easy to see that the diagram will only be deterministic (with respect to Z errors) on the condition that

$$(z_6 \leftrightarrow d) \wedge (z_6 \leftrightarrow e) \wedge (z_6 \leftrightarrow f)$$

from which we derive $z_6 = d \wedge e \wedge f$. Similar analysis can derive the values for the other variables, shown in the table below.


π, a
π, b
π, c
π, d
π, e
π, f
π, def
π, abc
$\pi, de\bar{f}$
$\pi, ab\bar{c}$
$\pi, d\bar{e}f$
$\pi, a\bar{b}c$
$\pi, \bar{d}ef$
$\pi, \bar{a}bc$
$\pi, \bar{d}\bar{e}f$
$\pi, \bar{a}\bar{b}c$
$\pi, \bar{d}e\bar{f}$
$\pi, \bar{a}b\bar{c}$
$\pi, d\bar{e}\bar{f}$
$\pi, a\bar{b}\bar{c}$
$\pi, c \oplus abc \oplus a\bar{b}c \oplus \bar{a}bc \oplus \bar{a}\bar{b}c$
$\pi, f \oplus def \oplus d\bar{e}f \oplus \bar{d}ef \oplus \bar{d}\bar{e}f$
$\pi, b \oplus abc \oplus ab\bar{c} \oplus \bar{a}bc \oplus \bar{a}b\bar{c}$
$\pi, e \oplus def \oplus de\bar{f} \oplus \bar{d}ef \oplus \bar{d}e\bar{f}$
$\pi, a \oplus abc \oplus ab\bar{c} \oplus a\bar{b}c \oplus a\bar{b}\bar{c}$
$\pi, d \oplus def \oplus de\bar{f} \oplus d\bar{e}f \oplus d\bar{e}\bar{f}$

(a)

(b)

(c)

Figure 8: (a) The error-correcting circuit with errors. (b) The reduced form (c) The unconditional form. **NB**: we write *ab* for $a \wedge b$ and $\overline{a}$ for $\neg a$.

Qubit i	x_i	z_i
6	$a \wedge b \wedge c$	$d \wedge e \wedge f$
5	$a \wedge b \wedge \neg c$	$d \wedge e \wedge \neg f$
4	$a \wedge \neg b \wedge c$	$d \wedge \neg e \wedge f$
3	$\neg a \wedge b \wedge c$	$\neg d \wedge e \wedge f$
2	$\neg a \wedge \neg b \wedge c$	$\neg d \wedge \neg e \wedge f$
1	$\neg a \wedge b \wedge \neg c$	$\neg d \wedge e \wedge \neg f$
0	$a \wedge \neg b \wedge \neg c$	$d \wedge \neg e \wedge \neg f$

Now we can write down the complete circuit, including the errors, as shown in Figure 8 (a). Notice that the choice of conditional formulae for the errors enforces the desired error model. By propagating the errors forward, as above, we arrive at the diagram of Figure 8 (b). It is easily verified that the boolean expressions in the remaining conditional vertices are all uniformly false, hence they can be dropped. Therefore we conclude that the correcting circuit does indeed correct any of the claimed errors.

The final thing we will check is that the composition of the encoding circuit, the correcting circuit, and the decoding circuit combine to yield the identity for a single qubit. The combined graph is rather large, so we will not show it here. Quantomatic was not able to carry out the proof completely automatically, and several lemmas had to be added as new rules. However, with some human intervention the circuit does indeed reduce to the desired form. The proof can be found in the appendix.

5 Discussion

We have shown that Steane's 7-qubit error-correcting code performs as advertised. This is perhaps not surprising. The main achievement here was to carry out the proof in the automated rewriting system Quantomatic. Since any practical quantum program will have to be implemented "underneath" an error-correcting code of some kind we view this as a first step towards mechanically verifying practical programs.

Since (to our knowledge) this is the largest test yet carried out using Quantomatic, a few comments are in order. There were two obstacles to this work. Firstly, the ZX-calculus with conditional vertices is not implemented in Quantomatic, so this part had to be done either by hand or via case analysis. In particular it was necessary to simplify the boolean expressions. It should be possible to add such features to the program. The more significant problem was the lack of good strategies for fully automatic work. Since many of the rewrite rules must be used as expansions rather than reductions, manual intervention is required. As glance at the appendix will tell the reader that we have already reached the stage where it is not easy for the human operator to see what he is doing.

The Steane code is not optimal, in the sense that it uses more qubits than the theoretical minimum of 5, which is attained by other codes. The reason we chose to study Steane code is that quantum logic gates can be implemented directly on the code-space, and these have a particularly simple form. The study of the encoded operations, and the verification of encoded circuits is a subject for future work.

References

[1] Bob Coecke & Ross Duncan (2011): *Interacting Quantum Observables: Categorical Algebra and Diagrammatics*. *New J. Phys* 13(043016), doi:10.1088/1367-2630/13/4/043016. Available at `http://iopscience.iop.org/1367-2630/13/4/043016/`.

[2] L. Dixon, R. Duncan, B. Frot, A. Kissinger, A. Merry & M. Soloviev: *Quantomatic*. Available at `https://sites.google.com/site/quantomatic/`.

[3] Lucas Dixon & Ross Duncan (2009): *Graphical Reasoning in Compact Closed Categories for Quantum Computation*. *Annals of Mathematics and Artificial Intelligence* 56(1), pp. 23–42, doi:10.1007/s10472-009-9141-x.

[4] R Duncan (2006): *Types for Quantum Computing*. Ph.D. thesis, Oxford University.

[5] R. Duncan & S. Perdrix (2010): *Rewriting measurement-based quantum computations with generalised flow*. In S. Abramsky, C. Gavoille, C Kirchner, F. Meyer auf der Heide & P. G. Spirakis, editors: *Automata, Languages and Programming, 37th International Colloquium, ICALP 2010, Proceedings Part II*, *Lecture Notes in Computer Science* 6199, Springer, pp. 285–296, doi:10.1007/978-3-642-14162-1_24.

[6] Ross Duncan (2013): *A graphical approach to measurement-based quantum computing*. In Chris Heunen, Mehrnoosh Sadrzadeh & Edward Grefenstette, editors: *Quantum Physics and Linguistics: A Compositional, Diagrammatic Discourse*, chapter 3, Oxford University Press, doi:10.1093/acprof:oso/9780199646296.003.0003.

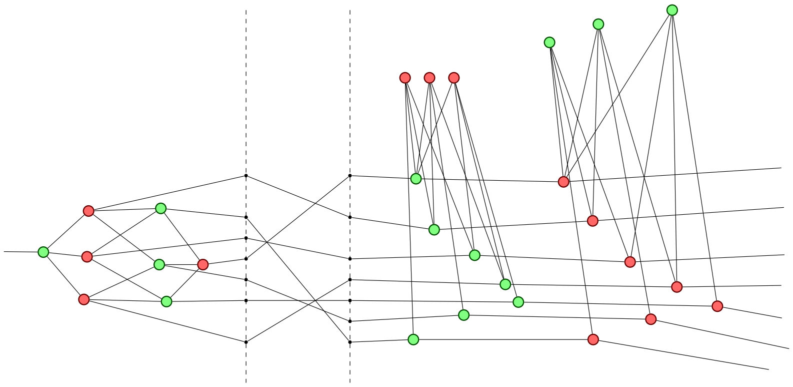
[7] N. David Mermin (2007): *Quantum Computer Science*. Cambridge University Press, doi:10.1017/CBO9780511813870.

[8] Andrew Steane (1996): *Multiple-Particle Interference and Quantum Error Correction*. *Proc. R. Soc. Lond. A* 8(1954), pp. 2551–2577, doi:10.1098/rspa.1996.0136.

A Putting it all together

In this appendix, we describe the mechanised proof that the complete Steane code is correct by composing the encoding circuit, the unconditional diagram for the error-correcting circuit (derived in Section 4), and the decoding circuit, and then showing that this combined diagram reduces to the identity. This calculation was done with Quantomatic.

Phase 1 We begin by composing the encoder to the error-corrector:

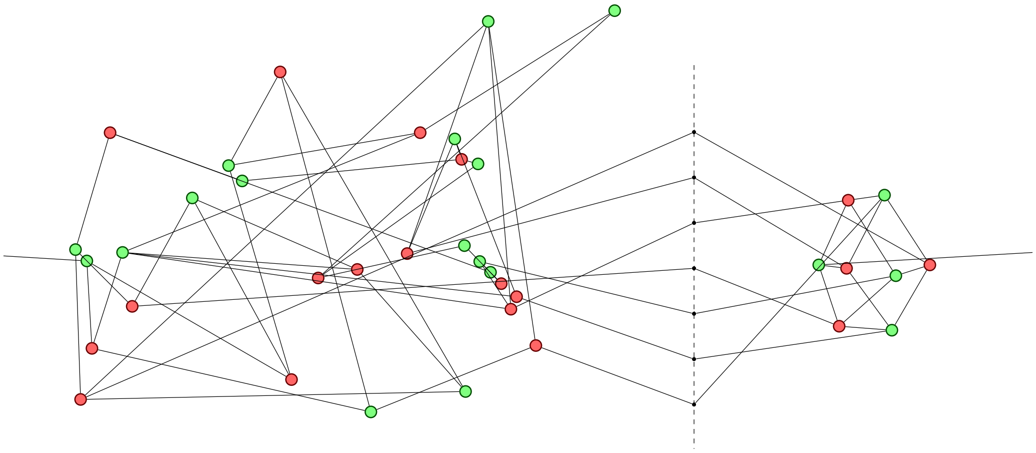


Note that the permutation between the two circuits is not strictly necessary. We rewrite this diagram as described in Table 1, and illustrated in Figure 9.

Graph	Rewrite sequence from previous
(1)	Initial graph
(2)	3 × spider, 3 × bialg, 4 × spider
(3)	bialg, 3 × spider, bialg, spider, 2 × bialg
(4)	altcycle4, bialg, 4 × spider, bialg, spider
(5)	bialg

Table 1: Descripition of rewrite sequence for Figure 9

Phase 2 Next, we adjoin the decoding circuit to diagram (5).



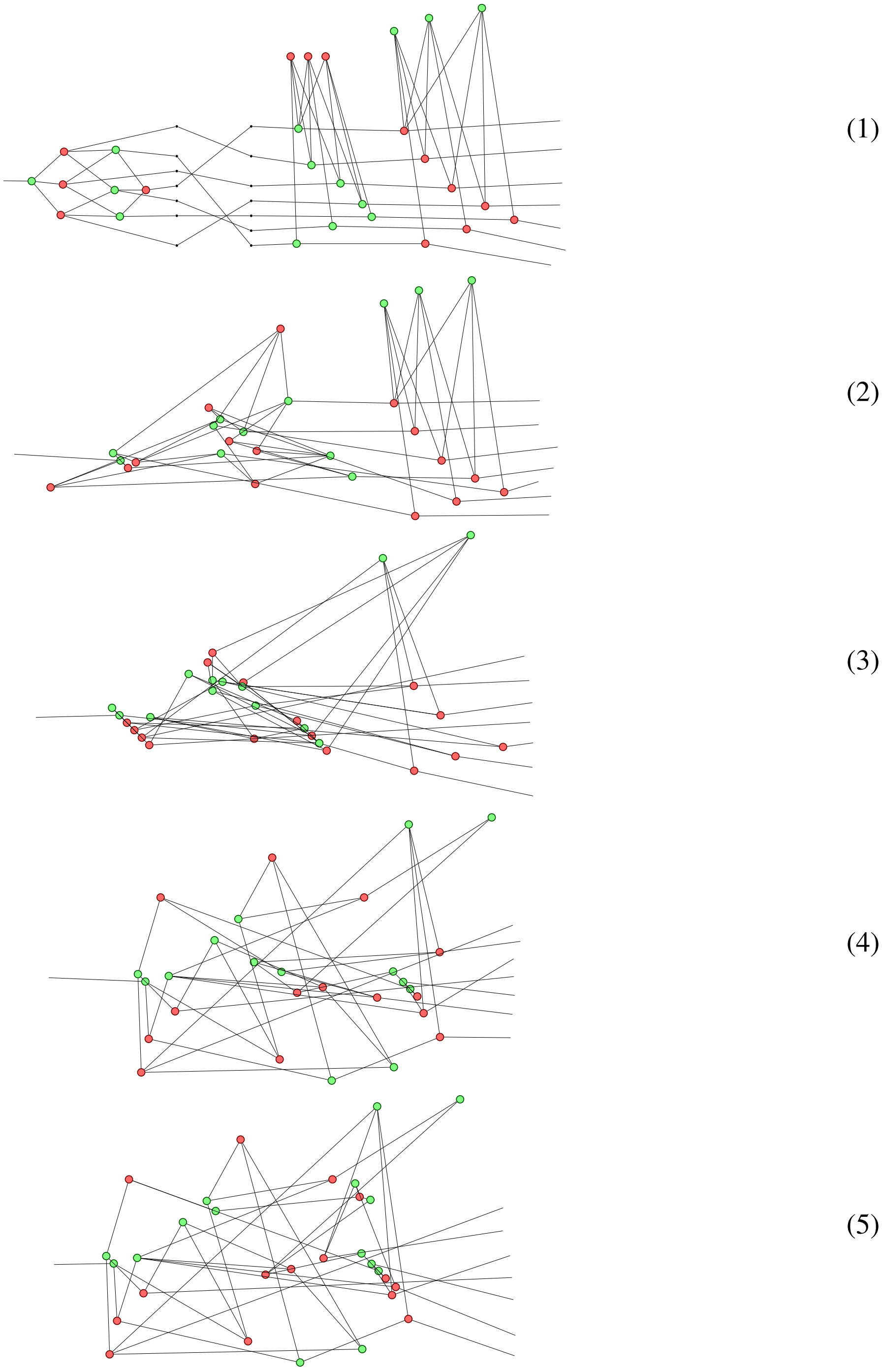
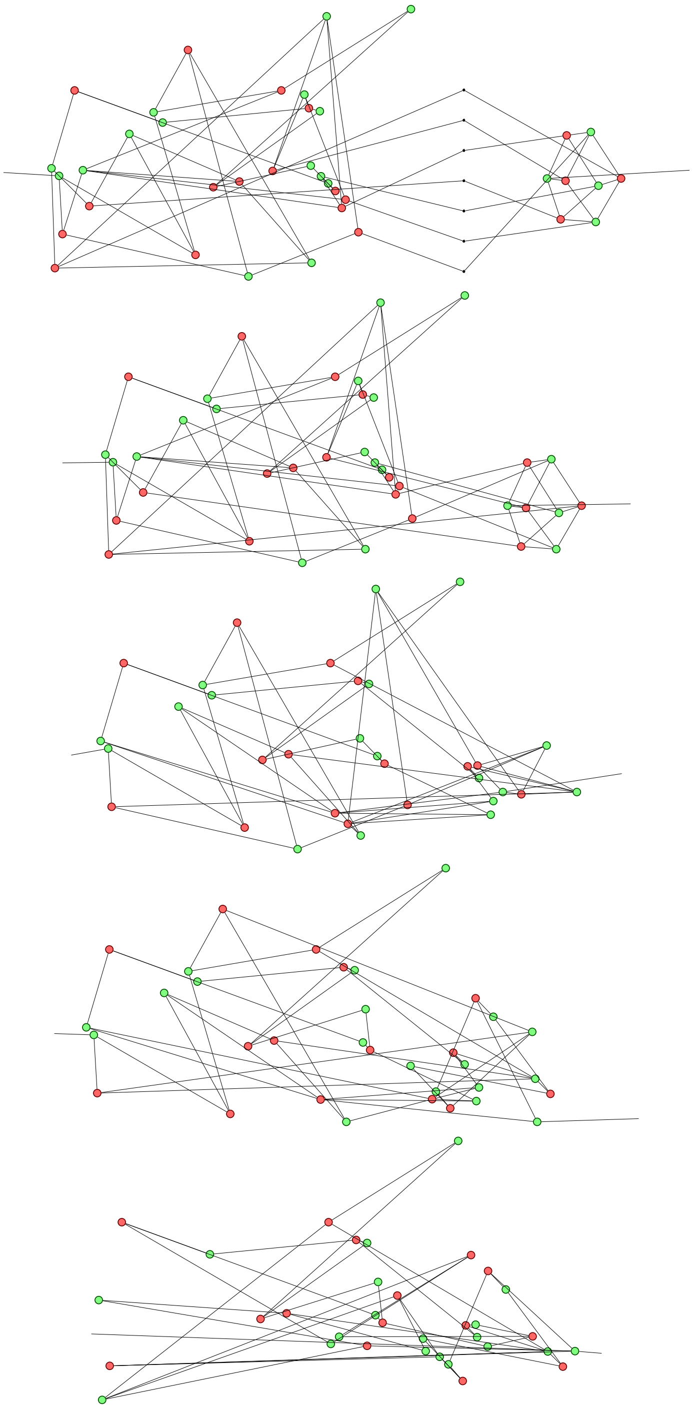


Figure 9: Rewrite sequence 1

The rewriting sequence is described in Table 2 and illustrated in Figures 10 and 11. The final diagram is a single wire, hence the cicruit rewrites to the identity, as required.

Graph	Rewrite sequence from previous
(6)	Initial graph
(7)	6 × spider, 1 × bialg, 2 × spider
(8)	2 × bialg, 4 × spider, 1 × bialg, 2 × spider
(9)	3 × bialg, 4 × spider, 1 × bialg, 4 × spider
(10)	15 × spider (only on green)
(11)	8 × spider (only on red)
(12)	1 × antiloop, 2 × bialg, 3 × x_abelian1
(13)	1 × hopf-banged, 1 × spider
(14)	6 × spider
(15)	4 × hopf-banged
(16)	1 × bialgebra1-rev, 4 × green spiders, 3 × red spiders
(17)	drop scalars (= 1)

Table 2: Description of rewrite sequence for Figures 10 and 11.



(6)

(7)

(8)

(9)

(10)

Figure 10: Rewrite sequence 2(a)

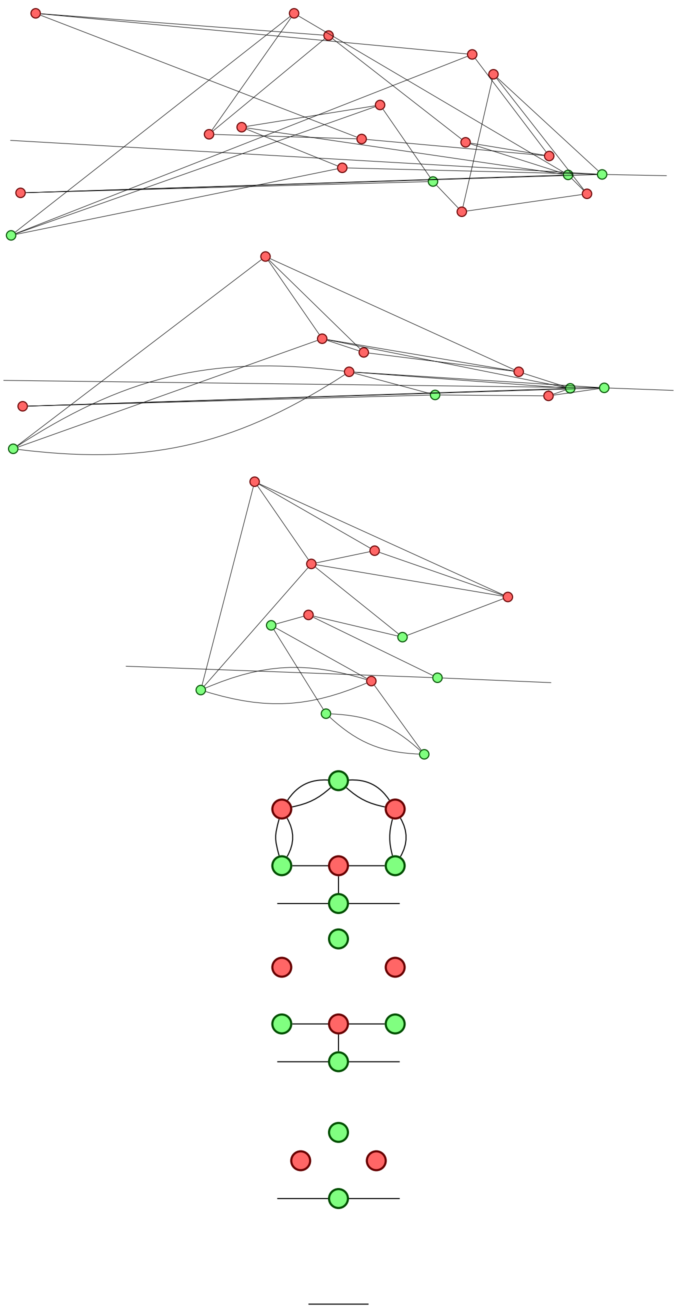


Figure 11: Rewrite sequence 2(b)