Fast and Effective Techniques for T-Count Reduction via Spider Nest Identities

Niel de Beaudrap
Department of Computer Science, University of Oxford, United Kingdom
niel.debeaudrap@cs.ox.ac.uk

Xiaoning Bian
Department of Mathematics & Statistics, Dalhousie University, Halifax, Canada
bian@dal.ca

Quanlong Wang
Department of Computer Science, University of Oxford, United Kingdom
Cambridge Quantum Computing Ltd., Cambridge, United Kingdom
quanlong.wang@cs.ox.ac.uk

Abstract

In fault-tolerant quantum computing systems, realising (approximately) universal quantum computation is usually described in terms of realising Clifford+T operations, which is to say a circuit of CNOT, Hadamard, and $\pi/2$-phase rotations, together with T operations ($\pi/4$-phase rotations). For many error correcting codes, fault-tolerant realisations of Clifford operations are significantly less resource-intensive than those of T gates, which motivates finding ways to realise the same transformation involving T-count (the number of T gates involved) which is as low as possible. Investigations into this problem [5, 21, 4, 3, 10, 6] has led to observations that this problem is closely related to NP-hard tensor decomposition problems [23] and is tantamount to the difficult problem of decoding exponentially long Reed-Muller codes [6]. This problem then presents itself as one for which must be content in practise with approximate optimisation, in which one develops an array of tactics to be deployed through some pragmatic strategy. In this vein, we describe techniques to reduce the T-count, based on the effective application of "spider nest identities": easily recognised products of parity-phase operations which are equivalent to the identity operation. We demonstrate the effectiveness of such techniques by obtaining improvements in the T-counts of a number of circuits, in run-times which are typically less than the time required to make a fresh cup of coffee.

2012 ACM Subject Classification Computer systems organization → Quantum computing

Keywords and phrases T-count, Parity-phase operations, Phase gadgets, Clifford hierarchy, ZX calculus

Digital Object Identifier 10.4230/LIPIcs.TQC.2020.11

Supplementary Material The software which produced our results may be found on GitHub, at `https://github.com/njross/optimizer`, commit 46b8ce0873ff09bf54cf704080b7daa252c48eba.

Funding N. de Beaudrap was supported in part by a Fellowship funded by a gift from Tencent Holdings (tencent.com), and by the EPSRC National Hub in Networked Quantum Information Technologies (NQIT.org). X. Bian is supported by NSERC and by AFOSR under Award No. FA9550-15-1-0331. Q. Wang is supported by Cambridge Quantum Computing Ltd. and by the AFOSR grant FA2386-18-1-4028. Our results were made possible in part by the use of the Dalhousie University Mathstat Cluster [11].

Acknowledgements We thank Earl Campbell, Luke Heyfron, Alexander Cowtan, Aleks Kissinger, and John van de Wetering for helpful discussions. We extend a very special thanks to Matthew Amy, who wrote a small extension of `feynver` [2] to allow verification of procedures which post-select the $|+\rangle$ state, for the express purpose of helping us to independently verify the correctness of reductions such as appear in this work and in Ref. [14]. X. Bian would like to thank his Ph.D. supervisor Peter Selinger for his support.

1 Introduction

To achieve practical scalable quantum computation, it is important to find effective (both useful and efficient) techniques to reduce the resources required to perform computations. Error correction, and in particular realising operations in a fault-tolerant way, is expected to be a particularly significant source of resource overheads. In most quantum error-correcting codes, Clifford group operations involve less overhead than non-Clifford gates, such as the T (or $\pi/4$ phase-rotation) gate. As the set of Clifford+T circuits is approximately universal for quantum computation [32], this motivates the *T-count* — or the number of T gates — as a quantity of interest in the resources required to realise a quantum computation.

On the other hand, in order to test the effectiveness of quantum technologies, it is helpful to be able to simulate the outcomes of quantum computations inasmuch as this is feasible. As circuits of Clifford operations can be efficiently simulated [22, 1], this motivates the approach of simulating quantum circuits by extending those efficient simulation techniques [9, 8], this again motivates the T-count as a measure of interest in the complexity of quantum circuits.

In this article, we consider the problem of reducing the T-count required to represent a unitary circuit provided as input. Following Heyfron and Campbell [23], we consider transformations of circuits which isolate a subcircuit of diagonal operations which is the only part of the algorithm with non-trivial T-count. The approach of Heyfron and Campbell [23] is to transform Clifford+T circuits, to circuits with the following structure:

1. An initial stage of CNOT gates; followed by
2. A stage of diagonal non-Clifford operations; followed by
3. A sequence of (possibly classically controlled) Clifford operations.

This allows Ref. [23] to reduce the problem of T-count reduction to an analysis of the diagonal non-Clifford portion of this circuit, in terms of *phase polynomials*. This builds on a sequence of results which revolve around such operations [5, 21, 4, 3, 10, 6] presented in various but similar ways, and in particular establishes a connection between T-count optimisation and difficult coding problems and tensor decomposition problems [6, 23]. Our approach is to elaborate on that of Campbell and Heyfron as follows:

- Reduce the complexity of the diagonal non-Clifford operation by more flexible (but essentially elementary) separation of the circuit into stages by allowing the first stage to contain arbitrary Clifford gates;
- Analyse the diagonal non-Clifford portion of the circuit directly in terms of "$\pi/4$-parity-phase operations" — essentially operators of the form $\exp(i\frac{\pi}{8}(Z \otimes \cdots \otimes Z))$ — rather than as phase polynomials, simplifying them through the efficient application of identities of such operations.

We call these "$\pi/4$-parity-phase operations" as they induce a $e^{i\pi/4}$ relative phase on standard basis states, depending on some parity computation $f(x) = x_{k_1} \oplus x_{k_2} \oplus \cdots \oplus x_{k_m}$. As each $\pi/4$-parity-phase gate can be realised in principle using a single T or $T^\dagger$ gate (and some CNOT gates), simplifying $\pi/4$-parity-phase circuits is directly productive to reducing T-count.

This line of investigation, first identified in the context of T-count by Amy, Maslov, and Mosca [4], was further developed upon by Gosset *et al.* [21], Amy and Mosca [6], Kissinger and van de Wetering [26], and Zhang and Chen [34]. In previous work [14], we described a family of identities of $\pi/4$-parity-phase operations — "spider nest identities"' — which, when used in combination with Heyfron and Campbell's "TODD" subroutine [23], led to new records in T-count for several benchmark circuits.

In this work, we report new techniques for T-count reduction through the use of spider nest identities, and compare their effectiveness (the reduced T count and run-times) against the best previous result found in the literature. While these techniques could easily be

combined with other high-performance reduction subroutines such as TODD, our results do not involve any other recently developed techniques beyond those of Ref. [14]. We obtain a number of new records for the T-count, obtained almost exclusively[1] in very practical run-times on a consumer-grade laptop. (For example, the second-largest circuit, on 768 qubits, was simplified in less than 3 minutes.) This opens the door to further improvements through the identification of further useful identities of $\pi/4$-parity-phase operations, and improved techniques for deploying these identities.

2 Preliminaries

We first set out some basic or existing results, using the following notation. Let $[n] := \{1, 2, \ldots, n\}$ and $\mathbb{1}$ be the 2×2 identity matrix. For sets $S, T \subseteq V$ we write $S \,\Delta\, T$ for the symmetric difference $(S \cup T) \setminus (S \cap T)$, and $\mathbf{x}^{(S)} \in \{0,1\}^V$ denote the incidence vector of S, where $x_j^{(S)} = 1$ if and only if $j \in S$. We let $\mathcal{P}^n := \{ i^k P_1 \otimes \cdots \otimes P_n \,|\, k \in \mathbb{Z} \;\&\; P_j \in \{\mathbb{1}, X, Y, Z\} \}$ denote the n-qubit Pauli group. We define the Clifford hierarchy (on n qubits) by defining $\mathcal{C}_1^n := \mathcal{P}_n$, and

$$\mathcal{C}_k^n = \{ U \in \mathrm{U}_n(\mathbb{C}) \,|\, \forall P \in \mathcal{P}^n .\; UPU^\dagger \in \mathcal{C}_{k-1}^n \} \tag{1}$$

for $k > 1$; we call $\mathcal{C}_k^n$ (for arbitrary n) the *k^{th} level of the Clifford hierarchy*. As an abuse of notation, we identify $\mathcal{C}_k^n$ as a subset of $\mathcal{C}_k^N$ for $n < N$; we may then write $S \in \mathcal{C}_2^n$ and $T \in \mathcal{C}_3^n$ for all $n \geqslant 1$.

Let $\mathcal{D}_k^n \subseteq \mathcal{C}_k^n$ be the subset of diagonal operations in the k^{th} level of the Clifford hierarchy. (We again identify $\mathcal{D}_k^n$ as a subset of $\mathcal{D}_k^N$ for $n < N$.) It is easy to show that $\mathcal{D}_k^n$ forms an abelian group. In particular: consider any diagonal operation as a product of operators $\exp\big(i\theta_x |x\rangle\langle x|\big)$ for various $x \in \{0,1\}^n$, and expand each $|x\rangle\langle x|$ as a linear combination of Pauli operators. Then one may show (see Ref. *e.g.* [14, Appendix A]) that $\mathcal{D}_k^n$ is generated by the operators $\omega \cdot \mathbb{1}^{\otimes n}$ for any global phase ω, together with all operations of the form $D_{S,k}$ for sets $S = \{s_1, \ldots, s_m\} \subseteq [n]$ for $m \geqslant 1$, defined by

$$D_{S,k} \;=\; \exp\Big(-\frac{i\pi}{2^k}\big(Z_{s_1} \otimes \cdots \otimes Z_{s_m}\big)\Big) \;=\; \exp\Big(-\frac{i\pi}{2^k} Z_S\Big) \;=\; \cos\big(\tfrac{\pi}{2^k}\big)\mathbb{1} - i\sin\big(\tfrac{\pi}{2^k}\big) Z_S \,, \tag{2}$$

where $Z_S = \bigotimes_{j\in S} Z_j$.[2] Note that $X_a Z_S X_a^\dagger = (-1)^{x_a^{(S)}} Z_S$, and that $\mathrm{CNOT}_{a,b}\, Z_S\, \mathrm{CNOT}_{a,b}^\dagger = Z_{S'}$, where here $S' = S \,\Delta\, \{a\}$ if $b \in S$ and $S' = S$ otherwise. From this it follows that

$$X_b\, D_{S,k}\, X_b^\dagger \;=\; D_{S,k}^{-1} \;\in\; \mathcal{D}_k^n \tag{3a}$$

if $b \in S$ (and $X_b\, D_{S,k}\, X_b^\dagger = D_{S,k}$ otherwise); and

$$\mathrm{CNOT}_{a,b}\, D_{S,k}\, \mathrm{CNOT}_{a,b}^\dagger \;=\; D_{S',k} \;\in\; \mathcal{D}_k^n \tag{3b}$$

so that $\mathcal{D}_k^n$ is preserved under conjugation by CNOT and X operations. Also note that $D_{S,k}^2 = D_{S,k-1}$, from which it follows that $\mathcal{D}_{k-1}^n \subseteq \mathcal{D}_k^n$.

---

[1] The one circuit which we did not simplify on a laptop was the largest benchmark circuit that we tested, acting on 1536 qubits and involving nearly two million T gates alone. This was instead simplified on Dalhousie University's Mathstat Cluster [11], which took less than 15 minutes to realise a 43% reduction in T-count.

[2] We define $D_{S,k}$ for all $k \in \mathbb{Z}$; however, as one may easily show $D_{S,0} = -\mathbb{1}^{\otimes n}$ and $D_{S,k} = \mathbb{1}^{\otimes n}$ for all $k < 0$ and $S \subseteq [n]$, these operations are of interest principally for $k > 0$.

We refer to the operators $D_{S,k+1}$, and their inverses, as "$\pi/2^k$-parity-phase" operations, as the action of $D_{S,k+1}$ on standard basis states is given by

$$D_{S,k+1}\,|z\rangle \;=\; \mathrm{e}^{i\pi/2^{k+1}}\exp\Big(i\,[\mathbf{x}^{(S)}\cdot z]\,\pi/2^k\Big)\,|z\rangle \tag{4}$$

inducing a relative phase of $\pi/2^k$ depending on the result of a parity computation $\mathbf{x}^{(S)}\cdot z = z_{s_1}\oplus z_{s_2}\oplus\cdots\oplus z_{s_m}$. More generally, we may refer to $\exp(\pm\frac{1}{2}i\theta Z_S)$ as a θ-parity-phase operation.

From Eqn. (3b), it follows that any operation $D_{S,k}$ can be reduced to an operation $D_{j,k}\propto\mathrm{diag}(1,\mathrm{e}^{2\pi i/2^k})$ acting on a single qubit j, by conjugation with an appropriate CNOT circuit. In particular, it follows that the operation $D_{S,3}$ can be easily realised with a T-count of 1. This allows us to approach the question of reducing T count by considering decompositions of unitaries involving few $\pi/4$-parity-phase operations, acting on many qubits. Amy and Mosca [6] noted the relevance of the operators $D_{S,k}$ in this context, and both Kissinger and van de Wetering [26] and Zhang and Chen [34] make direct use of them in their analysis of T count to achieve their results. (Litinski [27] similarly considers these operators in the context of compilation of quantum circuits to lattice surgery [24]).

An important role of $D_{S,3}$ gates for $S\subseteq[n]$ is their relationship to diagonal gates in $\mathcal{D}_3^n$ which are controlled-unitaries in a more straightforward sense, such as CS and CCZ:

$$\mathrm{C}S=\exp\Big(\frac{i\pi}{2}\,|11\rangle\langle 11|\Big),\qquad\qquad \mathrm{CC}Z=\exp\Big(i\pi\,|111\rangle\langle 111|\Big); \tag{5}$$

we may describe how to generate these from $D_{k,3}$ operations by decomposing the projectors $|11\rangle\langle 11|$ or $|111\rangle\langle 111|$ into tensor products of $|1\rangle\langle 1|=\frac{1}{2}\big(\mathbb{1}-Z\big)$, and expanding to obtain a product of $D_{S,3}$ gates. Disregarding any $D_{\emptyset,3}$ factors, which realise global phases, we obtain

$$\begin{aligned}\mathrm{C}S_{h,j}&\propto D_{\{h\},3}D_{\{j\},3}D^{-1}_{\{h,j\},3}\,;\\ \mathrm{CC}Z_{g,h,j}&\propto D_{\{g\},3}D_{\{h\},3}D_{\{j\},3}D^{-1}_{\{g,h\},3}D^{-1}_{\{g,j\},3}D^{-1}_{\{h,j\},3}D_{\{g,h,j\},3}\,.\end{aligned} \tag{6}$$

More generally, we may relate $(t-1)$-controlled $\pi/2^k$-phase gates to $\pi/2^{k-t+1}$-phase parity gates:

$$\prod_{\substack{S\in\wp(V)\\ S\neq\emptyset}} D_{S,k}^{(-1)^{|S|}}\;\propto\;\exp\Big(\frac{i\pi}{2^{k-|V|+1}}\,|1\rangle\langle 1|^{\otimes V}\Big), \tag{7}$$

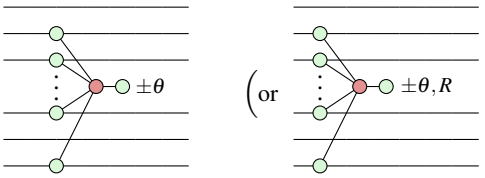
where the right-hand operator applies a phase of $\pi/2^{k-|T|-1}$ to those components of a state in which all of the qubits in T are in the state $|1\rangle$.

Circuits of parity-phase operations on n qubits which realise the identity, correspond in the notation of Amy and Mosca [6] to operators $U_{P_\mathbf{a}}$ for $\mathbf{a}\in\mathcal{C}_n\subseteq\mathbb{Z}_8^{2^n-1}$, where

$$P_\mathbf{a}(z)\;=\sum_{\substack{\mathbf{x}\in\{0,1\}^n\\ \mathbf{x}\neq\mathbf{0}}} a_x\big(x_1z_1\oplus x_2z_2\oplus\cdots\oplus x_nz_n\big) \tag{8}$$

and where $U_{P_\mathbf{a}}\,|\mathbf{z}\rangle=\exp\big(\frac{i\pi}{4}P_\mathbf{a}(\mathbf{z})\big)\,|\mathbf{z}\rangle$, which is identically $|\mathbf{z}\rangle$ for all $\mathbf{z}\in\{0,1\}^n$ when $\mathbf{a}\in\mathcal{C}_n$. Let $\mathrm{supp}(\mathbf{a})=\big\{\mathbf{x}\in\{0,1\}^n : a_\mathbf{x}\neq 0\big\}$. In this notation, each element $\mathbf{y}\in\mathrm{supp}(\mathbf{a})$ corresponds to a single phase-parity operator acting on the qubits j for which $y_j=1$; the relative phase induced by this operator is $a_\mathbf{y}\pi/4$; and the polynomial $P_\mathbf{a}$ describes a commuting product of such operations, for which $P_\mathbf{a}:\{0,1\}^n\to\mathbb{Z}_8$ is the all-zero function when $\mathbf{a}\in\mathcal{C}_n$.

We remark that a θ-phase parity operation U (such as an operator $D_{S,k}$) can be easily represented as tensor networks, using ZX diagrams (see Appendix A for an introduction to this notation),[3] with structure such as the following:



if classically conditioned on $\sum R \equiv 1 \pmod 2$ (9)

where horizontal wires represent qubits which are acted on by U, and $S \subseteq [n]$ is the subset of those qubits which have (light, green) degree-3 nodes on them. These are "phase gadgets", using the terminology of Kissinger and van de Wetering [26]. When the number of qubits acted on is m, we may refer to it as an "m-gadget". (If θ is an odd multiple of $\pi/4$, we may refer to it as a "T-phase m-gadget"; for θ an integer multiple of $\pi/2$, we refer to it as a "Clifford-phase m-gadget". If $m = 1$, we may also mildly abuse this terminology to refer to a simple green phase node as a "1-gadget".)

Remark.

The role played by the ZX calculus in our work is not an essential one, nor is expertise in the ZX calculus required to understand our results. However, in practice it did inform our line of investigation, by allowing us to obtain our results more quickly by identifying the objects of interest, and by making it easy to reason directly about the operators $D_{S,k}$. As the ZX calculus also provides a useful notation for visually representing the (non-local) unitary gates $D_{S,k}$ in a readable way, as in Eqn. (9), we use this notation in the article below. Readers should be able to understand our results by reading ZX diagrams simply as a straightforward alternative notation for quantum circuits (see Appendix A), the transformations of which are the subject of our work.

3 Phase gadget elimination tactics & spider nest identities

Reducing the T-count while preserving the meaning of a circuit, implicitly involves applying a mathematical identity. These are often identities of diagonal unitary circuits [4, 6, 34], though not always [21, 26].) In the special case of unitary circuits consisting solely of $\pi/4$-parity-phase operations, such a mathematical identity may be described in terms of a commuting product of operations which are proportional to the identity operator; and for any such identity, there is the question of how to effectively apply it to realise a significant reduction of T-count, as efficiently as possible.

In this section, we describe a broad framework for the reduction of T-count by means of the application of mathematical identities of commuting $\mathcal{D}_3^n$ operations. We also present some mathematical identities of this form — called "spider nest identities" — first presented in Ref. [14], and describe new techniques to use these identities to reduce T-count.

In the following, we use the terms "identity of $\pi/4$-parity-phase operations" or "identity of phase gadgets" (or simply "an identity") to refer to a circuit $\mathcal{J}$, whose T-count is at least 1 but which nevertheless realises the identity operation.

[3] In this article, where they occur, ZX diagrams may be read essentially as circuit diagrams, and in particular are read from left to right as with other circuit diagrams.

3.1 PHAGE tactics

We consider a particular approach to the reduction of $\mathcal{D}_3^n$ circuits by an analysis of families of non-trivial circuits which realise the identity transformation, which may be applied more broadly than we do here (and which in principle can be used to describe some existing techniques [6, 23]). For any family $\mathcal{F}$ of identities of $\pi/4$-parity-phase operations, there is an associated "phase gadget elimination tactic" (or PHAGE tactic) to reduce the T-count in a circuit $\mathbf{C}$ of such phase gadgets:

▶ **Phage Tactic** ($\mathcal{F}$).

1. *Determine whether there is an identity $\mathcal{J} \in \mathcal{F}$, such that $\mathbf{C}$ contains at least half of the T-gadgets which occur in $\mathcal{J}$ (or their inverses).*
2. *For any such identity $\mathcal{J}$, compute a circuit $\mathbf{C}_\mathcal{J}$ as the product of $\mathbf{C}$ and $\mathcal{J}^{-1}$. This may allow for simplifications (using the fact noted in Section 2 that $D_{S,k}^2 = D_{S,k-1}$), where by T-gadgets accumulate to form Clifford gadgets or to cancel altogether. Determine the resulting T-count.*
3. *Replace $\mathbf{C}$ with the circuit $\mathbf{C}_\mathcal{J}$ with the smallest T-count, if this is less than the T-count of $\mathbf{C}$ itself.*

The behaviour of a PHAGE tactic is in a sense "greedy", in that it selects some circuit $\mathbf{C}_\mathcal{J}$ which minimises the T count after a single application, ignoring the possibility of a more complicated sequence of reductions. The main principle of a PHAGE tactic is in that it selects a way to reduce the T-count, based on the comparison of a few different applicable identities of phase gadgets from a specific family $\mathcal{F}$. Such a tactic can then be applied again, or followed by other such "tactics".

In principle, the Tpar subroutine of Ref. [6], the TOOL and TODD subroutines of Ref. [23], and the results of Zhang and Chen [34] may be interpreted as algorithms to deploy PHAGE tactics, possibly more than once in sequence, and possibly with a random choice of family $\mathcal{F}$ (and where $\mathcal{F}$ itself may on occasion be a singleton set). This approach to T-count reduction can be distinguished from that of Kissinger and van de Wetering [26], in which phases may be reduced in unitary circuits (or more general tensor networks) which are not diagonal.

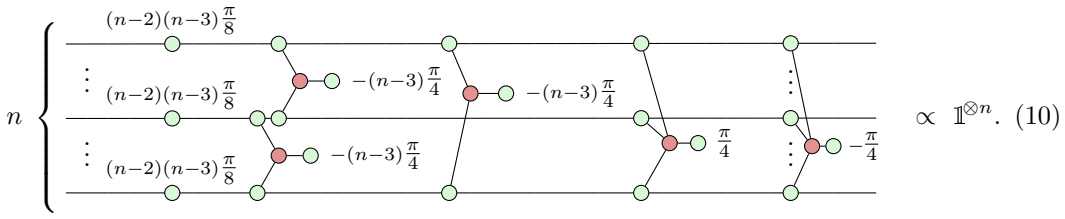
The difficulty in reducing the T-count arises from the fact that there are a very large number of identities of $\pi/4$-parity-phase operations, and a large number of subsets $S \subseteq [n]$ which one may consider. As Amy and Mosca observe [6], reducing the T-count is formally equivalent to decoding a length $2^n - 1$ punctured Reed-Muller code, in that the smallest T-count of a circuit amounts to the distance of a ciphertext to a valid codeword of such a code. However, no polynomial-time algorithms are known for the decoding problem on such codes. The difficulty is in formulating a successful *strategy* — a means of selecting an appropriately-sized family $\mathcal{F}$ of identities to try on a particular circuit. The question is then one of having a variety of tactics which one may efficiently explore and deploy to reduce the T-count.

3.2 Spider nest identities

We consider PHAGE tactics arising from identities of $\pi/4$-parity-phase operations (*i.e.*, of T-phase gadgets) which can be composed from some specific circuits — introduced in Ref. [14], and which we call "spider nest identities" — which realise the identity operator.

In qualitative terms, a "spider nest identity" consists of any circuit of phase-parity operations which realises an operation on n qubits which is proportional to the identity, in which only "very few" operations act on "many" qubits, and the vast majority act on "very

few" qubits. (In terms of the notation of Amy and Mosca [6], they would correspond to $\mathbf{a} \in \mathbb{Z}_8^{2^n-1}$ for which only very few $\mathbf{y} \in \mathrm{supp}(\mathbf{a})$ have Hamming weight larger than some low threshold $w > 0$; in the case of $\mathcal{D}_3^n$ operations, we set $w = 3$.) We generate these circuits from a minimal family of such circuits for $n \geqslant 4$, involving a single phase 4-gadget and various phase k-gadgets with $k \leqslant 3$:



$$\propto \mathbb{1}^{\otimes n}. \quad (10)$$

Here, the n-qubit circuit on the left-hand side of Eqn. (10) consists of a 1-gadget with phase $(n-2)(n-3)\frac{\pi}{8}$ on each line, a 2-gadget on each pair of lines with phase $-(n-3)\frac{\pi}{4}$, and a 3-gadget with phase $\frac{\pi}{4}$ on each set of three lines, and finally an n-gadget with phase angle $-\frac{\pi}{4}$. (For a proof of this identity, see Appendix B of Ref. [14]; in the case $n = 4$ this corresponds to R_{13} of Ref. [3].) The name "spider nest" here refers to the qualitative feature that it involves a few "large spiders", together with a large number of "small spiders".

Let $\mathcal{N}_S$ represent the circuit of phase gadgets on the left-hand side of Eqn. (10), acting on a set $S = \{1, 2, \ldots, n\}$ of cardinality n. How easily one may use this identity as part of a PHAGE tactic, to reduce T-count, is affected by the T-count of the circuit $\mathcal{N}_S$ itself. For a fixed value of n, and a T-phase gadget on 1 to 3 qubits, there is a question of whether or not such a gadget is involved in $\mathcal{N}_S$, as a number of the phase gadgets involved are Clifford-phase gadgets instead. In particular:

- If $n \equiv 1 \,(\mathrm{mod}\ 4)$ or $n \equiv 3 \,(\mathrm{mod}\ 4)$, all of the 2-gadgets in Eqn. (10) are Clifford-phase gadgets, which do not contribute to the T-count.
- If $n \equiv 2 \,(\mathrm{mod}\ 4)$ or $n \equiv 3 \,(\mathrm{mod}\ 4)$, all of the 1-gadgets in Eqn. (10) are Clifford-phase gadgets, which again do not contribute to the T-count.

Let $\mathbf{T}_n$ denote the T-count of $\mathcal{N}_S$: then

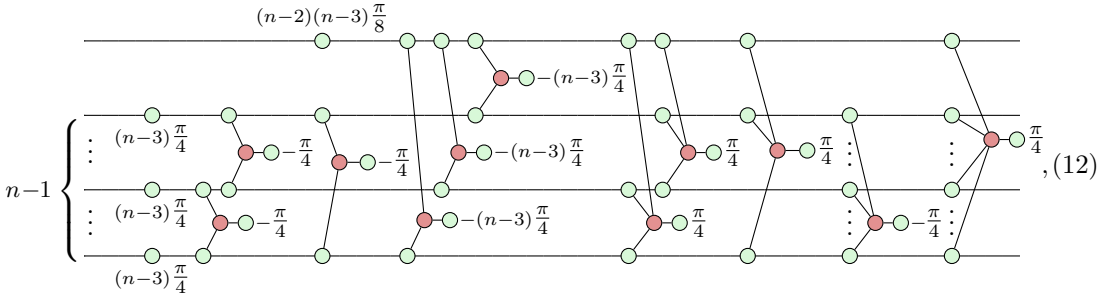
$$\mathbf{T}_n = \begin{cases} \frac{1}{6}n(n^2 + 6\delta_n - 1), & \text{for } n \text{ even;} \\ \frac{1}{6}n(n^2 - 3n + 6\delta_n + 2), & \text{for } n \text{ odd,} \end{cases} \quad (11)$$

where $\delta_n = 1$ if $n \equiv 0$ or $n \equiv 1$ modulo 4, and $\delta_n = 0$ if $n \equiv 2$ or $n \equiv 3$ modulo 4 (determining whether the 1-gadgets on each wire have T-count one or zero). In general, we have $\mathbf{T}_n = \frac{1}{6}n^3 - O(n^2) \pm O(n)$.

The scaling of $\mathbf{T}_n$ above might suggest that these circuits have at best a limited role to play in T-count reduction: for increasing sizes of wire-sets S, a somewhat large number of operations on a given subset S of wires must be present for substitution of $\mathcal{N}_S$ to yield a reduction in T-count. However, by composing multiple such circuits $\mathcal{N}_S$ for different subsets S, we may obtain a "composite" spider nest identity which has a smaller T-count, and which is thus more likely to be usable in practise for T-count reduction.

For instance, consider the specific circuit $\mathcal{N}_S \mathcal{N}_{S'}^{-1}$ where $|S| \geqslant 5$ and $S' = S \setminus \{r\}$ for some $r \in S$. As all of the operations in these circuits commute, it is possible to see that most of the phase 3-gadgets of $\mathcal{N}_S$ — the dominant contribution to $\mathbf{T}_n$ above — are cancelled by corresponding phase 3-gadgets of $\mathcal{N}_{S'}^{-1}$. (In many cases, most of the phase 1-gadgets of

$\mathcal{N}_S$ are similarly cancelled.) By collecting together the actions of the phase gadgets on each subset, we may show that $\mathcal{N}_S \mathcal{N}_{S'}^{-1}$ simplifies to a circuit of the following form:



, (12)

If $r = S \setminus S'$ represents the top qubit in the circuit above, note in particular that the dominant contributions to the size of the circuit are the phase 2-gadgets on all size-2 subsets of S', and the phase 3-gadgets which involve r and some size-2 subset of S'. If $\tilde{\mathbf{T}}_n$ denotes the T-count of the circuit above, we then have

$$\tilde{\mathbf{T}}_n = \begin{cases} n^2 - n + 2 + \delta_n & \text{for } n \text{ even;} \\ n^2 - 3n + 4 + \delta_n & \text{for } n \text{ odd,} \end{cases} \tag{13}$$

where again $\delta_n = 1$ if $n \equiv 0$ or $n \equiv 1$ modulo 4, and $\delta_n = 0$ if $n \equiv 2$ or $n \equiv 3$ modulo 4. In any case, we have $\tilde{\mathbf{T}}_n = n^2 - O(n)$.

3.3 Simple PHAGE tactics based on spider nest identities

Combining the two ideas above, we describe the PHAGE tactics which are used to achieve the T-count reductions seen in our results.

The first tactic is the reduction of phase-parity circuits by merging together $\pi/4$-parity-phase operations which act on sets of qubits in common, which may be described as the PHAGE tactic associated to the circuits consisting of mutually inverse pairs of T-phase gadgets on all possible sets of qubits. To do this to greatest effect (and also as simply as possible), we first use a circuit transformation procedure along the lines of Heyfron and Campbell [23], with modifications to improve performance. (In the context of reasoning about T count in terms of $\pi/4$-parity-phase operations, this technique was introduced in Ref. [14].) We describe this in more detail in the following Section, which describes our T-count reduction procedure.

Our other PHAGE tactic (or tactics, as they are similar but technically numerous) are novel, and are best described in terms of the following two sets of spider-nest identities on N qubit circuits:

- The family $\mathcal{F}_N^{(4)} = \{\mathcal{N}_S \mid S \subseteq [N] \text{ and } |S| = 4\}$, consisting of versions of the identity of Eqn. (10) applied to all subsets of $[N]$ of size 4
- The family

$$\mathcal{F}_N^{(5)} = \left\{ \mathcal{N}_S^{p_0} \mathcal{N}_{S_1}^{p_1} \mathcal{N}_{S_2}^{p_2} \mathcal{N}_{S_3}^{p_3} \mathcal{N}_{S_4}^{p_4} \mathcal{N}_{S_5}^{p_5} \;\middle|\; \begin{array}{l} S = \{q_1, q_2, q_3, q_4, q_5\} \text{ for distinct } q_j \in [N], \\ S_j = S \setminus \{q_j\} \text{ for } 1 \leqslant j \leqslant 5, \text{ and} \\ p_0 p_1 p_2 p_3 p_4 p_5 \in \{0,1\}^6 \setminus \{000000\} \end{array} \right\}, \tag{14}$$

  consisting of the 63 distinct identities for each set $S \subseteq [N]$ with $|S| = 5$, consisting of $\mathcal{N}_{S_j}$ applied to some or all subsets $S_j \subseteq S$ of size 4, and possibly also a copy of $\mathcal{N}_S$ on all the qubits of S, fusing together those phase-parity operations which act on common subsets $S' \subseteq S$.

These are the sets of all possible spider-nest identities on 4 or 5 qubits.[4]

For increasing values of N, the cardinalities of these families grow as $\frac{1}{24}n^4 + O(n^3)$ and $\frac{1}{120}n^5 + O(n^4)$ respectively — polynomial in size, but impractical to exhaustively iterate through for values of N which occur in common benchmark tests. This raises the question of how best to use them to realise T-count reductions. Our approach is to construct a list of 64 identities on four or five qubits, consisting of the elements of the sets $\mathcal{F}_4^{(4)} \cup \mathcal{F}_5^{(5)}$, and performing the following for each element $\mathcal{J}$ of this list:

1. Let s be the number of qubits on which $\mathcal{J}$ acts.
2. Repeat the following R times, for some fixed $R > 0$:
   a. Select a subset $S \subseteq [N]$ of size s uniformly at random.
   b. Select (from $\mathcal{F}_N^{(4)}$ if $s = 4$, or $\mathcal{F}_N^{(5)}$ if $s = 5$) the identity $\mathcal{K}$ acting on S, which is equivalent to $\mathcal{J}$ up to relabelling of the qubits.
   c. Apply the tactic **PHAGE**($\{\mathcal{K}\}$) associated with the singleton set $\{\mathcal{K}\}$.

This technique implicitly provides opportunities for identities to be applied in proportion to the number of isomorphic images of it exist in $\mathcal{F}_4^{(4)} \cup \mathcal{F}_5^{(5)}$. (For instance, isomorphic copies of the simplest identity $\mathcal{N}_{[4]}$ occurs six times in this set, and the identity of Eqn. (12) occurs five times.) As the probability that any one such identity will be useful when applied to a particular set $S \subseteq [N]$ of size 4 or 5 is small, it is important to choose a significantly large value of R: for our results, we took $R = 20\,000$.

We note that this particular strategy for T-count reduction is not particularly strongly suggested by the framework of PHAGE tactics induced by spider nest identities. Both the concept of a PHAGE tactic, and the range of possibilities for assembling spider nest identities, are broad enough that there is potential for much more sophisticated strategies to deploy them. Despite this, as we show in Section 5, in many cases we obtain the best known T-count for a number of circuits. Our result may therefore be considered a further proof of principle of the usefulness of spider nest identities, beyond the results of Ref. [14].

4 Reduction of T-count through simplification of parity-phase circuits

In this section, we describe how we applied the concept of T-count reduction via PHAGE tactics as part of a complete procedure to transform unitary circuits provided as input.

Remark.

Our results do not make heavy (explicit) use of the re-write rules of the ZX calculus: a reader who is content with circuits which involve intermediate measurements, and who is comfortable with reading a parity-phase gadget such as that of Eqn. (9) as a unitary operator, may interpret every diagram below as a circuit diagram. (See Appendix A for a guide to reading ZX diagrams.)

We take unitary circuits with gate-set $\{X, \mathrm{CNOT}, \mathrm{CCNOT}, Z, CZ, CCZ, H, S, T, \mathrm{SWAP}\}$ as input. For the sake of simplicity, we suppose that any multiply-controlled NOT gates with more than two controls are decomposed into CCNOT gates, for instance by computation and uncomputation on auxiliary qubits initialised to $|0\rangle$, or some more advanced technique.[5]

[4] The set $\mathcal{F}^{(5)}$ in particular is motivated by the reduction in T-count of the spider-nest identity shown in Eqn. (12), which is represented in five different ways in $\mathcal{F}^{(5)}$: once for each subset S_j of size 4.

[5] In our benchmarks, we consider the simple computation-uncomputation approach; other techniques (see *e.g.* Refs. [25, 20, 29]) are advisable in serious production work for optimising T-count.

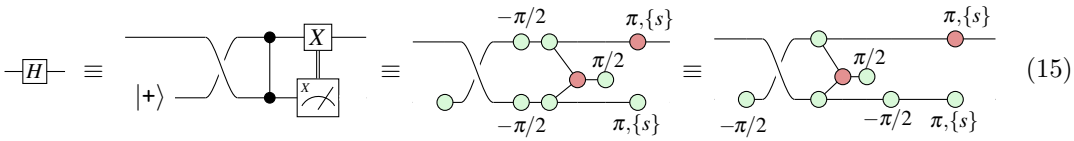
Our procedure follows and extends the approach of Heyfron and Campbell [23], of performing a transformation on circuits $\mathbf{C} \to \mathbf{C}_F \circ \mathbf{C}_\phi \circ \mathbf{C}_I$, where $\mathbf{C}_F$ and $\mathbf{C}_I$ consist entirely of Clifford gates, stabiliser state preparations, and stabiliser state measurements, and where $\mathbf{C}_\phi$ can be realised using only CNOT and T gates. We express the circuit $\mathbf{C}_\phi$ entirely in terms of phase gadgets, and so we describe as a "homogeneous" circuit. The objective of isolating such a circuit is that it provides us with the best opportunities to apply PHAGE tactics to reduce the T-count.

4.1 Circuit translation techniques

Our procedure, which we describe more explicitly in the next section, makes use of the following techniques.

H gate gadgetisation.

One of the techniques involved in isolating a $\mathcal{D}_3^N$ circuit is to replace Hadamard gates with a measurement-based gadget:



(15)

In the circuit second from the left, the two qubits are subject to a SWAP operation, followed by a $\mathrm{C}Z = \exp\big(i\pi\,|11\rangle\langle 11|\big)$ operation. The bottom qubit is measured finally with an X observable measurement (*i.e.*, in the $|\pm\rangle$ basis), and the top operation is acted on finally by an X operation only if the outcome is $|-\rangle$. The two diagrams on the right are ZX diagrams with additional annotations in the style of Ref. [18] (see also Appendix A). In particular, measurement is represented as a projection with a random outcome s which is heralded and may be used to control phase operations elsewhere. The leftmost ZX diagram describes the decomposition of the controlled-Z operation, using $\mathrm{C}Z_{h,j} \propto D_{\{h,j\},2}\, D_{\{h\},2}^{-1}\, D_{\{j\},2}^{-1}$. The final ZX diagram propogates the single-qubit $D_{\{*\},2}^{-1}$ operations towards the preparation and measurement of the second qubit, so that the second qubit is prepared in the $|\mathtt{-y}\rangle \propto |0\rangle - i\,|1\rangle$ state.

Extracting *H* gates from the circuit.

An obvious drawback of gadgetising H gates in this way is that it requires the use of auxiliary qubits. More directly important to our results is that, as the number of wires in a circuit increases, the more difficult it may be to successfully find opportunities to reduce the T count. Therefore, we attempt to transform the circuit in such a way that reduces the number of H gates from the part of the circuit with non-trivial T-count. This motivates us to define a subroutine `moveH` (which we describe at a high level in Appendix B), which transforms a circuit $\mathbf{C}$ over our gate-set, into a pair of circuits $(\mathbf{C}_F, \mathbf{C}')$, obtained by attempting to commute as many Hadamard gates of $\mathbf{C}$ to the end of the circuit as possible.

- We define $(\mathbf{C}_F, \mathbf{C}') = \mathtt{moveH}(\mathbf{C})$ in such a way that $\mathbf{C}_F \circ \mathbf{C}' \cong \mathbf{C}$ realises the same unitary, $\mathbf{C}_F$ contains only Clifford gates, $\mathbf{C}'$ contains no CCNOT gates, and where the total number of Hadamard gates in $(\mathbf{C}_F \circ \mathbf{C}')$ is at most the number of Hadamard gates in $\mathbf{C}$.

- We may use `moveH` twice, to attempt to extract Hadamard gates either from the end of the circuit $\mathbf{C}$, and also the beginning of the circuit $\mathbf{C}$. If we compute

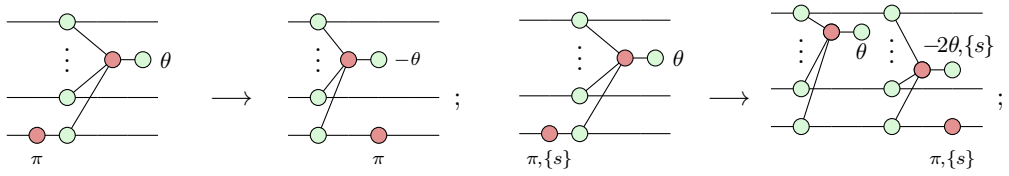
$$(\mathbf{C}_F, \mathbf{C}') = \texttt{moveH}(\mathbf{C}); \quad (\tilde{\mathbf{C}}_I, \tilde{\mathbf{C}}_M) = \texttt{moveH}\big((\mathbf{C}')^{-1}\big); \quad (\mathbf{C}_I, \mathbf{C}_M) = \big(\tilde{\mathbf{C}}_I^{-1}, \tilde{\mathbf{C}}_M^{-1}\big), \tag{16}$$

  then $(\mathbf{C}_F \circ \mathbf{C}_M \circ \mathbf{C}_I) \cong \mathbf{C}$, the number of Hadamard gates in $(\mathbf{C}_F \circ \mathbf{C}_M \circ \mathbf{C}_I)$ is at most the number of Hadamard gates in $\mathbf{C}$, and $\mathbf{C}_I$ and $\mathbf{C}_F$ only contain Clifford gates.

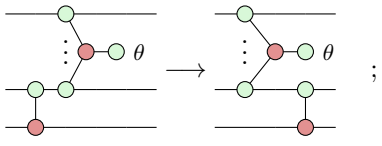
We call $\mathbf{C}_I$ and $\mathbf{C}_F$ the initial and final Clifford stages of the circuit, respectively, and $\mathbf{C}_M$ the main body of the circuit. We use this tripartite decomposition to allow us to condense the part of the circuit with non-trivial T-count in the main body, and to remove Clifford gates (H gates in particular) to the initial and final Clifford phases to the extent that this is possible.

Phase-gadgetisation.

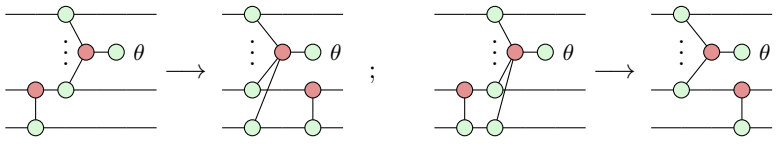
Through appropriate substitution of H gates by gadgets as in Eqn. (15), and substitution of CCZ with $\pi/4$-parity-phase operations as in Eqn. (6), we may transform the main body of the circuit so that it only contains SWAP gate, X gates, CNOT gates, CZ gates, and various phase gadgets (including powers of the T gate). We wish to transform this into a circuit consisting only of phase gadgets, by commuting everything apart from phase gadgets either to the beginning of the main body (and then removing it to the initial Clifford phase) or to the end of the main body (and then removing it to the final Clifford phase). In particular, we commute all SWAP, measurement, and X operations to the end of the circuit; we commute all preparation operations to the beginning of the circuit; and we commute each CNOT operation either to the beginning or the end according to a simple heuristic (described in Appendix B). This may transform various $D_{S,t}$ gates by Eqns. (3), changing the set S involved and/or negating the phase, according to the following commutation relations:



(17)



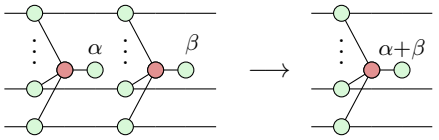
(18)



(19)

Phase gadget fusion.

A final simplifying technique is to simply multiply together any phase gadgets acting on the same set S of qubits:



. (20)

In some cases, this will reduce the T count by turning two gadgets with phases $\alpha = \frac{1}{4}k_1\pi$ and $\beta = \frac{1}{4}k_2\pi$ (for k_1 and k_2 odd) into a single gadget with phase $\alpha + \beta = \frac{1}{4}(k_1 + k_2)\pi$, where $k_1 + k_2$ is even.

4.2 Circuit translation procedure

Given a unitary circuit $\mathbf{C}$ over the gate-set $\{X, \mathrm{CNOT}, \mathrm{CCNOT}, Z, \mathrm{C}Z, \mathrm{CC}Z, H, S, T, \mathrm{SWAP}\}$, we transform $\mathbf{C}$ as follows:

1. We first replace CCNOT operations in $\mathbf{C}$ with $(\mathbb{1}\otimes\mathbb{1}\otimes H)\,\mathrm{CC}Z\,(\mathbb{1}\otimes\mathbb{1}\otimes H)$, yielding a circuit $\mathbf{C}'$.
2. Transform $\mathbf{C}' \to \mathbf{C}'_F \circ \mathbf{C}'_M \circ \mathbf{C}'_I$, with an initial Clifford stage $\mathbf{C}'_I$, a final Clifford stage $\mathbf{C}'_F$, and a main body $\mathbf{C}'_M$, using the procedure `moveH` to reduce the number of Hadamard gates in $\mathbf{C}'_M$ as much as possible.
3. Substitute the H gates in $\mathbf{C}'_M$ with Hadamard gadgets as in Eqn. (15), using a fresh bit label for each measurement outcome; and decompose CCZ operations in $\mathbf{C}$ using the formula of Eqn. (6), and represent T gates (on some qubit j) by $D_{\{j\},3}$. Call the resulting circuit $\mathbf{C}_M$.
4. We gadgetize $\mathbf{C}_M$ by commuting all gates which are not single-qubit phase gates or phase gadgets to the beginning or the end, removing these to the initial or final Clifford stages. This will generally add some number of measurements, and classically-conditioned Clifford operations, to the final Clifford stage, and some qubit preparations to the initial Clifford stage. This realises a transformation of circuits $\mathbf{C}'_F \circ \mathbf{C}_M \circ \mathbf{C}'_I \to \mathbf{C}_F \circ \mathbf{C}'_\phi \circ \mathbf{C}_I$.
5. As $\mathbf{C}'_\phi$ is now a homogeneous circuit of phase gadgets, we may commute them past one another to fuse gadgets on common subsets, yielding a circuit $\mathbf{C}_\phi$.
6. Apply the randomised procedure for applying PHAGE tactics based on spider nest identities described in Section 3.3.

Steps 1–5 realise a transformation $\mathbf{C} \to \mathbf{C}_F \circ \mathbf{C}_\phi \circ \mathbf{C}_I$. If the original circuit $\mathbf{C}$ acted on n qubits and had m Hadamard gates, then the number of Hadamard gates in $\mathbf{C}'_M$ which are replaced in Step 3 is some $\delta n \leqslant m$. Then the circuits $\mathbf{C}_I$, $\mathbf{C}_\phi$, and $\mathbf{C}_F$ all act on $N = n + \delta n$ qubits, and $\mathbf{C}_F$ has internal structure

$$\mathbf{C}_F \;=\; \tilde{\mathbf{C}}_F\,\mathbf{D}_{\delta n}\cdots\mathbf{D}_2\,\mathbf{D}_1\,, \tag{21}$$

where $\tilde{\mathbf{C}}_F$ is some general Clifford circuit, and the circuits $\mathbf{D}_j$ (for $1 \leqslant j \leqslant \delta n$) consist of the j^{th} measurement in the $|\pm\rangle$ basis with outcome s_j (denoted in ZX notation by a light green "π,$\{s_j\}$" node), followed by $\mathcal{D}_k^N$ operations conditioned on the outcome s_j.

In some instances, we find a significant reduction in the T-count simply from the fusion of phase gadgets in Step 5 of this transformation. These improvements are similar to those seen in Refs. [26, 34]. However, the purpose of this circuit transformation (as Ref. [23]) is to isolate a circuit $\mathbf{C}_\phi$ consisting entirely of $\mathcal{D}_3^N$ operations for some N, on which we can apply the PHAGE tactic of Step 6.

Note that δn, the number of additional "auxiliary" qubits involved in the circuit, is bounded above by how many Hadamard gates are either involved in $\mathbf{C}$ or are introduced from the decomposition of CCNOT gates. More precisely, it depends on how many of these gates can be commuted from the "main body" of $\mathbf{C}$ to the initial or final Clifford stages. For a circuit consisting of M gates, a bound for $N = n + \delta n$ which is substantially better than $N \leqslant n + M$ will be difficult to obtain, without some knowledge of the structure of $\mathbf{C}$. In several cases, we find that many or all of these Hadamard gates can be eliminated from the main body of the circuit: so, $N \leqslant n + M$ is likely a loose upper bound in a large number of practical examples.

The largest contributions to the asymptotic run-time of the procedure above are the complexity of `moveH` in Step 2; the cumulated complexity of computing the heuristic for moving Clifford gates out of the main body of the circuit in Step 4; and the complexity of performing a PHAGE tactics in Steps 5 and 6. For M the number of gates in the input circuit, the procedure `moveH` and the procedure to commute CNOT gates out the main body take time $O(M^2)$, essentially due to repeatedly commuting individual gates past $O(M)$ other gates (or computing the potential cost of doing so, in the case of the heuristic used for determining the direction to move CNOT gates). We use a hash table to store homogeneous circuits, allowing essentially $O(1)$ time to look up the phase associated with a phase gadget acting on a particular subset (which we set to 0 when no such phase gadget is present). In Step 5, fusing together pairs of phase gadgets can be made a part of initialising this hash table, and so takes time $O(M)$. In Step 6, applying a PHAGE tactic associated with some given identity $\mathcal{K}$ (which acts on at most 5 qubits) takes time $O(1)$; performing this for each of the 64 identities in $\mathcal{F}_4^{(4)} \cup \mathcal{F}_5^{(5)}$ on R uniformly random subsets takes time $O(R) = O(1)$, for R independent of M. Thus our procedure runs in time $O(M^2)$.

5 Results

We realised our techniques in Haskell code [7]. All but two of the circuits were obtained from Ref. [30]: the circuits "GF(2^{256}) Mult" and "GF(2^{512}) Mult" were obtained instead from Ref. [28]. With one exception, we ran our code for these benchmarks on a 2.5 GHz Intel Core i7 processor and 8 GB of 1867 MHz LPDDR3 memory, running Ubuntu Linux 18.04.4. The largest single benchmark circuit, "GF(2^{512}) Mult", was instead reduced on Dalhousie University's Mathstat Cluster [11]. The results are presented as Table 1, including comparisons to the best known reductions for which recorded times are available.[6]

Our results do not include an account of the cost of the Clifford group operations. These are also of interest in principle, though these will likely be significantly less expensive than T gates in the error-corrected setting in which the T-count becomes a meaningful quantity to reduce. We also do not describe the T-depth of our circuits, which may also be independently optimised from the T-count itself [4].

The circuits which were obtained using our techniques may be found on GitHub [7]. As our main aim was to consider reductions in T-count, our algorithm ignores the possibility that the measurement outcomes on the auxiliary qubits could be anything but $|+\rangle$: in the event of a $|-\rangle$ outcome, additional Clifford group operations would be required, which however would not affect the T-count. We verified our circuits using `feynver` [2], which was recently extended to accomodate circuits involving post-selection of $|+\rangle$ states on qubits which are maximally entangled with a set of other qubits.

[6] We do not present the best known T counts which do *not* have recorded times. We do note that for two of our results (for the circuits Mod Red_{21} and RC Adder_6) which improve on the known timed results, there are recorded untimed results which are still better: these may be found in Ref. [14].

Table 1 Comparison of our techniques to previously reported results. • In each case, "prev. opt." represents the best T-count with a time record (an asterisk indicates that the recorded time is an upper bound). For some circuits, better reductions without times have been reported: those indicated by [(a)] have a better reduction reported in Ref. [26], and those indicated by [(b)] have a better reduction reported in Ref. [14]. Where it was feasible to verify the correctness of our reduction with `feynver`, this is indicated; in all other cases the verification was too computationally expensive to carry out. • In each case, we also compare the number δn of additional "auxiliary" qubits required by our decomposition, to that of Ref. [23] (where results are available); in the case of [(c)], we may only infer an upper bound on the number of auxiliary qubits used by Ref. [23]. • In our results, those T-counts which are indicated in bold are those which reproduce or surpass the T-count of the best previously known timed result. Those with an asterisk also match or surpass the best previously known untimed result. • All results of Ref. [23] were obtained on the University of Sheffield's Iceberg HPC cluster. All results of Ref. [31] were obtained on a machine with a 2.9 GHz Intel Core i5 processor and 8 GB of 1867 MHz DDR3 memory, running OS X El Capitan. All of our results were obtained on a machine with a 2.5 GHz Intel Core i7 processor and 8 GB of 1867 MHz LPDDR3 memory, running Ubuntu Linux 18.04.4 — except those indicated by [(d)], which were obtained on Dalhousie University's Mathstat Cluster [11].

Circuit	**# qubits**			***T* count & optimisation**						
	n input	δn [23]	δn (ours)	init. #T	final #T (prev. opt.)	Ref.	time (s)	final #T (our results)	time (s)	Verified? (`feynver`)
Adder_8	24	71	41	399	212 [(a)]	[23]	227.81	**176 ***	24.62	✓
Barenco Tof_3	5	3	3	28	14 [(b)]	[23]	0.01*	**13 ***	0.07607	✓
Barenco Tof_4	7	7	7	56	24	[23]	0.45	25	1.884	✓
Barenco Tof_5	9	11	11	84	34	[23]	1.94	37	13.76	✓
Barenco Tof_{10}	19	31	31	224	84	[23]	460.33	97	24.49	✓
CSLA MUX_3	15	17	6	70	40 [(b)]	[23]	3.73	44	18.01	✓
CSUM MUX_9	30	12	12	196	74 [(a)]	[23]	36.57	84	23.98	✓
GF(2^4) Mult	12	7	0	112	56 [(b)]	[23]	0.55	**53 ***	0.8180	✓
GF(2^5) Mult	15	9	0	175	90 [(b)]	[23]	6.96	**88 ***	4.279	✓
GF(2^6) Mult	18	11	0	252	132 [(b)]	[23]	121.16	**128 ***	7.894	✓
GF(2^7) Mult	21	13	0	343	185 [(a)]	[23]	153.75	**167 ***	27.21	✓
GF(2^8) Mult	24	15	0	448	216 [(a)]	[23]	517.63	229	95.63	✓
GF(2^9) Mult	27	17	0	567	295	[23]	3213.53	306	24.79	✓
GF(2^{10}) Mult	30	19	0	700	351	[23]	23969.1	357	24.65	✓
GF(2^{16}) Mult	48	31	0	1 792	922	[23]	76312.5	972	25.65	✓ [(d)]
GF(2^{32}) Mult	96	–	0	7 168	4 128	[31]	1.834	**3 936 ***	26.07	✓ [(d)]
GF(2^{64}) Mult	192	–	0	28 672	16 448	[31]	58.341	**15 865 ***	29.73	–
GF(2^{128}) Mult	384	–	0	114 688	65 664	[31]	1744.746	**64 461 ***	48.78	–
GF(2^{131}) Mult	393	–	0	120 127	69 037	[31]	1953.353	**67 772 ***	53.39	–
GF(2^{163}) Mult	489	–	0	185 983	106 765	[31]	4955.927	**105 182 ***	66.27	–
GF(2^{256}) Mult	768	–	0	458 752	–	–	–	**260 539 ***	137.1	–
GF(2^{512}) Mult	1536	–	0	1 835 008	–	–	–	**1 046 964 ***	850.7 [(d)]	–
Mod5_4	5	6	0	28	16 [(b)]	[31]	0.001*	**7 ***	0.00899	✓
Mod Adder_{1024}	28	$\leqslant 270$ [(c)]	304	1 995	978	[23]	665.5	1 010	27.56	✓ [(d)]
Mod Mult_{55}	9	10	3	49	28 [(a)]	[23]	0.02	**19 ***	0.5775	✓
Mod Red_{21}	11	17	17	119	69 [(b)]	[23]	0.59	**65**	27.68	✓
QCLA Adder_{10}	36	28	25	238	157	[23]	366.1	**147 ***	24.96	✓
QCLA Com_7	24	19	18	203	81	[23]	170.77	84	24.21	✓
QCLA Mod_7	26	58	58	413	221 [(a)]	[23]	289.77	233	24.26	✓ [(d)]
RC Adder_6	14	21	10	77	45 [(b)]	[23]	0.97	**38**	30.70	✓
NC Toff_3	5	2	2	21	13	[23]	0.01*	**13 ***	0.04005	✓
NC Toff_4	7	4	4	35	19	[23]	0.06	**19 ***	0.5322	✓
NC Toff_5	9	11	6	49	25	[23]	0.4	26	2.910	✓
NC Toff_{10}	19	16	16	119	55	[23]	44.78	60	28.01	✓
VBE Adder_3	10	4	4	70	20	[23]	0.15	**20 ***	1.896	✓

6 Discussion

Our results show that our techniques, simple as they are, are competitive with the best known techniques for reducing T count. We expect that better results should be achievable by a more refined approach to using these concepts, within the more general framework which we have described of deploying PHAGE tactics. It is not clear which further ideas may prove useful, however. For instance, in experiments for how we might choose subsets to apply PHAGE tactics to, we found that it was not helpful to bias the sets of qubits towards those qubits which were acted on by many T-phase gadgets. More work will be required to find effective ways to bias or to narrow down the ways in which spider nest identities are used to simplify homogeneous circuits.

It is remarkable that the run-times for our results in Table 1 are not more varied. Over half of our results were obtained in an amount of time between 1 and 100 seconds, for circuits over which other leading techniques [23, 31] had times which ranged over more than six orders of magnitude. Indeed, in our tests on larger circuits (and in line with the asymptotic analysis of Section 4.2), we found that the most computationally expensive part of our procedure was the relatively mundane `moveH` and CNOT-commutation subroutines. Refining these elementary steps may provide yet further gains. Expanding the complexity of the subroutines to apply PHAGE tactics may also yield further gains without substantial increases in run-time.

We note an optimisation problem of interest is motivated by gadgetizing Hadamard gates as in Step 3. Simply put: given an n-qubit circuit with M gates over the gate-set $\{X, Z, S, \mathrm{CNOT}, \mathrm{C}Z, T, \mathrm{CC}Z\}$, to obtain an equivalent (unitary) circuit with the minimum number of H gates in between the first and the last non-Clifford gate.[7] Should this problem be solvable in $O(M^2 \operatorname{poly}\log(M))$ time, this may further contribute to the effectiveness of our approach to T-count reduction.

Finally, we remark that while the benchmarks which we have tested have become a commonplace standard for the evaluation of such techniques, they consist entirely of circuits to realise permutation operations which would not in themselves be difficult to realise classically. (Some of these, such as the "GF(2^n) Mult" series, may be motivated on the grounds of cryptography; albeit this motivation may become less important if standard cryptographic practise incorporates post-quantum cryptography.) A larger range of circuits, including ones are motivated by the more likely practical applications of fault-tolerant quantum computation, should be of general interest for future benchmark tests.

References

1 S. Aaronson and D. Gottesman. Improved simulation of stabilizer circuits. *Phys. Rev. A*, 70:052328, 2004. `arXiv:quant-ph/0406196`.

2 Matthew Amy. Towards large-scale functional verification of universal quantum circuits. In *Proceedings of QPL 2018*, pages 1–21, 2018. [arXiv:1901.09476]; see also [`https://github.com/meamy/feynman`]. `doi:10.4204/EPTCS.287.1`.

3 Matthew Amy, Jianxin Chen, and Neil J. Ross. A finite presentation of cnot-dihedral operators. *Electronic Proceedings in Theoretical Computer Science*, 266:84–97, 2018. [arXiv:1701.00140]. `doi:10.1007/978-3-642-12821-9_4`.

4 Matthew Amy, Dmitri Maslov, and Michele Mosca. Polynomial-time T-depth optimization of Clifford+T circuits via matroid partitioning. *IEEE Transactions on Computer-Aided*

[7] It seems plausible that this problem would remain equally difficult without CCZ gates.

Design of Integrated Circuits and Systems, 33(10):1476–1489, 2014. [arXiv:1303.2042]. doi:10.1109/TCAD.2014.2341953.

5 Matthew Amy, Dmitri Maslov, Michele Mosca, and Martin Roetteler. 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):818–830, 2013. [arXiv:1206.0758]. doi:10.1109/TCAD.2013.2244643.

6 Matthew Amy and Michele Mosca. T-count optimization and Reed-Muller codes. *IEEE Transactions on Information Theory*, 65(8):4771–4784, 2019. [arXiv:1601.07363]. doi:10.1109/TIT.2019.2906374.

7 Xiaoning Bian. "stomp-code" github. URL: https://github.com/onestruggler/stomp-code/tree/8df4f46228c2f413e0cf5f8b6d25c20b6460fc0e.

8 Sergey Bravyi, Dan Browne, Padraic Calpin, Earl Campbell, David Gosset, and Mark Howard. Simulation of quantum circuits by low-rank stabilizer decompositions. *Quantum*, 3:181, September 2019. [arXiv:1808.00128]. doi:10.22331/q-2019-09-02-181.

9 Sergey Bravyi and David Gosset. Improved classical simulation of quantum circuits dominated by Clifford gates. *Physical Review Letters*, 116:250501, 2016. [arXiv:1601.07601]. doi:10.1103/PhysRevLett.116.250501.

10 Earl T. Campbell and Mark Howard. A unified framework for magic state distillation and multi-qubit gate-synthesis with reduced resource cost. *Physical Review A*, 95:022316, 2017. [arXiv:1606.01904]. doi:10.1103/PhysRevA.86.022316.

11 Dalhousie University Mathstat Cluster. URL: https://www.mathstat.dal.ca/cluster/doku.php.

12 Bob Coecke and Ross Duncan. Interacting quantum observables: categorical algebra and diagrammatics. *New Journal of Physics*, 13:043016, 2011. [arXiv:0906.4725]. doi:10.1088/1367-2630/13/4/043016.

13 Bob Coecke and Aleks Kissinger. *Picturing Quantum Processes: A First Course in Quantum Theory and Diagrammatic Reasoning*. Cambridge University Press, 2017. doi:10.1017/9781316219317.

14 Niel de Beaudrap, Xiaoning Bian, and Quanlong Wang. Techniques to reduce $\pi/4$ -parity phase circuits, motivated by the zx calculus. In *to appear in Proceedings of QPL 2019*, 2019. [arXiv:1911.09039].

15 Niel de Beaudrap, Ross Duncan, Dominic Horsman, and Simon Perdrix. Pauli fusion: a computational model to realise quantum transformations from zx terms. In *Proceedings of QPL 2019*, page to appear, 2019. [arXiv:1904.12817].

16 Niel de Beaudrap and Dominic Horsman. The zx calculus is a language for surface code lattice surgery. *Quantum*, 4:218, 2020. [arXiv:1704.08670]. doi:10.22331/q-2020-01-09-218.

17 Ross Duncan, Aleks Kissinger, Simon Perdrix, and John van de Wetering. Graph-theoretic simplification of quantum circuits with the zx-calculus. [arXiv:1902.03178], 2019.

18 Ross Duncan and Simon Perdrix. Rewriting measurement-based quantum computations with generalised flow. In Samson Abramsky, Cyril Gavoille, Claude Kirchner, Friedhelm Meyer auf der Heide, and Paul G. Spirakis, editors, *Automata, Languages and Programming*, pages 285–296, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg. doi:10.1007/s10472-009-9141-x.

19 Simon Perdrix Emmanuel Jeandel and Renaud Vilmart. Completeness of the zx-calculus. [arXiv:1903.06035], 2019.

20 Craig Gidney. Halving the cost of quantum addition. *Quantum*, 2:74, 2018. [arXiv:1709.06648]. doi:10.1007/s11128-011-0297-z.

21 David Gosset, Vadym Kliuchnikov, Michele Mosca, and Vincent Russo. An algorithm for the t-count. *Quantum Info. Comput.*, 14(15–16):1261–1276, November 2014. [arXiv:1308.4134].

22 D. Gottesman. Stabilizer codes and quantum error correction. 1997. Ph.D thesis. arXiv:quant-ph/9705052.

23 Luke E. Heyfron and Earl T. Campbell. An efficient quantum compiler that reduces t count. *Quantum Science and Technology*, 4(1):015004, 2018. [arXiv:1712.01557]. doi:10.1038/srep01939.

24 C. Horsman, A. G Fowler, S. Devitt, and R. Van Meter. Surface code quantum computing by lattice surgery. *New Journal of Physics*, 14(12):123011, 2012. [arXiv:1111.4022].

25 Cody Jones. Low-overhead constructions for the fault-tolerant toffoli gate. *Phys. Rev. A*, 87:022328, February 2013. [arXiv:1212.5069]. doi:10.1103/PhysRevA.87.022328.

26 Aleks Kissinger and John van de Wetering. Reducing t-count with the zx-calculus. [arXiv:1903.10477], 2019.

27 Daniel Litinski. A Game of Surface Codes: Large-scale quantum computing with lattice surgery. *Quantum*, 3:128, 2019. [arXiv:1808.02892]. doi:10.1103/PhysRevB.96.205413.

28 Dmitri Maslov. *Reversible Logic Synthesis Benchmarks page*. Accessed February 2020. URL: http://webhome.cs.uvic.ca/~dmaslov.

29 Giulia Meuli, Mathias Soeken, Earl Campbell, Martin Roetteler, and Giovanni De Micheli. The role of multiplicative complexity in compiling low T-count oracle circuits. [arXiv:1908.01609], 2019.

30 Yunseong Nam, Neil J. Ross, Yuan Su, Andrew M. Childs, and Dmitri Maslov. "optimiser" github. https://github.com/njross/optimizer.

31 Yunseong Nam, Neil J. Ross, Yuan Su, Andrew M. Childs, and Dmitri Maslov. Automated optimization of large quantum circuits with continuous parameters. *npj Quantum Information*, 4(1):23, 2018. [arXiv:1710.07345]. doi:10.1038/s41534-018-0072-4.

32 Michael A Nielsen and Isaac Chuang. *Quantum computation and quantum information*. Cambridge University Press, Cambridge UK, 2000.

33 Renaud Vilmart. A Near-Optimal Axiomatisation of ZX-Calculus for Pure Qubit Quantum Mechanics. In *34th Annual ACM/IEEE Symposium on Logic in Computer Science*, pages 1–10, 2019. [arXiv:1812.09114]. doi:10.1109/LICS.2019.8785765.

34 Fang Zhang and Jianxin Chen. Optimizing t gates in clifford+t circuit as $\pi/4$ rotations around paulis. [arXiv:1903.12456], 2019.

A ZX diagram reference

The ZX calculus — first developed by Coecke and Duncan [12] (see also Refs. [13, 33, 19] for updated treatments, and Refs. [17, 16, 15, 26] for applications to quantum technology) — is a relatively recently developed notation for quantum operations, equipped with rules (the "calculus" part) to compute with this notation. This article does not make explicit use of the "calculus" part of the ZX calculus: while it *does* make statements about equivalences of diagrams which *could* be shown using the calculus, these can and should be understood in the same way that other papers make statements of equivalences of conventional circuit diagrams.

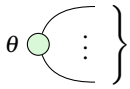
We use ZX notation at various points to describe quantum circuits, including circuits with classically controlled operations and non-local unitaries such as $\pi/4$-parity-phase operations. The ZX diagrams in this article can be read merely as a slightly unusual (but convenient) circuit notation. In this Appendix, we provide a reference for this notation, serving also as a glossary of sorts for various operations as they are represented in ZX diagrams, to allow readers to understand our results as well as conventional circuit diagrams would.

A.1 General statements

For the purposes of this article (and essentially all other practical purposes), ZX diagrams are representations of tensor networks. To represent quantum circuits, it is common to choose a direction in which to read the diagrams from "input" to "output". (In our paper, we draw

these diagrams with input on the left and output on the right, as with the usual circuit notation.) The ZX diagrams of our work are composed of three different kinds of tensor nodes:

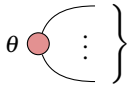
- **"Green" nodes** (which are lighter coloured in our article), which may have any number of indices, and as a tensor represents a sort of GHZ state over the standard basis. If a phase parameter θ is provided, the tensor also involves a relative phase of $e^{i\theta}$ between the two terms; otherwise $\theta = 0$ is assumed (and there is no relative phase).



$$= \quad |0\rangle^{\otimes n} + e^{i\theta} |1\rangle^{\otimes n} \tag{22}$$

  In principle, we also permit the border case of $n = 0$, in which case this represents the "tensor" $|0\rangle^{\otimes 0} + e^{i\theta}|1\rangle^{\otimes 0} = (1 + \cos(\theta)) + i\sin(\theta)$; though we don't make use of such nodes in our results.

- **"Red" nodes** (which are darker coloured in our article), which may have any number of indices, and are similar to green nodes except that they are expressed in terms of the $\{|+\rangle, |-\rangle\}$ basis.



$$= \quad |+\rangle^{\otimes n} + e^{i\theta} |-\rangle^{\otimes n} \tag{23}$$

- **"Hadamard" boxes**, which represent the usual 2×2 unitary Hadamard matrix.

$$= \quad |+\rangle\langle 0| + |-\rangle\langle 1| \tag{24}$$

To represent operations taking some qubits as input, we change of some of the "kets" in the tensor nodes to "bras" — but as $|0\rangle$, $|1\rangle$, $|+\rangle$ and $|-\rangle$ are real vectors, this change does not affect any of the tensor coefficients. This allows us to be flexible with our diagrams, and avoid committing to the indices of each node as being explicitly an "input" or an "output", unless it is a free index of the whole diagram. (In particular, this allows us to draw some closed indices by *vertical* wires, without confusion.)

In the rest of this appendix, we describe some simple examples (and simple extensions) of this notation, which the interested reader should find themselves able to verify by routine calculation.

A.2 Single-node diagrams

With (light) green or (dark) red nodes of degree 1, we may easily represent states of the $\{|0\rangle, |1\rangle\}$ basis or $\{|+\rangle, |-\rangle\}$ basis, albeit supernormalised by a factor of $\sqrt{2}$.

$$= |+\rangle^{\otimes 1} + |-\rangle^{\otimes 1} = \sqrt{2}\,|0\rangle\,; \tag{25}$$

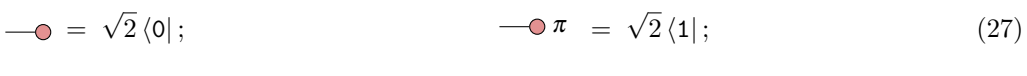
$$= |+\rangle^{\otimes 1} - |-\rangle^{\otimes 1} = \sqrt{2}\,|1\rangle\,; \tag{25}$$

$$= |0\rangle^{\otimes 1} + |1\rangle^{\otimes 1} = \sqrt{2}\,|+\rangle\,; \tag{26}$$

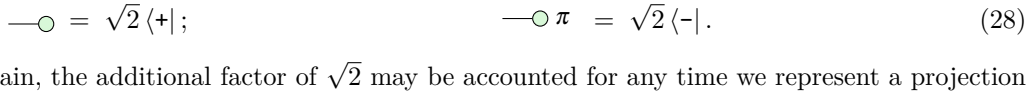
$$= |0\rangle^{\otimes 1} - |1\rangle^{\otimes 1} = \sqrt{2}\,|-\rangle\,. \tag{26}$$

More generally, green degree-1 nodes may be used to represent newly prepared qubits in the XY plane of the Bloch sphere, and red degree-1 nodes may be used to represent newly prepared qubits in the YZ plane of the Bloch sphere, up to the same supernormalisation of $\sqrt{2}$. This additional factor of $\sqrt{2}$ does not affect our results: the additional factor may be accounted for any time we represent the preparation of a qubit in one of these states.

We may also represent single-qubit measurements by degree-1 nodes oriented in the opposite direction. As re-orienting edges from the right of a node to the left corresponds to turning $|0\rangle$ to $\langle 0|$, turning $|1\rangle$ to $\langle 1|$, and so forth, we then have

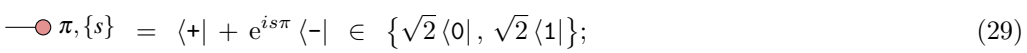


(27)

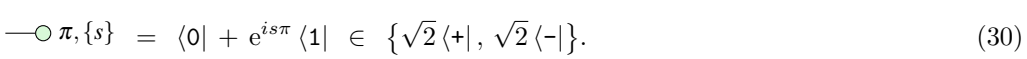


(28)

Again, the additional factor of $\sqrt{2}$ may be accounted for any time we represent a projection of a qubit in one of these states. To represent a measurement which may yield either $|0\rangle$ or $|1\rangle$, or either $|+\rangle$ or $|-\rangle$, we may introduce a variable $s \in \{0, 1\}$ representing whether a relative phase of π is absent in the result ($s = 0$, for the states $|0\rangle$ or $|+\rangle$) or present in the result ($s = 1$, for the states $|1\rangle$ or $|-\rangle$). We then represent measurement in the $\{|0\rangle, |1\rangle\}$ basis and the $\{|+\rangle, |-\rangle\}$ basis respectively as



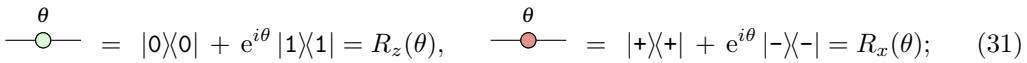
(29)



(30)

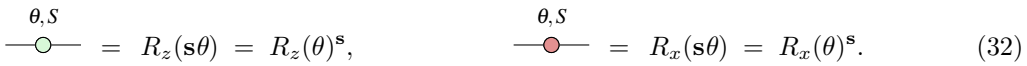
The bit s is in effect a random variable representing the measurement outcome.

In other ZX diagrams (including on nodes of degree 2 or higher), we may use a set $S = \{s_1, s_2, \ldots\}$ in place of the set $\{s\}$. This indicates a node in which the presence or absense of the phase of π depends on the parity $(s_1 \oplus s_2 \oplus \cdots)$ of the entire set S, rather than on the single bit s. For example, we may represent Z rotations and X rotations each by a single node of degree 2:



(31)

Then, the following diagrams represent the same operations, conditioned on the parity $\mathbf{s} = \bigoplus_j s_j$ of a set of bits $S = \{s_1, s_2, \ldots\}$:

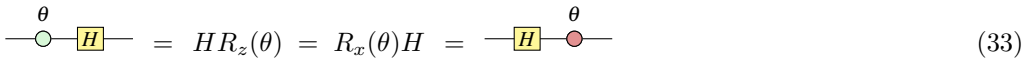


(32)

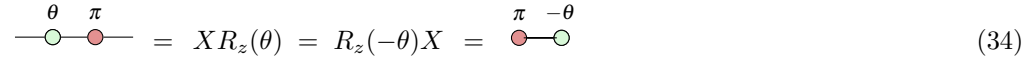
This feature of the ZX calculus does not play a prominent role in our work, but is present in our treatment of the Hadamard gadget (Eqn. (15) on page 10) and in principle useful to represent the circuits which we would obtain by representing conditionally-controlled Clifford operations in the ZX calculus.

A.3 Two-node diagrams

Diagrams of more than one node can be easily constructed simply by composing nodes on their edges. In many cases, this has the same meaning as in conventional quantum circuit diagrams (with the same "feature" that the algebra is read right-to-left, even though the diagram is read left-to-right): for example,

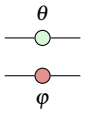


(33)



(34)

As with circuit diagrams, we may also represent the tensor product of operations by representing operations happening on different wires in parallel — for example:



$$= \; R_z(\theta) \otimes R_x(\varphi). \tag{35}$$

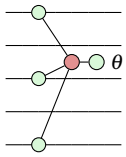
Not all "compositions" of nodes take these forms, however: in general we may compose any two nodes simply by connecting their edges (corresponding to contracting the shared indices of the tensor nodes). An especially important case in point is the way that CNOT operators are represented as ZX terms. As with single-qubit states, the usual representation of CNOT by ZX diagrams is not precisely normalised:

$$\begin{aligned} &= \; |0\rangle\langle 0| \otimes \langle 0|+\rangle \otimes |+\rangle\langle +| \; + \; |0\rangle\langle 0| \otimes \langle 0|-\rangle \otimes |-\rangle\langle -| \\ &\qquad + \; |1\rangle\langle 1| \otimes \langle 1|+\rangle \otimes |+\rangle\langle +| \; + \; |1\rangle\langle 1| \otimes \langle 1|-\rangle \otimes |-\rangle\langle -| \\ &= \; \tfrac{1}{\sqrt{2}}\,|0\rangle\langle 0| \otimes \mathbb{1} + \tfrac{1}{\sqrt{2}}\,|1\rangle\langle 1| \otimes X \; = \; \tfrac{1}{\sqrt{2}}\,\mathrm{CNOT}. \end{aligned} \tag{36}$$

(Again, the subnormalisation of this diagram does not affect our analysis, and can in principle be accounted for in the ZX representation of any circuit involving CNOT gates.) Note that the shared wire between the red and green dot does not have a specific interpretation as an "input" or an "output" of either — nor is this necessary to provide the interpretation of the diagram as an operator.

A.4 Multi-node diagrams

Composing the diagrams above, in series or in parallel (and with appropriate accounting for normalisation), suffices to represent an arbitrary unitary operation by the (slightly redundant) gate set consisting of arbitrary X and Z rotations, Hadamard gates, and CNOT operations. We may also more directly represent somewhat more "exotic" operators using ZX diagrams, and $\pi/4$-parity-phase operations are in this case the most relevant example: for instance,



$$\begin{aligned} &= \; \Big(\langle 0|_d \; + \; \mathrm{e}^{i\theta}\,\langle 1|_d\Big)\Big(|+\rangle_d\langle +{+}{+}|_{abc} + |-\rangle_d\langle -{-}|_{abc}\Big) \\ &\qquad \times \Big(|0\rangle\langle 0|_1 \otimes |0\rangle_a + |1\rangle\langle 1|_1 \otimes |1\rangle_a\Big)\Big(|0\rangle\langle 0|_3 \otimes |0\rangle_b + |1\rangle\langle 1|_3 \otimes |1\rangle_b\Big) \\ &\qquad \times \Big(|0\rangle\langle 0|_5 \otimes |0\rangle_c + |1\rangle\langle 1|_5 \otimes |1\rangle_c\Big) \\ &= \; \tfrac{1}{2\sqrt{2}} \sum_{a,b,c\in\{0,1\}} \Big(\langle 0|_d \; + \; \mathrm{e}^{i\theta}\,\langle 1|_d\Big)\Big(|+\rangle_d \; + \; (-1)^{a+b+c}\;|-\rangle_d\Big) \otimes |a,b,c\rangle\langle a,b,c|_{1,3,5} \\ &= \; \tfrac{1}{4} \sum_{a,b,c\in\{0,1\}} \Big(1 + \mathrm{e}^{i\theta} + (-1)^{a+b+c} - (-1)^{a+b+c}\mathrm{e}^{i\theta}\Big)\;|a,b,c\rangle\langle a,b,c|_{1,3,5} \\ &= \; \tfrac{1}{2} \sum_{\substack{a,b,c\in\{0,1\} \\ a\oplus b\oplus c=0}} |a,b,c\rangle\langle a,b,c|_{1,3,5} \quad + \quad \tfrac{1}{2} \sum_{\substack{a,b,c\in\{0,1\} \\ a\oplus b\oplus c=1}} \mathrm{e}^{i\theta}\;|a,b,c\rangle\langle a,b,c|_{1,3,5} \\ &= \; \tfrac{1}{2}\,\mathrm{e}^{i\theta/2}\,\exp\!\big(\tfrac{1}{2}i\theta(Z\otimes\mathbb{1}\otimes Z\otimes\mathbb{1}\otimes Z)\big). \end{aligned} \tag{37}$$

Again, the subnormalisation by a factor of $\frac{1}{2}$ does not affect our analysis, which is in principle about products of $D_{S,3}$ operators — merely denoted in our work by these phase gadgets, for convenience — which are proportional to the identity by a global phase.

The existence of rules for transforming ZX diagrams allows us to reason (*i.e.*, to compute) effectively about these diagrams without the need to expand their meaning algebraically as we have been doing in this Appendix. This has particularly motivated our use of the ZX calculus in our work, as a convenient notational tool and also as a means by which we performed our analysis.

For more information about the ZX calculus, and in particular for resources to learn about these diagrammatic computational methods, the interested reader is invited to visit [zxcalculus.com].

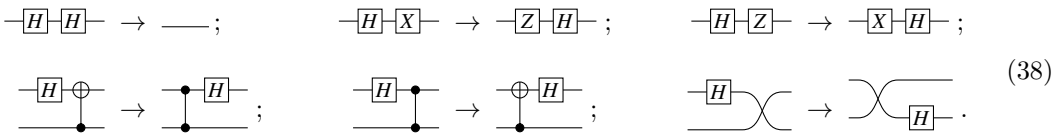
B Details of the moveH subroutine and CNOT-commutation heuristic

In this Appendix, we describe our procedures for H gate extraction and CNOT gate extraction (used in Steps 2 and 4 of the procedure described in Section 4.2) on a high level. For more details, the interested reader may view our source code on Github [https://github.com/onestruggler/fast-stomp].

B.1 The moveH subroutine

Our procedure for extracting H gates from a circuit are built on a subroutine moveH, which attempts to move each H gate as far to the right (the end of the circuit) as possible.

Representing the circuit as a list of gates in a particular order (without parallelisation), this procedure looks for the first H gate, and attempts to move it to the right. In doing so, it makes use of several simple commutation relations or opportunities for cancellation, for example:



(38)

If the procedure moves the H gate to a point that it precedes a second H gate, it proceeds recursively to attempt to move the second H gate before continuing with the first. When the procedure is finished attempting (successfully or otherwise) to move the second H gate, it returns to the task of moving the first — moving this gate past the other H gate, if the attempt to move it ended in failure. This process continues until the procedure has stopped trying to move what originally was the first H gate.

In attempting to move H gates, moveH may encounter situations in which no progress is possible, without trying to move or cancel other kinds of gates. For instance: in a circuit consisting only of an H gate followed by four T gates on a single wire, it is possible to move the H gate to the end, but only after "pushing" the T gate which follows it to the right, accumulating the other phase gates to form a Z gate. In general, if moveH encounters a gate G for which there is no commutation rule provided, it attempts instead to push G forward, to commute with, accumulate with, or cancel against gates further to its right. In doing so, moveH may encounter yet another gate F for which G has no provided commutation relation, in which case moveH will attempt to move F further to the right, and so on.

In some cases, there are fruitful opportunities for multi-gate substitutions which either reduce the number of H gates, or allows an H gate to be moved further to the right. For instance:

- If in moving an H gate to the right we encounter an S gate followed by an H gate, `moveH` first tries to move the second H gate. If this fails, we may apply the transformation

(39)

  This reduces the number of H gates by 1. We then move the S and Z gates further to the right, then continue by moving the new H gate to the right.
- If in moving an H gate to the right we encounter the *control* qubit of a CNOT gate, followed by an H gate on either the control or target, we again first try to move the H gate. If this fails, we may apply one of the transformations

(40)

  This doesn't directly reduce the number of H gates, but may make it possible to move the later H gate and the CNOT gate to the right before continuing further, thereby providing an alternative for at least one of the two H gates to be moved further to the right.

The details of all commutation relations which we define for all of the gates are not important, except that it is important to define these rules in such a way that the procedure terminates (rather than repeatedly commute two gates such as T and CCZ past one another, in an attempt to cancel them so that an H gate can be moved to the right of both). Different techniques will lead to different performances in the ability of `moveH` to reduce the number of H gates which precede any non-Clifford gate.

B.2 CNOT movement heuristic

In Step 4, we move all operations which are not single-qubit phase operations or phase-parity operations out of the main body of the circuit. The way that CNOT gates are treated aims, roughly, to avoid generating phase-parity operations on very large subsystems, but does so in a way that attempts to avoid performing too much computation.

The heuristic used to determine which direction to move a CNOT operation is as follows. For each CNOT gate, from the first in the circuit to the last, we compute the following:

1. Compute the set P_L of all phase-parity gadgets to the left which act on the target but not the control of the CNOT, and the set M_L of phase-parity gadgets to the left which act on its target and control both.
2. Similarly, compute the set P_R of phase-parity gadgets to the right which act on the target but not the control of the CNOT, and the set M_R of phase-parity gadgets to the left which act on its target and control both.
3. If $P_L - M_L < P_R - M_R$, we prefer to move the CNOT to the left; otherwise we prefer to move it to the right.

If no other CNOT gate acted on any qubits in common with this left-most CNOT gate, the quantity P_L (respectively, M_L) would correctly indicate how many phase-parity gadgets would act on one more qubit (respectively, one fewer) if we commuted that CNOT to the left. The difference $P_L - M_L$ then indicates the net change in the cumulative number of qubits acted on by the phase-gadgets to the left of the CNOT. Similar remarks apply for $P_R - M_R$, albeit with the important caveat that this figure may be inaccurate if there are further CNOT gates to the right whose targets coincide with the control of the CNOT under consideration.

The approach taken to produce our results is as follows. For the left-most CNOT in the circuit, compute P_L, M_L, P_R, and M_R. Commute the CNOT gate to the left if $P_L - M_L < P_R - M_R$, and otherwise to commute it to the right. If in commuting it to the right we encounter another CNOT gate with which it does not commute, we also commute that CNOT gate to the right (and any CNOT gates with which *those* do not commute, *etc.*) Having done this, we compute compute P_L, M_L, P_R, and M_R for the leftmost remaining CNOT gate in the circuit, where these may depend on the commutations which occurred for the previous CNOT gate. We proceed in this way, recursively from left to right, until no more CNOT gates are in the main body of the circuit.