

**Classical simulation complexity of restricted
models of quantum computation**

by

Dax Enshan Koh

B.S., Stanford University (2011)

M.Sc., University of Waterloo (2013)

Submitted to the Department of Mathematics
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2019

© Massachusetts Institute of Technology 2019. All rights reserved.

Signature redacted

Author

Department of Mathematics

May 3, 2019

Signature redacted

Certified by

Peter W. Shor

Morss Professor of Applied Mathematics

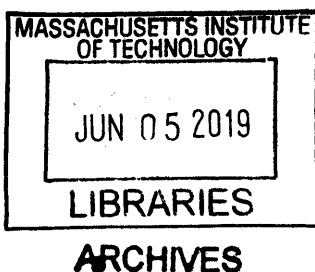
Thesis Supervisor

Signature redacted

Accepted by

Jonathan Kelner

Chairman, Department Committee on Graduate Theses



Classical simulation complexity of restricted models of quantum computation

by

Dax Enshan Koh

Submitted to the Department of Mathematics
on May 3, 2019, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

Abstract

Restricted models of quantum computation are mathematical models which describe quantum computers that have limited access to certain resources. Well-known examples of such models include the boson sampling model, extended Clifford circuits, and instantaneous quantum polynomial-time circuits. While unlikely to be universal for quantum computation, several of these models appear to be able to outperform classical computers at certain computational tasks, such as sampling from certain probability distributions. Understanding which of these models are capable of performing such tasks and characterizing the classical simulation complexity of these models—i.e. how hard it is to simulate these models on a classical computer—are some of the central questions we address in this thesis.

Our first contribution is a classification of various extended Clifford circuits according to their classical simulation complexity. Among these circuits are the conjugated Clifford circuits, which we prove cannot be efficiently classically simulated up to multiplicative or additive error, under certain plausible conjectures in computational complexity theory. Our second contribution is an estimate of the number of qubits needed in various restricted quantum computation models in order for them to be able to demonstrate quantum computational supremacy. Our estimate is obtained by fine-graining existing hardness results for these restricted models. Our third contribution is a new alternative proof of the Gottesman-Knill theorem, which states that Clifford circuits can be efficiently simulated by a classical computer. Our proof uses the sum-over-paths technique and establishes a correspondence between quantum circuits and a class of exponential sums. Our final contribution is a theorem characterizing the operations that can be efficiently simulated using a particular rebit simulator. An application of this result is a generalization of the Gottesman-Knill theorem that allows for the efficient classical simulation of certain nonlinear operations.

Thesis Supervisor: Peter W. Shor
Title: Morss Professor of Applied Mathematics

Acknowledgments

I would like to express my deep gratitude to my thesis supervisor Peter W. Shor for his unconditional support, encouragement and guidance throughout my time at MIT. Peter's creativity and mastery of the field has been a source of great inspiration for me.

I would like to thank my co-authors for their time, ideas and our many fruitful discussions: Scott Aaronson, Jacob D. Biamonte, Adam Bouland, Kaifeng Bu, Alexander M. Dalzell, Joseph F. Fitzsimons, Aram W. Harrow, Rolando L. La Placa, Zi-Wen Liu, Mauro E.S. Morales, Murphy Yuezhen Niu, Mark D. Penney, Christopher Perry, Robert W. Spekkens, Theodore J. Yoder, and Yechao Zhu. I have learned much from our collaborations. In addition, I would like to thank Mason Biamonte, Siong Thye Goh, Anand Natarajan, Hakop Pashayan, and many others with whom I have had many insightful discussions.

I have been fortunate to have had the opportunity to visit and collaborate with various research groups around the world. I would like to thank Huangjun Zhu, Robert Spekkens, and Beni Yoshida for hosting me on multiple visits to the Perimeter Institute for Theoretical Physics; Si-Hui Tan and Joseph Fitzsimons for hosting me at the Singapore University of Technology and Design; Hakop Pashayan and Stephen Bartlett for hosting me at the University of Sydney; Man-Hong Yung for hosting me at the Southern University of Science and Technology in Shenzhen; Alexander Rivosh and Andris Ambainis for hosting me at the University of Latvia; and Dirk Oliver Theis for hosting me at the University of Tartu. I would also like to thank Jacob D. Biamonte and the Deep Quantum Labs at the Skolkovo Institute of Science and Technology for hosting me as a research intern during the summer of 2018, and MISTI-Russia for making this possible.

I would like to thank members of my Thesis Examination Committee—Peter W. Shor, Michel X. Goemans, and Aram W. Harrow—for generously agreeing to oversee my thesis defense and for useful comments on this thesis. Also, I would like to thank members of my Qualifying Examination Committee—Peter W. Shor, Hung

Cheng, and Robert G. Gallager—for providing useful feedback during my qualifying examination.

I would like to acknowledge funding support from the National Science Scholarship awarded by the Agency for Science, Technology and Research (A*STAR), Singapore, as well as the Enabling Practical-scale Quantum Computing (EPiQC) Expedition, an NSF expedition in computing.

Last but not least, I would like to thank my family for their unwavering support and encouragement.

Contents

1	Introduction	23
1.1	Motivation	23
1.2	Organization and summary of results	27
2	Preliminaries	31
2.1	Notational guide	31
2.2	Quantum states, transformations and measurements	32
2.3	Quantum circuits	34
2.3.1	Examples of quantum gates	35
2.3.2	Universality	38
2.4	Pauli group	39
2.4.1	Properties of the Pauli group	39
2.4.2	\mathbb{F}_2 representation of the Pauli group	41
2.5	Clifford group	43
2.5.1	Elements of the Clifford group	45
2.6	Classical and quantum computational complexity theory	50
2.6.1	P, NP and the polynomial hierarchy	51
2.6.2	Complexity of counting	54
2.6.3	Space complexity	55
2.6.4	Classical randomized complexity	56
2.6.5	Bounded-error quantum polynomial time	58
2.6.6	Postselection	60
2.7	Notions of classical simulation of quantum computation	62

2.8	Restricted models of quantum computation	66
2.8.1	Clifford circuits	66
2.8.2	Instantaneous quantum polynomial-time circuits	68
2.8.3	Depth-one QAOA circuits	69
2.8.4	DQC1 circuits	73
2.8.5	Boson sampling model	73
3	Extended Clifford circuits and their classical simulation complexities	77
3.1	Motivation	78
3.2	Preliminary definitions and notations	80
3.3	Notions of classical simulation	84
3.4	Results and discussion	86
3.5	Proofs of main theorems	89
3.5.1	Rules for proving results in Table 3.1	89
3.5.2	Proof of Theorem 39: Strong(n) simulation of nonadaptive Clifford circuits with product inputs and computational basis outputs	90
3.5.3	Proof of Theorem 40: Strong(n) simulation of adaptive Clifford circuits with computational basis inputs and outputs	93
3.5.4	Proof of Theorem 46: Weak simulation of nonadaptive Clifford circuits with computational basis inputs and product outputs	95
3.5.5	Proof of Theorem 48: Strong(n) simulation of nonadaptive Clifford circuits with product inputs and computational basis outputs	98
3.5.6	Proof of Theorem 49: Strong(1) simulation of nonadaptive Clifford circuits with product inputs and outputs	99
3.5.7	Proof of Theorem 50: Weak(1) simulation of adaptive Clifford circuits with computational basis inputs and product outputs	100
3.5.8	Constructing circuits for 3-CNF formulas	101
3.5.9	Constructing C_f	102
3.5.10	Constructing Q_f	105

3.6	Concluding remarks	106
4	Conjugated Clifford circuits	107
4.1	Overview of results	107
4.1.1	Proof techniques	110
4.1.2	Relation to other works on modified Clifford circuits	113
4.2	Conjugated Clifford circuits	114
4.2.1	Postselection gadgets	115
4.3	Weak simulation of CCCs with multiplicative error	117
4.3.1	Classification results	117
4.3.2	Proofs of efficient classical simulation	120
4.3.3	Proofs of hardness	121
4.4	Weak simulation of CCCs with additive error	127
4.5	Evidence in favor of hardness conjecture	131
4.6	CCCs and other notions of simulation	134
4.7	Measurement-based quantum computing proof of multiplicative hardness for CCCs for certain U 's	136
4.8	Open Problems	146
5	How many qubits to reach quantum supremacy?	149
5.1	Motivation and outline of results	150
5.2	Background	154
5.2.1	Counting complexity and quantum supremacy	154
5.2.2	Degree-3 polynomials and the problem <code>poly3-NONBALANCED</code>	157
5.2.3	The permanent and the problem <code>per-int-NONZERO</code>	159
5.3	Lower Bounds	161
5.3.1	For IQP Circuits	161
5.3.2	For QAOA circuits	162
5.3.3	For boson sampling circuits	165
5.3.4	Evidence for conjectures	166
5.3.5	Number of qubits to achieve quantum supremacy	172

5.4	Reduction from poly3-NONBALANCED to per-int-NONZERO	174
5.5	Better-than-brute-force solution to poly3-NONBALANCED	178
5.6	Concluding remarks	180
6	Computing quopit Clifford circuit amplitudes by the sum-over-paths technique	183
6.1	Motivation and outline of results	184
6.2	Preliminary definitions and notation	186
6.3	Constructing sum-over-paths expressions for quopit Clifford circuits .	188
6.4	Evaluating the sum over paths	190
6.4.1	Step 1: Diagonalizing Θ	190
6.4.2	Step 2: Using the exponential sum formula	191
6.4.3	Running time	193
6.4.4	Proof of Theorem 84	193
6.5	Balancedness of quopit Clifford circuits	197
7	Classical simulation of quantum circuits by half Gauss sums	201
7.1	Outline of results	202
7.2	Half Gauss sums	204
7.2.1	Univariate case	204
7.2.2	Multivariate case	209
7.3	m -qudit Clifford circuits	213
7.4	Hardness results and complexity dichotomy theorems	219
7.4.1	Degree-3 polynomials	219
7.4.2	Without the periodicity condition	220
7.4.3	Other incomplete Gauss sums	222
7.4.4	Complexity dichotomy theorems	223
7.5	Tractable signatures in Holant problems	224
7.6	Appendix for Chapter 7	227
7.6.1	Exponential sum terminology	227
7.6.2	Properties of Gauss sum	228

7.6.3	Half Gauss sum for $\xi_d = -\omega_{2d}$ with even d	229
7.6.4	Relationship between half Gauss sums and zeros of a polynomial	230
8	Quantum simulation from the bottom up: the case of rebits	233
8.1	Introduction	234
8.1.1	Two kinds of simulation	236
8.1.2	Simulation using rebits	238
8.1.3	Our results	239
8.1.4	Related work	245
8.1.5	Notation	246
8.2	Quantum circuits with rebits	247
8.2.1	Rebit encoding and decoding of states	247
8.2.2	Rebit encoding and decoding of operators	249
8.2.3	Rebit encoding and decoding of measurements	257
8.2.4	Bottom-up tomography	260
8.3	Partial antiunitarity	262
8.3.1	Partial complex conjugation	262
8.3.2	Partial antiunitary operators	264
8.4	Simulating partial antiunitary operators	276
8.4.1	Rebit simulation of unitaries: top-down perspective	276
8.4.2	Rebit simulation of non-unitaries: bottom-up perspective	279
8.5	Universal gate sets for \mathbb{R} -unitaries	284
8.6	The \mathbb{R} -Clifford hierarchy	293
8.7	Discussion and open questions	302
8.8	Appendix for Chapter 8	305
8.8.1	A simple motivating example	305
8.8.2	Complex conjugation as a Gottesman-Knill simulation	306
8.8.3	\mathbb{R} -linear operators	308
8.8.4	The ring of \mathbb{R} -linear operators: algebraic properties	309
8.8.5	Equivalent expressions for the rebit encoding of a linear operator	312

8.8.6	Equivalence of norm definitions	313
8.8.7	Alternative formulation of Theorem 126	314
8.8.8	On orthogonal projections	315
8.8.9	Matrix representation of \mathbb{R} -linear operators	316
8.8.10	Proof of Lemma 162	319
8.8.11	Pauli sets with different allowed phases	321
A	Characterizations of the Clifford group	327
A.1	The single-qubit case	335
A.2	The multi-qubit case	340
A.3	Enumerating the Clifford group	349
	References	354

List of Figures

2-1	Example of a quantum circuit, represented as a quantum circuit diagram. The sequence of gates in the circuit may be written as $CY_{13}CX_{12}CS_{32}^\dagger CX_{12}CS_{23}CCZ_{123}H_1H_3$. We write G_i to mean that the gate G acts on the i th qubit.	34
2-2	Hasse diagram representing the poset (χ, \leq) , where χ comprises PH and all its levels. We write $A \leq B$ if the statement that $A \subseteq B$ is known to be true at the time of writing.	53
2-3	Hasse diagram representing the known subset relations (\subseteq) between the various classes introduced in Chapter 2.6.	63
3-1	Circuit diagram for the Clifford computational tasks considered in this chapter. The gates V_i and U_i^\dagger are arbitrary single qubit unitaries, B is a Clifford circuit, the input state is the all-zero computational basis state, and the output measurement is performed in the computational basis.	82
3-2	Relationships between different notions of classical simulation of Clifford computational tasks. An arrow from A to B ($A \rightarrow B$) means that an efficient A -simulation of a computational task implies that there is an efficient B -simulation for the same task. The statement $WK \equiv WK(n)$ is shorthand for $WK \rightarrow WK(n)$ and $WK(n) \rightarrow WK$. . .	85
3-3	Uncomputation trick, in which the output bits, except for those in the target register, are reset to their input values. The state evolves as follows: $(x, 1, \dots, 1, 0) \rightarrow (f(x), j_2, \dots, j_{a(N)}, 0) \rightarrow (f(x), j_2, \dots, j_{a(N)}, f(x)) \rightarrow (x, 1, \dots, 1, f(x))$, where $x = x_1 \dots x_n$	105

4-1 Relationships between different notions of classical simulation and summary of the hardness of simulating CCCs. An arrow from A to B ($A \rightarrow B$) means that an efficient A -simulation of a computational task implies that there is an efficient B -simulation for the same task. Note also that an $\text{weak}(n)$ simulation exists if and only if a weak simulation exists. For a proof of these relationships, see Chapter 3.3. The two curves indicate the boundary between efficiencies of simulation of U -CCCs, where U is not a Clifford operation times a Z rotation. “Hard” means that an efficient simulation of U -CCCs is not possible, unless PH collapses. “Conjectured hard” means that an efficient simulation of U -CCCs is not possible, if we assume Conjecture 69. “Easy” means that an efficient simulation of U -CCCs exists. Note that when U is a Clifford operation times a Z rotation, all the above notions become easy. 135

5-1 IQP circuit \mathcal{C}_f corresponding to the degree-3 polynomial $f(z) = z_1 + z_2 + z_1z_2 + z_1z_2z_3$. The unitary U_f implemented by the circuit has the property that $\langle 0|U_f|0\rangle = \text{gap}(f)/2^n$ where in this case $n = 3$ 158

5-2 Gadget that uses an ancilla qubit to implement the H gate within the QAOA framework. Here the gate Q is the diagonal two-qubit gate $\text{diag}(1, i, 1, -1)$ which can be written as $\exp(-i\frac{\pi}{4}(6|01\rangle\langle 01| + 2|11\rangle\langle 11|))$. Thus, it can be implemented by adding 8 constraints to the constraint Hamiltonian C . The \tilde{H} gate is implemented by applying the Hamiltonian B with $\beta = \pi/4$ 164

5-3 Gadget for each term in the degree-3 polynomial f . Unlabeled edges are assumed to have weight 1. The three dashed lines are connected via the XOR gadget to the dashed lines in the variable gadgets for the variables that appear in the term, as exemplified by the labeling of vertices v and v' in the context of Figure 5-5. If all three variables are true, the term gadget will contribute a cycle cover factor of -1 , excluding the factors of 4 from dotted edges. If at least one variable is false, the term will contribute a cycle cover factor of 1. 176

5-4 Gadget for each variable in the degree-3 polynomial f . The number of dashed lines is equal to the number of terms in which the variable appears, so this example is for a variable that appears in four terms. The dashed lines are connected to the dashed lines in the term gadget in which that variable appears via the XOR gadget, as exemplified by the labeling of vertices u and u' in the context of Figure 5-5. 176

5-5 XOR gadget that connects dotted lines from node u to u' in the variable gadget with dotted lines from node v to v' in the term gadget. The effect of the XOR gadget is that any cycle cover must use either the edge from u to u' or the edge from v to v' , but not both. Each XOR gadget contributes a factor of 4 to the weight of the cycle cover. . . . 177

6-1 Example of a quopit Clifford circuit. As explained in the text, we can assume without loss of generality that each register ends in a Fourier gate. 189

6-2 Labeled circuit corresponding to the circuit in Figure 6-1. The phase polynomial is read off to be $S(x_1, x_2, x_3) = a_2x_1 + a_3x_2 + a_1x_3 + x_3b_1 + b_2(a_1 + x_1) + b_3(a_1 + x_1 + x_2) + 2^{-1}a_1(a_1 - 1)$ 189

8-1 Commutative diagram illustrating the relationships between the sets O , $\mathbb{R}O$ and $O^{\mathbb{R}}$ 240

8-2	Diagram illustrating the relationships between different classes of operators considered in this chapter. A line from \mathcal{A} to \mathcal{B} (where \mathcal{A} is higher than \mathcal{B}) means that \mathcal{A} is a proper superset of \mathcal{B} . The ellipses represent the infinite towers of classes corresponding to different levels of the respective Clifford hierarchies.	241
8-3	The proof of 2-locality of rebit tomography using H , CZ , and single-rebit computational basis measurements. (a) Measuring Z , (b) Measuring X , and (c) Measuring $Y \otimes Y$	261
8-4	Compiling an orthogonal Clifford circuit on n qubits. The top line represents 1 qubit while the bottom represents $n - 1$ qubits.	302
8-5	The circuit, built from only orthogonal gates, implementing a $C^{n-1}V$ gate with two controls (the top two qubits), corresponding to $n = 3$. Further controls are added in the natural way, extending the CZ and CCX gates to include the new qubits as controls too.	321

List of Tables

- 2.1 A complete list of all 24 elements of the single-qubit Clifford group $\mathcal{C}_1/\mathcal{U}(1)$, written as products of H , S , $X = HS^2H$ and $Z = S^2$. The rows are indexed by permutations σ on $\{1, 2, 3\}$ written in cycle notation and the columns are indexed by $\alpha, \gamma \in \{+, -\}$. The integers $ijk \in \{1, 2, 3\}^3$, with distinct i, j, k , are defined by $i = \sigma(1)$, $j = \sigma(2)$ and $k = \sigma(3)$. The Clifford operator U corresponding to $i, j, k \in \{1, 2, 3\}$ and $\alpha, \gamma \in \{+, -\}$ is the unique U satisfying $UXU^\dagger = \alpha\sigma_i$ and $UZU^\dagger = \gamma\sigma_k$. The symbol in parenthesis (β), where $\beta \in \{+, -\}$, written below the Clifford operator U is defined by $UYU^\dagger = \beta\sigma_j$. For example, the entry HS with row index $ijk = 231$, column index $\alpha\gamma = -+$ and $\beta = -$ means that $U = HS$ satisfies $UXU^\dagger = -\sigma_2 = -Y$, $UYU^\dagger = -\sigma_3 = -Z$ and $UZU^\dagger = \sigma_1 = X$ 49

- 2.2 Definitions of BPP, RP, coRP and PP, obtained by replacing the placeholders **A**, **B** and **C** in Template 14 with those in the table. 57

3.1	Classification of the classical simulation complexities of families of Clifford circuits with different ingredients. P stands for efficiently classically simulable. #P stands for #P-hard. QC stands for QC-hard and PH stands for “if efficiently classically simulable, then the polynomial hierarchy collapses”. The proofs of JV 1–7 can be found in [143]. Theorems 39–50 are about cases not found in [143] and are the main results of this chapter. (i)–(xxvii) are results that follow immediately from these theorems by using the rules in Chapter 3.5.1. The 11 cases with boxed symbols are the core theorems, from which all other cases can be deduced using rules which we describe in Chapter 3.5.1. These include all the main theorems JV 1–7 and Theorems 39–50, except JV1 and JV6, which turn out to be special cases of Theorem 49 and Theorem 39 respectively.	87
4.1	Complete complexity classification of U -CCCs (where $U = R_z(\phi)R_x(\theta)$) with respect to weak simulation, as we vary ϕ and θ . The roman numerals in parentheses indicate the parts of Lemma 57 that are relevant to the corresponding box. All U -CCCs are either in PWK (i.e. can be efficiently simulated in the weak sense) or PH-supreme (i.e. cannot be simulated efficiently in the weak sense, unless the polynomial hierarchy collapses.)	118
7.1	Hardness of computing $Z_{1/2^k}(2, f)$, where $k \geq 0$ or $k \geq 1$, and f is a polynomial function with coefficients in \mathbb{Z} and domain \mathbb{Z}_2^n . Here, ‘periodic’ means that f satisfies the periodicity condition (7.3), and ‘aperiodic’ means that f does not necessarily satisfy it. The label FP means that $Z_{1/2^k}(d, f)$ can be computed in classical polynomial time, and #P-hard means that there is no efficient classical algorithm to compute $Z_{1/2^k}(d, f)$, unless the widely-believed conjecture $FP \neq \#P$ is false.	203
8.1	Several examples of our definition of simulation.	237

A.1 Number of elements in the n -qubit Clifford group $|\mathcal{C}_n/\mathcal{U}(1)|$ and the group of n -qubit operators generated by H, S, CZ , for $n = 1, 2, \dots, 6$. The cardinalities of these groups are related by $|\tilde{\mathcal{C}}_n| = |\mathcal{C}_n/\mathcal{U}(1)| = \frac{1}{8}|\langle H, S, CZ \rangle^n|$. The sequence $|\langle H, S, CZ \rangle^n|$ is recorded as sequence A003956 in the On-Line Encyclopedia of Integer Sequences (OEIS) [133]. 353

List of Publications

This thesis is based on the following publications and preprints:

1. Dax Enshan Koh, “Further extensions of Clifford circuits and their classical simulation complexities,” *Quantum Inf. Comput.* **17**, 0262–0282 (2017). arXiv:1512.07892. ECCC TR16-004.
2. Dax Enshan Koh, Mark D. Penney, and Robert W. Spekkens, “Computing quopit Clifford circuit amplitudes by the sum-over-paths technique,” *Quantum Inf. Comput.* **17**, 1081-1095 (2017). arXiv:1702.03316.
3. Dax Enshan Koh, Murphy Yuezhen Niu, and Theodore J. Yoder, “Quantum simulation from the bottom up: the case of rebits,” *J. Phys. A: Math. Theor.* **51**, 195302 (2018). arXiv:1708.09355.
4. Adam Bouland, Joseph F. Fitzsimons, and Dax Enshan Koh, “Complexity Classification of Conjugated Clifford Circuits.” In: 33rd Computational Complexity Conference (CCC 2018), R. A. Servedio, ed., vol. 102 of *Leibniz International Proceedings in Informatics (LIPIcs)*, Dagstuhl, Germany, pp. 21:1–21:25, Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik (2018). arXiv:1709.01805.
5. Alexander M. Dalzell, Aram W. Harrow, Dax Enshan Koh, and Rolando L. La Placa, “How many qubits are needed for quantum computational supremacy?” arXiv:1805.05224 (2018).
6. Kaifeng Bu and Dax Enshan Koh, “Classical simulation of quantum circuits by half Gauss sums.” arXiv:1812.00224 (2018).

Other publications and preprints that I contributed to during my PhD:

1. Zi-Wen Liu, Christopher Perry, Yechao Zhu, Dax Enshan Koh, and Scott Aaronson, “Doubly infinite separation of quantum information and communication,” *Phys. Rev. A* **93**, 012347 (2016). arXiv:1507.03546. ECCC TR16-016.
2. Mark D. Penney, Dax Enshan Koh, and Robert W. Spekkens, “Quantum circuit dynamics via path integrals: Is there a classical action for discrete-time paths?” *New J. Phys.* **19**, 073006 (2017). arXiv:1604.07452.
3. Jacob D. Biamonte, Mauro E. S. Morales, and Dax Enshan Koh, “Quantum Supremacy Lower Bounds by Entanglement Scaling.” arXiv:1808.00460 (2018).
4. Kaifeng Bu and Dax Enshan Koh, “Efficient classical simulation of Clifford circuits with nonstabilizer input states.” arXiv:1902.11257 (2019).

Chapter 1

Introduction

1.1 Motivation

Quantum computers, which were first proposed in the 1980s by Manin [164], Feynman [102], and others, promise to offer significant computational benefits compared to today's classical computers. This promise has been driven by the discovery of quantum algorithms designed to be run on quantum computers, which can potentially solve certain problems much faster than the best classical algorithms possible. A famous example of such a quantum algorithm is due to Shor [205, 207], whose eponymous algorithm can solve the factoring problem exponentially faster than the fastest classical algorithms we know today. Since Shor's discovery, several other quantum algorithms have been discovered [171]. For a comprehensive catalog of quantum algorithms, we refer the reader to the Quantum Algorithm Zoo [141], which at the time of writing contains about 60 quantum algorithms.

While significant progress has been made in the field of quantum algorithms, fast quantum algorithms alone are obviously not sufficient to provide convincing evidence that quantum computers would offer benefits compared to their classical counterparts. Two additional ingredients are needed. The first ingredient is experimental: we need to build quantum computers that are capable of running these quantum algorithms; otherwise, any advantage that these quantum algorithms possess cannot be actualized. The second ingredient is theoretical: we need to prove that the quantum algorithms

we design indeed provide a speedup (or offer some other advantage) over all possible classical algorithms for the same problem. While considerable progress has been made in developing these experimental and theoretical ingredients, the challenges faced remain large and attempts to address them continue to be active areas of research.

On the experimental front, one of the central challenges is that building a quantum computer is difficult. On the one hand, in order to use a quantum system as a quantum computer, we need to isolate it from the outside world. This is because quantum systems that are not perfectly isolated from the environment undergo decoherence and lose information to the environment [20, 140]. On the other hand, we need to be able to control the quantum system from the outside to allow inputs to be fed into the computer, and be able to read out the results of the measurements. This trade-off between the above requirements presents opposing design challenges to building quantum computers [189].

On the theoretical front, one of the main challenges is that proofs of computational hardness are notoriously difficult to establish—consider, for example, the long-standing $P \stackrel{?}{=} NP$ problem [4, 70]. We do not have a theoretical proof yet of any quantum algorithm that can outperform the best classical algorithm for a given task, in the standard paradigm of polynomial-versus-exponential running time, in a computational complexity setting. For example, even though Shor’s algorithm outperforms all known classical factoring algorithms, there is no provable guarantee that future yet-to-be-discovered classical factoring algorithms will not be able to match the performance of Shor’s algorithm. More generally, the question of whether the complexity classes BPP (the class of problems that can be efficiently solved with a classical randomized algorithm) and BQP (the class of problems that can be efficiently solved with a quantum algorithm) are equal remains open. Proving that these two classes are not equal would be a momentous breakthrough, not only because it would prove that quantum computers are exponentially more powerful than classical ones, but also because it would resolve major unsolved problems in classical computational complexity theory; for example, it would prove the long-standing conjecture that $P \neq PSPACE$ [16].

While a complete solution to the above experimental and theoretical challenges has yet to be found, progress has been made towards addressing them. On the experimental front, after years of research and development in fabrication and materials, the first quantum computers that might be able to outperform state-of-the-art classical computers are now in the pipeline. For example, Google announced last year that it had built Bristlecone, a 72-qubit quantum computer based on superconducting circuits [144]. We are now entering a pivotal era in quantum technology, which has been dubbed the Noisy Intermediate-Scale Quantum (NISQ) era [189]. Here, “noisy” emphasizes that experimentalists will likely have only imperfect control over the qubits in the quantum computer, and “intermediate-scale” refers to the size of near-term quantum computers, which are likely to be able to handle only between 50 and a few hundred qubits.

On the theoretical front, even though a proof of an unconditional separation between BPP and BQP remains out of reach, it has been possible to prove quantum-classical separations under additional conditions or assumptions. Popular approaches include proving separations in the black box model, or proving separations between restricted quantum computers and restricted classical computers. An example of a result using the former approach is an oracle separation¹ between BQP and BPP [29]. An example of a result using the latter approach is a separation between constant-depth quantum circuits and constant-depth classical circuits [43]. A third popular approach is to prove separations based on plausible complexity assumptions. Results of this type typically involve choosing a widely-believed conjecture C and a task T for which we can prove the following: (i) T can be performed efficiently on a quantum computer, (ii) If T can also be performed efficiently on a classical computer, then C is false. In other words, the plausibility of C gives evidence that quantum computers outperform classical computers. Examples of such results include theorems showing that quantum computers are able to efficiently perform sampling tasks that classical computers cannot, under the assumption that the polynomial hierarchy does not

¹More recent results have generalized this to an oracle separation between BQP and MA [226], and an oracle separation between BQP and PH [193].

collapse [5, 48, 97].

In light of the above experimental and theoretical advances, a question that has been frequently asked is: when can we expect to observe an empirical demonstration of a quantum advantage that is strongly supported by plausible theoretical assumptions? Performing such a demonstration would be a critical milestone in the development of quantum computers and the first step towards useful quantum computation. It would show what has come to be called ‘quantum supremacy’ [188] (or alternatively, ‘quantum computational supremacy’ [123]), a term which has recently come into vogue to describe such a demonstration of a quantum advantage, whose goal is to overturn the Extended Church-Turing Thesis [184, 238] as confidently as possible. Due to the recent developments of near-term quantum devices, there are expectations that quantum supremacy could be achieved in the next few years.

The potential imminence of quantum supremacy has sparked interest in understanding which quantum computational models are both (i) potentially realizable in the near term and (ii) capable of solving classically-hard problems. Models satisfying (i) are likely to not be capable of arbitrary universal quantum computation, but instead be limited by restrictions of some kind. We shall use the term *restricted models of quantum computation* to refer to quantum computers that have limited access to certain resources. Some examples of restricted quantum computing models include extended Clifford circuits [143, 148], boson sampling circuits [5] and instantaneous quantum polynomial time (IQP) circuits [48]. Because these models are not likely to require the full power of quantum computation, they are potentially easier to implement in the laboratory and are therefore conceivably good candidates for implementation in the near term. The remaining question, then, is whether these various restricted models also satisfy (ii), i.e. can they solve problems which are hard for classical computers? If not, can we show that these models can be efficiently simulated on a classical computer? Studying the *classical simulation complexities* of these restricted models—i.e. how hard it is to classically simulate these models—is one of the central questions we will address in this thesis.

Studying the classical simulation complexities of restricted quantum computing

models is useful not just in helping us to understand which models might be useful candidates for demonstrating quantum supremacy; it also yields useful insights on the relationship between quantum and classical computational power and helps us understand the origin of quantum speedup (if it exists), the reason why quantum supremacy could exist in the first place. To see this, suppose that we start with a restricted model that is efficiently classically simulable. If adding certain ingredients to the restricted model creates a new model that is hard to simulate classically, then we could regard those ingredients as an essential ‘resource’ for quantum computational power. In this thesis, we will see several examples of how adding ingredients to a restricted model of quantum computation changes its classical simulation complexity.

In the preceding paragraphs, we gave an overview of some of the main challenges and advances in the field of quantum computation, and motivated some of the broad questions that this thesis addresses. In the next section, we will elaborate on some of these questions, and summarize the main results and contributions of this thesis.

1.2 Organization and summary of results

This thesis is organized as follows. In Chapter 2, we introduce some of the basic definitions, notation and tools used in quantum computation and computational complexity theory. We describe various restricted models of quantum computation, and summarize some of their properties. An example of a restricted model we will discuss is the class of Clifford circuits, which has important applications in many subfields of quantum computation. An important result about Clifford circuits is the Gottesman-Knill Theorem, which states that Clifford circuits can be efficiently simulated by a classical computer [111].

In Chapter 3, we consider various modifications of Clifford circuits and study how the classical simulation complexities of the circuits change as the ingredients in the circuits are modified. Our results reveal a delicate relationship between the ingredients of the circuits and their classical simulability. In particular, we identify several instances where modest changes to the ingredients of the circuits lead to large

changes in their classical simulation complexities. These results shed some light on the relationship between quantum and classical computational power, and help to identify resources needed for a quantum computational speedup. This chapter is based on [148].

In Chapter 4, we introduce the class of conjugated Clifford circuits, which are a restricted model of quantum computation. These are circuits that can be constructed from Clifford circuits by conjugating each gate in the Clifford circuit by some fixed single-qubit unitary operator. We show that this conjugation breaks the Gottesman-Knill algorithm for simulating Clifford circuits, and that such circuits can give rise to sampling tasks which cannot be efficiently performed to constant multiplicative error on a classical computer, assuming a plausible complexity-theoretic conjecture. Furthermore, by making use of a stronger conjecture, we extend this hardness result to allow for the more realistic model of constant additive error. This work can be seen as progress towards classifying the computational power of all restricted quantum gate sets. This chapter, based on [36], is joint work with Adam Bouland and Joseph F. Fitzsimons.

In Chapter 5, we turn our focus to other restricted models of quantum computation, such as instantaneous quantum polynomial time circuits (IQP) [48], depth-one quantum adiabatic optimization algorithm (QAOA) circuits [97] and boson sampling circuits [5]. These restricted models have been proposed as potential candidates for a quantum supremacy demonstration, but one question that has been debated is how many qubits we need in these circuits before such a demonstration can occur. Previously, all the existing bounds ruling out classical simulation algorithms have been asymptotic in nature, and have been too coarse-grained to provide a number-of-qubits estimate for quantum supremacy. In this chapter, we refine existing hardness arguments and perform a number-of-qubits calculation by imposing a fine-grained complexity conjecture. We conclude that IQP circuits with 180 qubits, QAOA circuits with 360 qubits and boson sampling circuits with 90 photons are large enough for the task of producing samples from their output distributions up to constant multiplicative error to be intractable on current technology. This chapter, based on [76],

is joint work with Alexander M. Dalzell, Aram W. Harrow and Rolando L. La Placa.

In Chapter 6, we consider higher-dimensional analogues of (qubit) Clifford circuits, namely p -level Clifford circuits. Here, we take p to be an odd prime, and defer the case of general p to Chapter 7. Such p -level Clifford circuits are called *quopit Clifford circuits*. We explore the connection between quopit Clifford circuits and Feynman's sum-over-paths technique. In particular, we show that the sum-over-paths technique allows the amplitudes of arbitrary quopit Clifford circuits to be written as a product of Weil sums with quadratic polynomials, which can be computed efficiently. This gives an alternative proof of the Gottesman-Knill Theorem for p -level systems, and is an application of the circuit-polynomial correspondence which relates quantum circuits to low-degree polynomials [172]. This chapter, based on [150], is joint work with Mark D. Penney and Robert W. Spekkens.

In Chapter 7, we generalize the results in Chapter 6 to arbitrary d -level Clifford circuits (called qudit Clifford circuits), where $d \geq 2$ is an arbitrary integer, by relating the amplitudes of these circuits to a class of exponential sums, called periodic, quadratic, multivariate half Gauss sums. Furthermore, we show that these exponential sums become $\#P$ -hard to compute when we omit either the periodic or quadratic condition. This gives a new complexity dichotomy theorem, and highlights the role of periodicity in classical simulation. This chapter, based on [53], is joint work with Kaifeng Bu.

In Chapter 8, we study the task of using a rebit quantum computer (i.e. one that uses only real amplitudes) to simulate a qubit quantum computer (i.e. one that uses complex amplitudes). While it was known since the 1990s that such a simulation can be carried out efficiently [29], an interesting observation that had been noticed previously [170] but had not been explored much is that a rebit computer is able to efficiently simulate not just unitary, but also non-unitary (in fact, nonlinear) operators on the qubit computer. In this chapter, we give the first complete characterization of the qubit operators that can be simulated using this approach, by proving that they belong to a subgroup of the \mathbb{R} -linear operators, called the \mathbb{R} -unitary operators. One important application of our results, which ties to the theme of this thesis, is in

studying which nonlinear operations on quantum circuits can be efficiently simulated on a classical computer. In particular, we define a class of operators, called the \mathbb{R} -Clifford operators, and show that circuits composed of these operators can be efficiently classically simulated. The set of \mathbb{R} -Clifford operators is the union of the Clifford group with some nonlinear operators, and hence, our result enlarges the scope of the Gottesman-Knill Theorem by extending the efficient classical simulation algorithm to allow for the simulation of nonlinear operations as well. This chapter, based on [149], is joint work with Murphy Yuezhen Niu and Theodore J. Yoder.

Chapter 2

Preliminaries

2.1 Notational guide

In this section, we state some of the notational conventions that we will use throughout this thesis. The sets of complex, real and natural numbers are denoted by \mathbb{C} , \mathbb{R} and $\mathbb{N} = \{0, 1, 2, \dots\}$ respectively. The set of integers is denoted by \mathbb{Z} and the set of positive integers is denoted by $\mathbb{Z}^+ = \{1, 2, \dots\}$. For primes p , we write $\mathbb{F}_p = \{0, 1, \dots, p-1\}$ to denote the prime field of order p . For integers n , we write $\mathbb{Z}_n = \{0, 1, \dots, n-1\}$ to denote the ring of integers modulo n . We denote the imaginary unit $i = \sqrt{-1}$ and Euler's number $e = 2.718\dots$ in roman font. The symmetric group of degree n is denoted by \mathcal{S}_n . For groups A and B , we write $A \cong B$ to mean that A is isomorphic to B .

The Hermitian conjugate of a matrix A is written as A^\dagger , its transpose is written as A^T , and its trace is written as $\text{tr}(A)$. The commutator and anticommutator of matrices A and B are written as $[A, B] = [A, B]_- = AB - BA$ and $\{A, B\} = [A, B]_+ = AB + BA$ respectively. The symbol \otimes denotes the Kronecker product. The n -fold Kronecker product $A \otimes \dots \otimes A$ is abbreviated as $A^{\otimes n}$. The Kronecker delta function is denoted by δ_{ij} . We use the symbol I to denote the identity matrix; the size of I should be inferred from context. The set of $d \times d$ matrices is denoted by $\mathcal{L}(d)$. The group of $d \times d$ unitary matrices is denoted by $\mathcal{U}(d)$, and the group of $d \times d$ unitary matrices with determinant 1 is denoted by $\text{SU}(d)$.

The complex conjugation operator is denoted by K . When K acts on vectors or linear operators, we assume that it acts on them with respect to the computational basis. For a scalar, vector or matrix A , we sometimes write $K(A) = \bar{A}$. The real and imaginary parts of a scalar, vector or matrix A are defined in terms of K : the real part of A is given by $\Re A = \frac{1}{2}(A + K(A))$ and the imaginary part of A is given by $\Im A = \frac{1}{2i}(A - K(A))$. Hence we could write $I = \Re + i\Im$ and $K = \Re - i\Im$. We say that A is *real* if $\Re A = A$, and that A is *imaginary* if $\Im A = -iA$.

For operators A, B , we write $A \sim B$ if there exists $\theta \in \mathbb{R}$ such that $A = e^{i\theta}B$. Note that \sim defines an equivalence relation on the set of operators.

2.2 Quantum states, transformations and measurements

We assume some familiarity with quantum computation, but will provide all the necessary definitions and results. We refer the reader to the textbooks [180, 228] for additional background material.

In quantum mechanics, *quantum states* are described by *density operators* (i.e. positive semidefinite operators with unit trace) on a Hilbert space \mathcal{H} , the set of which we denote by $\mathcal{D}(\mathcal{H})$. Throughout this thesis, we will assume that \mathcal{H} is finite-dimensional. When $\mathcal{H} = (\mathbb{C}^d)^{\otimes n} = \mathbb{C}^d \otimes \dots \otimes \mathbb{C}^d$ (where $d \geq 2$ and $n \geq 1$ are integers), we call the quantum state ρ an *n-qudit state*. Here, n is the number of qudits, and d is the number of *levels* of each qudit. Special names are given for qudits with certain values of d . When $d = 2$, qudits are referred to as *qubits*, and when d is an odd prime integer, qudits are referred to as *quopits* [92]. For most of this thesis (except Chapters 6 and 7), we will take $d = 2$.

A quantum state ρ is *pure* if it has rank one. Equivalently, ρ is pure if it is an orthogonal projection onto a one-dimensional subspace, i.e. if ρ can be written as an outer product $|\psi\rangle\langle\psi|$ for some column vector (called a *ket*) $|\psi\rangle \in \mathcal{H}$ satisfying $\|\psi\|^2 := \langle\psi|\psi\rangle = 1$. Notations like “ $|\psi\rangle$ ” and “ $\langle\psi|$ ” are called *Dirac notation* or *bra-*

ket notation, which we will use throughout this thesis. A quantum state is *mixed* if it is not pure.

Quantum transformations are described by *completely positive and trace preserving* (CPTP) maps, also known as *quantum channels*. These are functions mapping density operators to density operators that can be written as $\mathcal{E} : \rho \mapsto \sum_i E_i \rho E_i^\dagger$, where $\sum_i E_i^\dagger E_i = I$. The preceding expression is known as the *Kraus representation* or *operator-sum representation* of the CPTP map [152, 180]. We say that a CPTP map is *unitary* if its Kraus representation has a single nonzero term. Equivalently, a unitary CPTP map is one that can be written as $\mathcal{E} : \rho \mapsto U \rho U^\dagger$ for some unitary operator U .

Quantum measurements are described by sets of *measurement operators* $\{M_m\}$ satisfying the completeness relation $\sum_i M_m^\dagger M_m = I$. Here, the indices m label the outcomes of the measurement. Given a state ρ and measurement operators $\{M_m\}_{m \in \Lambda}$, the probability of obtaining an outcome $m \in \Lambda$ when ρ is measured is determined by *Born's rule*: $\text{pr}(m) = \text{tr}(\rho M_m^\dagger M_m)$. If the outcome m occurs, the state after measurement is

$$\frac{M_m \rho M_m^\dagger}{\text{tr}(\rho M_m^\dagger M_m)}.$$

A quantum measurement $\{M_m\}$ is a *projective measurement* if the measurement operators M_m are Hermitian and $M_l M_m = \delta_{lm} M_m$ for all m and l , where δ_{lm} denotes the *Kronecker delta*. An example of a projective measurement is a *computational basis measurement* on a single qubit, which is given by the measurement operators $\{|0\rangle\langle 0|, |1\rangle\langle 1|\}$. In this case, the probability of obtaining an outcome $k \in \mathbb{F}_2$ when $\rho \in \mathcal{D}(\mathbb{C}^2)$ is measured is $\text{pr}(k) = \langle k | \rho | k \rangle$, and the post-measurement state is $|k\rangle\langle k|$.

While the most general description of a quantum system S undergoing evolution and measurement involves density operators, CPTP maps and general measurements, one can show¹ that S can equivalently be described using only pure states, unitary

¹To see this, we note that every mixed state can be viewed as being the reduced state of some pure state in a larger Hilbert space. Also, every CPTP map on a system A is equivalent to (i) adding an ancilla B in a well-defined state, (ii) performing a unitary on AB , and (iii) tracing out the system B [215]. Similarly, any general measurement on a system A is equivalent to (i) adding an ancilla B in a well-defined state, (ii) performing a unitary on AB , and (iii) performing a projective measurement on AB .

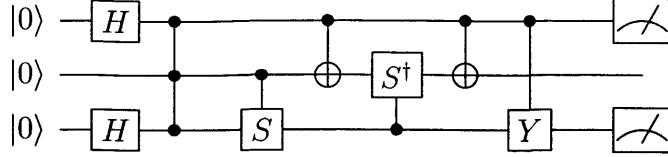


Figure 2-1: Example of a quantum circuit, represented as a quantum circuit diagram. The sequence of gates in the circuit may be written as $CY_{13}CX_{12}CS_{32}^{\dagger}CX_{12}CS_{23}CCZ_{123}H_1H_3$. We write G_i to mean that the gate G acts on the i th qubit.

transformations and projective measurements, if one treats \mathbf{S} as being part of a larger system [180]. This process of going from density operators, CPTP maps and general measurements to pure states, unitary transformations and projective measurements, respectively, is referred to as “going to the *Church of the larger Hilbert Space*” [213]. In this church, the evolution laws become simpler. For example, states can be represented using just kets $|\psi\rangle \in \mathcal{H}$, and unitary transformations can be written as $|\psi\rangle \mapsto U|\psi\rangle$, where U is a unitary operator on \mathcal{H} . Given a state $|\psi\rangle$ and a projective measurement $\{M_m\}$, Born’s rule states that probability of obtaining an outcome m is $\text{pr}(m) = \|M_m|\psi\rangle\|^2$, and the post-measurement state is $M_m|\psi\rangle / \|M_m|\psi\rangle\|$. Since working with the Church of the larger Hilbert Space loses no generality, this is the perspective we will take for most of this thesis.

2.3 Quantum circuits

Quantum circuits are a model of quantum computation represented by a sequence of quantum gates and measurements acting on an n -qudit quantum state. Here, a *quantum gate* is a unitary operation acting on a constant number of qudits (typically, between 1 and 3 qudits). Quantum circuits may be represented graphically as *quantum circuit diagrams*. An example of such a diagram is shown in Figure 2-1. For an introduction to quantum circuits, we refer the reader to Chapter 4 of [180].

In a quantum circuit diagram, a *wire* or a *register* refers to the horizontal line denoting the passage of a qudit in time. The *size* of a circuit refers to the number of

gates in the circuit. The *depth* of a circuit is the number of time steps in a circuit, where each time step contains gates acting on disjoint qudits. For a gate G and an index i , we write G_i to mean that the gate G is applied to the i th qudit (see caption of Figure 2-1).

2.3.1 Examples of quantum gates

In this thesis, we will make repeated use of several common quantum gates. For convenience, we will list some of these gates here. Since most of the thesis deals with n -qubit systems, we will restrict our attention in this chapter to only gates acting on qubits, and defer any treatment of qudit gates, for $d > 2$, to Chapters 6 and 7.

Among the most commonly-used gates are the (single-qubit) *Pauli matrices*, defined as:

$$I = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \quad X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \quad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}. \quad (2.1)$$

These operators are also sometimes denoted as $\sigma_0 = I$, $\sigma_1 = \sigma_x = X$, $\sigma_2 = \sigma_y = Y$ and $\sigma_3 = \sigma_z = Z$. Note that we have included the 2×2 identity operator in the definition of the Pauli matrices. We will study the Pauli matrices and their n -qubit generalizations in more detail in Chapter 2.4.

The Pauli operators may be used to define the so-called operation operators: the *rotation operator* about an axis $t \in \{x, y, z\}$ with angle $\theta \in [0, 2\pi)$ is

$$R_t(\theta) = e^{-i\theta\sigma_t/2} = \cos(\theta/2)I - i\sin(\theta/2)\sigma_t. \quad (2.2)$$

Note that the Pauli gates are a special case of the rotation operators: $\sigma_t = iR_t(\pi)$.

Other commonly-used single-qubit gates are the Hadamard gate H , the phase gate (also called the $\frac{\pi}{4}$ -gate) $S = \sqrt{Z}$ and the $\frac{\pi}{8}$ -gate $T = \sqrt{S}$. The matrix representations

of these gates are

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \quad S = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix}, \quad T = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{pmatrix}. \quad (2.3)$$

It can be shown that every single-qubit gate U can be written in terms of the rotation operators as follows:

$$U = e^{i\alpha} R_z(\phi) R_x(\theta) R_z(\lambda), \quad (2.4)$$

where $\alpha, \phi, \theta, \lambda \in [0, 2\pi)$ [180]. For example,

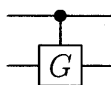
$$H = iR_z(\pi/2)R_x(\pi/2)R_z(\pi/2),$$

$S = e^{i\pi/4}R_z(\pi/2)$, and $T = e^{i\pi/8}R_z(\pi/8)$.

So far, all the gates above are single-qubit gates. One way to construct multiple-qubit gates is to use the *controlled* operation: given a gate G , we define the controlled- G gate to be

$$CG = |0\rangle\langle 0| \otimes I + |1\rangle\langle 1| \otimes G.$$

In a circuit diagram, the CG gate is drawn as

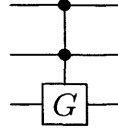


where the top register is called the *control register*, and the bottom one is called the *target register*.

The above process can be applied iteratively. For example, the controlled-controlled G gate is given by

$$CCG = C(CG) = (|00\rangle\langle 00| + |01\rangle\langle 01| + |10\rangle\langle 10|) \otimes I + |11\rangle\langle 11| \otimes G$$

and is drawn as



More generally, we define, for $k \geq 2$, $C^{(k)}G = C(C^{(k-1)}G)$, where $C^{(1)}G = CG$.

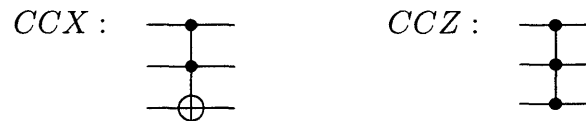
Examples of commonly-used 2-qubit controlled gates include the controlled-not gate CX and the controlled-phase gate CZ . These have matrix representations

$$CX = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}, \quad CZ = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix}.$$

and are drawn as follows:



Examples of commonly-used 2-qubit controlled gates are the Toffoli gate CCX and the controlled-controlled-phase gate CCZ . These are represented by 8×8 matrices and are drawn as follows:



Other common multi-qubit gates are the swap gate

$$\text{SWAP} = \sum_{x,y \in \mathbb{F}_2} |xy\rangle\langle yx|,$$

and the Fredkin gate $C(\text{SWAP})$.

2.3.2 Universality

In this subsection, we will discuss various notions of universality. The strongest sense of universality of a set of quantum gates (called a *gate set*) is the following:

Definition 1 ([12]). A gate set \mathcal{G} is *strictly universal* if there exists a constant $n_0 \in \mathbb{N}$ such that for all integers $n \geq n_0$, the subgroup generated by \mathcal{G} is dense in $SU(2^n)$.

Recall that $SU(2^n)$ is the group of n -qubit unitary matrices with determinant 1 (see Chapter 2.1). Note that Definition 1 does not place any requirements on the efficiency of generating all the unitaries in $SU(2^n)$. To do that, we need the *Solovay-Kitaev Theorem*, which is a general technique for converting statements about density to statements about efficiency:

Theorem 2. (Solovay-Kitaev Theorem [78]) Let \mathcal{G} be a finite set of k -qubit gates that is closed under taking inverses. If \mathcal{G} is strictly universal, then for all $\epsilon > 0$, any k -qubit unitary U can be ϵ -approximated by a finite sequence of gates from \mathcal{G} , where the length of the sequence is $O(\log^{3.97}(1/\epsilon))$. Moreover, the computation of the description of the sequence of gates approximating U can also be performed efficiently. Here, we say that a unitary V ϵ -approximates a unitary U if

$$D(U, V) = \|U - V\| = \sup_{\|\psi\|=1} \|(U - V)|\psi\rangle\| < \epsilon,$$

where $\|\cdot\|$ denotes the *operator norm*.

An implication of the Solovay-Kitaev Theorem is that one can efficiently change between strictly universal gate sets. Two well-known examples of strictly universal gate sets are *Kitaev's gate set* [145] given by $\{CS, H\}$ and the *Clifford+T* gate set given by $\{CZ, H, T\}$. More generally, one gets a universal gate set by appending any non-Clifford gate to the Clifford group (to be defined in section 2.5) [176, 177].

A weaker notion of universality is given by *computational universality*, which is defined as follows.

Definition 3 ([12]). A gate set \mathcal{G} is *computationally universal* if it can be used to simulate to within ϵ error any quantum circuit that uses n qubits and t gates from a strictly universal set, with only poly-logarithmic overhead in $(n, t, 1/\epsilon)$.

While strict universality implies computational universality, the converse does not hold. For example, the gate set $\{\text{Toffoli}, H\}$ is computationally universal [12, 203] but not strictly universal. To see that it is not strictly universal, note that the Toffoli and Hadamard gates are both real matrices, and hence cannot generate a dense subgroup of $SU(2^n)$. To see that it is computationally universal, we use the fact that it can be used to simulate Kitaev's gate set using the rebit encoding (see Chapter 8 and Theorem 2 of [12] for more details).

2.4 Pauli group

2.4.1 Properties of the Pauli group

In this section, we define the Pauli group [180] and discuss its properties. Recall the definition of the single-qubit Pauli matrices in Eq. (2.1).

It is straightforward to verify that the non-identity Pauli matrices $\sigma_1, \sigma_2, \sigma_3$ satisfy the following identities: for $i \in \{1, 2, 3\}$,

$$\sigma_i \sigma_j = \delta_{ij} I + i \epsilon_{ijk} \sigma_k, \quad (2.5)$$

$$\det(\sigma_i) = -1, \quad (2.6)$$

$$\text{tr}(\sigma_i) = 0. \quad (2.7)$$

where ϵ_{ijk} is the Levi-Civita symbol, and δ_{ij} is the Kronecker delta function.

The set of n -qubit Pauli matrices \mathcal{P}_n (for integers $n \geq 1$) is formed by taking n -fold tensor products of the single-qubit Pauli matrices:

$$\mathcal{P}_n = \{P_1 \otimes \dots \otimes P_n \in \mathcal{U}(2^n) : P_i \in \{I, X, Y, Z\} \text{ for } i = 1, \dots, n\}, \quad (2.8)$$

where $\mathcal{U}(k)$ is the group of $k \times k$ unitary matrices.

We now state a few properties that the n -qubit Pauli matrices satisfy. First, each of the Pauli matrices is both Hermitian and unitary, i.e. for all $P \in \mathcal{P}_n$,

$$P^\dagger = P, \quad P^\dagger P = I. \quad (2.9)$$

Second, the Pauli matrices form an orthogonal basis for the complex vector space of $2^n \times 2^n$ matrices (with respect to the Hilbert-Schmidt inner product). More precisely, for all $P, Q \in \mathcal{P}_n$,

$$\text{tr}(P^\dagger Q) = \text{tr}(PQ) = \begin{cases} 2^n, & P = Q \\ 0, & \text{otherwise.} \end{cases} \quad (2.10)$$

Thirdly, any pair of Pauli matrices either commute or anti-commute, i.e. for all $P, Q \in \mathcal{P}_n$, exactly one the following holds:

$$\text{either } [P, Q] = 0 \text{ or } \{P, Q\} = 0. \quad (2.11)$$

In the way it is defined above, the set \mathcal{P}_n is not a group. For example, it is not closed under matrix multiplication: $XY = iZ \notin \mathcal{P}_1$. To make it a group, we shall take the closure of the subset \mathcal{P}_n in the unitary group $\mathcal{U}(2^n)$. To this end, we define the *Pauli group* $\overline{\mathcal{P}}_n$ to be the subgroup of $\mathcal{U}(2^n)$ generated by the set \mathcal{P}_n , i.e.

$$\overline{\mathcal{P}}_n = \langle \mathcal{P}_n \rangle = \left\{ \prod_i U_i : U_i \in \mathcal{P}_n \forall i \right\}. \quad (2.12)$$

By Eq. (2.5), multiplying Pauli matrices with other Pauli matrices only produces multiples of i . Hence, it follows that

$$\overline{\mathcal{P}}_n = \{i^k P : k \in \mathbb{N}, P \in \mathcal{P}_n\}. \quad (2.13)$$

Now, for many applications, the global phase of the Pauli matrices is irrelevant, and hence, the following extension of the Pauli group is useful:

$$\tilde{\mathcal{P}}_n = \{e^{i\theta} P : \theta \in \mathbb{R}, P \in \mathcal{P}_n\}. \quad (2.14)$$

Recall that for matrices U and V , we write $U \sim V$ if there exists $\theta \in \mathbb{R}$ such that $U = e^{i\theta}V$ (see Chapter 2.1). It is easy to see that \sim is an equivalence relation. Let $[U] = \{V : V \sim U\}$ denote the equivalence class of matrices that are equivalent to U . For a set of matrices M , let $M/\mathcal{U}(1) = \{[U] : U \in M\}$ denote the set of equivalence classes formed from the elements of M . When the context is clear, we might occasionally abuse notation and write $[U]$ as U . Using this notation, the Pauli group modulo the unitary group $\mathcal{U}(1)$ may be written as

$$\mathcal{P}_n/\mathcal{U}(1) = \{[P] : P \in \mathcal{P}_n\}, \quad (2.15)$$

where the group multiplication is defined by $[U] \cdot [V] = [UV]$. Note that each element $[P_1 \otimes \dots \otimes P_n]$ is distinct, and hence, the elements of $\mathcal{P}_n/\mathcal{U}(1)$ can be uniquely labeled by the elements in \mathcal{P}_n .

The sets $\overline{\mathcal{P}}_n, \tilde{\mathcal{P}}_n$ and $\mathcal{P}_n/\mathcal{U}(1)$ are all groups, and we will sometimes refer to any of these groups as the Pauli group. While $\tilde{\mathcal{P}}_n$ is an uncountably infinite set, the sets $\mathcal{P}_n, \overline{\mathcal{P}}_n$ and $\mathcal{P}_n/\mathcal{U}(1)$ are all finite, with cardinalities given by

$$|\mathcal{P}_n/\mathcal{U}(1)| = |\mathcal{P}_n| = 4^n, \quad |\overline{\mathcal{P}}_n| = 4^{n+1}. \quad (2.16)$$

2.4.2 \mathbb{F}_2 representation of the Pauli group

In this section, we give an alternative representation of Pauli matrices in terms of vectors in \mathbb{F}_2^{2n} . Consider the map $\sigma : \mathbb{F}_2^{2n} \rightarrow \mathcal{P}_n/\mathcal{U}(1)$, defined by

$$\sigma : u = \begin{pmatrix} u_1 \\ u_2 \end{pmatrix} \mapsto \sigma^u = X^{u_1} Z^{u_2}, \quad (2.17)$$

where $u_j = (u_{j1}, \dots, u_{jn})^T$ for $j = 1, 2$, and $X^{u_1} Z^{u_2} = X^{u_{11}} Z^{u_{21}} \otimes \dots \otimes X^{u_{1n}} Z^{u_{2n}}$. For convenience, we have omitted the brackets when denoting the equivalence class $[\sigma^u]$ and have simply written it as σ^u . We will use the convention of denoting the elements of \mathbb{F}_2^{2n} by column vectors of length $2n$.

Note that σ is bijective, and so its inverse $\phi : \sigma^u \mapsto u$ exists. We shall call $u = \phi(\sigma^u)$ the \mathbb{F}_2 representation of σ^u . Note also that σ is a group isomorphism between the groups

$$(\mathbb{F}_2^{2n}, +) \cong (\mathcal{P}_n/U(1), \cdot). \quad (2.18)$$

We now show how commutation and anticommutation of Pauli matrices manifest themselves in the \mathbb{F}_2 representation. Let

$$\Lambda = \begin{pmatrix} 0_n & I_n \\ I_n & 0_n \end{pmatrix}, \quad (2.19)$$

where 0_n is the $n \times n$ zero matrix and I_n is the $n \times n$ identity matrix. This allows us to define the *symplectic inner product* $\langle \cdot, \cdot \rangle : \mathbb{F}_2^{2n} \times \mathbb{F}_2^{2n} \rightarrow \mathbb{F}_2$ as

$$\langle u, v \rangle = u^T \Lambda v. \quad (2.20)$$

Next, we recall the definition of a symplectic bilinear form:

Definition 4. A *symplectic bilinear form* on a vector space V over a field F is a function $w : V \times V \rightarrow F$ such that

1. (bilinearity) For all $a, b \in F$ and $x, y, z \in V$,

$$w(ax + by, z) = aw(x, z) + bw(y, z)$$

and

$$w(x, ay + bz) = aw(x, y) + bw(x, z).$$

2. (totally isotropic) $w(x, x) = 0$ for all $x \in V$.
3. (nondegenerate) If $w(x, y) = 0$ for all $y \in V$, then $x = 0$.

The pair (V, w) is called a *symplectic vector space*.

It is straightforward to check that $\langle \cdot, \cdot \rangle$ is a symplectic bilinear form on the vector

space \mathbb{F}_2^{2n} over the field \mathbb{F}_2 , i.e., $(\mathbb{F}_2^{2n}, \langle \cdot, \cdot \rangle)$ is a symplectic vector space. Since the underlying field is \mathbb{F}_2 , the symplectic bilinear form $\langle \cdot, \cdot \rangle$ is also symmetric².

Theorem 5. Let $u, v \in \mathbb{F}_2^{2n}$. Then,

$$[\sigma^u, \sigma^v] = 0 \iff \langle u, v \rangle = 0. \quad (2.21)$$

Proof. Consider

$$\sigma^u \sigma^v = X^{u_1} Z^{u_2} X^{v_1} Z^{v_2} \quad (2.22)$$

$$= (-1)^{v_1 \cdot u_2} (-1)^{u_1 \cdot v_2} X^{v_1} Z^{v_2} X^{u_1} Z^{u_2} \quad (2.23)$$

$$= (-1)^{\langle u, v \rangle} \sigma^v \sigma^u. \quad (2.24)$$

Hence,

$$[\sigma^u, \sigma^v] = 0 \iff (-1)^{\langle u, v \rangle} = 1 \iff \langle u, v \rangle = 0. \quad (2.25)$$

□

Remark. By Eq. (2.11), Pauli matrices either commute or anticommute. Since $\langle u, v \rangle \in \{0, 1\}$, the biconditional (2.21) is equivalent to

$$\{\sigma^u, \sigma^v\} = 0 \iff \langle u, v \rangle = 1. \quad (2.26)$$

Hence, two Pauli matrices commute (anticommute) if and only if the inner product of their \mathbb{F}_2 representations vanishes (does not vanish).

2.5 Clifford group

In this section, we introduce the Clifford group, which plays a prominent role in various subfields of quantum computation, such as quantum error correction and fault tol-

²Here, we use the fact that any symplectic bilinear form over a field of characteristic 2 is symmetric.

erance [45, 112], randomized benchmarking [163], measurement-based quantum computation [51, 191, 192] and classical simulation of quantum computation [40, 42, 81]. The Clifford group also plays an important role in other fields, such as quantum foundations [15, 186] and in the construction of optimal packings in Grassmannian spaces [62, 69, 208]. The term ‘Clifford group’, as used in the quantum computation literature, dates back to the pioneering works on quantum error correction by Calderbank, Rains, Shor and Sloane [60, 61, 206]. In [61, 176], the authors trace the usage of the term to the works of Bolt, Room and Wall [33, 34]. Note that the same name has been used for a different family of groups [66].

The n -qubit *Clifford group* (for integers $n \geq 1$) is defined as

$$\mathcal{C}_n = \left\{ U \in \mathcal{U}(2^n) : U\tilde{\mathcal{P}}_n U^\dagger = \tilde{\mathcal{P}}_n \right\}. \quad (2.27)$$

In other words, the Clifford group \mathcal{C}_n is the normalizer of the Pauli group $\tilde{\mathcal{P}}_n$ in the unitary group, and hence is a group. In particular, it is closed under multiplication and inverses. The elements of the Clifford group are called *Clifford operators*.

Let

$$\mathcal{P}_n^* = \mathcal{P}_n \setminus \{I\}, \quad (2.28)$$

where we have excluded from \mathcal{P}_n the only element without the negative eigenvalue -1 : unlike \mathcal{P}_n , every element of \mathcal{P}_n^* has eigenvalues ± 1 , each with multiplicity 2^{n-1} . Then, the Clifford group can be expressed in terms of the different variants of the Pauli group: $\overline{\mathcal{P}}_n, \mathcal{P}_n/\mathcal{U}(1), \mathcal{P}_n$ and \mathcal{P}_n^* , as the following proposition shows.

Proposition 6.

$$\mathcal{C}_n = \{U \in \mathcal{U}(2^n) : U\overline{\mathcal{P}}_n U^\dagger = \overline{\mathcal{P}}_n\} \quad (2.29)$$

$$= \{U \in \mathcal{U}(2^n) : [U] \cdot \mathcal{P}_n/\mathcal{U}(1) \cdot [U^\dagger] = \mathcal{P}_n/\mathcal{U}(1)\} \quad (2.30)$$

$$= \{U \in \mathcal{U}(2^n) : U\mathcal{P}_n U^\dagger \subseteq \pm\mathcal{P}_n\} \quad (2.31)$$

$$= \{U \in \mathcal{U}(2^n) : U\mathcal{P}_n^* U^\dagger \subseteq \pm\mathcal{P}_n^*\}. \quad (2.32)$$

Proof. We shall prove that Eq. (2.27) \subseteq Eq. (2.32). We leave the other cases as an exercise for the reader. Let U satisfy Eq. (2.27), and let $P \in \mathcal{P}_n^*$. Then $UPU^\dagger \in \tilde{\mathcal{P}}_n$, i.e.

$$UPU^\dagger = e^{i\theta}Q, \quad (2.33)$$

for some $\theta \in \mathbb{R}$ and $Q \in \mathcal{P}_n$. Since UPU^\dagger is Hermitian, $e^{i\theta}Q$ is Hermitian, i.e.

$$e^{i\theta}Q = e^{-i\theta}Q^\dagger. \quad (2.34)$$

Since Q is Hermitian, $e^{2i\theta} = 1$, which implies that

$$e^{i\theta} = \pm 1. \quad (2.35)$$

Now, suppose, for the sake of contradiction, that $Q = I$, then $UPU^\dagger = \pm I$. This implies that $P = \pm I$, which is a contradiction, since $\pm I \notin \mathcal{P}_n^*$. Hence, $Q \neq I$ which implies that $UPU^\dagger \in \pm\mathcal{P}_n^*$. \square

Now, each element in \mathcal{P}_n^* can be written as $\sigma^u = X^{u_1}Z^{u_2}$, for some $u \in \mathbb{F}_2^{2n} \setminus \{0\}$. Hence, it follows that

$$\begin{aligned} \mathcal{C}_n &= \{U \in \mathcal{U}(2^n) : \forall u \in \mathbb{F}_2^{2n} \setminus \{0\}, \exists a \in \mathbb{F}_2, \exists v \in \mathbb{F}_2^{2n} \setminus \{0\} \\ &\text{s.t. } U\sigma^uU^\dagger = (-1)^a\sigma^v\}. \end{aligned} \quad (2.36)$$

2.5.1 Elements of the Clifford group

We start by listing a few examples of Clifford operators. First, note that since the Pauli group is closed under multiplication, the Pauli group is a subset of the Clifford group:

$$\mathcal{P}_n \subseteq \mathcal{C}_n. \quad (2.37)$$

Indeed, Eq. (2.24) implies that

$$\sigma^u : \sigma^v \rightarrow (-1)^{\langle u, v \rangle} \sigma^v, \quad (2.38)$$

where we have used the following notation:

$$U : P \rightarrow Q \quad \text{means} \quad UPU^\dagger = Q. \quad (2.39)$$

For example, when $n = 1$, we get

$$\begin{aligned} X : X &\rightarrow X \\ Y &\rightarrow -Y \\ Z &\rightarrow -Z, \end{aligned} \quad (2.40)$$

$$\begin{aligned} Y : X &\rightarrow -X \\ Y &\rightarrow Y \\ Z &\rightarrow -Z, \end{aligned} \quad (2.41)$$

$$\begin{aligned} Z : X &\rightarrow -X \\ Y &\rightarrow -Y \\ Z &\rightarrow Z. \end{aligned} \quad (2.42)$$

But the Pauli group is not a strict subgroup of the Clifford group. It turns out that many of the operations we defined in Chapter 2.3 are also Clifford operators. For example, the following gates are Clifford operations, as can be seen from their actions on the Pauli group.

$$\begin{aligned} H : X &\rightarrow Z \\ Y &\rightarrow -Y \\ Z &\rightarrow X, \end{aligned} \quad (2.43)$$

$$\begin{aligned} S : X &\rightarrow Y \\ Y &\rightarrow -X \\ Z &\rightarrow Z, \end{aligned} \quad (2.44)$$

$$\begin{aligned}
CX : X \otimes I &\rightarrow X \otimes X \\
Y \otimes I &\rightarrow Y \otimes X \\
Z \otimes I &\rightarrow Z \otimes I \\
I \otimes X &\rightarrow I \otimes X \\
I \otimes Y &\rightarrow Z \otimes Y \\
I \otimes Z &\rightarrow Z \otimes Z,
\end{aligned} \tag{2.45}$$

$$\begin{aligned}
CY : X \otimes I &\rightarrow X \otimes Y \\
Y \otimes I &\rightarrow Y \otimes Y \\
Z \otimes I &\rightarrow Z \otimes I \\
I \otimes X &\rightarrow Z \otimes X \\
I \otimes Y &\rightarrow I \otimes Y \\
I \otimes Z &\rightarrow Z \otimes Z,
\end{aligned} \tag{2.46}$$

$$\begin{aligned}
CZ : X \otimes I &\rightarrow X \otimes Z \\
Y \otimes I &\rightarrow Y \otimes Z \\
Z \otimes I &\rightarrow Z \otimes I \\
I \otimes X &\rightarrow Z \otimes X \\
I \otimes Y &\rightarrow Z \otimes Y \\
I \otimes Z &\rightarrow I \otimes Z,
\end{aligned} \tag{2.47}$$

$$\begin{aligned}
\text{SWAP} : X \otimes I &\rightarrow I \otimes X \\
Y \otimes I &\rightarrow Y \otimes Y \\
Z \otimes I &\rightarrow Z \otimes I \\
I \otimes X &\rightarrow X \otimes I \\
I \otimes Y &\rightarrow Y \otimes I \\
I \otimes Z &\rightarrow Z \otimes I.
\end{aligned} \tag{2.48}$$

Observe that all the Clifford gates stated above can be expressed as products of CZ , H and S gates, as can be seen from the following identities:

$$\begin{aligned}
X &= HS^2H \\
Y &= iHS^2HS^2 \\
Z &= S^2 \\
CX_{12} &= H_2CZ_{12}H_2 \\
CY_{12} &= S_1H_2CZ_{12}H_2CZ_{12} \\
\text{SWAP}_{12} &= H_2CZ_{12}H_1H_2CZ_{12}H_1H_2CZ_{12}H_2.
\end{aligned} \tag{2.49}$$

Hence, $X, Y, Z, H, S, CX, CY, CZ, \text{SWAP} \in \langle H, S, CZ \rangle^n / \mathcal{U}(1)$, where we have used the following notation: if $\{g^{(1)}, \dots, g^{(s)}\}$ is a set of gates, we define $\langle g^{(1)}, \dots, g^{(s)} \rangle^n$ to be the set of n -qubit operations that are generated by circuits with gates $g^{(1)}, \dots, g^{(s)}$.

It turns out that we can prove a much stronger result than that stated above. The ability to express operators as products of H , S and CZ gates applies not just to the above gates, but to every Clifford gate, as the following theorem states.

Theorem 7. (Theorem 214)

$$\mathcal{C}_n / \mathcal{U}(1) = \begin{cases} \langle H, S \rangle / \mathcal{U}(1) & n = 1 \\ \langle H, S, CZ \rangle^n / \mathcal{U}(1) & n \geq 1. \end{cases}$$

We provide a complete proof of Theorem 7 in Appendix A (see Theorem 214), where we also explore various characterizations of the Clifford group. A similar proof of Theorem 7 may also be found in [109] (see Chapters 5.6 and 5.8) and [110] (see page 3).

From Theorem 7, one can show that the Clifford group is finite. In fact, its cardinality is given by the following theorem.

Theorem 8. (Theorem 218) Let $n \geq 1$. The number of elements in the n -qubit

σ	ijk	$\alpha\gamma$			
		++	+-	-+	--
I	123	I (+)	X (-)	Z (-)	XZ (+)
(23)	132	SHS (-)	$SHSX$ (+)	$SHSZ$ (+)	$SHSXZ$ (-)
(12)	213	S (-)	SX (+)	SZ (+)	SXZ (-)
(123)	231	HSZ (+)	$HSZX$ (-)	HS (-)	HSX (+)
(132)	312	SH (+)	SHX (-)	SHZ (-)	$SHXZ$ (+)
(13)	321	H (-)	HX (+)	HZ (+)	HXZ (-)

Table 2.1: A complete list of all 24 elements of the single-qubit Clifford group $\mathcal{C}_1/\mathcal{U}(1)$, written as products of H , S , $X = HS^2H$ and $Z = S^2$. The rows are indexed by permutations σ on $\{1, 2, 3\}$ written in cycle notation and the columns are indexed by $\alpha, \gamma \in \{+, -\}$. The integers $ijk \in \{1, 2, 3\}^3$, with distinct i, j, k , are defined by $i = \sigma(1)$, $j = \sigma(2)$ and $k = \sigma(3)$. The Clifford operator U corresponding to $i, j, k \in \{1, 2, 3\}$ and $\alpha, \gamma \in \{+, -\}$ is the unique U satisfying $UXU^\dagger = \alpha\sigma_i$ and $UZU^\dagger = \gamma\sigma_k$. The symbol in parenthesis (β), where $\beta \in \{+, -\}$, written below the Clifford operator U is defined by $UYU^\dagger = \beta\sigma_j$. For example, the entry HS with row index $ijk = 231$, column index $\alpha\gamma = -+$ and $\beta = -$ means that $U = HS$ satisfies $UXU^\dagger = -\sigma_2 = -Y$, $UYU^\dagger = -\sigma_3 = -Z$ and $UZU^\dagger = \sigma_1 = X$.

Clifford group is

$$|\mathcal{C}_n/\mathcal{U}(1)| = 2^{n(n+2)} \prod_{j=1}^n (4^j - 1). \quad (2.50)$$

For example, when $n = 1$, the Clifford group has 24 elements. A complete list of all 24 elements is given in Table 2.1.

2.6 Classical and quantum computational complexity theory

In this section, we will provide a summary of the necessary background in computational complexity theory that is relevant to this thesis. For additional background material, we refer the reader to [16, 212, 227].

An *alphabet* Σ is a nonempty finite set. The elements of Σ are called *symbols*. Throughout this section and for most of this thesis, we will take Σ to be the *binary alphabet* $\mathbb{F}_2 = \{0, 1\}$. A *string* w over an alphabet Σ is a finite sequence of symbols from Σ . The number of symbols in w is called its *length* and is denoted by $|w|$. The set of strings of length n over Σ is denoted by Σ^n . The set of all strings over Σ is denoted by $\Sigma^* = \bigcup_{i \in \mathbb{N}} \Sigma^i$, where $*$ is called the *Kleene star*.

A *language* L is a set of strings, i.e. $L \subseteq \Sigma^*$. We denote the indicator function of L by $L(x)$, i.e. $L(x) = 1$ if $x \in L$ and $L(x) = 0$ otherwise. The set $\mathbf{ALL} = \mathcal{P}(\{0, 1\}^*)$ is the set of all languages, where $\mathcal{P}(\cdot)$ is the power set operation. We will be studying subsets of \mathbf{ALL} that arise from placing restrictions on the resources available to various models of computation. Such subsets of \mathbf{ALL} are called *complexity classes*. A comprehensive online catalog of complexity classes may be found at the Complexity Zoo [9], which at the time of writing contains more than 530 classes. An inclusion diagram summarizing all the known relations between many of these classes may be found at the website [153]. We present a smaller variant of this diagram in Figure 2-3, which summarizes all the known relations between the classes involved in this section. Our diagram is not a strict sub-diagram of [153]; for example, unlike [153], we include the class $\text{co}A$ for each class A that is not known to be symmetric.

A common model of computation is the *Turing machine*, which is an abstract machine introduced by Turing [221] that manipulates symbols on an infinite strip of tape according to a finite set of rules. For a precise definition of a Turing machine, see [212]. We write the output of a Turing machine M on an input string x as $M(x)$. A language is *decidable* if there exists a Turing machine M such that for all $x \in \{0, 1\}^*$, $M(x) = L(x)$. When this holds, we say that M *decides* L . Let \mathbf{R} be the

set of decidable languages. Not all languages are decidable (for example, the *halting problem* is not decidable [221]); hence, $\mathbf{R} \subsetneq \mathbf{ALL}$.

2.6.1 P, NP and the polynomial hierarchy

Let $t : \mathbb{N} \rightarrow \mathbb{N}$. A language $L \in \mathbf{TIME}(t(n))$ if there exist $c > 0$ and a Turing machine M that decides L in $ct(n)$ time. One of the most important classes in complexity theory is the class \mathbf{P} (which stands for *polynomial-time*), which is defined as

$$\mathbf{P} = \bigcup_{c \geq 1} \mathbf{TIME}(n^c). \quad (2.51)$$

Informally speaking, \mathbf{P} represents the class of efficiently computable languages. Throughout this thesis, the term *efficient* is used as shorthand for ‘polynomial time’. Not all decidable languages are in \mathbf{P} . For example, by the *time-hierarchy theorem* [124], the class of languages that are decidable in exponential time

$$\mathbf{EXP} := \bigcup_{c \geq 0} \mathbf{TIME}(2^{n^c}) \quad (2.52)$$

is decidable and strictly larger than \mathbf{P} , i.e. $\mathbf{P} \subsetneq \mathbf{EXP}$.

A language L is in \mathbf{NP} (which stands for *nondeterministic polynomial time*) if there exist a polynomial-time Turing machine M and a polynomial q such that for all $x \in \{0, 1\}^*$,

$$x \in L \iff \exists y \in \{0, 1\}^{p(|x|)} M(x, y) = 1. \quad (2.53)$$

Informally, \mathbf{NP} is the set of languages for which the answer can be verified in polynomial time. It follows from the definitions that $\mathbf{P} \subseteq \mathbf{NP}$. One of the most famous open problems in theoretical computer science is whether this inclusion is strict, i.e. is $\mathbf{P} \subsetneq \mathbf{NP}$? While this problem, called the $\mathbf{P} \stackrel{?}{=} \mathbf{NP}$ problem [4], remains open at the time of writing, it is widely believed that

Conjecture 9.

$$P \neq NP.$$

A *Cook reduction* or a *polynomial-time Turing reduction* from a language A to a language B is a polynomial-time algorithm that solves A using a polynomial number of calls to a subroutine for B . We say that a language L is **NP-hard** if for $A \in \text{NP}$, A reduces to L . We say that L is **NP-complete** if it is in **NP** and it is **NP-hard**. A famous example of an **NP-complete** problem is the satisfiability problem, called **SAT**, which asks if a Boolean formula is satisfiable [71, 160].

For a complexity class C , define

$$\text{co}C = \{L : \{0, 1\}^* \setminus L \in C\}. \quad (2.54)$$

A class C is called *symmetric* if $C = \text{co}C$. From Eq. (2.54), it follows that **coNP** is the set of languages L for which there exist a polynomial-time Turing machine M and a polynomial q such that for all $x \in \{0, 1\}^*$,

$$x \in L \iff \forall y \in \{0, 1\}^{p(|x|)} M(x, y) = 1. \quad (2.55)$$

Let $k \geq 1$ be an integer. A language $L \in \Sigma_k^p$ if there exist a polynomial-time Turing machine M and a polynomial q such that for all $x \in \{0, 1\}^*$,

$$\begin{aligned} x \in L &\iff \exists u_1 \in \{0, 1\}^{q(|x|)} \forall u_2 \in \{0, 1\}^{q(|x|)} \dots Q_k u_k \in \{0, 1\}^{q(|x|)} \\ &\text{s.t. } M(x, u_1, \dots, u_k) = 1. \end{aligned}$$

where $Q_k = \forall$ if k is even and $Q_k = \exists$ if k is odd. Define $\Pi_k^p = \text{co}\Sigma_k^p$, and $\Sigma_0^p = \Pi_0^p = P$. It is easy to see that $\Sigma_1^p = \text{NP}$ and $\Pi_1^p = \text{coNP}$. The sets Σ_k^p and Π_k^p are said to belong to the *kth level of the polynomial hierarchy*. The union of these levels is the *polynomial hierarchy*:

$$\text{PH} = \bigcup_{k \in \mathbb{N}} (\Sigma_k^p \cup \Pi_k^p) = \bigcup_{k \in \mathbb{N}} \Sigma_k^p = \bigcup_{k \in \mathbb{N}} \Pi_k^p.$$

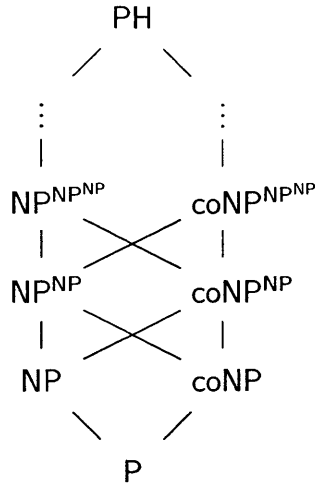


Figure 2-2: Hasse diagram representing the poset (χ, \leq) , where χ comprises PH and all its levels. We write $A \leq B$ if the statement that $A \subseteq B$ is known to be true at the time of writing.

The levels of the polynomial hierarchy can also be written in terms of *oracle complexity classes*. Let A be a complexity class which is defined in terms of a Turing machine M , and let L be a language. Define A^L to be the complexity class that results from replacing each instance of M in the definition of A by M^L , where M^L is the Turing machine M that has access to an oracle for the language L . This superscript notation can be extended to a set B of languages by

$$A^B = \bigcup_{L \in B} A^L.$$

It can be shown that $\Sigma_2^p = \text{NP}^{\text{NP}}$, $\Sigma_3^p = \text{NP}^{\text{NP}^{\text{NP}}}$, and so on. Similarly, $\Pi_2^p = \text{coNP}^{\text{NP}}$, $\Pi_3^p = \text{coNP}^{\text{NP}^{\text{NP}}}$, and so on (see Chapter 5.5 of [16]). A Hasse diagram showing all the known subset relations between the polynomial hierarchy and its levels is shown in Figure 2-2. It remains an open question whether any of the inclusions indicated in Figure 2-2 is strict, or whether any of the classes in the figure are equal to each other. We do know, however, that if any two classes in the figure are equal, then the polynomial hierarchy *collapses* to the minimum of the levels that the classes belong to, as the following proposition states.

Proposition 10. Let $k, l \in \mathbb{N}$ and $\Omega, \Theta \in \{\Pi, \Sigma\}$. If $(k, l) \neq (0, 0)$ and $(\Omega, k) \neq (\Theta, l)$, then

$$\Omega_k^p = \Theta_l^p \implies \text{PH} = \Sigma_{\min(k,l)}^p = \Pi_{\min(k,l)}^p.$$

In particular,

1. $\Sigma_k^p = \Sigma_{k+1}^p \implies \text{PH} = \Sigma_k^p$.
2. If $k \neq 0$, then $(\Sigma_k^p = \Pi_k^p \implies \text{PH} = \Sigma_k^p)$.

Here, we say that the “polynomial hierarchy collapses to the k th level” if $\text{PH} = \Sigma_k^p$, i.e. if $\Sigma_l^p = \Sigma_k^p$ for all $l \geq k$. We say that the polynomial hierarchy is *infinite* or that it *does not collapse* if $\text{PH} \neq \Sigma_k^p$ for all k , i.e. if each level of the hierarchy is distinct (by Proposition 10). Though not proven to be true, it is widely believed that

Conjecture 11. The polynomial hierarchy does not collapse.

Note that Conjecture 11 is a stronger conjecture than Conjecture 9 that $\text{P} \neq \text{NP}$, because if $\text{P} = \text{NP}$, then the polynomial hierarchy collapses to the zeroth level, by Proposition 10.

2.6.2 Complexity of counting

The class FP , which is the function problem analogue of the class P , is the set of functions $f : \{0, 1\}^* \rightarrow \{0, 1\}^*$ for which there exists a polynomial-time Turing machine M such that for all x , $M(x) = f(x)$. The class $\#\text{P}$, introduced by Valiant [222], is the set of functions $f : \{0, 1\}^* \rightarrow \mathbb{N}$ for which there exist a polynomial $p : \mathbb{N} \rightarrow \mathbb{N}$ and a polynomial-time Turing machine M such that for all $x \in \{0, 1\}^*$,

$$f(x) = \left| \left\{ y \in \{0, 1\}^{p(|x|)} : M(x, y) = 1 \right\} \right|.$$

If Conjecture 9 is true, then it follows that $\text{FP} \neq \#\text{P}$. We could extend the notion of completeness to $\#\text{P}$ problems. Let $f, g : \{0, 1\}^* \rightarrow \{0, 1\}^*$. We say that g reduces

to f if $g \in \text{FP}^f$. A function $f : \{0, 1\}^* \rightarrow \{0, 1\}^*$ is $\#\text{P}$ -hard if for all $q \in \#\text{P}$, q reduces to f . f is $\#\text{P}$ -complete if it is both in $\#\text{P}$ and is $\#\text{P}$ -hard. An example of a $\#\text{P}$ -complete problem is $\#\text{SAT}$, which is the problem of counting the number of satisfying assignments of a Boolean formula. An important result relating the class $\#\text{P}$ and the polynomial hierarchy is

Theorem 12. (Toda's Theorem [219])

$$\text{PH} \subseteq \text{P}^{\#\text{P}}.$$

In the above theorem, note that $\text{P}^{\#\text{P}}$ is the set of languages that can be efficiently decided by a Turing machine with access to a $\#\text{P}$ oracle.

2.6.3 Space complexity

Next, we turn our attention to space complexity. Let $t : \mathbb{N} \rightarrow \mathbb{N}$. Let $\text{SPACE}(t(n))$ be the set of languages L for which there exists a Turing machine machine M such that for $x \in \{0, 1\}^*$,

1. $x \in L \iff M(x) = 1$.
2. M halts on x .
3. M on input x uses at most $t(|x|)$ tape cells.

The classes PSPACE and EXPSPACE are defined analogously to the time complexity classes P and EXP :

$$\begin{aligned} \text{PSPACE} &= \bigcup_{c \geq 0} \text{SPACE}(n^c), \\ \text{EXPSPACE} &= \bigcup_{c \geq 0} \text{SPACE}(2^{n^c}). \end{aligned}$$

The classes PSPACE , EXPSPACE and some of the previously-mentioned classes

are related as follows:

$$P \subseteq NP \subseteq PH \subseteq P^{\#P} \subseteq PSPACE \subseteq EXP \subseteq EXPSPACE. \quad (2.56)$$

While it is known that $P \neq EXP$ and $PSPACE \neq EXPSPACE$, the question about whether any of the other inclusions in Eq. (2.56) are strict remains open. It follows from Eq. (2.56) that if Conjecture 9 holds, then the following weaker conjecture would also hold.

Conjecture 13.

$$P \neq PSPACE.$$

2.6.4 Classical randomized complexity

In this subsection, we will study complexity classes that are defined by Turing machines that have access to classical randomness. We'll be defining several classes using the following template.

Template 14. A language L is in $\boxed{\mathbf{A}}$ if there exist a polynomial-time Turing machine M and a polynomial $p : \mathbb{N} \rightarrow \mathbb{N}$ such that for all $x \in \{0, 1\}^*$,

$$\text{(Completeness)} \quad x \in L \implies \Pr_{r \in \{0,1\}^{p(|x|)}} [M(x, r) = 1] \boxed{\mathbf{B}},$$

$$\text{(Soundness)} \quad x \notin L \implies \Pr_{r \in \{0,1\}^{p(|x|)}} [M(x, r) = 1] \boxed{\mathbf{C}}.$$

The classes BPP (*bounded-error probabilistic polynomial time*), RP (*randomized polynomial time*), coRP and PP (*probabilistic polynomial time*) are defined by replacing the placeholders $\boxed{\mathbf{A}}$, $\boxed{\mathbf{B}}$ and $\boxed{\mathbf{C}}$ in Template 14 with those in the Table 2.2.

Note that the constants $2/3$ and $1/3$ in the definition of BQP are arbitrary. By the amplification lemma, the class BPP would remain the same if we set $\boxed{\mathbf{B}}$ to be $\geq p(n)$ and $\boxed{\mathbf{C}}$ to be $\leq q(n)$, for any $p(n) \leq 1 - 2^{-n^d}$ and $q(n) \geq 2^{-n^d}$ such that $p(n) - q(n) \geq n^{-c}$, where c and d are constants, and $n = |x|$ is the length of the

	A	B	C
BPP	$\geq 2/3$	$\leq 1/3$	
RP	$\geq 2/3$	$= 0$	
coRP	$= 1$	$\leq 1/3$	
PP	$> 1/2$	$\leq 1/2$	

Table 2.2: Definitions of BPP, RP, coRP and PP, obtained by replacing the placeholders **A**, **B** and **C** in Template 14 with those in the table.

input. Similarly, the class RP would remain unchanged if we set **B** to be $\geq p(n)$, for any $n^{-c} \leq p(n) \leq 1 - 2^{-n^d}$, where c and d are constants. Also, the class coRP would remain unchanged if we set **C** to be $\leq q(n)$, for any $2^{-n^d} \leq q(n) \leq 1 - 2^{-n^c}$, where c and d are constants.

The above bounds suggest that problems in BPP, RP and coRP are tractable; it is possible to achieve exponentially small errors with just polynomial running time. For problems in the class PP, however, it is not known how to achieve an exponentially small error with only polynomial running time, and hence, problems in PP are generally not regarded to be tractable problems. Some of the known relationships between the various complexity classes mentioned above are

Proposition 15.

1. $P \subseteq RP \subseteq NP \subseteq PP$ and $P \subseteq \text{coRP} \subseteq \text{coNP} \subseteq PP$.
2. $RP, \text{coRP} \subseteq BPP \subseteq PP$.
3. $PP \subseteq P^{\#P} = P^{PP}$.
4. [158, 211] $BPP \subseteq \Sigma_2^P \cap \Pi_2^P$.

At the time of writing, the following problems remain open.

Open problem 16.

1. Is $BPP \subseteq NP$, or is $NP \subseteq BPP$?

2. Is $BPP = P$?
3. Is $PP \subseteq PH$, or is $PH \subseteq PP$?

2.6.5 Bounded-error quantum polynomial time

The central complexity class capturing the power of quantum computation is the class BQP (*bounded-error quantum polynomial time*), which is the quantum generalization of the class BPP.

Definition 17. A language L is in BQP if there exist a polynomial-time Turing machine M and a polynomial p such that for all $x \in \{0, 1\}^*$,

$$M(x) = Q_x ,$$

where Q_x is a (description of a) quantum circuit over the gate set $\{H, T, CZ\}$ on $p(|x|)$ qubits such that

$$\begin{aligned} x \in L &\implies \left| \langle 1|_1 Q_x |0\rangle^{\otimes p(|x|)} \right|^2 \geq \frac{2}{3}, \\ x \notin L &\implies \left| \langle 1|_1 Q_x |0\rangle^{\otimes p(|x|)} \right|^2 \leq \frac{1}{3}. \end{aligned}$$

A few remarks about Definition 17 are in order. First, we note that the quantity $\left| \langle 1|_1 Q_x |0\rangle^{\otimes p(|x|)} \right|^2$ is the probability that one obtains an output 1 when the first qubit of the state produced from acting Q_x on $|0\rangle^{\otimes p(|x|)}$ is measured. Because the swap gate can be produced from the gate set $\{H, S, CZ\}$, it is without loss of generality that we chose the first qubit to be the one to be measured.

Second, we note that the description of the quantum circuit Q_x is produced by a polynomial-time Turing machine. This condition, called *uniformity*, ensures that one is not hiding extra power in the definition of BQP. For example, without the uniformity criterion, one could just define Q_x to output 1 if x encodes the string (M, t) , where M is a Turing machine that halts on t , and 0 otherwise. This would enable Q_x to solve the halting problem, which is not decidable. Subsequently, we

will use the following terminology: a *polynomial-time uniformly generated* family of quantum circuits is a set of circuits $\{C_n\}$ such that a classical Turing machine can produce a description of C_n on input n in time polynomial in the length of n .

Third, we chose the strictly universal Clifford+ T gate set $\{H, T, CZ\}$ for the quantum circuit Q_x . Such a choice is arbitrary, and due to the Solovay-Kitaev theorem (see Theorem 2), we are free to choose other universal gate sets as well.

Fourth, the bounds $2/3$ and $1/3$ are arbitrary. The amplification lemma could be used to achieve exponentially small errors without changing the definition of BQP.

Fifth, while we defined BQP in terms of the quantum circuit model, it is possible to arrive at an equivalent definition using other universal models of quantum computation, like the quantum Turing machine [29, 82], quantum adiabatic computation [14, 96], quantum walks [67] or quantum cellular automata [175].

Since classical randomness is a special case of quantum randomness, it follows that

Proposition 18. $BPP \subseteq BQP$.

It remains an open question whether this inclusion is proper:

Open problem 19. Is $BPP \subsetneq BQP$?

This was the open question that we alluded to in Chapter 1.1 of the introduction. A proof that BQP is strictly larger than BPP would show that there are languages which quantum computers can solve efficiently but which classical computers cannot. A plausible candidate for such a language is the (decision version of the) factoring problem, which is in BQP due to Shor's algorithm [205, 207]. Whether the factoring problem is in BPP is still an open question; if it can be shown that it is not in BPP, then Open Problem 19 would be solved in the positive.

One of tightest 'natural' upper bounds for BQP by a classical complexity class is the class PP:

Proposition 20 ([11, 77]). $BQP \subseteq PP$.

The above proposition may be proved using Feynman’s sum-over-paths technique [103] that expresses quantum amplitudes as an exponential sum of polynomial-sized terms. In Chapters 6 and 7, we will explore applications of this technique to proving results about the simulability of qudit Clifford circuits.

2.6.6 Postselection

Postselection is the ability to discard all runs of a computation except those that yield a particular result [2]. This definition includes the case when the result is an exponentially-unlikely event, and hence, postselection is not a realistic ability to possess in the real world. Nevertheless, we will show that considering the power of postselection yields useful results about the power of quantum computers in the real world. We begin by defining classical and quantum complexity classes that are equipped with the power of postselection.

A language L is in **PostBPP** if there exist a pair of polynomial-time Turing machines A and B and a polynomial p such that for all $x \in \{0, 1\}^*$,

1. $\Pr_{r \in \{0,1\}^{p(|x|)}}[B(x, r) = 1] > 0$,
2. $x \in L \implies \Pr_{r \in \{0,1\}^{p(|x|)}}[A(x, r) = 1 | B(x, r) = 1] \geq \frac{2}{3}$,
3. $x \notin L \implies \Pr_{r \in \{0,1\}^{p(|x|)}}[A(x, r) = 1 | B(x, r) = 1] \leq \frac{1}{3}$.

Here, the Turing machine B is called the *postselector*. From the definition, we see that we consider only the cases when $B(x, r) = 1$ and discard all other outcomes. Note that the class **PostBPP** is equivalent to the class BPP_{path} defined in [118].

The class **PostBQP** is the quantum analogue of **PostBPP** and may be defined as follows [2, 154]: a language L is in **PostBQP** if there exists a polynomial-time uniformly generated family $\{C_x\}_{x \in \{0,1\}^*}$ of quantum circuits such that for all $x \in \{0, 1\}^*$,

1. $\Pr[C_x(2) = 1] > 0$,
2. $x \in L \implies \Pr[C_x(1) = 1 | C_x(2) = 1] \geq \frac{2}{3}$,
3. $x \notin L \implies \Pr[C_x(1) = 1 | C_x(2) = 1] \leq \frac{1}{3}$.

Here, $C_x(k)$ denotes the output obtained when k th qubit of the state $C_x|\vec{0}\rangle$ is measured. Here, $k = 1$ is called the *output register*, and $k = 2$ is called the *postselection register*. Similarly to PostBPP, here we discard all rounds of the computation when $C_x(2) \neq 1$.

Since adding postselection cannot decrease the power of a computational model, we have

$$\text{BPP} \subseteq \text{PostBPP}, \quad \text{BQP} \subseteq \text{PostBQP}.$$

Since postselected classical computation is a special case of postselected quantum computation,

$$\text{PostBPP} \subseteq \text{PostBQP}.$$

Our next proposition relates PostBPP to the polynomial hierarchy.

Proposition 21. ([118])

$$\begin{aligned} \text{NP} &\subseteq \text{PostBPP} \subseteq \Sigma_3^p. \\ \text{coNP} &\subseteq \text{PostBPP} \subseteq \Pi_3^p. \end{aligned}$$

That is, PostBPP is between the first and third levels of the polynomial hierarchy. In fact, we could prove something slightly stronger [118]:

$$\text{P}^{\text{PostBPP}} \subseteq \Sigma_3^p \cap \Pi_3^p. \tag{2.57}$$

The relationship between PostBPP and the second level of the polynomial hierarchy remains an open question. We do know, however, that if $\text{PostBPP} \subseteq \Sigma_2^p$, then PH collapses to the third level [118]. Furthermore, it was proved in [31] that there is an oracle relative to which PostBPP is not contained in Σ_2^p .

The class PostBPP appears to be much less powerful than PostBQP though, as Aaronson's theorem suggests:

Theorem 22. (Aaronson’s Theorem [1, 2])

$$\text{PostBQP} = \text{PP}.$$

Hence, by Open Problem 16, the relationship between PostBQP and the polynomial hierarchy remains an open question. Using the above theorems, it follows that it is implausible that $\text{PostBPP} = \text{PostBQP}$:

Corollary 23. If $\text{PostBPP} = \text{PostBQP}$, then $\text{PH} = \Sigma_3^p$.

Proof. Assume that $\text{PostBPP} = \text{PostBQP}$. Then,

$$\begin{aligned} \text{PH} &\subseteq \text{P}^{\#\text{P}} = \text{P}^{\text{PP}} && \text{by Proposition 15 and Theorem 12} \\ &\subseteq \text{P}^{\text{PostBQP}} && \text{by Proposition 22} \\ &\subseteq \text{P}^{\text{PostBPP}} && \text{by assumption} \\ &\subseteq \Sigma_3^p && \text{by Eq. (2.57).} \end{aligned}$$

□

In other words, if $\text{PostBPP} = \text{PostBQP}$, then the polynomial hierarchy collapses to the third level. We conclude this section by referring the reader to Figure 2-3, which contains a summary of all the known subset relations between the various complexity classes that have been discussed so far.

2.7 Notions of classical simulation of quantum computation

A central question in this thesis that we have discussed is whether quantum computers can be efficiently classically simulated. But what does it mean to ‘classically simulate’ a quantum computation? It turns out that the term ‘classically simulate’ has been used to refer to a variety of different notions. In this section, we will seek to clarify

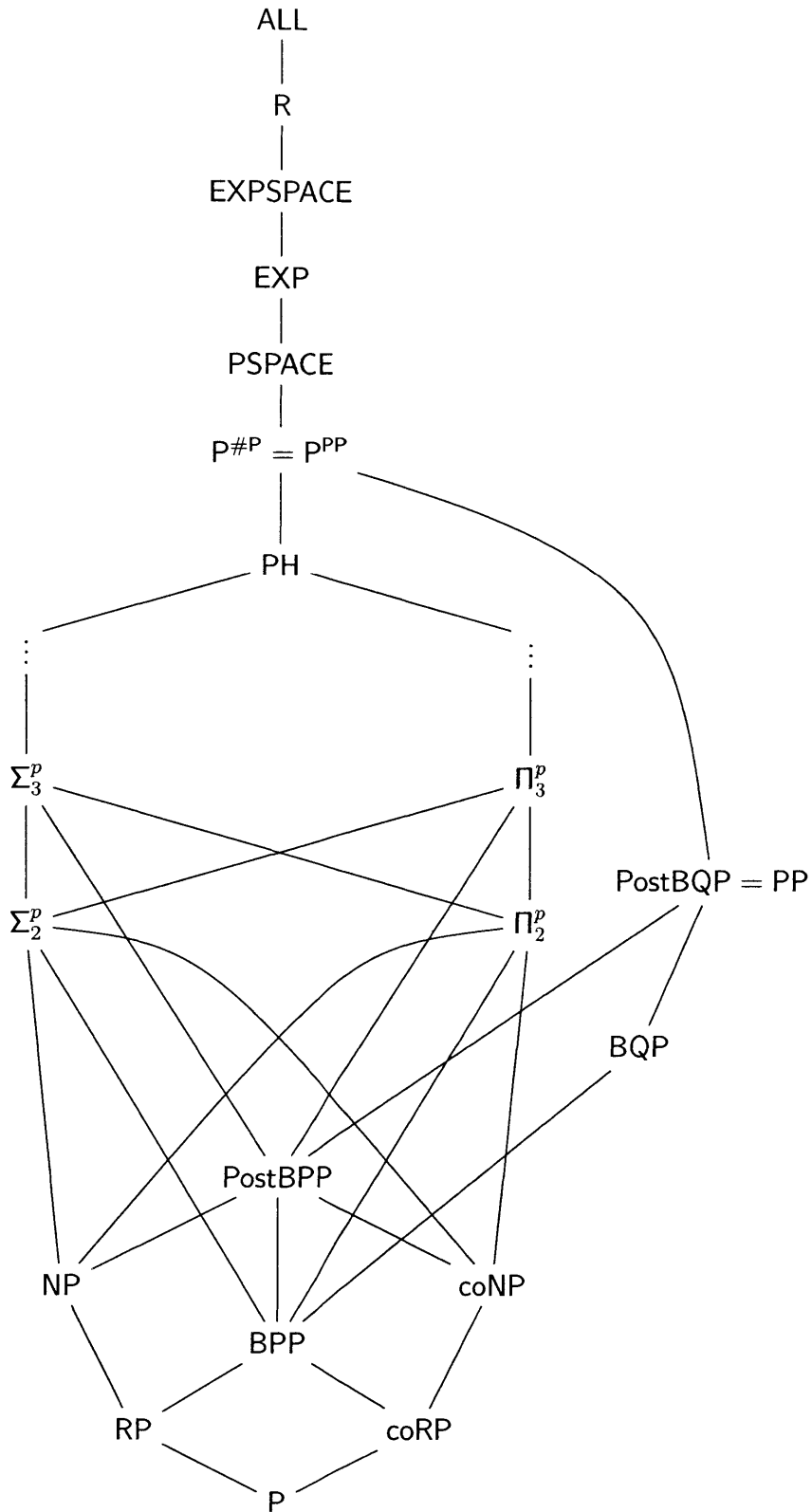


Figure 2-3: Hasse diagram representing the known subset relations (\subseteq) between the various classes introduced in Chapter 2.6.

the distinctions and relationships between some of these various notions. We begin with the notion of strong simulation.

Definition 24. A class of quantum circuits \mathcal{Q} can be (efficiently) *strongly simulated* if there exists a polynomial-time Turing machine M such that if $Q \in \mathcal{Q}$ is a circuit on n qubits, $I \subseteq [n]$, and $y \in \{0, 1\}^{|I|}$, then

$$M(Q, I, y) = |\langle y|_I Q |0^n\rangle|^2.$$

We call M a *strong simulator* of \mathcal{Q} , and write $\mathcal{Q} \in \text{PSTR}$ if \mathcal{Q} can be efficiently strongly simulated.

Lemma 25. Let \mathcal{Q} be a family of circuits over the gate set $\{H, S, CZ\}$. Then the strong simulation of \mathcal{Q} is $\#\text{P}$ -hard, i.e. if $\mathcal{Q} \in \text{PSTR}$, then $\text{FP} = \#\text{P}$.

One approach to proving Lemma 25 is to encode the solution of a $\#\text{P}$ -hard problem into the amplitudes of a quantum circuit; see [81] for an example of such a proof. We will give our own proof of Lemma 25 in the form of Theorems 39 and 40, where we prove that the strong simulation of a non-universal family of circuits is $\#\text{P}$ -hard. Due to Lemma 25, we do not expect the strong simulation of universal quantum circuits to be possible.

Next, we introduce the notion of weak simulation.

Definition 26. A class of quantum circuits \mathcal{Q} can be (efficiently) *weakly simulated* if there exist a polynomial-time Turing machine M and a polynomial $p : \mathbb{N} \rightarrow \mathbb{N}$ such that if $Q \in \mathcal{Q}$ is a circuit on n qubits, $I \subseteq [n]$, and $y \in \{0, 1\}^{|I|}$, then

$$\Pr_{r \in \{0,1\}^{p(|Q|)}}[M(Q, I, r) = y] = |\langle y|_I Q |0^n\rangle|^2.$$

We call M a *weak simulator* of \mathcal{Q} , and write $\mathcal{Q} \in \text{PWK}$ if \mathcal{Q} can be efficiently weakly simulated.

In other words, a weak simulation of a quantum circuit Q samples from the output distribution of Q . Now, suppose that \mathcal{Q} is a family of circuits over the gate set

$\{H, S, CZ\}$. If $\mathcal{Q} \in \text{PWK}$, then it follows from the definitions above that $\text{BPP} = \text{BQP}$. Moreover, it also follows that $\text{PostBPP} = \text{PostBQP}$, which implies that the polynomial hierarchy collapses to the third level, by Corollary 23. Hence, we obtain the following lemma (which is special case of Corollary 1 of [48]).

Lemma 27. Let \mathcal{Q} be a family of quantum circuits over the gate set $\{H, S, CZ\}$. If $\mathcal{Q} \in \text{PWK}$, then the polynomial hierarchy collapses to the third level.

In fact, this result can be improved to a second-level collapse by using a different argument from Fujii et al. [104]:

Lemma 28. Let \mathcal{Q} be a family of quantum circuits over the gate set $\{H, S, CZ\}$. If $\mathcal{Q} \in \text{PWK}$, then the polynomial hierarchy collapses to the second level.

We will elaborate on this result in Chapter 5 of this thesis, where we will subject Lemma 28 to a fine-grained analysis.

The difference between strong and weak simulation is that strong simulation involves calculating probabilities, whereas weak simulation involves sampling from probability distributions. The following proposition shows that strong simulation implies weak simulation.

Lemma 29. (Proposition 1 of [217]) Let \mathcal{Q} be a family of quantum circuits. Then $\mathcal{Q} \in \text{PSTR} \implies \mathcal{Q} \in \text{PWK}$.

In Chapter 3 (see Definitions 36 and 37), we will introduce the notations $\text{PSTR}(f(n))$ and $\text{PWK}(f(n))$, which are generalizations of the notations PSTR and PWK that take into account the number of qubits being measured.

So far, the notions of classical simulation we introduced are exact and do not take simulation error into account. But since real physical systems are susceptible to noise, any realistic notion of simulation should also take error into account. To this end, we will next discuss approximate notions of simulation.

Let $\mathcal{P} = \{p_z\}_z$ and $\mathcal{Q} = \{q_z\}_z$ be (discrete) probability distributions, and let $\epsilon \geq 0$. We say that \mathcal{Q} is a *multiplicative ϵ -approximation* of \mathcal{P} if for all z ,

$$|p_z - q_z| \leq \epsilon p_z. \tag{2.58}$$

We say that \mathcal{Q} is an *additive ϵ -approximation* of \mathcal{P} if

$$\frac{1}{2} \sum_z |p_z - q_z| \leq \epsilon. \quad (2.59)$$

Note that any multiplicative ϵ -approximation is also an additive $\epsilon/2$ -approximation, since summing Eq. (2.58) over all z produces Eq. (2.59).

Definition 30 (multiplicative (additive) error). A *weak simulation with multiplicative (additive) error $\epsilon > 0$* of a family of quantum circuits is a classical randomized algorithm that samples from a distribution that is a multiplicative (additive) ϵ -approximation of the output distribution of the circuit.

Note that from an experimental perspective, additive error is the more appropriate choice, since the fault-tolerance theorem merely guarantees additive closeness between the ideal and realized output distributions [13].

In [48], it was proved that Lemma 27 still holds if we replace the notion of weak simulation with weak simulation with multiplicative error. Strengthening Lemma 27 to the case of additive error remains an open question. We will elaborate on these approximate notions of simulation in Chapter 4.

2.8 Restricted models of quantum computation

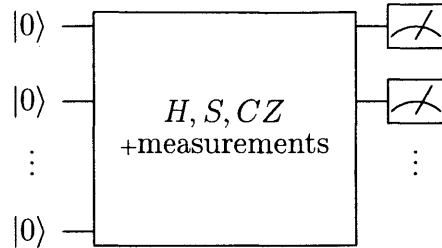
As defined in Chapter 1.1, a restricted model of quantum computation is one that has limited resources available to it. In the quantum circuit model, common ways of limiting resources include restricting the types of inputs or measurements allowed or restricting the structure of the circuit. In this section, we will introduce various well-known restricted models of quantum computation and discuss some of their properties.

2.8.1 Clifford circuits

Clifford circuits are quantum circuits whose gates belong to the Clifford group. Recall from Chapter 2.5 that the Clifford group is generated by the Hadamard, phase and

controlled- Z gates. We will refer to these generators as *basic Clifford gates*.

More precisely, a *Clifford circuit* is a quantum circuit of the following form:



1. Start with $|0\rangle^{\otimes n}$.
2. Apply a polynomial number of basic Clifford gates H, S, CZ , and intermediate measurements in the computational basis.
3. Measure a subset of the qubits in the computational basis.

If no intermediate measurements are performed in Step 2, we say that the Clifford circuit is *unitary*. The following theorem characterizes the complexity of simulating Clifford circuits.

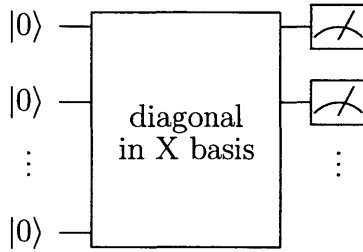
Theorem 31. ([81, 111]) Let \mathfrak{C} be the class of Clifford circuits (with intermediate measurements), and let $\mathfrak{U} \subset \mathfrak{C}$ be the class of unitary Clifford circuits. Then,

1. (Gottesman-Knill Theorem—strong version) $\mathfrak{U} \in \text{PSTR}$.
2. (Gottesman-Knill Theorem—weak version) $\mathfrak{C} \in \text{PWK}$.
3. If $\mathfrak{C} \in \text{PSTR}$, then $\text{FP} = \#\text{P}$.

By Theorems 29 and 31, unitary Clifford circuits can be efficiently simulated by a classical computer in both the weak and strong senses. If intermediate measurements are allowed in the Clifford circuit, then efficient weak simulation is still possible, though strong simulation becomes $\#\text{P}$ -hard. In Chapters 3 and 4, we will explore the interplay between different notions of simulation and the ingredients of a Clifford circuit.

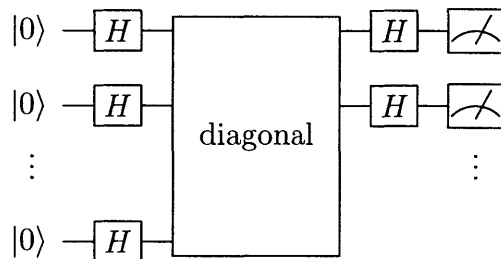
2.8.2 Instantaneous quantum polynomial-time circuits

Instantaneous quantum polynomial-time (IQP) circuits were first introduced by Shepherd and Bremner in [201,202]. These are circuits whose gates all commute, and hence are *temporally unstructured*. They are defined to be circuits of the following form:



1. Start with $|0\rangle^{\otimes n}$.
2. Apply gates which are diagonal in the X basis.
3. Measure a subset of the qubits in the computational basis.

Note that an IQP circuit can alternatively be represented in terms of gates that are diagonal in the Z basis (i.e. computational basis). In this representation, IQP circuits take the following form:



1. Start with $|0\rangle^{\otimes n}$.
2. Apply $H^{\otimes n}$.
3. Apply gates which are diagonal in the Z basis.
4. Apply $H^{\otimes n}$.

5. Measure a subset of the qubits in the computational basis.

Because $H^2 = I$, the above two representations of IQP circuits coincide. The next result characterizes the simulation complexity of IQP circuits.

Theorem 32. (Bremner-Jozsa-Shepherd [48]) Let \mathfrak{C} be the class of IQP circuits. If $\mathfrak{C} \in \text{PWK}$, then the polynomial hierarchy collapses to the third level.

As with Lemma 28, this result may be improved to a second-level collapse. In Chapter 5, we explore this in greater detail.

2.8.3 Depth-one QAOA circuits

The Quantum Approximate Optimization Algorithm (QAOA) is a hybrid quantum-classical variational algorithm introduced by Farhi, Goldstone and Gutmann that is designed to approximately solve combinatorial optimization problems [94].

The input to the algorithm is a sequence c_1, \dots, c_m of m clauses on n variables $z = (z_1, \dots, z_n)$, where each variable z_i takes values in $\{0, 1\}$. Each of these clauses c_α is a constraint on a subset of variables which is satisfied by some assignments of bits and unsatisfied by others. For each assignment $z \in \{0, 1\}^n$, write

$$c_\alpha(z) = \begin{cases} 1 & \text{if } z \text{ satisfies } c_\alpha \\ 0 & \text{otherwise.} \end{cases}$$

The objective function of an assignment is the number of satisfied clauses

$$c : \{0, 1\}^n \rightarrow \{0, 1, \dots, m\}$$

$$c(z) = \sum_{\alpha=1}^m c_\alpha(z).$$

The goal of the algorithm is to find a string $z^* \in \{0, 1\}^n$ for which $c(z^*)$ is close to the maximum of $c(\cdot)$, i.e.

$$c(z^*) \approx \max_z c(z).$$

To describe the QAOA, we first introduce some notation. Define

$$C_\alpha = \sum_{z \in \{0,1\}^n} c_\alpha(z) |z\rangle\langle z|,$$

and let

$$C = \sum_{\alpha=1}^m C_\alpha = \sum_{z \in \{0,1\}^n} c(z) |z\rangle\langle z|.$$

The operator C is called the *problem Hamiltonian*.

For $\gamma \in [0, 2\pi)$, let

$$U(C, \gamma) = e^{-i\gamma C} = \prod_{\alpha=1}^m e^{-i\gamma C_\alpha},$$

where the second equality holds because all C_α 's commute.

Define

$$B = \sum_{j=1}^n X_j,$$

where X_j is the n -qubit operator that acts as the Pauli- X operator on the j th qubit, and the identity operator on all other qubits. The operator B is called the *mixing Hamiltonian*.

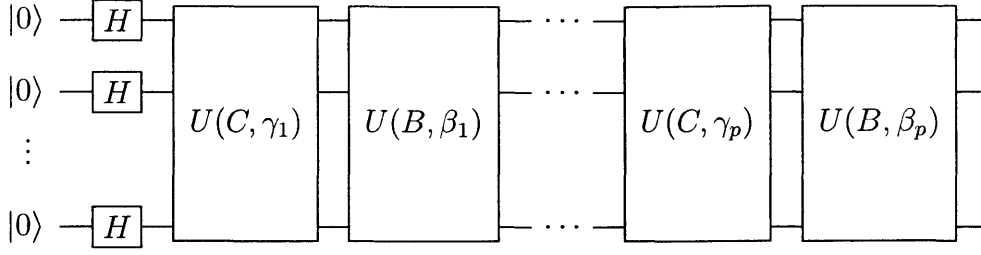
For $\beta \in [0, \pi)$, define

$$U(B, \beta) = e^{-i\beta B} = \prod_{j=1}^n e^{-i\beta X_j}.$$

Set $p \in \mathbb{Z}^+$. Let $\vec{\gamma} = \gamma_1 \dots \gamma_p$ and $\vec{\beta} = \beta_1 \dots \beta_p$, where $\gamma_i \in [0, 2\pi)$ and $\beta_i \in [0, \pi)$.

A p -level QAOA circuit corresponding to clauses c_1, \dots, c_m and sets of angles $\vec{\gamma}$

and $\vec{\beta}$ is



Note that the depth of the above circuit is $mp + p$. We denote the output of the circuit by

$$|\vec{\gamma}, \vec{\beta}\rangle = U(B, \beta_p)U(C, \gamma_p) \dots U(B, \beta_1)U(C, \gamma_1)H^{\otimes n} |0\rangle^{\otimes n}.$$

Let the expectation value of C with respect to the state $|\vec{\gamma}, \vec{\beta}\rangle$ be denoted by

$$F_p(\vec{\gamma}, \vec{\beta}) = \langle \vec{\gamma}, \vec{\beta} | C | \vec{\gamma}, \vec{\beta} \rangle.$$

The QAOA proceeds as follows. First, pick a p and start with sets of angles $\vec{\gamma}$ and $\vec{\beta}$ that make $F_p(\vec{\gamma}, \vec{\beta})$ as large as possible. Second, use a quantum computer to produce the state $|\vec{\gamma}, \vec{\beta}\rangle$, and then measure this state in the computational basis to get a string z , and evaluate $C(z)$. Now, repeat Step 2 with the same angles to get a good estimate of $F_p(\vec{\gamma}, \vec{\beta})$. Then, use a classical computer to search for optimal parameters $(\vec{\gamma}^*, \vec{\beta}^*)$ to maximize $F_p(\vec{\gamma}, \vec{\beta})$, i.e.

$$(\vec{\gamma}^*, \vec{\beta}^*) = \arg \max_{\vec{\gamma}, \vec{\beta}} F_p(\vec{\gamma}, \vec{\beta}).$$

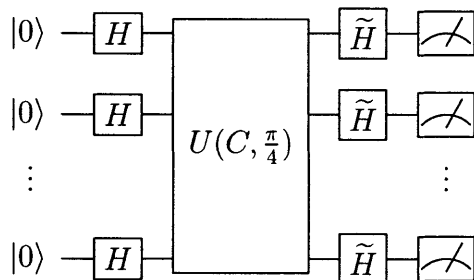
The performance of the QAOA may be measured by the approximation ratio

$$r = \frac{F_p(\vec{\gamma}^*, \vec{\beta}^*)}{\max_z c(z)}.$$

The QAOA has been applied to several combinatorial optimization problems. For example, Farhi, Goldstone and Gutmann showed that QAOA at level $p = 1$ can

solve the Max-Cut problem on 3-regular graphs with an approximation ratio of 0.692 [94], which is better than random guessing, but still not as good as the best known classical approximation algorithms for Max-Cut³. Another example where the QAOA has been applied is the combinatorial problem of bounded occurrence Max E3LIN2. For this problem, the QAOA briefly held the record for the best approximation ratio [95], until a more efficient classical algorithm was proposed [21]. Whether the QAOA is capable of producing better approximation ratios than classical algorithms remains an open question and is an active area of research.

In this thesis, we are interested in the $p = 1$ QAOA circuit. Taking $\beta_1 = \gamma_1 = \pi/4$, the $p = 1$ QAOA circuit becomes



where

$$\tilde{H} = e^{-i\pi/4} H S H = H \exp(-i\frac{\pi}{4} Z) H.$$

The above circuit has almost the same structure as an IQP circuit, except that the qubits are acted on by \tilde{H} instead of H before measurement. By modifying the proof of Theorem 32, we get a similar hardness result for $p = 1$ QAOA circuits.

Theorem 33. (Farhi-Harrow [97]) Let \mathfrak{C} be the class of $p = 1$ QAOA circuits. If

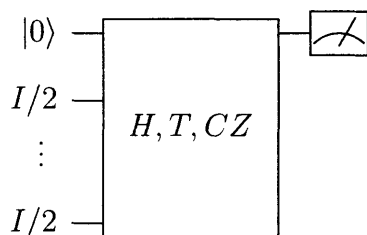
³Max-Cut is a combinatorial optimization problem whose goal is to find the maximum cut of a graph. Its approximate optimization above a minimum ratio r^* is known to be NP-hard. It has been proved that for general graphs, $r^* \geq 16/17 \approx 0.941$ [125], and for unweighted 3-regular graphs, $r^* \geq 331/332 \approx 0.997$ [26]. There is a gap between these hardness results and the best known approximation ratios by efficient classical algorithms: for general graphs, the Goemans-Williamson algorithm achieves an approximation ratio of $r \approx 0.879$ [107], whereas for unweighted 3-regular graphs, the Halperin-Livnat-Zwick algorithm gives an approximation ratio of $r \approx 0.933$ [117]. In contrast, the $p = 1$ QAOA algorithm of Farhi, Goldstone and Gutmann gives an approximation ratio of only 0.692 [94].

$\mathfrak{C} \in \text{PWK}$, then the polynomial hierarchy collapses to the third level.

As with Lemma 28 and Theorem 32, this result may be improved to a second-level collapse. We will discuss this in greater detail in Chapter 5.

2.8.4 DQC1 circuits

Deterministic quantum computation with one quantum bit (DQC1), introduced by Knill and Laflamme [146], is a restricted model of quantum computation, where the input consists of a single clean qubit in the computational basis, and noisy qubits in the maximally mixed state in all other registers. A polynomial-sized circuit over a universal gate set is applied to the state, before a single qubit is measured. For example, a DQC1 circuit over the universal gate set $\{H, T, CZ\}$ is given by:



1. Start with $|0\rangle\langle 0| \otimes (\frac{I}{2})^{\otimes n}$.
2. Apply a polynomial number of gates H, T, CZ .
3. Measure the first qubit in the computational basis.

The following theorem characterizes the classical simulation complexity of DQC1 circuits.

Theorem 34. (Fujii et al. [104]) Let \mathfrak{C} be the class of DQC1 circuits. If $\mathfrak{C} \in \text{PWK}$, then the polynomial hierarchy collapses to the second level.

2.8.5 Boson sampling model

In the boson sampling model [5], a system with n photons and m modes is represented by a superposition $\sum_R \alpha_R |R\rangle$, where $R = (r_1, \dots, r_m)$, where $r_i \in \mathbb{N}$ represents the

number of photons in mode $i \in \{1, \dots, m\}$ and $\sum_i r_i = n$.

Passing these photons through a linear optical network composed of beam splitters and phase shifters, which we call a *boson sampling circuit*, gives rise to a transformation on this Hilbert space. Valid transformations are represented by operators of the form $\phi(U)$, where U is an $m \times m$ unitary and ϕ is a fixed $\binom{n+m-1}{n}$ -dimensional representation of $\mathcal{U}(m)$. The unitary U fully describes the choice of circuit, and any U can be exactly implemented using only $m(m+1)/2$ total beam splitters and phase shifters [194]. We define $\phi(U)$ by its matrix elements $\langle R | \phi(U) | R' \rangle$, which are related to the permanent of $n \times n$ matrices formed from U . Here, the permanent of an $n \times n$ matrix A is given by the formula

$$\text{Per}(A) = \sum_{\sigma \in \mathcal{S}_n} \prod_{i=1}^n A_{i, \sigma(i)} \quad (2.60)$$

where \mathcal{S}_n is the group of permutations on $\{1, \dots, n\}$. Then, the matrix elements are

$$\langle R | \phi(U) | R' \rangle = \frac{\text{Per}(U_{(R,R')})}{\sqrt{r_1! \dots r_m! r'_1! \dots r'_m!}} \quad (2.61)$$

where $U_{(R,R')}$ is the $n \times n$ matrix formed by taking r_i copies of row i and r'_j copies of column j from U . As an example, if $n = 3$, $m = 2$, $R = (2, 1)$, $R' = (1, 2)$, and

$$U = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & i \\ -i & -1 \end{bmatrix} \quad (2.62)$$

then

$$U_{(R,R')} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & i & i \\ 1 & i & i \\ -i & -1 & -1 \end{bmatrix}. \quad (2.63)$$

This sampling task is called **BosonSampling** since it could be applied to any system of not only photons but any non-interacting bosons.

The following theorem characterizes the classical simulation complexity of the boson sampling model.

Theorem 35. (Aaronson-Arhipov [5]) Let \mathfrak{C} be the class of boson sampling circuits. If $\mathfrak{C} \in \text{PWK}$, then the polynomial hierarchy collapses to the third level.

As with Lemma 28, Theorem 32 and Theorem 33, this result may be improved to a second-level collapse. We will discuss this in greater detail in Chapter 5.

Chapter 3

Extended Clifford circuits and their classical simulation complexities

In this chapter, we study various modifications of Clifford circuits, called *extended Clifford circuits*, and study how the classical simulation complexities of these circuits change as the ingredients in the circuits are modified. We show, under plausible complexity conjectures, that whether such circuits are efficiently classically simulable depends delicately on the ingredients of the circuits. The modifications we consider give us 24 new combinations of ingredients compared to an earlier classification result by Jozsa and Van den Nest [143], and we give a complete classification of their classical simulation complexities. Our results provide more examples where seemingly modest changes to the ingredients of Clifford circuits lead to “large” changes in the classical simulation complexities of the circuits, and also include new examples of extended Clifford circuits that exhibit “quantum supremacy”, in the sense that it is not possible to efficiently classically sample from the output distributions of such circuits, unless the polynomial hierarchy collapses. The results in this chapter are based on [148].

3.1 Motivation

One of the central results about Clifford circuits is the Gottesman-Knill Theorem [111] (see Chapter 2.8.1), which states that such circuits can be efficiently simulated on a classical computer, and hence do not provide a speedup over classical computation. But this is known to be true only in a restricted setting — whether or not we can efficiently classically simulate such circuits seems to depend delicately on the ‘ingredients’ of the circuit, for example, on the types of inputs we allow, whether or not intermediate measurements are adaptive, the number of output lines, and even on the precise notion of what it means to *classically simulate* a circuit. These cases were considered by Jozsa and Van den Nest [143], who showed that many of these ‘extended’ Clifford circuits are in fact not classically simulable under plausible complexity assumptions.

One of the main motivations for studying extended Clifford circuits is that they shed light on the relationship between quantum and classical computational power. Are quantum computers more powerful than their classical counterparts? If so, what is the precise boundary between their powers? One approach to answering this question is to consider restricted models of quantum computation and study their classical simulation complexities, i.e. how hard it is to classically simulate them. For example, suppose that we start with a restricted model that is efficiently classically simulable. If adding certain ingredients to the restricted model creates a new class that is universal for quantum computation, then we could regard those ingredients as an essential ‘resource’ for quantum computational power [143]. Extended Clifford circuits, as a restricted model of quantum computation, are especially well-suited for this approach as they straddle the boundary between classical and quantum computational power. One could give many examples where adding a seemingly modest ingredient to an extended Clifford circuit changes it from being efficiently classically simulable to one that is likely not.

Understanding how the classical simulation complexities of extended Clifford circuits change when various ingredients are added is a central goal of this chapter.

In [143], Jozsa and Van den Nest tabulate the classical simulation complexities of extended Clifford circuits with 16 different combinations of ingredients. In particular, they consider the different combinations of ingredients that arise from 4 binary choices: computational basis inputs vs product state inputs, single-line outputs vs multiple-line outputs, nonadaptive measurements vs adaptive measurements, and weak vs strong simulation. They show that the classical simulation complexities of the extended Clifford circuits are of 4 different types (we use slightly different terminology here): (i) P, which means that the circuits can be efficiently simulated classically, (ii) QC, which means that the circuits are universal for quantum computation, (iii) #P, which means that the problem of classically simulating the circuits is a #P-hard problem, and (iv) PH, which means that if the circuits are efficiently classically simulable, then the polynomial hierarchy collapses.

In this chapter, we extend the results in [143] in two different ways. First, we study how the classical simulation complexity changes when we employ a weaker notion of simulation than strong simulation, which we call STR(n) simulation (short for strong- n simulation). While strong simulation requires that the joint probability as well as any marginal probabilities be computed, in STR(n) simulation, we require only that the joint probability be computed. Note that such a notion seems incomparable with weak simulation. Second, we study how the classical simulation complexity changes when we allow for general product measurements (called OUT(PROD)) instead of just the computational basis measurements (called OUT(BITS)) that were considered in [143]. With these additional ingredients, the number of different combinations of ingredients grows to 40. In Table 3.1, we tabulate the classical simulation complexities of each of these cases.

We now make a few remarks about the extended Clifford circuits labeled in Table 3.1 by PH. These are examples of ‘intermediate’ or restricted quantum circuit models which are not believed to be universal for quantum computation (or perhaps even classical computation), but which exhibit a form of ‘quantum supremacy’ [123, 188], in the sense that they can sample from distributions that are impossible to sample from classically, unless the plausible complexity assumption of the polynomial hier-

archy being infinite is false. In [143], Jozsa and Van den Nest give an example of such a circuit model: nonadaptive Clifford circuits with product state inputs and computational basis measurements. In this chapter, we show that the same behavior holds if we restricted our circuits to having computational basis inputs but allowed them to have arbitrary single-qubit measurements performed at the end of the circuit (see Theorem 46).

The rest of this chapter is structured as follows. In Chapter 3.2, we discuss various extensions of Clifford circuits. In Chapter 3.3, we introduce some basic definitions and notations and define different notions of classical simulation of quantum computation. In Chapter 3.4, we summarize our main results in the form of Table 3.1 and discuss some implications of our results. Our main theorems are Theorems 39–50, whose proofs are presented in Chapters 3.5.2–3.5.7.

3.2 Preliminary definitions and notations

Recall from Chapter 2.8.1 that a unitary Clifford circuit is one that comprises only the basic Clifford gates H , S and CZ , and that a Clifford circuit is one that consists of not just the basic Clifford gates but also single-qubit intermediate measurement gates in the computational basis.

We consider *Clifford computational tasks* of the following form:

1. Start with an n -qubit pure input state $|\psi_{\text{in}}\rangle$.
2. Apply to $|\psi_{\text{in}}\rangle$ a Clifford circuit B , which may be expressed as:

$$B(x_1, \dots, x_K) = C_K(x_1, \dots, x_K)M_{i_K(x_1, \dots, x_{K-1})}(x_K) \dots \\ C_2(x_1, x_2)M_{i_2(x_1)}(x_2)C_1(x_1)M_{i_1}(x_1)C_0, \quad (3.1)$$

where each $C_i(x_1, \dots, x_i)$ is a unitary Clifford circuit and $M_i(x)$ indicates a measurement on qubit line i with measurement result x . In general, B is taken to be an adaptive circuit, i.e. the i th unitary Clifford circuit C_i depends on

previous measurement results x_1, \dots, x_i . Let N denote the total number of gates in B . Assume that there are no extraneous qubits, so that $n = O(N)$.

3. Measure all n qubit lines using a projection-valued measure

$$\{|\beta_{y_1, \dots, y_n}\rangle\langle\beta_{y_1, \dots, y_n}|\}_{y_1, \dots, y_n},$$

with measurement outcome $y_1 \dots y_n \in \{0, 1\}^n$.

In this work, we restrict our attention to product state inputs and product measurements (i.e. arbitrary single-qubit measurements), i.e.

1. Inputs are $|\psi_{\text{in}}\rangle = |\alpha_1\rangle |\alpha_2\rangle \dots |\alpha_n\rangle$, where each $|\alpha_i\rangle \in \mathbb{C}^2$.
2. Measurement directions are $|\beta_{y_1, \dots, y_n}\rangle = |\beta_1^{y_1}\rangle |\beta_2^{y_2}\rangle \dots |\beta_n^{y_n}\rangle$, where each $|\beta_i^{y_i}\rangle \in \mathbb{C}^2$.

Note that for each i , by completeness, $|\beta_i^0\rangle\langle\beta_i^0| + |\beta_i^1\rangle\langle\beta_i^1| = I$. Hence, we need to just specify $\{|\beta_i^0\rangle\langle\beta_i^0|\}_i$ in order to completely specify the product measurement. A description of the Clifford computational task is thus given by the three-tuple

$$T = (|\alpha\rangle, B, |\beta\rangle), \tag{3.2}$$

where $|\alpha\rangle = |\alpha_1\rangle |\alpha_2\rangle \dots |\alpha_n\rangle$ is the initial state, B is the description of the Clifford circuit, and $|\beta\rangle = |\beta_1^0\rangle |\beta_2^0\rangle \dots |\beta_n^0\rangle$ are the measurement directions.

Now, each product state input can be seen as arising from applying a product unitary to the computational basis states, i.e. there exist single-qubit unitary operators V_1, \dots, V_n such that $V_1 \otimes \dots \otimes V_n |0 \dots 0\rangle = |\alpha\rangle$. Likewise, every product measurement operator can be seen as arising from applying a product unitary operator followed by measuring in the computational basis. More precisely, a measurement in the direction $|\beta_1^{y_1}\rangle |\beta_2^{y_2}\rangle \dots |\beta_n^{y_n}\rangle$ is equivalent to the application of a unitary operator $U_1^\dagger \otimes \dots \otimes U_n^\dagger$ followed by a measurement in the computational basis, where the y_i th (with zero indexing) column of U_i is given by $U_i |y_i\rangle = |\beta_i^{y_i}\rangle$.

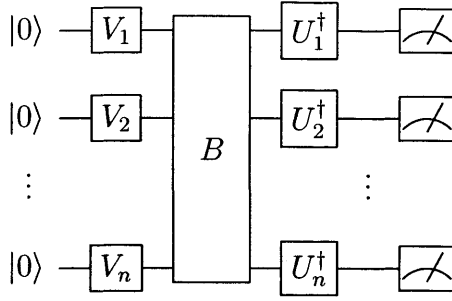


Figure 3-1: Circuit diagram for the Clifford computational tasks considered in this chapter. The gates V_i and U_i^\dagger are arbitrary single qubit unitaries, B is a Clifford circuit, the input state is the all-zero computational basis state, and the output measurement is performed in the computational basis.

Hence, the Clifford computational tasks we consider are of the structure shown in Figure 3-1. They may alternatively be represented by the 3-tuple

$$T = (\{V_i\}_{i=1}^n, B, \{U_i\}_{i=1}^n). \quad (3.3)$$

We will use the above two descriptions in Eqs. (3.2) and (3.3) of Clifford tasks interchangeably, and even allow for mixed descriptions, for example, $T = (|\alpha\rangle, B, \{U_i\}_{i=1}^n)$.

We'll now write down expressions for the probabilities of outcomes. For a computational task $T = (|\alpha\rangle, B, |\beta\rangle)$ and subset $I = \{i_1, \dots, i_s\} \subseteq [n]$, let $P_T^I(y_{i_1}, \dots, y_{i_s})$ be the marginal probability that the outputs y_{i_1}, \dots, y_{i_s} are obtained in the lines i_1, \dots, i_s . Define $P_T(y_1, \dots, y_n) = P_T^{[n]}(y_1, \dots, y_n)$ to be the probability of the outcome $y_1 \dots y_n$.

For the adaptive circuit described by Eq. (3.1), if the intermediate measurement results are $x_1 \dots x_K$, then the density operator of the final state is given by

$$B(x_1, \dots, x_K)[\rho_\alpha], \quad (3.4)$$

where $\rho_\alpha = |\alpha\rangle\langle\alpha|$. We use the notation $C[\rho]$ to denote the state that is obtained when we apply C to the density matrix ρ , i.e. $C[\rho] = C\rho C^\dagger$. The probability that the

result $x_1 \dots x_K$ occurs is given by

$$p(x_1, \dots, x_K) = p(x_K|x_1, \dots, x_{K-1})p(x_{K-1}|x_1, \dots, x_{K-2}) \dots p(x_2|x_1)p(x_1),$$

where

$$\begin{aligned} p(x_j|x_1, \dots, x_{j-1}) &= \text{tr}\{|x_j\rangle\langle x_j|_{i_j} C_{j-1}(x_1, \dots, x_{j-1}) M_{i_{j-1}(x_1, \dots, x_{j-2})}(x_{j-1}) \dots \\ &\quad \times C_1(x_1) M_{i_1}(x_1) C_0[\rho_\alpha]\}. \end{aligned} \quad (3.5)$$

The final output state is then given by

$$B[\rho_\alpha] = \sum_{x_1 \dots x_K} p(x_1, \dots, x_K) B(x_1, \dots, x_K) [\rho_\alpha].$$

Hence, the outcome probabilities are given by

$$p_T(y_1, \dots, y_n) = \langle \beta_{y_1, \dots, y_n} | B[\rho_\alpha] | \beta_{y_1, \dots, y_n} \rangle,$$

and the marginal probabilities are given by

$$p_T^I(y_{i_1}, \dots, y_{i_s}) = \sum_{y_{k_1} \dots y_{k_{n-s}}} p_T(y_1, \dots, y_n), \quad (3.6)$$

where $\{k_1, \dots, k_{n-s}\} = [n] - I$.

We consider the following 3 binary choices of ingredients:

1. Inputs: IN(BITS) vs IN(PROD)
2. Intermediate measurements: NONADAPT vs ADAPT
3. Outputs: OUT(BITS) vs OUT(PROD).

The first two cases have been considered in [143]: IN(BITS) and IN(PROD) refer to having computational basis inputs and product state inputs respectively, while NONADAPT and ADAPT refer to nonadaptive and adaptive measurements respec-

tively. Note that in [143], all the output measurements are performed in the computational basis (call this case $\text{OUT}(\text{BITS})$). In this chapter, we study how the classical simulation complexity changes when we allow for more general measurements. For the sake of symmetry with the inputs, we introduce the new ingredient $\text{OUT}(\text{PROD})$, which refers to product measurements, i.e. when the U_i 's in Eq. (3.3) are unrestricted. Note that we allow product measurements only at the output; intermediate measurements are always single-qubit measurements in the computational basis.

These 3 binary choices lead to $2^3 = 8$ different subsets of Clifford computational tasks. Let $\nu \in \{(\text{IN}(\text{BITS}), \text{NONADAPT}, \text{OUT}(\text{BITS})), (\text{IN}(\text{BITS}), \text{NONADAPT}, \text{OUT}(\text{PROD})), \dots\}$ be one of these 8 subsets. We shall denote the subset of Clifford computational tasks corresponding to ν by \mathcal{C}_ν . Note that unlike [143], we do not include $\text{OUT}(1)$ and $\text{OUT}(\text{MANY})$ as ingredients in our circuit. Instead, we assume without loss of generality that all n qubit lines are measured. This is justified by the principle of implicit measurement, which states that any unterminated quantum wires at the end of the circuit can be assumed to be measured [180]. The number of output lines we simulate will be specified by the notion of simulation instead. We discuss various notions of simulation in the next section.

3.3 Notions of classical simulation

In [143], Jozsa and Van den Nest consider two notions of classical simulation, namely weak (WK) and strong (STR) simulation. As defined in Chapter 2.7, a weak simulation involves providing a sample of the output distribution, whereas a strong simulation involves calculating the joint output probabilities as well as the marginal probabilities. Neither of these definitions places a restriction on the number of output registers to be simulated. To take this into account, we shall introduce finer-grained notions of simulation, namely $\text{STR}(f(n))$ and $\text{WK}(f(n))$ (short for strong- $f(n)$ and weak- $f(n)$) simulation.

Let $f(n)$ be either the constant function $f(n) = 1$ or the identity function $f(n) = n$ (in this chapter, we restrict our attention to these cases, though one might certainly

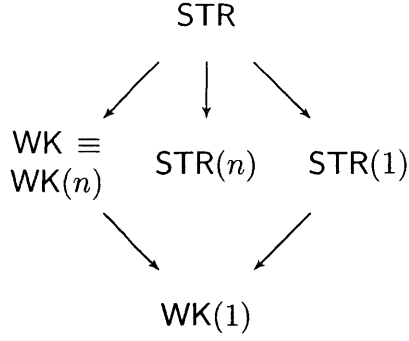


Figure 3-2: Relationships between different notions of classical simulation of Clifford computational tasks. An arrow from A to B ($A \rightarrow B$) means that an efficient A -simulation of a computational task implies that there is an efficient B -simulation for the same task. The statement $\text{WK} \equiv \text{WK}(n)$ is shorthand for $\text{WK} \rightarrow \text{WK}(n)$ and $\text{WK}(n) \rightarrow \text{WK}$.

consider other functions f , like $f(n) = \log(n)$.

Definition 36. ($\text{STR}(f(n))$) A $\text{STR}(f(n))$ simulation of a subset of Clifford computational tasks \mathcal{C}_ν is a deterministic classical algorithm that on input $\langle T, I, y \rangle$, where $T \in \mathcal{C}_\nu$ is a task on n qubits, $I = \{i_1, \dots, i_{f(n)}\} \subseteq [n]$ and $y_I = \{y_{i_1}, \dots, y_{i_{f(n)}}\}$, outputs $p_T^I(y_{i_1}, \dots, y_{i_{f(n)}})$.

Definition 37. ($\text{WK}(f(n))$) A $\text{WK}(f(n))$ simulation of a subset of Clifford computational tasks \mathcal{C}_ν is a randomized classical algorithm that on input $\langle T, I \rangle$, where $T \in \mathcal{C}_\nu$ is a task on n qubits and $I = \{i_1, \dots, i_{f(n)}\} \subseteq [n]$, outputs $y_{i_1}, \dots, y_{i_{f(n)}}$, with probability $p_T^I(y_{i_1}, \dots, y_{i_{f(n)}})$.

STR and WK simulations are defined in exactly the same way, except that we place no restrictions on the size of the subset of output lines $|I|$ in the simulation. Note that this agrees with the definitions of strong and weak simulations in Chapter 2.7 and [143].

Let $\mathcal{S} \in \{\text{STR}(n), \text{STR}(1), \text{STR}, \text{WK}(n), \text{WK}(1), \text{WK}\}$ be one of the 6 notions of simulation depicted in Figure 3-2. We define an \mathcal{S} -simulation of a subset of Clifford computational tasks \mathcal{C}_ν to be *efficient* if the simulation runs in $\text{poly}(N)$ -time, where N is the number of gates in the \mathcal{C}_ν -circuit. Let PS be the set of all tasks \mathcal{C}_ν that have

an efficient \mathcal{S} -simulation.

An immediate observation is that $\text{PWK} = \text{PWK}(n)$. The forward inclusion holds by definition, and the backward inclusion holds because we could sample from any subset I by just sampling from all n lines and ignoring the qubit lines that are not in I . From their definitions, we also immediately get the following inclusions: $\text{PSTR} \subseteq \text{PSTR}(1)$, $\text{PSTR} \subseteq \text{PSTR}(n)$ and $\text{PWK} \subseteq \text{PWK}(1)$. How does weak simulation compare with strong simulation? From Proposition 1 of [217], it follows that $\text{PSTR} \subseteq \text{PWK}$ and $\text{PSTR}(1) \subseteq \text{PWK}(1)$. Note that the notions $\text{PSTR}(n)$ and $\text{PSTR}(1)$ are in general incomparable – the forward inclusion ($\text{PSTR}(n) \subseteq \text{PSTR}(1)$) does not hold in general because computing a marginal distribution directly from the joint distribution involves summing an exponential number of terms and cannot be performed efficiently unless there is some structure in the problem. The backward inclusion ($\text{PSTR}(n) \supseteq \text{PSTR}(1)$) does not hold in general because knowing just the marginal distributions does not allow us to infer the joint distribution. We summarize the relationships between the different notions of simulation stated above in Figure 3-2.

3.4 Results and discussion

In Chapter 3.2, we introduced 3 binary choices of ingredients. In Chapter 3.3, we described 5 different notions of classical simulation (see Figure 3-2). This gives a total of $2^3 \times 5 = 40$ different cases, whose classical simulation complexities we classify in Table 3.1. The entries of the table should be understood as follows: for a subset of computational tasks \mathcal{C}_ν , and a notion of simulation \mathcal{S} ,

- P (classically efficiently simulable) means that $\mathcal{C}_\nu \in \text{PS}$.
- #P (which stands for #P-hard) means that an efficient \mathcal{S} -simulation of \mathcal{C}_ν would give rise to an efficient algorithm for the #P-complete problems.
- QC (which stands for quantum-computing universal) means that \mathcal{C}_ν is universal for quantum computation.

			Weak		Strong		
			WK(1)	WK(n)	STR(1)	STR(n)	STR
OUT (BITS)	NON- ADAPT	IN (BITS)	P (i)	P (ii)	P (iii)	P (iv)	\boxed{P} (JV4)
		IN (PROD)	P (v)	\boxed{PH} (JV7)	P (JV1)	$\boxed{\#P}$ (Thm39)	$\#P$ (JV6)
	ADAPT	IN (BITS)	P (vi)	\boxed{P} (JV5)	$\boxed{\#P}$ (JV2)	$\boxed{\#P}$ (Thm40)	$\#P$ (vii)
		IN (PROD)	\boxed{QC} (JV3)	QC (viii)	$\#P$ (ix)	$\#P$ (x)	$\#P$ (xi)
OUT (PROD)	NON- ADAPT	IN (BITS)	P (xii)	\boxed{PH} (Thm46)	P (xiii)	$\boxed{\#P}$ (Thm48)	$\#P$ (xiv)
		IN (PROD)	P (xv)	PH (xvi)	\boxed{P} (Thm49)	$\#P$ (xvii)	$\#P$ (xviii)
	ADAPT	IN (BITS)	\boxed{P} (Thm50)	PH (xix)	$\#P$ (xx)	$\#P$ (xxi)	$\#P$ (xxii)
		IN (PROD)	QC (xxiii)	QC (xxiv)	$\#P$ (xxv)	$\#P$ (xxvi)	$\#P$ (xxvii)

Table 3.1: Classification of the classical simulation complexities of families of Clifford circuits with different ingredients. P stands for efficiently classically simulable. $\#P$ stands for $\#P$ -hard. QC stands for QC-hard and PH stands for “if efficiently classically simulable, then the polynomial hierarchy collapses”. The proofs of JV 1–7 can be found in [143]. Theorems 39–50 are about cases not found in [143] and are the main results of this chapter. (i)–(xxvii) are results that follow immediately from these theorems by using the rules in Chapter 3.5.1. The 11 cases with boxed symbols are the core theorems, from which all other cases can be deduced using rules which we describe in Chapter 3.5.1. These include all the main theorems JV 1–7 and Theorems 39–50, except JV1 and JV6, which turn out to be special cases of Theorem 49 and Theorem 39 respectively.

- PH means that an efficient \mathcal{S} -simulation of \mathcal{C}_ν would imply a collapse of the polynomial hierarchy.

Our main results are Theorems 39–50, whose proofs we present in Chapter 3.5.2–3.5.7. Using the rules in Chapter 3.5.1, these theorems, together with the results¹ JV 1–7 from [143], give a complete classification of the classical simulation complexities of all the 40 cases.

A few remarks are in order. First, we note that the entries in the last two columns of Table 3.1 are identical. This means that even though the notions $\text{STR}(n)$ and $\text{STR}(1)$ seem to be incomparable, the former is not easier to perform than the latter for the Clifford computational tasks considered in this chapter. We note that Theorem 39, which generalizes (JV6), implies that being able to compute only the joint probabilities already suffices in enabling us to solve the $\#\text{P}$ -hard problems: we do not require the full power of strong simulation for that.

Second, we note the symmetry between inputs and outputs: for example, the 2nd and 5th rows of Table 3.1 are identical, i.e. the simulation complexity is the same whether product unitaries are applied at the beginning or at the end of the circuit. In particular, for (JV7), the key to collapsing the polynomial hierarchy was that the magic state $|\pi/4\rangle = 1/\sqrt{2}(|0\rangle + e^{i\pi/4}|1\rangle)$ together with postselection can simulate the $T = \text{diag}(1, e^{i\pi/4})$ gate. For Theorem 46, although we did not have magic state inputs at our disposal, we still managed to get a similar result to (JV7) by showing that the T gate can be simulated by arbitrary single-qubit measurements with postselection.

Third, we note that Theorem 49 is a generalization of JV1. In fact, a stronger result can similarly be shown to be true: for any constant b , there exists an efficient $\text{STR}(b)$ -simulation of circuits belonging to $\text{OUT}(\text{PROD})$, NONADAPT , $\text{IN}(\text{PROD})$. In [8], Aaronson and Gottesman present algorithms for simulating two separate classes of extended Clifford circuits: circuits with non-stabilizer initial states, and circuits with non-stabilizer gates. A consequence of their results is that it is efficient to simulate (in the $\text{STR}(b)$ -sense) nonadaptive tasks with either of the following ingredients:

1. product state inputs with computational basis measurements (which is the con-

¹JV = Jozsa and Van den Nest [143]

tent of JV1). 2. computational basis inputs with product measurements (which is the content of case xiii) – since this is equivalent to applying b single-qubit gates just before a computational basis measurement. Theorem 49 is slightly more general than either of these cases. Essentially, it combines the case involving product state inputs and the case involving product measurements and shows that the new task is still in $\text{PSTR}(b)$.

3.5 Proofs of main theorems

3.5.1 Rules for proving results in Table 3.1

In this section, we show that the entries in Table 3.1 that contain boxed symbols (for example, $\boxed{\text{PH}}$) can be used to deduce all the other entries in the table. Therefore, for a complete proof of the results in the table, it will suffice to prove just Theorems 39–50 as well as JV 1–6 (save JV1 and JV6). This is a straightforward consequence of a couple of simple rules (cf [143]), which we state explicitly here:

- If the classical simulation of a set of computational tasks \mathcal{A} is efficient, then the classical simulation of any subset of \mathcal{A} would also be efficient.
- If the classical simulation of a set of computational tasks \mathcal{A} is hard ($\#P$ -hard, QC-hard or PH-collapsing in the sense described above), then the classical simulation of any superset of \mathcal{A} would also be similarly hard.
- The set of computational tasks with $\text{IN}(\text{BITS})$ is a subset of the same set of tasks with $\text{IN}(\text{PROD})$. Write this as $\text{IN}(\text{BITS}) \subset \text{IN}(\text{PROD})$. Similarly, $\text{OUT}(\text{BITS}) \subset \text{OUT}(\text{PROD})$, $\text{NONADAPT} \subset \text{ADAPT}$.
- If the strong simulation of a set of tasks is efficient, then so are the $\text{STR}(1)$, $\text{STR}(n)$ and $\text{WK}(n)$ simulations of that set. If any of the latter three notions of simulation is efficient, then $\text{WK}(1)$ -simulation of that set is also efficient (as illustrated in Figure 3-2). In the opposite direction, if $\text{STR}(1)$ or $\text{STR}(n)$ simulation is $\#P$ -hard, then so is strong simulation. Similarly, if $\text{WK}(1)$ simulation

is QC-hard, then WK(n)-simulation is also QC-hard. Note that #P-hardness holds only for strong notions of simulation, and QC-hardness holds only for weak notions of simulation.

3.5.2 Proof of Theorem 39: Strong(n) simulation of nonadaptive Clifford circuits with product inputs and computational basis outputs

A 3-CNF formula f (i.e. a Boolean formula in conjunctive normal form [212]) with n variables and N clauses is of the form

$$f(x_1, \dots, x_n) = (a_{11} \vee a_{12} \vee a_{13}) \wedge (a_{21} \vee a_{22} \vee a_{23}) \wedge \dots \wedge (a_{N1} \vee a_{N2} \vee a_{N3}), \quad (3.7)$$

where each $a_{ij} \in \{x_1, \dots, x_n, \bar{x}_1, \dots, \bar{x}_n\}$. We assume that every variable x_1, \dots, x_n appears in the formula for f , so that $n \leq 3N$, i.e. $n = O(N)$.

We define AbsSAT to be the following problem: Given a 3-CNF formula $f : \{0, 1\}^n \rightarrow \{0, 1\}$, compute

$$S(f) = \left| \sum_{x \in \{0,1\}^n} (-1)^{f(x)} \right|.$$

We shall denote $\#_i(f) = |\{x | f(x) = i\}|$ for $i = 0, 1$. Then $S(f) = |\#_0(f) - \#_1(f)|$.

Lemma 38. AbsSAT is #P-hard.

Proof. We shall construct a reduction from the #P-complete problem #SAT to AbsSAT. Given a #SAT-instance $\phi(x_1, \dots, x_n)$, introduce a new variable y and define the Boolean formula

$$\tilde{\phi}(x_1, \dots, x_n, y) = \phi(x_1, \dots, x_n) \vee y.$$

Let $A(\varphi)$ denote the set of satisfying assignments to a Boolean formula φ . Then

$$A(\tilde{\phi}) = \{(x_1, \dots, x_n, 0) | (x_1, \dots, x_n) \in A(\phi)\} \cup \{(x_1, \dots, x_n, 1) | (x_1, \dots, x_n) \in \{0, 1\}^n\}.$$

Hence, $\#_1(\tilde{\phi}) = \#_1(\phi) + 2^n$, and $\#_0(\tilde{\phi}) = 2^{n+1} - \#_1(\tilde{\phi}) = 2^n - \#_1(\phi)$. This gives

$$S(\tilde{\phi}) = |\#_0(\tilde{\phi}) - \#_1(\tilde{\phi})| = |2^n - \#_1(\phi) - \#_1(\phi) - 2^n| = 2\#_1(\phi).$$

Solving the AbsSAT instance $\tilde{\phi}(x_1, \dots, x_n, y)$ gives $S(\tilde{\phi})$, from which $\#_1(\phi)$ can be found. Therefore, AbsSAT is #P-hard. \square

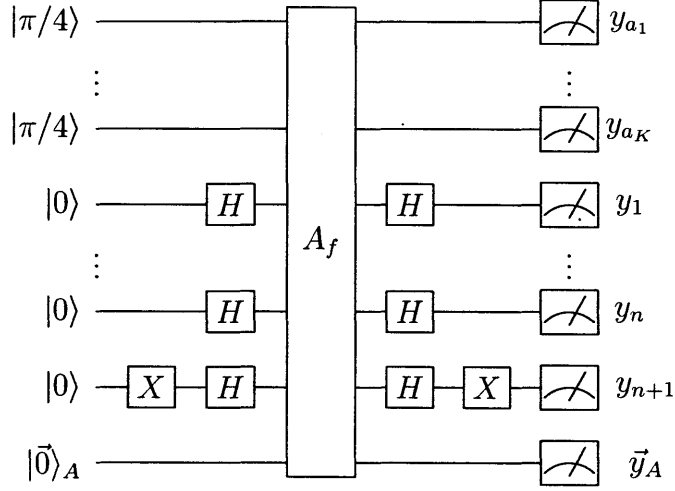
Theorem 39. Let $\nu = (\text{IN}(\text{PROD}), \text{NONADAPT}, \text{OUT}(\text{BITS}))$. Then the STR(n)-simulation of \mathcal{C}_ν is #P-hard.

Proof. Assume that there exists an efficient STR(n)-simulation S of \mathcal{C}_ν . We'll use S to construct an efficient algorithm for AbsSAT: On input $f : \{0, 1\}^n \rightarrow \{0, 1\}$, given as a 3-CNF formula with N clauses, where $n = O(N)$, construct a quantum circuit Q_f , consisting of only the basic Clifford gates and T gates, that acts on the following computational basis states as follows: (See Lemma 52 for the details of such a construction)

$$Q_f |x_1, \dots, x_n, 0\rangle |\vec{0}\rangle_A = |x_1, \dots, x_n, f(x_1, \dots, x_n)\rangle |\vec{0}\rangle_A.$$

Let K be the number of T gates in Q_f . For the j th T gate (acting on the l_j th line), for $j = 1, \dots, K$, introduce an ancilla line a_j , and replace the T gate with the CNOT gate CX_{l_j, a_j} . Call the resulting circuit A_f . It is straightforward to check that if each ancilla wire is initialized to the state $|\pi/4\rangle$, and measured at the end of the computation, and if the measurement outcomes are $0 \dots 0$, then the non-ancilla registers of A_f would implement Q_f . Hence, ignoring the ancilla registers, for the above measurement outcomes, we have $A_f : |x_1, \dots, x_n, y\rangle |\vec{0}\rangle_A \mapsto |x_1, \dots, x_n, y \oplus f(x_1, \dots, x_n)\rangle |\vec{0}\rangle_A$.

Let M_f be the following circuit:



If we postselect on the outcomes $y_{a_1} \dots y_{a_K} = 0 \dots 0$ for the ancilla registers, the nonancilla registers evolve as follows:

$$\begin{aligned}
|0 \dots 0, 0\rangle |\vec{0}\rangle_A &\rightarrow |0 \dots 0, 1\rangle |\vec{0}\rangle_A \\
&\rightarrow \frac{1}{\sqrt{2^{n+1}}} \sum_x |x\rangle (|0\rangle - |1\rangle) |\vec{0}\rangle_A \\
&\rightarrow \frac{1}{\sqrt{2^{n+1}}} \sum_x |x\rangle (|f(x)\rangle - |1 \oplus f(x)\rangle) |\vec{0}\rangle_A \\
&= \frac{1}{\sqrt{2^{n+1}}} \sum_x (-1)^{f(x)} |x\rangle (|0\rangle - |1\rangle) |\vec{0}\rangle_A \\
&\rightarrow \frac{1}{2^n} \sum_{xy} (-1)^{f(x)+x \cdot y} |y\rangle |1\rangle |\vec{0}\rangle_A \\
&\rightarrow \frac{1}{2^n} \sum_{xy} (-1)^{f(x)+x \cdot y} |y\rangle |0\rangle |\vec{0}\rangle_A.
\end{aligned}$$

Hence, the conditional probability of obtaining the all-zero string given that the ancilla measurements also reveal the all-zero string is

$$\Pr(0_1 \dots 0_{n+1}, \vec{0}_A | 0_{a_1} \dots 0_{a_K}) = \left| \frac{1}{2^n} \sum_x (-1)^{f(x)} \right|^2.$$

But the LHS of the above expression is equal to

$$\Pr(0_1 \dots 0_{n+1}, \vec{0}_A | 0_{a_1} \dots 0_{a_K}) = \frac{\Pr(0_{a_1} \dots 0_{a_K}, 0_1 \dots 0_{n+1}, \vec{0}_A)}{\Pr(0_{a_1} \dots 0_{a_K})}.$$

Now, $\Pr(0_{a_1} \dots 0_{a_K}) = 1/2^K$, since each ancilla bit has a probability of 1/2 of being measured zero.

Simplifying the above expressions, we get

$$\left| \sum_x (-1)^{f(x)} \right| = 2^{n+K/2} \sqrt{\Pr(0_{a_1} \dots 0_{a_K}, 0_1 \dots 0_{n+1}, \vec{0}_A)}.$$

But $\Pr(0_{a_1} \dots 0_{a_K}, 0_1 \dots 0_{n+1}, \vec{0}_A)$ is a joint outcome probability, and hence can be obtained by running S on $\langle M_f, 00 \dots 0 \rangle$. (The input to S is valid since M_f is a nonadaptive Clifford circuit with product state inputs.) Hence, the procedure given is an efficient algorithm for AbsSAT. Since AbsSAT is #P-hard, this implies that \mathcal{C}_ν is #P-hard as well. \square

3.5.3 Proof of Theorem 40: Strong(n) simulation of adaptive Clifford circuits with computational basis inputs and outputs

Theorem 40. Let $\nu = (\text{IN}(\text{BITS}), \text{ADAPT}, \text{OUT}(\text{BITS}))$. Then the STR(n)-simulation of \mathcal{C}_ν is #P-hard.

Proof. Assume that there exists an efficient STR(n)-simulation S of \mathcal{C}_ν . We'll use S to construct an efficient algorithm M for #SAT, i.e. given as input a 3-CNF formula $f : \{0, 1\}^n \rightarrow \{0, 1\}$, our goal is to find $\#f = \sum_x f(x)$.

$M =$ "On input $f : \{0, 1\}^n \rightarrow \{0, 1\}$, given as a 3-CNF formula,

1. Construct a classical circuit C_f consisting of only Toffoli gates that acts on the following computational basis states as follows: (see Lemma 51 for the details

of this construction)

$$C_f(x_1, \dots, x_n, 1, \vec{1}_A) = (x_1, \dots, x_n, f(x_1, \dots, x_n), \vec{1}_A).$$

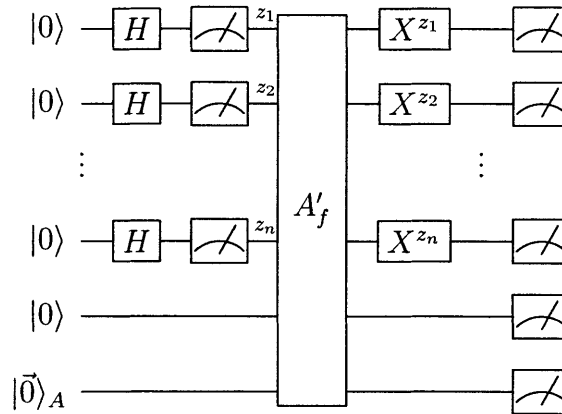
2. Simulate C_f with a Clifford circuit from \mathcal{A} : replace each Toffoli gate $T_{abc}(x, y, z) = (x, y, z \oplus xy)$ acting on lines a, b, c with $(CX_{bc})^x M_a(x)$. Call the resulting quantum circuit A_f . The circuit A_f acts on computational basis states as follows:

$$A_f |x_1, \dots, x_n, 1\rangle |\vec{1}\rangle_A \rightarrow |x_1, \dots, x_n, f(x_1, \dots, x_n)\rangle |\vec{1}\rangle_A.$$

By applying X gates (expressed as $X = HS^2H$) to the appropriate lines at the input and output of A_f , let A'_f be the circuit that acts on computational basis states as follows:

$$A'_f |x_1, \dots, x_n, 0\rangle |\vec{0}\rangle_A \rightarrow |x_1, \dots, x_n, f(x_1, \dots, x_n)\rangle |\vec{0}\rangle_A.$$

3. Let G_f be the following circuit:



4. Feed $\langle G_f, 00 \dots 01\vec{0}_A \rangle$ into S to find $p = p(00 \dots 01\vec{0}_A)$, the probability that the output is $00 \dots 01\vec{0}_A$.
5. Output $\#f = 2^n p$.

A straightforward calculation shows that the output of G_f on input $|00 \dots 0\rangle |\vec{0}\rangle_A$

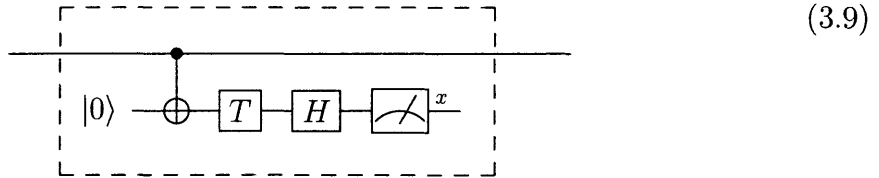
is $|0, \dots, 0, f(z)\rangle|\vec{0}\rangle_A$ if the intermediate measurement results are $z = z_1 \dots z_n$. Hence,

$$\begin{aligned}
 p &= p(0 \dots 0, 1, \vec{0}_A) = \sum_z p(0 \dots 0, 1, \vec{0}_A | z_1 \dots z_n) p(z_1 \dots z_n) \\
 &= \sum_z |\langle 0 \dots 01, \vec{0}_A | 0 \dots 0, f(z), \vec{0}_A \rangle|^2 \frac{1}{2^n} \\
 &= \frac{1}{2^n} \sum_x f(x).
 \end{aligned} \tag{3.8}$$

Hence, the output of M is $2^n p = \#f$. \square

3.5.4 Proof of Theorem 46: Weak simulation of nonadaptive Clifford circuits with computational basis inputs and product outputs

We follow a proof similar to that given in [48] that shows that if IQP circuits can be efficiently classically simulated in the weak sense, then the polynomial hierarchy collapses. Recall that the T gate is given by $T = \text{diag}(1, e^{i\pi/4})$. We first consider the following gadget \mathcal{G} :



Lemma 41.

$$\mathcal{G} : |\psi\rangle \mapsto \begin{cases} T|\psi\rangle & \text{if } x = 0 \\ ZT|\psi\rangle & \text{if } x = 1. \end{cases}$$

Proof. Applying the unitary gates in the circuit to the state $|\psi\rangle|0\rangle$ gives

$$\frac{1}{\sqrt{2}} [(T|\psi\rangle)|0\rangle + (ZT|\psi\rangle)|1\rangle].$$

Hence, we get the desired states when the ancilla wire is measured. \square

From the proof of Lemma 41, we note that the measurement outcomes $x = 0, 1$ occur with an equal probability. Note that if $x = 0$, then \mathcal{G} would have implemented the T gate.

Lemma 42. Let Q be an arbitrary quantum circuit comprising the basic Clifford gates and T gates. Let $\nu = (\text{IN}(\text{BITS}), \text{NONADAPT}, \text{OUT}(\text{PROD}))$. Then Q with postselection can be weakly simulated by \mathcal{C}_ν with postselection.

Proof. We first show how we can simulate the circuit Q using circuits from \mathcal{C}_ν with postselection. For each T gate in Q , we replace it by the gadget \mathcal{G} defined above. If the number of T gates is s , then this procedure produces a new circuit C with s new lines. Now, note that the new circuit C belongs to the class \mathcal{C}_ν since the HT gates together with the computational basis measurements implement a product measurement. Now, if we postselect on outcome 0 for all the measurements in the new lines, then each gadget \mathcal{G} would implement the T gate. Hence, \mathcal{C}_ν with postselection would weakly simulate Q . Now, since we have the resource of postselection, it follows that Q with postselection can be weakly simulated by \mathcal{C}_ν with postselection. \square

We now make the following definition (recall notation in Eq. (3.6): we use similar notation for conditional probabilities) to capture the power of subsets of Clifford computational tasks with postselection.

Definition 43. ($\text{post}\mathcal{C}_\nu\mathcal{P}$) Let \mathcal{C}_ν be a subset of Clifford computational tasks. A language $L \in \text{post}\mathcal{C}_\nu\mathcal{P}$ if there exist an error tolerance $0 < \epsilon < \frac{1}{2}$ and a uniform family $\{C_w\}_w$ of circuits in \mathcal{C}_ν with $n + p(n)$ lines (call these lines $l_1, \dots, l_n, a_1, \dots, a_p$, where $p = p(n)$), where $n = |w|$ and p is some polynomial, such that

$$\begin{aligned}
& p_{C_w}^{\{a_1, \dots, a_N\}}(00 \dots 0) > 0, \\
& w \in L \implies p_{C_w}^{\{l_1\}|\{a_1, \dots, a_N\}}(1|00 \dots 0) \geq 1 - \epsilon, \\
& w \notin L \implies p_{C_w}^{\{l_1\}|\{a_1, \dots, a_N\}}(0|00 \dots 0) \geq 1 - \epsilon.
\end{aligned} \tag{3.10}$$

We will use the definition of postBQP given in [48], which allows for multiple postselected lines. Note that this is equivalent to the definition given in [2] where

postBQP was introduced, which allows for only single lines. We now show that the class just defined is equal to postBQP.

Lemma 44. Let $\nu = (\text{IN}(\text{BITS}), \text{NONADAPT}, \text{OUT}(\text{PROD}))$. Then, $\text{post}\mathcal{C}_\nu\text{P} = \text{postBQP}$.

Proof. The forward direction is immediate, since extended Clifford circuits are a special case of general quantum circuits. To prove the backward direction, let $L \in \text{postBQP}$. Then there exist an error tolerance $0 < \epsilon < \frac{1}{2}$ and a uniform family $\{Q_w\}_w$ of quantum circuits consisting of the basic Clifford gates and T gates with $n + p(n)$ lines (call these lines $l_1, \dots, l_n, b_1, \dots, b_N$, where $p = p(n)$), where $n = |w|$ and p is some polynomial, such that

$$p_{Q_w}^{\{b_1, \dots, b_N\}}(00 \dots 0) > 0,$$

$$w \in L \implies p_{Q_w}^{\{l_1\}|\{b_1, \dots, b_p\}}(1|00 \dots 0) \geq 1 - \epsilon,$$

$$w \notin L \implies p_{Q_w}^{\{l_1\}|\{b_1, \dots, b_N\}}(0|00 \dots 0) \geq 1 - \epsilon.$$

By Lemma 42, for each Q_w , there exists an extended Clifford circuit $C_w \in \mathcal{C}_\nu$ that, with postselection, simulates Q_w with postselection. If s is the number of T gates in Q_w , then C_w has $n + p(n) + s$ lines. Postselecting on the last $p(n) + s$ lines, it follows that the set of circuits $\{C_w\}$ satisfies the definition given for $\text{post}\mathcal{C}_\nu\text{P}$. Hence, $L \in \text{post}\mathcal{C}_\nu\text{P}$. \square

Lemma 45. Let $\nu = (\text{IN}(\text{BITS}), \text{NONADAPT}, \text{OUT}(\text{PROD}))$. If $\mathcal{C}_\nu \in \text{PWK}(n)$, then $\text{post}\mathcal{C}_\nu\text{P} \subseteq \text{postBPP}$.

Proof. Let $L \in \text{post}\mathcal{C}_\nu\text{P}$. Then there exist an error tolerance $0 < \epsilon < \frac{1}{2}$ and a uniform family $\{C_w\}_w$ of circuits in \mathcal{C}_ν with $n + p(n)$ lines (call these lines $l_1, \dots, l_n, a_1, \dots, a_p$, where $p = p(n)$), where $n = |w|$ and p is some polynomial, such that Eq. (3.10) holds.

But $\mathcal{C}_\nu \in \text{PWK}(n)$. Hence, for all circuits $Q_w \in \mathcal{C}_\nu$, there exists a classical randomized circuit C_w with $n + p$ lines such that

$$p_{Q_w}^{\{l_1, \dots, l_n, a_1, \dots, a_p\}}(y) = p_{C_w}^{\{l_1, \dots, l_n, a_1, \dots, a_p\}}(y).$$

For any subsets $I, J \subseteq [n]$ of lines, similar relations hold for marginal probabilities and conditional probabilities: $p_{Q_w}^I(y) = p_{C_w}^I(y)$ and $p_{Q_w}^{I|J}(y|z) = p_{C_w}^{I|J}(y|z)$. This implies that

$$p_{Q_w}^{\{l_1\}\{a_1, \dots, a_p\}}(1|00 \dots 0) = p_{C_w}^{\{l_1\}\{a_1, \dots, a_p\}}(1|00 \dots 0),$$

and hence Q_w obey Eq. (3.10). This implies that $L \in \text{postBPP}$. Therefore, $\text{postC}_\nu\text{P} \subseteq \text{postBPP}$. \square

Theorem 46. Let $\nu = (\text{IN}(\text{BITS}), \text{NONADAPT}, \text{OUT}(\text{PROD}))$. If $C_\nu \in \text{PWK}(n)$, then PH collapses to the third level.

Proof. By Lemmas 44 and 45, if $C_\nu \in \text{PWK}(n)$, then $\text{postBPP} \supseteq \text{postC}_\nu\text{P} = \text{postBQP}$. By Corollary 23,

$$\text{PH} \subseteq \Sigma_3^p,$$

i.e. PH collapses to the third level. \square

3.5.5 Proof of Theorem 48: Strong(n) simulation of nonadaptive Clifford circuits with product inputs and computational basis outputs

Consider the proof of Theorem 39. Note that the circuit M_f is unitary. Hence, an even stronger result than Theorem 39 is true: if we replaced nonadaptive circuits with unitary ones (call this UNITARY), the simulation complexity is still $\#\text{P}$ -hard. In other words,

Lemma 47. Let $\nu = (\text{IN}(\text{PROD}), \text{UNITARY}, \text{OUT}(\text{BITS}))$. Then the STR(n)-simulation of C_ν is $\#\text{P}$ -hard.

The STR(n)-simulation of C_ν is equivalent to the following problem:

Input: $\langle T, y \rangle$, where $T = (|x\rangle, B, |\alpha\rangle)$, B is a unitary circuit, $x, y \in \{0, 1\}^n$, $\alpha = \alpha_1 \dots \alpha_n$ and each $|\alpha_i\rangle \in \mathbb{C}_2$.

Output: $p_T(y) = |\langle \alpha_1^{y_1} \dots \alpha_n^{y_n} | B|x \rangle|^2$.

Now, let $\mu = (\text{IN}(\text{BITS}), \text{UNITARY}, \text{OUT}(\text{PROD}))$, then the $\text{STR}(n)$ -simulation of \mathcal{C}_μ is equivalent to the following problem:

Input: $\langle T', y \rangle$, where $T' = (|\alpha_1^{y_1} \dots \alpha_n^{y_n}\rangle, B^\dagger, \{I_2\}_i)$, B is a unitary circuit, $y \in \{0, 1\}^n$ and I_2 is the 2×2 identity gate.

Output: $p_{T'}(x) = |\langle x | B^\dagger |\alpha_1^{y_1} \dots \alpha_n^{y_n}\rangle|^2 = |\langle \alpha_1^{y_1} \dots \alpha_n^{y_n} | B | x \rangle|^2 = p_T(y)$.

Since both problem instances can be transformed easily to each other, and since both problems involve calculating the same quantity, we conclude that the $\text{STR}(n)$ -simulation of \mathcal{C}_μ is also $\#\text{P}$ -hard. If it is $\#\text{P}$ -hard to simulate this class of unitary circuits, then it must be $\#\text{P}$ -hard to simulate the same class but with unitary circuits replaced by nonadaptive circuits. Therefore, we obtain the following theorem:

Theorem 48. Let $\nu = (\text{IN}(\text{BITS}), \text{NONADAPT}, \text{OUT}(\text{PROD}))$. Then the $\text{STR}(n)$ -simulation of \mathcal{C}_ν is $\#\text{P}$ -hard.

3.5.6 Proof of Theorem 49: Strong(1) simulation of nonadaptive Clifford circuits with product inputs and outputs

Theorem 49. Let $\nu = (\text{IN}(\text{PROD}), \text{NONADAPT}, \text{OUT}(\text{PROD}))$. Then, $\mathcal{C}_\nu \in \text{PSTR}(1)$.

Proof. We use the following notation: for any single-qubit operator O , let $O_1 = O \otimes I \otimes \dots \otimes I$. Given a Clifford computational task $T = (|\alpha_1 \dots \alpha_n\rangle, B, \{U, I, \dots, I\}) \in \mathcal{C}_\nu$, and a bit $i \in \{0, 1\}$, we shall describe an algorithm to compute $p_i := p_T^{\{1\}}(i)$. WLOG, B is a unitary circuit.

Since $p_0 + p_1 = 1$, it suffices to be able to calculate $p_0 - p_1$ efficiently. By Born's rule, this is given by

$$p_0 - p_1 = \langle \alpha | B^\dagger (UZU^\dagger)_1 B | \alpha \rangle. \quad (3.11)$$

Since the Pauli matrices $\{\sigma^i\}_i$ form a basis for the set of 2×2 matrices, we can write

$$U = \sum_{i=0}^3 a_i \sigma^i,$$

for some $a_i \in \mathbb{C}$. Hence,

$$UZU^\dagger = \sum_{i,j=0}^3 a_i \bar{a}_j \sigma^i Z \sigma^j.$$

But $\sigma^i Z \sigma^j$ is a Pauli operator. Since the basic Clifford gates map Pauli operators to Pauli operators,

$$B^\dagger (\sigma^i Z \sigma^j)_1 B = \gamma_{ij} P_1^{ij} \otimes \dots \otimes P_n^{ij}.$$

Putting this into Eq. (3.11), we get an expression for $p_0 - p_1$.

$$\begin{aligned} p_0 - p_1 &= \sum_{i,j=0}^3 a_i \bar{a}_j \gamma_{ij} \langle \alpha_1 \dots \alpha_n | P_1^{ij} \otimes \dots \otimes P_n^{ij} | \alpha_1 \dots \alpha_n \rangle \\ &= \sum_{i,j=0}^3 a_i \bar{a}_j \gamma_{ij} \prod_{k=1}^n \langle \alpha_k | P_k^{ij} | \alpha_k \rangle. \end{aligned} \quad (3.12)$$

We now analyze the running time of our algorithm. Computing $\gamma_{ij} P_1^{ij} \otimes \dots \otimes P_n^{ij}$ takes $O(n^2)$ -time. The formula given in Eq. (3.12) involves a sum of 9 terms. Each term involves computing n expectation values of 2×2 matrices. Hence, this step takes $O(n)$ -time. Overall, the algorithm runs in $O(n^2) = O(N^2)$ -time, where N is the number of gates in the circuit (which we assumed to contain no extraneous lines). Hence, $\mathcal{C}_\nu \in \text{PSTR}(1)$.

□

3.5.7 Proof of Theorem 50: Weak(1) simulation of adaptive Clifford circuits with computational basis inputs and product outputs

Theorem 50. Let $\nu = (\text{IN}(\text{BITS}), \text{ADAPT}, \text{OUT}(\text{PROD}))$. Then $\mathcal{C}_\nu \in \text{PWK}(1)$.

Proof. This is a special case of the results in Section VIIC of [8], which showed that an $\text{IN}(\text{BITS}), \text{NONADAPT}, \text{OUT}(\text{BITS})$ circuit containing d non-Clifford gates, where each gate acts on at most b qubits, can be classically simulated in the $\text{WEAK}(1)$ sense in $O(4^{2bd}n + n^2)$ -time. In our case, the circuits in \mathcal{C}_ν can be thought of as containing

exactly one non-Clifford gate on the first wire just before the computational-basis measurement. Hence, $d = b = 1$, which implies that the algorithm runs in $O(n^2)$ -time. This concludes the proof that $C_\nu \in \text{PWK}(1)$. \square

3.5.8 Constructing circuits for 3-CNF formulas

In the proof of Theorem 39, we used the fact that given a 3-CNF formula $f : \{0, 1\}^n \rightarrow \{0, 1\}$, we can efficiently construct a quantum circuit A_f comprising only the basic Clifford operations and T gates, which acts on the following computational basis states as follows:

$$A_f |x\rangle |0\rangle |0\rangle_A = |x\rangle |f(x)\rangle |0\rangle_A, \quad (3.13)$$

where $x \in \{0, 1\}^n$ and $|\cdot\rangle_A$ is an ancilla register of size $O(n)$.

A similar fact was used in the proof of Theorem 40, namely that given a 3-CNF formula $f : \{0, 1\}^n \rightarrow \{0, 1\}$, we can efficiently construct a classical circuit C_f comprising only Toffoli gates, which acts on the following computational basis states as follows:

$$C_f(x, 1, 1_A) = (x, f(x), 1_A), \quad (3.14)$$

where $x \in \{0, 1\}^n$ and A is an ancilla register of size $O(n)$.

Note that in both circuits C_f and A_f , we do not allow for the addition of more ancilla lines or for the discarding of any bit or qubits. This is because for the notion of $\text{STR}(n)$ simulation, all bit or qubit lines have to be accounted for. Hence, we make explicit the reference to the ancilla registers A . In this section, we present the details of the above constructions.

Recall the definition of a 3-CNF formula given in Eq. (3.7). As above, we assume that every variable x_1, \dots, x_n appears in the formula for f , so that $n \leq 3N$, i.e. $n = O(N)$.

3.5.9 Constructing C_f

We show that we can implement the function f using Toffoli gates alone. We denote the action of the Toffoli gate on lines i, j, k with inputs a, b, c by

$$\text{Tof}_{ijk}(\dots, a, \dots, b, \dots, c, \dots) = (\dots, a, \dots, b, \dots, c \oplus a \cdot b, \dots).$$

We use subscripts at the end to indicate a ‘marginalizing out’ of the values of all other wires, for example,

$$\text{Tof}_{143}(a, b, c, d, e)_{235} = (a, b, c \oplus a \cdot d, d, e)_{235} = (b, c \oplus a \cdot d, e).$$

Lemma 51. Let f be a 3-CNF formula of the form given by Eq. (3.7) with n variables and N clauses, where $n = O(N)$. Then there exists a classical circuit C_f consisting of $O(N)$ Toffoli gates on $n + 1 + s(N)$ lines, for some $s(N) = O(N)$ (where we do not allow for the addition of bit lines or the discarding of any bits), such that

$$C_f(x_1, \dots, x_n, \underbrace{1, \dots, 1}_{s(N)}) = (x_1, \dots, x_n, f(x_1, \dots, x_n), \underbrace{1, \dots, 1}_{s(N)}). \quad (3.15)$$

Remark. The ancilla bits are initialized to 1 instead of 0. This is because the Toffoli gate is universal only if we have the ability to prepare the state 1. In particular, if the inputs were always just 0’s, then it would not be possible to create the state 1. On the other hand, we can prepare 0 from 1 since the target bit of $\text{Tof}(1, 1, 1)$ is 0.

Proof. We first show how to compute f on the input (x_1, \dots, x_n) using AND, OR, NOT, COPY and SWAP gates. Let k_i and \bar{k}_i be the number of times x_i and \bar{x}_i , respectively, appear as literals in the formula for f , i.e. $\sum_i (k_i + \bar{k}_i) = 3N$. By assumption, every variable x_1, \dots, x_n appears in the formula for f , so $k_i + \bar{k}_i > 0$ for all i .

For each i , if $k_i > 0$, apply the COPY gate $k_i - 1$ times to x_i and the COPY gate followed by the NOT gate \bar{k}_i times to x_i . Otherwise, if $k_i = 0$ (i.e. $\bar{k}_i > 0$), apply the

NOT gate followed by the COPY gate \bar{k}_i times to x_i . This creates the state

$$\left(\underbrace{x_1, \dots, x_1}_{k_1}, \dots, \underbrace{x_n, \dots, x_n}_{k_n}, \underbrace{\bar{x}_1, \dots, \bar{x}_1}_{\bar{k}_1}, \dots, \underbrace{\bar{x}_n, \dots, \bar{x}_n}_{\bar{k}_n} \right).$$

Note that the number of gates that the above procedure involves is

$$\sum_{k_i > 0} [(k_i - 1) + 2\bar{k}_i] + \sum_{k_i = 0} 2\bar{k}_i \leq 2 \sum_i (k_i + \bar{k}_i) = 6N.$$

Applying the SWAP gate up to $3N$ times to the above state, we get the state

$$(a_{11}, a_{12}, a_{13}, a_{21}, \dots, a_{N1}, a_{N2}, a_{N3}). \quad (3.16)$$

We now apply the OR and AND gates according to the formula in Eq. (3.7) to get $(a_{11} \vee a_{12} \vee a_{13}) \wedge (a_{21} \vee a_{22} \vee a_{23}) \wedge \dots \wedge (a_{N1} \vee a_{N2} \vee a_{N3})$. This involves a total of $2N$ OR gates and $N - 1$ AND gates. Hence, the resulting circuit B_f , whose number of gates is bounded above by $6N + 3N + 2N + N - 1 = O(N)$, computes:

$$B_f(x_1, \dots, x_n) = f(x_1, \dots, x_n).$$

Note that the maximum width of B_f , which occurs when the state is given by Eq. (3.16), is $3N$.

We now use the fact that the Toffoli gate together with the ability to prepare the ancilla state 1 is universal for classical computing. In particular, they simulate the above gates as follows:

$$\begin{aligned} \neg x &= \text{Tof}_{123}(1, 1, x)_3, \\ x \wedge y &= [\text{Tof}_{123} \circ \text{Tof}_{453}(x, y, 1, 1, 1)]_3, \\ \text{COPY}(x) &= [\text{Tof}_{123} \circ \text{Tof}_{243}(x, 1, 1, 1)]_{13}, \\ x \vee y &= [\text{Tof}_{453} \circ \text{Tof}_{123} \circ \text{Tof}_{453} \circ \text{Tof}_{342} \circ \text{Tof}_{341}(x, y, 1, 1, 1)]_3, \\ \text{SWAP}(x, y) &= [\text{Tof}_{123} \circ \text{Tof}_{321} \circ \text{Tof}_{123}(x, 1, y)]_{13}. \end{aligned} \quad (3.17)$$

We append ancilla lines initialized to 1 to B_f , and replace all the gates in B_f by Toffoli gates according to the rules in Eq. (3.17), and apply additional swap gates (implemented by Toffoli gates) so that the first output of the circuit is $f(x_1, \dots, x_n)$. Note that we do not discard any bits. Each of the replacements increases the number of ancilla lines by at most 3 and the number of gates by at most 4. Hence, both the total number of lines $a(N)$ and the number of Toffoli gates in the new circuit B'_f are still $O(N)$. The action of B'_f on the computational basis states is given by:

$$B'_f(x_1, \dots, x_n, \vec{1}) = (f(x_1, \dots, x_n), j_2, \dots, j_{a(N)}),$$

where $(j_2, \dots, j_{a(N)})$ are junk bits.

We now make use of the *uncomputation trick* to reset the junk bits to 1. Since the Toffoli gates are their own inverse, the inverse of B'_f is obtained by applying the gates in B'_f in the reverse order. Consider the circuit B''_f that is formed as follows: first apply B'_f to $(x_1, \dots, x_n, \vec{1})$. Introduce a new ancilla line, called a , initialized to 0. Next, apply the CNOT gate CX_{1a} . Finally, apply B'^{-1}_f to the first $a(N)$ bits to reset them back to $(x_1, \dots, x_n, \vec{1})$. A circuit diagram for the above steps is shown in Figure 3-3. This gives

$$B''_f(x_1, \dots, x_n, \vec{1}, 0) = (x_1, \dots, x_n, \vec{1}, f(x_1, \dots, x_n)).$$

To get the required circuit C_f , we need to perform three more simple steps. First, the ancilla bit in the last register has to start from 1 instead of 0. This can be achieved by applying a NOT gate (implemented by the Toffoli gate and ancillas initialized to 1) to 0. Second, the CNOT gate has to be simulated by a Toffoli gate. This may be achieved by using the fact that

$$CX(a, b)_{12} = \text{Tof}_{123}(a, 1, b)_{13}.$$

Third, the output has to be of the form (3.15). This is obtained by applying swap gates at the end of the circuit. These steps add at most a constant number of gates

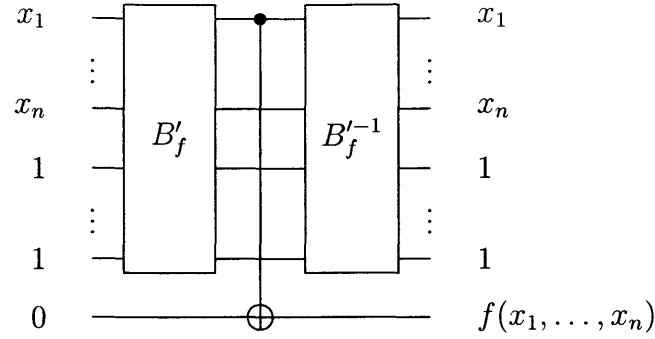


Figure 3-3: Uncomputation trick, in which the output bits, except for those in the target register, are reset to their input values. The state evolves as follows: $(x, 1, \dots, 1, 0) \rightarrow (f(x), j_2, \dots, j_{a(N)}, 0) \rightarrow (f(x), j_2, \dots, j_{a(N)}, f(x)) \rightarrow (x, 1, \dots, 1, f(x))$, where $x = x_1 \dots x_n$.

and a constant number of ancilla bits. Hence, the resulting circuit C_f has $O(N)$ gates acting on $O(N)$ lines.

□

3.5.10 Constructing Q_f

We now show how we can convert C_f to a circuit Q_f that involves only the basic Clifford gates and the T gate.

Lemma 52. Let f be a 3-CNF formula of the form given by Eq. (3.7) with n variables and N clauses, where $n = O(N)$. Then there exists a quantum circuit Q_f consisting of $O(N)$ basic Clifford gates and T gates on $n + 1 + s(N)$ lines (where we do not allow for the addition of qubit lines or the discarding of any qubits), such that

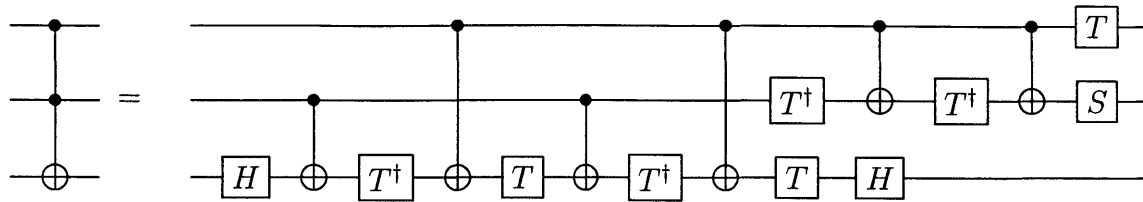
$$Q_f |x_1, \dots, x_n, 0, 0^{s(N)}\rangle = |x_1, \dots, x_n, f(x_1, \dots, x_n), 0^{s(N)}\rangle, \quad (3.18)$$

for some $s(N) = O(N)$.

Proof. Using Lemma 51, we have a circuit C_f comprising $O(N)$ Toffoli gates satisfying

$$C_f (x_1, \dots, x_n, 1, 1^{s(N)}) = (x_1, \dots, x_n, f(x_1, \dots, x_n), 1^{s(N)}).$$

Using the construction presented in [180], we express each Toffoli gate in terms of the basic Clifford gates, T and T^\dagger gates, as follows:



Since $T^8 = 1$, we replace each T^\dagger gate above by T^7 . These replacements increase the number of gates by a constant factor, and hence the total number of gates in the new circuit is still $O(N)$. Finally, we insert X (expressed as $X = HS^2H$) gates at the start and end of the circuit so that the ancilla lines start and terminate in the state $|0\rangle$. This gives us a quantum circuit obeying Eq. (3.18) with $O(N)$ wires and $O(N)$ gates.

□

3.6 Concluding remarks

We have demonstrated how the classical simulation complexities of extended Clifford circuits change when various ingredients in the circuits are varied. It would be interesting to study other ingredients of Clifford circuits as well, e.g., mixed input states [8], states (as well as transformations and measurements) with positive Wigner representations [166, 225], and non-commutative extensions like XS-stabilizer states [179]. Most of these extensions have previously been considered separately, and it will be fruitful to study the classical simulation complexities of computational tasks with these different combinations of ingredients.

Chapter 4

Conjugated Clifford circuits

By the Gottesman-Knill Theorem, the computational power of Clifford circuits is limited, as they can be efficiently simulated on a classical computer. In this chapter, we show that in contrast, “conjugated Clifford circuits” (CCCs)—where one additionally conjugates every qubit by the same one-qubit gate U —can perform hard sampling tasks. In particular, we fully classify the computational power of CCCs by showing that essentially any non-Clifford conjugating unitary U can give rise to sampling tasks which cannot be efficiently classically simulated to constant multiplicative error, unless the polynomial hierarchy collapses. Furthermore, by standard techniques, this hardness result can be extended to allow for the more realistic model of constant additive error, under a plausible complexity-theoretic conjecture. This work can be seen as progress towards classifying the computational power of all restricted quantum gate sets, and is based on joint work with Adam Bouland and Joseph F. Fitzsimons [36].

4.1 Overview of results

This chapter considers a new intermediate model of quantum computation which we call “conjugated Clifford circuits” (CCCs). In this model, we consider the power of quantum circuits whose gates are $(U^\dagger \otimes U^\dagger)(CZ)(U \otimes U)$, $U^\dagger H U$ and $U^\dagger S U$, where U is a fixed single-qubit gate. In other words, we consider the power of Clifford

circuits which are conjugated by an identical one-qubit gate U on each qubit. These gates manifestly perform a discrete subset of unitaries so this gate set is clearly not universal.

Although this transformation preserves the non-universality of the Clifford group, it is unclear if it preserves its computational power. The presence of generic conjugating unitaries (even the same U on each qubit, as in this model) breaks the Gottesman-Knill simulation algorithm [111], as the inputs and outputs of the circuit are not stabilizer states/measurements. Hence the intermediate states of the circuit are no longer efficiently representable by the stabilizer formalism. This, combined with prior results showing hardness for other modified versions of Clifford circuits [143, 148], leads one to suspect that CCCs may not be efficiently classically simulable. However prior to this work no hardness results were known for this model.

In this work, we confirm this intuition and provide two results in this direction. First, we provide a *complete classification* of the power of CCCs according to the choice of U . We do this by showing that *any* U which is not efficiently classically simulable by the Gottesman-Knill theorem suffices to perform hard sampling problems with CCCs¹. That is, for generic U , CCCs cannot be efficiently classically simulated to constant multiplicative error by a classical computer unless the polynomial hierarchy collapses. This result can be seen as progress towards classifying the computational complexity of restricted gate sets. Indeed, given a non-universal gate set G , a natural question is to classify the power of G when conjugated by the same one-qubit unitary U on each qubit, as this transformation preserves non-universality. Our work resolves this question for one of the most prominent examples of non-universal gate sets, namely the Clifford group. As few examples of non-universal gate sets are known², this closes one of the major gaps in our understanding of intermediate gate sets. Of course this does not complete the complexity classification of all gate sets, as there is no known classification of all possible non-universal gate sets. However it does make

¹More precisely, we show that any U that cannot be written as a Clifford times a Z -rotation suffices to perform hard sampling problems with CCCs. See Theorem 58 for the exact statement.

²The only examples to our knowledge are matchgates [142, 223], Clifford gates, diagonal gates, and subsets thereof.

progress towards this goal.

Second, we show that under an additional complexity-theoretic conjecture, classical computers cannot efficiently simulate CCCs to constant error in total variation distance. This is a more experimentally achievable model of error for noisy quantum computations. The proof of this result uses standard techniques introduced by Aaronson and Arkhipov [5], which have also been used in other models [32, 49, 50, 99, 165, 173].

This second result is interesting for two reasons. First, it means our results may have relevance to the empirical demonstration of quantum advantage (sometimes referred to as “quantum supremacy”) [7, 32, 188], as our results are robust to noise. Second, from the perspective of computational complexity, it gives yet another conjecture upon which one can base the supremacy of noisy quantum devices. As is the case with other quantum supremacy proposals [5, 49, 99, 165, 173], in order to show that simulation of CCCs to additive error still collapses the polynomial hierarchy, we need an additional conjecture stating that the output probabilities of these circuits are hard to approximate on average. Our conjecture essentially states that for most Clifford circuits V and most one-qubit unitaries U , it is $\#P$ -hard to approximate a constant fraction of the output probabilities of the CCC $U^{\otimes n}V(U^\dagger)^{\otimes n}$ to constant multiplicative error. We prove that this conjecture is true in the worst case – in fact, for all non-Clifford U , there exists a V such that some outputs are $\#P$ -hard to compute to multiplicative error. However, it remains open to extend this hardness result to the average case, as is the case with other supremacy proposals as well [5, 49, 99, 165, 173]. To the best of our knowledge our conjecture is independent of the conjectures used to establish other quantum advantage results such as boson sampling [5], Fourier sampling [99] or IQP [49, 50]. Therefore our results can be seen as establishing an alternative basis for belief in the advantage of noisy quantum devices over classical computation.

One final motivation for this work is that CCCs might admit a simpler fault-tolerant implementation than universal quantum computing, which we conjecture to be the case. It is well-known that many stabilizer error-correcting codes, such as the 5-qubit and 7-qubit codes [84, 155, 214], admit transversal Clifford operations [109].

That is, performing fault-tolerant Clifford operations on the encoded logical qubits can be done in a very simple manner – by simply performing the corresponding Clifford operation on the physical qubits. This is manifestly fault-tolerant, in that an error on one physical qubit does not “spread” to more than 1 qubit when applying the gate. In contrast, performing non-Clifford operations fault-tolerantly on such codes requires substantially larger (and non-transversal) circuits – and therefore the non-transversal operations are often the most resource intensive. The challenge in fault-tolerantly implementing CCCs therefore lies in performing the initial state preparation and measurement. Initial preparation of non-stabilizer states in these codes is equivalent to the challenge of producing magic states, which are already known to boost Clifford circuits to universality using adaptive Clifford circuits [44, 45] (in contrast our construction would only need non-adaptive Clifford circuits with magic states). Likewise, measuring in a non-Clifford basis would require performing non-Clifford one-qubit gates prior to fault-tolerant measurement in the computational basis. Therefore the state preparation/measurement would be the challenging part of fault-tolerantly implementing CCCs in codes with transversal Cliffords. It remains open if there exists a code with transversal conjugated Cliffords³ and easy preparation and measurement in the required basis. Such a code would not be ruled out by the Eastin-Knill Theorem [89], which states that the set of transversal gates must be discrete for all codes which correct arbitrary one qubit errors. Of course this is not the main motivation for exploring the power of this model – which is primarily to classify the space between BPP and BQP – but an easier fault-tolerant implementation could be an unexpected bonus of our results.

4.1.1 Proof techniques

To prove these results, we use several different techniques.

³Of course one can always “rotate” a code with transversal Clifford operations to obtain a code with transversal conjugated Cliffords. If the code previously had logical states $|0\rangle_L, |1\rangle_L$, then by setting the states $|0\rangle'_L = U_L^\dagger |0\rangle_L$ and $|1\rangle'_L = U_L^\dagger |1\rangle_L$, one obtains a code in which the conjugated Clifford gates (conjugated by U) are transversal. However having the ability to efficiently fault-tolerantly prepare $|0\rangle_L$ in the old code does not imply the same ability to prepare $|0\rangle'_L$ in the new code.

4.1.1.1 Proof Techniques: classification of exact sampling hardness

To prove exact (or multiplicative) sampling hardness for CCCs for essentially all non-Clifford U , we use the notion of postselection introduced by Aaronson [2]. As described in Chapter 2.6.6, postselection is the (non-physical) ability to discard all runs of the computation which do not achieve some particular outcomes. Our proof works by showing that postselecting such circuits allows them to perform universal quantum computation. Hardness then follows from known techniques [2, 5, 48] (see Chapter 2.6.6).

One technical subtlety that we face in this proof, which is not present in other results, is that our postselected gadgets perform operations which are not closed under inversion. This means that one cannot use the Solovay-Kitaev theorem (see Theorem 2) to change quantum gate sets [78]. This is a necessary step in the proof that $\text{PostBQP} = \text{PP}$ [2], which is a key part of the hardness proof (see [37]). Fortunately, it turns out that we can get away without inverses due to a recent inverse-free Solovay-Kitaev theorem of Sardharwalla et al. [198], which removes the needs for inverses if the gate set contains the Paulis. Our result would have been much more difficult to obtain without this prior result. To our knowledge this is the first application of their result to structural complexity.

A further difficulty in the classification proof is that the postselection gadgets we derive do not work for all non-Clifford U . In general, most postselection gadgets give rise to non-unitary operations, and for technical reasons we need to work with unitary postselection gadgets to apply the results of [198]. Therefore, we instead use several different gadgets which cover different portions of the parameter space of U 's. Our initial proof of this fact used a total of seven postselection gadgets found by hand. We later simplified this to two postselection gadgets by conducting a brute-force search for suitable gadgets using Christopher Granade and Ben Criger's QuaEC package [114]. We include this simplified proof in this chapter.

A final difficulty that one often faces with postselected universality proofs is that one must show that the postselection gadgets boost the original gate set to univer-

sality. In general this is a nontrivial task; there is no simple test of whether a gate set is universal, though some sufficient (but not necessary) criteria are known [199]. Prior gate set classification theorems have solved this universality problem using representation theory [35, 199] or Lie theory [37, 181]. However, in our work we are able to make use of a powerful fact: the Clifford group plus *any* non-Clifford unitary is universal. This follows from results of Nebe, Rains and Sloane [176, 177, 218] classifying the invariants of the Clifford group⁴. As a result, our postselected universality proofs are much simpler than in other gate set classification theorems.

4.1.1.2 Proof techniques: additive error

To prove hardness of simulation to additive error, we follow the techniques of [5, 49, 99, 173]. In these works, to show hardness of sampling from some probability distribution with additive error, one combines three different ingredients. The first is anti-concentration – showing that for these circuits, the output probabilities in some large set T are somewhat large. Second, one uses Markov’s inequality to argue that, since the simulation error sums to ϵ , on some other large set of output probabilities S , the error must be below a constant multiple of the average. If S and T are both large, they must have some intersection – and on this intersection $S \cap T$, the imagined classical simulation is not only a simulation to additive error, but also to multiplicative error as well (since the output probability in question is above some minimum). Therefore a simulation to some amount ϵ of additive error implies a multiplicative simulation to the output probabilities on a constant fraction of the outputs. The impossibility of such a simulation is then obtained by assuming that computing these output probabilities is multiplicatively hard on average. In particular, one assumes that it is a $\#\text{P}$ -hard task to compute the output probability on $|S \cap T|/2^n$ -fraction of the outputs. This leads to a collapse of the polynomial hierarchy by known techniques [5, 48].

We follow this technique to show hardness of sampling with additive error. In

⁴However we note that in our proofs we will only use the fact that the Clifford group plus any non-Clifford element is universal on a qubit. This version of the theorem admits a direct proof using the representation theory of $SU(2)$.

our case, the anticoncentration theorem follows from the fact that the Clifford group is a “2-design” [229, 239] – i.e. a random Clifford circuit behaves equivalently to a random unitary up to its second moment – and therefore must anticoncentrate, as a random unitary does (the fact that unitary designs anticoncentrate was also shown independently by several groups [119, 122, 165]). This is similar to the hardness results for IQP [49] and DQC1 [173], in which the authors also prove their corresponding anticoncentration theorems. In contrast it is open to prove the anticoncentration theorem used for Boson Sampling and Fourier Sampling [5, 99], though these models have other complexity-theoretic advantages⁵. Therefore the only assumption needed is the hardness-on-average assumption. We also show that our hardness assumption is true for worst-case inputs. This result follows from combining known facts about BQP with the classification theorem for exact sampling hardness.

4.1.2 Relation to other works on modified Clifford circuits

While we previously discussed the relation of our results to prior work on gate set classification and sampling problems, here we compare our results to prior work on Clifford circuits. In Chapter 3, we categorized the computational power of a number of modified versions of Clifford circuits. The closest related result to the results in this chapter is the statement in [143] that if the input state to a Clifford circuit is allowed to be an arbitrary tensor product of one-qubit states, then such circuits cannot be efficiently classically simulated unless the polynomial hierarchy collapses. Their hardness result uses states of the form $|0\rangle^{\otimes n/2} |\alpha\rangle^{\otimes n/2}$, where $|\alpha\rangle = \cos(\pi/8)|0\rangle + i \sin(\pi/8)|1\rangle$ is a magic state. They achieve postselected hardness via the use of magic states to perform T gates, using a well-known construction (see e.g. [45]). So in the [143] construction there are different input states on different qubits. In contrast, our result requires the same input state on every qubit – as well as measurement in that basis at the end of the circuit. This ensures our modified circuit can be interpreted

⁵For instance, for these models it is known to be #P-hard to *exactly* compute most output probabilities of their corresponding circuit. This is a necessary but not sufficient condition for the supremacy conjectures to be true, which require it to be #P-hard to *approximately* compute most output probabilities of their corresponding circuit.

as the action of a discrete gate set, and therefore our result has relevance for the classification of the power of non-universal gate sets.

4.2 Conjugated Clifford circuits

Recall that a (unitary) Clifford circuit is a one that consists of the computational basis state $|0\rangle^{\otimes n}$ being acted on by the basic Clifford gates, before being measured in the computational basis. We define conjugated Clifford circuits (CCCs) similarly to Clifford circuits, except that each basic Clifford gate G is replaced by a conjugated basic Clifford gate $(U^{\otimes k})^\dagger g U^{\otimes k}$, where $k = 1$ when $g = H, S$ and $k = 2$ when $g = CZ$. In other words,

Definition 53. Let U be a single-qubit unitary gate. A U -conjugated Clifford circuit (U -CCC) on n qubits is defined to be a quantum circuit with the following structure:

1. Start with $|0\rangle^{\otimes n}$.
2. Apply gates from the set $\{U^\dagger H U, U^\dagger S U, (U^\dagger \otimes U^\dagger) CZ (U \otimes U)\}$.
3. Measure each qubit in the computational basis.

Because the intermediate U and U^\dagger gates cancel, we may equivalently describe a U -CCC as follows:

1. Start with $|0\rangle^{\otimes n}$.
2. Apply $U^{\otimes n}$.
3. Apply gates from the set $\{H, S, CZ\}$.
4. Apply $(U^\dagger)^{\otimes n}$.
5. Measure each qubit in the computational basis.

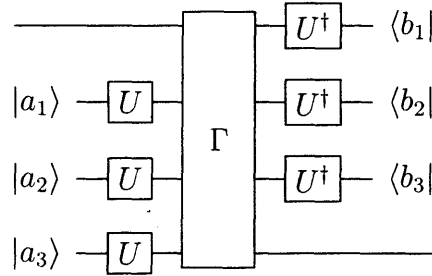
4.2.1 Postselection gadgets

Our results involve the use of postselection gadgets to simulate unitary operations. In this section, we introduce some terminology to describe these gadgets.

Definition 54. Let U be a single-qubit operation. Let $k, l \in \mathbb{Z}^+$ with $k > l$. A k -to- l U -CCC postselection gadget G is a postselected circuit fragment that performs the following procedure on an l -qubit system:

1. Introduce a set T of $(k - l)$ ancilla registers in the state $|a_1 \dots a_{k-l}\rangle$, where $a_1 \dots a_{k-l} \in \{0, 1\}^{k-l}$.
2. Apply $U^{\otimes(k-l)}$ to the set T of registers.
3. Apply a k -qubit Clifford operation Γ to both the system and ancilla.
4. Choose a subset S of $(k - l)$ registers and apply $(U^\dagger)^{\otimes(k-l)}$ to S .
5. Postselect on the subset S of qubits being in the state $|b_1 \dots b_{k-l}\rangle$, where $b_1 \dots b_{k-l} \in \{0, 1\}^{k-l}$.

An example of a 4-to-1 U -CCC postselection gadget is the circuit fragment described by the following diagram:



Let G be a U -CCC postselection gadget as described in Definition 54. The *action* $A(G)$ (also denoted A_G) of G is defined to be the linear operation that it performs, i.e.

$$A(G) = A_G = \langle b_1 \dots b_l |_S \left(\prod_{i \in S} U_i^\dagger \right) \Gamma \left(\prod_{i \in T} U_i \right) |a_1 \dots a_l \rangle_T, \quad (4.1)$$

and the *normalized action* of G , when it exists, is

$$\tilde{A}_G = \frac{A_G}{(\det A_G)^{2^{-t}}}. \quad (4.2)$$

Note that the above normalization is chosen so that $\det \tilde{A}_G = 1$.

We say that a U -CCC postselection gadget G is *unitary* if there exist $\alpha \in \mathbb{C} \setminus \{0\}$ and a unitary operator U such that $A_G = \alpha U$. It is straightforward to check that the following are equivalent conditions for gadget unitarity.

Lemma 55. A U -CCC postselection gadget G is unitary if and only if either one of the following holds:

1. There exists $\gamma > 0$ such that $A_G^\dagger A_G = \gamma I$,
2. $\tilde{A}_G^\dagger \tilde{A}_G = I$, i.e. \tilde{A}_G is unitary.

Similarly, we say that a U -CCC postselection gadget G is *Clifford* if there exist $\alpha \in \mathbb{C} \setminus \{0\}$ and a Clifford operator U such that $A_G = \alpha U$. The following lemma gives a necessary condition for a gadget to be Clifford.

Lemma 56. If G is a Clifford U -CCC postselection gadget, then

$$A_G X A_G^\dagger \propto X \text{ or } A_G X A_G^\dagger \propto Y \text{ or } A_G X A_G^\dagger \propto Z, \quad (4.3)$$

and

$$A_G Z A_G^\dagger \propto X \text{ or } A_G Z A_G^\dagger \propto Y \text{ or } A_G Z A_G^\dagger \propto Z. \quad (4.4)$$

Proof. If G is a Clifford U -CCC postselection gadget, then there exist $\alpha \in \mathbb{C} \setminus \{0\}$ and a Clifford operation Γ such that $A_G = \alpha \Gamma$. Since Γ is Clifford, $\Gamma X \Gamma^\dagger$ is a Pauli operator. But $\Gamma X \Gamma^\dagger \not\sim I$, otherwise, $X \sim I$, which is a contradiction. Hence, $\Gamma X \Gamma^\dagger \sim X$ or Y or Z , which implies Eq. (4.3). The proof of Eq. (4.4) is similar, with X replaced with Z . \square

4.3 Weak simulation of CCCs with multiplicative error

4.3.1 Classification results

In this section, we classify the hardness of weakly simulating U -CCCs as we vary U . As we shall see, it turns out that the classical simulation complexities of the U -CCCs associated with this notion of simulation are all of the following two types: the U -CCCs are either efficiently simulable, or are hard to simulate to constant multiplicative error unless the polynomial hierarchy collapses. To facilitate exposition, we will introduce the following terminology to describe these two cases: Let \mathcal{C} be a class of quantum circuits. Following the terminology in [148], we say that \mathcal{C} is in PWK if it is efficiently simulable in the weak sense by a classical computer. We say that \mathcal{C} is *PH-supreme* (or that it exhibits *PH-supremacy*) if it satisfies the property that if \mathcal{C} is efficiently simulable in the weak sense by a classical computer to constant multiplicative error, then the polynomial hierarchy (PH) collapses.

The approach we take to classifying the U -CCCs is to decompose each U into the form given by Eq. (2.4),

$$U = e^{i\alpha} R_z(\phi) R_x(\theta) R_z(\lambda), \quad (4.5)$$

and study how the classical simulation complexity changes as we vary α, ϕ, θ and λ . Two simplifications can immediately be made. First, the outcome probabilities of the U -CCC are independent of α , since α appears only in a global phase. Second, the probabilities are also independent of λ . To see this, note that the outcome probabilities are all of the form:

$$|\langle b | R_z(-\lambda)^{\otimes n} V R_z(\lambda)^{\otimes n} | 0 \rangle|^2 = |\langle b | V | 0 \rangle|^2, \quad (4.6)$$

which is independent of λ . In the above expression, $b \in \{0, 1\}^n$ and

$$V = R_x(-\theta)^{\otimes n} R_z(-\phi)^{\otimes n} \Gamma R_z(\phi)^{\otimes n} R_x(\theta)^{\otimes n}$$

$\phi \backslash \theta$	$\pi\mathbb{Z}$	$\frac{\pi}{2}\mathbb{Z}_{\text{odd}}$	$(\frac{\pi}{2}\mathbb{Z})^c$
$\frac{\pi}{2}\mathbb{Z}$	PWK (i, ii)	PWK (ii)	PH-supreme (iv)
$(\frac{\pi}{2}\mathbb{Z})^c$	PWK (i)	PH-supreme (iii)	PH-supreme (iv)

Table 4.1: Complete complexity classification of U -CCCs (where $U = R_z(\phi)R_x(\theta)$) with respect to weak simulation, as we vary ϕ and θ . The roman numerals in parentheses indicate the parts of Lemma 57 that are relevant to the corresponding box. All U -CCCs are either in PWK (i.e. can be efficiently simulated in the weak sense) or PH-supreme (i.e. cannot be simulated efficiently in the weak sense, unless the polynomial hierarchy collapses.)

for some Clifford circuit Γ . The equality follows from the fact that the computational basis states are eigenstates of $R_z(\lambda)^{\otimes n}$ with unit-magnitude eigenvalues.

Hence, to complete the classification, it suffices to just restrict our attention to the two-parameter family $\{R_z(\phi)R_x(\theta)\}_{\phi,\theta}$ of unitaries. We first prove the following lemma (see Table 4.1 for a summary):

Lemma 57. Let $U = R_z(\phi)R_x(\theta)$, where $\phi, \theta \in [0, 2\pi)$. Then

- U -CCCs are in PWK, if
 - (i) $\phi \in [0, 2\pi)$ and $\theta \in \pi\mathbb{Z}$, or
 - (ii) $\phi \in \frac{\pi}{2}\mathbb{Z}$ and $\theta \in \frac{\pi}{2}\mathbb{Z}$.
- U -CCCs are PH-supreme, if
 - (iii) $\phi \notin \frac{\pi}{2}\mathbb{Z}$ and $\theta \in \frac{\pi}{2}\mathbb{Z}_{\text{odd}}$, or
 - (iv) $\theta \notin \frac{\pi}{2}\mathbb{Z}$.

We defer the proof of Lemma 57 to Sections 4.3.2 and 4.3.3. Lemma 57 allows us to prove our main theorem:

Theorem 58. Let U be a single-qubit unitary operator. Consider the following two statements:

(A) U -CCC is in PWK.

(B) There exist a single-qubit Clifford operator $\Gamma \in \langle S, H \rangle$ and $\lambda \in [0, 2\pi)$ such that⁶

$$U \sim \Gamma R_z(\lambda). \quad (4.7)$$

Then,

1. (B) implies (A).
2. If the polynomial hierarchy is infinite, then (A) implies (B).

In other words, if we assume that the polynomial hierarchy is infinite, then U -CCCs are PH-supreme if and only if they cannot be written in the form $U \sim \Gamma R_z(\lambda)$, where Γ is a Clifford circuit and $R_z(\lambda)$ is a Z -rotation.

Proof.

1. Since $R_z(\lambda) |0\rangle \sim |0\rangle$, it follows that for any Γ , $\Gamma R_z(\lambda)$ -CCCs have the same outcome probabilities as Γ -CCCs. But C -CCCs are efficiently simulable, by the Gottesman-Knill Theorem, since $\Gamma \in \langle S, H \rangle$. Hence, U -CCCs are in PWK.
2. Let U be such that U -CCCs are in PWK. Using the decomposition in Eq. (2.4), write $U = e^{i\alpha} R_z(\phi) R_x(\theta) R_z(\lambda)$. Since we assumed that the polynomial hierarchy is infinite, Lemma 57 implies that

(a) $\theta \in \pi\mathbb{Z}$, or

(b) $\theta \in \frac{\pi}{2}\mathbb{Z}$ and $\phi \in \frac{\pi}{2}\mathbb{Z}$.

In Case (a), $\theta \in 2\pi\mathbb{Z}$ or $\pi\mathbb{Z}_{\text{odd}}$. If $\theta \in 2\pi\mathbb{Z}$, then

$$U \sim R_z(\phi) R_x(2\pi\mathbb{Z}) R_z(\lambda) = I \cdot R_z(\phi + \lambda),$$

⁶or alternatively, we could restrict the range of λ to be in $[0, \pi]$, since any factor of $R_z(\pi/2) \sim S$ can be absorbed into the Clifford operator Γ .

which is of the form given by Eq. (4.7). If $\pi\mathbb{Z}_{odd}$, then

$$U \sim R_z(\phi)R_x(\pi\mathbb{Z}_{odd})R_z(\gamma) \sim R_z(\phi)XR_z(\gamma) = XR_z(\gamma - \phi),$$

which is again of the form given by Eq. (4.7).

In Case (b),

$$\begin{aligned} U &\in e^{i\alpha}R_z(\pi\mathbb{Z}/2)R_x(\pi\mathbb{Z}/2)R_z(\gamma) \\ &= e^{i\alpha}R_z(\pi\mathbb{Z}/2)HR_z(\pi\mathbb{Z}/2)HR_z(\gamma). \end{aligned} \quad (4.8)$$

But the elements of $R_z(\pi\mathbb{Z}/2)$ are of the form S^j , for $j \in \mathbb{Z}$, up to a global phase. Therefore, $R_z(\pi\mathbb{Z}/2)HR_z(\pi\mathbb{Z}/2)H$ is Clifford, and U is of the form Eq. (4.7).

□

Hence, Theorem 58 tells us that under the assumption that the polynomial hierarchy is infinite, U -CCCs can be simulated efficiently (in the weak sense) if and only if $U \sim \Gamma R_z(\lambda)$ for some single qubit Clifford operator Γ , i.e. if U is a Clifford operation times a Z -rotation.

4.3.2 Proofs of efficient classical simulation

In this section, we prove Cases (i) and (ii) of Lemma 57.

4.3.2.1 Proof of Case (i): $\phi \in [0, 2\pi)$ and $\theta \in \pi\mathbb{Z}$

Theorem 59. Let $U = R_z(\phi)R_x(\theta)$. If $\phi \in [0, 2\pi)$ and $\theta \in \pi\mathbb{Z}$, then U -CCCs are in PWK.

Proof. First, we consider the case where $\theta \in 2\pi\mathbb{Z}$. In this case, $U = R_z(\phi)$, and the amplitudes of the U -CCC can be written as

$$\langle y | R_z(-\phi)^{\otimes n} \Gamma R_z(\phi)^{\otimes n} | x \rangle \sim \langle y | \Gamma | x \rangle \quad (4.9)$$

for some Clifford operation Γ and computational basis states $|x\rangle$ and $|y\rangle$. By the Gottesman-Knill Theorem, these U -CCCs can be efficiently weakly simulated.

Next, we consider the case where $\theta \in \pi\mathbb{Z}_{\text{odd}}$. In this case, $U = R_z(\phi)R_x(\pi) \sim R_z(\phi)X$, and the amplitudes of the U -CCC can be written as

$$\langle y | X^{\otimes n} R_z(-\phi)^{\otimes n} \Gamma R_z(\phi)^{\otimes n} X^{\otimes n} | x \rangle \sim \langle \bar{y} | \Gamma | \bar{x} \rangle \quad (4.10)$$

for some Clifford operation Γ and computational basis states $|x\rangle$ and $|y\rangle$, where \bar{z} is the bitwise negation of z . By the Gottesman-Knill Theorem, these U -CCCs can be efficiently weakly simulated.

Putting the above results together, we get that U -CCCs are in PWK. \square

4.3.2.2 Proof of Case (ii): $\phi \in \frac{\pi}{2}\mathbb{Z}$ and $\theta \in \frac{\pi}{2}\mathbb{Z}$

Theorem 60. Let $U = R_z(\phi)R_x(\theta)$. If $\phi \in \frac{\pi}{2}\mathbb{Z}$ and $\theta \in \frac{\pi}{2}\mathbb{Z}$, then U -CCCs are in PWK.

Proof. The elements of $R_z(\frac{\pi}{2}\mathbb{Z})$ are of the form S^j , where $j \in \mathbb{Z}$, up to a global phase. Therefore, $U = R_z(\phi)R_x(\theta) = R_z(\phi)HR_z(\theta)H$ is a Clifford operation, and so, the U -CCCs consist of only Clifford gates. By the Gottesman-Knill Theorem, these U -CCCs can be efficiently (weakly) simulated. \square

4.3.3 Proofs of hardness

In this section, we prove Cases (iii) and (iv) of Lemma 57. Our proof uses postselection gadgets, similar to the techniques used in [37, 48]. One can also prove hardness using techniques from measurement-based-quantum computing, at least for certain U . We give such a proof in Chapter 4.7 for the interested reader; we believe this proof may be more intuitive for those who are familiar with measurement-based quantum computing.

We start by proving a lemma that will be useful for the proofs of hardness.

Lemma 61. (Sufficient condition for PH-supremacy) Let U be a single-qubit gate. If there exists a unitary non-Clifford U -CCC postselection gadget G , then U -CCCs are PH-supreme.

Proof. Suppose such a gadget G exists. Then, since the Clifford group plus any non-Clifford gate is universal [176, 177, 218], the Clifford group plus G must be universal on a single qubit. Then, by the inverse-free Solovay-Kitaev Theorem of Sardharwalla et al. [198], using polynomially many gates from the set G, H, S one can compile any desired one-qubit unitary V to inverse exponential accuracy (since in particular $\langle H, S \rangle$ contains the Paulis). In particular, since any three-qubit unitary can be expressed as a product of a constant number of CZs and one-qubit unitaries, one can compile any gate in the set $\{CCZ, \text{controlled-}H, \text{all one-qubit gates}\}$ to inverse exponential accuracy with polynomial overhead.

In his proof that $\text{PostBQP} = \text{PP}$, Aaronson showed that postselected poly-sized circuits of the above gates can compute any language in PP [2]. Furthermore, as his postselection succeeds with inverse exponential probability, compiling these gates to inverse exponential accuracy is sufficient for performing arbitrary PP computations.

Hence, by using polynomially many gadgets for G, CZ, H and S , one can compile Aaronson's circuits⁷ for computing PP to inverse exponential accuracy, and hence these circuits can compute PP -hard problems. PH -supremacy then follows from the techniques of [5, 48]. Namely, a weak simulation of such circuits with constant multiplicative error would place $\text{PP} \subseteq \text{BPP}^{\text{NP}} \subseteq \Delta_3$ by Stockmeyer counting, and hence by Toda's theorem this would result in the collapse of PH to the third level. In fact, by the arguments of Fujii et al. [104], one can collapse PH to the second level as well, by placing coC=P in SBP , and we refer the interested reader to their work for the complete argument. □

⁷More specifically, we compile the circuit given by $(U^\dagger)^{\otimes n}$, then Aaronson's circuit, then $U^{\otimes n}$, as we need to cancel the U 's at the beginning and the U^\dagger 's at the end in order to perform Aaronson's circuit which starts and measures in the computational basis. However as the U, U^\dagger are one-qubit gates, one can cancel them to inverse exponential accuracy using our gates, and hence this construction suffices.

4.3.3.1 Proof of Case (iii): $\phi \notin \frac{\pi}{2}\mathbb{Z}$ and $\theta \in \frac{\pi}{2}\mathbb{Z}_{\text{odd}}$

Let $U = R_z(\phi)R_x(\theta)$. Consider the following U -CCC postselection gadget:

$$I(\phi, \theta) = \begin{array}{c} \text{---} \bullet \text{---} \boxed{U^\dagger} \text{---} \langle 0| \\ | \\ \text{---} \bullet \text{---} \boxed{U} \text{---} |0\rangle \end{array} \quad (4.11)$$

We now prove some properties about $I(\phi, \theta)$.

Theorem 62.

1. The action of $I(\phi, \theta)$ is

$$A_{I(\phi, \theta)} = \begin{pmatrix} \cos^2 \frac{\theta}{2} & \frac{i}{2} \sin \theta e^{-i\phi} \\ -\frac{i}{2} \sin \theta e^{i\phi} & -\sin^2 \frac{\theta}{2} \end{pmatrix}. \quad (4.12)$$

2. $I(\phi, \theta)$ is a unitary gadget if and only if $\theta \in \frac{\pi}{2}\mathbb{Z}_{\text{odd}}$. When $I(\phi, \theta)$ is unitary,

$$\tilde{A}_{I(\phi, \theta)} = \frac{i}{\sqrt{2}} \begin{pmatrix} 1 & i(-1)^k e^{-i\phi} \\ -i(-1)^k e^{i\phi} & -1 \end{pmatrix}, \quad (4.13)$$

where $k = \frac{\theta}{\pi} - \frac{1}{2}$.

3. $I(\phi, \theta)$ is a Clifford gadget if and only if $\phi \in \frac{\pi}{2}\mathbb{Z}$ and $\theta \in \frac{\pi}{2}\mathbb{Z}_{\text{odd}}$.
4. $I(\phi, \theta)$ is a unitary non-Clifford gadget if and only if $\phi \notin \frac{\pi}{2}\mathbb{Z}$ and $\theta \in \frac{\pi}{2}\mathbb{Z}_{\text{odd}}$.

Proof.

1. By direct calculation.
2. By Eq. (4.12),

$$A_{I(\phi, \theta)}^\dagger A_{I(\phi, \theta)} = \begin{pmatrix} \cos^2 \frac{\theta}{2} & \frac{i}{4} \sin(2\theta) e^{-i\phi} \\ -\frac{i}{4} \sin(2\theta) e^{i\phi} & \sin^2 \frac{\theta}{2} \end{pmatrix}. \quad (4.14)$$

If $\theta \in \frac{\pi}{2}\mathbb{Z}_{\text{odd}}$, then $A_{I(\phi,\theta)}^\dagger A_{I(\phi,\theta)} = \frac{1}{2}I$, which implies that $I(\phi,\theta)$ is a unitary gadget, by Lemma 55. Conversely, assume that $I(\phi,\theta)$ is a unitary gadget. Suppose that $\theta \notin \frac{\pi}{2}\mathbb{Z}_{\text{odd}}$. Then $\sin(2\theta) \neq 0$, which implies that $A_{I(\phi,\theta)}^\dagger A_{I(\phi,\theta)} \not\propto I$, which is a contradiction. Hence, $\theta \in \frac{\pi}{2}\mathbb{Z}_{\text{odd}}$.

Next, $k = \frac{\theta}{\pi} - \frac{1}{2}$ implies that $\theta = \frac{\pi}{2}(2k+1)$. Since $\theta \in \frac{\pi}{2}\mathbb{Z}_{\text{odd}}$, it follows that $k \in \mathbb{Z}$. Then $\sin \theta = (-1)^k$, $\cos^2 \frac{\theta}{2} = \frac{1}{2}$ and $\sin^2 \frac{\theta}{2} = \frac{1}{2}$. Hence,

$$A_{I(\phi,\theta)} = \begin{pmatrix} \frac{1}{2} & \frac{i}{2}(-1)^k e^{-i\phi} \\ -\frac{i}{2}(-1)^k e^{i\phi} & -\frac{1}{2} \end{pmatrix}. \quad (4.15)$$

Hence, $\det A_{I(\phi,\theta)} = -\frac{1}{2}$. Plugging this and Eq. (4.15) into Eq. (4.2) gives Eq. (4.13).

3. (\Leftarrow) Let $\phi \in \frac{\pi}{2}\mathbb{Z}$ and $\theta \in \frac{\pi}{2}\mathbb{Z}_{\text{odd}}$. Write $\phi = \frac{\pi}{2}l$ and $\theta = \frac{\pi}{2}(2k+1)$. Then, by Eq. (4.13),

$$\tilde{A}_{I(\phi,\theta)} = \frac{i}{\sqrt{2}} \begin{pmatrix} 1 & i^{1+2k+3l} \\ i^{3+2k+l} & -1 \end{pmatrix}. \quad (4.16)$$

Now, it is straightforward to check that for all $k, l \in \mathbb{Z}$, $\tilde{A}_{I(\phi,\theta)} X \tilde{A}_{I(\phi,\theta)}^\dagger \in \{-X, Z, -Z\}$ and $\tilde{A}_{I(\phi,\theta)} Z \tilde{A}_{I(\phi,\theta)}^\dagger \in \{-Y, X, Y, -X\}$. This shows that $\tilde{A}_{I(\phi,\theta)}$ maps the Pauli group to itself, under conjugation, which implies that $\tilde{A}_{I(\phi,\theta)}$ is Clifford.

(\Rightarrow) Assume that $I(\phi,\theta)$ is a Clifford gadget. Suppose that $\phi \notin \frac{\pi}{2}\mathbb{Z}$ or $\theta \notin \frac{\pi}{2}\mathbb{Z}_{\text{odd}}$. But $I(\phi,\theta)$ is unitary, and hence, $\theta \in \frac{\pi}{2}\mathbb{Z}_{\text{odd}}$. So $\phi \notin \frac{\pi}{2}\mathbb{Z}$. By Lemma 56, $\tilde{A}_{I(\phi,\theta)} X \tilde{A}_{I(\phi,\theta)}^\dagger \sim X$ or Y or Z . But, as we compute,

$$\tilde{A}_{I(\phi,\theta)} X \tilde{A}_{I(\phi,\theta)}^\dagger = \begin{pmatrix} (-1)^k \sin \phi & -e^{-i\phi} \cos \phi \\ -e^{i\phi} \cos \phi & -(-1)^k \sin \phi \end{pmatrix}. \quad (4.17)$$

If $\tilde{A}_{I(\phi,\theta)} X \tilde{A}_{I(\phi,\theta)}^\dagger \sim X$ or Y , then $\sin \phi = 0$, which is a contradiction, since $\phi \notin \frac{\pi}{2}\mathbb{Z}$. Hence, $\tilde{A}_{I(\phi,\theta)} X \tilde{A}_{I(\phi,\theta)}^\dagger \sim Z$, which implies that $\cos \phi = 0$. But this also contradicts $\phi \notin \frac{\pi}{2}\mathbb{Z}$. Hence, $\phi \in \frac{\pi}{2}\mathbb{Z}$ and $\theta \in \frac{\pi}{2}\mathbb{Z}_{\text{odd}}$.

4. Follows from Parts 2 and 3 of Theorem 62.

□

Theorem 63. Let $U = R_z(\phi)R_x(\theta)$. If $\phi \notin \frac{\pi}{2}\mathbb{Z}$ and $\theta \in \frac{\pi}{2}\mathbb{Z}_{\text{odd}}$, then U -CCCs are PH-supreme.

Proof. By Theorem 62, when $\phi \notin \frac{\pi}{2}\mathbb{Z}$ and $\theta \in \frac{\pi}{2}\mathbb{Z}_{\text{odd}}$, then $I(\phi, \theta)$ is a unitary non-Clifford U -CCC postselection gadget. Hence, by Lemma 61, U -CCCs are PH-supreme. □

4.3.3.2 Proof of Case (iv): $\theta \notin \frac{\pi}{2}\mathbb{Z}$

Let $U = R_z(\phi)R_x(\theta)$. Consider the following U -CCC postselection gadget:

$$J(\phi, \theta) = \begin{array}{c} \text{-----} \\ \quad \quad \quad \bullet \\ \quad \quad \quad | \\ \quad \quad \quad \bullet \\ |0\rangle \text{---} [U] \text{---} [S] \text{---} [U^\dagger] \text{---} \langle 0| \end{array} \quad (4.18)$$

We now prove some properties about $J(\phi, \theta)$.

Theorem 64.

1. The action of $J(\phi, \theta)$ is

$$\begin{aligned} A_{J(\phi, \theta)} &= \frac{1}{\sqrt{2}} e^{-i\frac{\pi}{4}} \begin{pmatrix} i + \cos \theta & 0 \\ 0 & 1 + i \cos \theta \end{pmatrix} \\ &= \frac{i}{\sqrt{2}} e^{-i\frac{\pi}{4}} \sqrt{1 + \cos^2 \theta} S^\dagger R_z(2 \tan^{-1}(\cos \theta)). \end{aligned} \quad (4.19)$$

2. $J(\phi, \theta)$ is a unitary gadget for all $\theta, \phi \in [0, 2\pi)$. The normalized action is

$$\tilde{A}_{J(\phi, \theta)} \sim S^\dagger R_z(2 \tan^{-1}(\cos \theta)). \quad (4.20)$$

3. $J(\phi, \theta)$ is a Clifford gadget if and only if $\theta \in \frac{\pi}{2}\mathbb{Z}$.
4. $J(\phi, \theta)$ is a unitary non-Clifford gadget if and only if $\theta \notin \frac{\pi}{2}\mathbb{Z}$.

Proof.

1. By direct calculation.
2. The determinant of $A_{J(\phi, \theta)}$ is

$$\det A_{J(\phi, \theta)} = \frac{1}{2}(1 + \cos^2 \theta) \neq 0 \quad (4.21)$$

for all θ and ϕ . Hence, $A_{J(\phi, \theta)} \propto S^\dagger R_z(2 \tan^{-1}(\cos \theta))$ for all θ and ϕ , which implies that $J(\phi, \theta)$ is a unitary gadget for all θ and ϕ .

Hence,

$$\tilde{A}_{J(\phi, \theta)} = \frac{A_{J(\phi, \theta)}}{\sqrt{\det A_{J(\phi, \theta)}}} = ie^{-i\frac{\pi}{4}} S^\dagger R_z(2 \tan^{-1}(\cos \theta)).$$

- 3.

$$\begin{aligned} J(\phi, \theta) \text{ is a Clifford gadget} &\Leftrightarrow S^\dagger R_z(2 \tan^{-1}(\cos \theta)) \text{ is Clifford} \\ &\Leftrightarrow R_z(2 \tan^{-1}(\cos \theta)) \text{ is Clifford} \\ &\Leftrightarrow 2 \tan^{-1}(\cos \theta) \in \frac{\pi}{2}\mathbb{Z} \quad (4.22) \\ &\Leftrightarrow \cos \theta \in \{0, 1, -1\} \end{aligned}$$

$$\Leftrightarrow \theta \in \frac{\pi}{2}\mathbb{Z}. \quad (4.23)$$

4. Follows from Parts 2 and 3 of Theorem 64.

where to get Eq. (4.22), we used the fact that $R_z(\phi)$ is a Clifford operation if and only if $\phi \in \frac{\pi}{2}\mathbb{Z}$. □

Theorem 65. Let $U = R_z(\phi)R_x(\theta)$. If $\theta \notin \frac{\pi}{2}\mathbb{Z}$, then U -CCCs are PH-supreme.

Proof. By Theorem 64, when $\theta \notin \frac{\pi}{2}\mathbb{Z}$, then $I(\phi, \theta)$ is a unitary non-Clifford U -CCC postselection gadget. Hence, by Lemma 61, U -CCCs are PH-supreme. □

4.4 Weak simulation of CCCs with additive error

Here, we show how to achieve additive hardness of simulating conjugated Clifford circuits, under additional hardness assumptions. Specifically, we will show that under these assumptions, there is no classical randomized algorithm which given a one-qubit unitary U and a Clifford circuit V , samples the output distribution of V conjugated by U 's up to constant ℓ_1 error.

In the following, let V be a Clifford circuit on n qubits, U be a one-qubit unitary which is not a Z -rotation times a Clifford, and $y \in \{0, 1\}^n$ be an n -bit string. Define

$$p_{y,U,V} = |\langle y | (U^\dagger)^{\otimes n} V U^{\otimes n} |0^n\rangle|^2.$$

In other words $p_{y,U,V}$ is the probability of outputting the string y when applying the circuit V conjugated by U 's to the all 0's state, and then measuring in the computational basis. Let the corresponding probability distribution on y 's given U and V be denoted $D(U, V)$.

Theorem 66. Assuming that PH is infinite and Conjecture 67, then there is no classical algorithm which given a one-qubit unitary U and an n -qubit Clifford circuit V , outputs a probability distribution which is $1/100$ close to $D(U, V)$ in total variation distance.

Conjecture 67. For any U which is not equal to a Z -rotation times a Clifford, it is $\#P$ -hard to approximate a $6/50$ fraction of the $p_{y,U,V}$ over the choice of y, V to within multiplicative error $1/2 + o(1)$.

In order to prove this we'll actually prove a more general theorem described below; the result will then follow from simply setting $a = c = 1/5$, $\epsilon = 1/100$. One can in general plug in any values they like subject to the constraints; for instance one can strengthen the hardness assumption by assuming computing a smaller fraction of the $p_{y,U,V}$ is still $\#P$ -hard to obtain larger allowable error in the simulation. These parameters are similar to those appearing in other hardness conjectures, for example those used for IQP [49].

Theorem 68. Pick constants $0 < \epsilon, a, c < 1$ such that $(1 - a)^2/2 - c > 0$ and $\frac{2\epsilon}{ac} < 1$. Then assuming Conjecture 69, given a one-qubit unitary U and an n -qubit Clifford circuit V , one cannot weakly simulate the distribution $D(U, V)$ with a randomized classical algorithm with total variation distance error ϵ , unless the polynomial hierarchy collapses to the third level.

Conjecture 69. For any U which is not equal to a Z-rotation times a Clifford, it is #P-hard to multiplicatively approximate $(1 - a)^2/2 - c$ fraction of the $p_{y,U,V}$ over the choice of (y, V) , up to multiplicative error $\frac{2\epsilon}{ac} + o(1)$.

Proof of Theorem 68. Suppose by way of contradiction that there exists a classical poly-time randomized algorithm which given inputs U, V outputs samples from a distribution $D'(U, V)$ such that $\frac{1}{2}|D(U, V) - D'(U, V)|_1 < \epsilon$. In particular, let $q_{y,U,V}$ be the probability that $D'(U, V)$ outputs y – i.e. the probability that the simulation outputs y under inputs U, V .

By our simulation assumption, for all U, V we have that $\sum_y |q_{y,U,V} - p_{y,U,V}| \leq 2\epsilon$. Therefore by Markov's inequality, given our constant $0 < c < 1$, we have that for all U and V there exists a set $S' \subseteq \{0, 1\}^n$ of output strings y of size $|S'|/2^n > 1 - c$, such that for all $y \in S'$,

$$|q_{y,U,V} - p_{y,U,V}| \leq \frac{2\epsilon}{c2^n}.$$

In particular, by averaging over V 's, we see that for any U as above, there exists a set $S \subset \{0, 1\}^n \times \mathcal{C}_n$ of pairs (y, V) such that for all $(y, V) \in S$, $|q_{y,U,V} - p_{y,U,V}| \leq \frac{2\epsilon}{c2^n}$. Furthermore S has measure at least $(1 - c)$ over a uniformly random choice of (y, V) .

We now show the following anticoncentration lemma (similar theorems were shown independently in [119, 122, 165]):

Lemma 70. For any fixed U and y as above, and for any constant $0 < a < 1$, we have that at least $\frac{(1-a)^2}{2}$ fraction of the Clifford circuits V have the property that

$$p_{y,U,V} \geq \frac{a}{2^n}.$$

We will prove Lemma 70 shortly. First, we will show why this implies Theorem 68.

In particular, by averaging Lemma 70 over y 's, we see that for any U as above, there exists a set $T \subset \{0, 1\}^n \times \mathcal{C}_n$ of pairs (y, V) such that for all $(y, V) \in T$, $p_{y,U,V} \geq \frac{a}{2^n}$. Furthermore T has measure at least $\frac{(1-a)^2}{2}$ over a uniformly random choice of (y, V) . Since we assumed that $(1-a)^2/2 + (1-c) > 1$, then $S \cap T$ must be nonempty, and in particular must contain $(1-a)^2/2 - c$ fraction of the pairs (y, V) . On this set $S \cap T$, we have that

$$q_{y,U,V} \leq p_{y,U,V} + \frac{2\epsilon}{c2^n} = p_{y,U,V} + \frac{2\epsilon}{ac} \frac{a}{2^n} \leq \left(1 + \frac{2\epsilon}{ac}\right) p_{y,U,V},$$

and likewise

$$q_{y,U,V} \geq p_{y,U,V} - \frac{2\epsilon}{c2^n} = p_{y,U,V} - \frac{2\epsilon}{ac} \frac{a}{2^n} \geq \left(1 - \frac{2\epsilon}{ac}\right) p_{y,U,V}.$$

Since $1 - \frac{2\epsilon}{ac} > 0$ (which we guaranteed by assumption), $q_{y,U,V}$ is a multiplicative approximation to $p_{y,U,V}$ with multiplicative error $\frac{2\epsilon}{ac}$ for (y, V) in the set $S \cap T$. The set $S \cap T$ contains at least $(1-a)^2/2 - c$ fraction of the total pairs (y, V) .

On the other hand, by Conjecture 69 we have that computing a $(1-a)^2/2 - c$ fraction of the $p_{y,U,V}$ to this level of multiplicative error is a #P-hard task. So approximating $p_{y,U,V}$ to this level of multiplicative error for this fraction of outputs is both #P-hard, and achievable by our simulation algorithm. This collapses PH to the third level by known arguments [5, 48]. In particular, by applying Stockmeyer's approximate counting algorithm [216] to $p_{y,U,V}$, one can multiplicatively approximate $q_{y,U,V}$ to multiplicative error $\frac{1}{\text{poly}}$ in FBPP^{NP} for those elements in $S \cap T$. But since $q_{y,U,V}$ is a $\frac{2\epsilon}{ac}$ -approx to $p_{y,U,V}$, this is a $\frac{2\epsilon}{ac} + o(1)$ multiplicative approximation to $p_{y,U,V}$ in $S \cap T$. Hence a #P-hard quantity is in FBPP^{NP} . This collapses PH to the third level by Toda's theorem [219].

To complete our proof of Theorem 68, we will prove Lemma 70.

Proof of Lemma 70. To prove this, we will make use of the fact that the Clifford group is an exact 2-design⁸ [229, 239]. The fact that the Clifford group is a 2-design means

⁸The Clifford group is also a 3-design, but we will only need the fact it is a 2-design for our proof.

that for any polynomial p over the variables $\{V_{ij}\}$ and their complex conjugates, which is of degree at most 2 in the V_{ij} 's and degree at most 2 in the V_{ij}^* 's, we have that

$$\frac{1}{|\mathcal{C}_n|} \sum_{V \in \mathcal{C}} p(V, V^*) = \int p(V, V^*) dV,$$

where \mathcal{C}_n denotes the Clifford group and the integral dV is taken over the Haar measure. In other words, the expectation values of low-degree polynomials in the entries of the matrices are exactly identical to the expectation values over the Haar measure.

In particular, note that $p_{y,U,V}$ is a degree-1 polynomial in the entries of V and their complex conjugates, and $p_{y,U,V}^2$ is a degree-2 polynomial in these variables. Therefore, since the Clifford group is an exact 2-design, we have that for any y and U ,

$$\frac{1}{|\mathcal{C}_n|} \sum_{V \in \mathcal{C}_n} p_{y,U,V} = \int p_{y,U,V} dV = \frac{1}{2^n}$$

and

$$\frac{1}{|\mathcal{C}_n|} \sum_{V \in \mathcal{C}_n} p_{y,U,V}^2 = \int p_{y,U,V}^2 dV = \frac{2}{2^{2n} - 1} \left(1 - \frac{1}{2^n}\right),$$

where the values of these integrals over the Haar measure are well known – see for instance Appendix D of [121].

Following [49], we now invoke the Paley-Zygmund inequality, which states that:

Proposition 71. Given a parameter $0 < a < 1$, and a non-negative random variable p of finite variance, we have

$$\Pr[p \geq a\mathbb{E}[p]] \geq (1 - a)^2 \mathbb{E}[p]^2 / \mathbb{E}[p^2].$$

Applying this inequality to the random variable $p_{y,U,V}$ over the choice of the Clifford circuit V , we have that

$$\Pr_V \left[p_{y,U,V} \geq \frac{a}{2^n} \right] \geq (1 - a)^2 \frac{2^{-2n}}{\frac{2 - 2^{-n+1}}{2^{2n} - 1}} = (1 - a)^2 \frac{1 - 2^{-2n}}{2 - 2^{-n+1}} \geq \frac{(1 - a)^2}{2}$$

which implies the claim. □

This completes the proof of Theorem 68. □

4.5 Evidence in favor of hardness conjecture

In Chapter 4.4, we saw that by assuming an average case hardness conjecture (namely Conjecture 69), we could show that a weak simulation of CCCs to additive error would collapse the polynomial hierarchy. A natural question is: what evidence do we have that Conjecture 69 is true?

In this section, we show that the worst-case version of Conjecture 69 is true. In fact, we show that for any $U \neq CR_Z(\theta)$ for a Clifford C , there exist a Clifford circuit V and an output y such that computing $p_{y,U,V}$ is $\#P$ -hard to constant multiplicative error. Therefore certainly *some* output probabilities of CCCs are $\#P$ -hard to compute. Conjecture 69 is merely conjecturing further that computing a large fraction of such output probabilities is just as hard.

Theorem 72 (Worst-case version of Conjecture 69). For any U which is not equal to a Z -rotation times a Clifford, there exist a Clifford circuit V and string $y \in \{0, 1\}^n$ such that it is $\#P$ -hard to multiplicatively approximate a $p_{y,U,V}$ to multiplicative error $1/2 - o(1)$.

Proof. This follows from combining the ideas from the proof of Lemma 57 with previously known facts about BQP. In particular, we will use the following facts:

1. There exists a uniform family of poly-size BQP^9 circuits C_x where $x \in \{0, 1\}^n$ using a gate set with algebraic entries such that computing $|\langle 0^n | C_x | 0^n \rangle|^2$ to multiplicative error $1/2$ is $\#P$ -hard [49].
2. For any poly-sized quantum circuit C over a gate set with algebraic entries, any non-zero output probability has magnitude at least inverse exponential [154].

⁹Even IQP suffices here [49].

3. As shown in the proof of Theorem 58, for any U which is not a Clifford gate times a Z rotation, there is a postselection gadget G which performs a unitary but non-Clifford one-qubit operation. Furthermore all ancilla qubits in G begin in the state $|0\rangle$.

From these facts, we can now prove the theorem. Let $p = |\langle 0^n | C_x | 0^n \rangle|^2$. By Fact 2, the circuit C_x from Fact 1 either has $p = 0$ or $p \geq 2^{-O(n^c)}$ for some constant c . Now suppose we compile the circuit C_x from Fact 1 using Clifford gates plus the postselection gadget G – call this new circuit with postselection C'_x . By Sardharwalla et al. [198] we can compile this circuit with accuracy $\epsilon = 2^{-O(n^c)-100}$ with only polynomial overhead.

Let $\ell \in \{0, 1\}^k$ be the string of postselection bits of the circuit C'_x (which without loss of generality are the last bits of the circuit), and let α is the probability that all postselections succeed. Note α is a known and easily calculated quantity, since each postselection gadget is unitary so succeeds with a known constant probability.

Let $p' = |\langle 0^n \ell | C'_x | 0^{n+k} \rangle|^2 / \alpha$. Then we have that:

- If $p = 0$ then $p' \leq 2^{-O(n^c)-100}$.
- If $p \neq 0$ then $p - 2^{-O(n^c)-100} \leq p' \leq p + 2^{-O(n^c)-100}$. Since $p \geq 2^{-O(n^c)}$, this is a multiplicative approximation to p with error 2^{-100} .

Now suppose that one can compute $|\langle 0^n \ell | C'_x | 0^{n+k} \rangle|^2$ to multiplicative error γ to be chosen shortly. Then immediately one can compute $p' = |\langle 0^n \ell | C'_x | 0^{n+k} \rangle|^2 / \alpha$ to the same amount of multiplicative error – call this estimate p'' . By the above argument, if $p = 0$ then $p'' < 2^{-O(n^c)-100}(1 + \gamma)$. On the other hand if $p > 0$ then $p' > 2^{-O(n^c)}$, so $p'' > 2^{-O(n^c)}(1 - \gamma)$. So long as γ is chosen such that $2^{-100}(1 + \gamma) < (1 - \gamma)$ these two cases can be distinguished – which holds in particular if $\gamma \approx 1/2$.

Therefore, if $p'' < 2^{-O(n^c)}$ then we can infer that $p = 0$. If $p'' > 2^{-O(n^c)}(1 - \gamma)$, then $p > 0$ so p'' is a γ approximation to p' and hence a $\gamma + 2^{-100} + \gamma 2^{-100}$ approximation to p . In either case we have computed a $\gamma + 2^{-100} + \gamma 2^{-100}$ approximation to p . Therefore, if $\gamma = 1/2 - 2^{-99}$, then we have computed a $1/2$ -multiplicative approximation to p ,

which is $\#P$ -hard by Fact 1. Therefore, computing the probability that the CCC corresponding to C'_x outputs $|0^n \ell\rangle$ to multiplicative error $1/2 - 2^{-99}$ is $\#P$ -hard. One can similarly improve this hardness to $1/2 - o(1)$. \square

Given that the worst-case version of Conjecture 69 is true, a natural question to ask is how difficult it would be to prove the average-case conjecture. To do so would in particular prove quantum advantage over classical computation with realistic error, and merely assuming the polynomial hierarchy is infinite. In some ways this would be stronger evidence for quantum advantage over classical computation than Shor's factoring algorithm, as there are no known negative complexity-theoretic consequences if factoring is contained in P .

Unfortunately, recent work has shown that proving Conjecture 69 would be a difficult task. Specifically, Aaronson and Chen [7] demonstrated an oracle relative to which PH is infinite, but classical computers can efficiently weakly simulate quantum devices to constant additive error. Therefore, any proof which establishes quantum advantage with additive error under the assumption that PH is infinite must be non-relativizing. In particular this implies any proof of Conjecture 69 would require non-relativizing techniques – in other words it could not remain true if one allows for classical oracle class in the circuit. This same barrier holds for proving the similar average-case hardness conjectures to show advantage for Boson Sampling, IQP, DQC1, or Fourier sampling. Therefore any proof of Conjecture 69 would require facts specific to the Clifford group. We leave this as an open problem. We also note that it remains open to prove the average-case *exact* version of Conjecture 69 - i.e. whether it is hard to exactly compute a large fraction of $p_{y,U,C}$. We believe this may be a more tractable problem to approach than Conjecture 69. However this remains open, as is the analogous average-case exact conjecture corresponding to IQP. We note the corresponding average-case exact conjecture for Boson Sampling and Fourier sampling are known to be true [5, 99], though these models are not known to anticoncentrate.

4.6 CCCs and other notions of simulation

For completeness, in this section we summarize the simulability of U -CCCs when U is not a Clifford rotation times a Z rotation. There are various notions of classical simulation at play here. The results of this chapter so far have focused on notions of approximate *weak* simulation. A *weak* simulation of a family of quantum circuits is a classical randomized algorithm that samples from the same distribution as the output distribution of the circuit. On the other hand, a *strong* simulation of a family of quantum circuits is a classical algorithm that computes not only the joint probabilities, but also any marginal probabilities of the outcomes of the measurements in the circuit. Following [148], we can further refine these definitions according to the number of qubits being measured: a *strong(1)* simulation computes the marginal output probabilities on individual qubits, and a *strong(n)* simulation computes the probability of output strings $y \in \{0, 1\}^n$. Similarly, a *weak(1)* simulation samples from the marginal output probabilities on individual qubits, and a *weak(n)* simulation samples from $p(y_1, \dots, y_n)$. A *weak⁺* simulation samples from the same distribution on all n output qubits up to constant additive error. Our previous results have shown that efficient *weak(n)* simulations (Theorem 58), *weak⁺* simulations (Theorem 68), and *strong(n)* simulations (Theorem 72) of CCCs are implausible. However it is natural to ask if it is possible to simulate single output probabilities of CCCs. It turns out the answer to this question is yes. This follows immediately from Theorem 5 of [148], which showed more generally that Clifford circuits with product inputs or measurements have an efficient *strong(1)* and *weak(1)* simulation. Therefore this completes the complexity classification of the simulability of such circuits. We note that IQP has identical properties in this regard. This emphasizes that the difficulty in simulating CCCs (or IQP circuits) comes from the difficulty of simulating all of the *marginal* probability distributions contained in the output distribution, where the marginal is taken over a large number of output bits. The probabilities of computing individual output bits of either model are easy for classical computation. This is summarized in Figure 4-1.

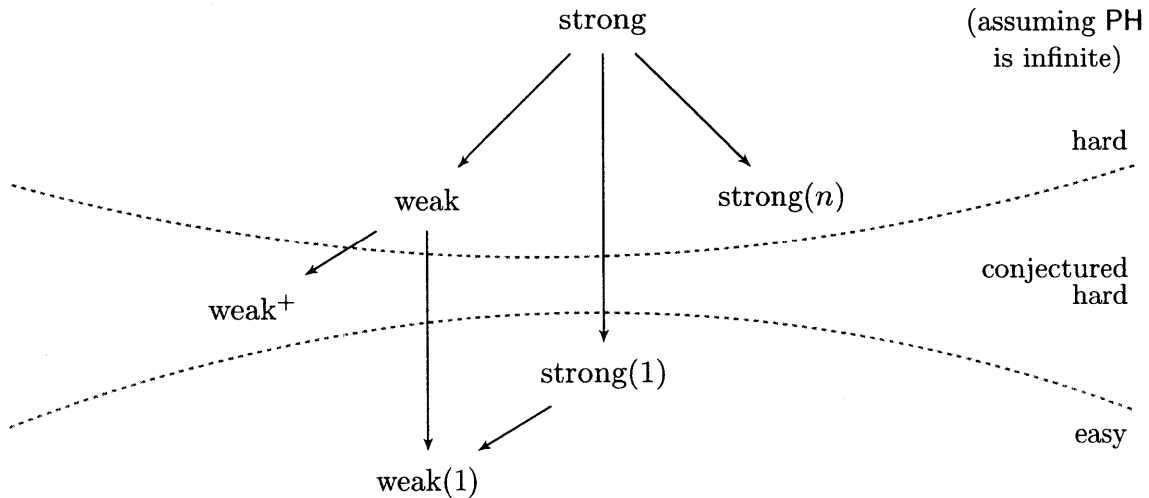


Figure 4-1: Relationships between different notions of classical simulation and summary of the hardness of simulating CCCs. An arrow from A to B ($A \rightarrow B$) means that an efficient A -simulation of a computational task implies that there is an efficient B -simulation for the same task. Note also that a $\text{weak}(n)$ simulation exists if and only if a weak simulation exists. For a proof of these relationships, see Chapter 3.3. The two curves indicate the boundary between efficiencies of simulation of U -CCCs, where U is not a Clifford operation times a Z rotation. “Hard” means that an efficient simulation of U -CCCs is not possible, unless PH collapses. “Conjectured hard” means that an efficient simulation of U -CCCs is not possible, if we assume Conjecture 69. “Easy” means that an efficient simulation of U -CCCs exists. Note that when U is a Clifford operation times a Z rotation, all the above notions become easy.

4.7 Measurement-based quantum computing proof of multiplicative hardness for CCCs for certain U 's

In this section, we will prove the following theorem using techniques from measurement-based quantum computing (MBQC).

Theorem 73. CCCs with $U = R_Z(\theta)H$ cannot be efficiently weakly classically simulated to multiplicative error $1/2$ unless the polynomial hierarchy collapses, for any θ which is not an integer multiple of $\pi/4$.

This is a weaker version of Lemma 57. We include it for pedagogical reasons, as it provides a different way of understanding the main theorem using MBQC techniques, and it includes a more detailed walkthrough of the hardness construction. Furthermore, it does not rely on the theorem that the Clifford group plus any non-Clifford element is universal; instead one can directly prove postselected universality by finding a qubit rotation by an irrational multiple of π .

As in the proof of Lemma 57, we will first show that CCCs can perform universal quantum computation (i.e., the class BQP) under postselection. This first step will make extensive use of ideas from Measurement-Based Quantum Computation [51]. Next, we will show that these circuits can furthermore perform PostBQP under postselection. This extension requires the inverse-free Solovay-Kitaev theorem of Sardharwalla et al. [198].

Theorem 74. Postselected CCCs can be used to simulate universal quantum computation under the choice of $U = R_Z(\theta)H$, and for any choice of θ other than integer multiples of $\pi/4$.

Proof. We will first describe the proof without reference to Measurement Based Quantum Computing (MBQC) so as to be understood by the broadest possible audience. We will then summarize the proof in MBQC language for those familiar with the area.

Our proof will make use of four gadgets to show that under postselection, we can perform arbitrary 1-qubit gates in this model. For the first gadget, consider the following quantum circuit:

$$\begin{array}{c}
 |\psi\rangle \text{ --- } \bullet \text{ --- } [H] \text{ --- } \text{meter} \text{ --- } \langle 0| \\
 |0\rangle \text{ --- } [H] \text{ --- } \bullet \text{ --- } \text{ --- } |\psi'\rangle
 \end{array} \tag{4.24}$$

Here the notation $\langle 0|$ denotes that we postselect that measurement outcome on obtaining the state $|0\rangle$, and the two-qubit gate is controlled- Z . This gadget performs teleportation [113]. One can easily calculate that $|\psi'\rangle = H|\psi\rangle$ – in other words, this gadget performs the H gate [48, 113]. Likewise, if one postselects the first outcome to be $|1\rangle$, then the gate performed is XH . By chaining these gadgets together, one can perform any product of these operations. For instance, the following circuit performs HXH :

$$\begin{array}{c}
 |\psi\rangle \text{ --- } \bullet \text{ --- } [H] \text{ --- } \text{meter} \text{ --- } \langle 1| \\
 |0\rangle \text{ --- } [H] \text{ --- } \bullet \text{ --- } [H] \text{ --- } \text{meter} \text{ --- } \langle 0| \\
 |0\rangle \text{ --- } [H] \text{ --- } \bullet \text{ --- } \text{ --- } |\psi'\rangle
 \end{array}$$

The correctness follows from the fact that the order in which quantum measurements are taken is irrelevant. By stringing together n of these, we can perform n gates from the set $\{H, XH\}$. These generate a finite set of one-qubit gates which contain the Paulis.

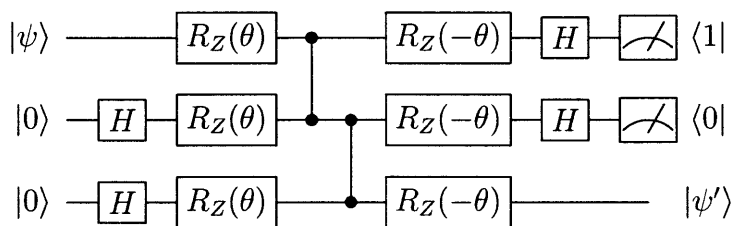
Now clearly circuits composed of these gadgets do not have the form of conjugated Clifford circuits with $U = R_Z(\theta)H$. But we can easily correct this by inserting $R_Z(\theta)$'s at the beginning of each line, and $R_Z(-\theta)$'s at the end of each line.

$$\begin{array}{c}
 |\psi\rangle \text{ --- } [R_Z(\theta)] \text{ --- } \bullet \text{ --- } [R_Z(-\theta)] \text{ --- } [H] \text{ --- } \text{meter} \text{ --- } \langle 0| \\
 |0\rangle \text{ --- } [H] \text{ --- } [R_Z(\theta)] \text{ --- } \bullet \text{ --- } [R_Z(-\theta)] \text{ --- } \text{ --- } |\psi'\rangle
 \end{array} \tag{4.25}$$

Clearly, this is equivalent to our original gadget as the Z rotations commute through and cancel. Now the gadget has the property that

- Every input line begins with $R_Z(\theta)$, and every output line ends with $R_Z(-\theta)$.
- Every ancillary input begins with $|0\rangle$ then applies $R_Z(\theta)H$.
- Every ancillary output applies $HR_Z(-\theta)$ and measures in the computational basis.
- All gates in between are Clifford.

When composing such gadgets, the $R_Z(-\theta)$ at the end of each output line cancels with the $R_Z(\theta)$ at the beginning of each input line. Hence composing gadgets with the above properties will always form a CCC. For instance our prior circuit performing HXH becomes



Thus, by simply replacing our input state $|\psi\rangle$ with the state $H|0\rangle$, and our output state with a Hadamard followed by measurement, this postselected circuit would be simulating the circuit which starts in the state $H|0\rangle$, applies HXH , then applies H and measures. Furthermore, this state will have the form of a CCC. More generally, by stringing n such gadgets together to form a CCC, clearly one can simulate any one-qubit quantum circuit where the initial state is $H|0\rangle$, one performs n gates from the set $\{H, XH\}$, and then applies H and measures.

This allows us to simulate one-qubit gates from the set $\{H, XH\}$ with postselected CCC circuits. However, such gates are not universal for a single qubit. In order to show postselected CCCs can perform universal quantum computation, we will need to find a way to simulate all single qubit gates. To do so, we will consider adding features to our gadget. So far the Clifford part of our CCCs are all commuting; let's consider adding a non-commuting one-qubit gate X to make a new gadget:

$$\begin{array}{c}
 |\psi\rangle \text{ --- } [R_Z(\theta)] \text{ --- } \bullet \text{ --- } [X] \text{ --- } [R_Z(-\theta)] \text{ --- } [H] \text{ --- } \text{meter} \text{ --- } \langle 0| \\
 |0\rangle \text{ --- } [H] \text{ --- } [R_Z(\theta)] \text{ --- } \bullet \text{ --- } [R_Z(-\theta)] \text{ --- } |\psi'\rangle
 \end{array} \tag{4.26}$$

By commuting the $R_Z(\theta)$ rightwards on both lines, and noting that

$$\text{--- } [R_Z(\theta)] \text{ --- } [X] \text{ --- } [R_Z(-\theta)] \text{ ---}$$

is equivalent to

$$\text{--- } [R_Z(2\theta)] \text{ --- } [X] \text{ ---}$$

we can see this performs the same quantum operation as

$$\begin{array}{c}
 |\psi\rangle \text{ --- } \bullet \text{ --- } [R_Z(2\theta)] \text{ --- } [X] \text{ --- } [H] \text{ --- } \text{meter} \text{ --- } \langle 0| \\
 |0\rangle \text{ --- } [H] \text{ --- } \bullet \text{ --- } |\psi'\rangle
 \end{array}$$

which since $HX = ZH$, is equivalent to

$$\begin{array}{c}
 |\psi\rangle \text{ --- } \bullet \text{ --- } [R_Z(2\theta)] \text{ --- } [H] \text{ --- } [Z] \text{ --- } \text{meter} \text{ --- } \langle 0| \\
 |0\rangle \text{ --- } [H] \text{ --- } \bullet \text{ --- } |\psi'\rangle
 \end{array} \tag{4.27}$$

By direct computation, gadget (4.26) (which is equivalent to gadget (4.27)) performs the operation $HR_Z(2\theta)$. Let us call this gate $G_0(\theta)$. Likewise, if one postselects on $|1\rangle$, one obtains the gate $G_1(\theta) = XHR_Z(2\theta)$. (This gadget is well-known in MBQC; see below).

Therefore, by applying our gadgets (4.25) and (4.26), we can create postselected CCCs to simulate the evolution of a one-qubit circuit which evolves by gates in the set $\{H, XH, G_0(\theta), G_1(\theta)\}$. Intuitively, as long as the choice of θ is not pathological, these gates will generate all one-qubit gates. Therefore we have all one-qubit gates at our disposal via these gadgets. We will prove this statement rigorously in Lemma 77, which we defer to Chapter 4.7. In fact, we show that as long as θ is not set to $k\pi/4$ for some integer k , then the set of one qubit gates generated by these gadgets

is universal on a qubit. Thus postselected CCC's (where $\theta \neq k\pi/4$) can simulate arbitrary one-qubit operations.

To prove that postselected CCC's can perform universal quantum computation, we need to show how to perform an entangling two qubit gate. We can then appeal to the result of Brylinski & Brylinski [52] and Bremner et al. [47] that any entangling two-qubit gate, plus the set of all-one-qubit gates, is universal for quantum computation. But performing entangling two-qubit gates is trivial in our setup, since the Clifford group (and the conjugated Clifford group) contains entangling two-qubit gates. For example, we can easily perform the controlled- Z gate between qubits with the following gadget:

$$(4.28)$$

This gadget clearly has the correct form, and hence composes with the gadgets (4.25) and (4.26) to form universal quantum circuits. This shows how to simulate BQP with postselected CCCs.

We can now recast this proof in the language of measurement-based quantum computing. Our result essentially follows from that fact that measuring graph states in the bases $HR_Z(2\theta)$ and H , combined with postselection, is universal for quantum computing. More formally, let E be series of controlled- Z operations that create a graph state out of $H^{\otimes n} |0\rangle^{\otimes n}$ (we will specify the cluster state later). Let $U = R_Z(\theta)H$ for some θ to be specified later. Then consider creating the CCC for the Clifford circuit $C = X^S E$, where the notation X^S denotes that we apply an X gate to some subset $S \subseteq [n]$ of the qubits. We have that

$$\begin{aligned} H^{\otimes n} R_Z(-\theta)^{\otimes n} X^S E R_Z(\theta)^{\otimes n} H^{\otimes n} |0\rangle^{\otimes n} &= H^{\otimes n} R_Z(-\theta)^{\otimes n} X^S R_Z(\theta)^{\otimes n} E H^{\otimes n} |0\rangle^{\otimes n} \\ &= H^{\otimes n} (X R_Z(2\theta))^S E H^{\otimes n} |0\rangle^{\otimes n} \\ &= H^{\otimes n} (X R_Z(2\theta))^S |\text{cluster}\rangle \\ &= \left((Z H R_Z(2\theta))^S \otimes H^{\bar{S}} \right) |\text{cluster}\rangle, \end{aligned}$$

where the first equality follows from the fact that R_Z and E commute as they are both diagonal in the Z basis, the second follows from the fact that on the lines without an X the $R_Z(\theta)$ and the $R_Z(-\theta)$ cancel, and on the lines with an X we have $R_Z(-\theta)XR_Z(\theta) = XR_Z(2\theta)$, the third follows from the fact that E is constructed such that $EH^{\otimes n}|0\rangle^{\otimes n} = |\text{cluster}\rangle$, and the fourth from the fact that $HX = ZH$. Now since we're measuring in the Z basis at the end of the circuit, the last row of Z 's can be ignored, so the circuit is equivalent to:

$$\left((HR_Z(2\theta))^S \otimes H^{\bar{S}} \right) |\text{cluster}\rangle.$$

Now we simply need to show that measurement based quantum computation with postselection on such a state is universal for quantum computing. In other words, we need to show that if we can construct a cluster state and measure some qubits in the H basis and others in the $HR_Z(2\theta)$ basis, and postselect on the outcomes, then we can perform universal quantum computation. It was previously known to be universal for MBQC if different θ 's occur on each qubit [51]. In our setup we do not have this flexibility, but we instead have the additional ability to postselect.

Universality of this model follows from the fact that by preparing an appropriate cluster state (using the standard trick to perform 1-qubit gates with MBQC), this gives us the ability to apply the one-qubit gate $HR_Z(2\theta)$ using postselection. Likewise, postselecting on $|1\rangle$ performs the operation $XHR_Z(2\theta)$. As discussed previously, by Lemma 77, as long as θ is not set to $k\pi/4$ for some integer k , this is a universal gate set on a qubit. The addition of entangling two-qubit operations on the cluster state (namely, controlled- Z) boosts this model to universality. \square

We have now shown that postselected CCCs can perform BQP under postselection. We now extend this to show they can perform $\text{PostBQP} = \text{PP}$ under postselection. This requires using the inverse-free Solovay-Kitaev algorithm of [198]. From this, the hardness result follows via known techniques [5, 48].

Theorem 75. Postselected CCCs with $U = R_Z(\theta)H$ can decide any language in $\text{PostBQP} = \text{PP}$, for any choice of θ other than integer multiples of $\pi/4$.

Proof. To prove this, we will apply Aaronson’s result that Postselected BQP circuits, denoted PostBQP, can decide any language in PP. Aaronson’s proof works by showing that a particular universal quantum gate set – namely the gate set G consisting of Toffoli, controlled-Hadamard, and one qubit gates – can decide PP under postselection.

We previously showed that our postselected CCCs can perform a different universal quantum gate G' consisting of controlled- Z , $HR_Z(2\theta)$, $XHR_Z(2\theta)$, H and XH . Therefore, in order to show that postselected CCCs can compute PP, we need to show how to simulate Aaronson’s gate set G using our gate set G' .

One difficulty is that we must be extremely accurate in our simulation of these gates. This is because postselected quantum circuits may postselect on exponentially tiny events. Therefore, in order to simulate Aaronson’s postselected circuits for PP, we will need to simulate each gate to inverse exponential accuracy.

Normally in quantum computing this simulation is handled by the Solovay-Kitaev Theorem, which roughly states that any universal gate set can simulate any other universal gate set to error ϵ with only $\text{polylog}(1/\epsilon)$ overhead. Therefore with polynomial overhead, one can obtain inverse exponential accuracy in the simulation. This is why the choice of gate set is irrelevant in the definition of PostBQP. One catch, however, is that the Solovay-Kitaev theorem requires that the gate set is closed under inversion, i.e. for any gate $g \in G$, we have $g^{-1} \in G$ as well. This is an essential part of the construction of this theorem (which makes use of group commutators). It is an open problem to remove this requirement [78, 154]. As a corollary, it is open whether or not the class PostBQP can still compute all languages in PP if the gate set used is not closed under inversion. It is possible the class could be weaker with non-inversion-closed gate sets.

Unfortunately, the gate set G' we have at our disposal is not closed under inversion. Furthermore, since we obtained the gates using postselection gadgets, it is not clear how to generate the inverses of the gadgets, as postselection is a non-reversible operation. Therefore we cannot appeal to the Solovay-Kitaev theorem to show we can compute languages in PP.

Fortunately, however, even though our gate set does not have inverses, it does have a special property – namely, our set of one qubit gates contains the Pauli group. It turns out that recently, [198] proved a Solovay-Kitaev theorem for any set of one qubit gates containing the Paulis, but which is not necessarily closed under inversion. Therefore, by this result, even though our gate set is not closed under inversion, we can still apply any one-qubit gate to inverse exponential accuracy with merely polynomial overhead. So we can apply arbitrary one-qubit gates.

It turns out this is sufficient to apply gates from Aaronson’s gate set G consisting of Toffoli, controlled-H and one qubit gates with inverse exponential accuracy. To see this, first note that it is well-known one can construct controlled-V operations for arbitrary one-qubit gates V using a finite circuit of controlled-NOT and one-qubit gates – see [180] for details. Furthermore, it is possible to construct Toffoli using a finite circuit of one qubit gates and controlled-V operations [180]. This, together with the fact that controlled-NOT is equal to controlled- Z conjugated by Hadamard on one qubit, shows that each gate in G has an exact decomposition as a finite number of controlled- Z gates and one-qubit gates. Hence, using controlled- Z gates and one-qubit gates compiled to exponential accuracy, one can obtain circuits from G with inverse exponential accuracy. Thus, our gate set G' can efficiently simulate gates from G , and hence our postselected CCCs can compute all languages in $\text{PostBQP} = \text{PP}$ as well. \square

From this, the hardness result follows via known techniques [5, 48].

Corollary 76. Conjugated Clifford circuits cannot be weakly simulated classically to multiplicative error unless the polynomial hierarchy collapses to the third level, for the choice of $U = R_Z(\theta)H$ for any θ which is not an integer multiple of $\pi/4$.

To complete our proof, we merely need to show the following lemma:

Lemma 77. So long as θ is not an integer multiple of $\pi/4$, then the gates $HR_Z(2\theta)$ and $XHR_Z(2\theta)$ are universal on a qubit. Furthermore, so long as θ is not an integer multiple of $\pi/4$, then at least one of these gates is a rotation of the Bloch sphere by an irrational multiple of π .

Proof. For convenience of notation, define $G_0 = -iHR_Z(2\theta)$ and $G_1 = -XHR_Z(2\theta)$. We will actually begin by proving something stronger: namely, that as long as θ is not an integer multiple of $\pi/4$, then one of the rotations G_0 and G_1 is by an irrational multiple of π .

We will prove this by contradiction. Suppose that both G_0 and G_1 are rotations by rational multiples of π , call their rotation angles ϕ_0 and ϕ_1 , respectively. By direct computation, first eigenvalue of G_0 is given by

$$\frac{1}{2\sqrt{2}} \left(-2\sin(\theta) - i\sqrt{6 + 2\cos(2\theta)} \right).$$

Since this must be equal to $e^{\pm i\phi_0/2}$, and by considering the real part of this equation, we have that

$$\cos(\phi_0/2) = -\frac{\sin(\theta)}{\sqrt{2}}. \quad (4.29)$$

By an identical argument, for gate G_1 we have that

$$\cos(\phi_1/2) = -\frac{\cos(\theta)}{\sqrt{2}}. \quad (4.30)$$

Squaring these terms and summing them, we obtain that

$$\cos^2(\phi_0/2) + \cos^2(\phi_1/2) = \frac{1}{2}.$$

Or, applying the fact $\cos^2 t = \frac{1+\cos 2t}{2}$ and simplifying, one can see this is equivalent to

$$\cos(\phi_0) + \cos(\phi_1) + \cos(0) = 0.$$

Since we are assuming by way of contradiction that G_0, G_1 are of finite order, we are assuming that ϕ_0, ϕ_1 are rational multiples of π . Previously, Crosby [73] and Włodarski [235] classified all possible solutions to the equation $\cos(\alpha_1) + \cos(\alpha_2) + \cos(\alpha_3)$ where each α_i are rational multiples of π . The four possible solution families to this equation (assuming without loss of generality that $0 \leq \alpha_i \leq \pi$) are [235]

- $\{\beta, \pi - \beta, \pi/2\}$ where $0 \leq \beta \leq \pi$

- $\{\delta, \frac{2\pi}{3} - \delta, \frac{2\pi}{3} + \delta\}$ where $0 \leq \delta \leq \frac{\pi}{3}$
- $\{\frac{2\pi}{5}, \frac{4\pi}{5}, \frac{\pi}{3}\}$
- $\{\frac{\pi}{5}, \frac{3\pi}{5}, \frac{2\pi}{3}\}$.

Since we have that one of our three angles is 0, the latter two cases are immediately ruled out, and we must have that the angles $\{\phi_0, \phi_1\}$ are either $\{\pi/2, \pi\}$ or $\{2\pi/3, 2\pi/3\}$. One can easily see that the first solution corresponds to $\theta = k\pi/2$ for an integer k , and the second solution corresponds to $\theta = k\pi/4$ for an odd integer k .

Therefore, so long as θ is not an integer multiple of $\pi/4$, we have a contradiction, as there are no further solutions to these equations where the ϕ_i are rational multiples of π . So if θ is set to any value other than $k\pi/4$ for an integer k , we have that at least one of the gates G_0 and G_1 is a rotation by an irrational multiple of π .

Now what remains to be shown is that the gates G_0 and G_1 are universal in the general case. This can be shown easily by the classification of continuous subgroups of $SU(2)$. The continuous subgroups of $SU(2)$ are $U(1)$ (corresponding to all rotations about one axis), $U(1) \times \mathbb{Z}_2$ (corresponding to all rotations about an axis a , plus a rotation by π through another axis perpendicular to a), and $SU(2)$. By our prior result we know that either G_0 or G_1 generates all rotations about its axis of rotation on the Bloch sphere. Therefore, if we can show that neither G_0 nor G_1 are rotations by angle π we are done, as these then must generate all of $SU(2)$. However this follows immediately from Eqs. (4.29) and (4.30), since these equations imply that we can have either $\phi_0 = \pi$ or $\phi_1 = \pi$ only when θ is a rational multiple of $\pi/2$. Hence, as long as θ is not a rational multiple of $\pi/4$, neither G_0 nor G_1 is a rotation by π , and furthermore one is a rotation by an irrational multiple of π . These gates generate a continuous group which is neither $U(1)$ nor $U(1) \times \mathbb{Z}_2$, and therefore by the above observation these generate all of $SU(2)$.

□

4.8 Open Problems

Our work leaves open a number of problems.

- What is the computational complexity of commuting CCCs? In other words, can the gate set CZ, S conjugated by a one-qubit gate U ever give rise to quantum advantage? Note that this does not follow from Bremner, Jozsa and Shepherd’s results [48], as their hardness proof uses the gate set CZ, T or CCZ, CZ, Z conjugated by one-qubit gates. If this is true, it would say that the “intersection” of CCCs and IQP remains computationally hard. One can also consider the computational power of arbitrary fragments of the Clifford group, which were classified in [115]. Perhaps by studying such fragments of the Clifford group one could achieve hardness with lower depth circuits (see additional question below).
- We showed that Clifford circuits conjugated by tensor-product unitaries are difficult to simulate classically. A natural extension of this question is: suppose your gate set consists of all two-qubit Clifford gates, conjugated by a unitary U which is *not* a tensor product of the same one-qubit gate. Can one show that all such circuits are difficult to simulate classically (say exactly)? Such a theorem could be a useful step towards classifying the power of all two-qubit gate sets.
- Generic Clifford circuits have a depth which is linear in the number of qubits [8]. In particular the lowest-depth decomposition for a generic Clifford circuit over n qubits to date has depth $14n - 4$ [168]. Such depth will be difficult to achieve in near-term quantum devices without error-correction. As a result, others have considered quantum supremacy experiments with lower-depth circuits. For instance, Bremner, Shepherd and Montanaro showed advantage for a restricted version of IQP circuits with depth $O(\log n)$ [50] with long-range gates (which becomes depth $O(n^{1/2} \log n)$ if one uses SWAP gates to simulate long-range gates using local operations on a square lattice). We leave open the problem of determining if quantum advantage can be achieved with CCCs of lower depth

(say $O(n^{1/2})$ or $O(n^{1/3})$) with local gates only.

- In order to establish quantum supremacy for CCCs, we conjectured that it is $\#P$ -hard to approximate a large fraction of the output probabilities of randomly chosen CCCs (Conjecture 69). Is it also $\#P$ -hard to exactly compute that large of a fraction of the output probabilities? This is a necessary but not sufficient condition for Conjecture 69 to be true, and we believe it may be a more approachable problem.

Chapter 5

How many qubits to reach quantum supremacy?

In the last few chapters, we described several restricted models of quantum computation that are potential candidates for a quantum supremacy demonstration. All the arguments that we have described for quantum supremacy require some sort of computational assumption related to the limitations of classical computation. One common assumption that we have used is that the polynomial hierarchy does not collapse, which leads to the conclusion that any classical simulation of certain families of quantum circuits requires time scaling worse than any polynomial in the size of the circuits. However, one limitation of this approach is that the asymptotic nature of this conclusion prevents us from calculating exactly how many qubits these quantum circuits must have for their classical simulation to be intractable on modern classical supercomputers.

The goal of this chapter is to refine the above-mentioned quantum supremacy arguments and perform a number-of-qubits calculation by imposing fine-grained versions of the non-collapse assumption. The first version, called `poly3-NSETH(a)`, states that 2^{an} time steps are required by any non-deterministic algorithm that, given a degree-3 polynomial f on n variables over the field \mathbb{F}_2 , accepts if f is not a balanced function. The second version, called `per-int-NSETH(b)` states that 2^{bn} time steps are required by any non-deterministic algorithm that, given an $n \times n$ integer matrix A

accepts if the permanent of A is nonzero. Naive, brute-force algorithms rule out either assumption when $a, b > 1$. Additionally, a non-trivial algorithm by Lokshtanov, Paturi, Tamaki, Williams and Yu [162] rules out $\text{poly3-NSETH}(a)$ for $a > 0.9965$. While improvements to their analysis might yield a better bound, we argue that a completely different approach would likely need to be developed to rule out $a = 1/2$. Taking $a = 1/2$ and $b = 0.999$ ($b = 1$ is ruled out by subexponential improvements over brute force), we conclude that Instantaneous Quantum Polynomial-Time (IQP) circuits with 180 qubits, Quantum Approximate Optimization Algorithm (QAOA) circuits with 360 qubits and boson sampling circuits (i.e. linear optical networks) with 90 photons are large enough for the task of producing samples from their output distributions up to constant multiplicative error to be intractable on current technology. This chapter is based on joint work with Alexander Dalzell, Aram Harrow and Rolando L. La Placa [76].

5.1 Motivation and outline of results

Quantum (computational) supremacy (QCS) is the goal of carrying out a computational task on a quantum computer that cannot be performed by any classical computer [188]. Ingredients of this include choosing an appropriate task, building a quantum device that can perform it, ideally verifying that it was done correctly, and finally using arguments from complexity theory to support the claim that no classical computer can do the same [123]. Recent advances indicate that the experimental ingredient might be available in the next several years, but the choice of task, its verification, and its complexity-theoretic justification remain important open theoretical research questions. In particular, based on the current status of complexity theory, establishing limitations on classical computing for the purpose of assessing how close we are to demonstrating QCS requires making conjectures, and thus we are presented with a range of choices. If we make stronger conjectures then we can use a smaller and more restricted quantum computer while ruling out the existence of more powerful classical simulation algorithms. Weaker conjectures, on the other hand, are

more defensible and can be based on more widely studied mathematical principles.

A leading example of a strong conjecture is the Quantum Threshold Assumption (QUATH) proposed by Aaronson and Chen [7], which states that there is no efficient (i.e. polynomial-time) classical algorithm that takes as input a description of a random quantum circuit C , and decides whether $|\langle 0^n | C | 0^n \rangle|^2$ is greater or less than the median of all $|\langle 0^n | C | 0^n \rangle|^2$ values, with success probability at least $\frac{1}{2} + \Omega(\frac{1}{2^n})$ over the choice of C . This conjecture gives one of the strongest possible statements about the hardness of simulating quantum circuits that is not already ruled out by known simulations.

A weaker conjecture is the statement that the polynomial hierarchy (PH) does not collapse (see Conjecture 11). Under this assumption, it has been shown that there cannot exist an efficient classical algorithm to produce samples from the output distribution of certain families of quantum circuits [5, 6, 27, 36, 37, 48, 97, 104, 119, 143, 148, 174, 217], up to constant multiplicative error. The three families we focus on in this chapter are Instantaneous Quantum Polynomial-time (IQP) circuits [48, 201], Quantum Approximate Optimization Algorithm (QAOA) circuits [94, 97], and boson sampling circuits (i.e. linear optical networks) [5], all of which are among those whose simulation is hard (see Chapter 2.8). Indeed, a key selling point for work in QCS is that it could be based not on the conjectured hardness of a particular quantum circuit family or even quantum mechanics in general, but instead on highly plausible, purely classical computational conjectures, such as the non-collapse of the PH.

However, the non-collapse of the PH is in a sense too weak of a conjecture to be practically useful. The conjecture rules out polynomial-time simulation algorithms for these families of circuits, but does not describe a concrete superpolynomial lower bound. Thus, assuming only the non-collapse of the PH would be consistent with a simulation of an n -qubit quantum system running in time $n^{f(n)}$ for an arbitrarily slowly growing function $f(n)$, say $\log \log \log \log(n)$. A stronger conjecture might lead to a requirement that simulation algorithms be exponential time, meaning that there is some constant c for which its runtime is $\geq 2^{cn}$. Even this, though, is not strong enough; it remains possible that the constant c is sufficiently small that we cannot

rule out a scenario where highly parallelized state-of-the-art classical supercomputers, which operate at as many as 10^{17} floating-point operations per second, are able to simulate any circuit that might be experimentally built in the near-term. For example, Neville et al. [178], as well as Clifford and Clifford [68] recently developed classical algorithms that produce samples from the output of boson sampling circuits, the former of which has been shown to simulate $n = 30$ photons on a standard laptop in just half an hour, contradicting the belief of many that 20 to 30 photons are sufficient to demonstrate a definitive quantum advantage over classical computation. A stronger conjecture that restricts the value of the exponential factor c , a so-called “fine-grained” conjecture, is needed to move forward on assessing the viability of QCS protocols. The framework of fine-grained complexity has gathered much interest in its own right in the last decade (see [234] for a survey), yielding unexpected connections between the fine-grained runtime of solutions to different problems.

In this chapter, we examine existing QCS arguments for IQP, QAOA, and boson sampling circuits from a fine-grained perspective. While many previous arguments [5,48,97] center on the counting complexity class PP , which can be related to quantum circuits via postselection [2], the fine-graining process runs more smoothly when we use the counting class coC=P instead. The class coC=P is the set of languages for which there exists an efficient classical probabilistic algorithm that accepts with probability exactly $1/2$ only on inputs not in the language. It can be related to quantum circuits via non-determinism: $\text{coC=P} = \text{NQP}$ [100], where NQP , a quantum analogue of NP , is the class of languages for which there exists an efficient quantum circuit that has non-zero acceptance probability only on inputs in the language. Moreover, this equality still holds when we restrict NQP to quantum computations with IQP, QAOA, or boson sampling circuits. Additionally, it is known that if coC=P were to be equal to NP , the PH would collapse to the second level [100,220]. Thus, by making the assumption that there is a problem in coC=P that does not admit a non-deterministic polynomial-time solution, i.e. $\text{coC=P} \not\subseteq \text{NP}$, we conclude that there does not exist a classical simulation algorithm that samples from the output distribution of IQP or QAOA circuits up to constant multiplicative error, for this would imply $\text{NP} = \text{NQP} = \text{coC=P}$, contradicting

the assumption.

To make a fine-grained version of this statement, we pick a specific coC=P -complete problem related to the number of zeros of degree-3 polynomials over the field \mathbb{F}_2 , which we call `poly3-NONBALANCED` and we assume that `poly3-NONBALANCED` does not have a non-deterministic algorithm running in fewer than $T(n)$ time steps for an explicit function $T(n)$. We choose $T(n) = 2^{an-1}$ for a fixed constant a and call this conjecture the degree-3 polynomial Non-deterministic Strong Exponential Time Hypothesis (`poly3-NSETH(a)`). It is clear that `poly3-NSETH(a)` is false when $a > 1$ due to the brute-force deterministic counting algorithm that iterates through each of the 2^n possible inputs to the function f . However, a non-trivial algorithm by Lokshtanov, Paturi, Tamaki, Williams and Yu (LPTWY) [162] gives a better-than-brute-force, deterministic algorithm for counting zeros to systems of degree- k polynomial that rules out `poly3-NSETH(a)` whenever $a > 0.9965$. This constant may be improvable while keeping the same basic method but, as we discuss in Chapter 5.5, we expect any such improvements to be small. Refuting `poly3-NSETH(a)` for values of a substantially below 1 would require the development of novel techniques.

Assuming `poly3-NSETH(a)`, we derive a fine-grained lower bound on the runtime for any classical simulation algorithm for QAOA and IQP circuits with n qubits. In essence, what we show is that a classical simulation algorithm that beats our lower bounds could be used as a subroutine to break `poly3-NSETH(a)`. Then, we repeat the process for boson sampling circuits with n photons by replacing `poly3-NSETH(a)` with a similar conjecture we call `per-int-NSETH(b)` involving the permanent of $n \times n$ integer-valued matrices. In this case, however, there is no known algorithm that can rule out any values of b when $b < 1$. Accordingly, the lower bound we derive on the simulation time of boson sampling circuits when we take $b = 0.999$ is essentially tight, matching the runtime of the naive simulation algorithm up to factors logarithmic in the total runtime. Very recently, a similar approach was applied to obtain lower bounds on the difficulty of computing output probabilities of quantum circuits based on the SETH conjecture [128]. Our work has the disadvantage of using a less well-studied and possibly stronger conjecture (`poly3-NSETH(a)`) but the advantage of

ruling out classical algorithms for sampling, i.e. for the same tasks performed by the quantum computer.

Our lower bound leads us to conclude that classically simulating general IQP circuits with $90/a$ qubits, QAOA circuits with $180/a$ qubits, or boson sampling circuits with $90/b$ photons would require one century for today’s fastest supercomputers, which we consider to be a good measure of intractability. We believe values for a and b leading to plausible conjectures are $a = 1/2$, which is substantially below best known better-than-brute-force algorithms, and $b = 0.999$, which is roughly equivalent to asserting that the best-known brute force algorithm is optimal up to subexponential factors. The relative factor of two in the number of qubits for QAOA circuits comes from a need for ancilla qubits in constructing a QAOA circuit to solve the `poly3-NONBALANCED` problem. However, these circuits must have 10^4 to 10^7 gates for these bounds to apply. By comparison, factoring a 1024-bit integer, which is sufficiently beyond the capabilities of today’s classical computers running best known algorithms, has been estimated to require more than 2000 qubits and on the order of 10^{11} gates using Shor’s algorithm [195].

5.2 Background

5.2.1 Counting complexity and quantum supremacy

The computational assumptions underlying our work and many previous QCS results utilize a relationship between quantum circuits and counting complexity classes that is not seen to exist for classical computation. To understand this relationship, we quickly review several definitions and key results.

Let $n \geq 1$, and $f : \{0, 1\}^n \rightarrow \{0, 1\}$ be a Boolean function. The *gap* of f is defined to be

$$\text{gap}(f) = \sum_{x \in \{0,1\}^n} (-1)^{f(x)}. \tag{5.1}$$

Note that the number of zeros of f may be written in terms of the gap, as follows:

$$|\{x \in \{0, 1\}^n : f(x) = 0\}| = \frac{1}{2}(2^n + \text{gap}(f)). \quad (5.2)$$

Various complexity classes may be defined in terms of the gap. The class $\#P$ that we introduced in Chapter 2.6 may be defined to be the class of functions $f : \{0, 1\}^* \rightarrow \mathbb{N}$ for which there exist a polynomial p and a polynomial-time Turing machine M such that for all $x \in \{0, 1\}^*$,

$$\begin{aligned} f(x) &= |\{y \in \{0, 1\}^{p(|x|)} : M(x, y) = 0\}| \\ &= \frac{1}{2}(2^{p(|x|)} + \text{gap}(M(x, \cdot))). \end{aligned} \quad (5.3)$$

Thus, $\#P$ consists of functions that *count* the number of zeros of a polynomial-time computable Boolean function.

Similarly, the class PP may be defined in terms of the gap: a language L is in PP if and only if there exist a polynomial p and a polynomial-time Turing machine M such that for all $x \in \{0, 1\}^*$,

$$\begin{aligned} x \in L &\iff |\{y \in \{0, 1\}^{p(|x|)} : M(x, y) = 0\}| \\ &< |\{y \in \{0, 1\}^{p(|x|)} : M(x, y) = 1\}| \\ &\iff \text{gap}(M(x, \cdot)) < 0. \end{aligned} \quad (5.4)$$

The class NP may be defined similarly, but where

$$\begin{aligned} x \in L &\iff |\{y \in \{0, 1\}^{p(|x|)} : M(x, y) = 1\}| \neq 0 \\ &\iff \text{gap}(M(x, \cdot)) \neq 2^{p(|x|)}. \end{aligned} \quad (5.5)$$

and the class coC=P , where

$$\begin{aligned} x \in L &\iff |\{y \in \{0, 1\}^{p(|x|)} : M(x, y) = 0\}| \\ &\neq |\{y \in \{0, 1\}^{p(|x|)} : M(x, y) = 1\}| \end{aligned}$$

$$\iff \text{gap}(M(x, \cdot)) \neq 0. \quad (5.6)$$

By interpreting M as a probabilistic algorithm and y as the random string of bits used by M , we can redefine NP , PP , and coC=P as the classes of languages for which there exists a polynomial-time Turing machine M whose acceptance probability on input x is non-zero, at least $1/2$, and not equal to $1/2$, respectively, only when x is in the language.

Of these classes, only NP is known to be part of the polynomial hierarchy (PH) (see Figure 2-3). Furthermore, the other three classes, $\#\text{P}$, PP , and coC=P , which we refer to as counting classes, are known to be hard: Toda's theorem [219] tells us that a $\#\text{P}$ or PP oracle is sufficient to solve any problem in the PH in polynomial time, and other work by Toda and Ogiwara [220] shows that there is a randomized reduction from any problem in the PH to a coC=P problem. Stated another way, if PP or coC=P were to be contained in a level of the PH , the PH would necessarily collapse, meaning that the entire PH would be contained within one of its levels. For example, if $\text{P} = \text{NP}$, then the entire PH would be equal to P , its zeroth level. The assumption that the PH does not collapse is thus a stronger version of the statement $\text{P} \neq \text{NP}$, and it is widely believed for similar reasons.

Furthermore, these counting classes can be connected to quantum circuits. As we described in Chapter 2.6.6, Aaronson showed that $\text{PP} = \text{PostBQP}$ [2] (see Theorem 22), where PostBQP is the set of problems solvable by quantum circuits that have the (unphysical) power to choose, or *postselect* the value of measurement outcomes that normally would be probabilistic. By contrast, classical circuits endowed with this same power form the class PostBPP which is known to lie in the third level of the PH [118].

The story is similar for coC=P . It was shown that $\text{coC=P} = \text{NQP}$ [100], where NQP is the quantum generalization of the class NP , defined to be the set of languages L for which there exists a polynomial-time uniformly generated family of circuits $\{C_x\}$ such that for all strings x , x is in the language L if and only if the quantum circuit C_x has a non-zero acceptance probability. This can also be thought of as PostBQP with

one-sided error. If there existed an efficient classical algorithm to produce samples from the output distribution of quantum circuits up to constant multiplicative error, then NP would be equal to NQP, and therefore to coC=P , leading to the collapse of the PH (to the second level [100]).

5.2.2 Degree-3 polynomials and the problem

poly3-NONBALANCED

The three models (IQP, QAOA and boson sampling) that we will use in this chapter are especially amenable to our analysis due to their natural connection to *specific* counting problems.

The specific counting problem we will use for our analysis of IQP and QAOA is called **poly3-NONBALANCED**. The input to the problem is a polynomial over the field \mathbb{F}_2 in n variables with degree at most 3 and no constant term. Since the only non-zero element in \mathbb{F}_2 is 1, every term in the polynomial has coefficient 1. One example could be $f(z) = z_1 + z_2 + z_1z_2 + z_1z_2z_3$. Evaluating f for a given string z to determine whether $f(z) = 0$ or $f(z) = 1$ can be done efficiently, but since there are 2^n possible strings z , the brute-force method takes exponential time to count the number of strings z for which $f(z) = 0$, or equivalently, to compute $\text{gap}(f)$ where gap is given by Eq. (5.1). LPTWY [162] gave a deterministic algorithm for computing the gap of degree-3 polynomials in time scaling slightly better than brute force, but it still has exponential time — $\text{poly}(n)2^{0.9965n}$.

The question posed by **poly3-NONBALANCED** is whether $\text{gap}(f) \neq 0$, that is, whether f has the same number of 0 and 1 outputs. Thus, **poly3-NONBALANCED** is in the class coC=P .

The problem **poly3-NONBALANCED** is a natural problem to work with because there is an elegant correspondence between degree-3 polynomials and IQP circuits involving Pauli Z gates, controlled- Z (CZ) gates, and controlled-controlled- Z (CCZ) gates

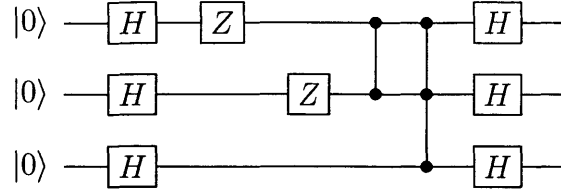


Figure 5-1: IQP circuit \mathcal{C}_f corresponding to the degree-3 polynomial $f(z) = z_1 + z_2 + z_1z_2 + z_1z_2z_3$. The unitary U_f implemented by the circuit has the property that $\langle 0|U_f|0\rangle = \text{gap}(f)/2^n$ where in this case $n = 3$.

[172]. Specifically, if f is degree 3 then let

$$U'_f = \sum_{z \in \mathbb{F}_2^n} (-1)^{f(z)} |z\rangle\langle z| \quad (5.7)$$

and let $U_f = H^{\otimes n} U'_f H^{\otimes n}$. We can implement an IQP circuit \mathcal{C}_f that evaluates to U_f as follows: if the term z_i appears in f , then within the diagonal portion of \mathcal{C}_f we perform the gate Z on qubit i ; if the term $z_i z_j$ appears, we perform the CZ gate between qubits i and j ; and if the term $z_i z_j z_k$ appears, we perform the CCZ gate between the three qubits. For example, for the polynomial $f(z) = z_1 + z_2 + z_1 z_2 + z_1 z_2 z_3$, the circuit \mathcal{C}_f is shown in Figure 5-1.

The crucial property of this correspondence is that $\langle \bar{0}|U_f|\bar{0}\rangle = \frac{\text{gap}(f)}{2^n}$, where $|\bar{0}\rangle$ is shorthand for the starting $|0\rangle^{\otimes n}$ state. This is easily seen by noting that the initial set of H gates generates the equal superposition state $|B\rangle = \sum_{x=0}^{2^n-1} |x\rangle/\sqrt{2^n}$, so $\langle \bar{0}|U_f|\bar{0}\rangle = \langle B|U'_f|B\rangle$ where U'_f is implemented by the internal diagonal portion of \mathcal{C}_f . Since U'_f applies a (-1) phase to states $|z\rangle$ for which $f(z) = 1$,

$$\langle \bar{0}|U_f|\bar{0}\rangle = \sum_{y=0}^{2^n-1} \sum_{x=0}^{2^n-1} (-1)^{f(x)} \langle y|x\rangle / 2^n = \sum_{x=0}^{2^n-1} (-1)^{f(x)} / 2^n = \text{gap}(f)/2^n.$$

Thus, $\text{gap}(f)$ can be computed by calculating the amplitude of the $|\bar{0}\rangle$ state produced by the circuit. If we define acceptance to occur when $|\bar{0}\rangle$ is measured, then the circuit \mathcal{C}_f has non-zero acceptance probability only when $\text{gap}(f) \neq 0$. This illustrates an explicit NQP algorithm for poly3-NONBALANCED, which was guaranteed to exist since $\text{NQP} = \text{coC=P}$.

Also crucial to note is that `poly3-NONBALANCED` is complete for the class `coC=P`. This is shown by adapting Montanaro’s proof [172] that computing $\text{gap}(f)$ for a degree-3 polynomial f over \mathbb{F}_2 is $\#P$ -complete. In that proof, Montanaro reduces from the problem of computing $\text{gap}(g)$ for an arbitrary boolean function g , which is $\#P$ -complete by definition. Since whether $\text{gap}(g) \neq 0$ is `coC=P`-complete by definition, and the reduction has $\text{gap}(g) \neq 0$ if and only if $\text{gap}(f) \neq 0$, this also shows that `poly3-NONBALANCED` is `coC=P`-complete. One immediate consequence of this fact is that `NIQP`, the class `NQP` restricted to quantum circuits of the IQP type, is equal to `coC=P` (and hence `NQP`), since the circuit C_f is an `NIQP` solution to a `coC=P`-complete problem.

5.2.3 The permanent and the problem `per-int-NONZERO`

In close analogy to the correspondence between degree-3 polynomials and IQP circuits composed of Z , CZ , and CCZ gates, there is a correspondence between matrix permanents and boson sampling circuits.

We have already seen in the definition of the linear optical model that any amplitude in a boson sampling circuit on n photons can be recast as the permanent of an $n \times n$ matrix, but the converse is also true: the permanent of any $n \times n$ matrix can be encoded into the amplitude of a boson sampling circuit on n photons, up to a known constant of proportionality.

To see how this works, given an $n \times n$ complex matrix A , we will construct a $2n \times 2n$ unitary matrix U_A whose upper-left $n \times n$ block is equal to cA for some $c > 0$. If we take $R = R' = (1^n, 0^n)$ (i.e. 1 repeated n times, followed by 0 repeated n times), then we will have $\text{Per}(U_{A(R,R')}) = c^n \text{Per}(A)$. Thus $\text{Per}(A)$ is proportional to a particular boson sampling amplitude with c an easily computable proportionality constant.

We can choose c to be $\leq \|A\|^{-1}$, where $\|A\|$ is the largest singular value of A . (Note that if we want the proportionality to hold uniformly across some class of A , we should choose c to satisfy $c\|A\| \leq 1$ for all A in this class.) Then $\{cA, \sqrt{I_n - c^2 A^\dagger A}\}$ are Kraus operators for a valid quantum operation, where I_n is the $n \times n$ identity matrix,

and

$$\begin{bmatrix} cA \\ \sqrt{I_n - c^2 A^\dagger A} \end{bmatrix} \quad (5.8)$$

is an isometry. We can extend this isometry to the following unitary.

$$U_A = \begin{bmatrix} cA & D \\ \sqrt{I_n - c^2 A^\dagger A} & -\frac{1}{\sqrt{I_n - c^2 A^\dagger A}} cA^\dagger D \end{bmatrix} \quad (5.9)$$

where $D = (I_n + c^2 A(I_n - c^2 A^\dagger A)^{-1} A^\dagger)^{-1/2}$, which is well-defined since the argument of the inverse square root is positive definite and Hermitian. Thus the permanent of an arbitrary $n \times n$ matrix can be encoded into a boson sampling circuit with n photons and $2n$ modes.

The matrix permanent is playing the role for boson sampling circuits that the gap of degree-three polynomials played for IQP circuits with Z , CZ , and CCZ gates; thus, it is natural to use the computational problem of determining if the permanent of an integer-valued matrix is not equal to 0, which we call `per-int-NONZERO`, in place of `poly3-NONBALANCED`.

In fact, `per-int-NONZERO` and `poly3-NONBALANCED` have several similarities. For example, like computing the number of zeros of a degree-3 polynomial, computing the permanent of an integer-valued matrix is $\#P$ -complete, a fact famously first demonstrated by Valiant [222], and later reproved by Aaronson [3] using the linear optical framework. This completeness extends to `per-int-NONZERO`, which we show in Chapter 5.4 is `coC=P`-complete by reduction from `poly3-NONBALANCED`.

Additionally, for both problems, the best known algorithm is exponential and has runtime close to or equaling 2^n . While `poly3-NONBALANCED` can be solved in $\text{poly}(n)2^{0.9965n}$ time, the best known algorithm for computing the permanent [30] requires $2^{n-\Omega(\sqrt{n/\log \log(n)})}$ deterministic time, which is only a subexponential improvement over the naive algorithm that utilizes Ryser's formula for the permanent [197] and requires at least $n2^n$ basic arithmetic operations. Using Ryser's formula is an improvement over the $O(n!)$ time steps implied by Eq. (2.60), but its scaling

is reminiscent of that required to solve a $\#\text{P}$ problem by brute force. In principle it is possible that a faster algorithm exists for `per-int-NONZERO`, where we do not care about the actual value of the permanent, only whether it is nonzero, but such methods are only known in special cases, such as nonnegative matrices.

Crucially, our construction shows that boson sampling circuits can solve `per-int-NONZERO` in non-deterministic polynomial time, since given A we have shown how to construct a circuit corresponding to unitary U_A with acceptance probability that is non-zero only when $\text{Per}(A)$ is non-zero. This shows that `NBosonP`, the linear optical analogue of `NIQP`, is equal to `coC=P` and by extension, to `NQP`.

5.3 Lower Bounds

5.3.1 For IQP Circuits

In the previous section, we described how to construct an n -qubit IQP circuit C_f corresponding to a degree-3 polynomial f over n variables such that the acceptance probability of C_f is non-zero if and only if $\text{gap}(f) \neq 0$. The number of terms in f , and hence the number of internal diagonal gates in C_f is at most

$$g_1(n) \equiv (n^3 + 5n)/6. \tag{5.10}$$

Now, suppose we had a classical algorithm that, for any q , produces samples from the output distribution of any IQP circuit with q qubits and $g_1(q)$ internal gates, up to some multiplicative error constant, in $s_1(q)$ time steps for some function s_1 . Throughout, we will assume all classical algorithms run in the Word RAM model of computation.

Using this algorithm to simulate the IQP circuit C_f generates a non-deterministic classical algorithm for `poly3-NONBALANCED` running in $s_1(n)$ time steps. That is, the classical probabilistic algorithm that results from this simulation accepts on at least one computational path only if the function f is not balanced.

Now, we impose a fine-grained version of the non-collapse assumption, which we

motivate later in the section.

Conjecture 78. (poly3-NSETH(a)) Any non-deterministic classical algorithm (in the Word RAM model of computation) that solves poly3-NONBALANCED requires in the worst case 2^{an-1} time steps, where n denotes the number of variables in the poly3-NONBALANCED instance.

In the Word RAM model with word size w , memory is infinite and basic arithmetic operations on words of length w take one time step. For concreteness, we assume that $w = \log_2(N)$ where N is the length of the input encoding the degree-3 polynomial ($N = O(g(n) \log_2(n))$). This way the words can index the locations where the input data is stored. The Word RAM model has previously been used for fine-grained analyses [234] and aims to represent how a real computer operates as faithfully as possible.

Our conjecture immediately yields a lower bound on the simulation function s_1 :

$$s_1(n) \geq 2^{an-1}. \tag{5.11}$$

This lower bound result relies on poly3-NSETH(a), which we have not yet motivated. In particular, for our qubit calculations we will take the specific value of $a = 1/2$. This value is comfortably below the best known limit $a < 0.9965$ from [162], whose algorithm is reproduced in Chapter 5.5. In Chapter 5.3.4, we attempt to provide additional motivation for poly3-NSETH($1/2$) by showing its consistency with other fine-grained conjectures.

To our knowledge, the best known upper bound on $s_1(n)$ comes from the the naive $\text{poly}(n)2^n$ simulation algorithm that updates each of the 2^n amplitudes describing the state vector after each gate is performed, so this lower bound is not tight.

5.3.2 For QAOA circuits

To perform the same analysis for QAOA circuits, we will turn the IQP circuit C_f into a QAOA circuit. The modifications required are straightforward. We set p , the

number of rounds of QAOA computation, equal to 1, and both rotation angles γ and β to $\pi/4$. The first layer of Hadamard gates in C_f is already built into the QAOA framework. To implement the Z , CZ , and CCZ gates we write $Z = \exp(-i4\gamma |1\rangle\langle 1|)$, $CZ = \exp(-i4\gamma |11\rangle\langle 11|)$, and $CCZ = \exp(-i4\gamma |111\rangle\langle 111|)$ and build our constraint Hamiltonian C accordingly: for each Z gate we add four copies of the constraint that is satisfied only when the bit acted upon is 1; for each CZ gate we add four copies of the constraint that is satisfied when both bits involved are 1; and for each CCZ gate we add four copies of the constraint that is satisfied when all three bits involved are 1. Now, the operation $\exp(-i\gamma C)$ has exactly the effect of all the Z , CZ , and CCZ gates combined.

The final step is to implement the final column of H gates, which is not built into the QAOA framework. First we write $H = H\tilde{H}^\dagger\tilde{H}$, where

$$\tilde{H} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & -i \\ -i & 1 \end{bmatrix} = \exp\left(-i\frac{\pi}{4}X\right) = \exp(-i\beta X). \quad (5.12)$$

And since $\tilde{H}^\dagger = H \exp(-i\frac{\pi}{4}Z)H$, we can replace the H gate on each qubit with $\exp(-i\gamma 2 |0\rangle\langle 0|)H\tilde{H}$. Thus, the first part of this expression can be performed by adding two copies of the $|0\rangle\langle 0|$ constraint to C . As described in [97], the H gate can be implemented by introducing an ancilla qubit and eight new constraints between the original qubit and the ancilla. The original qubit is measured and if outcome $|0\rangle$ is obtained, the state of the ancilla is H applied to the input state on the original qubit. Thus we have teleported the H gate onto the ancilla qubit within the QAOA framework. This is described in full in [97], and we reproduce the gadget in Figure 5-2.

After replacing each H gate with the gadget from Figure 5-2, every qubit begins with an H gate, is acted upon by $\exp(-i\gamma C)$, and ends with a \tilde{H} gate, which is implemented by the $\exp(-i\beta B)$ step of the QAOA framework. Thus, the resulting circuit is a QAOA circuit.

We had to introduce one ancilla per qubit in C_f , so our QAOA circuit has $2n$

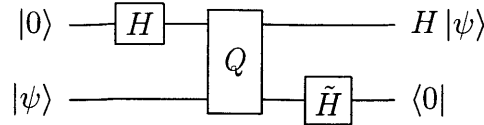


Figure 5-2: Gadget that uses an ancilla qubit to implement the H gate within the QAOA framework. Here the gate Q is the diagonal two-qubit gate $\text{diag}(1, i, 1, -1)$ which can be written as $\exp(-i\frac{\pi}{4}(6|01\rangle\langle 01| + 2|11\rangle\langle 11|))$. Thus, it can be implemented by adding 8 constraints to the constraint Hamiltonian C . The \tilde{H} gate is implemented by applying the Hamiltonian B with $\beta = \pi/4$.

qubits, instead of just n . However, it is still true that $\langle \bar{0} | V_f | \bar{0} \rangle \propto \text{gap}(f)$, where V_f is now the unitary implemented by this new QAOA circuit and $|\bar{0}\rangle$ is the state $|0\rangle^{\otimes 2n}$. Hence the acceptance probability is non-zero if and only if f is not balanced.

The circuit requires 4 constraints per term in the polynomial f , and an additional 10 constraints per qubit for the Hadamard gates at the end of the computation (2 from introducing \tilde{H} and 8 from the gadget in Figure 5-2). This yields at most

$$g_2(n) \equiv (2n^3 + 40n)/3 \tag{5.13}$$

constraints.

As in the IQP case, we suppose a classical simulation algorithm produces samples from the output distribution of QAOA circuits with q qubits and $g_2(q)$ constraints, up to multiplicative error constant, in time $s_2(q)$. Then, under the same conjecture $\text{poly3-NSETH}(a)$, we have

$$s_2(2n) \geq 2^{an-1}, \tag{5.14}$$

which simplifies to

$$s_2(n) \geq 2^{\frac{an}{2}-1}. \tag{5.15}$$

The exponentiality of this lower bound is weaker by a factor of two in comparison to the lower bound for IQP circuits in Eq. (5.11), due to the fact that one ancilla was introduced per variable to turn the circuit C_f into a QAOA circuit. However, the best known upper bound for QAOA simulation is the naive $\text{poly}(n)2^n$ brute-force algorithm, as was the case for IQP circuits. This indicates that one might be able to

eliminate the factor of two by replacing `poly3-NONBALANCED` with another problem. Such a problem should be solvable by a QAOA circuit but not by non-deterministic algorithms running much faster than brute force. We leave this for future work.

5.3.3 For boson sampling circuits

The story for boson sampling circuits is nearly identical, except using a conjecture related to the problem `per-int-NONZERO` instead of `poly3-NONBALANCED`.

Given an integer-valued $n \times n$ matrix A , we showed in the previous section how to construct a boson sampling circuit with n photons, described by unitary U_A , that has non-zero acceptance probability only when $\text{Per}(A) \neq 0$. This circuit has $2n$ modes, and hence requires at most

$$g_3(n) \equiv 2n^2 + n \tag{5.16}$$

circuit elements, that is beam splitters and phase shifters.

Paralleling our IQP and QAOA analysis, we suppose we have a classical algorithm that produces samples from the output distribution of a boson sampling circuit with q photons and $g_3(q)$ total beam splitters and phase shifters, up to some multiplicative error constant, in $s_3(q)$ time steps for some function s_3 .

Using this algorithm to simulate the boson sampling circuit described by U_A generates a non-deterministic algorithm for `per-int-NONZERO` running in $s_3(n)$ time steps.

We replace Conjecture `poly3-NSETH(a)` with the version for `per-int-NONZERO`

Conjecture 79. [`per-int-NSETH(b)`] Any non-deterministic classical algorithm (in the Word RAM model of computation) that solves `per-int-NONZERO` requires in the worst case 2^{bn-1} time steps, where n is the number of rows in the `per-int-NONZERO` instance.

Unlike `poly3-NSETH(a)`, as far as we are aware there is no known better-than-brute force algorithm ruling out the conjecture for any value $b < 1$. The algorithm in [30], which is better-than-brute-force by subexponential factors rules out $b = 1$.

This conjecture implies a lower bound on the simulation function

$$s_3(n) \geq 2^{bn-1}. \tag{5.17}$$

Producing samples from the output of boson sampling circuits naively requires one to compute the permanent for many of the amplitudes. However, in the case of a binary output, where acceptance is defined to correspond to exactly one photon configuration, only one permanent need be calculated — the one associated with the accepting configuration. Thus the asymptotic scaling of this lower bound when $b = 1 - \delta$ is essentially tight with naive simulation methods as $\delta \rightarrow 0$, since Ryser’s formula can be used to evaluate the permanent and simulate a boson sampling circuit in $O(n2^n)$ time steps.

5.3.4 Evidence for conjectures

Whereas previous quantum supremacy arguments only ruled out simulation algorithms with polynomial runtime, our analysis also rules out some algorithms with exponential runtime. These conclusions come at the expense of imposing stronger, fine-grained conjectures, but such assumptions are necessary for extracting the fine-grained lower bounds we seek.

Thus, our conjectures are necessarily less plausible than the statement that the PH does not collapse, and definitively proving our conjectures is impossible without simultaneously settling major open problems in complexity theory. However, we can give evidence for these conjectures by thinking about how one might try to refute them, and showing how they fit into the landscape of previously proposed fine-grained conjectures.

We start with poly3-NSETH(a) and discuss why certain techniques for refuting it cannot work, how current techniques fall short of refuting it for values of a significantly lower than 1, and why we should expect that completely different techniques would be needed to produce algorithms that rule out $a < 1/2$. Then, we discuss how poly3-NSETH(a) fits in consistently with other results in fine-grained complexity theory.

Finally, we discuss how $\text{per-int-NSETH}(b)$ is similar and different in these regards.

The conjecture $\text{poly3-NSETH}(a)$ asserts that determining whether a boolean function is balanced takes non-deterministic exponential time, where that boolean function takes the form of a degree-3 polynomial. It is worth noting that we can prove this conjecture with $a = 1$ for boolean functions in the black box setting, where the non-deterministic algorithm can only interact with the boolean function by querying its value on certain inputs.

Theorem 80. Let $f : \{0, 1\}^n \rightarrow \{0, 1\}$ be a boolean function. A non-deterministic algorithm with black-box access to f that accepts if and only if $|\{x : f(x) = 0\}| \neq 2^{n-1}$, that is, if and only if f is not balanced, must make at least $2^{n-1} + 1$ queries to f . Moreover, this bound is optimal.

Proof. First we prove the lower bound on the number of queries. Suppose M is a non-deterministic algorithm with black-box access to f that accepts whenever f is not balanced. Let f_0 be a Boolean function that is not balanced; thus, at least one computation path of M accepts if $f = f_0$. Choose one such path and let $S \subset \{0, 1\}^n$ be the set of queries made by M on this computation path. Suppose for contradiction that $|S| \leq 2^{n-1}$. Since at most half the possible inputs are in S , it is possible to construct another Boolean function f_1 that is balanced and agrees with f_0 on the set S . Since f_0 and f_1 agree on S , the computation that accepted when $f = f_0$ will proceed identically and accept when $f = f_1$. Thus M accepts when $f = f_1$, which is balanced, yielding a contradiction. We conclude that $|S| \geq 2^{n-1} + 1$.

We can see that it is possible for M to achieve this bound as follows: M non-deterministically chooses $2^{n-1} + 1$ of the 2^n possible inputs to f and queries f on these inputs. If all of the queries yield the same value, it accepts. Otherwise, it rejects. If f is balanced, M will reject no matter which set of queries it makes, whereas if f is not balanced, there is at least one set of $2^{n-1} + 1$ inputs on which f takes the same value and M will accept, so the algorithm succeeds. \square

Theorem 80 shows that the $\text{poly3-NSETH}(1)$ conjecture cannot be disproved using an algorithm that simply evaluates the degree-3 polynomial f for different inputs.

Indeed, the algorithm by LPTWY [162] exploits the fact that the Boolean functions are degree-3 polynomials in order to refute $\text{poly3-NSETH}(a)$ for $a > 0.9965$. Refuting $\text{poly3-NSETH}(a)$ for even smaller values of a would require more techniques that further utilize the structure associated with the poly3-NONBALANCED problem.

In fact, the algorithm in [162] is substantially more general than what is necessary for our purposes; their deterministic algorithm counts the number of solutions to a system of m degree- k polynomial equations over finite field \mathbb{F}_q . The problem poly3-NONBALANCED is concerned only with the case where $m = 1$, $k = 3$, $q = 2$, and all that matters is whether the number of zeros is equal to half the possible inputs. For this special case, the algorithm is considerably simpler, and we reproduce it in Chapter 5.5. The basic technique is as follows: we fix some fraction $(1 - \delta)$ of the n variables and call R the number of zeros of f consistent with those fixed values. We can compute in time $O(2^{0.15n+0.85\delta n})$ a representation of R as a polynomial with integer coefficients over the $(1 - \delta)n$ fixed variables. Then, (even though R has an exponential number of monomials in its representation) it is noted that one can evaluate R for all $2^{(1-\delta)n}$ possible inputs in total time $O(2^{(1-\delta)n})$, as long as $\delta < 0.0035$. By evaluating and summing R on all of its inputs, we compute the total number of zeros, and the total runtime is $O(2^{(1-\delta)n})$, which is better than brute force when we choose δ positive.

Note that this algorithm is deterministic, and giving it the power of non-determinism can only make it faster. However, by inspection of the algorithm from [162], we see no clear way for non-determinism to be directly utilized to further accelerate the algorithm. This is consistent with the finding in Theorem 80 that asymptotically speaking the best non-deterministic algorithms are no faster than the best deterministic algorithms for the NONBALANCED problem in the black-box setting. However, it is worth mentioning that a gap between best-known deterministic and non-deterministic algorithms has been observed for certain NP-hard problems, for example in [236], where the problem of determining the *unsatisfiability* of a system of m degree-2 polynomials in n variables over \mathbb{F}_2 is shown to be possible in $O(2^{n/2})$ non-deterministic time, an improvement over best-known $O(2^{0.8765n})$ deterministic solution from LPTWY

[162]. Additionally, when randomness is added to non-determinism yielding what is known as a Merlin-Arthur protocol, the unsatisfiability of boolean circuits in n variables has been shown to require only $O(2^{n/2})$ time, an improvement over the best-known deterministic $O(2^n)$. These results cast some doubt on the assumption that non-determinism is a useless resource for solving `poly3-NONBALANCED` or `per-int-NONZERO`. On the other hand, unsatisfiability is an inherently different and easier problem than those we consider since unsatisfiability is hard for NP but not for the entire PH.

Additionally, we mention that the authors of LPTWY [162] were concerned primarily with showing that better-than-brute-force algorithms were possible, perhaps leaving room for optimization of their constants. In our reproduction of their algorithm when $m = 1$, $k = 3$, and $q = 2$ in Chapter 5.5, we have followed their analysis and optimized the constants where possible yielding a slightly better runtime than what is stated explicitly in their paper.

The conclusion is that techniques exist to rule out `poly3-NSETH(1)` but not for values of a much lower than 1, even after some attempt at optimization. Moreover, we now provide evidence that drastically different techniques would need to be used if one wished to rule out `poly3-NSETH(1/2)`; that is, ruling out `poly3-NSETH(a)` when $a < 1/2$ could not be done by making only slight modifications or improvements using the same approach from [162]. Our reasoning stems from the tradeoff between the two contributions to the runtime of the algorithm: first, the computation of the polynomial representation for R and second the evaluation of R for all $2^{(1-\delta)n}$ possible inputs. When δ is smaller than 0.0035, the second contribution dominates for a total runtime $O(2^{(1-\delta)n})$. However, if this step were to be improved to allow for δ to exceed 1/2, the first contribution to the runtime would begin to dominate for a total runtime of $O(2^{0.15n+0.85\delta n}) > O(2^{n/2})$. In other words, if we try to fix fewer than half of the variables, computing the representation of R (which involves cycling through the $2^{\delta n}$ strings of unfixed variables) will necessarily take longer than evaluating R and ultimately it will be impossible to produce an algorithm with runtime below $2^{n/2}$ through this method. While this is no proof, it increases the plausibility of `poly3-`

NSETH(1/2).

Next we discuss how poly3-NSETH(a) contrasts with previously proposed fine-grained conjectures. Well-known conjectures include the Exponential Time Hypothesis (ETH), which claims that there exists some c such that no $O(2^{cn})$ time algorithm for k -SAT exists, and the Strong Exponential-Time Hypothesis (SETH) [59,132], which states that for any ϵ one can choose k large enough such that there is no $O(2^{(1-\epsilon)n})$ algorithm for k -SAT. In other words, SETH states that no algorithm for k -SAT does substantially better than the naive brute-force algorithm when k is unbounded.

There is substantial evidence for ETH and SETH, even beyond the fact that decades of research on the SAT problem have failed to refute them. For instance, SETH implies fine-grained lower bounds on problems in P that match long-established upper bounds. One example is the orthogonal vectors (OV) problem, which asks if a set of n vectors has a pair that is orthogonal. There is a brute-force $O(n^2)$ solution to OV, but $O(n^{2-\epsilon})$ is impossible for any $\epsilon > 0$ assuming SETH [231, 232]. Thus, SETH being true would provide a satisfying rationale for why attempts to find faster algorithms for problems like OV have failed. On the other hand, the refutation of SETH would imply the existence of novel circuit lower bounds [138].

There are yet more fine-grained conjectures: replacing the problem k -SAT with $\#k$ -SAT yields $\#ETH$ and $\#SETH$, the counting versions of ETH and SETH. These hypotheses have interesting consequences of their own; for example, $\#ETH$ implies that computing the permanent cannot be done in subexponential time [80]. Additionally, if k -TAUT is the question of whether a k -DNF formula is satisfied by *all* its inputs (which is coNP-complete), then the statement that no $O(2^{(1-\epsilon)n})$ algorithm exists for k -TAUT with unbounded k is called the Non-deterministic Strong Exponential Time Hypothesis (NSETH) [63]. Like SETH, NSETH's refutation would imply circuit lower bounds [63,138]. Additionally, NSETH is consistent with unconditional lower bounds that have been established in proof complexity [23,190].

The conjecture poly3-NSETH(a) is similar to NSETH in that it asserts the non-existence of non-deterministic algorithms for a problem that is hard for coNP (indeed, poly3-NONBALANCED is hard for the whole PH), and it is similar to $\#SETH$ in that it

considers a counting problem. It is different from all of these conjectures because it is not based on satisfiability formulas, but rather on degree-3 polynomials over the field \mathbb{F}_2 , a problem that has been far less studied. Additionally, `poly3-NSETH`(a) goes beyond previous conjectures to assert not only that algorithms require $O(2^{an})$ time, but that they actually require at least 2^{an-1} time steps. It is not conventional to worry about constant prefactors as we have in this analysis, but doing so is necessary to perform practical runtime estimates. On this front, our analysis is robust in the sense that if `poly3-NSETH`(a) or `per-int-NSETH`(b) were to fail by only a constant prefactor, the number of additional qubits we would estimate would increase only logarithmically in that constant.

We are unable to show that `poly3-NSETH`(a) is formally implied by any of the previously introduced conjectures. However, assuming `ETH`, we can prove that the deterministic version of `poly3-NSETH`(a) holds for at least some a , i.e. that there does not exist a deterministic $O(2^{an})$ time algorithms for `poly3-NONBALANCED`.

Theorem 81. Assuming `ETH`, there exists a constant a such that every deterministic algorithm that solves `poly3-NONBALANCED` requires $O(2^{an})$ time.

Proof. Suppose for contradiction that no such constant existed; thus for any a there is an algorithm for `poly3-NONBALANCED` running in less than $O(2^{an})$ time. We give a reduction from k -SAT to `poly3-NONBALANCED` showing that this leads to a contradiction with `ETH`.

The reduction is similar to that from [172] showing that counting the number of zeros of a degree-3 polynomial is $\#\text{P}$ -complete. Given a k -SAT instance ϕ with n variables and m clauses, we can use the sparsification lemma to assume that m is $O(n)$ [80, 132]. Then we introduce one additional variable x_{n+1} and examine the formula $\phi' = x_{n+1}(1 - \phi)$. Note that ϕ is satisfiable if and only if ϕ' is not balanced. There is a quantum circuit C made up only of $O(m)$ CCZ and $O(m)$ Hadamard gates, that computes the value of $\phi'(z)$ into an auxiliary register for any input z on the first $n + 1$ qubits. The circuit also requires $O(m)$ ancilla qubits that begin and end in the $|0\rangle$ state. As described in [172], the circuit can be associated with a degree-3 polynomial

f — the H gates are replaced by gadgets similar to that in Figure 5-2, introducing more ancilla qubits but turning the circuit into an IQP circuit — that has $O(n + m)$ variables, such that $\text{gap}(f) = \text{gap}(\phi')$. Thus, given any constant c , we can choose a small enough such that a $O(2^{an})$ time algorithm for determining whether f is not balanced implies a $O(2^{cn})$ algorithm for k -SAT. Since we assumed the existence of the former, ETH is contradicted, proving the claim. \square

A variant of this claim shows that, like computing the permanent, computing the number of zeros to a degree-3 polynomial over \mathbb{F}_2 cannot be done in subexponential time, assuming $\#$ ETH. This observation provides a link between $\text{poly3-NSETH}(a)$ and $\text{per-int-NSETH}(b)$.

In comparison to $\text{poly3-NSETH}(a)$, $\text{per-int-NSETH}(b)$ has advantages and disadvantages. There is no analogous black-box argument we can make for $\text{per-int-NSETH}(b)$. On the other hand, there is no known non-trivial algorithm that rules out the conjecture for any $b < 1$, making it possible that solving per-int-NONZERO with Ryser’s formula is essentially optimal. The possible optimality of Ryser’s formula is also bolstered by work in [139], where it is unconditionally proven that a monotone circuit requires $n(2^{n-1} - 1)$ multiplications to compute the permanent, essentially matching the complexity of Ryser’s formula. This was recently extended to show similar lower bounds on monotone circuits that estimate output amplitudes of quantum circuits [128], Of course, $\text{per-int-NSETH}(1 - \delta)$ for vanishing δ goes further and asserts that computation via Ryser’s formula is optimal even with the power of non-determinism. Thus our conjecture formalizes the statement that non-determinism cannot significantly speed up computing whether the permanent is nonzero.

5.3.5 Number of qubits to achieve quantum supremacy

We can use the lower bounds on the runtime of a hypothetical classical simulation algorithm for IQP, QAOA, and boson sampling circuits in Eqs. (5.11), (5.15), and (5.17) to estimate the minimum number of qubits required for classical simulation of these circuit models to be intractable.

The fastest supercomputers today can perform at 10^{17} FLOPs (floating-point operations per second)¹. Using our lower bounds, we can determine the number of qubits/photons q such that the lower bound on $s_i(q)$ is equal to $10^{17} \cdot 60 \cdot 60 \cdot 24 \cdot 365 \cdot 100$, the maximum number of floating-point operations today's supercomputers can perform in one century, for $i = 1, 2, 3$. For IQP circuits it is $90/a$ qubits (from Eq. (5.11)), for QAOA circuits it is $180/a$ qubits (from Eq. (5.15)), and for boson sampling circuits it is $90/b$ photons (from Eq. (5.17)). We take $a = 1/2$ and $b = 0.999$, and these estimates become 180 qubits for IQP circuits, 360 qubits for QAOA circuits, and 90 photons for boson sampling circuits. For these values of a, b , the number of circuit elements needed or the lower bound to apply is $g_1(180) = 972,000$ gates for IQP circuits, $g_2(180) = 3,890,000$ constraints for QAOA circuits, and $g_3(90) = 16,300$ beam splitters and phase shifters for boson sampling circuits.

Thus, assuming one operation in the Word RAM model of computation corresponds to one floating-point operation on a supercomputer, and assuming our conjectures $\text{poly3-NSETH}(1/2)$ and $\text{per-int-NSETH}(0.999)$, we conclude that classically simulating circuits of the sizes quoted above would take at least a century on modern classical technology, a timespan we take to be sufficiently intractable.

If, additionally, we assume that the runtime of the classical simulation algorithm grows linearly with the number of circuit elements (like, for example, the naive simulation algorithm that updates the state vector after each gate), then we can make a similar statement for circuits with many fewer gates. The cost of this reduction in gates is only a few additional qubits, due to the exponential scaling of the lower bound. We can estimate the number of qubits required by finding q such that $s_i(q)/g_i(q) = 10^{17} \cdot 60 \cdot 60 \cdot 24 \cdot 365$, the maximum number of supercomputer operations in one year, for $i = 1, 2, 3$. We conclude that an IQP circuit with 206 qubits and 100 gates, a QAOA circuit with 433 qubits and 100 constraints, and a boson sampling circuit with 97 photons and 100 linear optical elements each would require at least one century — one year per element in the circuit — to be simulated using a classical

¹A list of the fastest supercomputers is maintained at <https://www.top500.org/statistics/list/>

simulation algorithm of this type running on state-of-the-art supercomputers.

The relative factor of two in the estimate for QAOA circuits is a direct consequence of the fact that one ancilla qubit was introduced per variable in order to implement the H gates at the end of the IQP circuit C_f within the QAOA framework. This illustrates how our estimate relies on finding a natural problem for these restricted models of quantum circuits and an efficient way to solve that problem within the model. Indeed, an earlier iteration of this estimate based on the satisfiability problem instead of the degree-3 polynomial problem or matrix permanent required many ancilla qubits and led to a qubit estimate above 10,000.

5.4 Reduction from poly3-NONBALANCED to per-int-NONZERO

Valiant famously showed that computing the permanent of an integer matrix is #P-hard by reduction from #3SAT [222]. A concise reproduction of this proof can be found in [16]. The main idea for our reduction is the same, the only change being in the details of the clause and variable gadgets we use for the construction.

There is a bijective correspondence between $n \times n$ matrices and directed graphs with n vertices, where the entry A_{ij} of a matrix A corresponds to the edge weight from vertex i to vertex j in the associated graph G_A . A cycle cover of G_A is a subset of the edges of G_A forming some number of cycles in which each vertex appears in exactly one cycle. The weight of a cycle cover is the product of the weights of all the edges traversed by one of the cycles. From the definition of the permanent in Eq. (2.60), we can see that the sum of the weights of all the cycle covers of G_A is given by $\text{Per}(A)$.

It will be straightforward to convert the reduction from #3SAT to computing the permanent into a reduction from poly3-NONBALANCED to per-int-NONZERO since degree-3 polynomials and 3-CNF formulas have a common structure in the sense that both involve n variables where groups of three variables appear together in

terms/clauses.

Suppose we are given a degree-3 polynomial f with n variables and m clauses. We build a corresponding graph G_f by including one term gadget for each of the m terms and one variable gadget for each of the n variables, and then connecting them in a certain way. These gadgets are shown in Figure 5-3 and Figure 5-4. If a term of f has fewer than three variables, we can repeat one of the variables that appears in that term (e.g. $x_1x_2 = x_1x_2x_2$), and thereby assume that each term has three variables. Each term gadget has three dotted edges corresponding to the three variables that appear in that term. A variable that appears t times will have t dotted edges in its variable gadget. Thus, each dotted variable edge from some node u to node u' has a corresponding dotted edge from node v to node v' in a term gadget associated with a term in which that variable appears. Each such pair of dotted edges indicates that the nodes u, u', v and v' should be connected using the XOR gadget shown in Figure 5-5. Thus, the dotted edges are not part of the final graph. The XOR gadget has the effect of ensuring that any cycle cover of the graph uses one of the two dotted edges but not both. The effective weight of an edge connected to an XOR gadget is 4.

Every cycle cover of G_f corresponds to some setting of the variables z_1, \dots, z_n . If the cycle cover traverses the solid lines at the top of the variable gadget associated with variable z_j , then the corresponding setting has $z_j = 1$. In this case, the cycle cover cannot also traverse the dotted lines at the bottom of the z_j variable gadget. Thus, due to the XOR gadget, the cycle cover *must* traverse the dotted lines corresponding to z_j in each term gadget associated with a term in which z_j appears.

On the other hand, if the cycle cover uses the dotted lines in the z_j gadget instead of the solid lines at the top, this corresponds to $z_j = 0$, and the cycle cover cannot also traverse the edges corresponding to z_j in the term gadgets associated with terms in which z_j appears.

When all three dotted edges of a term gadget are traversed, this corresponds to all three variables in the term being set to 1. There is only one way to cycle cover the term gadget in this case, and it has a weight of -1 , excluding the factors of 4 that come from the dotted edges in the XOR gadget. Meanwhile, if at least one dotted

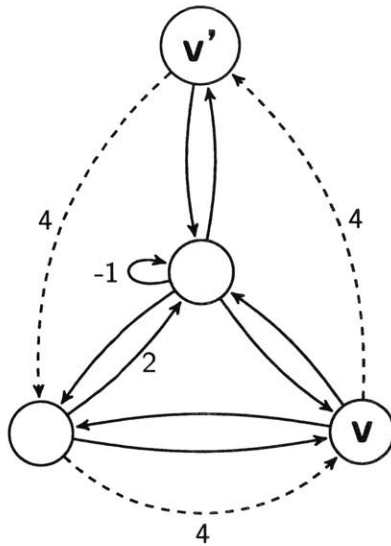


Figure 5-3: Gadget for each term in the degree-3 polynomial f . Unlabeled edges are assumed to have weight 1. The three dashed lines are connected via the XOR gadget to the dashed lines in the variable gadgets for the variables that appear in the term, as exemplified by the labeling of vertices v and v' in the context of Figure 5-5. If all three variables are true, the term gadget will contribute a cycle cover factor of -1 , excluding the factors of 4 from dotted edges. If at least one variable is false, the term will contribute a cycle cover factor of 1.

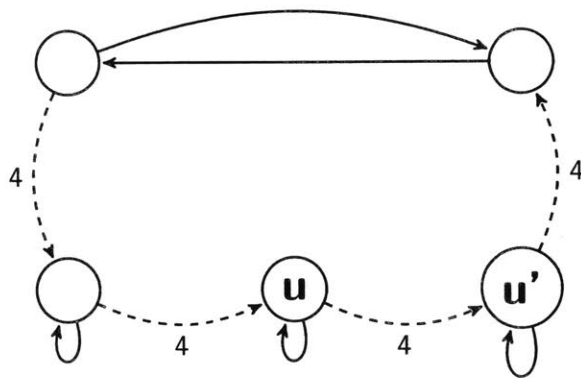


Figure 5-4: Gadget for each variable in the degree-3 polynomial f . The number of dashed lines is equal to the number of terms in which the variable appears, so this example is for a variable that appears in four terms. The dashed lines are connected to the dashed lines in the term gadget in which that variable appears via the XOR gadget, as exemplified by the labeling of vertices u and u' in the context of Figure 5-5.

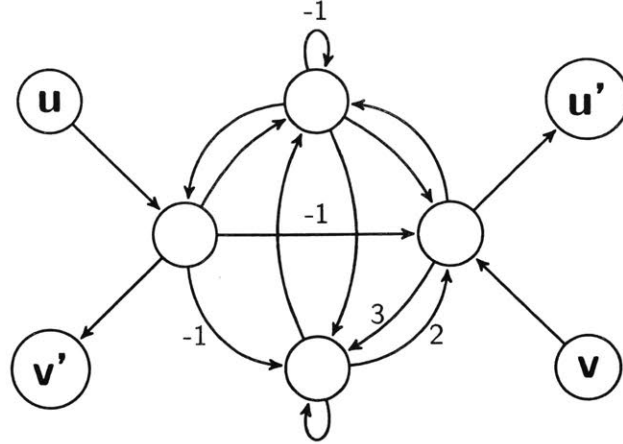


Figure 5-5: XOR gadget that connects dotted lines from node u to u' in the variable gadget with dotted lines from node v to v' in the term gadget. The effect of the XOR gadget is that any cycle cover must use either the edge from u to u' or the edge from v to v' , but not both. Each XOR gadget contributes a factor of 4 to the weight of the cycle cover.

edge in the term gadget is not traversed, the total weight of all cycle covers will contribute a factor of 1, again excluding the factors of 4. Thus, each assignment z for which $f(z) = 0$ corresponds to cycle covers that satisfy an even number of terms, with total weight 4^{3m} since exactly $3m$ XOR gadgets are involved. Each assignment for which $f(z) = 1$ corresponds to cycle covers that satisfy an odd number of terms, with total weight -4^{3m} . Thus, the total cycle cover weight of G_f , and by extension the permanent of the integer-valued matrix corresponding to G_f is non-zero if and only if $\text{gap}(f) \neq 0$. The number of vertices in G_f is a polynomial in the number of variables of f , so this completes the reduction from `poly3-NONBALANCED` to `per-int-NONZERO`. Since `poly3-NONBALANCED` is `coC=P-complete`, `per-int-NONZERO` is `coC=P-complete` as well.

5.5 Better-than-brute-force solution to poly3-NONBALANCED

LPTWY [162] gave a better-than-brute-force randomized algorithm that determines whether a system of m degree- k polynomial equations over finite field \mathbb{F}_q has a solution (i.e. a setting of the variables that makes all m polynomials equal to 0). They also derandomized this procedure to create a better-than-brute-force deterministic algorithm that *counts* the number of solutions to a system of m degree- k polynomial equations over finite field \mathbb{F}_q . Applying their deterministic algorithm for the special case $m = 1$, $k = 3$, $q = 2$ (for which it is considerably simpler) yields a deterministic solution for poly3-NONBALANCED. We give a simple reproduction of their algorithm in this case below.

Theorem 82. There is a deterministic algorithm for poly3-NONBALANCED running in time $\text{poly}(n)2^{(1-\delta)n}$ where $\delta = 0.0035$.

Proof. The algorithm beats brute force by finding a clever way to efficiently represent the number of zeros of a degree-3 polynomial with n variables when $(1 - \delta)n$ of the variables have been fixed. Then, by summing the number of zeros associated with the $2^{(1-\delta)n}$ possible settings of these variables, the algorithm computes the total number of zeros in $\text{poly}(n)2^{(1-\delta)n}$ time, which is better than brute-force $\text{poly}(n)2^n$.

First we describe the algorithm. The input is the degree-3 polynomial f , which has n variables. In the following we have $x \in \{0, 1\}^n$, and we let y be the first $(1 - \delta)n$ bits of x and a be the last δn bits of x . Following the notation from [162], we define

$$\hat{Q}_l(y, a) = 1 - (1 - f(x))^l \sum_{j=0}^{l-1} \binom{l+j-1}{j} f(x)^j. \quad (5.18)$$

In [24], it is shown that if $f(x) \equiv 0 \pmod{2}$, then $\hat{Q}_l(y, a) \equiv 0 \pmod{2^l}$ and if $f(x) \equiv 1 \pmod{2}$, then $\hat{Q}_l(y, a) \equiv 1 \pmod{2^l}$. We define

$$R_l(y) = \sum_{a \in \{0,1\}^{\delta n}} \hat{Q}_l(y, a) \quad (5.19)$$

and observe that $R_l(y)$ gives the number of settings $x \pmod{2^l}$ for which $f(x) = 1$ and the first $(1 - \delta)n$ bits of x are y .

The algorithm operates by enumerating all values of y , computing $R_l(y)$ when $l = \delta n$ (which is large enough so that the number of settings for which $f(x) = 1$ will never exceed 2^l for a given value of y), and summing all the results. This gives the total number of inputs x for which $f(x) = 1$. The algorithm rejects if this number is 2^{n-1} , and otherwise accepts.

There are two contributions to the runtime. The first is the computation of a representation of $R_{\delta n}(y)$ as a sum of monomials in $(1 - \delta)n$ variables of y with integer coefficients. Each monomial has degree at most $6\delta n - 3$. The number of possible monomials with coefficient 1 over a variables with degree at most b is

$$M(a, b) = \binom{a+b}{b} \leq (1 + a/b)^b (1 + b/a)^a \quad (5.20)$$

and, from Eq. (5.18), it is apparent that $\hat{Q}_{\delta n}(y, a)$ can be computed by a polynomially long sequence of sums or products of a pair of polynomials, where a product always includes either the polynomial $(1 - f(x))$ or $f(x)$, which have degree only 3. Thus each step in the sequence takes time at most $\text{poly}(n)M((1 - \delta)n, 6\delta n - 3)$. For a certain value of a , a polynomial number of such steps required to create a representation of $\hat{Q}_{\delta n}(y, a)$ and then $R_{\delta n}$ is the sum over $2^{\delta n}$ such representations (one for each setting of a). Thus the total time is also bounded by $\text{poly}(n)2^{\delta n}M((1 - \delta)n, 6\delta n - 3)$.

The second contribution to the runtime is the evaluation of this polynomial for all points y , given its representation computed as described. It is shown in Lemma 2.3 of [162] that this evaluation can be performed in time $\text{poly}(n)2^{(1-\delta)n}$, so long as the representation of $R_{\delta n}$ has fewer than $2^{0.15(1-\delta)n}$ monomials. This is satisfied as long as

$$M((1 - \delta)n, 6\delta n - 3) \leq 2^{0.15(1-\delta)n}, \quad (5.21)$$

which, using Eq. (5.20), can be seen to occur whenever $\delta < 0.0035$. This is an improvement on the general formula in [162], which when evaluated for $k = 3$ and

$q = 2$ yields a bound of $\delta < 0.00061$.

Assuming δ satisfies this bound, the total runtime is the sum of the two contributions, $\text{poly}(n)2^{\delta n}M((1 - \delta)n, 6\delta n - 3) + \text{poly}(n)2^{(1-\delta)n}$. The first term is smaller than $\text{poly}(n)2^{(0.85\delta+0.15)n}$, so the second term dominates, and the total runtime is $\text{poly}(n)2^{(1-\delta)n}$, proving the theorem. \square

Consider ways in which the runtime could be improved. Suppose the evaluation time were to be improved such that the polynomial $R_{\delta n}$ could be evaluated in time $\text{poly}(n)2^{(1-\delta)n}$ even when $\delta > 0.5$. With no further changes to the algorithm, the first contribution to the runtime stemming from the time required to compute the representation of $R_{\delta n}$ would now dominate and the runtime would still exceed $2^{0.5n}$. Moreover, as long as R_l is expressed as a sum over $2^{\delta n}$ terms as in Eq. (5.19), it is hard to see how any current techniques would allow this representation to be computed in less than $2^{0.5n}$ time when $\delta > 0.5$.

Stated another way, this method of beating brute force by enumerating over only a fraction $(1 - \delta)n$ of the variables and evaluating the number of solutions when those variables have been fixed in $2^{(1-\delta)n}$ time will surely break down when $\delta > 0.5$ because there will be more variables not fixed than fixed, and the preparation of the efficient representation of the number of zeros will become the slowest step.

5.6 Concluding remarks

Previous quantum supremacy arguments proved that polynomial-time simulation algorithms for certain kinds of quantum circuits would imply unexpected algorithms for classical counting problems within the polynomial-time hierarchy. We have taken this further by showing that even somewhat mild improvements over exponential-time best-known simulation algorithms would imply non-trivial and unexpected algorithms for specific counting problems in certain cases. Thus, by conjecturing that these non-trivial classical counting algorithms cannot exist, we obtain lower bounds on the runtime of the simulation algorithms. In the case of boson sampling circuits, these lower bounds are essentially asymptotically tight when the strongest form of

our conjecture is imposed.

The two versions of the conjecture that we introduce, $\text{poly3-NSETH}(a)$ and $\text{per-int-NSETH}(b)$, are fine-grained manifestations of the assumption that the PH does not collapse. While unproven, the non-collapse conjecture is extremely plausible; its refutation would entail many unexpected ramifications in complexity theory. This contrasts with the assumption that factoring has no efficient classical algorithm, which would also entail hardness of simulation but is less plausible because the consequences of its refutation on our current understanding of complexity theory would be minimal. Of course, the fine-grained nature of $\text{poly3-NSETH}(a)$ and $\text{per-int-NSETH}(b)$ makes them less plausible than the non-collapse of the PH, but they are in line with current knowledge and beliefs in fine-grained complexity theory when $a \leq 1/2$ and $b < 1$.

The main motivation for imposing these fine-grained conjectures was to make an estimate of how large quantum circuits must be to rule out practical classical simulation on state-of-the-art classical computers. Our estimate relies on $\text{poly3-NSETH}(1/2)$ and $\text{per-int-NSETH}(0.999)$, but it is somewhat robust to failure of these conjectures in the sense that if they fail in favor of mildly weaker versions, our estimate will increase only slightly. For example, replacing these conjectures with the slightly weaker $\text{poly3-NSETH}(1/2d)$ and $\text{per-int-NSETH}(1/d)$ increases the qubit estimate by only a factor of d , and replacing 2^{cn-1} time steps with $2^{cn-1}/d$ time steps in either conjecture (i.e. $c \in \{a, b\}$) increases the estimate by only $\log_2(d)$ qubits.

Our qubit estimates of fewer than 200 qubits for IQP circuits, fewer than 400 qubits for QAOA circuits, and fewer than 100 photons for boson sampling circuits are beyond current experimental capabilities but potentially within reach in the near future. Additionally, our estimate for boson sampling circuits is consistent with recently improved simulation algorithms [68, 178] that can simulate circuits with up to as many as 50 photons but would quickly become intractable for higher numbers of photons.

It is worth comparing our approach with using a fine-grained version of the conjecture that $\text{PP} \not\subseteq \Sigma_3^p$, which is the complexity theoretic conjecture proposed in Aaronson-Arkhipov [5]. To understand the range of possible fine-grained conjectures,

we might start with oracle bounds, analogous to our Theorem 80. Known oracle lower bounds for the majority function show only that Σ_3 circuits that compute the majority of an oracle function (the oracle analogue of PP) need size $\Omega(2^{n/5})$. This would correspond to taking a or b equal to $1/5$ which would increase the number of qubits required for quantum supremacy by a factor of 2.5 or 5 respectively. The proof is also more complex, involving the switching lemma [75]. Thus our approach based on coC=P instead of PP yields both a much simpler proof and a tighter bound.

A significant shortcoming in our analysis is that it only rules out simulation algorithms with multiplicative error (or with minor modification, exponentially small additive error), and not algorithms with $O(1)$ additive error. Experimental noise in real quantum systems without fault tolerance is likely to be large enough that most realistic devices could not achieve the noise rates for which our bounds apply. While some previous quantum supremacy arguments have ruled out polynomial-time simulation algorithms with additive error by imposing additional conjectures, it is unclear how to extend this to the exponential-time fine-grained setting while making a defensible conjecture.

Additionally, while the conjectures $\text{poly3-NSETH}(a)$ and $\text{per-int-NSETH}(b)$ are consistent with other fine-grained conjectures like SETH, NSETH, and $\#\text{SETH}$, it is an open question whether it is possible to prove a concrete relationship with one of these conjectures.

Finally, we conclude by noting that our analysis would likely be applicable to many other classes of quantum circuits whose efficient classical simulation entails the collapse of the PH, including DQC1 circuits (see Chapter 2.8.4) [104], various kinds of extended Clifford circuits that were discussed in Chapter 3 [143, 148], and conjugated Clifford circuits that were discussed in Chapter 4 [36].

Chapter 6

Computing quopit Clifford circuit amplitudes by the sum-over-paths technique

By the Gottesman-Knill Theorem [111], the outcome probabilities of Clifford circuits can be computed efficiently. In this chapter, we present an alternative proof of this result for quopit Clifford circuits (i.e., Clifford circuits on collections of p -level systems, where p is an odd prime) using Feynman's sum-over-paths technique [103], which allows the amplitudes of arbitrary quantum circuits to be expressed in terms of a weighted sum over computational paths. For a general quantum circuit, the sum over paths contains an exponential number of terms, and no efficient classical algorithm is known that can compute the sum. For quopit Clifford circuits, however, we show that the sum over paths takes a special form: it can be expressed as a product of Weil sums with quadratic polynomials, which can be computed efficiently. This provides a method for computing the outcome probabilities and amplitudes of such circuits efficiently, and is an application of the circuit-polynomial correspondence which relates quantum circuits to low-degree polynomials. This chapter is based on joint work with Mark D. Penney and Robert W. Spekkens [150].

6.1 Motivation and outline of results

The original proof of the Gottesman-Knill Theorem makes use of the *stabilizer* formulation of quantum mechanics, in which the state of the system at each time step is represented not by the amplitudes of the state vector, but by a set of Pauli operators which stabilize it [109]. Using this approach, the problem of computing the outcome probabilities of Clifford circuits can be reduced to computing inner products between stabilizer states [8]. The latter can be done efficiently using the stabilizer formalism, and hence the outcome probabilities can be computed efficiently.

Besides the stabilizer formalism, other techniques have been used to compute the outcome probabilities of Clifford circuits efficiently (for some examples, see [79, 81, 143, 224]). In this chapter, we present a different method from these that is explicitly based upon Feynman's sum-over-paths technique [77, 103, 186]. We restrict our attention to Clifford circuits acting on collections of quopits, i.e., p -level systems where p is an odd prime [92], and defer the treatment of general d -level systems to Chapter 7. In this approach, the amplitudes of quantum circuits are expressed in terms of a weighted sum over computational paths.

For general quantum circuits, such a sum over paths involves an exponential number of terms, and no efficient algorithm exists to compute this sum, unless $\#P$ -complete problems can also be solved efficiently. However, building on the work of Dawson *et al.* [77], we show that for quopit Clifford circuits, the sum over paths takes a special form: it can be expressed as a product of Weil sums [230] with quadratic polynomials. The problem of evaluating Weil sums explicitly is in general difficult, but for Weil sums with quadratic polynomials, the sum can be computed efficiently. This gives an efficient algorithm to compute amplitudes of quopit Clifford circuits, and therefore gives an alternative proof of the Gottesman-Knill Theorem for quopits.

The sum-over-paths technique has previously been used to answer computational complexity questions about the power of quantum computation. For example, by considering quantum circuits comprising only gates from the universal gate set of Toffoli and Hadamard gates, Dawson *et al.* provide a simple proof of the complexity-

theoretic result that $\text{BQP} \subseteq \text{PP}$ (first proved by [11]), one of the tightest ‘natural’ upper bounds for BQP [77]. Dawson et al. then ask what other universal gate sets are amenable to the sum-over-paths approach. An extension of this question, that we address in this chapter, is to ask not just about universal gate sets, but also about gate sets corresponding to restricted models of quantum computation.

Another example is the class of linear algebraic quantum circuits (which are closely related to Clifford circuits) studied by Bacon et al. [19], who noted that the sum-over-paths technique introduced by Dawson et al. implied that the computation of outcome probabilities in such circuits (assuming all registers are measured) can be reduced to the computation of Weil sums for quadratic polynomials, implying efficient classical simulation of such circuits. When specialized to the case of quopits, however, the group of unitaries implementable in a linear algebraic quantum circuit is a proper subgroup of those implementable by a quopit Clifford circuit because the generating gate set does not include the phase gate (R in Eq. (6.1), which corresponds to a phase space squeezing operation). In this respect, our result generalizes theirs. Furthermore, we here provide an explicit expression for not just the outcome probabilities, as Bacon et al. do, but the amplitudes as well.

The sum-over-paths technique makes explicit a correspondence between quantum circuits and low-degree polynomials, known as the *circuit-polynomial correspondence* [172]. This correspondence can be exploited in two different directions. In the first direction, using quantum circuit concepts, it enables one to prove classical results about polynomials. For example, the Gottesman-Knill Theorem, which is a theorem about quantum circuits, can be used to provide an efficient algorithm to compute the gap of degree-2 polynomials over \mathbb{F}_2 [172]. In the second direction, known classical results about polynomials can be used to provide algorithms for simulating classes of quantum circuits. Our result, in which we use classical results about degree-2 polynomials to simulate quopit Clifford circuits, provides an example of the second direction. Note that while the polynomials in [77] and [172] are over \mathbb{F}_2 , our results about quopit systems involve polynomials over the field \mathbb{F}_p where p is an odd prime.

The rest of the chapter is structured as follows. In Chapter 6.2, we introduce the

relevant definitions and notations and describe the problem of interest. In Chapter 6.3, we review the sum-over-paths technique and show how to construct sum-over-paths expressions for quopit Clifford circuits. In Chapter 6.4, we show how the sum-over-paths expression can be computed classically in polynomial time. In Chapter 6.5, we show how our results can be used to show that unitary operations implemented by quopit Clifford circuits are necessarily balanced.

6.2 Preliminary definitions and notation

In this chapter, p will always denote an odd prime. We shall work over the finite field \mathbb{F}_p of characteristic p , which is the set of integers modulo p . The set of $n \times n$ matrices over \mathbb{F}_p is denoted by $M_n(\mathbb{F}_p)$, and the group of invertible $n \times n$ matrices over \mathbb{F}_p is denoted by $GL_n(\mathbb{F}_p)$.

We confine our attention to quopit systems, i.e., p -level quantum systems where p is an odd prime. A quopit Clifford circuit acting on quopit systems is defined to be any circuit consisting of only the following gates, called *quopit Clifford gates*: the Fourier gate F , the phase gate R and the sum gate Σ , which are the p -level generalizations, respectively, of the Hadamard, phase and CX gates of qubit Clifford circuits defined in Chapter 2.3. They are defined as follows:

$$\begin{aligned}
 F &\equiv \frac{1}{\sqrt{p}} \sum_{s,t \in \mathbb{F}_p} \chi(st) |s\rangle\langle t|, \\
 R &\equiv \sum_{t \in \mathbb{F}_p} \chi(t(t-1)2^{-1}) |t\rangle\langle t|, \\
 \Sigma &\equiv \sum_{s,t \in \mathbb{F}_p} |s, s+t\rangle\langle s, t|,
 \end{aligned} \tag{6.1}$$

where $\chi(a) \equiv \exp(2\pi ia/p)$, and $2^{-1} = (p+1)/2$ is the inverse of 2 modulo p . For the sum gate, we write Σ_{ab} to indicate that a and b are the control and target registers respectively, i.e. $\Sigma_{ab} = \sum_{s,t \in \mathbb{F}_p} |s\rangle\langle s|_a \otimes |s+t\rangle\langle t|_b$.

For a given circuit, let n denote the number of registers (i.e. number of quopits), and N denote the number of gates. We make the following additional assumptions

about the circuit (for an example, see the circuit diagram in Figure 6-1):

- The inputs to the circuit are computational basis states $|a\rangle$, where $a \in \mathbb{F}_p^n$.
- Measurements are performed only at the end of the circuit, i.e., there are no intermediate measurements, and *all* quopits are measured at the end of the circuit. Also, measurements are performed in the computational basis. Hence, the possible measurement outcomes lie in the set \mathbb{F}_p^n . A measurement outcome of $b \in \mathbb{F}_p^n$ is associated with the computational basis vector $|b\rangle$.
- There are no extraneous quopits, i.e. every quopit is acted on by at least one gate, so that $n = O(N)$.

The problem we are interested in, which we call \mathcal{P} , is the following: given a quopit Clifford circuit acting on the input state $|a\rangle$, where $a \in \mathbb{F}_p^n$, compute the probability amplitude associated with the outcome $b \in \mathbb{F}_p^n$. Formally, \mathcal{P} may be stated as follows:

Given a description of a quopit Clifford circuit that implements the unitary U , as well as strings $a, b \in \mathbb{F}_p^n$, compute $\langle b|U|a\rangle$.

Note that a description of a quopit Clifford circuit C is a specification of the gates in C as well as the registers on which they act.

If C were allowed to be a general quantum circuit with gates chosen from some universal discrete gate set, then the problem \mathcal{P} would be $\#\text{P}$ -hard. But for the quopit Clifford circuits C that we consider, \mathcal{P} can be solved in polynomial-time. We now describe a proof of this result that is based on the sum-over-paths formulation of quopit Clifford circuits.

6.3 Constructing sum-over-paths expressions for quopit Clifford circuits

In this section, we review the sum-over-paths technique applied to quopit Clifford circuits that was introduced in Section III of Ref. [186]. Without loss of generality, we assume that each register of the Clifford circuit terminates in a Fourier gate just before it is measured (we shall refer to circuits with this property as *standard-form quopit Clifford circuits*). If this were not the case, for each register that does not terminate in a Fourier gate, we could pad the circuit by inserting 4 Fourier gates before the measurement is performed, since $F^4 = \mathbb{I}$. The Fourier gates that appear just before a measurement shall be called *terminal* Fourier gates. All other Fourier gates will be called *non-terminal*.

For a quopit Clifford circuit C with input labeled by $a = a_1 \dots a_n \in \mathbb{F}_p^n$ and measurement outcome labeled by $b = b_1 \dots b_n \in \mathbb{F}_p^n$, we shall label wires of C at every time step to create a *labeled circuit* as follows (See Figures 6-1 and 6-2 for an example):

1. Label the input wires by a_1, \dots, a_n .
2. Going from left to right of the circuit diagram for C , label the wires at each subsequent time step as follows:
 - (a) For each phase gate R and identity gate \mathbb{I} (i.e. when we have a bare wire), if the label at the input is s , then we label the output by s .
 - (b) For each sum gate Σ , if the labels at the inputs are (s, t) , then label the outputs by $(s, s + t)$. Here the first element in the pair is the control register, and the second element in the pair is the target register.
 - (c) For the l th non-terminal Fourier gate F , we introduce an auxiliary variable x_l , and regardless of the input to the Fourier gate, we label the output by x_l .
3. Label the output wires by b_1, \dots, b_n .

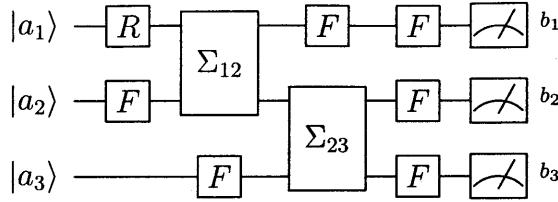


Figure 6-1: Example of a quopit Clifford circuit. As explained in the text, we can assume without loss of generality that each register ends in a Fourier gate.

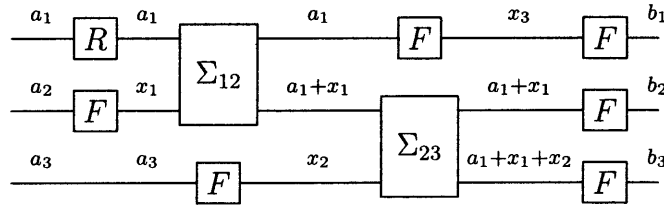


Figure 6-2: Labeled circuit corresponding to the circuit in Figure 6-1. The phase polynomial is read off to be $S(x_1, x_2, x_3) = a_2x_1 + a_3x_2 + a_1x_3 + x_3b_1 + b_2(a_1 + x_1) + b_3(a_1 + x_1 + x_2) + 2^{-1}a_1(a_1 - 1)$.

We shall associate each quopit Clifford circuit with a polynomial over \mathbb{F}_p , which is called the *phase polynomial* [106]. The variables in the phase polynomial are the auxiliary variables $x = (x_1, \dots, x_\alpha)$, where α is the number of non-terminal Fourier gates in the circuit. For each gate G in the circuit, let $\text{in}(G)$ and $\text{out}(G)$ be the input and output labels of that gate in the labeled circuit. The phase polynomial associated with a quopit Clifford circuit is the polynomial S over \mathbb{F}_p defined by (see Figure 6-2 for an example):

$$\begin{aligned}
 S(x) = & \sum_{\text{Fourier gates } F} \text{in}(F)\text{out}(F) \\
 & + \sum_{\text{phase gates } R} 2^{-1}\text{in}(R)(\text{in}(R) - 1). \tag{6.2}
 \end{aligned}$$

The theorem relating the circuit amplitudes and the phase polynomial, which appears as Theorem 3 in [186], is the following:

Theorem 83. Let C be a standard-form quopit Clifford circuit on n registers that

implements the Clifford operation U . Let α be the number of non-terminal Fourier gates and let $S(x)$ be the phase polynomial associated with C . Then,

$$\langle b|U|a\rangle = \frac{1}{p^{(n+\alpha)/2}} \sum_{x \in \mathbb{F}_p^\alpha} \chi(S(x)). \quad (6.3)$$

6.4 Evaluating the sum over paths

Using Theorem 83, the problem \mathcal{P} is reduced to evaluating the sum in Eq. (6.3). In this section, we describe how this sum may be evaluated.

First, we note that $S(x)$ is a degree-2 polynomial in the variables $x = (x_1, \dots, x_\alpha)$ as well as the variables $a_1, \dots, a_n, b_1, \dots, b_n$. This is due to the fact that $S(x)$ is a sum of terms which are at most quadratic, since the input and output labels of each gate are linear in the variables x_i, a_i and b_i . Hence, we can write $S(x)$ as

$$S(x) = x^T \Theta x + \eta^T x + \zeta = \sum_{i,j=1}^{\alpha} \Theta_{ij} x_i x_j + \sum_{i=1}^{\alpha} \eta_i x_i + \zeta, \quad (6.4)$$

where $\Theta \in M_\alpha(\mathbb{F}_p)$ can be chosen to be symmetric, $\eta \in \mathbb{F}_p^\alpha$ and $\zeta \in \mathbb{F}_p$. Note that while η and ζ are dependent on a and b , Θ is independent of (a, b) , because otherwise $S(x)$ as a polynomial in x_i, a_i and b_i would have a degree that exceeds 2.

Substituting Eq. (6.4) into Eq. (6.3) gives

$$\langle b|U|a\rangle = \frac{\chi(\zeta)}{p^{(n+\alpha)/2}} \sum_{x \in \mathbb{F}_p^\alpha} \chi(x^T \Theta x + \eta^T x). \quad (6.5)$$

The above sum can be evaluated using the following two steps.

6.4.1 Step 1: Diagonalizing Θ

In the first step, we diagonalize the matrix Θ , by making use of the following theorem:

Theorem 84. There is a polynomial-time algorithm T that when given a symmetric matrix $\Theta \in M_\alpha(\mathbb{F}_p)$ outputs an invertible matrix $L \in \text{GL}_\alpha(\mathbb{F}_p)$ such that $L^T \Theta L$ is diagonal.

Proof. The proof is essentially an algorithmic implementation of the standard proof that any quadratic form over a field that is not of characteristic 2 is diagonalizable (see Propositions 6.20 and 6.21 of [161]). We present a polynomial-time algorithm in Chapter 6.4.4. \square

By making use of Theorem 84, and the change of variables $\mu = L^T \eta$ and $x = Ly$, we can rewrite $x^T \Theta x + \eta^T x = y^T \Lambda y + \mu^T y$, where $\Lambda = L^T \Theta L$ is a diagonal matrix. By this change of variables, the sum in Eq. (6.5) becomes

$$\begin{aligned} \sum_{x \in \mathbb{F}_p^\alpha} \chi(x^T \Theta x + \eta^T x) &= \sum_{y_1, \dots, y_\alpha \in \mathbb{F}_p} \chi \left(\sum_{i=1}^{\alpha} \lambda_i y_i^2 + \mu_i y_i \right) \\ &= \prod_{i=1}^{\alpha} \sum_{y_i \in \mathbb{F}_p} \chi(\lambda_i y_i^2 + \mu_i y_i), \end{aligned} \quad (6.6)$$

where the λ_i are the diagonal entries of Λ , and the μ_i are the components of μ . We see from Eq. (6.6) that one needs only to compute the (much simpler) sums over a single variable. Such sums are called *Weil sums* [161], and we will show in the next step how to compute them.

6.4.2 Step 2: Using the exponential sum formula

The second step makes use of the following theorem about exponential sums (see Theorem 5.33 of [161]):

Theorem 85. The sum

$$\sum_{y \in \mathbb{F}_p} \chi(\lambda y^2 + \mu y)$$

can be explicitly evaluated as follows:

1. If $\lambda = \mu = 0$, then it equals p .
2. If $\lambda = 0$ and $\mu \neq 0$, then it equals 0.

3. If $\lambda \neq 0$, then it equals

$$i^{\varepsilon(p)} \chi(-4^{-1} \lambda^{-1} \mu^2) \left(\frac{\lambda}{p}\right) \sqrt{p}, \quad (6.7)$$

where $\left(\frac{\lambda}{p}\right)$ is the Legendre symbol and $\varepsilon(p) = 0$ if p is congruent to 1 mod 4 and $\varepsilon(p) = 1$ otherwise. The inverses are modular multiplicative inverses modulo p .

If we partition the indices $i \in \{1, \dots, \alpha\}$ into the following sets

$$\begin{aligned} X &= \{i \in \{1, \dots, \alpha\} | \lambda_i \neq 0\}, \\ Y &= \{i \in \{1, \dots, \alpha\} | \lambda_i = 0, \mu_i = 0\}, \\ Z &= \{i \in \{1, \dots, \alpha\} | \lambda_i = 0, \mu_i \neq 0\}, \end{aligned}$$

then by using the exponential sum formula in Theorem 85 to evaluate the sums in Eq. (6.6), we obtain

$$\begin{aligned} \langle b|U|a \rangle &= \frac{\chi(\zeta)}{p^{(n+\alpha)/2}} \left(\prod_{i \in X} i^{\varepsilon(p)} \chi(-4^{-1} \lambda_i^{-1} \mu_i^2) \left(\frac{\lambda_i}{p}\right) \sqrt{p} \right) \\ &\quad \times \left(\prod_{j \in Y} p \right) \left(\prod_{k \in Z} 0 \right) \\ &= p^{-(n+r-\alpha)/2} \delta_{0,|Z|} i^{r\varepsilon(p)} \left(\frac{\prod_{i \in X} \lambda_i}{p} \right) \\ &\quad \times \chi \left(\zeta - 4^{-1} \sum_{i \in X} \lambda_i^{-1} \mu_i^2 \right), \end{aligned} \quad (6.8)$$

where $r = |X|$ is the rank of Θ , i.e. the number of nonzero diagonal entries in Λ . Note that we used the multiplicative property of the Legendre symbol: $\left(\frac{\prod_{i \in X} \lambda_i}{p}\right) = \prod_{i \in X} \left(\frac{\lambda_i}{p}\right)$, and the fact that when $|Z| = 0$, $|X| + |Y| = \alpha$. Here, $\delta_{x,y}$ is the Kronecker delta.

Now, the Legendre symbol $\left(\frac{a}{p}\right)$ takes values in the set $\{-1, 0, 1\}$ and vanishes only when $a \equiv 0 \pmod{p}$. Since $\lambda_i \neq 0$ for $i \in X$, by definition, we get the following

simple expression for the outcome probabilities:

$$|\langle b|U|a\rangle|^2 = \frac{1}{p^{n+r-\alpha}} \delta_{0,|Z|}. \quad (6.9)$$

6.4.3 Running time

We now analyze the running time of the above procedure. The evaluation of the matrix element $\langle b|U|a\rangle$ involved four main steps: First, given a decomposition of U in terms of quopit Clifford gates, we employed the labeling procedure described in Chapter 6.3 to label the Clifford circuit. Second, from the labeled circuit, we computed the phase polynomial $S(x)$ defined by Eq. (6.2). Third, we used Theorem 84 to diagonalize $S(x)$, and fourth, we used the exponential sum formula in Theorem 85 to calculate the matrix element in Eq. (6.8).

Steps 1, 2, and 4 take time that is linear in the size of the circuit. Step 3 involves matrix diagonalization which can be carried out in polynomial time, as we show in Chapter 6.4.4. Hence, the algorithm that we give here to compute $\langle b|U|a\rangle$ runs in polynomial time.

6.4.4 Proof of Theorem 84

We shall describe a polynomial-time algorithm T that, when given a symmetric matrix $\Theta \in M_\alpha(\mathbb{F}_p)$, outputs an invertible matrix $L \in \text{GL}_\alpha(\mathbb{F}_p)$ such that $L^T \Theta L$ is diagonal. As we assumed in the main text, p denotes an odd prime. The proof is essentially an algorithmic implementation of Propositions 6.20 and 6.21 of [161].

We use the following notation in the proof: The matrix direct sum is denoted by $A \oplus B = \text{diag}(A, B)$. The $n \times n$ identity matrix is denoted by \mathbb{I}_n . The Kronecker delta is denoted by δ_{ij} . The components of a matrix $A \in M_n(\mathbb{F}_p)$ are denoted by A_{ij} , with the indices taking values $i, j = 1, \dots, n$. Likewise, the components of a vector $v \in \mathbb{F}_p^n$ are denoted by v_i , with $i = 1, \dots, n$. The inverse a^{-1} is defined to be the multiplicative inverse modulo p of $a \in \mathbb{F}_p$, i.e. the unique $b \in \mathbb{F}_p$ for which $ab \equiv 1 \pmod{p}$.

We first describe a subroutine, termed Algorithm 1, that we will need to call repeatedly.

Proof of correctness: We shall prove that Algorithm 1 works as described. If $A = 0$, then $(P, a, B) = (I, 0, 0)$, which satisfies Eq. (6.10), as required. Hence, for the rest of the proof, we shall assume that $A \neq 0$.

We first claim that $c^T A c = a$. Indeed, in the YES case in Line 5, $c^T A c = \sum_{kl} A_{kl} \delta_{ik} \delta_{il} = A_{ii} = a$, and in the NO case in Line 8, we have $A_{ii} = 0$ for all i . Hence, $c^T A c = \sum_{kl} A_{kl} (\delta_{ik} + \delta_{jk})(\delta_{il} + \delta_{jl}) = A_{ij} + A_{ji} = 2A_{ij} = a$. Note that in both cases, $a \neq 0$. Hence a^{-1} exists.

Next, consider the quadratic form f corresponding to the matrix A :

$$f(t_1, \dots, t_n) = \sum_{ij} A_{ij} t_i t_j = t^T A t. \quad (6.16)$$

Define

$$f'(y_1, \dots, y_n) = f(Cy) = \sum_{ij} A_{ij} (Cy)_i (Cy)_j. \quad (6.17)$$

By expanding Eq. (6.17), and using Eq. (6.11), Eq. (6.12) and Eq. (6.13), we obtain

$$f'(y_1, \dots, y_n) = a \left(y_1 + a^{-1} \sum_{l>1} b_l y_l \right)^2 + g(y_2, \dots, y_n). \quad (6.18)$$

Consider the matrix D defined in Eq. (6.14). It is easy to see that its inverse has components given by

$$D_{ij}^{-1} = \begin{cases} b_j a^{-1} & i = 1, j \neq 1 \\ \delta_{i,j} & \text{otherwise.} \end{cases} \quad (6.19)$$

Let $x = D^{-1}y$. Then $x_1 = y_1 + a^{-1} \sum_{l>1} b_l y_l$ and $x_i = y_i$, for all $i > 1$. Hence,

$$f'(y_1, \dots, y_n) = a x_1^2 + g(x_2, \dots, x_n). \quad (6.20)$$

Algorithm 1 Subroutine for Algorithm T

Input: a symmetric matrix $A \in M_n(\mathbb{F}_p)$, where $n \in \mathbb{Z}^+$.

Output: a 3-tuple (P, a, B) , where $P \in \text{GL}_n(\mathbb{F}_p)$, $a \in \mathbb{F}_p$ and $B \in M_{n-1}(\mathbb{F}_p)$ is a symmetric matrix, such that

$$P^T AP = a \oplus B. \quad (6.10)$$

- 1: **if** $A = 0$ **then**
- 2: output $(\mathbb{I}_n, 0, 0)$.
- 3: **else** let $a \in \mathbb{F}_p \setminus \{0\}$ and $c \in \mathbb{F}_p^n \setminus \{0\}$ be defined as follows:
- 4: Check if there exists i such that $A_{ii} \neq 0$.
- 5: **if YES then**
- 6: Let $I = \min\{i | A_{ii} \neq 0\}$.
- 7: Set $a = A_{II}$, $c_j = \delta_{Ij}$ for $j = 1, \dots, n$.
- 8: **else NO**
- 9: Let $(I, J) = \min\{(i, j) | A_{ij} \neq 0\}$. (where the minimum is taken with respect to some lexicographic ordering)
- 10: Set $a = 2A_{IJ}$, $c_k = \delta_{Ik} + \delta_{Jk}$ for $k = 1, \dots, n$.
- 11: Choose M for which $c_M \neq 0$.
- 12: Construct the nonsingular matrix $C \in \text{GL}_n(\mathbb{F}_p)$ defined by:

$$C_{ij} = \begin{cases} c_i & j = 1 \\ \delta_{i+1,j} & j \neq 1, i < M \\ 0 & j \neq 1, i = M \\ \delta_{i,j} & j \neq 1, i > M. \end{cases} \quad (6.11)$$

for $i, j = 1, \dots, n$.

- 13: Compute

$$b_l = \sum_{ij} c_i A_{ij} C_{jl}, \quad (6.12)$$

for $l = 1, \dots, n$.

- 14: Define

$$g(y_2, \dots, y_n) = \sum_{k>1, l>1} \left(\sum_{ij} A_{ij} C_{ik} C_{jl} \right) y_k y_l - a^{-1} \left(\sum_{i>1} b_i y_i \right)^2. \quad (6.13)$$

- 15: Construct the matrix $D \in \text{GL}_n(\mathbb{F}_p)$ defined by

$$D_{ij} = \begin{cases} -b_j a^{-1} & i = 1, j \neq 1 \\ \delta_{i,j} & \text{otherwise.} \end{cases} \quad (6.14)$$

16: Compute the coefficient matrix $B \in M_{n-1}(\mathbb{F}_p)$ of the quadratic form g using

$$B_{ij} = 2^{-1} [g(e_i + e_j) - g(e_i) - g(e_j)], \quad (6.15)$$

where e_i is the i th unit vector.

17: Compute $P = CD \in \text{GL}_n(\mathbb{F}_p)$.

18: Output (P, a, B) .

Now the LHS of Eq. (6.20) is equal to

$$f'(y) = f'(Dx) = f(CDx) = f(Px) = x^T P^T APx, \quad (6.21)$$

and the RHS of Eq. (6.20) is equal to

$$ax_1^2 + g(x_2, \dots, x_n) = ax_1^2 + \sum_{i,j=1}^n B_{ij}x_i x_j = x^T (a \oplus B)x, \quad (6.22)$$

where we used the fact that B is the coefficient matrix of g .

Equating Eq. (6.21) and Eq. (6.22) then gives $x^T P^T APx = x^T (a \oplus B)x$. Since this holds for all x , we obtain

$$P^T AP = a \oplus B. \quad (6.23)$$

□

We are now ready to describe the algorithm T :

Algorithm 2 Algorithm T for matrix diagonalization

Input: a symmetric matrix $\Theta \in M_\alpha(\mathbb{F}_p)$, where $\alpha \geq 1$.

Output: an invertible matrix $L \in \text{GL}_\alpha(\mathbb{F}_p)$ such that $L^T \Theta L$ is diagonal.

- 1: **if** $\alpha = 1$ **then**
 - 2: Output $L = \mathbb{I}_1 \in M_1(\mathbb{F}_p)$.
 - 3: **else** Set $\Theta_\alpha = \Theta$.
 - 4: **for** $k = \alpha, \alpha - 1, \dots, 2$ **do**
 - 5: Run Algorithm 1 on Θ_k to get output (P_k, a_k, Θ_{k-1}) .
 - 6: **for** $s = 1, \dots, \alpha$ **do**
 - 7: Set $\tilde{P}_s = \mathbb{I}_{\alpha-s} \oplus P_s$.
 - 8: Output $L = \tilde{P}_\alpha \tilde{P}_{\alpha-1} \dots \tilde{P}_2$.
-

Proof of correctness: If $\alpha = 1$, then Θ is already diagonal. Hence, setting $L = \mathbb{I}$ to

be the identity matrix gives the diagonal matrix $L^T\Theta L = \Theta$. Otherwise, we note that for the FOR loop with variable k in Line 4 of Algorithm 2, the output of Algorithm 1 on Θ_k is the 3-tuple (P_k, a_k, Θ_{k-1}) that satisfies

$$P_k^T \Theta_k P_k = a_k \oplus \Theta_{k-1}. \quad (6.24)$$

Now it is straightforward to show by induction that

$$\tilde{P}_{\alpha-s}^T \dots \tilde{P}_{\alpha-1}^T \tilde{P}_{\alpha}^T \Theta_{\alpha} \tilde{P}_{\alpha} \tilde{P}_{\alpha-1} \dots \tilde{P}_{\alpha-s} = a_{\alpha} \oplus a_{\alpha-1} \oplus \dots \oplus a_{\alpha-s} \oplus \Theta_{\alpha-s-1}, \quad (6.25)$$

for all $s = 0, 1, \dots, \alpha - 2$.

Hence, by using $s = \alpha - 2$ and $\Theta_{\alpha} = \Theta$, we get

$$\tilde{P}_2^T \dots \tilde{P}_{\alpha-1}^T \tilde{P}_{\alpha}^T \Theta \tilde{P}_{\alpha} \tilde{P}_{\alpha-1} \dots \tilde{P}_2 = a_{\alpha} \oplus a_{\alpha-1} \oplus \dots \oplus a_2 \oplus \Theta_1. \quad (6.26)$$

Since each \tilde{P}_s is invertible, their product $L = \tilde{P}_{\alpha} \tilde{P}_{\alpha-1} \dots \tilde{P}_2$ is also invertible. Therefore, writing $a_1 = \Theta_1 \in \mathbb{F}_p$, we get that

$$L^T \Theta L = \text{diag}(a_{\alpha}, \dots, a_2, a_1) \quad (6.27)$$

is a diagonal matrix.

It is straightforward to see that Algorithm 2 runs in polynomial time in the size of the matrix α .

6.5 Balancedness of quopit Clifford circuits

Quopit Clifford gates have the property that their nonzero matrix elements relative to the computation basis have the same absolute value. A gate with this property (and the matrix representing it) is called *balanced* [77, 186].

Definition 86. A gate G acting on n qudits represented by a unitary U_G is balanced if there is a constant $c \in \mathbb{R}_{\geq 0}$ and functions $f : (\mathbb{Z}_d)^n \times (\mathbb{Z}_d)^n \rightarrow \mathbb{R}$ and $g : (\mathbb{Z}_d)^n \times$

$(\mathbb{Z}_d)^n \rightarrow (\mathbb{Z}_d)^n$ such that for all $a, b \in (\mathbb{Z}_d)^n$,

$$\langle b | U_G | a \rangle = c e^{if(a,b)} \delta_{0,g(a,b)}. \quad (6.28)$$

As noted in [186], only if all gates in a circuit are balanced can one use the sum-over-paths technique to evaluate its functionality. We shall refer to the number c as the *weight* of the gate G . By convention, whenever the basis is not specified, it is assumed that the balanced property is defined with respect to the computational basis. Hence, the Fourier, phase and sum gates defined in Eq. (6.1) are balanced with weights $p^{-1/2}$, 1 and 1 respectively.

Unitary operations implemented by circuits consisting of balanced gates are not balanced in general. For example, consider a circuit consisting of the Hadamard gate H and the gate V , given by

$$V = \frac{1}{\sqrt{2}} \begin{pmatrix} e^{-i\theta} & -e^{i\theta} \\ e^{-i\theta} & e^{i\theta} \end{pmatrix}.$$

It is straightforward to check that both H and V are balanced and unitary. The product VH , however, is

$$VH = \begin{pmatrix} -i \sin(\theta) & \cos(\theta) \\ \cos(\theta) & -i \sin(\theta) \end{pmatrix},$$

which is not balanced in general. For example, when $\theta = \pi/6$, the entries of VH have absolute values $1/2$ and $\sqrt{3}/2$.

For quopit Clifford circuits, however, unitary operations implemented by quopit Clifford gates are always balanced. This is a direct consequence of Eq. (6.9). To see this, recall that Θ (defined in Eq. (6.4)) is independent of (a, b) . Hence, $r = \text{rank}(\Theta)$ is independent of (a, b) , which implies that the nonzero terms of $|\langle b | U | a \rangle|$, which are equal to $p^{-(n+r-\alpha)/2}$, are independent of (a, b) . This result is summarized in the following theorem:

Theorem 87. Let U be a unitary operation on n quopits that is implemented by

a standard-form Clifford circuit C with α non-terminal Fourier gates, and phase polynomial $S(x)$. Let r be the rank of the coefficient matrix of the quadratic form corresponding to the degree-2 terms in $S(x)$. Then U is a balanced matrix with weight $p^{-(n+r-\alpha)/2}$.

Chapter 7

Classical simulation of quantum circuits by half Gauss sums

In this chapter, we give an efficient classical algorithm to evaluate a certain class of exponential sums¹, namely the periodic, quadratic, multivariate half Gauss sums. We show that these exponential sums become $\#\text{P}$ -hard to compute when we omit either the periodic or quadratic condition. We apply our results about these exponential sums to the classical simulation of quantum circuits, and give an alternative proof of the Gottesman-Knill theorem for qudit Clifford circuits, where $d \geq 2$ is an arbitrary integer.

Our work improves on prior results in a number of ways. First, while the results of [172] and Chapter 6 are restricted to qubit and quopit systems, respectively, our results hold for all d -level systems. In doing so, we address a limitation of the approach used in Chapter 6, where the proof of the Gottesman-Knill theorem works only for d -level systems, where d is restricted to be an odd prime. Second, while previous works on tractable exponential sums are based on Gauss sums [57, 150, 161], ours are based on half Gauss sums, which are generalization of Gauss sums. Consequently, we find

¹Exponential sums have been extensively studied in number theory [127] and have a rich history that dates back to the time of Gauss [105]. Besides their applications in quantum computation, they have found numerous applications in communication theory [185], graph theory [108], coding theory [131, 210], cryptography [209, 210], algorithms [210] and many other areas of applied mathematics. We refer the reader to [151] for a summary of various applications of exponential sums.

a larger class of tractable exponential sums compared to previous works. Third, we generalize the existing definition of affine signatures [57] to arbitrary dimensions, and use our results about half Gauss sums to show that the Holant problem for the set of affine signatures is tractable. Fourth, we demonstrate the importance of a periodicity condition, which has not been previously explored, to the classical simulation of quantum circuits. This chapter is based on joint work with Kaifeng Bu [53].

7.1 Outline of results

The complexity of evaluating the exponential sum

$$Z(d, f) = \sum_{x_1, \dots, x_n \in \mathbb{Z}_d} \omega_d^{f(x_1, \dots, x_n)}, \quad (7.1)$$

where $d, n \in \mathbb{Z}^+$ are positive integers, $\omega_d = \exp(2\pi i/d)$ is a d th root of unit, and $f(x_1, \dots, x_n)$ is a polynomial with integer coefficients, has been studied in previous works. In particular, it was proved that $Z(d, f)$ can be evaluated in $\text{poly}(n)$ time when f is a quadratic polynomial. This was first proved for the case when d is a prime number [161], before being generalized to the case when d is an arbitrary positive integer [57]. On the other hand, when f is a polynomial of degree ≥ 3 , the problem of evaluating such exponential sums was proved to be $\#\text{P}$ -hard [57, 90].

In this chapter, we consider the following generalization of the above exponential sum:

$$Z_{1/2}(d, f) = \sum_{x_1, \dots, x_n \in \mathbb{Z}_d} \xi_d^{f(x_1, \dots, x_n)}. \quad (7.2)$$

Here, ξ_d is a square root of ω_d (i.e. $\xi_d^2 = \omega_d$) that additionally satisfies $\xi_d^{d^2} = 1$.

Unlike $Z(d, f)$, the sum $Z_{1/2}(d, f)$ may not be evaluable in $\text{poly}(n)$ time even when f is a quadratic polynomial—the properties of the coefficients of the quadratic polynomial f are crucial to determining the efficiency of evaluating $Z_{1/2}(d, f)$. Assuming plausible complexity assumptions, we prove that a necessary and sufficient condition

$Z_{1/2^k}(2, f)$		$\deg(f) = 1$	$\deg(f) = 2$	$\deg(f) \geq 3$
periodic	$k \geq 0$	FP	FP	#P-hard
aperiodic	$k \geq 1$	FP	#P-hard	#P-hard

Table 7.1: Hardness of computing $Z_{1/2^k}(2, f)$, where $k \geq 0$ or $k \geq 1$, and f is a polynomial function with coefficients in \mathbb{Z} and domain \mathbb{Z}_2^n . Here, ‘periodic’ means that f satisfies the periodicity condition (7.3), and ‘aperiodic’ means that f does not necessarily satisfy it. The label FP means that $Z_{1/2^k}(d, f)$ can be computed in classical polynomial time, and #P-hard means that there is no efficient classical algorithm to compute $Z_{1/2^k}(d, f)$, unless the widely-believed conjecture $\text{FP} \neq \text{\#P}$ is false.

to guarantee the efficiency of evaluating $Z_{1/2}(d, f)$ for quadratic polynomials f is a periodicity condition, which states that

$$\xi_d^{f(x_1, \dots, x_n)} = \xi_d^{f(x_1 \pmod{d}, \dots, x_n \pmod{d})}, \quad (7.3)$$

for all variables $x_1, \dots, x_n \in \mathbb{Z}$. More precisely, we prove that for quadratic polynomials f satisfying the periodicity condition, $Z_{1/2}(d, f)$ can be evaluated in $\text{poly}(n)$ time, and that without the periodicity condition, there is no efficient algorithm to evaluate $Z_{1/2}$ unless the widely-believed assumption that $\text{FP} \neq \text{\#P}$ is false. This is summarized by our main theorem:

Theorem 88. (Restatement of Theorem 94 and results in Chapter 7.4.2) Let $f \in \mathbb{Z}[x_1, \dots, x_n]$ be a quadratic polynomial over n variables x_1, \dots, x_n satisfying the periodicity condition. Then $Z_{1/2}(d, f)$ can be computed in polynomial time. If either the quadratic or periodic condition is omitted, then $(d, f) \mapsto Z_{1/2}(d, f)$ is #P-hard to compute.

We consider the case $d = 2$, and study the complexities of evaluating more general exponential sums, namely those of the form:

$$Z_{1/2^k}(2, f) = \sum_{x_1, \dots, x_n \in \mathbb{Z}_2} \omega_{2^{k+1}}^{f(x_1, \dots, x_n)}, \quad (7.4)$$

where $k \geq 0$ is an integer and f is a polynomial with n variables. Our classification

results are summarized in Table 7.1.

Next, we apply Theorem 88 to the classical simulation of Clifford circuits. In particular, we show that the output probabilities of Clifford circuits can be expressed in terms of half Gauss sums:

Theorem 89. (Simplified version of Theorem 100) Let C be an m -qudit Clifford circuit. Let $a \in \mathbb{Z}_d^m$ and $b \in \mathbb{Z}_d^k$. Then the probability of obtaining the outcome b when the first k qudits of $C|a\rangle$ are measured is given by

$$P(b|a) := \|\langle b|_{1..k} C|a\rangle_{a..m}\|^2 = \frac{1}{d^l} Z_{1/2}(d, \phi), \quad (7.5)$$

where $l \in \mathbb{Z}$ and ϕ is a quadratic polynomial that satisfies the periodicity condition (7.3). Moreover, l and ϕ can be computed efficiently.

Since half Gauss sums can be computed efficiently, Theorem 89 implies that there is an efficient strong simulation of Clifford circuits. This gives an alternative proof (that does not make use of stabilizer techniques) of the Gottesman-Knill Theorem [111].

7.2 Half Gauss sums

7.2.1 Univariate case

Given two nonzero integers a, d with $d > 0$ and $\gcd(a, d) = 1$, the Gauss sum² [156] is defined as:

$$G(a, d) = \sum_{x \in \mathbb{Z}_d} \omega_d^{ax^2}, \quad (7.6)$$

where $\omega_d = \exp(2\pi i/d)$ is a root of unity. It has been proved that the Gauss sum $G(a, d)$ can be computed in polynomial time in $\log a$ and $\log d$ [156]. Several useful properties of Gauss sum $G(a, d)$ have been provided in Chapter 7.6.2.

²also referred to as the “univariate quadratic homogeneous Gauss sum”. See Chapter 7.6.1.

In this section, we define a generalization of the Gauss sum, called the half Gauss sum³: given two nonzero integers a, d with $d > 0$ and $\gcd(a, d) = 1$, let

$$G_{1/2}(a, d) = \sum_{x \in \mathbb{Z}_d} \xi_d^{ax^2}. \quad (7.7)$$

Here, ξ_d is a chosen square root of ω_d such that $\xi_d^{d^2} = 1$. This condition is chosen so that the summation over the ring \mathbb{Z}_d is well-defined, i.e. if $x \equiv y \pmod{d}$, then $\xi_d^{ax^2} = \xi_d^{ay^2}$. Note that such a condition on ξ_d has also been used in the investigation of reflection positivity in parafermion algebra to ensure that the twisted product is well-defined [134, 135].

For $d = 1$, $G_{1/2}(a, 1) = 1$, which is trivial. So it remains to deal with the non-trivial case $d \geq 2$. ξ_d can be chosen to be $\pm\omega_{2d}$ when d is even. Here, we choose $\xi_d = -\omega_{2d} = \omega_d^{(d+1)/2}$ if d is odd and $\xi_d = \omega_{2d}$ if d is even.

We will now present properties of the half Gauss sum.

Proposition 90. The half Gauss sum satisfies the following properties:

1. If d is odd, then

$$G_{1/2}(a, d) = G(a(d+1)/2, d). \quad (7.8)$$

2. If d is even, then

$$G_{1/2}(a, d) = G_{1/2}(a(N_1 + bN_2), b)G_{1/2}(aN_2, c), \quad (7.9)$$

where $d = bc$, $\gcd(b, c) = 1$, $2|b$, and N_1 and N_2 are integers satisfying $N_1c + N_2b = 1$.

Proof.

³also referred to as the “univariate quadratic homogeneous half Gauss sum”. See Chapter 7.6.1. Also, note that our definition of “half Gauss sum” differs from that used in [28].

1. If d is odd, $\gcd((d+1)/2, d) = 1$ and $\gcd(a, d) = 1$. Thus, we have $\gcd(a(d+1)/2, d) = 1$. Therefore, we have

$$G_{1/2}(a, d) = \sum_{x \in \mathbb{Z}_d} \xi_d^{ax^2} = \sum_{x \in \mathbb{Z}_d} \omega_d^{a \frac{d+1}{2} x^2} = G(a(d+1)/2, d).$$

2. If d is even, then a must be odd, as $\gcd(a, d) = 1$. Hence,

$$G_{1/2}(a, d) = \sum_{x \in \mathbb{Z}_d} \xi_d^{ax^2} = \sum_{x \in \mathbb{Z}_d} \omega_{2d}^{ax^2}.$$

Moreover, d can be decomposed as $d = bc$, with $\gcd(b, c) = 1$. Since d is even, one of b and c has to be divisible by 2. Without loss of generality, we assume that $2|b$. Hence, $c \equiv 1 \pmod{2}$. Since $\gcd(b, c) = 1$, there exist two integers N_1 and N_2 such that $N_1c + N_2b = 1$. According to the Chinese remainder theorem, there exists an isomorphism $\mathbb{Z}_d \rightarrow \mathbb{Z}_b \times \mathbb{Z}_c$, $x \mapsto (y, z)$ with $x \equiv y \pmod{b}$ and $x \equiv z \pmod{c}$. In fact, we can choose the map $x = N_2bz + N_1cy$, which can also be written as

$$x = y + N_2b(z - y) = z + N_1c(y - z).$$

Thus,

$$\omega_{2d}^{ax^2} = \omega_{2b}^{aN_1x^2} \omega_{2c}^{aN_2x^2}.$$

Moreover,

$$\omega_{2b}^{aN_1x^2} = \omega_{2b}^{aN_1[y^2 + 2bN_2(z-y) + N_2^2b^2(y-z)^2]} = \omega_{2b}^{aN_1y^2},$$

where the last equality comes from the fact that $2|b$, and

$$\begin{aligned} \omega_{2c}^{aN_2x^2} &= \omega_{2c}^{aN_2[z^2 + 2N_1c(y-z) + N_1^2c^2(y-z)^2]} \\ &= \omega_{2c}^{aN_2z^2} \omega_{2c}^{aN_2N_1^2c^2(y-z)^2} \end{aligned}$$

$$= \omega_{2c}^{aN_2z^2} \omega_{2c}^{aN_2N_1^2c^2(y^2+z^2)}.$$

Since $\omega_{2c}^{c^2} = (-1)^c = -1$ and N_1 is odd as $N_2b + N_1c = 1$, we have

$$\begin{aligned} \omega_{2c}^{aN_2x^2} &= \omega_{2c}^{aN_2z^2} (-1)^{aN_2(y^2+z^2)} = (-\omega_{2c})^{aN_2z^2} (-1)^{aN_2y^2} \\ &= \xi_c^{aN_2z^2} (-1)^{aN_2y^2}. \end{aligned}$$

Thus,

$$\begin{aligned} \omega_{2d}^{ax^2} &= \omega_{2b}^{aN_1y^2} \xi_c^{aN_2z^2} (-1)^{aN_2y^2} = \omega_{2b}^{a(N_1+bN_2)y^2} \xi_c^{aN_2z^2} \\ &= \xi_b^{a(N_1+bN_2)y^2} \xi_c^{aN_2z^2}. \end{aligned}$$

Since $c(N_1 + bN_2) + b(1 - c)N_2 = 1$, we have $\gcd(N_1 + bN_2, b) = 1$. Thus, $\gcd(a(N_1 + bN_2), b) = 1$. Besides, $\gcd(aN_2, c) = 1$. Therefore, we find that

$$\begin{aligned} G_{1/2}(a, d) &= \sum_{y \in \mathbb{Z}_b, z \in \mathbb{Z}_c} \xi_b^{a(N_1+bN_2)y^2} \xi_c^{aN_2z^2} \\ &= G_{1/2}(a(N_1 + bN_2), b) G_{1/2}(aN_2, c). \end{aligned}$$

□

Now, any even number d can always be decomposed into $d = 2^m c$, where $m \geq 1$ and c is odd. It is straightforward to see that

$$G_{1/2}(a, d) = G_{1/2}(a(N_1 + 2^m N_2), 2^m) G_{1/2}(aN_2, c),$$

where $N_2 2^m + N_1 c = 1$. As c is odd, it can be rewritten as a Gauss sum by Proposition 90. So we need only to evaluate the half Gauss sum for $d = 2^m$, i.e., $G_{1/2}(a, 2^m)$.

Proposition 91. If $m \geq 3$, then

$$G_{1/2}(a, 2^m) = 2G_{1/2}(a, 2^{m-2}). \quad (7.10)$$

Moreover,

$$G_{1/2}(a, 2) = 1 + i^a, \quad (7.11)$$

$$G_{1/2}(a, 2^2) = 2\omega_8^a. \quad (7.12)$$

Proof. First, $G_{1/2}(a, 2)$ and $G_{1/2}(a, 2^2)$ can be obtained by direct calculation.

Second, for $m \geq 3$,

$$\begin{aligned} G_{1/2}(a, 2^m) &= \sum_{x \in [2^m]} \omega_{2^{m+1}}^{ax^2} \\ &= \sum_{x \in [2^{m-1}]} \left[\omega_{2^{m+1}}^{ax^2} + \omega_{2^{m+1}}^{a(x+2^{m-1})^2} \right] \\ &= \sum_{x \in [2^{m-1}]} \omega_{2^{m+1}}^{ax^2} \left[1 + \omega_{2^{m+1}}^{a2^m x + a2^{2m-2}} \right] \\ &= \sum_{x \in [2^{m-1}]} \omega_{2^{m+1}}^{ax^2} [1 + (-1)^x] \\ &= \sum_{y \in [2^{m-2}]} \omega_{2^{m+1}}^{a(2y)^2} [1 + (-1)^{2y}] \\ &= 2 \sum_{y \in [2^{m-2}]} \omega_{2^{m+1}}^{4ay^2} = 2 \sum_{y \in [2^{m-2}]} \omega_{2^{m-1}}^{ay^2} \\ &= 2G_{1/2}(a, 2^{m-2}). \end{aligned}$$

□

Based on the above properties of the half Gauss sum $G_{1/2}(\cdot, \cdot)$ and the fact that the Gauss sum $G(\cdot, \cdot)$ can be calculated in $\text{poly}(\log a, \log d)$ -time, we obtain the following corollary:

Corollary 92. Given two nonzero integers a, d with $d > 0$ and $\gcd(a, d) = 1$, the half Gauss sum can be calculated in $\text{poly}(\log a, \log d)$ time.

Note that we chose $\xi = \omega_{2d}$ for all even numbers d . However, ξ_d could also have been chosen to be $-\omega_{2d}$ for all even numbers d . This case is similar to the case $\xi = \omega_{2d}$, and we include a discussion of this in Chapter 7.6.3.

7.2.2 Multivariate case

In this section, we consider a generalization of the Gauss sum (7.6) to the multivariate case:

$$Z(d, f) = \sum_{x_1, \dots, x_n \in \mathbb{Z}_d} \omega_d^{f(x_1, \dots, x_n)}, \quad (7.13)$$

where each x_i is summed over the finite ring \mathbb{Z}_d , and $f(x_1, \dots, x_n)$ is a quadratic polynomial with integer coefficients. The multivariate quadratic Gauss sum (7.13) has been proved to be evaluable in polynomial time [57].

Next, we consider an analogous multivariate generalization of the half Gauss sum:

$$Z_{1/2}(d, f) = \sum_{x_1, \dots, x_n \in \mathbb{Z}_d} \xi_d^{f(x_1, \dots, x_n)}, \quad (7.14)$$

where $f(x_1, \dots, x_n) = \sum_{i \leq j \in [n]} \alpha_{ij} x_i x_j + \sum_{i \in [n]} \beta_i x_i + \gamma_0$ is a quadratic polynomial with integer coefficients. Unlike $Z(d, f)$, the half Gauss sum $Z_{1/2}(d, f)$ may not be efficiently evaluable even for quadratic polynomials. It turns out that the existence of an efficient algorithm depends on a periodicity condition that we will now describe.

We say that a polynomial f satisfies the *periodicity condition*⁴ if

$$\xi_d^{f(x_1, \dots, x_n)} = \xi_d^{f(x_1 \pmod{d}, \dots, x_n \pmod{d})}, \quad (7.16)$$

for all variables $x_1, \dots, x_n \in \mathbb{Z}$. This periodicity condition can also be regarded as the well-definedness condition of $Z_{1/2}$ on \mathbb{Z}_d . If d is an odd number, then $\xi_d = -\omega_{2d}$, i.e., $\xi_d^d = 1$, which implies that the periodicity condition can always be satisfied for odd d . However, the periodicity condition may not always be satisfied in the case of even d .

Proposition 93. Let d be even, and let $f(x_1, \dots, x_n) = \sum_{i \leq j \in [n]} \alpha_{ij} x_i x_j + \sum_{i \in [n]} \beta_i x_i +$

⁴More generally, we say that a function $g : \mathbb{Z}^n \rightarrow \mathbb{C}$ is *periodic* with period d if

$$g(x_1, \dots, x_n) = g(x_1 \pmod{d}, \dots, x_n \pmod{d}) \quad (7.15)$$

for all variables $x_1, \dots, x_n \in \mathbb{Z}$.

γ_0 , be a quadratic polynomial. Then, f satisfies the periodicity condition if and only if the cross terms α_{ij} ($i < j$) and linear terms β_i are all even.

Proof. It is easy to verify that the quadratic polynomial f satisfies the periodicity condition if all the cross terms α_{ij} ($i < j$) and linear terms β_i are even.

In the other direction, if f satisfies the periodicity condition, then $\xi_d^{f(x_1, \dots, x_n)} = \xi_d^{f(x_1 \pmod{d}, \dots, x_n \pmod{d})}$ for any $x_1, \dots, x_d \in \mathbb{Z}$. Thus, for any i ,

$$\xi_d^{\alpha_{ii}x_i^2 + \beta_i x_i} = \xi_d^{\alpha_{ii}(x_i+d)^2 + \beta_i(x_i+d)},$$

for any $x_i \in \mathbb{Z}$ by choosing $x_j = 0$ if $j \neq i$. Besides, ξ_d satisfies the conditions $\xi_d^{2d} = 1$ and $\xi_d^{d^2} = 1$. Thus, $\xi_d^{\beta_i d} = (-1)^{\beta_i} = 1$, which implies that β_i is even. Due to the arbitrary choice of i , all linear terms β_i are even. Besides, for any fixed i and j with $i < j$, we can choose $x_k = 0$ for any $k \neq i, j$. This gives

$$\xi_d^{\alpha_{ii}x_i^2 + \alpha_{jj}x_j^2 + \alpha_{ij}x_i x_j + \beta_i x_i + \beta_j x_j} = \xi_d^{\alpha_{ii}(x_i+d)^2 + \alpha_{jj}x_j^2 + \alpha_{ij}(x_i+d)x_j + \beta_i(x_i+d) + \beta_j x_j},$$

for any $x_i, x_j \in \mathbb{Z}$. This implies that α_{ij} is even. Since i, j were arbitrarily chosen, all the cross terms α_{ij} are even. \square

The periodicity condition of the polynomial f plays an important role in the efficient evaluation of the exponential sum $Z_{1/2}$. We denote the set of quadratic polynomials satisfying the periodic condition by $\mathcal{F}_2^{\text{p.c.}}$. For any quadratic polynomial f satisfying this periodicity condition, the exponential sum $Z_{1/2}(d, f)$ can be evaluated in polynomial time given the description of f .

Theorem 94. If $f \in \mathcal{F}_2^{\text{p.c.}}$ is a quadratic polynomial satisfying the periodicity condition, then $Z_{1/2}(d, f)$ can be evaluated in polynomial time.

Proof. Consider the expression

$$f(x_1, \dots, x_n) = \sum_{i \leq j \in [n]} \alpha_{ij} x_i x_j + \sum_{i \in [n]} \beta_i x_i + \gamma_0,$$

with the cross term α_{ij} ($i < j$) and linear term β_i being even. We may assume that $\gamma_0 = 0$, as it only contributes an additive constant term to $Z_{1/2}(d, f)$.

Case (i): All diagonal terms α_{ii} are even. In this case, $Z_{1/2}(d, f) = Z(d, f/2)$, which can be evaluated in polynomial time [57].

Case (ii): There exists at least one diagonal term α_{ii} that is odd.

Case (iia): d is odd. Then, $\xi_d = \omega_d^{(d+1)/2}$. Thus, $Z_{1/2}(d, f) = Z(d, \frac{d+1}{2}f)$, which can be evaluated in polynomial time [57].

Case (iib): $d = 2^m$. Then, $\xi_d = \omega_{2d}$. Since there exists at least one diagonal term α_{ii} that is odd, we assume that α_{11} is odd without loss of generality. Since α_{11} is odd, it is invertible in \mathbb{Z}_{2d} with $2d = 2^{m+1}$. We can rewrite the quadratic polynomial f to separate the term involving x_1 :

$$f(x_1, \dots, x_n) = \alpha_{11}[x_1^2 + x_1 f_1(\hat{x}_1, x_2, \dots, x_n)] + f_2(\hat{x}_1, x_2, \dots, x_n),$$

where \hat{x}_1 denotes that there is no x_1 in the polynomial. Here, f_1 is a linear function over $n - 1$ variables $\{x_2, \dots, x_n\}$ with

$$f_1(\hat{x}_1, x_2, \dots, x_n) = \sum_{j \geq 2} \alpha_{11}^{-1} \alpha_{1j} x_j + \alpha_{11}^{-1} \beta_1,$$

and f_2 is a quadratic polynomial with even cross terms and linear terms over $n - 1$ variables $\{x_2, \dots, x_n\}$.

Since the cross terms and linear terms are even,

$$f_1 = 2f'_1 = 2 \left(\sum_{j \geq 2} \frac{\alpha_{11}^{-1} \alpha_{1j}}{2} x_j + \frac{\alpha_{11}^{-1} \beta_1}{2} \right).$$

Thus,

$$f = \alpha_{11}(x_1 + f'_1)^2 + f',$$

where f' is a quadratic polynomial with even cross terms and linear terms over $n - 1$

variables $\{x_2, \dots, x_n\}$. Therefore,

$$\begin{aligned} Z_{1/2}(d, f) &= \sum_{x_1, \dots, x_n \in \mathbb{Z}_d} \xi_d^{\alpha_{11}(x_1+f_1)^2+f'} = \sum_{x_2, \dots, x_n \in \mathbb{Z}_d} \xi_d^{f'} \sum_{x_1 \in \mathbb{Z}_d} \xi_d^{\alpha_{11}(x_1+f_1)^2} \\ &= Z_{1/2}(d, f') G_{1/2}(\alpha_{11}, d), \end{aligned}$$

where the last equality comes from the fact that the summation over $x_1 \in \mathbb{Z}_d$ is independent of the value of f' . This reduces the evaluation of $Z_{1/2}(d, f)$ to $Z_{1/2}(d, f')$ where f' is a quadratic polynomial over $n-1$ variables with even cross terms and linear terms. We can repeat this step until all the diagonal terms are even, which then reduces to Case (i).

Case (iic): $d = 2^m c$ with c being odd and $c \geq 3$. Then, $\xi_d = \omega_{2d}$. Since there exists at least one diagonal term α_{ii} that is odd, then without loss of generality, the first t diagonal terms α_{ii} ($1 \leq i \leq t$) are odd and the other diagonal terms α_{ii} ($i \geq t+1$) are even.

Now, we can rewrite f as follows

$$f(x_1, \dots, x_n) = \sum_{i=1}^t x_i^2 + f_1(x_1, \dots, x_n),$$

where the coefficients of the quadratic form f_1 are all even. Hence, $f = \sum_{i=1}^t x_i^2 + 2f'_1$, with $f'_1 = f_1/2$.

Since $\gcd(2^m, c) = 1$, there exist two integers N_1 and N_2 such that $N_2 2^m + N_1 c = 1$. Adopting a process similar to that used in the proof of Proposition 90, we find, using the Chinese remainder theorem, that there exists an isomorphism $\mathbb{Z}_d \rightarrow \mathbb{Z}_{2^m} \times \mathbb{Z}_c$, $x_i \rightarrow (y_i, z_i)$ with $x_i \equiv y_i \pmod{2^m}$ and $x_i \equiv z_i \pmod{c}$. Thus, we have

$$\begin{aligned} &Z_{1/2}(d, f) \\ &= \sum_{x_1, \dots, x_n \in \mathbb{Z}_d} \xi_d^{\sum_{i=1}^t x_i^2} \omega_d^{f'_1(x_1, \dots, x_n)} \\ &= \sum_{y_1, \dots, y_n \in \mathbb{Z}_{2^m}} \sum_{z_1, \dots, z_n \in \mathbb{Z}_c} \xi_{2^m}^{\sum_{i=1}^t (N_1 + 2^m N_2) y_i^2} \xi_c^{\sum_{i=1}^t N_2 z_i^2} \omega_{2^m}^{N_1 f'_1(y_1, \dots, y_n)} \omega_c^{N_2 f'_1(z_1, \dots, z_n)} \end{aligned}$$

$$\begin{aligned}
&= \sum_{y_1, \dots, y_n \in \mathbb{Z}_{2^m}} \xi_{2^m}^{\sum_{i=1}^t (N_1 + 2^m N_2) y_i^2} \omega_{2^m}^{N_1 f'_1(y_1, \dots, y_n)} \sum_{z_1, \dots, z_n \in \mathbb{Z}_c} \xi_c^{\sum_{i=1}^t N_2 z_i^2} \omega_c^{N_2 f'_1(z_1, \dots, z_n)} \\
&= \sum_{y_1, \dots, y_n \in \mathbb{Z}_{2^m}} \xi_{2^m}^{\sum_{i=1}^t (N_1 + 2^m N_2) y_i^2} \omega_{2^m}^{(N_1 + 2^m N_2) f'_1(y_1, \dots, y_n)} \\
&\quad \times \sum_{z_1, \dots, z_n \in \mathbb{Z}_c} \xi_c^{\sum_{i=1}^t N_2 z_i^2} \omega_c^{N_2 f'_1(z_1, \dots, z_n)} \\
&= Z_{1/2}(2^m, (N_1 + 2^m N_2) f) Z_{1/2}(c, N_2 f),
\end{aligned}$$

where the second last equality comes from the fact that $\omega_{2^m}^{2^m} = 1$. This reduces the computation of $Z_{1/2}(d, f)$ to Case (iia) and Case (iib). \square

Above, we have shown the existence of efficient algorithms to evaluate half Gauss sums with quadratic polynomials that satisfy the periodicity condition. We note, however, that if we omit either the periodicity or quadratic conditions, then these sums become hard to compute (assuming some complexity conjecture). We will return to a discussion of this in Chapter 7.4.

Finally, we note here that there is a nice relationship between half Gauss sums $Z_{1/2}(d, f)$ and the number of zeros of functions of the form $f(x) - k \pmod{d}$ or $\pmod{2d}$. We explore this further in Chapter 7.6.4.

7.3 m -qudit Clifford circuits

In this section, we apply our results on the half Gauss sum to Clifford circuits. Let $d \geq 2$ be an integer. The m -qudit Clifford group is the set of operations (called *Clifford operations*) on m qudits that are generated by the following gates: X, Y, Z, F, G, CZ [98, 135–137].

Here, X, Y and Z are the d -level Pauli matrices defined by

$$X|k\rangle = |k+1\rangle, \quad Y|k\rangle = \xi_d^{1-2k}|k-1\rangle, \quad Z|k\rangle = \omega_d^k|k\rangle, \quad (7.17)$$

F is the Fourier gate defined by

$$F |k\rangle = \frac{1}{\sqrt{d}} \sum_{l=0}^{d-1} \omega_d^{kl} |l\rangle, \quad (7.18)$$

G is the Gaussian gate defined by

$$G |k\rangle = \xi_d^{k^2} |k\rangle, \quad (7.19)$$

and CZ is the controlled- Z gate defined by

$$CZ |k_1, k_2\rangle = \omega_d^{k_1 k_2} |k_1, k_2\rangle. \quad (7.20)$$

It is straightforward to check that the gates (7.17)–(7.20) satisfy the following algebraic relations [135, 136]:

$$\begin{aligned} X^d &= Y^d = Z^d = F^4 = G^{2d} = (FG)^3 q_d^{-1} = I, \\ XYX^{-1}Y^{-1} &= YZY^{-1}Z^{-1} = ZXZ^{-1}X^{-1} = \omega_d, \\ XYZ &= \xi_d, \quad FXF^{-1} = Z, \quad GXG^{-1} = Y^{-1}, \end{aligned}$$

where

$$q_d = \frac{1}{\sqrt{d}} \sum_{j=0}^{d-1} \xi_d^{j^2}.$$

From the above identities, it is easy to see that the X and Y gates can be expressed in terms of the other gates, and so the following gate set suffices to generate the Clifford group: $\mathcal{C} = \{Z, G, F, CZ\}$. An m -qudit Clifford circuit is a circuit with m registers and whose gates are all Clifford operations. We shall assume that the Clifford circuit is unitary, i.e. there are no intermediate measurements in the circuit.

Without loss of generality, we will assume that (i) each register of the Clifford circuit C begins with an F gate and ends with an F^\dagger gate, and that (ii) the internal circuit (i.e. the full circuit minus the first and last layers) consists of only gates in \mathcal{C} .

In other words, C is of the form

$$C = (F^\dagger)^{\otimes m} C' F^{\otimes m}, \quad (7.21)$$

where the internal circuit C' comprises only gates in \mathcal{C} . This loses no generality because any Clifford circuit can be transformed into a circuit of the above form, first, by inserting 4 F gates at the start of each register and the pair $F^\dagger F$ at the end of each register, and second, by compiling the internal circuit using only gates in \mathcal{C} .

For each m -qudit Clifford circuit, we adopt the following labeling scheme: divide each horizontal wire of the internal part of C into segments, with each segment corresponding to a portion of the wire which is either between 2 F gates, or between an F gate and an F^\dagger gate. It is easy to verify that the total number of segments is given by $n = h - m$, where h is the total number of F or F^\dagger gates (including those in the first and last layers) in C . Label the segments x_1, \dots, x_n .

We will also use the following terminology. The leftmost labels on each register are called *inceptive indices*. The rightmost labels on each register are called *terminal indices*. All other indices are called *internal indices*. For a set of indices $I = \{i_1, \dots, i_s\}$, we use x_I to denote the tuple $(x_{i_1}, \dots, x_{i_s})$.

Definition 95. Let C be a Clifford circuit with labels $\{x_1, \dots, x_n\}$. The *phase polynomial* of C is the polynomial

$$S_C(x_1, \dots, x_n) = 2 \sum_{\gamma \in \Gamma} \prod_{i \in I_\gamma} x_i + \sum_{g \in \mathcal{G}} \prod_{j \in I_g} x_j^2, \quad (7.22)$$

where Γ is the set of internal F, Z, CZ gates, and \mathcal{G} is the set of G gates in C .

We now show that if C is a Clifford circuit, then its phase polynomial S_C is a quadratic polynomial that satisfies the periodicity condition.

Proposition 96. If C is a Clifford circuit, then $S_C \in \mathcal{F}_2^{\text{p.c.}}$.

Proof. Since each gate in C is incident on at most 2 segments, the degree of the polynomial is at most 2. The only terms which can have odd coefficients are terms of

the form x_i^2 . Hence, the remaining terms, which are all either linear and cross terms, have even coefficients, which implies that $S_C \in \mathcal{F}_2^{\text{p.c.}}$. \square

The reverse direction is also true: for every polynomial $S \in \mathcal{F}_2^{\text{p.c.}}$, there exists a Clifford circuit C such that $S = S_C$, as the following proposition shows:

Proposition 97. Let \mathcal{A} be the class of Clifford circuits. The function

$$\Theta : \mathcal{A} \rightarrow \mathcal{F}_2^{\text{p.c.}} \quad (7.23)$$

$$C \mapsto S_C \quad (7.24)$$

is surjective.

Proof. Let

$$\mathcal{S} = \sum_{i \leq j \in [n]} \alpha_{ij} x_i x_j + \sum_{i \in [n]} \beta_i x_i \in \mathcal{F}_2^{\text{p.c.}},$$

i.e. α_{ij} is even for $i < j$ and β_i is even for all i . Construct the circuit $C = (F^\dagger)^{\otimes n} C' F^{\otimes n}$, where C' is defined as follows:

1. for each $i \in [n]$, apply the gate G α_{ii} times.
2. for each $i < j \in [n]$, apply the gate CZ $\alpha_{ij}/2$ times.
3. for each $i \in [n]$, apply the gate Z $\beta_i/2$ times.

Then,

$$S_C = \sum_{i \in [n]} \alpha_{ii} x_i^2 + 2 \left(\sum_{i < j \in [n]} \frac{\alpha_{ij}}{2} x_{ij} + \sum_{i \in [n]} \frac{\beta_i}{2} x_i \right) = \mathcal{S},$$

which implies that Θ is surjective. \square

We now show that the amplitudes of Clifford circuits can be expressed in terms of half Gauss sums.

Theorem 98. Let $C = (F^\dagger)^{\otimes m} C' F^{\otimes m}$ be an m -qubit Clifford circuit with h F or F^\dagger gates and $n = h - m$ labels x_1, \dots, x_n . Then,

$$\langle 0 |^{\otimes m} C | 0 \rangle^{\otimes m} = \frac{1}{\sqrt{d^h}} \sum_{x_1, \dots, x_n \in \mathbb{Z}_d} \xi_d^{S_C(x_1, \dots, x_n)} = \frac{1}{\sqrt{d^h}} Z_{1/2}(d, S_C). \quad (7.25)$$

Proof. Apply the sum-over-paths technique [77, 150] to the Clifford circuit C . \square

Theorem 98 can be easily generalized to also allow us to compute amplitudes with arbitrary input or output computational basis states:

Proposition 99. Let $C = (F^\dagger)^{\otimes m} C' F^{\otimes m}$ be an m -qudit Clifford circuit with h F or F^\dagger gates and $n = h - m$ labels x_1, \dots, x_n . Let $a, b \in \mathbb{Z}_d^m$. Then,

$$\langle b | C | a \rangle = \frac{1}{\sqrt{d^h}} Z_{1/2}(d, S_C + 2a \cdot x_I + 2b \cdot x_F), \quad (7.26)$$

where I and J are the inception and terminal indices (written in order) of C respectively.

Proof. We start by writing

$$\begin{aligned} \langle b | (F^\dagger)^{\otimes m} C' F^{\otimes m} | a \rangle &= \langle 0^m | (X^\dagger)^b (F^\dagger)^{\otimes m} C' F^{\otimes m} X^a | 0^m \rangle \\ &= \langle 0^m | (F^\dagger)^{\otimes m} (Z^\dagger)^b C' Z^a F^{\otimes m} | 0^m \rangle. \end{aligned}$$

Note that $C^* = (F^\dagger)^{\otimes m} (Z^\dagger)^b C' Z^a F^{\otimes m}$ is itself a Clifford circuit, and we could apply Theorem 98 to it:

$$\langle b | C | a \rangle = \frac{1}{\sqrt{d^h}} Z_{1/2}(d, S_{C^*}),$$

where

$$S_{C^*}(x_1, \dots, x_n) = S_C(x_1, \dots, x_n) + 2a \cdot x_I + 2b \cdot x_F.$$

\square

A corollary of the above result is that we can express the probabilities of outcomes of qudit Clifford circuits in terms of half Gauss sums even when only a subset of registers is measured. This was previously shown to hold for quopit Clifford circuits [167], i.e., qudit Clifford circuits, where d is an odd prime.

Theorem 100. Let $C = (F^\dagger)^{\otimes m} C' F^{\otimes m}$ be an m -qudit Clifford circuit with h F or F^\dagger gates and $n = h - m$ labels x_1, \dots, x_n . Assume that C' contains at least one F gate on each register. Let I be the inceptive indices, J be the internal indices, F be the first k terminal indices, and E be the last $m - k$ terminal indices. Let $a \in \mathbb{Z}_d^m$ and $b \in \mathbb{Z}_d^k$. Then the probability

$$P(b|a) = \|\langle b|_{1..k} C |a\rangle_{a..m}\|^2 \quad (7.27)$$

of obtaining the outcome b when the first k qudits of $C |a\rangle$ are measured is given by

$$P(b|a) = \frac{1}{d^{n+k}} Z_{1/2}(d, \phi), \quad (7.28)$$

where

$$\begin{aligned} \phi(x_I, y_I, x_F, y_F, x_J, y_J, w_E) &= S_c(x_I, x_J, x_F, w_E) - S_c(y_I, y_J, y_F, w_E) \\ &\quad + 2a \cdot (x_I - y_I) + 2b \cdot (x_F - y_F). \end{aligned} \quad (7.29)$$

Proof.

$$\begin{aligned} P(b|a) &= \|\langle b|_{1..k} U |a\rangle_{a..m}\|^2 \\ &= \sum_{\beta \in \mathbb{Z}_d^{m-k}} |\langle b\beta | C |a\rangle|^2 \\ &= \sum_{\beta \in \mathbb{Z}_d^{m-k}} \left| \frac{1}{\sqrt{h}} Z_{1/2}(d, S_C + 2a \cdot x_I + 2(b, \beta) \cdot (x_F, x_E)) \right|^2 \\ &= \frac{1}{d^h} \sum_{x, y \in \mathbb{Z}_d^n} \xi_d^{S_C(x) - S_C(y) + 2a \cdot (x_I - y_I) + 2b \cdot (x_F - y_F)} \sum_{\beta \in \mathbb{Z}_d^{m-k}} \omega_d^{\beta \cdot (x_E - y_E)} \\ &= \frac{1}{d^{h-m+k}} \sum_{x_I, y_I \in \mathbb{Z}_d^n} \sum_{x_F, y_F \in \mathbb{Z}_d^k} \sum_{x_J, y_J \in \mathbb{Z}_d^{n-2m}} \sum_{w_E \in \mathbb{Z}_d^{m-k}} \xi_d^{\phi(x_I, y_I, x_F, y_F, x_J, y_J, w_E)} \\ &= \frac{1}{d^{n+k}} Z_{1/2}(d, \phi). \end{aligned} \quad (7.30)$$

where in the fifth line, we used the property that

$$\sum_{\beta \in \mathbb{Z}_d^{m-k}} \omega_d^{\beta \cdot (x_E - y_E)} = d^{m-k} \delta_{x_E, y_E}. \quad (7.31)$$

□

Since half Gauss sums can be computed efficiently, the above proof gives an alternative proof of the Gottesman-Knill Theorem [111] for all qudit Clifford circuits:

Corollary 101. (Gottesman-Knill Theorem—strong version) Qudit Clifford circuits acting on computational basis input states can be efficiently simulated (in the strong sense [81]) by a classical computer.

Since strong simulation implies weak simulation [217], Corollary 101 implies that there is an efficient classical algorithm that samples from the output distributions of qudit Clifford circuits.

7.4 Hardness results and complexity dichotomy theorems

In this section, we show that extending the definition of the (polynomial) Gauss sum in various ways leads to intractable exponential sums. See Table 7.1 for a summary of our results.

7.4.1 Degree-3 polynomials

We consider circuits that are over the Clifford+ CCZ gate set, where CCZ is the controlled-controlled- Z gate defined by

$$CCZ |x_i, x_j, x_k\rangle = \omega_d^{x_i x_j x_k} |x_i, x_j, x_k\rangle. \quad (7.32)$$

For simplicity, we consider circuits of the form $(F^\dagger)^{\otimes n} D F^{\otimes n} |0\rangle^{\otimes n}$, where D is a diagonal circuit consisting of gates $\{Z, G, CZ, CCZ\}$. By the sum-over-paths tech-

nique, it is straightforward to show that the amplitudes of such circuits are of the form

$$\langle 0|^{\otimes n} (F^\dagger)^{\otimes n} D F^{\otimes n} |0\rangle^{\otimes n} = \frac{1}{d^n} \sum_{x_1, \dots, x_n \in \mathbb{Z}_d} \xi_d^{f(x_1, \dots, x_n)}, \quad (7.33)$$

where f is degree-3 polynomial.

We first consider the case when d is odd. Applying each gate in D j times, where $j \in \{0, \dots, d-1\}$, we obtain

$$\langle 0|^{\otimes n} (F^\dagger)^{\otimes n} D^j F^{\otimes n} |0\rangle^{\otimes n} = \frac{1}{d^n} \sum_{x_1, \dots, x_n \in \mathbb{Z}_d} \xi_d^{jf(x_1, \dots, x_n)}.$$

By Eq. (7.60), we have

$$\#\{f \equiv k \pmod{d}\} = d^{n-1} \sum_{j=0}^{d-1} \xi_d^{-kj} \langle 0|^{\otimes n} (F^\dagger)^{\otimes n} D^j F^{\otimes n} |0\rangle^{\otimes n}. \quad (7.34)$$

Thus, we have reduced the problem of counting the number of zeros of degree-3 polynomials to the problem of computing the amplitudes of quantum circuits $\langle 0|^{\otimes n} (F^\dagger)^{\otimes n} D^j F^{\otimes n} |0\rangle^{\otimes n}$ (this follows from the fact that the Fourier transformation can be carried out in $O(d^2)$ -time, which is independent of n). Therefore, if the output of quantum circuits with the form $\langle 0|^{\otimes n} (F^\dagger)^{\otimes n} D F^{\otimes n} |0\rangle^{\otimes n}$ can be computed in $\text{poly}(n)$ time, then the number of zeros for degree-3 polynomial can also be evaluated in polynomial time. Similar arguments also hold when d is even (to see this, note that we can repeat the gates in D $1, \dots, 2d-1$ times and use Eq. (7.59)).

7.4.2 Without the periodicity condition

In this section, we show, under plausible complexity assumptions, that in order for the exponential sum $Z_{1/2}(d, f)$ to be tractable, we need the periodicity condition to

hold. To see this, we consider the case $d = 2$:

$$Z_{1/2}(2, f) = \sum_{x_1, \dots, x_n \in \mathbb{Z}_2} i^{f(x_1, \dots, x_n)},$$

where

$$f(x_1, \dots, x_n) = \sum_{i \leq j \in [n]} \alpha_{ij} x_i x_j + \gamma_0 \quad (7.35)$$

is a quadratic polynomial with integer (not necessarily even) coefficients α_{ij} . Note that $x_i^2 = x_i$ for all $x_i \in \mathbb{Z}_2$, and so there's no need for an additional linear term $\sum_i \beta_i x_i$ in f .

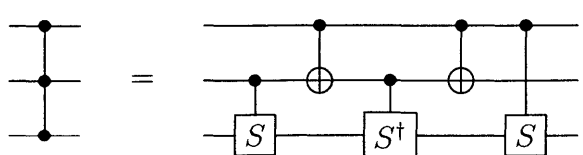
Now, consider the strictly universal⁵ gate set $\mathcal{G} = \{H, Z, CS\}$, where $CS = \text{diag}(1, 1, 1, i)$ is the controlled-phase gate, which satisfies $CS |x_i, x_j\rangle = i^{x_i x_j} |x_i, x_j\rangle$. By the sum-over-paths technique [77], if $U_{\mathcal{G}}$ is the unitary implemented by a circuit over the gate set \mathcal{G} , then

$$\langle 0 | U_{\mathcal{G}} | 0 \rangle \propto Z_{1/2}(2, f), \quad (7.36)$$

where f is of the form Eq. (7.35).

We now show that Eq. (7.36) is hard to compute. Let g be a degree-3 polynomial over \mathbb{Z}_2 . Then by the circuit-polynomial correspondence [172], there exists a unitary U implemented by a circuit C over the gate set $\{H, Z, CZ, CCZ\}$ that satisfies $\langle 0 | U | 0 \rangle \propto \text{gap}(g)$.

Now, construct a circuit $C_{\mathcal{G}}$ that is equivalent to C , but which consists of only gates in \mathcal{G} . To achieve this, we replace all the CZ and CCZ gates in C by circuit gadgets comprising only H and CS gates. This may be achieved by making use of the following circuit identity (which follows from Lemma 6.1 of [22]):



$$\text{CZ} = S \text{---} CZ \text{---} S^\dagger \text{---} CZ \text{---} S \quad (7.37)$$

⁵Note that Z is not needed for universality, since $\{H, CS\}$ is already universal (see [145] or Theorem 1 of [12])

as well as the following identities:

$$\begin{aligned} CZ &= (CS)^2, \\ C(S^\dagger) &= (CS)^3, \\ CX_{12} &= H_2 CZ_{12} H_2, \end{aligned}$$

which allow us to express CCZ and CZ completely in terms of H and CS .

If we denote the unitary implemented by C_G by U_G , then

$$\text{gap}(g) \propto \langle 0|U|0\rangle = \langle 0|U_G|0\rangle \propto Z_{1/2}(2, f), \quad (7.38)$$

with proportionality constants that can be computed efficiently.

But since g is a degree-3 polynomial, $\text{gap}(g)$ is $\#\text{P}$ -hard to compute (see Theorem 1 of [90]). Hence, it follows that $Z_{1/2}(2, f)$ is also $\#\text{P}$ -hard to compute.

7.4.3 Other incomplete Gauss sums

In this section, we restrict our attention to $d = 2$, and consider incomplete Gauss sums of the form:

$$Z_{1/2^k}(2, f) = \sum_{x_1, \dots, x_n \in \mathbb{Z}_2} \omega_{2^{k+1}}^{f(x_1, \dots, x_n)} \quad (7.39)$$

with $k \geq 2$. For $k = 2$, the exponential sum

$$Z_{1/4} = \sum_{x_1, \dots, x_n \in \mathbb{Z}_2} \omega_8^{f(x_1, \dots, x_n)},$$

where f is unrestricted, corresponds to the universal gate set $\{H, T, CZ\}$. Hence, computing such sums is $\#\text{P}$ -hard. However, for quadratic polynomials f satisfying the periodicity condition, we can reduce the evaluation of $Z_{1/4}(2, f)$ to the evaluation of $Z_{1/2}(2, f')$, for some quadratic polynomial f' satisfying the periodicity condition. This implies that $Z_{1/4}(2, f)$ can be evaluated in $\text{poly}(n)$ time. More generally, for any $k \geq 2$, if f is a quadratic polynomial satisfying the periodicity condition, the

incomplete Gauss sum $Z_{1/2^k}(2, f)$ can be reduced to $Z_{1/2}(2, f')$.

Lemma 102. Let $d = 2$, and let $f = \sum_{i \leq j} \alpha_{ij} x_i x_j + \sum_i \beta_i x_i$ be a quadratic polynomial. Then f satisfies the periodicity condition

$$\omega_{2^{k+1}}^{f(x_1, \dots, x_n)} = \omega_{2^{k+1}}^{f((x_1 \bmod 2), \dots, (x_n \bmod 2))}, \quad (7.40)$$

if and only if $2^{k-1} | \alpha_{ii}$, $2^k | \alpha_{ij}$ ($i \neq j$) and $2^k | \beta_i$. Thus, $Z_{1/2}^{k+1}(2, f) = Z_{1/2}(2, f/2^{k-1})$, where $f/2^{k-1}$ satisfies the periodicity condition for $\omega_2 = \sqrt{-1}$.

Proof. It is easy to verify that the quadratic polynomial f satisfies the periodicity condition if $2 | \alpha_{ii}$, $4 | \alpha_{ij}$ ($i < j$) and $4 | \beta_i$.

For any i ,

$$\omega_{2^{k+1}}^{\alpha_{ii} x_i^2 + \beta_i x_i} = \omega_{2^{k+1}}^{\alpha_{ii} (x_i+2)^2 + \beta_i (x_i+2)}$$

for any $x_i \in \mathbb{Z}$, which implies that $2^{k-1} | \alpha_{ii}$ and $2^k | \beta_i$.

Moreover, for any fixed i and j with $i < j$, we can choose $x_k = 0$ for any $k \neq i, j$ to get

$$\omega_{2^{k+1}}^{\alpha_{ii} x_i^2 + \alpha_{ii} x_j^2 + \alpha_{ij} x_i x_j + \beta_i x_i + \beta_j x_j} = \omega_{2^{k+1}}^{\alpha_{ii} (x_i+2)^2 + \alpha_{ii} x_j^2 + \alpha_{ij} (x_i+2) x_j + \beta_i (x_i+2) + \beta_j x_j}$$

for any $x_i, x_j \in \mathbb{Z}$. This implies that $2^k | \alpha_{ij}$. Since i, j were arbitrarily chosen, it follows that all cross terms α_{ij} satisfy $2^k | \alpha_{ij}$. □

7.4.4 Complexity dichotomy theorems

In 1979, Valiant introduced the complexity class $\#\text{P}$ to characterize the computational complexity of solving counting problems [222], and ever since then, this has been a subject of much research.

Among the many important results arising from this research are the complexity dichotomy theorems, which have attracted considerable attention [54, 55, 58, 72, 87,

88, 108]. These theorems roughly state that for certain classes of counting problems, each problem in the class is either efficiently computable or $\#P$ -hard. (See [56] for an overview.)

These dichotomy theorems have applications in the study of exponential sums. An example of such a theorem was provided by [57], which proved that computing Gauss sums $Z(d, f)$ can be performed efficiently when $\deg(f) \leq 2$ and is $\#P$ -hard when $\deg(f) \geq 3$. Note that the polynomials considered by [57] all satisfy the periodicity condition. Hence, if we combine the $\#P$ -hardness result with Theorem 94, we arrive at a new dichotomy theorem: if $\deg(f) \leq 2$, then the exponential sum $Z_{1/2}(d, f)$ is computable in polynomial time. Otherwise, if $\deg(f) \geq 3$, then computing $Z_{1/2}(d, f)$ is $\#P$ -hard.

Furthermore, for the class of aperiodic exponential sums, our results imply another new complexity dichotomy theorem: if $\deg(f) \leq 1$, then the exponential sum $Z_{1/2}$ is computable in polynomial time; otherwise, if $\deg(f) \geq 2$, then computing $Z_{1/2}(d, f)$ is $\#P$ -hard. For a summary of these results, see Table 7.1.

7.5 Tractable signatures in Holant problems

In this section, we will apply our results about half Gauss sums to an important framework called the Holant framework, which we will now describe. Let \mathcal{F} be a set of functions, where each element $f \in \mathcal{F} : \mathbb{Z}_d^n \rightarrow \mathbb{C}$. A signature grid $\Omega = (G, \mathcal{F})$ is a tuple, where $G = (V, E)$ is a hypergraph and each $v \in V$ is assigned a function $f_v \in \mathcal{F}$ with arity equal to the number of hyperedges incident on it. A \mathbb{Z}_d assignment σ for every $e \in E$ gives an evaluation $\prod_v f_v(\sigma|_{E(v)})$, where $E(v)$ denotes the set of edges incident on v . Given an input instance Ω , we are interested in computing

$$\text{Holant}_\Omega = \sum_{\sigma: E \rightarrow \mathbb{Z}_d} \prod_v f_v(\sigma|_{E(v)}). \quad (7.41)$$

Affine signatures over \mathbb{Z}_2 and \mathbb{Z}_3 were defined in [58, 233]. In this section, we give a definition of affine signatures over \mathbb{Z}_d , for $d \geq 2$:

1. **Affine signature over \mathbb{Z}_d :** Let f be a signature of arity n with inputs x_1, \dots, x_n over the domain \mathbb{Z}_d . We say that f is *affine* if it has the following form

$$\lambda \chi_{A\vec{x}=0} \xi_d^{g(x_1, \dots, x_n)}, \quad (7.42)$$

where $\lambda \in \mathbb{C}$, ξ_d is a chosen square root of $\omega_d = \exp(2\pi i/d)$ satisfying $\xi_d^{d^2} = 1$, A is a matrix over \mathbb{Z}_d , χ is a 0–1 indicator function such that $\chi_{A\vec{x}=0} = 1$ if and only if $A\vec{x} = 0$, and $g(x_1, \dots, x_n) \in \mathbb{Z}[x_1, \dots, x_n]$ is a quadratic polynomial with even cross terms and linear terms. Let \mathcal{A} be the set of all affine signatures. It is straightforward to check that \mathcal{A} is closed under multiplication.

2. **Degenerate function on n variables:** Let

$$\mathcal{D} = \{\otimes_i [f_i(0), f_i(1), \dots, f_i(d-1)] \mid f_i(j) \in \mathbb{C}\} \quad (7.43)$$

be the set of functions that can be expressed as a tensor product of unary functions.

3. **The set \mathcal{P} :** Let \mathcal{P} be the set of functions that can be written as the composition of unary functions with the binary equality relation $=_2$, where $=_2(i, j)$ is equal to 1 for $i = j$ and is equal to 0 otherwise.

Theorem 103. Given a class of functions \mathcal{F} , if $\mathcal{F} \subseteq \mathcal{A}$ or $\mathcal{F} \subseteq \mathcal{P}$, $\text{Holant}(\mathcal{F})$ is computable in polynomial time.

Proof. (1) If $\mathcal{F} \subseteq \mathcal{P}$, then following [58], we can group the variables into connected components if these variables are connected by the binary equality relation $=_2$. In any connected component, let us start with a variable with a value in \mathbb{Z}_d , and follow any edges labeled by the binary equality relation. There is at most one extension of this assignment, i.e., each variable in this connected component must take the same value as the value that was taken at the beginning. Then we can easily compute the value of the Holant by simply multiplying all the values. There are at most d values, as we have d choices at the starting edge.

(2) If $\mathcal{F} \subseteq \mathcal{A}$, then the method in [58] may not work, as Gaussian elimination may not be applicable for general \mathbb{Z}_d . To get around this, we consider the inner product representation of the Holant problem $\text{Holant}(\mathcal{F})$, which can be written as

$$\text{Holant}(\mathcal{F}) = (\otimes_e \langle \text{GHZ}_e |) (\otimes_v |f_v\rangle), \quad (7.44)$$

where $|\text{GHZ}_e\rangle$ denotes the GHZ state on $(\mathbb{C}_d)^{\otimes |e|}$, where $|e|$ denotes the number of vertices incident on the edge e . For example, if $|e| = 1, 2, 3$, then GHZ_e is $|+\rangle = \sum_{i=0}^{d-1} |i\rangle$, $|\text{Bell}\rangle = \sum_{i=0}^{d-1} |ii\rangle$ and $|\text{GHZ}\rangle = \sum_{i=0}^{d-1} |iii\rangle$, respectively.

Since $f_v \in \mathcal{A}$,

$$|f_v\rangle = \sum_{x_1, \dots, x_k \in \mathbb{Z}_d} \chi_{A_v \vec{x}=0} \zeta_d^{g_v(x_1, \dots, x_k)} |x_1, \dots, x_k\rangle, \quad (7.45)$$

where g_v is a quadratic polynomial with even cross and linear terms, and k denotes the arity of f_v . If we omit the term $\chi_{A_v \vec{x}=0}$ in the above expression, then the remaining expression is just a stabilizer state, which we denote as $|\text{STAB}\rangle_v$. Now, consider $\sum_{i=1}^k A_{1,i} x_i + A_{1,k+1} = 0 \pmod{d}$ that is given by the first line of $A\vec{x} = 0$. We can add an ancilla qudit with $\langle 0 | \Pi_j (CX)^{A_{1j}} X^{A_{1,k+1}} | 0 \rangle$ with control qudit being $j \in 1, \dots, k$. Then, $|f_v\rangle$ can be written as

$$|f_v\rangle = \langle 0 |^{\otimes m_v} \prod_{i,j} (CX)^{A_{ij}} X^{A_{i,k+1}} |\text{STAB}\rangle_v |0\rangle^{\otimes m_v}, \quad (7.46)$$

where m_v is the number of rows in A_v . Therefore,

$$\text{Holant}(\mathcal{F}) = (\otimes_e \langle \text{GHZ}_e |) (\otimes_v \langle 0 |^{\otimes m_v}) (\otimes_v \Pi_{i,j} (CX)^{A_{ij}} X^{A_{i,k+1}} |\text{STAB}\rangle_v |0\rangle^{\otimes m_v}),$$

which is just a product of two stabilizer states. It can be computed in polynomial time by the Gottesman-Knill theorem [111]. \square

While Theorem 103 addresses the question about which functions lead to tractable Holant problems, we leave open the question about which functions lead to intractable Holant problems. More specifically, can we prove that for any class of functions \mathcal{F}

not in \mathcal{P} or \mathcal{A} , the problem $\text{Holant}(\mathcal{F})$ is $\#\text{P}$ -hard?

7.6 Appendix for Chapter 7

7.6.1 Exponential sum terminology

In this section, we summarize some of the terminology used in this chapter. An *exponential sum* is a sum of the form

$$\sum_{x \in A} e^{f(x)}, \quad (7.47)$$

where $A \subseteq V$ is a finite set, V is an arbitrary set, and $f : V \rightarrow \mathbb{C}$ is a complex-valued function.

The exponential sums used in this chapter are all *incomplete Gauss sums*⁶, which are sums of the form

$$Z_I(d, b, f) = \sum_{x_1, \dots, x_n \in \mathbb{Z}_d} \omega_b^{f(x_1, \dots, x_n)} \quad (7.48)$$

where $d, n, b \in \mathbb{Z}^+$ satisfy $d \leq b$ and f is a polynomial with integer coefficients.

Two special cases of incomplete Gauss sums are the *Gauss sum*, defined as

$$Z(d, f) = Z_I(d, d, f) = \sum_{x_1, \dots, x_n \in \mathbb{Z}_d} \omega_d^{f(x_1, \dots, x_n)}. \quad (7.49)$$

and the *half Gauss sum*, defined as

$$Z_{1/2}(d, f) = \sum_{x_1, \dots, x_n \in \mathbb{Z}_d} \xi_d^{f(x_1, \dots, x_n)}. \quad (7.50)$$

With this terminology, note that Gauss sums are a special case of half Gauss sums, which are in turn a special case of incomplete Gauss sums.

When f is quadratic, $Z(d, f)$ and $Z_{1/2}(d, f)$ reduce to the (multivariate) quadratic Gauss sum (7.13) and (multivariate) quadratic half-Gauss sum (7.14) respectively.

⁶Here, we generalized the definition of “incomplete Gauss sums” used in [93, 159] to the multivariate case.

When $n = 1$ and f is a homogeneous quadratic polynomial (i.e. $f(x) = ax^2$), the sums $Z(d, f)$ and $Z_{1/2}(d, f)$ reduce to the univariate quadratic homogeneous Gauss sum (7.6) (which is usually just referred to as a Gauss sum [156]) and univariate quadratic homogeneous half-Gauss sum (7.7) respectively. Note that univariate quadratic Gauss sums are also called Weil sums [161] (see also Chapter 6).

7.6.2 Properties of Gauss sum

In this section, we state some basic properties of the Gauss sum $G(\cdot, \cdot)$ [156]. Given two nonzero integers a, d with $d > 0$ and $\gcd(a, d) = 1$,

$$G(a, d) = \sum_{x \in \mathbb{Z}_d} \omega_d^{ax^2}.$$

The Gauss sum satisfies the following properties:

(1) If d is odd, then

$$G(a, d) = \left(\frac{a}{d}\right) G(1, d), \quad (7.51)$$

where $\left(\frac{a}{d}\right)$ is the Jacobi symbol. Moreover,

$$G(1, d) = \begin{cases} \sqrt{d}, & d \equiv 1 \pmod{4} \\ i\sqrt{d}, & d \equiv 3 \pmod{4}. \end{cases} \quad (7.52)$$

(2) If $d = 2^k$, then for $k \geq 4$,

$$G(a, 2^k) = 2G(a, 2^{k-1}). \quad (7.53)$$

(3) If $d = bc$ with $\gcd(b, c) = 1$, then

$$G(a, bc) = G(ab, c)G(ac, b). \quad (7.54)$$

7.6.3 Half Gauss sum for $\xi_d = -\omega_{2d}$ with even d

In Chapter 7.2.1, we took $\xi_d = \omega_{2d}$ when d is even. Note that in this case, ξ_d could also have been chosen to be $\pm\omega_{2d}$. Here, we consider the case $\xi_d = \omega_{2d}$ for even numbers d . To distinguish these two cases, we define $G_{1/2}(a, d)_+$ for $\xi_d = \omega_{2d}$ and $G_{1/2}(a, d)_-$ for $\xi_d = -\omega_{2d}$ for even d . First, we discuss the properties of $G_{1/2}(a, d)_-$.

Lemma 104. If d is even, then

$$G_{1/2}(a, d)_- = G_{1/2}(a(N_1 + bN_2), b)_- G_{1/2}(aN_2, c), \quad (7.55)$$

where $d = bc$, $\gcd(b, c) = 1$, $2|b$ and integers N_1 and N_2 satisfy $N_1c + N_2b = 1$.

Proof. Following the approach in the proof of Proposition 90, we get

$$\begin{aligned} \xi_d^{ax^2} &= (-1)^{ax^2} \omega_{2b}^{aN_1x^2} \omega_{2c}^{aN_2x^2} = (-1)^{ay^2} \omega_{2b}^{aN_1y^2} \xi_c^{aN_2z^2} (-1)^{aN_2y^2} \\ &= (-\omega_{2b})^{a(N_1+bN_2)y^2} \xi_c^{aN_2z^2} \\ &= \xi_b^{a(N_1+bN_2)y^2} \xi_c^{aN_2z^2}. \end{aligned}$$

□

Lemma 105. If $m \geq 3$, then

$$G_{1/2}(a, 2^m)_- = 2G_{1/2}(a, 2^{m-2})_+. \quad (7.56)$$

Proof. For $m \geq 3$,

$$\begin{aligned} G_{1/2}(a, 2^m)_- &= \sum_{x \in [2^m]} (-\omega_{2^{m+1}})^{ax^2} \\ &= \sum_{x \in [2^{m-1}]} \left[(-\omega_{2^{m+1}})^{ax^2} + (-\omega_{2^{m+1}})^{a(x+2^{m-1})^2} \right] \\ &= \sum_{x \in [2^{m-1}]} (-\omega_{2^{m+1}})^{ax^2} \left[1 + (-\omega_{2^{m+1}})^{a2^m x + a2^{2m-2}} \right] \\ &= \sum_{x \in [2^{m-1}]} (-1)^{ax^2} \omega_{2^{m+1}}^{ax^2} [1 + (-1)^x] \end{aligned}$$

$$\begin{aligned}
&= \sum_{y \in [2^{m-2}]} \omega_{2^{m+1}}^{a(2y)^2} [1 + (-1)^{2y}] \\
&= 2 \sum_{y \in [2^{m-2}]} \omega_{2^{m+1}}^{4ay^2} = 2 \sum_{y \in [2^{m-2}]} \omega_{2^{m-1}}^{ay^2} \\
&= 2G_{1/2}(a, 2^{m-2})_+.
\end{aligned}$$

□

7.6.4 Relationship between half Gauss sums and zeros of a polynomial

In this section, we show that there is a nice relationship between half Gauss sums $Z_{1/2}(d, f)$ and the number of zeros of functions of the form $f(x) - k \pmod{d}$ or $\pmod{2d}$. If d is even, then $\xi_d = \omega_{2d}$ and $\xi_d^{2d} = 1$, which means that the exponential sum $Z_{1/2}(d, f)$ can be rewritten as

$$Z_{1/2}(d, f) = \sum_{j=0}^{2d-1} \xi_d^j \# \{f \equiv j \pmod{2d}\}, \quad (7.57)$$

where $\# \{f \equiv j \pmod{2d}\}$ denotes the number of solutions $(x_1, \dots, x_n) \in \mathbb{Z}_d^n$ such that $f(x_1, \dots, x_n) \equiv j \pmod{2d}$. Thus,

$$Z_{1/2}(d, kf) = \sum_{j=0}^{2d-1} \xi_d^{kj} \# \{f \equiv j \pmod{2d}\}. \quad (7.58)$$

By taking the inverse Fourier transformation, we obtain

$$\# \{f \equiv j \pmod{2d}\} = \frac{1}{2d} \sum_{k=0}^{2d-1} \xi_d^{-kj} Z_{1/2}(d, kf). \quad (7.59)$$

Similarly, if d is odd, then $\xi_d = \omega_d^{\frac{d+1}{2}}$ and

$$\# \{f \equiv j \pmod{d}\} = \frac{1}{d} \sum_{k=0}^{d-1} \xi_d^{-kj} Z_{1/2}(d, kf). \quad (7.60)$$

Thus, the problem of evaluating $Z_{1/2}(d, f)$ is equivalent to the problem of counting the number of solutions of the equations $f \equiv k \pmod{2d}$ (or $f \equiv k \pmod{d}$), up to an inverse Fourier transformation.

Chapter 8

Quantum simulation from the bottom up: the case of rebits

Typically, quantum mechanics is thought of as a linear theory with unitary evolution governed by the Schrödinger equation. While this is technically true and useful for a physicist, with regards to *computation* it is an unfortunately narrow point of view. Just as a classical computer can simulate highly nonlinear functions of classical states, so too can the more general quantum computer simulate nonlinear evolutions of quantum states. In this chapter, we detail one particular simulation of nonlinearity on a quantum computer, showing how the entire class of \mathbb{R} -unitary evolutions (on n qubits) can be simulated using a unitary, real-amplitude quantum computer (consisting of $n + 1$ qubits in total). These operators can be represented as the sum of a linear and antilinear operator, and add an intriguing new set of nonlinear quantum gates to the toolbox of the quantum algorithm designer.

An important application of our results, which ties to the theme of this thesis, is in studying which nonlinear operations on quantum circuits can be efficiently classically simulated. We define a class of operators, called the \mathbb{R} -Clifford operators, and show that circuits composed of these operators can be efficiently classically simulated.

This perspective of using the physical operators that we have to simulate non-physical ones that we do not is what we call bottom-up simulation, and we give some examples of its broader implications. This chapter is based on joint work with

Murphy Yuezhen Niu and Theodore J. Yoder [149].

8.1 Introduction

Simulation is a ubiquitous task in the modern world with diverse uses from fundamental research (e.g. particle physics simulations in the LHC) to entertainment (e.g. virtual reality headsets). Computers are often associated with simulation due to their wide ranging capabilities as (finite instances of) universal Turing machines. Like classical computers, universal quantum computers are expected to be powerful simulators offering potentially even greater efficiency for some very important quantum tasks, such as chemistry [17, 18, 157] and fermionic simulations [41, 46]. Moreover, the notion of simulation is not limited to a correspondence between especially disparate systems – for instance, quantum error-correcting codes can be said to simulate a handful of encoded qubits with many physical ones.

Yet, it is quite common for a simulator P to be developed in terms of end-goals, that is, for the purpose of modeling specific operators O_L on the simulated system L that are deemed interesting. Such a design, which we call top-down, reveals the set of operators O_P on P that are necessary to simulate O_L . However, with access to a universal simulator, like a quantum computer, it may be more relevant to start from the bottom with operators we can definitely perform on P , and ask what operators on L can be simulated. In contrast to the top-down simulation, this bottom-up simulation by definition takes full advantage of the capability of the simulator.

In this chapter, we provide a fleshed-out example of a bottom-up simulation, a nonlinear n -qubit quantum computer being simulated by a linear, real-amplitude quantum computer. The simulator consists of $n + 1$ *rebits*, which mathematically means that its states are normalized vectors restricted to $\mathbb{R}^{2^{n+1}}$ and it has the ability to perform orthogonal linear operators. Although the top-down version of this simulation has been noted many times in the past beginning with Bennett et. al. [25], the bottom-up viewpoint, while less often considered (with just a brief mention in McKague, Mosca, and Gisin [170] and some special-case use in [64, 83]), reveals exciting new

phenomena. In particular, the $(n + 1)$ -rebit computer is able to efficiently simulate, not just unitary, but also non-unitary (indeed nonlinear) operators on the n -qubit computer. Thus, a major takeaway is that nonlinearity can be simulated by linear systems on a larger space¹.

We completely characterize the operators that can be simulated using this bottom-up approach to rebit simulation. It happens that the simulable operators are a subgroup of the so-called \mathbb{R} -linear operators, which we call \mathbb{R} -unitary. We give universal gate sets for the \mathbb{R} -unitaries, which we note can be constructed from just *partial antiunitary* operators, i.e. those that act unitarily on some subspace of the n -qubit Hilbert space and antiunitarily on the rest. Furthermore, the (orthogonal) Clifford hierarchy [113] on rebits maps to a Clifford hierarchy, dubbed the \mathbb{R} -Clifford hierarchy, contained within the \mathbb{R} -unitaries. In the spirit of the Gottesman-Knill theorem [111], we show that the second level of the \mathbb{R} -Clifford hierarchy, which is strictly larger than the Clifford group, is classically efficiently simulable. We also explore the efficiency of our rebit simulation for general \mathbb{R} -unitary circuits.

A good reason to consider bottom-up simulation of quantum computers is for algorithm design. Our results show that when designing a quantum algorithm using the circuit model, the designer has at their disposal not just unitary operators, but also the set of \mathbb{R} -unitary operators. Examples of this utility are the quantum simulation of the Majorana equation [64] and measurement of entanglement monotones [83]. It is important to note, however, that while the rebit simulator can model nonlinear operators, this simulation does not allow us to exceed the power of quantum computers. After all, a rebit simulator is just a special case of a quantum circuit. This conclusion that non-linear operators can be simulated by quantum computers does not contradict the results of [10], since the \mathbb{R} -linear operators are not among the non-linear operators described in [10] that imply polynomial-time solutions for NP-complete and $\#P$ -complete problems. We expect of course that generalizations of our result exist, and more exotic operators (though still not those of the type found

¹At this point, an interested reader may refer to Chapter 8.8.1, where we present a simple example of such a simulation.

in [10]) will become simulable when more rebits (as a function of n) are included in the simulator.

8.1.1 Two kinds of simulation

Here we offer a definition of simulation that suits our needs, but also applies to most other uses of the term. A list of examples is provided in Table 8.1.

A simulation is a tuple $(L, P, \mathcal{P}, O_{(\cdot)})$ where L and P are sets of states of the logical (the simulated) and the physical (the simulator) state spaces. The map $\mathcal{P} : L \rightarrow P$ serves to relate the two. Because the simulator should be faithful to the simulated space, we require that \mathcal{P} is injective – i.e. $\mathcal{P}(a) = \mathcal{P}(b)$ implies $a = b$. But we do not require it to be surjective – in general, $\text{Range}(\mathcal{P}) \subseteq P$ and there is a partial inverse $\mathcal{L} : \text{Range}(\mathcal{P}) \rightarrow L$.

The final element $O_{(\cdot)}$ of the tuple specifies either the operators we want to simulate – i.e. $O_{(\cdot)} = O_L : L \rightarrow L$ corresponding to a top-down simulation – or the operators the simulator can support – i.e. $O_{(\cdot)} = O_P : \text{Range}(\mathcal{P}) \rightarrow \text{Range}(\mathcal{P})$ corresponding to a bottom-up simulation. Once one set of operators (either O_L or O_P) is specified, the other can be determined by using the established maps \mathcal{P} and \mathcal{L} .

As an example, the Gottesman-Knill theorem [111] provides a method for classical computers to simulate a limited, non-universal, n -qubit quantum computer. To be exact, the quantum computer is only allowed to perform Clifford gates that act on at most two qubits at a time (e.g. the gate set $\{H, S, CX\}$ is sufficient) and single-qubit Pauli measurements, which make up the set of operators O_L . Furthermore, L is the set of stabilizer states (which is closed under the aforementioned operators O_L) and P is the set of $n \times (2n + 1)$ binary matrices. A stabilizer state $|\psi\rangle$ of n -qubits is special because there are n independent and mutually commuting Pauli operators p_i , $i = 1, 2, \dots, n$ so that $p_i |\psi\rangle = |\psi\rangle$. Since each such n -qubit Pauli can be specified using $2n + 1$ bits (the last bit is to track a \pm sign), these bit strings form the rows of a full rank matrix in P . This describes the map \mathcal{P} .

Gottesman-Knill as described is a top-down simulation – the goal is to simulate the Clifford gates O_L . Indeed, the simulator can achieve this efficiently, since each

Task	Simulated space L	Simulator space P	Mapping \mathcal{P}	operators O_L
Hamiltonian Sim. [74]	n qubits	2D qubit lattice	CMP map [74]	All Hamiltonians
Fermionic Sim. [46]	M Fermi modes	Q qubits	Bravyi-Kitaev [46]	2-body Hamiltonian
Gottesman-Knill [111]	Stabilizer states	Binary matrices	Heisenberg repr.	Stabilizer operators
Quantum codes [109]	k qubits	n qubits, $n > k$	Encoder	Universal gate set
Compiling [78]	n qubits	n qubits + m ancillas	$ \psi\rangle \mapsto \psi\rangle \otimes 0\rangle^{\otimes m}$	All unitaries ³
Rebits [170]	n qubits	$n + 1$ rebits	Eq. (8.1)	\mathbb{R} -unitaries

Table 8.1: Several examples of our definition of simulation.

operator in O_P (i.e. a classical manipulation of the binary matrices in P) that corresponds to an operator in O_L takes constant time. Yet, important insights come from the bottom-up viewpoint. For instance, the classical simulation can exactly (and efficiently) calculate the entire probability distribution that the quantum computer can only sample from! The classical simulation is actually *more* powerful than the Clifford quantum computer it was designed to simulate². As this example demonstrates, it is in this bottom-up manner that unexpected phenomena can occur in simulations. For another example, see Chapter 8.8.2 where we show that Gottesman-Knill can also simulate complex conjugation of stabilizer states.

²There are actually several variants of efficiently-simulable Clifford circuits – see [143] and [148]. To be concrete, pick the variant in which the Clifford circuit is nonadaptive, the input is in the computational basis, and the objective is so-called *strong simulation*, i.e. to calculate the probability a bit string y is observed when measuring any subset of output qubits. Theorem 4 in [143] shows as a corollary of Gottesman-Knill that this particular strong simulation task is efficient on a classical computer. Furthermore, as shown in Proposition 1 of [217], the ability to strongly simulate a class of circuits implies the ability to classically efficiently sample from it. So, indeed the classical simulation is strictly more powerful than the quantum computation in this case.

³Compiling looks like a rather trivial simulation until considered from the bottom-up perspective. The nontriviality is that the simulator has a set of operators O_P that is much smaller than O_L . For instance, O_P is often finite.

8.1.2 Simulation using rebits

The task we focus on in this chapter is the simulation of n qubits using $n + 1$ rebits. In particular, we make use of the single-ancilla rebit encoding of qubits [170]. Using the definition of simulation given in Section 8.1.1, our rebit simulation is a tuple (L, P, \mathcal{P}, O) , where L is the set of n -qubit states, P is the set of $(n + 1)$ -rebit states, and the *encoding* map $\mathcal{P} : L \rightarrow P$ takes n -qubit states to $(n + 1)$ -rebit states as follows:

$$\mathcal{P} : |\psi\rangle \mapsto \Re |\psi\rangle \otimes |0\rangle_a + \Im |\psi\rangle \otimes |1\rangle_a, \quad (8.1)$$

where subscripts a indicate the “ancilla” rebit, and \Re and \Im take the real and imaginary parts, respectively. The inverse of \mathcal{P} is (as we show later) the *decoding* map

$$\mathcal{L} : |\phi\rangle \mapsto (\langle 0| + i \langle 1|)_a |\phi\rangle. \quad (8.2)$$

Handed only an unknown n -qubit state, the encoding operator \mathcal{P} is nonlinear and thus unphysical. However, n -qubit states with only real amplitudes can be encoded simply by appending an ancilla $|0\rangle$ to the register. Since quantum algorithms (i.e. those solving problems in BQP [29]) start on the all-zeroes state $|0\rangle^{\otimes n}$, the inability to start a rebit simulation from an arbitrary, unknown state is not a problem for this standard computational model.⁴

Our primary task is to study rebit simulations corresponding to different sets O of operators. Our main results concern bottom-up simulation using rebits (i.e. simulations corresponding to sets $O = O_P$), though to draw a contrast, we include a discussion of top-down simulation using rebits (where we choose $O = O_L$) in Section 8.4.1.

⁴Indeed, a similar situation is forced upon error-correcting stabilizer codes, for which only certain states (e.g. stabilizer states and maybe certain magic states depending on the code) are able to be fault-tolerantly prepared.

8.1.2.1 Bottom-up simulation using rebits

We now introduce some terminology for sets of operators that will be useful for describing the bottom-up simulation using rebits. Rather than specify O_P or O_L directly, we specify instead a physically motivated set of operators O that we can actually implement on $n + 1$ qubits in the lab. In general, however, these operators O do not map P to P , i.e. rebits to rebits. Thus, we need to restrict the simulator to operators O_P that do. To do this, take $O_P = O \cap R_{n+1} \subseteq O$, where R_{n+1} is the set of real linear operators on $n + 1$ qubits. Correspondingly, there is a set of operators O_L that behave the same way on the n -qubit states in L that operators in O_P behave on P : i.e. $A \in O_L$ if and only if there is $B \in O_P$ such that $A|\psi\rangle = \mathcal{L}(BP(|\psi\rangle))$ for all $|\psi\rangle \in L$. With a convenient abuse of notation, we denote this set $O_L = \mathcal{L}(O_P)$. These are the operators on L being simulated.

We use the following language to describe these sets O , O_P , and O_L (see also Figure 8-1). If O is the set of \mathbb{S} operators⁵ (for example, the unitary operators), then we call O_P the set of real \mathbb{S} operators (denoted $O^{\mathbb{R}}$) on P , and O_L the set of \mathbb{R} - \mathbb{S} operators (denoted $\mathbb{R}O$) on L . For example, if $O = U_{n+1}$ is the set of $(n + 1)$ -qubit unitary operators, then $O_P = U_{n+1}^{\mathbb{R}}$ is the set of $(n + 1)$ -rebit *real unitary* operators (also called orthogonal operators) and $O_L = \mathbb{R}U_n$ is the set of n -qubit *\mathbb{R} -unitary* operators.⁶

8.1.3 Our results

In this chapter, we study the rebit simulation from a bottom-up perspective. Thus, one important task which we undertake is to determine, for various sets of \mathbb{S} operators, what the corresponding set \mathbb{R} - \mathbb{S} is. In the case of unitaries, we can then proceed to find universal gate sets for the \mathbb{R} -unitaries. We are led naturally to consider the efficiency of simulating these gate sets, and for specific subgroups (the \mathbb{R} -Clifford subgroup) we can even find efficient classical simulations.

⁵ \mathbb{S} is a placeholder for the word used to describe operators in the set O . For example, \mathbb{S} could stand for the word ‘unitary’ or ‘linear’.

⁶Note that *\mathbb{R} -unitary* (which we pronounce as ‘R unitary’) should not be confused with real unitary, which means real and unitary. The same goes for \mathbb{R} -linear, \mathbb{R} -Pauli, etc.

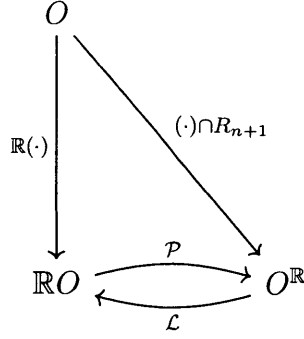


Figure 8-1: Commutative diagram illustrating the relationships between the sets O , $\mathbb{R}O$ and $O^{\mathbb{R}}$.

In this subsection, we aim to highlight these results with a self-contained, albeit brief, presentation. References are provided to theorems and proofs in the main text (i.e. the following sections) where the details can be perused. See Figure 8-2 for a summary of the various sets of operators considered in this chapter.

8.1.3.1 Characterization of various rebit simulators

First, we find \mathbb{R} -S for the following subsets S: (1) linear operators, (2) unitary operators, (3) Pauli operators, (4) operators in the k th level of the Clifford hierarchy. The following theorem summarizes our results.

Theorem 106. Let Γ be an operator on n qubits.

1. (Theorem 122) Γ is \mathbb{R} -linear if and only if there exist complex linear operators A and B such that Γ can be written as

$$\Gamma = A + BK, \tag{8.3}$$

where K denotes the complex conjugation operator, $K : |\psi\rangle \mapsto \Re|\psi\rangle - i\Im|\psi\rangle$.

2. (Theorem 127) Γ is \mathbb{R} -unitary if and only if $\Gamma = A + BK$ is \mathbb{R} -linear and A and B are complex linear operators satisfying the ‘unitarity’ constraints

$$A^\dagger A + B^T \bar{B} = I$$

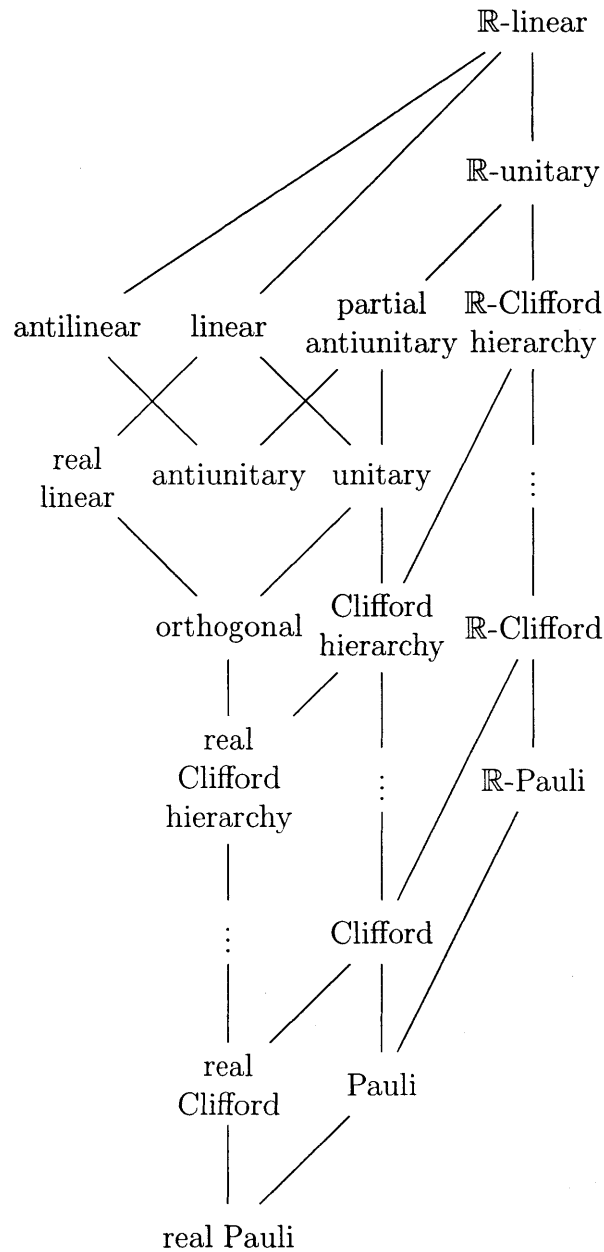


Figure 8-2: Diagram illustrating the relationships between different classes of operators considered in this chapter. A line from \mathcal{A} to \mathcal{B} (where \mathcal{A} is higher than \mathcal{B}) means that \mathcal{A} is a proper superset of \mathcal{B} . The ellipses represent the infinite towers of classes corresponding to different levels of the respective Clifford hierarchies.

$$A^\dagger B + B^T \bar{A} = 0. \quad (8.4)$$

3. (Theorem 167) Γ is \mathbb{R} -Pauli if and only if it can be written as

$$\Gamma = PK^b, \quad (8.5)$$

where $P = i^c p_1 \otimes \dots \otimes p_n$ is a Pauli operator (where $c \in \{0, 1, 2, 3\}$ and $p_j \in \{I, X, Y, Z\}$), and $b \in \{0, 1\}$.

4. (Theorem 170) Γ is in the k th level of the \mathbb{R} -Clifford hierarchy \mathbb{RC}_k if and only if $\Gamma(\mathbb{RC}_1)\Gamma^\dagger \subseteq \mathbb{RC}_{k-1}$, where \mathbb{RC}_1 is the set of \mathbb{R} -Paulis.

The reader might recognize that, in operator theory and linear algebra⁷, the term \mathbb{R} -linear is also used to describe a map $f : V \rightarrow V'$, where V and V' are complex vector spaces, that satisfies

$$f(ax + by) = af(x) + bf(y) \quad (8.6)$$

for all $x, y \in V$ and $a, b \in \mathbb{R}$. It turns out⁸ that this definition is equivalent to that in Part 1 of Theorem 106 (see Theorem 122). It may not be surprising that \mathbb{R} -linearity turns out to be the defining characteristic of the simulated system; after all, the encoding map \mathcal{P} defined in Eq. (8.1) is blatantly \mathbb{R} -linear in the linear algebraic sense of Eq. (8.6). More detail on the linear algebraic definition of \mathbb{R} -linearity is included for reference in Appendices 8.8.3 and 8.8.4.

The \mathbb{R} -unitary operators (part 2 of Theorem 106) are of special importance. Since they can be simulated by (real) unitary operators acting on rebits, they correspond to the set of operators that can be simulated by physical systems using the rebit encoding. This is an example of bottom-up simulation: we start with the set of operators that we can perform on rebits (i.e. real unitary operators) and derive the

⁷See, for example, [91, 129, 130]. Note that our definition of \mathbb{R} -linearity coincides with their definition of real linearity.

⁸More accurately, we chose the terminology so that the two definitions of \mathbb{R} -linearity coincide.

set of operators (i.e. \mathbb{R} -unitaries) that we can simulate on qubits. The rest of our results will concern various properties of the \mathbb{R} -unitary operators.

8.1.3.2 \mathbb{R} -unitaries as products of partial antiunitaries

Our second result concerns the class of partial antiunitary operators, which are a subset (but not a subgroup) of the \mathbb{R} -unitary operators. In [170], McKague, Mosca, and Gisin note that partial antiunitaries – described as operators “which act only on a subspace” – as well as products of partial antiunitaries, can be simulated using the rebit encoding. However, a precise definition of partial antiunitarity was not provided in [170]. In this chapter, we propose the following definition:

Definition 107. Let $S \subseteq \mathcal{H}$ be a subspace of a complex (finite-dimensional) vector space \mathcal{H} . An operator Γ on \mathcal{H} is a *partial antiunitary* operator with respect to S if

- (i) Γ is additive, i.e. $\Gamma(\psi + \phi) = \Gamma(\psi) + \Gamma(\phi)$ for all $\psi, \phi \in \mathcal{H}$.
- (ii) Γ is unitary on S^\perp , i.e. $\langle \Gamma(\psi), \Gamma(\phi) \rangle = \langle \psi, \phi \rangle$ for all $\psi, \phi \in S^\perp$.
- (iii) Γ is antiunitary on S , i.e. $\langle \Gamma(\psi), \Gamma(\phi) \rangle = \langle \phi, \psi \rangle$ for all $\psi, \phi \in S$.
- (iv) $\langle \Gamma(\psi), \Gamma(\phi) \rangle = 0$ for all $\psi \in S, \phi \in S^\perp$.

Perhaps surprisingly, this definition leads to the following relation between partial antiunitaries and \mathbb{R} -unitaries.

Theorem 108. (Theorem 163) For any \mathbb{R} -unitary operator Γ , there exists an integer $0 < k \leq (n + 1)2^n(2^n - 1)/2$ and partial antiunitary operators $\Gamma_1, \dots, \Gamma_{k-1}, \Gamma_k$ such that $\Gamma = \Gamma_k \Gamma_{k-1} \dots \Gamma_1$.

Theorem 108 tells us that in order to simulate an \mathbb{R} -unitary, it suffices to just simulate sequences of partial antiunitary operators. Indeed, we can find universal gate sets for the \mathbb{R} -linear operators that consist of only finitely many partial antiunitary gates. Some of the simplest are presented here in terms of the partial antiunitaries

CK_i (a single-qubit gate on qubit i) and CCK_{ij} (a two-qubit gate on qubits i, j) defined by (i.e. they conjugate only the amplitudes of basis states in which the “control” qubits are 1)

$$CK_i : \sum_{x \in \{0,1\}^n} (a_x + ib_x) |x\rangle \mapsto \sum_{\substack{x \in \{0,1\}^n \\ x_i=0}} (a_x + ib_x) |x\rangle + \sum_{\substack{x \in \{0,1\}^n \\ x_i=1}} (a_x - ib_x) |x\rangle \quad (8.7)$$

$$CCK_{ij} : \sum_{x \in \{0,1\}^n} (a_x + ib_x) |x\rangle \mapsto \sum_{\substack{x \in \{0,1\}^n \\ x_i x_j=0}} (a_x + ib_x) |x\rangle + \sum_{\substack{x \in \{0,1\}^n \\ x_i x_j=1}} (a_x - ib_x) |x\rangle \quad (8.8)$$

as well as the unitary operators controlled-controlled- Z (CCZ), global phase $G(\theta) |\psi\rangle = e^{i\theta} |\psi\rangle$, and Hadamard gate H .

Theorem 109. (Proposition 160) The gate sets

$$\{H, CCK, G(\pi/4)\} \text{ and } \{H, CCZ, CK, G(\pi/4)\}$$

are able to approximately⁹ simulate any \mathbb{R} -unitary operator.

8.1.3.3 Computational complexity and a generalization of the Gottesman-Knill Theorem

Third, we explore the efficiency of the rebit encoding with regards to universal gate sets for the top-down *and* bottom-up simulations.

Theorem 110. Let C be a depth- d circuit on n qubits.

1. (Theorem 164) If C is a unitary circuit consisting of gates from $\{H, T, \text{CNOT}\}$, then C (applied to $|0\rangle^{\otimes n}$) can be simulated using either an orthogonal circuit of depth at most dn on $n + 1$ rebits or an orthogonal circuit of depth at most d on $2n$ rebits.

⁹We have a specific notion of approximation in mind. In principle, any metric on P (the simulator space) implies a metric on L (the simulated space), and likewise metrics can be defined for operators on those spaces, even if those operators on L are nonlinear. To make an operator norm on \mathbb{R} -linears for instance, one instead evaluates the operator norm of their simulations. See the definitions at the end of Section 8.2.2.1 for more rigor on the \mathbb{R} -linear operator norm.

2. (Theorem 166) If C is an \mathbb{R} -unitary circuit consisting of gates from

$$\{H, CCZ, CK, G(\pi/4)\},$$

then C (applied to $|0\rangle^{\otimes n}$) can be simulated using either an orthogonal circuit of depth at most dn on $n + 1$ rebits or an orthogonal circuit of depth at most $d\lceil\log_2 n\rceil$ on $2n$ rebits.

Further exploration of the \mathbb{R} -Clifford circuits reveals an efficient classical simulation, enlarging the scope of the Gottesman-Knill simulation.

Theorem 111. If C is an \mathbb{R} -Clifford circuit on n qubits, then there is an efficient classical algorithm to sample from $C|0\rangle^{\otimes n}$ in time $O(n^2)$, when the output qubits of the circuit are measured in the computational basis.

The \mathbb{R} -Clifford circuits are a strictly larger set of operators than the Clifford circuits. For instance, CK is an \mathbb{R} -Clifford gate that is not even linear, much less Clifford.

8.1.4 Related work

The use of rebits in quantum computation dates back to the 1990s, where it was shown that real amplitudes (or even rational amplitudes [11]) suffice for universal quantum computation [29]. This was first proven in the quantum Turing machine model [29], before direct proofs of this result for the quantum circuit model were found [12, 39, 145, 196, 203]. These circuit-model proofs all involve proving the existence of computationally universal gate sets that consist of only gates with real coefficients. A simple example of such a gate set is the set containing Toffoli and Hadamard gates [12, 203].

Besides the single-ancilla rebit encoding, other rebit encodings have been proposed. An example is the subsystem-division-respecting encoding introduced by [170]. Unlike the single-ancilla encoding, local operators remain local under this encoding. As noted in [169], an implication of this result is that no experiment can distinguish

between quantum mechanics with real amplitudes and quantum mechanics with complex amplitudes, unless one makes assumptions about the dimensions of systems.

Rebits have also been studied in relation to several different topics in quantum information theory. For example, it was shown that states and measurements on a real Hilbert space are sufficient for the maximal violation of Bell inequalities [170, 183]. Another example is in computational complexity theory: as we discussed above, choosing to use rebits instead of qubits does not change the power of BQP. It was shown in [169] that such a choice also does not change the power of many other quantum complexity classes, like QMA, QCMA or QIP(k).

Finally, we note that the techniques used in analyzing rebit circuits are closely related to that used in analyzing quaternionic circuits. It can be shown, for example, that quaternionic quantum circuits are no more powerful than complex quantum circuits [101], just as complex quantum circuits are no more powerful than real ones.

8.1.5 Notation

We denote the set of (complex) linear operators on $\mathcal{H}_n(\mathbb{C})$ by L_n , and the set of real linear operators by R_n , i.e.

$$R_n = \{\Gamma \in L_n : \Im\Gamma = 0\}. \quad (8.9)$$

The group of unitary operators on $\mathcal{H}_n(\mathbb{C})$ is denoted by $U_n = \{U \in L_n : U^\dagger U = I\}$. The group of orthogonal operators is denoted by $T_n = \{T \in U_n : \Im T = 0\}$. Technically, while it is clear that L_n and U_n cannot act on the rebit space $\mathcal{H}_n(\mathbb{R})$ (due to failing closure), it is possible for the operators in R_n and T_n to act on either $\mathcal{H}_n(\mathbb{C})$ or $\mathcal{H}_n(\mathbb{R})$ (i.e. on qubits or rebits). We generally let context sort this ambiguity out, and moreover, because all these operators can be represented as matrices independent of the vector spaces on which they act, this subtlety should not be an issue.

For operators A, B , we write $A \sim B$ if there exists $\theta \in \mathbb{R}$ such that $A = e^{i\theta} B$ (see Chapter 2).

8.2 Quantum circuits with rebits

We start by introducing the rebit encoding and decoding maps for quantum circuits. A quantum circuit is in general described in terms of its states, operators and measurements. We will now describe each of these separately.

8.2.1 Rebit encoding and decoding of states

We restrict our attention to pure states. Note that we do not lose any generality with this restriction since any mixed state can be purified to a pure state by adding ancilla qubits [180].

Definition 112. The *rebit encoding* of a quantum state $|\psi\rangle \in \mathcal{H}_n(\mathbb{C})$ is the state $\mathcal{P}(|\psi\rangle) \in \mathcal{H}_{n+1}(\mathbb{R})$ defined by

$$\mathcal{P}(|\psi\rangle) = \Re |\psi\rangle \otimes |0\rangle_a + \Im |\psi\rangle \otimes |1\rangle_a. \quad (8.10)$$

Here, the real and imaginary parts (denoted by \Re and \Im respectively) of $|\psi\rangle$ are defined with respect to the computational basis.

Following the terminology in Section 8.1.2, the space of qubits $\mathcal{H}_n(\mathbb{C})$ may be thought of as the *logical space* L and the space of rebits $\mathcal{H}_{n+1}(\mathbb{R})$ may be thought of as the *physical space* P that encodes states in $\mathcal{H}_n(\mathbb{C})$ through the map \mathcal{P} .

Explicitly, if

$$|\psi\rangle = \sum_{i_1, \dots, i_n \in \{0,1\}^n} \psi_{i_1, \dots, i_n} |i_1, \dots, i_n\rangle,$$

then

$$\mathcal{P}(|\psi\rangle) = \sum_{i_1, \dots, i_n, j \in \{0,1\}^{n+1}} \psi_{i_1, \dots, i_n, j} |i_1, \dots, i_n, j\rangle,$$

where we have used the notation $\psi_{i_1, \dots, i_n, 0} = \Re(\psi_{i_1, \dots, i_n})$ and $\psi_{i_1, \dots, i_n, 1} = \Im(\psi_{i_1, \dots, i_n})$.

The normalization condition $\sum_{i_1, \dots, i_n \in \{0,1\}^n} |\psi_{i_1, \dots, i_n}|^2 = 1$ becomes

$$\sum_{i_1, \dots, i_n, j \in \{0,1\}^{n+1}} \psi_{i_1, \dots, i_n, j}^2 = 1 \quad (8.11)$$

in the encoded space. For example, when $n = 1$ and each entry of the state $|\psi\rangle$ is written in terms of its real and imaginary parts, $\mathcal{P}(\cdot)$ acts on $|\psi\rangle$ as follows:

$$\mathcal{P} : (a + ib) |0\rangle + (c + id) |1\rangle \mapsto a |00\rangle + b |01\rangle + c |10\rangle + d |11\rangle. \quad (8.12)$$

Incidentally, this discussion shows that \mathcal{P} preserves the l_2 norm.

Proposition 113. Denote the l_2 norm for $|\psi\rangle \in \mathcal{H}_n(\mathbb{C})$ as $\| |\psi\rangle \| = |\langle \psi | \psi \rangle|^{1/2}$. Then $\| |\psi\rangle \| = \| \mathcal{P}(|\psi\rangle) \|$ for all $|\psi\rangle \in \mathcal{H}_n(\mathbb{C})$.

Next, we show that the map \mathcal{P} is invertible and that its inverse $\mathcal{L} : \mathcal{H}_{n+1}(\mathbb{R}) \rightarrow \mathcal{H}_n(\mathbb{C})$ is given by

$$\mathcal{L} : |\phi\rangle \mapsto (\langle 0| + i \langle 1|)_a |\phi\rangle. \quad (8.13)$$

We refer to $\mathcal{L}(|\phi\rangle)$ as the rebit decoding of the rebit state $|\phi\rangle$.

Proposition 114. The maps \mathcal{P} and \mathcal{L} defined in Eq. (8.10) and Eq. (8.13) are inverses of each other.

Proof. For all $|\psi\rangle$ and $|\phi\rangle$,

$$\mathcal{L} \circ \mathcal{P} |\psi\rangle = (\langle 0| + i \langle 1|)_a (\Re |\psi\rangle \otimes |0\rangle_a + \Im |\psi\rangle \otimes |1\rangle_a) = \Re |\psi\rangle + i \Im |\psi\rangle = |\psi\rangle$$

and

$$\begin{aligned} \mathcal{P} \circ \mathcal{L} |\phi\rangle &= \mathcal{P}(\langle 0|_a |\phi\rangle + i \langle 1|_a |\phi\rangle) \\ &= \langle 0|_a |\phi\rangle \otimes |0\rangle_a + \langle 1|_a |\phi\rangle \otimes |1\rangle_a \\ &= (|0\rangle\langle 0| + |1\rangle\langle 1|)_a |\phi\rangle \\ &= |\phi\rangle. \end{aligned}$$

Hence, $\mathcal{L} \circ \mathcal{P} = I$ and $\mathcal{P} \circ \mathcal{L} = I$. □

We noted in Section 8.1.3.1 that \mathcal{P} is \mathbb{R} -linear. Here we also note that \mathcal{L} is linear. We collect these observations in the following proposition.

Proposition 115. \mathcal{P} is \mathbb{R} -linear on $\mathcal{H}_n(\mathbb{C})$ and \mathcal{L} is linear on $\mathcal{H}_{n+1}(\mathbb{R})$. That is, for all $|\psi_1\rangle, |\psi_2\rangle \in \mathcal{H}_n(\mathbb{C})$ and all $a, b \in \mathbb{R}$,

$$\mathcal{P}(a|\psi_1\rangle + b|\psi_2\rangle) = a\mathcal{P}(|\psi_1\rangle) + b\mathcal{P}(|\psi_2\rangle), \quad (8.14)$$

and for all $|\phi_1\rangle, |\phi_2\rangle \in \mathcal{H}_{n+1}(\mathbb{R})$ and $\alpha, \beta \in \mathbb{R}$,

$$\mathcal{L}(\alpha|\phi_1\rangle + \beta|\phi_2\rangle) = \alpha\mathcal{L}(|\phi_1\rangle) + \beta\mathcal{L}(|\phi_2\rangle). \quad (8.15)$$

8.2.2 Rebit encoding and decoding of operators

Having defined how states are encoded, we now proceed to describe how operators on qubits are encoded. Let $\Gamma : \mathcal{H}_n(\mathbb{C}) \rightarrow \mathcal{H}_n(\mathbb{C})$ be any operator on the set of n -qubit states. We define the rebit encoding of Γ to be the map $\mathcal{P}(\Gamma) : \mathcal{H}_{n+1}(\mathbb{R}) \rightarrow \mathcal{H}_{n+1}(\mathbb{R})$ given by

$$\mathcal{P}(\Gamma) : |\phi\rangle \mapsto \mathcal{P}(\Gamma\mathcal{L}(|\phi\rangle)). \quad (8.16)$$

Note that we have used the same symbol \mathcal{P} to denote the rebit encoding of both states and operators.

Let $W : \mathcal{H}_{n+1}(\mathbb{R}) \rightarrow \mathcal{H}_{n+1}(\mathbb{R})$ be any operator on the set of $(n+1)$ -rebit states. We define the rebit decoding of W to be the map $\mathcal{L}(W) : \mathcal{H}_n(\mathbb{C}) \rightarrow \mathcal{H}_n(\mathbb{C})$ given by

$$\mathcal{L}(W) : |\psi\rangle \mapsto \mathcal{L}(W\mathcal{P}(|\psi\rangle)). \quad (8.17)$$

The above definitions are chosen so that the rebit encoding \mathcal{P} of operators and the rebit decoding \mathcal{L} of operators are inverses of each other.

It is easily established that \mathcal{P} and \mathcal{L} for operators are group homomorphisms.

Proposition 116. For all $\Gamma_1, \Gamma_2 : \mathcal{H}_n(\mathbb{C}) \rightarrow \mathcal{H}_n(\mathbb{C})$, we have $\mathcal{P}(\Gamma_1\Gamma_2) = \mathcal{P}(\Gamma_1)\mathcal{P}(\Gamma_2)$. Likewise, for all $W_1, W_2 : \mathcal{H}_{n+1}(\mathbb{R}) \rightarrow \mathcal{H}_{n+1}(\mathbb{R})$, we have $\mathcal{L}(W_1W_2) = \mathcal{L}(W_1)\mathcal{L}(W_2)$.

Proof. We just prove the first statement here; the second is just as simple an appli-

cation of the definitions. For all $|\phi\rangle \in \mathcal{H}_n(\mathbb{C})$,

$$\begin{aligned}
\mathcal{P}(\Gamma_1)\mathcal{P}(\Gamma_2)|\phi\rangle &= \mathcal{P}(\Gamma_1)\mathcal{P}(\Gamma_2\mathcal{L}(|\phi\rangle)) \\
&= \mathcal{P}(\Gamma_1\mathcal{L}(\mathcal{P}(\Gamma_2\mathcal{L}(|\phi\rangle)))) \\
&= \mathcal{P}(\Gamma_1\Gamma_2\mathcal{L}(|\phi\rangle)) \\
&= \mathcal{P}(\Gamma_1\Gamma_2)|\phi\rangle.
\end{aligned}$$

using Proposition 114 in the second-to-last step. \square

The definitions also imply that the encodings of states and operators behave naturally together, and likewise for decodings of the two,

$$\mathcal{P}(\Gamma)\mathcal{P}(|\phi\rangle) = \mathcal{P}(\Gamma|\phi\rangle), \quad (8.18)$$

$$\mathcal{L}(W)\mathcal{L}(|\psi\rangle) = \mathcal{L}(W|\psi\rangle). \quad (8.19)$$

These relations are also easily checked. Similarly to the case of states, the rebit encoding and decodings of operators have the same linearity properties.

Proposition 117. \mathcal{P} (for operators) is \mathbb{R} -linear on L_n and \mathcal{L} (for operators) is linear on R_{n+1} . That is, for all $\Gamma_1, \Gamma_2 \in L_n$ and all $a, b \in \mathbb{R}$,

$$\mathcal{P}(a\Gamma_1 + b\Gamma_2) = a\mathcal{P}(\Gamma_1) + b\mathcal{P}(\Gamma_2), \quad (8.20)$$

and for all $W_1, W_2 \in R_{n+1}$ and all $\alpha, \beta \in \mathbb{R}$,

$$\mathcal{L}(\alpha W_1 + \beta W_2) = \alpha\mathcal{L}(W_1) + \beta\mathcal{L}(W_2). \quad (8.21)$$

The proof, left to the reader, is a straightforward application of the definitions. We remark that the above treatment can also be extended in a similar way to superoperators, i.e. operators acting on operators. Such a generalization is used in, for example, Proposition 181.

The treatment so far deals with general operators. Since quantum mechanics is

a linear (in fact, unitary) theory, we shall henceforth restrict our attention to only those operators on rebits which are linear, and adopt the terminology described in Section 8.1.2.1 to describe various subsets of the linear transformations.

Let $O_n \subseteq L_n$ be any subset of linear operators on $\mathcal{H}_n(\mathbb{C})$. We are interested in the following two classes:

1. $O_n^{\mathbb{R}} := O_n \cap R_n$
2. $\mathbb{R}O_n = \mathcal{L}(O_{n+1}^{\mathbb{R}})$

which correspond respectively to the real **S** and **R-S** sets defined in Section 8.1.2.1.

In general, the sets O_n need not have additional structure. In this chapter though, many of the sets O_n that we consider are also groups. When this is the case, since R_n is a subgroup of L_n and the intersection of subgroups is a subgroup, the set $O_n^{\mathbb{R}}$ is also a subgroup of L_n . Since \mathcal{L} is a group homomorphism (Proposition 116), and group homomorphisms preserve subgroups, we get the following result:

Proposition 118. Let $O_n \subseteq L_n$ be any subset of linear operators on $\mathcal{H}_n(\mathbb{C})$. If O_n is a (proper) subgroup of L_n , then $\mathbb{R}O_n$ is a (proper) subgroup of $\mathbb{R}L_n$.

In Sections 8.2.2.1 and 8.2.2.2, we prove parts 1 and 2 of Theorem 106, thereby characterizing the sets $O_n^{\mathbb{R}}$ and $\mathbb{R}O_n$, when (i) $O_n = L_n$ is the set of linear operators, and when (ii) $O_n = U_n$ is the set of unitary operators. We defer the proof of the rest of Theorem 106 to Section 8.6 where the \mathbb{R} -Clifford hierarchy is discussed.

8.2.2.1 \mathbb{R} -linear operators

Consider the set L_n of linear operators. In this section, we characterize the sets (i) $L_n^{\mathbb{R}}$ (the real linear operators) and (ii) $\mathbb{R}L_n$ (the \mathbb{R} -linear operators). Part (i) is straightforward, since $L_n^{\mathbb{R}} = L_n \cap R_n = R_n$ is the set of real linear operators. Part (ii) is more involved. To begin, we prove the following lemma.

Lemma 119. Let $W : \mathcal{H}_{n+1}(\mathbb{R}) \rightarrow \mathcal{H}_{n+1}(\mathbb{R})$ be a real linear operator. Then

$$\mathcal{L}(W) = \left(\frac{1}{2} \text{tr}_a[W(I_a - iX_a Z_a)] \right) + \left(\frac{1}{2} \text{tr}_a[W(Z_a + iX_a)] \right) K. \quad (8.22)$$

Proof. We consider the evolution of an arbitrary state $|\psi\rangle$ under $\mathcal{L}(W) = \mathcal{L} \circ W \circ \mathcal{P}$:

$$\begin{aligned}
|\psi\rangle &\xrightarrow{\mathcal{P}} \Re|\psi\rangle|0\rangle_a + \Im|\psi\rangle|1\rangle_a \\
&\xrightarrow{W} W|0\rangle_a \Re|\psi\rangle + W|1\rangle_a \Im|\psi\rangle \\
&\xrightarrow{\mathcal{L}} [(\langle 0|_a + i\langle 1|_a)W|0\rangle_a \Re + (\langle 0|_a + i\langle 1|_a)W|1\rangle_a \Im]|\psi\rangle. \quad (8.23)
\end{aligned}$$

Hence,

$$\begin{aligned}
\mathcal{L}(W) &= (\langle 0|_a + i\langle 1|_a)W|0\rangle_a \Re + (\langle 0|_a + i\langle 1|_a)W|1\rangle_a \Im \\
&= \frac{1}{2}(\langle 0|_a + i\langle 1|_a)W|0\rangle_a (I + K) + \frac{1}{2i}(\langle 0|_a + i\langle 1|_a)W|1\rangle_a (I - K) \\
&= \frac{1}{2}[(\langle 0|_a W|0\rangle_a + i\langle 1|_a W|0\rangle_a - i\langle 0|_a W|1\rangle_a + \langle 1|_a W|1\rangle_a)I \\
&\quad + (\langle 0|_a W|0\rangle_a + i\langle 1|_a W|0\rangle_a + i\langle 0|_a W|1\rangle_a - \langle 1|_a W|1\rangle_a)K] \\
&= \frac{1}{2}\{\text{tr}_a[W(|0\rangle\langle 0| + i|0\rangle\langle 1| - i|1\rangle\langle 0| + |1\rangle\langle 1|)]I \\
&\quad + \text{tr}_a[W(|0\rangle\langle 0| + i|0\rangle\langle 1| + i|1\rangle\langle 0| - |1\rangle\langle 1|)]K\} \\
&= \left(\frac{1}{2}\text{tr}_a[W(I_a - iX_a Z_a)]\right) + \left(\frac{1}{2}\text{tr}_a[W(Z_a + iX_a)]\right) K. \quad (8.24)
\end{aligned}$$

□

The above proposition shows that the \mathbb{R} -linear operators can always be written in the form $A + BK$. It happens the converse is also true: any operator of the form $A + BK$ is an \mathbb{R} -linear operator, as the following lemma proves.

Lemma 120. Let $A, B : \mathcal{H}_n(\mathcal{C}) \rightarrow \mathcal{H}_n(\mathcal{C})$ be complex linear operators. Then,

$$\mathcal{P}(A + BK) = \Re A \otimes I + \Im A \otimes XZ + \Re B \otimes Z + \Im B \otimes X, \quad (8.25)$$

which is a real linear operator.

Proof. We consider the evolution of an arbitrary state $|\phi\rangle$ under $\mathcal{P}(A + BK) = \mathcal{P} \circ (A + BK) \circ \mathcal{L}$. Writing $A + BK = C\Re + D\Im$, where $C = A + B$ and $D = i(A - B)$,

we get

$$\begin{aligned}
|\phi\rangle &\xrightarrow{\mathcal{L}} \langle 0|_a \cdot |\phi\rangle + i \langle 1|_a \cdot |\phi\rangle \\
&\xrightarrow{A+BK} C \langle 0|_a \cdot |\phi\rangle + D \langle 1|_a \cdot |\phi\rangle \\
&\xrightarrow{\mathcal{P}} (\Re C \langle 0|_a \cdot |\phi\rangle + \Re D \langle 1|_a \cdot |\phi\rangle) \otimes |0\rangle_a \\
&\quad + (\Im C \langle 0|_a \cdot |\phi\rangle + \Im D \langle 1|_a \cdot |\phi\rangle) \otimes |1\rangle_a \\
&= (\Re C \otimes |0\rangle\langle 0| + \Im C \otimes |1\rangle\langle 0| + \Re D \otimes |0\rangle\langle 1| + \Im D \otimes |1\rangle\langle 1|) |\phi\rangle.
\end{aligned} \tag{8.26}$$

$$\tag{8.27}$$

Hence,

$$\begin{aligned}
\mathcal{P}(A + BK) &= \Re(A + B) \otimes |0\rangle\langle 0| + \Im(A + B) \otimes |1\rangle\langle 0| \\
&\quad + \Re(i(A - B)) \otimes |0\rangle\langle 1| + \Im(i(A - B)) \otimes |1\rangle\langle 1| \\
&= \Re A \otimes (|0\rangle\langle 0| + |1\rangle\langle 1|) + \Re B \otimes (|0\rangle\langle 0| - |1\rangle\langle 1|) \\
&\quad + \Im A \otimes (|1\rangle\langle 0| - |0\rangle\langle 1|) + \Im B \otimes (|1\rangle\langle 0| + |0\rangle\langle 1|) \\
&= \Re A \otimes I + \Im A \otimes XZ + \Re B \otimes Z + \Im B \otimes X.
\end{aligned} \tag{8.28}$$

□

Note that Eq. (8.25) is a generalization of Eq. (9) of [101]. Indeed, when $B = 0$, we obtain an expression for the rebit encoding \mathcal{P} of the linear operator A in terms of its real and imaginary parts¹⁰:

$$\mathcal{P}(A) = \Re A \otimes I + \Im A \otimes XZ. \tag{8.29}$$

Lemmas 119 and 120 tell us that an operator Γ is \mathbb{R} -linear if and only if it can be written in the form $\Gamma = A + BK$, where A and B are complex linear operators. It turns out that operators of the form $A + BK$ have the following alternative characterization (see Chapter 8.8.3):

¹⁰See Chapter 8.8.5 for equivalent expressions of $\mathcal{P}(A)$.

Theorem 121. Let V and V' be complex vector spaces, and $f : V \rightarrow V'$ be a function on V . Then, there exist linear maps A and B such that $f = A + BK$ if and only if

$$f(ax + by) = af(x) + bf(y) \quad (8.30)$$

for all $a, b \in \mathbb{R}$ and $x, y \in V$.

Proof. See Theorem 178 of Chapter 8.8.3. □

Hence, we obtain the following equivalent notions of \mathbb{R} -linearity.

Theorem 122. Let $\Gamma : \mathcal{H}_n(\mathbb{C}) \rightarrow \mathcal{H}_n(\mathbb{C})$ be a n -qubit operator. Then the following three statements are equivalent.

1. $\Gamma \in \mathbb{R}L_n$, i.e. Γ is \mathbb{R} -linear.
2. There exist complex linear operators A and B such that $\Gamma = A + BK$.
3. $\Gamma(a|\psi\rangle + b|\phi\rangle) = a\Gamma(|\psi\rangle) + b\Gamma(|\phi\rangle)$ for all $a, b \in \mathbb{R}$ and $|\psi\rangle, |\phi\rangle \in \mathcal{H}_n(\mathbb{C})$.

Before moving on, we address the notion of an operator norm for \mathbb{R} -linear operators. We start by recalling the operator norm for linear operators.

Definition 123. For $\Gamma \in L_n$, the operator norm, denoted $\|\Gamma\|$ is defined as

$$\|\Gamma\| = \inf\{\epsilon \geq 0 : \|\Gamma|\psi\rangle\| \leq \epsilon\|\psi\rangle\|, \forall |\psi\rangle \in \mathcal{H}_n(\mathbb{C})\}. \quad (8.31)$$

Equivalently, this is the largest singular value of Γ . Recall $\|\psi\rangle\|$ is the l_2 norm from Proposition 113.

For \mathbb{R} -linear operators, we advocate a norm built from the norm for linear operators and the encoding map \mathcal{P} .

Definition 124. For $\Gamma \in \mathbb{R}L_n$, let $\|\Gamma\| = \|\mathcal{P}(\Gamma)\|$ be the operator norm.

This norm satisfies the triangle inequality due to the \mathbb{R} -linearity of \mathcal{P} from Proposition 117. To justify using the same notation in Definitions 123 and 124, we do have to check that they coincide when $\Gamma \in L_n$. We verify this in Proposition 186

in Chapter 8.8.6. With the operator norm on \mathbb{R} -linears, we also gain the ability to define ϵ -approximations, and we follow [203] in the inclusion of an ancilla system.

Definition 125. For $\tilde{n} \geq n$, we say $\tilde{\Gamma} \in \mathbb{R}L_{\tilde{n}}$ ϵ -approximates $\Gamma \in \mathbb{R}L_n$ with ancilla $|\psi\rangle \in \mathcal{H}_{\tilde{n}-n}(\mathbb{C})$ if

$$\|\langle \psi | \tilde{\Gamma} | \psi \rangle - \Gamma\| = \|\mathcal{P}(\langle \psi | \tilde{\Gamma} | \psi \rangle) - \mathcal{P}(\Gamma)\| \leq \epsilon, \quad (8.32)$$

where the \mathbb{R} -linearity of \mathcal{P} (Proposition 117) justifies the equality.

8.2.2.2 \mathbb{R} -unitary operators

Consider the set U_n of unitary operators on n -qubits. Then $U_n^{\mathbb{R}} := U_n \cap R_n = T_n$ is the set of real unitary (i.e. orthogonal) operators. To find necessary and sufficient conditions for a operator to be \mathbb{R} -unitary, we compute the image of the set of orthogonal operators under \mathcal{L} :

Theorem 126. Let $A+BK \in \mathbb{R}L_n$ be an \mathbb{R} -linear operator. Then $\mathcal{P}(A+BK) \in T_{n+1}$ if and only if

$$\begin{aligned} A^\dagger A + B^T \bar{B} &= I \\ A^\dagger B + B^T \bar{A} &= 0. \end{aligned} \quad (8.33)$$

Proof. $\mathcal{P}(A+BK) = \Re A \otimes I + \Im A \otimes XZ + \Re B \otimes Z + \Im B \otimes X$ is orthogonal if and only if

$$\begin{aligned} I \otimes I &= (\Re A \otimes I + \Im A \otimes XZ + \Re B \otimes Z + \Im B \otimes X)^T \\ &\quad (\Re A \otimes I + \Im A \otimes XZ + \Re B \otimes Z + \Im B \otimes X) \\ &= (\Re A^T \Re A + \Im A^T \Im A + \Re B^T \Re B + \Im B^T \Im B) \otimes I \\ &\quad + (\Re A^T \Im B - \Im A^T \Re B - \Re B^T \Im A + \Im B^T \Re A) \otimes X \\ &\quad + (\Re A^T \Im A - \Im A^T \Re A - \Re B^T \Im B + \Im B^T \Im B) \otimes XZ \\ &\quad + (\Re A^T \Re B + \Im A^T \Im B + \Re B^T \Re A + \Im B^T \Im A) \otimes Z \end{aligned}$$

$$\begin{aligned}
&= (\Re(A^\dagger A) + \Re(B^\dagger B)) \otimes I + (\Im(A^\dagger B) - \Im(B^\dagger A)) \otimes X \\
&\quad + (\Im(A^\dagger A) - \Im(B^\dagger B)) \otimes XZ + (\Re(A^\dagger B) + \Re(B^\dagger A)) \otimes Z \quad (8.34)
\end{aligned}$$

which holds if and only if

$$\Re(A^\dagger A) + \Re(B^\dagger B) = I \quad (8.35)$$

$$\Im(A^\dagger B) - \Im(B^\dagger A) = 0 \quad (8.36)$$

$$\Im(A^\dagger A) - \Im(B^\dagger B) = 0 \quad (8.37)$$

$$\Re(A^\dagger B) + \Re(B^\dagger A) = 0 \quad (8.38)$$

which holds if and only if

$$\begin{aligned}
A^\dagger A + B^T \bar{B} &= I \\
A^\dagger B + B^T \bar{A} &= 0.
\end{aligned} \quad (8.39)$$

□

In Chapter 8.8.4 (see Proposition 183), we show that an \mathbb{R} -linear operator satisfies Eq. (8.33) if and only if it is a unitary element of the \mathbb{R} -linear group with respect to the dagger operator \dagger defined by

$$(A + BK)^\dagger = A^\dagger + B^T K. \quad (8.40)$$

This, together with Theorem 126, implies the following theorem.

Theorem 127. Let $A + BK \in \mathbb{R}L_n$ be an \mathbb{R} -linear operator. Then $A + BK$ is \mathbb{R} -unitary if and only if $A + BK$ is a unitary element with respect to the \dagger operator defined by Eq. (8.40).

Since the unitary operators are a proper subgroup of the linear operators, Proposition 118 implies that the \mathbb{R} -unitaries are a proper subgroup of the \mathbb{R} -linear operators. In particular, there exist \mathbb{R} -linear operators, like $I + K$, which are not \mathbb{R} -unitary.

Note that since Eqs. (8.33) are not scale invariant, any \mathbb{R} -unitary Γ gives rise to a family of non- \mathbb{R} -unitary elements $c\Gamma$, for $c \in \mathbb{C}$, $|c| \neq 1$.

We note that in the proof of Theorem 126, we used the property that a matrix W is orthogonal if and only if $W^T W = I$. But this is equivalent to the condition that $W W^T = I$. In Chapter 8.8.7, we present an alternative formulation of Theorem 126 that uses the latter criterion for orthogonality.

Finally, we notice that the group of \mathbb{R} -unitary operators has a center $\mathcal{Z}(\mathbb{R}U_n) = \{\pm I\} = \mathbb{R}\mathcal{Z}(U_n)$ in contrast to the center of the unitary group $\mathcal{Z}(U_n) = \{e^{i\phi} I : \phi \in \mathbb{R}\}$, consisting of all global phases. Since arbitrary global phases do not commute with the rest of the \mathbb{R} -unitary group, we might expect to actually observe them. After setting up a framework for encoding and decoding measurements in the next section, we show how such a measurement of the global phase works in Section 8.2.4.

8.2.3 Rebit encoding and decoding of measurements

The final circuit ingredient is measurement. Recall that measurements of quantum systems may be described by a collection of measurement operators $\{M_m\}$ satisfying [180]

$$\sum_m M_m^\dagger M_m = I, \quad (8.41)$$

where the probability that the result m occurs when a state $|\psi\rangle$ is measured is given by

$$\|M_m |\psi\rangle\|^2. \quad (8.42)$$

Let $\{M_m\}$ be a collection of measurement operators on $\mathcal{H}_{n+1}(\mathbb{R})$. We assume that each M_m is real, i.e. $\Im M_m = 0$. Let $\{F_m\}$ be a collection of operators on $\mathcal{H}_n(\mathbb{C})$. We say that a $\{F_m\}$ is a *rebit decoding* of $\{M_m\}$, or that $\{M_m\}$ is a *rebit encoding* of $\{F_m\}$, if for all $|\phi\rangle \in \mathcal{H}_n(\mathbb{R})$, we have

$$\|F_m \mathcal{L} |\phi\rangle\|^2 = \|M_m |\phi\rangle\|^2. \quad (8.43)$$

The collection $\{F_m\}$ can hence be thought of a set of “ \mathbb{R} -measurement operators” on

the space $\mathcal{H}_n(\mathbb{C})$. The probability that the result m occurs when a state $|\psi\rangle \in \mathcal{H}_n(\mathbb{C})$ is measured is given by

$$\|F_m |\psi\rangle\|^2. \quad (8.44)$$

Our next proposition relates $\{M_m\}$ to a rebit decoding of it.

Proposition 128. $\{\mathcal{L}(M_m)\}$ is a rebit decoding of $\{M_m\}$.

Proof. This follows from noting that

$$\|\mathcal{L}(M_m)\mathcal{L}|\phi\rangle\|^2 = \|\mathcal{L}(M_m|\phi\rangle)\|^2 = \|M_m|\phi\rangle\|^2. \quad (8.45)$$

□

It turns out that $\mathcal{L}(M_m)$ also obeys the completeness relation given in Eq. (8.41):

Proposition 129. Let $F_m = \mathcal{L}(M_m)$. Then,

$$\sum_m F_m^\dagger F_m = I.$$

Proof. To prove this, consider

$$\begin{aligned} \sum_m F_m^\dagger F_m &= \sum_m \mathcal{L}(M_m)^\dagger \mathcal{L}(M_m) \\ &= \sum_m \mathcal{L}(M_m^T) \mathcal{L}(M_m), \quad \text{by Proposition 181} \\ &= \mathcal{L}\left(\sum_m M_m^T M_m\right) \\ &= \mathcal{L}(I) = I. \end{aligned} \quad (8.46)$$

□

We now give a top-down example. How is a computational basis measurement on qubits simulated using the rebit encoding? The following proposition tells us that this may be done by simply performing a computational basis measurement on the first n rebits in the encoded circuit, and discarding the ancilla rebit.

Proposition 130. The rebit encoding of the computational basis measurement described by measurement operators $\{|m\rangle\langle m|\}_m$ is given by $\{|m\rangle\langle m| \otimes I_a\}_m$.

Proof. Substituting $W = |m\rangle\langle m| \otimes I_a$ into Eq. (8.22) and using the fact that the Pauli matrices X , Y and Z are traceless, we obtain

$$\mathcal{L}(|m\rangle\langle m| \otimes I_a) = |m\rangle\langle m|. \quad (8.47)$$

Using Proposition 128 completes the proof. Alternatively, we may verify directly that

$$\begin{aligned} \|(|m\rangle\langle m| \otimes I_a)\mathcal{P}(|\psi\rangle)\|^2 &= \|(\langle m| \otimes I_a)(\Re|\psi\rangle \otimes |0\rangle_a + \Im|\psi\rangle \otimes |1\rangle_a)\|^2 \\ &= \|\langle m| \Re|\psi\rangle |0\rangle_a + \langle m| \Im|\psi\rangle |1\rangle_a\|^2 \\ &= \langle m| \Re|\psi\rangle^2 + \langle m| \Im|\psi\rangle^2 \\ &= |\langle m| \Re|\psi\rangle + i\langle m| \Im|\psi\rangle|^2 \\ &= |\langle m|\psi\rangle|^2 \\ &= \|(|m\rangle\langle m|)|\psi\rangle\|^2. \end{aligned}$$

□

We now consider the following bottom-up example. What does it mean to measure the ancilla qubit in the computational basis? The following proposition reveals that this corresponds to measuring the eigenvalue of the complex conjugation operator K , thereby projecting an n -qubit quantum state to its real or imaginary part.

Proposition 131. The rebit decoding of the set of measurement operators $\{(I + Z_a)/2, (I - Z_a)/2\} = \{|0\rangle\langle 0|_a, |1\rangle\langle 1|_a\}$ is $\{(I + K)/2, (I - K)/2\} = \{\Re, \Im\}$.

Proof. By Eq. (8.22),

$$\mathcal{L}(Z_a) = \left(\frac{1}{2}\text{tr}_a[Z_a(I_a - iX_aZ_a)]\right) + \left(\frac{1}{2}\text{tr}_a[Z_a(Z_a + iX_a)]\right) K = K, \quad (8.48)$$

since the Pauli matrices X , Y and Z are all traceless. Using Proposition 128 completes the proof. □

8.2.4 Bottom-up tomography

Given a pure $n + 1$ rebit state $|\phi\rangle$, we sketch a procedure to find the amplitudes up to a global ± 1 sign using only orthogonal operators and single-rebit measurements in the computational basis. This is the natural form of tomography on the simulation space P and translates through the decoding map \mathcal{L} to a tomographic procedure on n -qubit states that makes use of \mathbb{R} -unitary operators, computational basis measurements, and projection onto the eigenspaces of complex conjugation K .

Theorem 132. An n -rebit density matrix is determined exactly by measurement of the $(4^n + 2^n)/2$ observables

$$\mathcal{O} = \{p = p_1 \otimes p_2 \otimes \cdots \otimes p_n : p_j \in \{I, X, Y, Z\}, p = \bar{p}\}. \quad (8.49)$$

Moreover, any given $p \in \mathcal{O}$ can be measured using single-rebit computational basis measurements, along with H and CZ gates.

Proof. Let $\rho = |\phi\rangle\langle\phi|$ be the real density matrix corresponding to $|\phi\rangle$, a generic n -rebit state. Then, because $\rho = \rho^\dagger = \bar{\rho} = \rho^T$, there is a decomposition

$$\rho = \sum_{p \in \mathcal{O}} a_p p \quad (8.50)$$

for some real coefficients $a_p \in \mathbb{R}$. Since $\text{tr}(\rho) = 1$ and all $p \neq I^{\otimes n}$ satisfy $\text{tr}(p) = 0$, we have $a_{I^{\otimes n}} = 2^{-n}$. Evidently then, learning the values of $a_p = \text{tr}(p\rho)$ by repeatedly measuring the observable p effectively learns ρ . Given ρ , $|\phi\rangle$ is determined up to a sign ± 1 .

The size of \mathcal{O} can be calculated by noticing that any $p \in \mathcal{O}$ must have an even number of Y s in its tensor product due to the reality condition $p = \bar{p}$. Thus,

$$|\mathcal{O}| = 3^n + 3^{n-2} \binom{n}{2} + 3^{n-4} \binom{n}{4} + \cdots + 3^{n-2\lfloor n/2 \rfloor} \binom{n}{2\lfloor n/2 \rfloor} = (4^n + 2^n)/2. \quad (8.51)$$

Notice that this is strictly greater than the number of rebit measurements that measure a subset of rebits each in the X -basis and another disjoint subset each in the

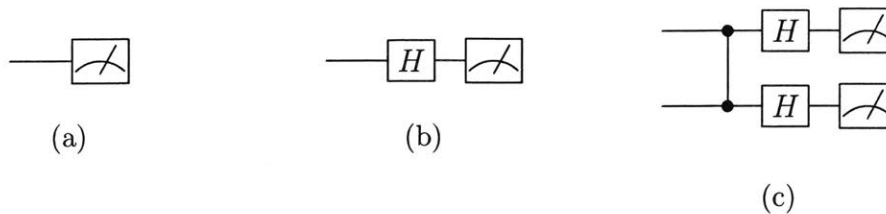


Figure 8-3: The proof of 2-locality of rebit tomography using H , CZ , and single-rebit computational basis measurements. (a) Measuring Z , (b) Measuring X , and (c) Measuring $Y \otimes Y$.

Z -basis – there are 3^n such measurements, corresponding to observables in

$$\mathcal{O}' = \{p_1 \otimes p_2 \otimes \cdots \otimes p_n : p_j \in \{I, X, Z\}\}. \quad (8.52)$$

Since $|\mathcal{O}'| = 3^n < (4^n + 2^n)/2 = |\mathcal{O}|$ for all $n > 1$, there is no way to satisfy tomographic locality [120] in rebit tomography. That is, there is no way to completely learn a rebit state $|\phi\rangle$ using only single-rebit orthogonal gates and computational basis measurements. We must therefore use at least a two-rebit gate, and it turns out this (in particular, the CZ gate) is sufficient, as we show next. Therefore, real quantum mechanics satisfies a slightly looser axiom of tomographic *2-locality*.

To show tomographic 2-locality, we need only show how observables in \mathcal{O} can be measured using the gates H and CZ and single-rebit measurements in the computational basis (i.e. measurements of Z on a single rebit). Since any element of \mathcal{O} is made up of a tensor product of X , Z , and $Y \otimes Y$, we need only show how to measure these using the components given. The circuits in Fig. 8-3 demonstrate this. \square

Because Theorem 132 allows us to learn an $(n+1)$ -rebit state $|\phi\rangle$ up to a \pm sign, we can also determine $|\psi\rangle = \mathcal{L}(|\phi\rangle)$ up to a \pm sign, including the (typically unobservable) global phase of $|\psi\rangle$. Mapped to the simulated space, the operators used for rebit tomography in the proof of Theorem 132 become \mathbb{R} -unitaries and measurements of observables Z and K (using the encoding and decoding of measurements results in Section 8.2.3).

8.3 Partial antiunitarity

In the previous section, we characterized the set of operators that a bottom-up rebit simulation can simulate as the \mathbb{R} -unitary operators. In this section, we pay special attention to a subset of the \mathbb{R} -unitaries that we call *partial antiunitary*. These operators are mathematically an interpolation between unitary and antiunitary operators, which are each special cases. Partial antiunitaries, as we show in the Section 8.5, are also sufficient to densely generate the entire group of \mathbb{R} -unitaries. We find it convenient to start by defining the partial complex conjugation operator, which is partial antiunitary itself and is central to the study of the entire set of partial antiunitaries.

8.3.1 Partial complex conjugation

Let $L \subseteq \{0, 1\}^n$ be a language. We define the *partial complex conjugation* operator with respect to L to be $K_L : \mathcal{H}_n(\mathbb{C}) \rightarrow \mathcal{H}_n(\mathbb{C})$, where

$$K_L : |\psi\rangle \mapsto \sum_{x \notin L} \langle x | \psi \rangle |x\rangle + \sum_{x \in L} \langle \psi | x \rangle |x\rangle. \quad (8.53)$$

This generalizes the notion of complex conjugation. Indeed, when $L = \{0, 1\}^n$, we get $K_L = K$. Note that the identity operator I is another special case of K_L : when $L = \emptyset$, we obtain $K_L = I$. Two other examples of K_L that will be important in this chapter are the *controlled complex conjugation* operator CK and the *controlled-controlled complex conjugation* operator CCK , which are defined as follows: The CK operator on the j th register, denoted CK_j , is the partial complex conjugation operator with $L = \{x \in \{0, 1\}^n | x_j = 1\}$, i.e.,

$$CK_j : |\psi\rangle \mapsto \sum_{x: x_j=0} \langle x | \psi \rangle |x\rangle + \sum_{x: x_j=1} \langle \psi | x \rangle |x\rangle, \quad (8.54)$$

and the CCK operator on the i th and j th registers, where $i \neq j$, denoted CCK_{ij} is the partial complex conjugation operator with $L = \{x \in \{0, 1\}^n | x_i = x_j = 1\} = \{x \in$

$\{0, 1\}^n |x_i x_j = 1\}$, i.e.,

$$CCK_{ij} : |\psi\rangle \mapsto \sum_{x: x_i x_j = 0} \langle x | \psi \rangle |x\rangle + \sum_{x: x_i x_j = 1} \langle \psi | x \rangle |x\rangle. \quad (8.55)$$

We can also express K_L in terms of orthogonal projections (see Chapter 8.8.8). By denoting

$$\mathbb{H}(L) = \text{span}\{|x\rangle : x \in L\}, \quad \mathbb{H}^\perp(L) = \text{span}\{|x\rangle : x \notin L\}, \quad (8.56)$$

and

$$\Pi_L = \text{proj}_{\mathbb{H}(L)} = \sum_{x \in L} |x\rangle\langle x|, \quad \Theta_L = \text{proj}_{\mathbb{H}^\perp(L)} = \sum_{x \notin L} |x\rangle\langle x|, \quad (8.57)$$

we may express K_L as follows:

Proposition 133.

$$K_L = \Theta_L + K\Pi_L = \Theta_L + \Pi_L K. \quad (8.58)$$

Proof. First, we note that since Π_L is a matrix with only real entries, $K\Pi_L = \Pi_L K$. Now,

$$\begin{aligned} (\Theta_L + K\Pi_L) |\psi\rangle &= \sum_{x \notin L} |x\rangle \langle x | \psi \rangle + K \sum_{x \in L} |x\rangle \langle x | \psi \rangle \\ &= \sum_{x \notin L} \langle x | \psi \rangle |x\rangle + \sum_{x \in L} \langle \psi | x \rangle |x\rangle = K_L |\psi\rangle, \end{aligned} \quad (8.59)$$

which completes the proof. \square

Its image under \mathcal{P} is given by

Theorem 134.

$$\mathcal{P}(K_L) = \Theta_L \otimes I_a + \Pi_L \otimes Z_a. \quad (8.60)$$

Proof. By using Eq.(8.25),

$$\begin{aligned} \mathcal{P}(K_L) &= \Re\Theta_L \otimes I_a + \Im\Theta_L \otimes X_a Z_a + \Re\Pi_L \otimes Z_a + \Im\Pi_L \otimes X_a \\ &= \Theta_L \otimes I_a + \Pi_L \otimes Z_a. \end{aligned} \quad (8.61)$$

□

Remark 135. An immediate consequence of Eq. (8.60) is that K_L is not just \mathbb{R} -linear but also \mathbb{R} -unitary, as the RHS of Eq. (8.60) is orthogonal.

8.3.2 Partial antiunitary operators

Let \mathcal{H} be a complex vector space with inner product $\langle \cdot, \cdot \rangle$. A *unitary* operator on \mathcal{H} is a function $f : \mathcal{H} \rightarrow \mathcal{H}$ for which

$$\langle f(x), f(y) \rangle = \langle x, y \rangle, \quad \text{for all } x, y \in \mathcal{H}.$$

An *antiunitary* operator on V is a function $g : \mathcal{H} \rightarrow \mathcal{H}$ for which

$$\langle g(x), g(y) \rangle = \langle y, x \rangle, \quad \text{for all } x, y \in \mathcal{H}.$$

Note that a consequence of these definitions is that unitary operators are linear and antiunitary operators are antilinear¹¹.

We shall now generalize the above definitions of unitarity and antiunitarity. Let $\Xi \subseteq \mathcal{H}$ be a subspace of \mathcal{H} . Let $f, g : \mathcal{H} \rightarrow \mathcal{H}$ be operators on \mathcal{H} . We say that f is *unitary on Ξ* if

$$\langle f(x), f(y) \rangle = \langle x, y \rangle, \quad \text{for all } x, y \in \Xi.$$

We say that g is *antiunitary on Ξ* if

$$\langle g(x), g(y) \rangle = \langle y, x \rangle, \quad \text{for all } x, y \in \Xi.$$

¹¹Let \mathcal{H} be a complex vector space. A function $\mathcal{H} \rightarrow \mathcal{H}$ is an antilinear operator if

$$f(ax + by) = \bar{a}f(x) + \bar{b}f(y)$$

for all $x, y \in \mathcal{H}$ and $a, b \in \mathbb{C}$. To show that unitarity implies linearity, consider the expression

$$\|f(ax + by) - af(x) - bf(y)\|^2 = \langle f(ax + by) - af(x) - bf(y), f(ax + by) - af(x) - bf(y) \rangle$$

and show that it is equal to zero. By definition of a norm, this implies that $f(ax + by) = af(x) + bf(y)$. The proof that antiunitarity implies antilinearity proceeds similarly.

Hence, a unitary (antiunitary) operator is one that is unitary (antiunitary) on \mathcal{H} . We may now define a partial antiunitary operator¹² as follows:

Definition 136. Let $\Xi \subseteq \mathcal{H}$ be a subspace of a complex (finite-dimensional) vector space \mathcal{H} . An operator Γ on \mathcal{H} is a *partial antiunitary* operator with respect to Ξ if

- (i) Γ is additive, i.e. $\Gamma(\psi + \phi) = \Gamma(\psi) + \Gamma(\phi)$ for all $\psi, \phi \in \mathcal{H}$.
- (ii) Γ is unitary on Ξ^\perp , i.e. $\langle \Gamma(\psi), \Gamma(\phi) \rangle = \langle \psi, \phi \rangle$ for all $\psi, \phi \in \Xi^\perp$.
- (iii) Γ is antiunitary on Ξ , i.e. $\langle \Gamma(\psi), \Gamma(\phi) \rangle = \langle \phi, \psi \rangle$ for all $\psi, \phi \in \Xi$.
- (iv) $\langle \Gamma(\psi), \Gamma(\phi) \rangle = 0$ for all $\psi \in \Xi$ and $\phi \in \Xi^\perp$.

The partial antiunitary operators include unitary and antiunitary operators as special cases: a unitary operator on \mathcal{H} is a partial antiunitary operator with respect to $\{0\}$, and an antiunitary operator on \mathcal{H} is a partial antiunitary operator with respect to \mathcal{H} .

The above definition of partial antiunitarity could also be phrased in terms of orthogonal projections:

Proposition 137. An operator Γ on \mathcal{H} is a partial antiunitary operator with respect to a subspace $\Xi \subseteq \mathcal{H}$ if and only if for all $\psi, \phi \in \mathcal{H}$,

- (i') $\Gamma(\psi + \phi) = \Gamma(\psi) + \Gamma(\phi)$.
- (ii') $\langle \Gamma\Theta\psi, \Gamma\Theta\phi \rangle = \langle \Theta\psi, \Theta\phi \rangle$.
- (iii') $\langle \Gamma\Pi\psi, \Gamma\Pi\phi \rangle = \langle \Pi\phi, \Pi\psi \rangle$.
- (iv') $\langle \Gamma\Pi\psi, \Gamma\Theta\phi \rangle = 0$.

where $\Pi = \text{proj}_\Xi$ and $\Theta = \text{proj}_{\Xi^\perp}$.

¹²The term *partial antiunitary* appears on Page 2 of [170], where it was used as an example of an operator that can be simulated by the rebit encoding. However, an exact definition was not given in [170]. Also, [170] does not address whether the rebit encoding allows for the simulation of more than just products of partial antiunitaries. In this chapter, we give a precise definition of partial antiunitarity and address the above question.

Proof. Equivalence holds since $\psi = \Theta\psi$ and $\phi = \Theta\phi$ for all $\psi, \phi \in \Xi^\perp$, and $\psi = \Pi\psi$ and $\phi = \Pi\phi$ for all $\psi, \phi \in \Xi$. \square

We will now give a third characterization of partial antiunitary operators.

Theorem 138. An operator Γ on \mathcal{H} is a partial antiunitary operator with respect to a subspace $\Xi \subseteq \mathcal{H}$ if and only if for all $\psi, \phi \in \mathcal{H}$,

$$\langle \Gamma\psi, \Gamma\phi \rangle = \langle \Theta\psi, \Theta\phi \rangle + \langle \Pi\phi, \Pi\psi \rangle, \quad (8.62)$$

where $\Pi = \text{proj}_\Xi$ and $\Theta = \text{proj}_{\Xi^\perp}$.

Proof.

(\implies) Assume that Γ is a partial antiunitary with respect to Ξ . Let $\psi, \phi \in \mathcal{H}$. Then we could write $\psi = \Pi\psi + \Theta\psi$ and $\phi = \Pi\phi + \Theta\phi$. This implies that

$$\begin{aligned} \langle \Gamma\psi, \Gamma\phi \rangle &= \langle \Gamma(\Pi\psi + \Theta\psi), \Gamma(\Pi\phi + \Theta\phi) \rangle \\ &= \langle \Gamma\Pi\psi + \Gamma\Theta\psi, \Gamma\Pi\phi + \Gamma\Theta\phi \rangle, \quad \text{by additivity} \\ &= \langle \Gamma\Pi\psi, \Gamma\Pi\phi \rangle + \langle \Gamma\Pi\psi, \Gamma\Theta\phi \rangle + \langle \Gamma\Theta\psi, \Gamma\Pi\phi \rangle + \langle \Gamma\Theta\psi, \Gamma\Theta\phi \rangle \\ &= \langle \Pi\phi, \Pi\psi \rangle + 0 + 0 + \langle \Theta\psi, \Theta\phi \rangle, \quad \text{by Proposition 137 (ii'), (iii'), (iv')} \\ &= \langle \Theta\psi, \Theta\phi \rangle + \langle \Pi\phi, \Pi\psi \rangle. \end{aligned}$$

(\impliedby) Assume that Eq. (8.62) holds.

(i') We first show that Eq. (8.62) implies that Γ is additive. Consider the expression

$$\begin{aligned} & \|\Gamma(\psi + \phi) - \Gamma(\psi) - \Gamma(\phi)\|^2 \\ &= \langle \Gamma(\psi + \phi) - \Gamma(\psi) - \Gamma(\phi), \Gamma(\psi + \phi) - \Gamma(\psi) - \Gamma(\phi) \rangle \\ &= \langle \Gamma(\psi + \phi), \Gamma(\psi + \phi) \rangle + \langle \Gamma(\psi), \Gamma(\psi) \rangle + \langle \Gamma(\phi), \Gamma(\phi) \rangle \\ & \quad + [-\langle \Gamma(\psi + \phi), \Gamma(\psi) \rangle - \langle \Gamma(\psi + \phi), \Gamma(\phi) \rangle + \langle \Gamma(\psi), \Gamma(\phi) \rangle + c.c.] \\ &= \langle \Theta(\psi + \phi), \Theta(\psi + \phi) \rangle + \langle \Theta(\psi), \Theta(\psi) \rangle + \langle \Theta(\phi), \Theta(\phi) \rangle \end{aligned}$$

$$\begin{aligned}
& +[-\langle \Theta(\psi + \phi), \Theta(\psi) \rangle - \langle \Theta(\psi + \phi), \Theta(\phi) \rangle + \langle \Theta(\psi), \Theta(\phi) \rangle + c.c.] \\
& +(\Theta \leftrightarrow \Pi) \\
= & \langle \Theta(\psi + \phi) - \Theta(\psi) - \Theta(\phi), \Theta(\psi + \phi) - \Theta(\psi) - \Theta(\phi) \rangle + (\Theta \leftrightarrow \Pi) \\
= & 0,
\end{aligned}$$

where *c.c.* stands for complex conjugate, and where the last line follows because both Π and Θ are linear. By the properties of a norm, $\Gamma(\psi + \phi) = \Gamma(\psi) + \Gamma(\phi)$.

(ii')

$$\langle \Gamma\Theta\psi, \Gamma\Theta\phi \rangle = \langle \Theta^2\psi, \Theta^2\phi \rangle + \langle \Pi\Theta\phi, \Pi\Theta\psi \rangle = \langle \Theta\psi, \Theta\phi \rangle.$$

(iii')

$$\langle \Gamma\Pi\psi, \Gamma\Pi\phi \rangle = \langle \Theta\Pi\psi, \Theta\Pi\phi \rangle + \langle \Pi^2\phi, \Pi^2\psi \rangle = \langle \Pi\phi, \Pi\psi \rangle.$$

(iv')

$$\langle \Gamma\Pi\psi, \Gamma\Theta\phi \rangle = \langle \Theta\Pi\psi, \Theta^2\phi \rangle + \langle \Pi\Theta\phi, \Pi^2\psi \rangle = 0.$$

Note that we used the facts that $\Pi^2 = \Pi$, $\Theta^2 = \Theta$ and $\Theta\Pi = \Pi\Theta = 0$.

□

Note that we could have defined a notion of *partial unitarity* analogously. It would then follow that an operator Γ is a partial unitary with respect to Ξ if and only if Γ is a partial antiunitary with respect to Ξ^\perp , since $(\Xi^\perp)^\perp = \Xi$ for finite-dimensional vector spaces. Since we could easily relate the two notions, we will (somewhat arbitrarily) choose to phrase all subsequent results in terms of partial antiunitaries.

We now give an example of a partial antiunitary operator that is neither unitary nor antiunitary in general:

Proposition 139. K_L is a partial antiunitary operator with respect to $\mathbb{H}(L)$.

Proof. We shall make use of Theorem 138. Then,

$$\langle K_L\psi, K_L\phi \rangle = \langle (\Theta_L + K\Pi_L)\psi, (\Theta_L + K\Pi_L)\phi \rangle, \quad \text{by Eq. (8.58)}$$

$$= \langle \Theta_L \psi, \Theta_L \phi \rangle + \langle K \Pi_L \psi, \Theta_L \phi \rangle + \langle \Theta_L \psi, K \Pi_L \phi \rangle + \langle K \Pi_L \psi, K \Pi_L \phi \rangle.$$

But $\langle K \Pi_L \psi, \Theta_L \phi \rangle = \langle \Pi_L K \psi, \Theta_L \phi \rangle = \langle \Pi_L^\dagger \Pi_L K \psi, \Theta_L \phi \rangle = \langle \Pi_L K \psi, \Pi_L \Theta_L \phi \rangle = 0$, where we used the identities $\Pi_L K = K \Pi_L$, $\Pi_L = \Pi_L^\dagger \Pi_L$ and $\Pi_L \Theta_L = 0$, as well as the definition of the adjoint. Likewise, $\langle \Theta_L \psi, K \Pi_L \phi \rangle = 0$. Finally, $\langle K \Pi_L \psi, K \Pi_L \phi \rangle = K(\langle \Pi_L \psi, \Pi_L \phi \rangle) = \langle \Pi_L \phi, \Pi_L \psi \rangle$. Hence, $\langle K_L \psi, K_L \phi \rangle = \langle \Theta_L \psi, \Theta_L \phi \rangle + \langle \Pi_L \phi, \Pi_L \psi \rangle$, which means that K_L is a partial antiunitary operator with respect to $\mathbb{H}(L)$. \square

Next, we show that multiplying a partial antiunitary with respect to $\mathbb{H}(L)$ with K_L produces a unitary operator.

Proposition 140. Let Γ be a partial antiunitary (on \mathcal{H}) with respect to $\mathbb{H}(L)$. Then ΓK_L is a unitary operator on \mathcal{H} .

Proof. Assume that Γ is a partial antiunitary with respect to $\mathbb{H}(L)$. Then for all $\psi, \phi \in \mathcal{H}$,

$$\begin{aligned} \langle \Gamma K_L \psi, \Gamma K_L \phi \rangle &= \langle \Theta_L K_L \psi, \Theta_L K_L \phi \rangle + \langle \Pi_L K_L \psi, \Pi_L K_L \phi \rangle \\ &= \langle \Theta_L (\Theta_L + \Pi_L K) \psi, \Theta_L (\Theta_L + \Pi_L K) \phi \rangle \\ &\quad + \langle \Pi_L (\Theta_L + \Pi_L K) \psi, \Pi_L (\Theta_L + \Pi_L K) \phi \rangle \\ &= \langle \Theta_L \psi, \Theta_L \phi \rangle + \langle \Pi_L K \psi, \Pi_L K \phi \rangle. \end{aligned}$$

But the latter term in the sum is equal to $\langle \Pi_L K \psi, \Pi_L K \phi \rangle = \langle K \Pi_L \psi, K \Pi_L \phi \rangle = K(\langle \Pi_L \psi, \Pi_L \phi \rangle) = \langle \Pi_L \psi, \Pi_L \phi \rangle$. Hence,

$$\begin{aligned} \langle \Gamma K_L \psi, \Gamma K_L \phi \rangle &= \langle \Theta_L \psi, \Theta_L \phi \rangle + \langle \Pi_L \psi, \Pi_L \phi \rangle \\ &= \langle \Theta_L \psi, \Theta_L \phi \rangle + \langle \Theta_L \psi, \Pi_L \phi \rangle + \langle \Pi_L \psi, \Theta_L \phi \rangle + \langle \Pi_L \psi, \Pi_L \phi \rangle \\ &= \langle \Theta_L \psi + \Pi_L \psi, \Theta_L \phi + \Pi_L \phi \rangle = \langle \psi, \phi \rangle. \end{aligned}$$

Since this holds for all $\phi, \psi \in \mathcal{H}$, ΓK_L is unitary. \square

While the product¹³ of two partial antiunitary operators may not be partial an-

¹³We define the product of two partial antiunitary operators A and B to be their composition,

tiunitary (we show this later in Theorem 148), the product of a partial antiunitary operator with either a unitary or antiunitary operator is always partial antiunitary, as the following theorem shows.

Theorem 141. Let $\Xi \subseteq \mathcal{H}$ be a subspace of a complex vector space \mathcal{H} . Let U be a unitary operator on \mathcal{H} and V be an antiunitary operator on \mathcal{H} . Then the following statements are equivalent.¹⁴

- (I) Γ is a partial antiunitary with respect to Ξ .
- (II) ΓU is a partial antiunitary with respect to $U^\dagger(\Xi)$.
- (III) $U\Gamma$ is a partial antiunitary with respect to Ξ .
- (IV) ΓV is a partial antiunitary with respect to $(V^\dagger(\Xi))^\perp$.
- (V) $V\Gamma$ is a partial antiunitary with respect to Ξ^\perp .

Proof. We first show that if U is either unitary or antiunitary, then

$$\text{proj}_\Xi U = U \text{proj}_{U^\dagger(\Xi)}. \quad (8.63)$$

To see this, let $\{|a_i\rangle\}_{i=1}^s$ be a basis for Ξ . Then $\{U^\dagger|a_i\rangle\}_{i=1}^s$ is a basis for $U^\dagger(\Xi)$. Hence, by Proposition 188 in Chapter 8.8.8,

$$\begin{aligned} \text{proj}_\Xi U &= \sum_{i=1}^s |a_i\rangle\langle a_i|U \\ &= U \sum_{i=1}^s U^\dagger|a_i\rangle\langle a_i|U \\ &= U \text{proj}_{U^\dagger(\Xi)}. \end{aligned}$$

Now, (I) holds if and only if for all $\psi, \phi \in \mathcal{H}$,

$$\langle \Gamma\psi, \Gamma\phi \rangle = \langle \text{proj}_{\Xi^\perp}\psi, \text{proj}_{\Xi^\perp}\phi \rangle + \langle \text{proj}_\Xi\phi, \text{proj}_\Xi\psi \rangle. \quad (8.64)$$

i.e. $AB := A \circ B$.

¹⁴Here, we use $U^\dagger(\Xi)$ to denote the set $\{U^\dagger\phi \mid \phi \in \Xi\}$.

Since

$$\begin{aligned}
(*) \iff \langle \Gamma U\psi, \Gamma U\phi \rangle &= \langle \text{proj}_{\Xi^\perp} U\psi, \text{proj}_{\Xi^\perp} U\phi \rangle + \langle \text{proj}_{\Xi} U\phi, \text{proj}_{\Xi} U\psi \rangle \\
&= \langle U \text{proj}_{U^\dagger(\Xi^\perp)} \psi, U \text{proj}_{U^\dagger(\Xi^\perp)} \phi \rangle \\
&\quad + \langle U \text{proj}_{U^\dagger(\Xi)} \phi, U \text{proj}_{U^\dagger(\Xi)} \psi \rangle \\
&= \langle \text{proj}_{U^\dagger(\Xi)^\perp} \psi, \text{proj}_{U^\dagger(\Xi)^\perp} \phi \rangle + \langle \text{proj}_{U^\dagger(\Xi)} \phi, \text{proj}_{U^\dagger(\Xi)} \psi \rangle.
\end{aligned}$$

Therefore, (I) is equivalent to (II).

Since $\langle U\Gamma\psi, U\Gamma\phi \rangle = \langle V\Gamma\phi, V\Gamma\psi \rangle = \langle \Gamma\psi, \Gamma\phi \rangle$, (I) is equivalent to (III) and (V), by using Eq. (8.64).

Finally, since

$$\begin{aligned}
\text{Eq. (8.64)} \iff \langle \Gamma V\psi, \Gamma V\phi \rangle &= \langle \text{proj}_{\Xi^\perp} V\psi, \text{proj}_{\Xi^\perp} V\phi \rangle + \langle \text{proj}_{\Xi} V\phi, \text{proj}_{\Xi} V\psi \rangle \\
&= \langle V \text{proj}_{V^\dagger(\Xi^\perp)} \psi, V \text{proj}_{V^\dagger(\Xi^\perp)} \phi \rangle \\
&\quad + \langle V \text{proj}_{V^\dagger(\Xi)} \phi, V \text{proj}_{V^\dagger(\Xi)} \psi \rangle \\
&= \langle \text{proj}_{V^\dagger(\Xi)^\perp} \phi, \text{proj}_{V^\dagger(\Xi)^\perp} \psi \rangle \\
&\quad + \langle \text{proj}_{V^\dagger(\Xi)} \psi, \text{proj}_{V^\dagger(\Xi)} \phi \rangle,
\end{aligned}$$

(I) is equivalent to (IV). □

We now use Theorem 141 to prove the following corollary.

Corollary 142. Let Ξ be a subspace of the n -qubit Hilbert space $\mathcal{H}_n(\mathbb{C})$. If Γ is a partial antiunitary with respect to Ξ , then there exist a language $L \subseteq \{0, 1\}^n$, with $|L| = \dim(\Xi)$, and a unitary operator U mapping $\mathbb{H}(L)$ to Ξ , such that ΓU is a partial antiunitary with respect to $\mathbb{H}(L)$.

Proof. We are given that $\mathbb{H}(L) = U^\dagger(\Xi)$. By (I) \iff (II) of Theorem 141, ΓU is a partial antiunitary with respect to $U^\dagger(\Xi) = \mathbb{H}(L)$ since Γ is a partial antiunitary with respect to Ξ . □

We are now ready to combine the result above from Corollary 142 with Proposition 140 to show that any partial antiunitary operator can be written as a product

of K_L with unitary operators.

Theorem 143. Let Ξ be a subspace of the n -qubit Hilbert space $\mathcal{H}_n(\mathbb{C})$. If Γ is a partial antiunitary operator with respect to Ξ , then there exist a language $L \subseteq \{0, 1\}^n$, with $|L| = \dim \Xi$, and unitary operators U and V , with V mapping Ξ to $\mathbb{H}(L)$, such that

$$\Gamma = UK_LV. \quad (8.65)$$

Proof. By Corollary 142, Γ being a partial antiunitary with respect to Ξ implies that ΓV^\dagger is a partial antiunitary with respect to $V(\Xi) = \mathbb{H}(L)$. By Proposition 140, $\Gamma V^\dagger K_L = U$ for some unitary U . Since K_L is its own inverse, this implies that $\Gamma = UK_LV$, which completes the proof of the theorem. \square

Theorem 143 tells us that if we wanted to simulate any arbitrary partial antiunitary operator, it would suffice to just use products of unitary operators and partial complex conjugation. Next, we show in the next two theorems that partial antiunitary operators are a special case of \mathbb{R} -linear operators, and that not every \mathbb{R} -linear operator is partial antiunitary. In fact, the partial antiunitaries, unlike the \mathbb{R} -unitaries, are not a subgroup of the \mathbb{R} -linear operators.

Theorem 144. If Γ is a partial antiunitary operator, then it is \mathbb{R} -unitary.

Proof. By Theorem 143, we could write $\Gamma = UK_LV$, where $L \subseteq \{0, 1\}^n$ and U and V are unitaries. By Remark 135, K_L is \mathbb{R} -unitary. Since the \mathbb{R} -unitaries are closed under multiplication, $\Gamma = UK_LV$ is also \mathbb{R} -unitary. \square

Our characterizations of partial antiunitary operators thus far (Definition 136, Proposition 137, and Theorem 138) all involve universal quantifiers and do not provide us with an algorithm to decide if a given \mathbb{R} -linear operator is partial antiunitary. In the following theorem – our fourth characterization of partial antiunitary operators – we give necessary and sufficient conditions, which can be checked efficiently, that A and B must satisfy, in order for $\Gamma = A + BK$ to be partial antiunitary.

Theorem 145. An operator Γ on \mathcal{H} is a partial antiunitary operator with respect to a subspace $\Xi \subseteq \mathcal{H}$ if and only if $\Gamma = A + BK$ for complex operators A and B satisfying

$$A^\dagger B = 0 \tag{8.66}$$

$$A^\dagger A = \Theta \tag{8.67}$$

$$B^\dagger B = \bar{\Pi} \tag{8.68}$$

where $\Theta = \text{proj}_{\Xi^\perp}$ and $\Pi = \text{proj}_\Xi$.

Proof. We first prove the forward direction. Let Γ be a partial antiunitary operator with respect to Ξ . Then by Theorem 143, there exist unitaries U and V , and a language L satisfying $V(\Xi) = \mathbb{H}(L)$ such that

$$\begin{aligned} \Gamma &= UK_L V \\ &= U(\Theta_L + \Pi_L K)V \\ &= (U\Theta_L V) + (U\Pi_L \bar{V})K \\ &\equiv A + BK, \end{aligned} \tag{8.69}$$

where $A = U\Theta_L V$ and $B = U\Pi_L \bar{V}$.

Next, we check that the conditions Eq.(8.66)–(8.68) are satisfied.

$$\begin{aligned} A^\dagger B &= (U\Theta_L V)^\dagger (U\Pi_L \bar{V}) \\ &= V^\dagger \Theta_L \Pi_L \bar{V} \\ &= 0. \end{aligned} \tag{8.70}$$

$$\begin{aligned} A^\dagger A &= (U\Theta_L V)^\dagger (U\Theta_L V) \\ &= V^\dagger \Theta_L V \\ &= \Theta. \end{aligned} \tag{8.71}$$

$$\begin{aligned}
B^\dagger B &= (U\Pi_L\bar{V})^\dagger(U\Pi_L\bar{V}) \\
&= V^T\Pi_L\bar{V} \\
&= V^\dagger\bar{\Pi}_L V \\
&= \bar{\Pi}.
\end{aligned} \tag{8.72}$$

where we used the following identities that follow from Eq.(8.63):

$$\Theta_L V = \text{proj}_{\mathbb{H}^\perp(L)} V = V \text{proj}_{V^\dagger(\mathbb{H}^\perp(L))} = V \text{proj}_{\xi^\perp} = V\Theta. \tag{8.73}$$

and

$$\Pi_L V = \text{proj}_{\mathbb{H}(L)} V = V \text{proj}_{V^\dagger(\mathbb{H}(L))} = V \text{proj}_\xi = V\Pi. \tag{8.74}$$

We now prove the backward direction. Let $\Gamma = A + BK$, with A and B satisfying Eq.(8.66)–(8.68). Let ψ and ϕ be arbitrary. Then,

$$\begin{aligned}
\langle \Gamma\psi, \Gamma\phi \rangle &= \langle (A + BK)\psi, (A + BK)\phi \rangle \\
&= \langle A\psi, A\phi \rangle + \langle A\psi, B\bar{\phi} \rangle + \langle B\bar{\psi}, A\phi \rangle + \langle B\bar{\psi}, B\bar{\phi} \rangle \\
&= \langle \psi, A^\dagger A\phi \rangle + \langle \psi, A^\dagger B\bar{\phi} \rangle + \langle \bar{\psi}, B^\dagger A\phi \rangle + \langle \psi, B^\dagger B\bar{\phi} \rangle \\
&= \langle \psi, \Theta\phi \rangle + 0 + 0 + \langle \bar{\psi}, \bar{\Pi}\bar{\phi} \rangle \\
&= \langle \Theta\psi, \Theta\phi \rangle + \langle \Pi\phi, \Pi\psi \rangle.
\end{aligned} \tag{8.75}$$

□

Theorem 145 gives us an efficient algorithm (in terms of the dimensions of the matrices A and B) for deciding if a given \mathbb{R} -linear operator $A + BK$ is partial antiunitary. We simply need to compute the matrices $s_1 = A^\dagger B$, $s_2 = A^\dagger A$ and $s_3 = B^\dagger B$, and check that $s_1 = 0$, $s_2 = s_2^\dagger = s_2^\dagger$ (which is the definition of an orthogonal projection) and $s_3 = \overline{1 - s_2}$ (since $\Theta + \Pi = I$). The theorem also gives us the following corollary.

Corollary 146. Let $\Gamma = A + BK$ be \mathbb{R} -unitary. Then Γ is partial antiunitary with respect to a subspace Ξ if and only if

$$A^\dagger A = \Theta \quad \text{and} \quad A^\dagger B = 0, \quad (8.76)$$

where $\Theta = \text{proj}_{\Xi^\perp}$.

Proof. The forward direction follows immediately from Eq. (8.66) and Eq. (8.67). Next, we prove the backward direction. Assume that Eq. (8.76) holds. Then, Eq. (8.66) and Eq. (8.67) are immediately satisfied. Since Γ is \mathbb{R} -unitary,

$$A^\dagger A + B^T \bar{B} = I. \quad (8.77)$$

Hence,

$$B^\dagger B = \overline{(B^T \bar{B})} = \overline{I - A^\dagger A} = \overline{I - \Theta} = \bar{\Pi}. \quad (8.78)$$

□

We conclude this section by proving some closure properties of the set of partial antiunitaries. First, we show that the partial antiunitaries are closed under inverses (i.e. under \dagger).

Theorem 147. If Γ is partial antiunitary, then Γ^\dagger is also partial antiunitary.

Proof. If Γ is partial antiunitary, then by Theorem 143, there exist a language L and unitaries U and V such that $\Gamma = UK_L V$. Hence, $\Gamma^\dagger = V^\dagger K_L U^\dagger$. Since partial antiunitaries are closed under multiplication by unitaries (by Theorem 141), Γ^\dagger is partial antiunitary.¹⁵ □

Next, we show that the partial antiunitaries are not closed under multiplication. As a consequence, they are not a subgroup of the \mathbb{R} -unitaries, i.e. there exist \mathbb{R} -unitary operators that are not partial antiunitary.

¹⁵If Γ is partial antiunitary with respect to Ξ and Γ^\dagger is partial antiunitary with respect to Σ , one may wonder about the relation between Ξ and Σ . Since, by Theorem 145, $\text{proj}_{\Xi^\perp} = A^\dagger A$ and $\text{proj}_{\Sigma^\perp} = A A^\dagger$, it is clear that at the very least $\dim \Xi = \dim \Sigma$, and thus these spaces are related by a unitary. To actually find the unitary, perform the polar decomposition of $A = UP$ into a unitary U and positive semi-definite P . Then $\text{proj}_{\Xi^\perp} = P^2$ and $\text{proj}_{\Sigma^\perp} = U \text{proj}_{\Xi^\perp} U^\dagger$.

Theorem 148. There exist partial antiunitaries Γ and Δ such that $\Gamma\Delta$ is not partial antiunitary.

Proof. On a single-qubit, let K_1 be the partial complex conjugation operator with respect to the language $L = \{1\}$ (equivalently, $K_1 = CK_1$ is the controlled complex conjugation operator on the first (and only) register) and consider single-qubit operators $\Gamma = K_1H$ and $\Delta = SHK_1$. These are both partial antiunitary by Theorem 141. Their product, however, is

$$\begin{aligned}\Gamma\Delta &= K_1HSHK_1 = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 0 \\ 0 & k \end{pmatrix} \begin{pmatrix} e^{i\pi/4} & e^{-i\pi/4} \\ e^{-i\pi/4} & e^{i\pi/4} \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & k \end{pmatrix} \\ &= \frac{1}{\sqrt{2}} e^{i\pi/4} \begin{pmatrix} 1 & -ik \\ k & -i \end{pmatrix},\end{aligned}\tag{8.79}$$

where k is the complex conjugation operator¹⁶.

Hence, $\Gamma\Delta = A + BK$, where

$$A = \frac{1}{\sqrt{2}} e^{i\pi/4} \begin{pmatrix} 1 & 0 \\ 0 & -i \end{pmatrix}, \quad B = \frac{1}{\sqrt{2}} e^{i\pi/4} \begin{pmatrix} 0 & -i \\ 1 & 0 \end{pmatrix},\tag{8.80}$$

from which it follows that

$$A^\dagger A = \frac{1}{2}I, \quad A^\dagger B = \frac{1}{2}Y \neq 0,\tag{8.81}$$

which does not satisfy Eq. (8.76). Hence, $\Gamma\Delta$ is not partial antiunitary.¹⁷

□

¹⁶In this proof, we represented \mathbb{R} -linear operators by matrices. Note that the elements of these matrices are themselves \mathbb{R} -linear operators, and do not belong to a field. We discuss matrix representations of \mathbb{R} -linear operators further in Chapter 8.8.9.

¹⁷Alternatively, we can show more generally that any $\Omega = \begin{pmatrix} a & bk \\ ck & d \end{pmatrix}$ is not partial antiunitary when either a and b are both nonzero or c and d are both nonzero. This is because setting $\Omega = A + BK$ and calculating $A^\dagger B = \begin{pmatrix} \bar{a} & 0 \\ 0 & \bar{d} \end{pmatrix} \begin{pmatrix} 0 & b \\ c & 0 \end{pmatrix} = \begin{pmatrix} 0 & \bar{a}b \\ \bar{d}c & 0 \end{pmatrix} \neq \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$ in contradiction with Theorem 145 and in particular Eq. (8.66).

8.4 Simulating partial antiunitary operators

In the previous section, we studied various properties of the set of partial antiunitary operators. In this section, we consider examples of these operators and study their rebit simulation. We do this in two parts. First, we consider partial antiunitaries which are also unitary. This is in line with the top-down approach taken by the initial use of rebit simulation [25], where it is shown that real-amplitude quantum computers are just as powerful as those with complex amplitudes. Second, we consider partial antiunitaries which are neither unitary nor linear – operators for which a bottom-up simulation is necessary.

8.4.1 Rebit simulation of unitaries: top-down perspective

We study the rebit encoding of unitary operators from a circuit point of view. The encoding function $\mathcal{P} : U_n \rightarrow T_{n+1}$ is a group homomorphism (see Proposition 116), and hence $\mathcal{P}(UV) = \mathcal{P}(U)\mathcal{P}(V)$ for all unitary operators $U, V \in U_n$. This means that if U is a unitary operator that is implemented by a circuit C consisting of unitary gates G_1, \dots, G_k , then $\mathcal{P}(U)$ is an orthogonal operator that can be implemented by the circuit $\mathcal{P}(C)$, where $\mathcal{P}(C)$ is the circuit formed from C by replacing each gate G_i in C with $\mathcal{P}(G_i)$.

We make use of the following conventions. Let G be a gate contained in an n -qubit circuit. We write G_{i_1, \dots, i_s} to mean that G acts on registers $i_1, \dots, i_s \in \{1, \dots, n\}$. The encoded gate $\mathcal{P}(G)$ is now a gate in an $(n+1)$ -rebit circuit. We will continue to use the labels $1, \dots, n$ to denote the first n registers, and will use the subscript a to denote the last (ancilla) register.

We will now give examples of various common gates G_i and their rebit encodings $\mathcal{P}(G_i)$.

Proposition 149. Under the encoding $\mathcal{P}(\cdot)$, the gates below transform as follows

(1) $X_i \mapsto X_i$

(2) $Y_i \mapsto X_i \cdot X_a \cdot Z_i \cdot Z_a = -Y_i \cdot Y_a$

$$(3) Z_i \mapsto Z_i$$

$$(4) H_i \mapsto H_i$$

$$(5) S_i \mapsto CX_{ia} \cdot CZ_{ia} = H_a \cdot CZ_{ia} \cdot H_a \cdot CZ_{ia}$$

$$(6) T_i \mapsto CH_{ia} \cdot CZ_{ia}$$

$$(7) CX_{ij} \mapsto CX_{ij}$$

$$(8) CZ_{ij} \mapsto CZ_{ij}$$

$$(9) CS_{ij} \mapsto CCX_{ija} \cdot CCZ_{ija} = H_a \cdot CCZ_{ija} \cdot H_a \cdot CCZ_{ija}$$

$$(10) CCZ_{ijk} \mapsto CCZ_{ijk}$$

$$(11) Y(\theta)_i := \cos(\theta/2)I - i \sin(\theta/2)Y_i \mapsto Y(\theta)_i$$

$$(12) e^{i\theta/2}Z(\theta)_i := e^{i\theta/2}(\cos(\theta/2)I - i \sin(\theta/2)Z_i) \mapsto CY(2\theta)_{ia}$$

Proof. Gates G_i in (1), (3), (4), (7), (8), (10), and (11) have only real entries, and hence they map to themselves under \mathcal{P} . For the other cases, we make use of Eq. (8.29).

- For (2), $Y_i = iX_iZ_i \mapsto (XZ)_i \otimes (XZ)_a = X_iX_aZ_iZ_a$.
- For (5), $S_i = |0\rangle\langle 0| + i|1\rangle\langle 1| \mapsto |0\rangle\langle 0|_i \otimes I_a + |1\rangle\langle 1|_i \otimes X_aZ_a = C(XZ)_{ia} = CX_{ia}CZ_{ia} = H_aCZ_{ia}H_aCZ_{ia}$.
- For (6), $T = |0\rangle\langle 0| + e^{i\pi/4}|1\rangle\langle 1| = |0\rangle\langle 0| + \frac{1}{\sqrt{2}}|1\rangle\langle 1| + \frac{1}{\sqrt{2}}i|1\rangle\langle 1| \mapsto (|0\rangle\langle 0| + \frac{1}{\sqrt{2}}|1\rangle\langle 1|)_i \otimes I_a + \frac{1}{\sqrt{2}}|1\rangle\langle 1|_i \otimes X_aZ_a \mapsto |0\rangle\langle 0|_i \otimes I_a + \frac{1}{\sqrt{2}}|1\rangle\langle 1|_i \otimes (I + XZ)_a = C(\frac{1}{\sqrt{2}}(I + XZ))_{ia} = C(HZ)_{ia} = CH_{ia}CZ_{ia}$.
- For (9), $CS_{ij} = |0\rangle\langle 0|_i \otimes I_j + |1\rangle\langle 1|_i \otimes S_j = (|00\rangle\langle 00| + |01\rangle\langle 01| + |10\rangle\langle 10| + i|11\rangle\langle 11|)_{ij} \mapsto (|00\rangle\langle 00| + |01\rangle\langle 01| + |10\rangle\langle 10|)_{ij} \otimes I_a + |11\rangle\langle 11|_{ij} \otimes X_aZ_a = CC(XZ)_{ija} = CCX_{ija}CCZ_{ija} = H_aCCZ_{ija}H_aCCZ_{ija}$.
- For (12), $e^{i\theta/2}Z(\theta)_i = |0\rangle\langle 0|_i + e^{i\theta}|1\rangle\langle 1|_i \mapsto (|0\rangle\langle 0| + \cos(\theta)|1\rangle\langle 1|)_i \otimes I_a - i \sin(\theta)|1\rangle\langle 1|_i \otimes Y_a = |0\rangle\langle 0|_i \otimes I_a + |1\rangle\langle 1|_i \otimes (\cos(\theta)I_a - i \sin(\theta)Y_a) = CY(2\theta)_{ia}$.

□

In quantum mechanics, the global phase of a quantum state does not play any physical role. Therefore, two unitary operators U_1 and U_2 that differ by a global phase (i.e. $U_1 = e^{i\theta}U_2$ for some $\theta \in \mathbb{R}$) are physically equivalent. How does this equivalence manifest in the rebit encoding? To start answering this question, we first define $G(\theta) = e^{i\theta}I$ to be the global phase operator with angle θ . Since $G(\theta)$ is unitary, we could find its image under the rebit encoding $\mathcal{P}(\cdot)$. As we shall now see, the image of $G(\theta)$ under \mathcal{P} is the rotation matrix $R(\theta) = \begin{pmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{pmatrix}$ acting on the ancilla register.

Proposition 150. $\mathcal{P} : G(\theta) \mapsto R(\theta)_a$.

Proof. We compute the action of \mathcal{P} on $G(\theta)$.

$$\begin{aligned} \mathcal{P} : G(\theta) = (\cos \theta + i \sin \theta)I &\mapsto (\cos \theta)I \otimes \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}_a + (\sin \theta)I \otimes \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}_a \\ &= I \otimes \begin{pmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{pmatrix}_a = R(\theta)_a. \end{aligned}$$

□

Since the encoding of the global phase operator is restricted to the ancilla, it follows that measurements on the other n rebits will not give information about the global phase. By Proposition 130 these are exactly the measurements of rebits in P that correspond to qubit measurements in the simulated space L . On the other hand, measurements on the ancilla can yield information about the global phase of the simulated state, as we described in Section 8.2.4.

8.4.2 Rebit simulation of non-unitaries: bottom-up perspective

We now give examples of the rebit simulation of various partial antiunitaries that are not unitary (and not linear). As noted earlier, these are therefore operators for which a bottom-up simulation is necessary. We then discuss the partial complex conjugation operator K_L for an arbitrary language $L \subseteq \{0,1\}^n$ and make a connection between the complexity of deciding L and the complexity of simulating K_L .

As before, we use the indices i, j, k to refer to any of the first n registers, and a to refer to the ancilla register. Given that we are allowed to perform any orthogonal gate from T_{n+1} on the $n+1$ rebits in the simulator, to find non-unitary simulations we can just try various orthogonal gates and see if any decode to non-unitary operators on the n qubit system being simulated. But notice, if an orthogonal gate $T \in T_n$ does not act on the ancilla register, its image under \mathcal{L} is itself, since $\mathcal{L}(T \otimes I_a) = T$. Hence, any gate not in $\mathcal{P}(U_n)$ must necessarily act nontrivially on the a th register. For instance,

Proposition 151. Under the rebit decoding $\mathcal{L}(\cdot)$, the gates below transform as follows

$$(I) H_a \mapsto G\left(\frac{\pi}{4}\right)K$$

$$(II) Z_a \mapsto K$$

$$(III) CZ_{ia} \mapsto CK_i$$

$$(IV) CCZ_{ija} \mapsto CCK_{ij}$$

$$(V) C^h Z_{c_1, c_2, \dots, c_h, a} \mapsto C^h K_{c_1, c_2, \dots, c_h}.$$

Proof. Note that (II) was proved in Eq.(8.48). So next, we prove (I): First, note that

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} = \begin{pmatrix} \cos \frac{\pi}{4} & -\sin \frac{\pi}{4} \\ \sin \frac{\pi}{4} & \cos \frac{\pi}{4} \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} = R\left(\frac{\pi}{4}\right) Z,$$

which implies that

$$\mathcal{L}(H_a) = \mathcal{L}\left(R\left(\frac{\pi}{4}\right) Z_a\right) = \mathcal{L}\left(R\left(\frac{\pi}{4}\right)\right) \mathcal{L}(Z_a) = G\left(\frac{\pi}{4}\right) K. \quad (8.82)$$

where we used the fact that \mathcal{L} is a homomorphism for the second equality, and Eq. (150) and Eq. (8.48) for the third equality.

For (III), we make use of Eq. (8.17). By denoting $\psi_{x0} = \Re\psi_x$ and $\psi_{x1} = \Im\psi_x$, we get

$$\begin{aligned} \mathcal{L}(CZ_{ia}) \left(\sum_x \psi_x |x\rangle \right) &= \mathcal{L}\left(CZ_{ia} \sum_{xa} \psi_{xa} |xa\rangle\right) \\ &= \mathcal{L}\left(\sum_{xa} (-1)^{x_i a} \psi_{xa} |xa\rangle\right) \\ &= \mathcal{L}\left(\sum_x |x\rangle (\psi_{x0} |0\rangle + (-1)^{x_i} \psi_{x1} |1\rangle)\right) \\ &= \sum_x |x\rangle (\psi_{x0} + i(-1)^{x_i} \psi_{x1}) \\ &= \sum_{x:x_i=0} (\psi_{x0} + i\psi_{x1}) |x\rangle + \sum_{x:x_i=1} (\psi_{x0} - i\psi_{x1}) |x\rangle \\ &= \sum_{x:x_i=0} \psi_x |x\rangle + \sum_{x:x_i=1} \bar{\psi}_x |x\rangle \\ &= CK_i \left(\sum_x \psi_x |x\rangle \right). \end{aligned}$$

Hence, $\mathcal{L}(CZ_{ia}) = CK_i$. For (IV) and (V) the arguments are analogous and straightforward. \square

8.4.2.1 Simulation of the partial complex conjugation operator

Next, we consider the partial complex conjugation operator K_L defined in Eq. (8.53), where $L \subseteq \{0, 1\}^n$ is some language. We start by expressing the image of K_L under \mathcal{P} (characterized, for example, in Theorem 134) in terms of the indicator function of the language L .

Proposition 152. Let $L \subseteq \{0, 1\}^n$. Then,

$$\mathcal{P}(K_L) = \sum_{x \in \{0,1\}^n} \sum_{a \in \{0,1\}} (-1)^{aL(x)} |xa\rangle\langle xa|, \quad (8.83)$$

where $L(x)$ is the indicator function of L .

Proof. We make use of Theorem 134.

$$\begin{aligned} \mathcal{P}(K_L) &= \Theta_L \otimes I + \Pi_L \otimes Z \\ &= \sum_{x \notin L} |x\rangle\langle x| \otimes \sum_{a \in \{0,1\}} |a\rangle\langle a| + \sum_{x \in L} |x\rangle\langle x| \otimes \sum_{a \in \{0,1\}} (-1)^a |a\rangle\langle a| \\ &= \sum_{x \in \{0,1\}^n} \sum_{a \in \{0,1\}} (-1)^{aL(x)} |xa\rangle\langle xa|. \end{aligned} \quad (8.84)$$

□

We now show that if $L \subseteq \{0, 1\}^n$ is decidable by some quantum circuit C_L , then we can construct a quantum circuit that implements the operator $\mathcal{P}(K_L)$. Here, we say that C_L decides L if when given input $|x\rangle$, C_L outputs $|L(x)\rangle$. More precisely, taking into account all ancilla registers, the action of C_L may be described by

$$C_L |x\rangle_{1,\dots,n} |0\rangle_\alpha = |L(x)\rangle_1 |j(x)\rangle_{2,\dots,n,\alpha}, \quad (8.85)$$

where $j(x)$ are junk bits.

Proposition 153. Let $L \subseteq \{0, 1\}^n$. Let $C_L = (C_L)_{1,\dots,n,a}$ be a quantum circuit that acts on basis states according to Eq. (8.85). Then

$$\mathcal{P}(K_L)_{1,\dots,n,a} = \langle 0|_\alpha (C_L^\dagger)_{1,\dots,n,a} C Z_{1a} (C_L)_{1,\dots,n,a} |0\rangle_\alpha. \quad (8.86)$$

Proof. Starting with the input state $|\phi\rangle = \sum_{xa} \psi_{xa} |xa\rangle_{1,\dots,n,a}$, and appending an

ancilla register α initialized to $|0\rangle$, the system evolves as follows:

$$\begin{aligned}
\sum_{xa} \psi_{xa} |xa\rangle_{1,\dots,n,a} |0\rangle_\alpha &\xrightarrow{C_L} \sum_{xa} \psi_{xa} |L(x)\rangle_1 |j(x)\rangle_{2,\dots,n,\alpha} |a\rangle_a \\
&\xrightarrow{CZ_{1a}} \sum_{xa} \psi_{xa} (-1)^{aL(x)} |L(x)\rangle_1 |j(x)\rangle_{2,\dots,n,\alpha} |a\rangle_a \\
&\xrightarrow{C_L^\dagger} \sum_{xa} \psi_{xa} (-1)^{aL(x)} |x\rangle_{1,\dots,n} |a\rangle_a |0\rangle_\alpha.
\end{aligned}$$

Hence,

$$\langle 0|_\alpha (C_L^\dagger)_{1,\dots,n,a} CZ_{1a} (C_L)_{1,\dots,n,a} |0\rangle_\alpha \left(\sum_{xa} \psi_{xa} |xa\rangle \right) = \sum_{xa} \psi_{xa} (-1)^{aL(x)} |x\rangle_{1,\dots,n} |a\rangle_a \quad (8.87)$$

$$= \mathcal{P}(K_L) \sum_{xa} \psi_{xa} |xa\rangle, \quad (8.88)$$

where the last line follows from Eq. (8.83). Therefore, Eq. (8.86) is true since the above equality holds for all $|\phi\rangle$. \square

So far, our discussion has dealt with the case where n is fixed. We now consider the case where n is allowed to grow arbitrarily.

Corollary 154. Let $L \in \mathcal{P}$. Then $\{\mathcal{P}(K_L)_{1,\dots,n,a}\}_n$ can be implemented by a uniform family of polynomial-sized quantum circuits $\{Q_n\}_n$ that comprise only orthogonal gates.

Proof. $L \in \mathcal{P}$ means that there exists a uniform family of polynomial-sized classical circuits $\{C_n\}_n$, where each C_n comprises only reversible gates, say Toffoli gates and NOT gates, such that $C_n(x, 0) = (L(x), j(x))$. Let \tilde{C}_n be a quantum circuit formed from C_n by replacing each classical gate by its quantum counterpart so that $\tilde{C}_n |x\rangle_{1,\dots,n} |0\rangle_\alpha = |L(x)\rangle_1 |j(x)\rangle_{2,\dots,n,\alpha}$. We then construct the circuit

$$(C_L^\dagger)_{1,\dots,n,a} CZ_{1a} (C_L)_{1,\dots,n,a}.$$

By Proposition 153, the circuit $Q_n = (\tilde{C}_L^\dagger)_{1,\dots,n,a} CZ_{1a} (\tilde{C}_L)_{1,\dots,n,a}$, after discarding

ancilla rebits, implements $\mathcal{P}(K_L)$. Furthermore, since C_n is polynomial-sized, \tilde{C}_n and hence Q_n are also polynomial-sized. Finally, both the quantum Toffoli gate and the quantum NOT gate (i.e. the X gate) are orthogonal gates. Hence, the set $\{\mathcal{P}(K_L)_{1,\dots,n,a}\}_n$ has the desired properties. \square

Corollary 155. Let $L \in \mathcal{P}$. Then $\{(K_L)_{1,\dots,n,a}\}_n$ can be implemented by a uniform family of polynomial-sized \mathbb{R} -unitary quantum circuits $\{Q_n\}_n$ that comprise only CCZ , H gates and exactly one CK gate.

Proof. The circuit Q_n constructed in the proof of Corollary 154 is

$$(\tilde{C}_L^\dagger)_{1,\dots,n,a} CZ_{1a} (\tilde{C}_L)_{1,\dots,n,a},$$

and it implements the operator $\mathcal{P}(K_L)$. By applying the rebit decoding operator \mathcal{L} to this, we find that

$$K_L = \mathcal{L}(\tilde{C}_L^\dagger)_{1,\dots,n,a} \mathcal{L}(CZ_{1a}) \mathcal{L}(\tilde{C}_L)_{1,\dots,n,a} = L(\tilde{C}_L^\dagger)_{1,\dots,n,a} CK_1(\tilde{C}_L)_{1,\dots,n,a},$$

where we used the fact that the circuits \tilde{C}_L and its inverse are unchanged by the unencoding (this holds since they do not act on the ancilla register). Now, the Toffoli and X gate can be simulated by H and CCZ . Hence, $\{Q_n\}_n$ can be simulated by only the gates CCZ and H and a single CK gate. \square

Let us now specialize Eq. (8.86) to the controlled-controlled complex conjugation operator CCK , which is a special case of K_L . While we know from Proposition (151) that CCK can be simulated using one CCZ gate, the following equivalent simulation makes use of the previous discussion and shows that the ancilla need only be operated on by a single two-qubit gate.

Proposition 156.

$$\mathcal{P}(CCK_{ij}) = \langle 0 |_\alpha CCX_{ij\alpha} CZ_{\alpha\alpha} CCX_{ij\alpha} | 0 \rangle_\alpha. \quad (8.89)$$

Proof. We shall use the construction of the circuit described in Proposition 153, with

a few modifications to the labels. Recall that CCK corresponds to the language $L = \{x : x_i x_j = 1\}$, i.e. $L(x) = L(x_i x_j)$. Note that the circuit $C_L = CCX_{ij\alpha}$ maps $|x\rangle_{1,\dots,n} |0\rangle_\alpha \mapsto |x_i x_j\rangle_\alpha |x\rangle_{1,\dots,n} = |L(x)\rangle_\alpha |x\rangle_{1,\dots,n}$, which is of the form given in Eq. (8.85), except that the labels 1 and α are switched. Hence, by replacing the labels 1α in the CZ gate in Eq. (8.86) with $\alpha\alpha$, we obtain Eq. (8.89). \square

8.5 Universal gate sets for \mathbb{R} -unitaries

In Section 8.4, we found several examples of nonunitary operators, like K , CK and CCK , that can be simulated by orthogonal quantum circuits via the rebit encoding. The goal of this section is to find universal sets of gates for the \mathbb{R} -unitaries. We first introduce some definitions.

Definition 157. Let \mathcal{G}_1 and \mathcal{G}_2 be two gate sets¹⁸. We say that \mathcal{G}_2 *exactly simulates* \mathcal{G}_1 if for all $G \in \mathcal{G}_1$, there exists a circuit C formed using gates in \mathcal{G}_2 and ancilla registers (that can be initialized to any computational basis state) such that $\langle a | G | b \rangle = \langle a | C | b \rangle$ for all vectors a, b . We say that \mathcal{G}_2 *approximately simulates* \mathcal{G}_1 if any gate in \mathcal{G}_1 can be approximated in the operator norm (e.g. Definition 124) to within arbitrary accuracy by a sequence of gates from \mathcal{G}_2 .

Write $\mathcal{G}_1 \leq \mathcal{G}_2$ if \mathcal{G}_2 exactly simulates \mathcal{G}_1 ; and $\mathcal{G}_1 \lesssim \mathcal{G}_2$ if \mathcal{G}_2 approximately simulates \mathcal{G}_1 . If $\mathcal{G}_1 \leq \mathcal{G}_2$ and $\mathcal{G}_2 \leq \mathcal{G}_1$, i.e. if the gate sets exactly simulate each other, we write $\mathcal{G}_1 \equiv \mathcal{G}_2$, and say that \mathcal{G}_1 and \mathcal{G}_2 are *exact-simulation equivalent* to each other. If $\mathcal{G}_1 \lesssim \mathcal{G}_2$ and $\mathcal{G}_2 \lesssim \mathcal{G}_1$, i.e. if the gate sets approximately simulate each other, we write $\mathcal{G}_1 \cong \mathcal{G}_2$, and say that \mathcal{G}_1 and \mathcal{G}_2 are *approximate-simulation equivalent* to each other. Note that exact-simulation equivalence is a special case of approximate-simulation equivalence, and that approximate-simulation is equivalent to strict universality defined in [12]. Note also that \leq is a transitive relation, i.e. if $\mathcal{G}_1 \leq \mathcal{G}_2$ and $\mathcal{G}_2 \leq \mathcal{G}_3$, then $\mathcal{G}_1 \leq \mathcal{G}_3$. If either one or both of the first two \leq signs in the previous sentence is changed to \lesssim , then $\mathcal{G}_1 \lesssim \mathcal{G}_3$.

¹⁸gate set here refers to a set of gates, which may be either finite or infinite.

For a gate set $\mathcal{G} = \{g_1, \dots, g_s\}$, we say that g_1, \dots, g_s generate \mathfrak{G} if \mathfrak{G} is the set of all operators that can be written as circuits using gates from \mathcal{G} . We emphasize that we choose to specify generators g_j independently from their support, which can be chosen arbitrarily. That is, we view each g_j as a gate that may act on any set of qubits in the circuit. We denote the generated set as $\mathfrak{G} = \langle \mathcal{G} \rangle = \langle g_1, \dots, g_s \rangle$.

Examples of gate sets that approximately simulate the special unitary group on n qubits $SU(2^n)$ include the Clifford+T set $\{H, CZ, T\}$ [38], and Kitaev's gate set $\{H, CS\}$ [145] (also, see Theorem 1 of [12]). An example of gate set that approximately simulates the orthogonal operators T_n is $\{CCX, H\}$ (Theorem 3.2 of [203]). Since $CCZ_{ijk} = H_k CCX_{ijk} H_k$, it follows that CCZ and H can simulate CCX . Hence, $\{CCZ, H\}$ is also capable of approximately simulating T_n .

Other orthogonal gate sets, or even single gates, can simulate all the unitaries U_n via the rebit encoding. Rudolph and Grover [196] provide an example: the controlled-Y rotation $CY(\theta)$ for any θ that is an irrational multiple of π (e.g. $\theta = \pi/e$). More specifically, CY can be approximated to any desired accuracy by some power of $CY(\theta)$ and is orthogonal, so it can be applied in the rebit encoding without the ancilla. Single-qubit unitaries can be compiled (to any accuracy) from $CY(\theta)$ using parts (11) and (12) of Proposition 149.

Our main interest in this section is simulating $\mathbb{R}U_n$ using the rebit encoding. We now show that the set \mathbb{G} defined by

$$\mathbb{G} := \{H, CCZ, CCK, G(\frac{\pi}{4})K\} \tag{8.90}$$

can approximately simulate $\mathbb{R}U_n$.

Theorem 158. \mathbb{G} approximately simulates $\mathbb{R}U_n$.

Proof. Theorem 3.2 of [203] shows that $\{CCX, H\}$ approximately simulates T_{n+1} . Hence, the image of this set under \mathcal{L} (which is bijective) gives an approximate simulation for $\mathbb{R}U_n$. There are four cases corresponding to the gates acting on different sets of wires that we need to consider, namely, (i) CCZ gate acting on three numbered registers, (ii) H gate acting on a numbered register, (iii) CCZ gate acting on 2

numbered registers and the ancilla register, (iv) H gate acting on the ancilla register. From Propositions 149 and 151, the corresponding gates under \mathcal{L} are as follows:

$$CCZ_{ijk} \mapsto CCZ_{ijk} \quad (8.91)$$

$$H_i \mapsto H_i \quad (8.92)$$

$$CCZ_{ija} \mapsto CCK_{ij} \quad (8.93)$$

$$H_a \mapsto G(\frac{\pi}{4})K. \quad (8.94)$$

Hence, the set $\mathbb{G} = \{H, CCZ, CCK, G(\frac{\pi}{4})K\}$ approximately simulates $\mathbb{R}U_n$. \square

Since the gates in \mathbb{G} are contained in $\mathbb{R}U_n$, Theorem 158 tells us that $\mathbb{G} \cong \mathbb{R}U_n$. We now give some more examples¹⁹ of exact and approximate simulation with the goal of using these relations to find other universal gates sets starting from \mathbb{G} .

Lemma 159.

$$(i) \quad CCX \leq \{H, CCZ\} \quad (8.95)$$

$$(ii) \quad K \leq CK \leq CCK \quad (8.96)$$

$$(iii) \quad CS \leq \{G(\frac{\pi}{4}), CCK\} \quad (8.97)$$

$$(iv) \quad \{H, CCZ\} \lesssim \{H, CS\} \leq \{H, G(\frac{\pi}{4}), CCK\} \quad (8.98)$$

$$(v) \quad CCK \leq \{CCZ, H, CK\} \quad (8.99)$$

$$(vi) \quad \text{If } L \in \mathbb{P}, \text{ then } K_L \lesssim \{H, CCZ, CK\} \lesssim \{H, CCK, G(\frac{\pi}{4})\}. \quad (8.100)$$

Proof. To show that $G_1 \leq G_2$ or $G_1 \lesssim G_2$, it suffices to show that each gate in G_1 can be approximately simulated by a circuit consisting of gates in G_2 . (i) follows from the fact that

$$CCX_{ijk} = H_k CCZ_{ijk} H_k. \quad (8.101)$$

¹⁹Whenever a gate set consists of exactly one gate, we drop the curly braces and denote the set by the element it contains. For example, if G is a gate such that $\{G\}$ is exactly simulated by a gate set \mathcal{G} , we write $G \leq \mathcal{G}$ rather than $\{G\} \leq \mathcal{G}$.

(ii) follows from the facts that

$$K = \langle 1|_b CK_b |1\rangle_b, \quad CK_i = \langle 1|_b CCK_{ib} |1\rangle_b. \quad (8.102)$$

For (iii), taking \mathcal{L} on both sides of Item 9 of Proposition 149 gives:

$$CS_{ij} = \mathcal{L}(H_a \cdot CCZ_{ija} \cdot H_a \cdot CCZ_{ija}) = G(\frac{\pi}{4}) \cdot K \cdot CCK_{ij} \cdot G(\frac{\pi}{4}) \cdot K \cdot CCK_{ij}. \quad (8.103)$$

Hence, $CS \leq \{G(\frac{\pi}{4}), K, CCK\}$. But $K \leq CCK$ from Eq. (8.96). Hence, $CS \leq \{G(\frac{\pi}{4}), CCK\}$.

For (iv), $\{H, CS\}$ can approximately simulate any unitary [145] including $\{H, CCZ\}$.

Hence, $\{H, CCZ\} \lesssim \{H, CS\}$. Combining this result with Eq. (8.97) produces (iii).

For (v), by taking \mathcal{L} on both sides of Eq. (8.89), we obtain

$$CCK_{ij} = \mathcal{L}(\langle 0|_\alpha CCX_{ij\alpha} CZ_{\alpha\alpha} CCX_{ij\alpha} |0\rangle_\alpha) \quad (8.104)$$

$$= \langle 0|_\alpha CCX_{ij\alpha} CK_\alpha CCX_{ij\alpha} |0\rangle_\alpha. \quad (8.105)$$

Hence $CCK \leq \{CCX, CK\}$. By combining this result with Eq. (8.95), we get Eq. (8.99).

For (vi), we use Corollary 155, which says that if $L \in \mathbb{P}$, then K_L can be implemented by a uniform family of polynomial-sized quantum circuits that comprise only CCZ , H gates and exactly one CK gate. Hence, $K_L \lesssim \{H, CCZ, CK\}$. By using Eq. (8.96) and Eq. (8.98), $\{H, CCZ, CK\} \lesssim \{H, CCK, G(\frac{\pi}{4})\}$. \square

Using Lemma 159, we now give examples of various finite gate sets which are approximate-simulation equivalent to \mathbb{G} and therefore also to $\mathbb{R}U_n$. We start with finite gate sets.

Proposition 160. The following finite gates sets are all exact-simulation equivalent or approximate-simulation equivalent to one another. Hence, they are all approximate-simulation equivalent to $\mathbb{R}U_n$.

(i) $\{H, CCZ, CCK, G(\frac{\pi}{4})K, K\}$

$$(ii) \{H, CCZ, CCK, K, G(\frac{\pi}{4})\}$$

$$(iii) \{H, CCZ, CCK, G(\frac{\pi}{4})\}$$

$$(iv) \{H, CS, CCK, G(\frac{\pi}{4})\}$$

$$(v) \{H, CCK, G(\frac{\pi}{4})\}$$

$$(vi) \{H, CCZ, CK, G(\frac{\pi}{4})\}.$$

Proof.

- (i) \equiv (ii): Clearly, $G(\frac{\pi}{4})K \leq \{K, G(\frac{\pi}{4})\}$, so (i) \leq (ii). Also, $G(\frac{\pi}{4}) = G(\frac{\pi}{4})K \cdot K$, $G(\frac{\pi}{4}) \leq \{G(\frac{\pi}{4})K, K\}$, so (ii) \leq (i).
- (ii) \equiv (iii): Since $K \leq CCK$, by Eq. (8.96), so (ii) \leq (iii). Conversely, (iii) \subset (ii), so (iii) \leq (ii).
- (iii) \cong (iv) \cong (v): By Eq. (8.98), $\{H, CCZ\} \lesssim \{H, G(\frac{\pi}{4}), CCK\}$. Hence, (iii) \lesssim (iv). By Eq. (8.97), $CS \leq \{G(\frac{\pi}{4}), CCK\}$, so (iv) \leq (v). But (v) \subset (iii), so (v) \leq (iii).
- (v) \cong (vi): By Eq. (8.99), $CCK \leq \{CCZ, H, CK\}$, so (v) \leq (vi). By Eq. (8.98), $\{H, CCZ\} \lesssim \{H, G(\frac{\pi}{4}), CCK\}$, and by Eq. (8.96), $CK \leq CCK$, so (vi) \leq (v).

□

We can also find various infinite gate sets which that are approximate-simulation equivalent to $\mathbb{R}U_n$.

Proposition 161. The following infinite gates sets are approximate-simulation equivalent to $\mathbb{R}U_n$.

$$(vii) \{H, G(\frac{\pi}{4})\} \cup \{K_L : L \subseteq P\}$$

$$(viii) \{H, CS, G(\frac{\pi}{4})\} \cup \{K_L : L \subseteq P\}$$

- (ix) all the above gate sets in this list as well as in Proposition 160 with $G(\frac{\pi}{4})$ replaced by $\{G(\theta) : \theta \in [0, 2\pi)\}$.

Proof. We'll continue the numbering from Proposition 160.

- (vii) \cong (v): By Eq. (8.100), $K_L \lesssim \{H, CCK, G(\frac{\pi}{4})\}$. Hence, (vii) \lesssim (v). Also, (v) \subseteq (vii), so (v) \leq (vii).
- (vii) \equiv (viii): (vii) \subseteq (viii), so (vii) \leq (viii). Using $CS \leq \{G(\frac{\pi}{4}), CCK\}$ (from Eq. (8.97)) and $CCK \leq K_L$, we get (viii) \leq (vii).
- We first show that $LHS := \mathbb{R}U_n \cong RHS := \{H, CCZ, CCK, K\} \cup \{G(\theta) : \theta \in [0, 2\pi)\}$. Since $G(\frac{\pi}{4}) \in \{G(\theta) : \theta \in [0, 2\pi)\}$,

$$LHS = \mathbb{R}U_n \cong \{H, CCZ, CCK, K, G(\frac{\pi}{4})\} \leq RHS.$$

Conversely, the gates in RHS are all either unitary or are K or CCK . By Proposition 151, these are all images of orthogonal matrices under \mathcal{L} . Hence, $RHS \leq LHS$, which completes the proof of $LHS \cong RHS$. Next, notice that the set RHS is identical to the set (ii) in Proposition 160, except that $G(\frac{\pi}{4})$ is replaced by $G(\theta)$. Hence, making this replacement in all the above proofs, we get (ix).

□

Finally, we give a set of operators that exactly simulates $\mathbb{R}U_n$. Denote the set of operators which can be expressed as a product of partial antiunitary operators by

$$\begin{aligned} \langle \text{partial antiunitaries} \rangle &:= \langle V : V \text{ is a partial antiunitary operator} \rangle \\ &= \{W : \exists \text{ partial antiunitaries } W_1, \dots, W_k \\ &\quad \text{such that } W = W_1 \dots W_k\}. \end{aligned} \quad (8.106)$$

We show that the set in Eq. (8.106) is exact-simulation equivalent to the \mathbb{R} -unitary operators $\mathbb{R}U_n$. That is, any operator $\Gamma \in \mathbb{R}U_n$ can be written as the product $W_k W_{k-1} \dots W_1$ of some partial antiunitaries W_j . Indeed, we can even take $W_j \in \mathbb{R}U_n$ as well, implying we need no extra ancilla qubits. Since the \mathbb{R} -unitaries are the image

of the orthogonal operators under the decoding map \mathcal{L} , it is useful to have a lemma relating to the compiling of orthogonal operators.

Lemma 162. Let $W \in T_n$ be an n -qubit orthogonal operator. Then W can be written as the product of single-qubit orthogonal operators and multiply-controlled Z operators $C^h Z$ on the same n -qubits.

Proof. The proof is essentially the realization that the compilation scheme for unitaries in Chapter 4 of [180] works for compiling orthogonal gates into products of the claimed orthogonal gates as well. We complete the proof in Chapter 8.8.10. \square

Theorem 163.

$$\mathbb{R}U_n \equiv \langle \text{partial antiunitaries} \rangle. \quad (8.107)$$

Proof.

(\leq) This direction follows from the compiling lemma, Lemma 162. Let $\Gamma \in \mathbb{R}U_n$ and $\mathcal{P}(\Gamma) = W \in T_{n+1}$. Then the lemma provides us with a sequence of orthogonal gates $V_j \in T_{n+1}$ such that $V_k V_{k-1} \dots V_1 = W$. Thus, $\mathcal{L}(V_k) \mathcal{L}(V_{k-1}) \dots \mathcal{L}(V_1) = U$. Since V_j is orthogonal, if it is not supported on the rebit ancilla then $\mathcal{L}(V_j) = V_j$. If the ancilla is in its support there are two cases: either (1) V_j is a $C^h Z$ gate and so $\mathcal{L}(V_j)$ is a $C^h K$ gate or (2) V_j is a single-qubit orthogonal gate and so $\mathcal{L}(V_j)$ is a global phase gate. Therefore, $\Gamma = \mathcal{L}(V_k) \mathcal{L}(V_{k-1}) \dots \mathcal{L}(V_1)$ is indeed a sequence of unitaries alternating with partial complex conjugation operators, i.e.

$$\Gamma = U_{k'} K_{L_{k'}} U_{k'-1} K_{L_{k'-1}} U_{k'-2} \dots U_1 K_{L_1} U_0. \quad (8.108)$$

Since U_j and K_{L_j} are both partial antiunitary for all j , we have found a product of partial antiunitaries making Γ .

(\geq) This direction follows from Theorem 144, which says that all partial antiunitaries are \mathbb{R} -unitary. Products of partial antiunitaries, like those found in $\langle \text{partial antiunitaries} \rangle$, must also be \mathbb{R} -unitary because it is a group.

\square

To conclude this section, we discuss how efficient the rebit encoding is for simulating (1) an arbitrary unitary circuit (top-down simulation) and (2) an arbitrary \mathbb{R} -unitary circuit (bottom-up simulation). In the top-down case, we consider the universal gate set $\{H, T, CX\}$ [180]. In the bottom-up case, using Proposition 160, we focus on the universal gate set $\{H, CCZ, CK, G(\pi/4)\}$ because it is relatively simple. Similar analyses could be done with any other universal sets of gates.

Let us start with the top-down case. Say we have a depth- d circuit consisting of gates from $\{H, T, CX\}$ on n -qubits. What is the depth and width of a rebit circuit required to simulate it? We provide two approaches trading off depth and width.

Theorem 164. Let C be an n -qubit, depth- d unitary circuit using gates from the Clifford+T gate set $\{H, T, CX\}$. Then C (applied to $|0\rangle^{\otimes n}$) can be simulated (i.e. we can make $\mathcal{P}(C|0\rangle^{\otimes n})$) using either

1. an orthogonal circuit of depth at most dn on $n + 1$ rebits, or
2. an orthogonal circuit of depth at most d on $2n$ rebits.

Proof. To show the first statement, we proceed using the \mathcal{P} mappings of the gates $\{H, T, CX\}$ from Proposition 149. Since T gates that happen in parallel (at most n at once) must access the ancilla simultaneously, we get a depth blowup by a factor of n of the orthogonal circuit over the unitary circuit it is simulating.

The following rebit encoding was used in [170] to make rebit simulations local. Here we use it to prove the second statement, removing the depth blowup of the first statement at the cost of using more rebits. The idea is quite simple: encode the ancilla rebit so that some logical Pauli operator (in this case, logical Y) is accessible n times in parallel. The obvious code for this is the classical redundancy code. Let $|\bar{0}\rangle, |\bar{1}\rangle$ be the encoded $|0\rangle$ and $|1\rangle$. These should be $+1$ -eigenstates of the stabilizers $Y_j Y_{j+1}$ for all $j = 1, 2, \dots, n - 1$ and the ± 1 -eigenstates of the encoded Z operator $\bar{Z} = Z_1 Z_2 \dots Z_n$. Working it out, the states are

$$|\bar{0}\rangle = \frac{1}{\sqrt{2^{n-1}}} \sum_{\substack{x \in \{0,1\}^n \\ |x| \text{ even}}} (-1)^{|x|/2} |x\rangle, \quad |\bar{1}\rangle = \frac{1}{\sqrt{2^{n-1}}} \sum_{\substack{x \in \{0,1\}^n \\ |x| \text{ odd}}} (-1)^{(|x|-1)/2} |x\rangle. \quad (8.109)$$

where $|x|$ is the Hamming weight of the string x , i.e. the number of 1's in x . From Proposition 149, the simulation of T_i for any $i = 1, 2, \dots, n$ is

$$CH_{ia}CZ_{ia} = \frac{1}{2}(I + Z_i) + \frac{1}{2}(I - Z_i)\frac{1}{\sqrt{2}}(I - iY_a). \quad (8.110)$$

Since encoded Y is $\bar{Y} = Y_{ai}$ for any qubit ai in the encoded ancilla, we can modify the simulation of T_i to

$$CH_{ia}CZ_{ia} = \frac{1}{2}(I + Z_i) + \frac{1}{2}(I - Z_i)\frac{1}{\sqrt{2}}(I - iY_{ai}). \quad (8.111)$$

With this modification, the simulations of T_i and T_j for $i \neq j$ are orthogonal operators with disjoint support and can be performed in parallel. \square

Next we discuss efficiency of the rebit bottom-up simulation in the same manner, i.e. we are concerned with the simulation of an \mathbb{R} -unitary circuit on n qubits with depth d . The notion of depth is not immediately obvious for circuits constructed from the gates $\{H, CCZ, CK, G(\pi/4)\}$, but we can use the following definition. This gives us a well-defined notion of depth that leads to a theorem similar to Theorem 164.

Definition 165. An n -qubit, depth-1 \mathbb{R} -unitary circuit consists of gates G_i , $i = 1, 2, \dots, s$ such that $[G_i, G_j] = 0$ (i.e. all gates mutually commute) and all $q \in \{1, 2, \dots, n\}$ is in the support of exactly one gate G_i . A depth- d \mathbb{R} -unitary circuit equals the product of d depth-1 \mathbb{R} -unitary circuits.

Gates $\{H, CCZ, CK, G(\pi/4)\}$ have supports of sizes 1, 3, 1, 0 respectively, and, when all supports are disjoint, only CK and $G(\pi/4)$ do not commute. Definition 165 generalizes the notion of circuit depth from the unitary to the \mathbb{R} -unitary case, because two unitary gates having disjoint support implies that they commute.

We have the following theorem.

Theorem 166. Let C be an n -qubit, depth- d \mathbb{R} -unitary circuit using gates from $\{H, CCZ, CK, G(\pi/4)\}$. Then C (applied to $|0\rangle^{\otimes n}$) can be simulated (i.e. we can make $\mathcal{P}(C|0\rangle^{\otimes n})$) using either

1. a circuit of depth at most dn on $n + 1$ rebits
2. or a circuit of depth at most $d\lceil\log_2 n\rceil$ on $2n$ rebits.

Proof. The first part follows the same reasoning as the proof of the first part of Theorem 164. In this case, the 1-qubit CK gates are the problem. At most n CK can occur in parallel, but to simulate each we need access to the ancilla (CK_i is simulated by CZ_{ia} by Proposition 151).

The second part also follows similar reasoning to that of Theorem 164. We encode the rebit in a redundancy code, although this time one in which the encoded Z operator is accessible in parallel. The code states are simply the redundancy states

$$|\bar{0}\rangle = |0\rangle^{\otimes n}, \quad |\bar{1}\rangle = |1\rangle^{\otimes n}. \quad (8.112)$$

With this, the encoded Z is $\bar{Z} = Z_{ai}$ for any qubit in the ancilla ai and encoded Y is $\bar{Y} = X_{a1}X_{a2}\dots X_{a,n-1}Y_{an}$. Thus, CK_i and CK_j gates (for $i \neq j$) can be simulated in parallel. However, $G(\pi/4)$ requires depth $\lceil\log_2 n\rceil$ to simulate: decode the states (i.e. $|\bar{0}\rangle \rightarrow |0\rangle^{\otimes n}$ and $|\bar{1}\rangle \rightarrow |1\rangle|0\rangle^{\otimes n-1}$ by $\lceil\log n\rceil$ timesteps of CX gates), apply a Y -rotation to the first qubit, and re-encode. \square

8.6 The \mathbb{R} -Clifford hierarchy

An intriguing consequence of rebit simulation is an extension of the standard Clifford hierarchy of unitary operators into the more general \mathbb{R} -unitaries. At the second level of this \mathbb{R} -Clifford hierarchy, we obtain an extension of the famous Gottesman-Knill theorem, allowing us to efficiently classically simulate \mathbb{R} -linear quantum circuits of a restricted class that is analogous to but larger than the standard Clifford circuits.

We begin by defining the Clifford hierarchy and the \mathbb{R} -Clifford hierarchy. The standard Pauli group (on n -qubits) is

$$\mathcal{C}_n(1) = \{e^{i\alpha}(p_1 \otimes p_2 \otimes \dots \otimes p_n) : p_j \in \{I, X, Y, Z\}, \alpha \in \mathbb{R}\}. \quad (8.113)$$

An analogous Pauli group incorporating complex conjugation may be defined as

$$\mathcal{C}'_n(1) = \{i^c(p_1 \otimes p_2 \otimes \cdots \otimes p_n)K^b : p_j \in \{I, X, Y, Z\}, c \in \{0, 1, 2, 3\}, b \in \{0, 1\}\}, \quad (8.114)$$

which we, for now, call simply the primed Paulis. Shortly, we show the primed Paulis are actually the \mathbb{R} -Paulis.

Notice that the two definitions have different global phases — in the case of $\mathcal{C}_n(1)$ the phase is arbitrary, while for $\mathcal{C}'_n(1)$ it is restricted to powers of i . This is intentional and should be expected, because the rebit encoding tracks global phases. Note that the definition of the Pauli group in Eq. (8.113) differs from that defined in [180]:

$$G_n(1) = \{i^c(p_1 \otimes p_2 \otimes \cdots \otimes p_n) : p_j \in \{I, X, Y, Z\}, c \in \{0, 1, 2, 3\}\}. \quad (8.115)$$

We define the Pauli group differently simply to ease some of the arguments below (specifically, Lemma 174). Allowing arbitrary phases via definition $\mathcal{C}_n(1)$ is also more consistent with operators in higher levels of the hierarchy having arbitrary phases as well.

Now, we appeal to the discussion of bottom-up simulation in Section 8.1.2.1, taking the Paulis $\mathcal{C}_n(1)$ as the set of operators \mathcal{S} . The real Paulis on n rebits is the set of orthogonal Paulis $\mathcal{C}_n^{\mathbb{R}}(1) = \mathcal{C}_n(1) \cap R_n = \mathcal{C}_n(1) \cap T_n$, and the \mathbb{R} -Paulis are $\mathbb{R}\mathcal{C}_n(1) = \mathcal{L}(\mathcal{C}_n^{\mathbb{R}}(1))$. The following theorem shows the primed Paulis are the \mathbb{R} -Paulis, thus establishing the Pauli-like description, Eq. (8.114), of the set of \mathbb{R} -Paulis as appropriate.

Theorem 167. $\mathcal{L}(\mathcal{C}_{n+1}^{\mathbb{R}}(1)) := \mathbb{R}\mathcal{C}_n(1) = \mathcal{C}'_n(1)$.

Proof. First, note that $p \in \mathcal{C}_{n+1}(1)$ is orthogonal if and only if p contains an even number of Pauli Y s and a real phase (i.e. $e^{i\alpha}$ from Eq. (8.113) is ± 1) or p contains an odd number of Pauli Y s and an imaginary phase ($\pm i$). Since \mathcal{L} is a homomorphism, we need only consider its action on a basis set of orthogonal Paulis, namely $\{X_i, iY_i, Z_i, X_a, iY_a, Z_a\}$ where $i = 1, 2, \dots, n$ indicates a data qubit and a indicates

the rebit ancilla. We find

$$\mathcal{L}(X_i) = X_i, \quad (8.116)$$

$$\mathcal{L}(iY_i) = iY_i, \quad (8.117)$$

$$\mathcal{L}(Z_i) = Z_i, \quad (8.118)$$

$$\mathcal{L}(X_a) = iK, \quad (8.119)$$

$$\mathcal{L}(iY_a) = -iI, \quad (8.120)$$

$$\mathcal{L}(Z_a) = K, \quad (8.121)$$

all of which are elements of $\mathcal{C}'_n(1)$. This shows $\mathcal{L}(\mathcal{C}_{n+1}^{\mathbb{R}}) \subseteq \mathcal{C}'_n(1)$. However, the reverse direction, $\mathcal{L}(\mathcal{C}_{n+1}^{\mathbb{R}}(1)) \supseteq \mathcal{C}'_n(1)$ follows from Eqs. (8.116-8.121) as well. Let $p = i^c X^{\vec{x}} Z^{\vec{z}} K^b \in \mathcal{C}'_n(1)$, where $X^{\vec{x}}$ for $\vec{x} \in \{0, 1\}^n$ means $\bigotimes_{j=1}^n X_j^{\vec{x}_j}$ and likewise for $Z^{\vec{z}}$. Then,

$$p = \mathcal{L}(X^{\vec{x}})\mathcal{L}(Z^{\vec{z}})\mathcal{L}(iY_a)^c\mathcal{L}(Z_a^b) = \mathcal{L}(X^{\vec{x}}Z^{\vec{z}}(iY_a)^c Z_a^b). \quad (8.122)$$

Since $X^{\vec{x}}Z^{\vec{z}}(iY_a)^c Z_a^b$ is an orthogonal Pauli, this proves $\mathcal{C}'_n(1) \subseteq \mathcal{L}(\mathcal{C}_{n+1}^{\mathbb{R}}(1))$. \square

The \mathbb{R} -Paulis are a group just as the standard Paulis are.

Corollary 168. $\mathbb{R}\mathcal{C}_n(1)$ is a group for all n .

Proof. Follows from Proposition 118. \square

The upper levels of the standard Clifford hierarchy are defined recursively

$$\mathcal{C}_n(k) = \{U \in U_n : UC_n(1)U^\dagger \subseteq \mathcal{C}_n(k-1)\}. \quad (8.123)$$

Note that the first level is the Pauli group and the second level is the Clifford group. In Chapter 8.8.11, we show how using Pauli sets with different allowed global phases (e.g. $G_n(1)$ instead of $\mathcal{C}_n(1)$) leads to the same Clifford hierarchy for $k \geq 2$.

We would like a similar recursion to Eq. (8.123) to hold for the \mathbb{R} -Clifford hierarchy. Thus, we proceed similarly to the Pauli case above and define the *primed Clifford*

hierarchy as

$$\mathcal{C}'_n(k) = \{U \in \mathbb{R}L_n : U(\mathcal{C}'_n(1))U^\dagger \subseteq \mathcal{C}'_n(k-1)\}. \quad (8.124)$$

Our goal now is to show that the primed Clifford hierarchy is equivalent to the \mathbb{R} -Clifford hierarchy. That is, $\mathcal{C}'_n(k)$ is exactly $\mathbb{R}\mathcal{C}_n(k) := \mathcal{L}(\mathcal{C}_n(k) \cap T_n)$. Because \mathbb{R} -unitaries are mapped to orthogonal operators in the physical rebit encoding, we find it natural to define the orthogonal Clifford hierarchy

$$\mathcal{D}_n(1) = \mathcal{C}_n(1) \cap T_n, \quad (8.125)$$

$$\mathcal{D}_n(k) = \{U \in T_n : U\mathcal{D}_n(1)U^T \subseteq \mathcal{D}_n(k-1)\}. \quad (8.126)$$

Then, it is worth noting the following definition of the orthogonal hierarchy as the real Cliffords.

Lemma 169. $\mathcal{D}_n(k) = \mathcal{C}_n(k) \cap T_n := \mathcal{C}_n^{\mathbb{R}}(k)$ for all k .

Proof. The second equality is a definition of notation. To prove the first, we proceed inductively, with $k = 1$ already satisfying the claim by definition. Let $U \in \mathcal{C}_n(k) \cap T_n$. Then $U\mathcal{D}_n(1)U^T \subseteq U\mathcal{C}_n(1)U^T \subseteq \mathcal{C}_n(k-1)$ and $U\mathcal{D}_n(1)U^T \subseteq T_n$ implying that $U\mathcal{D}_n(1)U^T \subseteq \mathcal{C}_n(k-1) \cap T_n = \mathcal{D}_n(k-1)$ and thus, $U \in \mathcal{D}_n(k)$ by definition.

In the other direction, let $U \in \mathcal{D}_n(k)$. We notice that for all $p \in \mathcal{C}_n(1)$, there exists a phase $e^{-i\alpha}$ such that $e^{-i\alpha}p \in \mathcal{C}_n(1) \cap T_n = \mathcal{D}_n(1)$. This is because $p = e^{i\alpha}X^{\bar{x}}Z^{\bar{z}}$ implies $pp^T = e^{i2\alpha}I$. Thus, $U\mathcal{D}_n(1)U^T \subseteq \mathcal{D}_n(k-1) = \mathcal{C}_n(k-1) \cap T_n$ implies $U\mathcal{C}_n(1)U^T \subseteq \mathcal{C}_n(k-1)$. So $U \in \mathcal{C}_n(k)$ by definition, and thus $U \in \mathcal{C}_n(k) \cap T_n$. \square

Now we are in a position to prove that the primed hierarchy is indeed the \mathbb{R} -Clifford hierarchy.

Theorem 170. $\mathcal{L}(\mathcal{C}_{n+1}^{\mathbb{R}}(k)) := \mathbb{R}\mathcal{C}_n(k) = \mathcal{C}'_n(k)$ for all k .

Proof. The $k = 1$ case is proven in Theorem 167. For the rest, we proceed inductively.

Let $U \in \mathcal{C}_{n+1}^{\mathbb{R}}(k)$. Then $U(\mathcal{C}_{n+1}^{\mathbb{R}}(1))U^T \subseteq \mathcal{C}_{n+1}^{\mathbb{R}}(k-1)$. Taking \mathcal{L} of both sides, we get

$$\mathcal{L}(U)\mathcal{C}'_n(1)\mathcal{L}(U)^\dagger = \mathcal{L}(U)\mathcal{L}(\mathcal{C}_{n+1}^{\mathbb{R}}(1))\mathcal{L}(U)^\dagger \quad (8.127)$$

$$= \mathcal{L}(UC_{n+1}^{\mathbb{R}}(1)U^T) \subseteq \mathcal{L}(C_{n+1}^{\mathbb{R}}(k-1)) = C'_n(k-1) \quad (8.128)$$

using the inductive hypothesis. Thus, $\mathcal{L}(U) \in C'_n(k)$, showing $\mathcal{L}(C_{n+1}^{\mathbb{R}}(k)) \subseteq C'_n(k)$.

For the other direction, let $U \in C'_n(k)$. Then,

$$U\mathcal{L}(C_{n+1}^{\mathbb{R}}(1))U^\dagger = UC'_n(1)U^\dagger \subseteq C'_n(k-1) = \mathcal{L}(C_{n+1}^{\mathbb{R}}(k-1)). \quad (8.129)$$

Taking \mathcal{P} of both sides, we find $\mathcal{P}(U)C_{n+1}^{\mathbb{R}}(1)\mathcal{P}(U)^T \subseteq C_{n+1}^{\mathbb{R}}(k-1)$, which implies $\mathcal{P}(U) \in C_{n+1}^{\mathbb{R}}(k)$. Thus, $U \in \mathcal{L}(C_{n+1}^{\mathbb{R}}(k))$ and showing $C'_n(k) \subseteq \mathcal{L}(C_{n+1}^{\mathbb{R}}(k))$. \square

There are some corollaries of Theorem 170. For instance, just as standard Cliffords form a group, so do the \mathbb{R} -Cliffords.

Corollary 171. $\mathbb{R}C_n(2)$ is a group.

Proof. Follows from Proposition 118. \square

While the group $G_n(1)$ defined in Eq. (8.115) is a subgroup of $C'_n(1) = \mathbb{R}C_n(1)$, the groups $C_n(1)$ and $\mathbb{R}C_n(1)$ are incomparable:

Proposition 172. $C_n(1) \not\subseteq \mathbb{R}C_n(1)$ and $\mathbb{R}C_n(1) \not\subseteq C_n(1)$.

Proof. The first noninclusion follows from the fact that the operator $e^{i\pi/4}I$ is in $C_n(1)$ but not in $\mathbb{R}C_n(1)$. The second noninclusion follows from the fact that K is in $\mathbb{R}C_n(1)$ but not in $C_n(1)$. \square

Furthermore, the groups $C_n(2)$ and $\mathbb{R}C_n(2)$ are also incomparable:

Proposition 173. $C_n(2) \not\subseteq \mathbb{R}C_n(2)$ and $\mathbb{R}C_n(2) \not\subseteq C_n(2)$.

Proof. The first noninclusion follows from the fact that the operator $e^{i\pi/8}I$ is in $C_n(2)$ but not in $\mathbb{R}C_n(2)$. To see the latter, note that $e^{i\pi/8}Ke^{-i\pi/8} = e^{i\pi/4}K \notin \mathbb{R}C_n(1)$. The second noninclusion follows from the fact that K is in $\mathbb{R}C_n(2)$ but not in $C_n(2)$. \square

However, Propositions 172 and 173 notwithstanding, it turns out that if we disregard global phases, then the groups $C_n(k)$ and $\mathbb{R}C_n(k)$ (for $k = 1, 2$) are no longer

incomparable. More precisely, for at least the first two levels, the \mathbb{R} -Clifford hierarchy $\mathbb{R}\mathcal{C}_n(k)$ is strictly larger than the standard hierarchy $\mathcal{C}_n(k)$. A lemma regarding some structure of the standard Pauli and Clifford groups helps us show this.

Lemma 174. $\{UU^T : U \in \mathcal{C}_n(2)\} = \{p : p = p^T \in \mathcal{C}_n(1)\} \subseteq \mathcal{C}_n(1)$.

Proof. That any symmetric Pauli $p = p^T \in \mathcal{C}_n(1)$ can be written as UU^T for some $U \in \mathcal{C}_n(2)$ is not hard to see. Let $p = e^{i\alpha} X^{\vec{x}} Z^{\vec{z}}$ with $\vec{x} \cdot \vec{z} = 0$, enforcing the symmetry of p . Let $J_x = \{j : \vec{x}_j = 1 - \vec{z}_j = 1\}$, $J_y = \{j : \vec{x}_j = \vec{z}_j = 1\}$, $J_z = \{j : 1 - \vec{x}_j = \vec{z}_j = 1\}$. Since $|J_y|$ is even, we can partition it into two equal sized subsets J_y^1 and J_y^2 with a one-to-one mapping $\sigma : J_y^1 \rightarrow J_y^2$. Let

$$U = e^{i\alpha/2} \left(\prod_{i \in J_y^1} CX_{i\sigma(i)} \right) \left(\prod_{j \in J_x \cup J_y^1} H_j \right) \left(\prod_{k \in J_x \cup J_y \cup J_z} S_k \right). \quad (8.130)$$

One can now calculate that

$$\begin{aligned} UU^T &= e^{i\alpha} \left(\prod_{i \in J_y^1} CX_{i\sigma(i)} \right) \left(\prod_{j \in J_x \cup J_y^1} H_j \right) \left(\prod_{k \in J_x \cup J_y \cup J_z} Z_k \right) \left(\prod_{j \in J_x \cup J_y^1} H_j \right) \\ &\quad \times \left(\prod_{i \in J_y^1} CX_{i\sigma(i)} \right) \\ &= e^{i\alpha} \left(\prod_{i \in J_y^1} CX_{i\sigma(i)} \right) \left(\prod_{j \in J_x \cup J_y^1} X_j \right) \left(\prod_{k \in J_y^2 \cup J_z} Z_k \right) \left(\prod_{i \in J_y^1} CX_{i\sigma(i)} \right) \\ &= e^{i\alpha} \left(\prod_{j \in J_x \cup J_y} X_j \right) \left(\prod_{k \in J_y \cup J_z} Z_k \right) \\ &= e^{i\alpha} X^{\vec{x}} Z^{\vec{z}} = p. \end{aligned} \quad (8.131)$$

For the other direction, we need $UU^T \in \mathcal{C}_n(1)$ for any $U \in \mathcal{C}_n(2)$, but it suffices²⁰

²⁰Here, we show that if $VpV^\dagger = \pm p$ for all $p \in \mathcal{C}_n(1)$, then $V \in \mathcal{C}_n(1)$. Indeed, since the Paulis form a basis, we may write $V = \sum_{q \in P_n} \alpha_q q$ for complex coefficients α_q , where $P_n = \{p_1 \otimes p_2 \otimes \cdots \otimes p_n : p_j \in \{I, X, Y, Z\}\}$ is the set of Pauli operators without global phases. Let $p \in \mathcal{C}_n(1)$ be supported on one qubit (i.e. $p = X_i, Y_i, Z_i$ for qubit i). Then $VpV^\dagger = \pm p$ implies that exactly half of the coefficients α_q are zero (e.g. in the case of $+$, all α_q such that $\{p, q\} = 0$ are zero). Repeating for all qubits i and all X_i, Y_i, Z_i , all but one coefficient are zeroed, thus implying $V \in \mathcal{C}_n(1)$.

to show that for all $p \in \mathcal{C}_n(1)$, $(UU^T)p(UU^T)^\dagger = \pm p$ where the choice of sign may depend on p . In fact, it suffices to show that

$$(UU^T)p(UU^T)^\dagger = ap \tag{8.132}$$

for any complex number a . This is because, squaring both sides and using the fact that $p^2 \sim I$ for all $p \in \mathcal{C}_n(1)$, we get $(UU^T)p^2(UU^T)^\dagger = p^2 = a^2p^2$, implying $a = \pm 1$.

We now prove Eq. (8.132). Notice that for any $q \in \mathcal{C}_n(1)$,

$$q \sim \bar{q}. \tag{8.133}$$

Also, notice that since $\mathcal{C}_n(2)$ is closed under taking inverses, $U^\dagger \in \mathcal{C}_n(2)$, which implies that $U^\dagger pU \in \mathcal{C}_n(1)$.

Hence,

$$\begin{aligned} (UU^T)p(UU^T)^\dagger &= UU^T p \bar{U} U^\dagger \\ &\sim UU^T \bar{p} \bar{U} U^\dagger, \quad \text{by applying Eq. (8.133) to } p \in \mathcal{C}_n(1). \\ &= U \left(\overline{U^\dagger p U} \right) U^\dagger \\ &\sim U(U^\dagger p U)U^\dagger, \quad \text{by applying Eq. (8.133) to } U^\dagger p U \in \mathcal{C}_n(1) \\ &= p. \end{aligned} \tag{8.134}$$

□

We write $A \lesssim B$ if for all $a \in A$, there exists $\alpha \in \mathbb{R}$ such that $e^{i\alpha} a \in B$.

Proposition 175. $G_n(1) \cup G_n(1)K = \mathbb{R}\mathcal{C}_n(1)$, $\mathcal{C}_n(1) \lesssim \mathbb{R}\mathcal{C}_n(1)$, and $\mathcal{C}_n(2) \lesssim \mathbb{R}\mathcal{C}_n(2)$.

Proof. The first and second statements follow easily from definitions Eqs. (8.113), (8.114), and (8.115). It remains to prove the third inclusion. Let $U \in \mathcal{C}_n(2)$. Then by Lemma 174, UU^T is Pauli, i.e.

$$UU^T = e^{i\beta} X^{\bar{x}} Z^{\bar{z}} \tag{8.135}$$

for some $\beta \in \mathbb{R}$ and $\vec{x}, \vec{z} \in \{0, 1\}^n$. To complete the proof, it suffices to find some $\alpha \in \mathbb{R}$ such that $e^{i\alpha}U \in \mathbb{RC}_n(2)$. To this end, we choose $\alpha = -\beta/2$, and show that $U' := e^{-i\beta/2}U \in \mathbb{RC}_n(2)$.

Let $\zeta = i^c X^{\vec{x}} Z^{\vec{z}} K^b \in \mathbb{RC}_n(1)$ be arbitrary. If $b = 0$, then

$$\begin{aligned} U'\zeta U'^{\dagger} &= e^{-i\beta/2}U(i^c X^{\vec{x}} Z^{\vec{z}})e^{i\beta/2}U^{\dagger} \\ &= U(i^c X^{\vec{x}} Z^{\vec{z}})U^{\dagger} \\ &\in G_n(1) \subseteq \mathbb{RC}_n(1), \end{aligned} \tag{8.136}$$

where, since $U \in \mathbb{C}_n(2)$ is Clifford and $\zeta = i^c X^{\vec{x}} Z^{\vec{z}} \in G_n(1)$, Theorem 194 in Chapter 8.8.11 shows $U\zeta U^{\dagger} \in P_n$. If $b = 1$, then

$$\begin{aligned} U'\zeta U'^{\dagger} &= e^{-i\beta/2}U(i^c X^{\vec{x}} Z^{\vec{z}} K)e^{i\beta/2}U^{\dagger} \\ &= e^{-i\beta}U(i^c X^{\vec{x}} Z^{\vec{z}})U^T K \\ &= e^{-i\beta}U(i^c X^{\vec{x}} Z^{\vec{z}})U^{\dagger}(UU^T)K \\ &= e^{-i\beta}U(i^c X^{\vec{x}} Z^{\vec{z}})U^{\dagger}e^{i\beta}X^{\vec{x}} Z^{\vec{z}} K \\ &= U(i^c X^{\vec{x}} Z^{\vec{z}})U^{\dagger}X^{\vec{x}} Z^{\vec{z}} K \\ &\in G_n(1)K \subseteq \mathbb{RC}_n(1) \end{aligned} \tag{8.137}$$

where the last line follows because $U(i^c X^{\vec{x}} Z^{\vec{z}})U^{\dagger} \in G_n(1)$ and $X^{\vec{x}} Z^{\vec{z}} \in G_n(1)$. Since $G_n(1)$ is closed under multiplication, the expression in the second-to-last line of Eq. (8.137) is of the form Eq. (8.114). \square

A final corollary of Theorem 170 corollary is a Gottesman-Knill-esque efficient classical simulation of $\mathbb{RC}_n(2)$ circuits and a generating set for them.

Corollary 176. Let $U \in \mathbb{RC}_n(2)$ be an n -qubit \mathbb{R} -linear operator. Then U can be constructed from $O(n^2)$ gates from $\{H, S, K, CX, CK\}$. Moreover, a classical computer can sample from $U|0\rangle^{\otimes n}$ in time $O(n^2)$.

Proof. The first statement can be proved by compiling $V = \mathcal{P}(U) \in \mathbb{C}_{n+1}^{\mathbb{R}}(2)$ using gates from the orthogonal Clifford group $\{H, Z, CX\}$. That this is possible with the

requisite number of gates follows the same Clifford compiling argument from [180] (see their Theorem 10.6).

The first step is to argue that $\{H, Z\}$ generate all single-qubit orthogonal Cliffords. A single-qubit orthogonal Clifford is uniquely specified up to a phase (which for orthogonal operators is just ± 1) by its action on the Paulis $\{X, iY, Z\}$. However, since $\det(X) = \det(Z) = -1$ and $\det(iY) = 1$, orthogonal Cliffords must map $\{X, Z\}$ to $\{\pm X, \pm Z\}$ and $\{iY\}$ to $\{\pm iY\}$. That appropriate sequences of H and Z can achieve all these mappings can be checked directly by enumeration. Moreover, the phase ± 1 can be provided by $(HZ)^4 = -I$.

The second part of the proof is inductive on the number of qubits $n + 1$. There are $O(n)$ recursive steps, each using $O(n)$ gates from our gate set $\{H, Z, CX\}$. Say $VX_1V^T = g$ and $VZ_1V^T = h$ for $g, h \in \mathcal{C}_{n+1}^{\mathbb{R}}(1)$. There are simplifications that can be made without loss of generality, however. First, we note that g and h anticommute, which means that on some qubit j , the Paulis there locally anticommute. We can apply a SWAP P between qubits 1 and j (which can be constructed from three CX gates), so that we get $V'X_1V'^T = p_1 \otimes g'$ and $V'Z_1V'^T = p_2 \otimes h'$ with $p_1, p_2 \in \mathcal{C}_1(1)$, $g', h' \in \mathcal{C}_n(1)$, $\{p_1, p_2\} = 0$, $[g', h'] = 0$, and instead compile $V' = PV$. Second, by applying H before or after V' , we can switch the roles of X_1 and Z_1 and change p_1, p_2 . The final result is that, without loss of generality, we have two cases to consider: either

$$VX_1V^T = X \otimes g, \quad VZ_1V^T = Z \otimes h \quad (8.138)$$

or

$$VX_1V^T = (iY) \otimes g, \quad VZ_1V^T = Z \otimes h \quad (8.139)$$

where in both cases $g, h \in \mathcal{C}_n(1)$ and $[g, h] = 0$.

We claim the circuit in Fig. 8-4 implements these two cases and does so using the allowed gates $\{H, Z, CX\}$. The controlled n -qubit orthogonal Paulis (e.g. controlled- g from qubit c to qubits t_1, t_2, \dots, t_n) can be implemented by a depth at most $2n$ circuit of CX and $CZ = (I \otimes H)CX(I \otimes H)$ gates. If $g = g_1 \otimes g_2 \otimes \dots \otimes g_n$ where each $g_j \in \{X, iY, Z\}$, then performing controlled- g_j from qubit c to t_j for all j (in

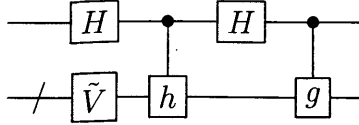


Figure 8-4: Compiling an orthogonal Clifford circuit on n qubits. The top line represents 1 qubit while the bottom represents $n - 1$ qubits.

any order; they commute) implements controlled- g . Controlled- X and controlled- Z operators are simply CX and CZ , while controlled- iY is CX followed by CZ . Finally, the circuit in Fig. 8-4 guarantees the correct behavior of V on the first qubit and thus \tilde{V} is an n qubit orthogonal Clifford, which can be compiled using the same process. The recursion continues until the base case of 1-qubit, discussed earlier.

Given the described compilation, CHP simulation [8] of the orthogonal Clifford circuit for V in the rebit encoding suffices to simulate the \mathbb{R} -Clifford U , and does so in time $O(n^2)$. □

8.7 Discussion and open questions

Our bottom-up simulation paradigm provides a unified framework for realizing any arbitrary antiunitary or partial antiunitary transformations which are otherwise non-physical and cannot be simulated directly. The rebit simulation can be applied to measurements of a large variety of quantum mechanical properties as well as the detection and simulation of exotic phases of matters. These applications all require either antiunitary or partial antiunitary transformations that are unphysical for quantum mechanical systems.

For example, evaluating the entanglement of an arbitrary partition of a generic quantum system usually takes $O(2^N)$ measurements for an N -qubit system [126]. To avoid such a resource overhead, entanglement monotones such as concurrence [237] and 3-tangle [86] are proposed to provide convex and monotonic measures that do not increase under local operations and classical communication. However, both the concurrence and 3-tangle are defined by the expectation values of an antiunitary operator which cannot be directly measured. To directly measure these entanglement

monotones, an extra qubit is needed to simulate the complex conjugation on the original system [83]. Our result on partial antiunitary simulation thus further generalizes such approaches to larger systems where the concurrence of only a subsystem can be measured.

As another example, time-reversal symmetry and particle-hole symmetry are two important ingredients for defining either bosonic or fermionic symmetry protected topological phases [65, 116]. The system symmetry is defined by the invariance of the system Hamiltonian \hat{H} under the conjugation of the corresponding antiunitary transformation U_c for either the time-reversal or particle-hole symmetry as: $U_c \hat{H} U_c^\dagger = \hat{H}$. Partial-time-reversal and partial-particle-hole symmetries corresponding to the invariance under partial antiunitary transformations are also used for constructing nonlocal order parameters in detecting fermionic symmetry protected topological phases in (1+1) dimension [204]. Detection of these symmetry protected topological phases is exceedingly hard since these symmetry operators are non-physical and cannot be directly measured. Being able to simulate both antiunitary and partial antiunitary transformation with our rebit encoding can potentially simplify the detection procedure for topological phases proposed in [187].

Our results add new tools to the existing dictionary of quantum simulation gadgets using qubits. The Majorana equation [64], for example, is one candidate for describing the dynamics of neutrino or other particles outside the standard model. Simulating the Majorana equation in quantum systems necessitates the application of the complex conjugation of the wave function, which is readily available in our bottom-up rebit simulation.

We conclude this section by listing a few directions that an extension of this project might take.

Mixed states. Our treatment in this chapter has been restricted to just pure states, unitary transformations and projective measurements. This suffices since, we could always “go to the Church of the Larger Hilbert Space” by considering mixed states as being part of a larger system described by a pure state (see Chapter 2.2). Never-

theless, describing quantum systems using the smaller Hilbert Space has also proven to be fruitful, as it allows for the study of noisy quantum systems without any reference to a fictitious external system. Our bottom-up approach to rebits might benefit from such an approach. What is the rebit generalization of completely positive and trace preserving maps? Can they be described in terms of some generalized Kraus operators?

Quaternions. In this chapter, we studied the relationship between computing using real and complex amplitudes. But the real numbers and complex numbers are just the two base levels of the Cayley-Dickson construction [200]. The next level of the construction are the quaternions, which was studied in the context of computation by [101]. It would be interesting to apply the techniques from our chapter to study computing based on quaternions (or even other levels of the Cayley-Dickson construction) from a bottom-up perspective.

Compiling. We showed (Theorem 163) that an arbitrary \mathbb{R} -unitary Γ can always be written as products of partial antiunitaries and further noted that these products always take the form of an alternating sequence $U_N K_{L_N} U_{N-1} K_{L_{N-1}} \dots U_1 K_{L_1} U_0$ of unitaries U_j and partial complex conjugations K_{L_j} over languages L_j . But given Γ and desired accuracy ϵ , how efficient is it to determine the length of the sequence required to approximate Γ to within operator norm ϵ and also the specific unitaries and languages? In principle, applying Solovay-Kitaev [78] in the simulator space P provides an algorithm and upper bounds, but it is well-known even in unitary compilation that Solovay-Kitaev is not optimal. The exact question $\epsilon = 0$ is also interesting and leads to the definition of a minimum N for which exact compilation of Γ (call it e.g. the “conjugation depth” of Γ) is possible. For instance, the conjugation depth of any unitary is zero, the partial antiunitaries have conjugation depth one by definition, and Theorem 148 shows that some \mathbb{R} -unitaries have conjugation depth at least two. Ideally, the conjugation depth of Γ might be determined from some simple property of Γ .

Clifford hierarchy. In our discussion of the Clifford hierarchy, we have focused on the first two levels of the hierarchy. For example, in Proposition 175, we showed that, up to a global phase, the \mathbb{R} -Clifford hierarchy (for the first two levels) is bigger than the standard Clifford hierarchy. Does an analogous result hold for higher levels of the hierarchy? Next, we see from the definitions in Eq. (8.113) and Eq. (8.114) that to get from $\mathcal{C}_n(1)$ to $\mathbb{R}\mathcal{C}_n(1)$, we need to append the K gate to the list of generators of the Pauli group. Also, we see from Corollary 176 that to get from $\mathcal{C}_n(2)$ to $\mathbb{R}\mathcal{C}_n(2)$, we need to append both the K and CK gate to the list of generators of the Clifford group. Can the k th level (for $k > 2$) of the \mathbb{R} -Clifford hierarchy be obtained by appending gates to the corresponding level of the standard Clifford hierarchy?

8.8 Appendix for Chapter 8

8.8.1 A simple motivating example

In this section, we present a simple motivating example to illustrate how nonunitary transformations can be simulated using the rebit encoding. Consider a general one-qubit state with complex amplitudes:

$$|\psi\rangle = (a + ib)|0\rangle + (c + id)|1\rangle,$$

where $a, b, c, d \in \mathbb{R}$. The single-ancilla rebit encoding is performed by introducing an additional register and encoding the state $|\psi\rangle$ as

$$|\psi'\rangle = a|00\rangle + b|01\rangle + c|10\rangle + d|11\rangle.$$

To illustrate some nonlinear transformations that we can simulate using this rebit encoding, suppose we perform Z on the second qubit, getting the state

$$|\chi'\rangle = a|00\rangle - b|01\rangle + c|10\rangle - d|11\rangle,$$

which evidently is the rebit encoding of the complex conjugation of $|\psi\rangle$,

$$|\bar{\psi}\rangle = (a - ib)|0\rangle + (c - id)|1\rangle.$$

Thus, via the rebit encoding, we have simulated the *antiunitary* complex conjugation operation $K : |\psi\rangle \rightarrow \Re|\psi\rangle - i\Im|\psi\rangle$. Now, consider a more complicated example in which we perform a controlled- Z operation on $|\psi'\rangle$ to get the state

$$|\phi'\rangle = a|00\rangle + b|01\rangle + c|10\rangle - d|11\rangle.$$

We observe that $|\phi'\rangle$ is the rebit encoding of the state

$$|\phi\rangle = (a + ib)|0\rangle + (c - id)|1\rangle.$$

Hence, we have shown how to simulate the nonlinear transformation

$$(a + ib)|0\rangle + (c + id)|1\rangle \mapsto (a + ib)|0\rangle + (c - id)|1\rangle$$

using the rebit encoding. This transformation is an example of what we call a *partial antiunitary* operator.

8.8.2 Complex conjugation as a Gottesman-Knill simulation

In the introduction, we discussed how viewing Gottesman-Knill as a bottom-up simulation implies that a more “advanced” Clifford quantum computer equipped with Pauli measurements that report entire probability distributions instead of just samples from them can be efficiently classically simulated as well. Here we expand on this bottom-up viewpoint, by showing that Gottesman-Knill also gives an efficient classical simulation of circuits consisting of Clifford gates and complex conjugation.

Gottesman-Knill is a bottom-up simulation (L, P, \mathcal{P}, O_P) where L is the set of

rank one density matrices

$$L = \{\rho = |\psi\rangle\langle\psi| : \exists p_j \in \mathcal{C}_n(1) \text{ s.t. } \rho = \prod_{j=1}^n \frac{1}{2}(I + p_j)\}, \quad (8.140)$$

with $\mathcal{C}_n(1)$ the Pauli group on n qubits (see Eq. (8.113)). The physical space P is

$$P = \{S : S \subseteq \mathcal{C}_n(1); |S| = n; e^{i\alpha} I \in \langle S \rangle \Rightarrow e^{i\alpha} = 1\} \quad (8.141)$$

where $\langle S \rangle$ for a set of Paulis S is the group generated by S . A Pauli $p = \pm p_1 \otimes p_2 \otimes \cdots \otimes p_n$ on n qubits can be specified by $2n + 1$ classical bits [8]. All Paulis $p \in S \in P$ must have this form (i.e. with global phase such that $p^2 = I$) by the final condition on S in Eq. (8.141).

We next describe the map \mathcal{P} . Starting with a stabilizer state $\rho = |\psi\rangle\langle\psi|$, identify n linearly independent Paulis p_i such that $p_i |\psi\rangle = |\psi\rangle$. These exist because ρ can be written as in Eq. (8.140). Set $\mathcal{P}(\rho) = \{p_i : i = 1, 2, \dots, n\}$.

The bottom-up view stresses that we should define the set of operators O_P that the simulator is capable of implementing. In this case, the most general operator the simulator can perform is any function $f : P \rightarrow P$. If we wanted to, we could also restrict to those functions f that are efficiently computable (say, in polynomial time in n). Regardless, we do not have a characterization of $\mathcal{L}(O_P)$ for either of these choices of O_P . Our immediate goal is to show that complex conjugation (of density matrices) $K_{\text{dm}} : |\psi\rangle\langle\psi| \rightarrow \Re|\psi\rangle\langle\psi| - i\Im|\psi\rangle\langle\psi|$ is an element of $\mathcal{L}(O_P)$ (for either choice of O_P) and thus that Gottesman-Knill simulation is intriguingly more powerful than typically imagined.

Proposition 177. K_{dm} is efficiently classically simulable by the Gottesman-Knill Theorem.

Proof. Suppose S represents a density matrix $|\psi\rangle\langle\psi|$. Let $\#Y(q) = |\{p_i = Y : i = 1, 2, \dots, n\}|$ for a Pauli $q = e^{i\alpha} q_1 \otimes q_2 \otimes \cdots \otimes q_n$ and

$$S' = \{(-1)^{\#Y(p)} p : p \in S\}. \quad (8.142)$$

We claim $S' = \mathcal{P}(K_{\text{dm}}(|\psi\rangle\langle\psi|))$. To show this, write $|\psi\rangle\langle\psi| = \prod_{j=1}^n \frac{1}{2}(I + p_j)$ where $p_j \in S$ for all j . Then $K_{\text{dm}}(|\psi\rangle\langle\psi|) = \prod_{j=1}^n \frac{1}{2}(I + (-1)^{\#Y(p_j)} p_j)$. Thus, $(-1)^{\#Y(p_j)} p_j \in \mathcal{P}(K_{\text{dm}}(|\psi\rangle\langle\psi|))$ for all j , which exactly matches the composition of S' . \square

8.8.3 \mathbb{R} -linear operators

In this section, we consider operators of the form $A+BK$, where A and B are complex linear operators. Our first result is a proof of Theorem 121 (restated here as Theorem 178):

Theorem 178. ([130]) Let V and V' be complex vector spaces, and $f : V \rightarrow V'$ be a function on V . Then, there exist linear maps A and B such that $f = A + BK$ if and only if

$$f(ax + by) = af(x) + bf(y) \quad (8.143)$$

for all $a, b \in \mathbb{R}$ and $x, y \in V$.

Proof. The forward direction holds since $(A + BK)(ax + by) = a(A + BK)x + b(A + BK)y$ for all $x, y \in V$ and $a, b \in \mathbb{R}$. To prove the backward direction, assume that f satisfies Eq.(8.143). Let $z \in V$. Let the standard basis of V be $\{e_i\}_i$. Hence, we can write $z = \sum_j z_j e_j$ for some $z_j \in \mathbb{C}$. Then,

$$\begin{aligned} f(z) &= f\left(\sum_j z_j e_j\right) \\ &= f\left(\sum_i (\Re z_j + i \Im z_j) e_j\right) \\ &= \sum_j \Re z_j f(e_j) + \Im z_j f(i e_j), \quad \text{by } \mathbb{R}\text{-linearity} \\ &= \sum_j \frac{z_j + \bar{z}_j}{2} f(e_j) + \frac{z_j - \bar{z}_j}{2i} f(i e_j) \\ &= \sum_j \frac{1}{2}(f(e_j) - i f(i e_j)) z_j + \sum_j \frac{1}{2}(f(e_j) + i f(i e_j)) \bar{z}_j \\ &= \sum_j A_j z_j + \sum_j B_j K z_j \end{aligned}$$

$$= (A + BK)z \quad (8.144)$$

where A is the (complex-valued) matrix whose j th column is $A_j = \frac{1}{2}(f(e_j) - if(ie_j))$, and B is the matrix whose j th column is $B_j = \frac{1}{2}(f(e_j) + if(ie_j))$. Hence, $f = A + BK$, where A and B are (complex) linear maps on V . \square

As pointed out in Section 8.1.3.1, in linear algebra, the term \mathbb{R} -linear is used to describe a map satisfying Eq. (8.143). Our terminology in this chapter was chosen so that the two definitions of \mathbb{R} -linearity coincide.

We conclude this section with a few remarks about \mathbb{R} -linear operators satisfying Eq. (8.143). First, note that linear operators and antilinear operators are both special cases of \mathbb{R} -linear operators. (A linear operator $g : V \rightarrow V$ is one that satisfies $g(ax + by) = ag(x) + bg(y)$ for all $a, b \in \mathbb{C}$ and $x, y \in V$, and an antilinear operator h is one that satisfies $h(ax + by) = \bar{a}h(x) + \bar{b}h(y)$ for all $a, b \in \mathbb{C}$ and $x, y \in V$.)

Second, note that when A and B are complex linear operators, the operator A is linear while the operator BK is antilinear. Hence, Theorem 178 implies that any \mathbb{R} -linear operator can be written as a sum of a linear operator and an antilinear operator.

8.8.4 The ring of \mathbb{R} -linear operators: algebraic properties

In Section 8.8.3, we showed that every \mathbb{R} -linear operator on $\mathcal{H}_n(\mathbb{C})$ can be written as $A + BK$, where A and B are linear operators on $\mathcal{H}_n(\mathbb{C})$. In this section, we show that the set of \mathbb{R} -linear operators $\mathbb{R}L_n$ forms a ring with identity²¹, with addition + given by

$$(A + BK) + (C + DK) = (A + C) + (B + D)K \quad (8.145)$$

and multiplication \star given by

$$(A + BK) \star (C + DK) = (AC + B\bar{D}) + (AD + B\bar{C})K. \quad (8.146)$$

²¹For an introduction to ring theory, see [85], for example.

Proposition 179. Let $n \in \mathbb{Z}^+$. Then $(\mathbb{R}L_n, +, \star)$ is a ring with identity. The multiplicative identity is $I + 0K$.

Proof. It is straightforward to check that $\mathbb{R}L_n$ satisfies the properties of a ring with identity (see Chapter 7 of [85]). \square

It is easy to check that $(\mathbb{R}L_n, +, \star)$ is neither a division ring nor a commutative ring, and hence is not a field. Note that the multiplication in Eq. (8.146) was defined so that for any vector $v \in \mathcal{H}_n(\mathbb{C})$,

$$((A + BK) \star (C + DK))v = (A + BK)((C + DK)v). \quad (8.147)$$

More generally, the set of vectors in $\mathcal{H}_n(\mathbb{C})$ forms a module over $\mathbb{R}L_n$, as the following proposition states.

Proposition 180. The set $\mathcal{H}_n(\mathbb{C})$ is a left module over the ring $(\mathbb{R}L_n, +, \star)$, with the addition on $\mathcal{H}_n(\mathbb{C})$ being the usual addition of functions, and the module action \circ of $\mathbb{R}L_n$ on $\mathcal{H}_n(\mathbb{C})$ given by

$$(A + BK) \circ v := (A + BK)v = Av + B\bar{v}.$$

Proof. It is straightforward to check that the above satisfies the properties of a left module (see Chapter 10 of [85]). \square

It is useful to equip the ring $(\mathbb{R}L_n, +, \star)$ with the operator \dagger defined as follows:

$$(A + BK)^\dagger = A^\dagger + B^T K. \quad (8.148)$$

We now show that the \dagger (super)operator is the image of the transpose map $(\cdot)^T$ under the rebit decoding. More precisely, let \mathcal{E} be an operator on $L_{n+1}^{\mathbb{R}}$. Define $\mathcal{L}(\mathcal{E})$ to be the unique operator $\tilde{\mathcal{E}}$ such that

$$\tilde{\mathcal{E}}(A + BK) = \mathcal{L}(\mathcal{E}(\mathcal{P}(A + BK))).$$

It then follows that

Proposition 181. $\mathcal{L}((\cdot)^T) = (\cdot)^\dagger$.

Proof.

$$\begin{aligned}
A + BK &= (\Re A + i\Im A) + (\Re B + i\Im B)K \\
&\xrightarrow{\mathcal{P}} \Re A \otimes I + \Im A \otimes XZ + \Re B \otimes Z + \Im B \otimes X \\
&\xrightarrow{(\cdot)^T} \Re A^T \otimes I + \Im A^T \otimes ZX + \Re B^T \otimes Z + \Im B^T \otimes X \\
&= \Re A^T \otimes I + \Im(-A^T) \otimes XZ + \Re B^T \otimes Z + \Im B^T \otimes X \\
&\xrightarrow{\mathcal{L}} (\Re A^T - i\Im A^T) + (\Re B^T + i\Im B^T)K \\
&= A^\dagger + B^T K = (A + BK)^\dagger.
\end{aligned} \tag{8.149}$$

Hence, $\mathcal{L}((\cdot)^T) = (\cdot)^\dagger$. □

It is straightforward to check that the operator \dagger is an involutive antiautomorphism²² Hence, we obtain the following proposition.

Proposition 182. The ring $(\mathbb{R}L_n, +, \star)$ together with the operator \dagger defined in Eq. (8.148) is a \dagger -ring.

Note that the involutive antiautomorphism \dagger generalizes the definition of the adjoint of linear operators. Indeed, when $B = 0$ in Eq. (8.148), we recover $A^\dagger = (A + 0K)^\dagger = A^\dagger$. We may now generalize the notion of unitarity to \dagger -rings with identity. Let R be a \dagger -ring with identity 1. We say that $U \in R$ is a *unitary element* with respect to \dagger if

$$U^\dagger U = 1. \tag{8.150}$$

Applying the above definition to the \dagger -ring $(\mathbb{R}L_n, +, \star)$, we get that an element $A + BK \in \mathbb{R}L_n$ is a unitary element if and only if

$$(A + BK)^\dagger \star (A + BK) = I. \tag{8.151}$$

²²Let R be a ring, and let $\ast : R \rightarrow R$. R together with \ast is a \ast -ring if for all $x, y \in R$, (i) $(x^\ast)^\ast = x$, (ii) $(x + y)^\ast = x^\ast + y^\ast$, (iii) $(xy)^\ast = y^\ast x^\ast$. The map \ast is called an *involutive antiautomorphism*.

The group of unitary elements are called $\mathbb{R}U_n$ in the main text, as a result of Theorem 127 showing that they are the simulated operators of a real unitary rebit simulator.

We now give an equivalent condition for the unitarity of \mathbb{R} -linear operators.

Proposition 183. An element $A + BK \in \mathbb{R}L_n$ is a unitary element with respect to \dagger if and only if

$$\begin{aligned} A^\dagger A + B^T \bar{B} &= I, \\ A^\dagger B + B^T \bar{A} &= 0. \end{aligned} \tag{8.152}$$

Proof. An element $A + BK \in \mathbb{R}L_n$ is a unitary element if and only if $I = (A + BK)^\dagger \star (A + BK) = (A^\dagger + B^T K) \star (A + BK) = (A^\dagger A + B^T \bar{B}) + (A^\dagger B + B^T \bar{A})K$ if and only if $A^\dagger A + B^T \bar{B} = I, A^\dagger B + B^T \bar{A} = 0$. \square

8.8.5 Equivalent expressions for the rebit encoding of a linear operator

From Eq. (8.29), we find that the rebit encoding of a linear operator is given by $\mathcal{P}(A) = \Re A \otimes I + \Im A \otimes XZ$. In this section, we derive alternative expressions for Eq. (8.29).

Proposition 184. The rebit encoding of a linear operator A is equal to

$$\mathcal{P}(A) = \bar{A} \otimes |\otimes\rangle\langle\otimes| + A \otimes |\odot\rangle\langle\odot| = (\bar{A} \otimes I)(I \otimes |\otimes\rangle\langle\otimes| + A^T \otimes |\odot\rangle\langle\odot|), \tag{8.153}$$

where $|\otimes\rangle\langle\otimes| = \frac{1}{2}(I + Y)$ and $|\odot\rangle\langle\odot| = \frac{1}{2}(I - Y)$ are the orthogonal projectors onto the $+1$ and -1 eigenspaces of the Pauli matrix Y , respectively.

Proof. Substituting into Eq. (8.29) the identities $\Re A = 1/2(A + \bar{A})$, $\Im A = 1/(2i)(A - \bar{A})$ and $XZ = -iY$, we obtain

$$\mathcal{P}(A) = \frac{1}{2} [(A + \bar{A}) \otimes I - (A - \bar{A}) \otimes Y]$$

$$= \frac{1}{2} [A \otimes (I - Y) + \bar{A} \otimes (I + Y)],$$

which is equal to $\bar{A} \otimes |\otimes\rangle\langle\otimes| + A \otimes |\odot\rangle\langle\odot|$. \square

In particular, if $A = U$ is unitary,

$$\mathcal{P}(U) = (\bar{U} \otimes I) (I \otimes |\otimes\rangle\langle\otimes| + U^T U \otimes |\odot\rangle\langle\odot|). \quad (8.154)$$

To compare with the rebit simulation of linear operators in [12], we calculate the action of \mathcal{P} on states written in the computational basis as follows:

Proposition 185.

$$\mathcal{P}(A) : \sum_{ij} \psi_{ij} |ij\rangle \mapsto \sum_i [(\psi_{i0} \Re A - \psi_{i1} \Im A) |i\rangle] |0\rangle + \sum_i [(\psi_{i0} \Im A + \psi_{i1} \Re A) |i\rangle] |1\rangle. \quad (8.155)$$

Proof.

$$\mathcal{P}(A) : \sum_{ij} \psi_{ij} |ij\rangle = \sum_{ij} \psi_{ij} [(\Re A |i\rangle) |j\rangle + (\Im A |i\rangle) \otimes XZ |j\rangle]. \quad (8.156)$$

But $XZ |j\rangle = (-1)^j |1-j\rangle$. Plugging this into Eq. (8.156) and expanding out the j index, we obtain Eq. (8.155). \square

By setting $\psi_{i'j'} = \delta_{i'i} \delta_{j'j}$ in Eq. (8.155), we obtain

$$\mathcal{P}(A) |i\rangle |0\rangle = (\Re A |i\rangle) |0\rangle + (\Im A |i\rangle) |1\rangle, \quad (8.157)$$

$$\mathcal{P}(A) |i\rangle |1\rangle = -(\Im A |i\rangle) |0\rangle + (\Re A |i\rangle) |1\rangle, \quad (8.158)$$

which is equivalent to Definition 1 of [12].

8.8.6 Equivalence of norm definitions

In this section, we provide a proof that on L_n the operator norm for \mathbb{R} -linear operators coincides with that for linear operators.

Proposition 186. Let $A \in L_n$ be a linear operator on n -qubits. Then $\|A\|$ as defined by Definitions 123 and 124 are the same.

Proof. For clarity, denote $\|A\|_l$ as the operator norm for linear operators from Definition 123 and $\|A\|_r$ as the operator norm for \mathbb{R} -linear operators from Definition 124. Because $\|A\|_l$ is the largest singular value of A and $\|A\|_r$ is the largest singular value of $\mathcal{P}(A)$, we need only show these coincide. Say $A = UDV$ for unitaries $U, V \in U_n$ and diagonal matrix D so that $\text{sing}(A) = \{D_{ii}, \forall i\}$ are the singular values of A and $\|A\|_l = \max\{|\lambda| : \lambda \in \text{sing}(A)\}$. Now, $\mathcal{P}(A) = \mathcal{P}(U)\mathcal{P}(D)\mathcal{P}(V)$, and the singular value decomposition of $\mathcal{P}(D)$ is easily calculated using Proposition 184,

$$\mathcal{P}(D) = \bar{D} \otimes |\otimes\rangle \langle \otimes| + D \otimes |\odot\rangle \langle \odot| = (I \otimes SH)(\bar{D} \otimes |0\rangle \langle 0| + D \otimes |1\rangle \langle 1|)(I \otimes HS^\dagger). \quad (8.159)$$

Thus, $\text{sing}(\mathcal{P}(A)) = \text{sing}(\mathcal{P}(A)) = \text{sing}(A) \cup \text{sing}(\bar{A})$. Finally, $\|A\|_r = \|\mathcal{P}(A)\|_l = \max\{|\lambda| : \lambda \in \text{sing}(\mathcal{P}(A))\} = \|A\|_l$. \square

8.8.7 Alternative formulation of Theorem 126

In Theorem 126, we showed that for an \mathbb{R} -linear operator $A + BK$, the operator $\mathcal{P}(A + BK)$ is orthogonal if and only if $A^\dagger A + B^T \bar{B} = I$ and $A^\dagger B + B^T \bar{A} = 0$.

We now find an equivalent condition for orthogonality.

Theorem 187. Let $A + BK$ be an \mathbb{R} -linear operator. Then $\mathcal{P}(A + BK)$ is orthogonal if and only if

$$\begin{aligned} AA^\dagger + BB^\dagger &= I, \\ AB^T + BA^T &= 0. \end{aligned} \quad (8.160)$$

In the proof of Theorem 126, we used the property that a matrix W is orthogonal if and only if $W^T W = I$. But this is equivalent to the condition that $W W^T = I$. Repeating the proof of Theorem 126 using this condition yields Eq. (8.160). An alternative approach, which we use here, is to directly show that Eq. (8.33) is equivalent to Eq. (8.160).

Proof.

$$\begin{aligned}
& A^\dagger A + B^T \bar{B} = I, \quad A^\dagger B + B^T \bar{A} = 0 \\
\iff & I = (A^\dagger A + B^T \bar{B}) + (A^\dagger B + B^T \bar{A})K \\
& = (A^\dagger + B^T K) \star (A + BK) \\
\iff & I = (A + BK) \star (A^\dagger + B^T K) \\
& = (AA^\dagger + BB^\dagger) + (BA^T + AB^T)K \\
\iff & AA^\dagger + BB^\dagger = I, \quad AB^T + BA^T = 0, \tag{8.161}
\end{aligned}$$

where we used the star product \star defined in Eq. (8.146), and the fact that left inverses are equal to right inverses in a ring (See Chapter 8.8.4). \square

8.8.8 On orthogonal projections

In this section, we recall some definitions about orthogonal projections. Let $\mathbb{H} \subseteq \mathcal{H}$ be a subspace of a vector space \mathcal{H} . The orthogonal complement of \mathbb{H} is the set $\mathbb{H}^\perp = \{u \in \mathcal{H} | \langle v, u \rangle = 0 \ \forall v \in \mathbb{H}\}$. For finite-dimensional vector spaces, $(\mathbb{H}^\perp)^\perp = \mathbb{H}$, and \mathbb{H} and \mathbb{H}^\perp are complementary subspaces, i.e. $\mathbb{H} \cap \mathbb{H}^\perp = \{0\}$ and $\mathbb{H} \oplus \mathbb{H}^\perp = \mathcal{H}$, where \oplus denotes direct sum. Moreover, for any $v \in \mathcal{H}$, there exist a unique $a \in \mathbb{H}$ and a unique $b \in \mathbb{H}^\perp$ such that $v = a + b$. The map $v \mapsto a$ is called the orthogonal projection onto \mathbb{H} , and we shall denote it by $\text{proj}_{\mathbb{H}}(\cdot)$. It then follows that the map $v \mapsto b$ is equal to $\text{proj}_{\mathbb{H}^\perp}(\cdot)$. Two immediate consequences are that $\text{proj}_{\mathbb{H}} + \text{proj}_{\mathbb{H}^\perp} = I$ and that $\text{proj}_{\mathbb{H}} \circ \text{proj}_{\mathbb{H}^\perp} = \text{proj}_{\mathbb{H}^\perp} \circ \text{proj}_{\mathbb{H}} = 0$. It is also easy to verify that orthogonal projections are linear operators that are idempotent and hermitian, i.e. $\text{proj}_{\mathbb{H}} = \text{proj}_{\mathbb{H}}^2 = \text{proj}_{\mathbb{H}}^\dagger$.

An alternate characterization of orthogonal projections is as follows:

Proposition 188. Let $\mathbb{H} \subseteq \mathcal{H}$ be a subspace of a vector space \mathcal{H} . Let $\{|a_i\rangle\}_{i=1}^s$ be an orthonormal basis for S . Then P is an orthogonal projection onto \mathbb{H} if and only if

$$P = \sum_{i=1}^s |a_i\rangle\langle a_i|. \tag{8.162}$$

Note that Proposition 188 implies that the formula in Eq. (8.162) is in fact independent of the basis chosen for \mathbb{H} , i.e. if $\{|a_i\rangle\}_{i=1}^s$ and $\{|b_i\rangle\}_{i=1}^s$ are two bases for \mathbb{H} , then $\sum_{i=1}^s |a_i\rangle\langle a_i| = \sum_{i=1}^s |b_i\rangle\langle b_i|$.

8.8.9 Matrix representation of \mathbb{R} -linear operators

In this section, we develop a matrix notation for the \mathbb{R} -linear operators. We define the *matrix representation* of an \mathbb{R} -linear operator $A + BK$ to be the matrix $[A + BK]$ whose (μ, ν) th element is the column vector $\begin{pmatrix} A_{\mu\nu} \\ B_{\mu\nu} \end{pmatrix} \in \mathbb{C}^2$. Each of these elements $\begin{pmatrix} A_{\mu\nu} \\ B_{\mu\nu} \end{pmatrix}$ can be seen as belonging to the ring $R = \left\{ \begin{pmatrix} a \\ b \end{pmatrix} : a, b \in \mathbb{C} \right\}$, where addition is defined by

$$\begin{pmatrix} a \\ b \end{pmatrix} + \begin{pmatrix} c \\ d \end{pmatrix} = \begin{pmatrix} a + c \\ b + d \end{pmatrix}$$

and multiplication is defined by

$$\begin{pmatrix} a \\ b \end{pmatrix} \begin{pmatrix} c \\ d \end{pmatrix} = \begin{pmatrix} ac + b\bar{d} \\ ad + b\bar{c} \end{pmatrix}.$$

We shall sometimes also denote the column vector $\begin{pmatrix} a \\ b \end{pmatrix}$ belonging to R by $a + bk$, where k is treated as a *formal* symbol (for example, see its use in the proof of Theorem 148). With this definition, the set of matrices $[A + BK]$ forms a matrix ring under the *usual*²³ rules of matrix addition and matrix multiplication.

Note that the matrix representation of an \mathbb{R} -linear operator is unique, as the following theorem, which expresses the matrix representation of an \mathbb{R} -linear operator Γ in terms of Γ , shows

²³ Unlike usual matrix multiplication, the elements of the matrix belong to a ring, while the elements in the column vector representation of a vector belong to a field that is different from the ring. For a more rigorous treatment, we should treat the vector space \mathbb{C}^n as a unital left R -module [85].

Theorem 189. Let Γ be an \mathbb{R} -linear operator. Then its matrix elements are given by²⁴

$$\Gamma_{\mu\nu} = \frac{1}{2} (\Gamma(e_\nu) - i\Gamma(ie_\nu) | \Gamma(e_\nu) + i\Gamma(ie_\nu))^T e_\mu, \quad (8.163)$$

where e_μ is the μ th basis vector defined by $e_\mu(\nu) = \delta_{\mu\nu}$.

Proof. Since Γ is an \mathbb{R} -linear operator, we can write $\Gamma = A + BK$, where A and B are both linear operators.

$$\begin{aligned} RHS &= \frac{1}{2} (\Gamma(e_\nu) - i\Gamma(ie_\nu) | \Gamma(e_\nu) + i\Gamma(ie_\nu))^T e_\mu \\ &= \frac{1}{2} (\langle e_\mu, \Gamma(e_\nu) - i\Gamma(ie_\nu) \rangle, \langle e_\mu, \Gamma(e_\nu) + i\Gamma(ie_\nu) \rangle)^T \\ &= \frac{1}{2} (\underbrace{\langle e_\mu, \Gamma(e_\nu) \rangle}_{(0)} - i \underbrace{\langle e_\mu, \Gamma(ie_\nu) \rangle}_{(1)}, \underbrace{\langle e_\mu, \Gamma(e_\nu) \rangle}_{(0)} + i \underbrace{\langle e_\mu, \Gamma(ie_\nu) \rangle}_{(1)})^T, \end{aligned}$$

where for $\alpha = 0, 1$,

$$\begin{aligned} (\alpha) &= \langle e_\mu, \Gamma(i^\alpha e_\nu) \rangle \\ &= \langle e_\mu, (A + BK)(i^\alpha e_\nu) \rangle \\ &= \langle e_\mu, i^\alpha A(e_\nu) + (-1)^\alpha i^\alpha B(e_\nu) \rangle \\ &= i^\alpha (A_{\mu\nu} + (-1)^\alpha B_{\mu\nu}). \end{aligned}$$

Hence, $(0) - i(1) = 2A_{\mu\nu}$ and $(0) + i(1) = 2B_{\mu\nu}$.

So,

$$\begin{aligned} RHS &= \frac{1}{2} (2A_{\mu\nu}, 2B_{\mu\nu})^T \\ &= \begin{pmatrix} A_{\mu\nu} \\ B_{\mu\nu} \end{pmatrix} \\ &= (A + BK)_{\mu,\nu} = \Gamma_{\mu\nu}. \end{aligned}$$

□

²⁴The notation $(A|B)$ refers to the augmented matrix formed from the matrices A and B .

Note that when Γ is linear, the expression in Eq. (8.163) reduces to the following familiar expression.

$$\begin{aligned}
\Gamma_{\mu\nu} &= \frac{1}{2} (\Gamma(e_\nu) + \Gamma(e_\nu) | \Gamma(e_\nu) - \Gamma(e_\nu))^T e_\mu \\
&= (\Gamma(e_\nu) | 0)^T e_\mu \\
&= (\langle e_\mu, \Gamma(e_\nu) \rangle, 0)^T \\
&= \langle e_\mu, \Gamma(e_\nu) \rangle + 0K \\
&= \langle e_\mu, \Gamma(e_\nu) \rangle.
\end{aligned}$$

We will now illustrate the use of Theorem 189 in an example.

Example 190. Let $\Gamma : \mathbb{C}^2 \rightarrow \mathbb{C}^2$ be an \mathbb{R} -linear operator. Then

$$\Gamma : \begin{pmatrix} \alpha \\ \beta \end{pmatrix} \mapsto \begin{pmatrix} 2\Re\alpha + 2\bar{\beta} \\ 3\beta \end{pmatrix}$$

if and only if the matrix representation of Γ is

$$\begin{pmatrix} 1+k & 2k \\ 0 & 3 \end{pmatrix}.$$

Proof. The backward direction follows from matrix multiplication

$$\begin{aligned}
\begin{pmatrix} 1+k & 2k \\ 0 & 3 \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \end{pmatrix} &= \begin{pmatrix} 1 & 0 \\ 0 & 3 \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \end{pmatrix} + \begin{pmatrix} 1 & 2 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} \bar{\alpha} \\ \bar{\beta} \end{pmatrix} \\
&= \begin{pmatrix} \alpha + \bar{\alpha} + 2\bar{\beta} \\ 3\beta \end{pmatrix} = \begin{pmatrix} 2\Re\alpha + 2\bar{\beta} \\ 3\beta \end{pmatrix}
\end{aligned}$$

To obtain the forward direction, we make use of Theorem 189:

$$\Gamma_{\mu,0} = \frac{1}{2} \left[\begin{pmatrix} 2 \\ 0 \end{pmatrix} - i \begin{pmatrix} 0 \\ 0 \end{pmatrix} \mid \begin{pmatrix} 2 \\ 0 \end{pmatrix} + i \begin{pmatrix} 0 \\ 0 \end{pmatrix} \right]^T e_\mu \quad (8.164)$$

$$= \begin{pmatrix} 1 & 0 \\ 1 & 0 \end{pmatrix} e_\mu \quad (8.165)$$

and

$$\Gamma_{\mu,1} = \frac{1}{2} \left[\begin{pmatrix} 2 \\ 3 \end{pmatrix} - i \begin{pmatrix} -2i \\ 3i \end{pmatrix} \middle| \begin{pmatrix} 2 \\ 3 \end{pmatrix} + i \begin{pmatrix} -2i \\ 3i \end{pmatrix} \right]^T e_\mu \quad (8.166)$$

$$= \begin{pmatrix} 0 & 3 \\ 2 & 0 \end{pmatrix} e_\mu. \quad (8.167)$$

Hence, $\Gamma_{00} = \begin{pmatrix} 1 \\ 1 \end{pmatrix} = 1+K$, $\Gamma_{10} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} = 0$, $\Gamma_{01} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} = 2K$ and $\Gamma_{11} = \begin{pmatrix} 3 \\ 0 \end{pmatrix} = 3$. □

8.8.10 Proof of Lemma 162

Here we sketch the proof of Lemma 162 following Chapter 4.5 of [180]. The first thing to note is that an orthogonal gate W on an d -dimensional system (i.e. represented as a $d \times d$ matrix) can be broken down into a product of at most $d(d-1)/2$ “two-level” orthogonal gates. A two-level orthogonal gate V is one with non-zero off-diagonal entries in at most two rows and the corresponding two columns.

To show this, we write $W = (w_{ij})$ with matrix elements w_{ij} for $i, j \in \{1, 2, \dots, d\}$. We note that for any i, j with $i > j$, we can find a two-level orthogonal gate V such that $(VW)_{ij} = 0$. This is done by choosing V such that $V_{ab} = 0$ except for (letting $N_{ij} = \sqrt{w_{jj}^2 + w_{ji}^2}$)

$$V_{jj} = w_{jj}/N_{ij}, \quad V_{ji} = w_{ij}/N_{ij} \quad (8.168)$$

$$V_{ij} = w_{ij}/N_{ij}, \quad V_{ii} = -w_{jj}/N_{ij} \quad (8.169)$$

and $V_{aa} = 1$ whenever $a \notin \{i, j\}$. One can check V is orthogonal and it is clearly

two-level. Calculating the (i, j) element of VW we get

$$(VW)_{ij} = \sum_h V_{ih}W_{hj} = V_{ii}W_{ij} + V_{ij}W_{jj} = (-w_{jj}w_{ij} + w_{ij}w_{jj})/N_{ij} = 0. \quad (8.170)$$

Incidentally, the (j, j) element of VW is unity

$$(VW)_{jj} = \sum_h V_{jh}W_{hj} = V_{ji}W_{ij} + V_{jj}W_{jj} = (w_{ji}^2 + w_{jj}^2)/N_{ij} = 1. \quad (8.171)$$

Meanwhile, other elements in column j are unchanged, $(VW)_{aj} = W_{aj}$ for $a \notin \{i, j\}$. Also, if $W_{ib} = W_{jb} = 0$ for $b < j < i$, then $(VW)_{ab} = W_{ab}$ for any a as well.

Repeating this reduction with two-level orthogonal gates V_1, V_2, \dots, V_{d-1} , we can zero all off-diagonal elements in the first column of $V_{d-1}V_{d-2}\dots V_1W$. Since this product is still orthogonal, all off-diagonal elements of the first row are also zeroed. Now the same process can be repeated on the remaining block matrix in rows and columns 2 through d , and so on until only a 2×2 block in the lower right remains. The inverse of this remaining matrix is two-level. In summary, we obtain two-level orthogonals V_j such that

$$V_k V_{k-1} \dots V_1 W = I \quad (8.172)$$

and thus $W = V_1^T V_2^T \dots V_k^T$ is a product of two-level orthogonals. Also, k is at most $(d-1) + (d-2) + \dots + 1 = d(d-1)/2$.

In the remainder of the proof we need to write two-level orthogonals as a product of $C^h Z$ gates and arbitrary single-qubit orthogonal gates. We note that $C^h X$ (which equals $C^h Z$ up to Hadamards, which are orthogonal, on the target) can swap two n -qubit basis states $|x\rangle$ and $|y\rangle$ when they differ in at most one place. Let $x_j \oplus y_j = 1$ but $x_i \oplus y_i = 0$ for $i \neq j$. Then, letting C be the $C^{n-1}X$ gate with target j and controls on all the remaining $n-1$ qubits, we find

$$\left(\bigotimes_{i=1}^n X^{1-x_i} \right) C \left(\bigotimes_{i=1}^n X^{1-x_i} \right) : \quad |x\rangle \mapsto |y\rangle \quad (8.173)$$

$$|y\rangle \mapsto |x\rangle \quad (8.174)$$

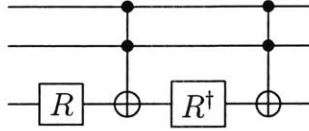


Figure 8-5: The circuit, built from only orthogonal gates, implementing a $C^{n-1}V$ gate with two controls (the top two qubits), corresponding to $n = 3$. Further controls are added in the natural way, extending the CZ and CCX gates to include the new qubits as controls too.

$$|z\rangle \mapsto |z\rangle, \quad z \neq x, y. \quad (8.175)$$

Say we have a two-level orthogonal V acting non-trivially on $|x_1\rangle$ and $|x_2\rangle$. On those two states V applies an orthogonal operator, call it \tilde{V} . First, we can swap $|x_1\rangle$ with other basis states in sequence y_1, y_2, \dots, y_m , where subsequent states differ in at most one bit, so that y_m differs in one bit, bit j , from x_2 . Now, applying $C^{n-1}\tilde{S}$ with target j and controls the remaining $n - 1$ qubits and reversing the swaps from the first step, we implement the two-level orthogonal gate V .

The last step is breaking down $C^{n-1}\tilde{V}$ into C^hX gates and single-qubit rotations. Since \tilde{V} is orthogonal and single-qubit, we can write without loss of generality

$$\tilde{V} = \cos\theta - i\sin\theta Y. \quad (8.176)$$

Define $R = \cos(\theta/2) - i\sin(\theta/2)Y$, such that $RR^\dagger = I$ and $RXR^\dagger X = R^2 = \tilde{V}$. The circuit in Fig. 8-5 implements $C^{n-1}\tilde{V}$.

Finally, we note that this entire process breaks an arbitrary n -qubit orthogonal gate into at most $(n+1)2^{n-1}(2^n - 1)C^hZ$ gates, giving an upper bound on the number of K_L gates that may need to be part of the \mathbb{R} -linear compilation in Eq. (8.108).

8.8.11 Pauli sets with different allowed phases

Our definition of the standard Pauli group $\mathcal{C}_n(1)$ in Eq. (8.113) differs from the Pauli group $G_n(1)$ in Eq. (8.115) in terms of the *allowed phases* of the Pauli operators: while the Pauli operators in $\mathcal{C}_n(1)$ are allowed to have arbitrary phases, those in $G_n(1)$ have

phases which are constrained to be powers of i . In this section, we show that choosing different allowed phases has little effect on the definition of higher levels of the Clifford hierarchy.

We start by introducing some definitions. Let

$$P_n = \{I, X, Y, Z\}^{\otimes n} = \{p_1 \otimes p_2 \otimes \dots \otimes p_n : p_j \in \{I, X, Y, Z\} \text{ for all } j \in 1, 2, \dots, n\} \quad (8.177)$$

denote the set of Pauli operators without any phases.

Definition 191. Let $\Omega \subseteq \mathbb{R}$. The *first level of the Ω -Clifford hierarchy* (called the *Ω -Pauli set*) is

$$\mathcal{C}_n(1; \Omega) = \{e^{i\alpha} p : p \in P_n, \alpha \in \Omega\}. \quad (8.178)$$

For $k \geq 2$, the *k th level of the Ω -Clifford hierarchy* is

$$\mathcal{C}_n(k; \Omega) = \{U \in U_n : U\mathcal{C}_n(1; \Omega)U^\dagger \subseteq \mathcal{C}_n(k-1; \Omega)\}. \quad (8.179)$$

Note that Eq. (8.178) generalizes the Pauli groups discussed above: $\mathcal{C}_n(1) = \mathcal{C}_n(1; \mathbb{R})$ and $G_n(1) = \mathcal{C}_n(1; \frac{\pi}{2}\mathbb{Z})$. Also, $P_n = \mathcal{C}_n(1; 2\pi\mathbb{Z})$. Note that $\mathcal{C}_n(1; \Omega)$ is in general not a group, and hence we refer to it as a *Pauli set* and not a *Pauli group*.

An immediate consequence of the definition of the Ω -Clifford hierarchy is that it is closed under multiplication by global phases:

Theorem 192. Let $k \geq 2$. If $\theta \in \mathbb{R}$ and $V \in \mathcal{C}_n(k; \Omega)$, then $e^{i\theta}V \in \mathcal{C}_n(k; \Omega)$.

For $k \geq 2$, how does the set $\mathcal{C}_n(k; \Omega)$ depend on Ω ? In the rest of this section, we will show that as long as Ω is nonempty and π -periodic (to be defined next), the set $\mathcal{C}_n(k; \Omega)$ is in fact independent of Ω and is equal to the standard Clifford hierarchy $\mathcal{C}_n(k)$.

Definition 193. A subset $\Omega \subseteq \mathbb{R}$ is π -periodic if for all $\alpha \in \Omega$,

$$\alpha \in \Omega \iff \alpha + \pi \in \Omega. \quad (8.180)$$

Note that while the subsets \mathbb{R} and $\frac{\pi}{2}$ are π -periodic, the subset $2\pi\mathbb{Z}$ is not.

Theorem 194. Let $\Omega_1, \Omega_2 \subseteq \mathbb{R}$ be nonempty π -periodic sets. Then for all $k \geq 2$,

$$\mathcal{C}_n(k; \Omega_1) = \mathcal{C}_n(k; \Omega_2). \quad (8.181)$$

Proof. To prove Eq. (8.181), it suffices to prove that for all $k \geq 2$,

$$\mathcal{C}_n(k; \Omega_1) \subseteq \mathcal{C}_n(k; \Omega_2), \quad (8.182)$$

since, by symmetry, interchanging the roles of Ω_1 and Ω_2 will give the opposite inclusion. We shall proceed by induction on k .

We start with the base case $k = 2$. Let $V \in \mathcal{C}_n(2; \Omega_1)$. Then

$$V\mathcal{C}_n(1; \Omega_1)V^\dagger \subseteq \mathcal{C}_n(1; \Omega_1). \quad (8.183)$$

Our goal is to show that

$$V\mathcal{C}_n(1; \Omega_2)V^\dagger \subseteq \mathcal{C}_n(1; \Omega_2). \quad (8.184)$$

To this end, pick an arbitrary element $e^{i\alpha}p \in \mathcal{C}_n(1; \Omega_2)$, where $\alpha \in \Omega_2$ and $p \in P_n$. Since Ω_1 is nonempty, there exists $\beta \in \Omega_1$ such that $e^{i\beta}p \in \mathcal{C}_n(1; \Omega_1)$. By Eq. (8.183), there exist $\gamma \in \Omega_1$ and $q \in P_n$ such that

$$V(e^{i\beta}p)V^\dagger = e^{i\gamma}q, \quad (8.185)$$

which implies that

$$e^{i(\beta-\gamma)}VpV^\dagger = q. \quad (8.186)$$

Taking squares on both sides, and using the property that $p^2 = q^2 = 1$, we get

$$e^{2i(\beta-\gamma)} = 1, \quad (8.187)$$

which implies that

$$e^{i\gamma} = \pm e^{i\beta}. \quad (8.188)$$

Substituting this into Eq. (8.185), we get

$$VpV^\dagger = \pm q. \quad (8.189)$$

Therefore,

$$\begin{aligned} V(e^{i\alpha}p)V^\dagger &= \pm e^{i\alpha}q \\ &= e^{i\alpha}q \text{ or } e^{i(\alpha+\pi)}q \\ &\in \mathcal{C}_n(1; \Omega_2), \end{aligned} \quad (8.190)$$

where in the last step, we used the π -periodicity of Ω_2 to obtain the inclusion $\alpha + \pi \in \Omega_2$.

Since $e^{i\alpha}p \in \mathcal{C}_n(1; \Omega_2)$ was chosen arbitrarily, we obtain Eq. (8.184), which shows that $V \in \mathcal{C}_n(2; \Omega_2)$. Hence, $\mathcal{C}_n(2; \Omega_1) \subseteq \mathcal{C}_n(2; \Omega_2)$.

Next, we prove the inductive step. Assume that Eq. (8.182) holds for k . Let $V \in \mathcal{C}_n(k+1; \Omega_1)$. Pick an arbitrary element $e^{i\alpha}p \in \mathcal{C}_n(1; \Omega_2)$, where $\alpha \in \Omega_2$ and $p \in P_n$. As above, there exists $\beta \in \Omega_1$ such that $e^{i\beta}p \in \mathcal{C}_n(1; \Omega_1)$. Then,

$$\begin{aligned} V(e^{i\alpha}p)V^\dagger &= e^{i(\alpha-\beta)}V(e^{i\beta}p)V^\dagger \\ &\in e^{i(\alpha-\beta)}\mathcal{C}_n(k; \Omega_1) \\ &= \mathcal{C}_n(k; \Omega_1), \quad \text{by Theorem 192} \\ &\subseteq \mathcal{C}_n(k; \Omega_2), \quad \text{by induction hypothesis.} \end{aligned} \quad (8.191)$$

Hence, $V \in \mathcal{C}_n(k+1; \Omega_2)$. This implies that $\mathcal{C}_n(k+1; \Omega_1) \subseteq \mathcal{C}_n(k+1; \Omega_2)$, which completes the proof. \square

Corollary 195. Let $\Omega \subseteq \mathbb{R}$ be a nonempty π -periodic set. Then for all $k \geq 2$,

$$\mathcal{C}_n(k; \Omega) = \mathcal{C}_n(k) = G_n(k), \quad (8.192)$$

i.e. $\mathcal{C}_n(k; \Omega)$ is independent of Ω .

Proof. Since $\mathcal{C}_n(1) = \mathcal{C}_n(1; \mathbb{R})$ and $G_n(1) = \mathcal{C}_n(1; \frac{\pi}{2}\mathbb{Z})$, and \mathbb{R} and $\frac{\pi}{2}\mathbb{Z}$ are both nonempty and π -periodic, Corollary 195 follows directly from Theorem 194. \square

Appendix A

Characterizations of the Clifford group

In this appendix, we will describe various characterizations of the Clifford group that are useful in this thesis. The results here may be compared with the results in Chapter 8, where a rebit analogue of the Clifford group is developed. One of the main results we will prove here is that the Clifford group is generated by the H , S and CZ gates. Our proof of this statement follows an argument similar to the inductive argument given by Gottesman (for example, see page 13 of [110] and sections 5.6 and 5.8 of [109]). This appendix is an elaboration of some of the results stated in Section 2.5.1.

We begin with an axiomatic characterization of a group that we denote by $\tilde{\mathcal{C}}_n$. We will show later (Corollary 215) that $\tilde{\mathcal{C}}_n$ is isomorphic to the Clifford group $\mathcal{C}_n/\mathcal{U}(1)$.

Definition 196. Let $\tilde{\mathcal{C}}_n$ be the set of functions $f : \overline{\mathcal{P}}_n \rightarrow \overline{\mathcal{P}}_n$ satisfying

1. Unitality: $f(I) = I$.
2. (Anti-)commutation preserving: for all $p, q \in \overline{\mathcal{P}}_n$,

$$[p, q]_{\pm} = 0 \implies [f(p), f(q)]_{\pm} = 0.$$

3. Homogeneity: for all $p \in \overline{\mathcal{P}}_n$ and $k \in \mathbb{N}$, $f(i^k p) = i^k f(p)$.
4. Multiplicativity: for all $p, q \in \overline{\mathcal{P}}_n$, $f(pq) = f(p)f(q)$.

5. Hermiticity: for all $p \in \mathcal{P}_n^*$, $f(p)^\dagger = f(p)$.

where $[A, B]_\pm = AB \pm BA$ is the anticommutator/commutator.

First, we show that if U is a Clifford operation, then the map $P \mapsto UPU^\dagger$ is in $\tilde{\mathcal{C}}_n$.

Proposition 197. Let $U \in \mathcal{C}_n$. Then $U(\cdot)U^\dagger \in \tilde{\mathcal{C}}_n$.

Proof. It is straightforward to check that $U(\cdot)U^\dagger$ satisfies the properties listed in Definition 196:

1. Unitality: $UIU^\dagger = UU^\dagger = I$.
2. (Anti-)commutation preserving: Let $[p, q]_\pm = 0$. Then, $[UpU^\dagger, UqU^\dagger]_\pm = U[p, q]_\pm U^\dagger = 0$.
3. Homogeneity: $U(i^k p)U^\dagger = i^k UpU^\dagger$.
4. Multiplicativity: $U(pq)U^\dagger = UpU^\dagger UqU^\dagger$.
5. Hermiticity: $(UpU^\dagger)^\dagger = UpU^\dagger$.

□

Next, we give an alternative characterization of $\tilde{\mathcal{C}}_n$ in terms of the action of its elements on the Pauli operators X_i and Z_i .

Proposition 198. Let $f : \overline{\mathcal{P}}_n \rightarrow \overline{\mathcal{P}}_n$. Then $f \in \tilde{\mathcal{C}}_n$ if and only if

- (A) $f(i^k X_1^{u_1} Z_1^{v_1} \dots X_n^{u_n} Z_n^{v_n}) = i^k f(X_1)^{u_1} f(Z_1)^{v_1} \dots f(X_n)^{u_n} f(Z_n)^{v_n}$ for all $k \in \mathbb{N}$ and $u_i, v_i \in \mathbb{F}_2$.
- (B) $[f(X_i), f(X_j)] = [f(Z_i), f(Z_j)] = 0$ for all i, j .
 $[f(X_i), f(Z_j)] = 0$ for all $i \neq j$.
 $\{f(X_i), f(Z_i)\} = 0$ for all i .
- (C) $f(X_i)^\dagger = f(X_i)$ and $f(Z_i)^\dagger = f(Z_i)$ for all i .

Proof.

(\Leftarrow) Let f satisfy (A), (B) and (C). We'll show that f satisfies properties (1)–(5) of Definition 196.

1. Taking $k = u_1 = v_1 = \dots = u_n = v_n = 0$ in (A) gives $f(I) = I$.

2. Let

$$p = i^k X^u Z^v = i^k X_1^{u_1} Z_1^{v_1} \dots X_n^{u_n} Z_n^{v_n} \in \overline{\mathcal{P}}_n$$

and

$$q = i^l X^x Z^z = i^l X_1^{x_1} Z_1^{z_1} \dots X_n^{x_n} Z_n^{z_n} \in \overline{\mathcal{P}}_n.$$

Assume that $[p, q]_{\pm} = 0$. Then, by Eqs. 2.21 and 2.26, $(-1)^{u \cdot z + x \cdot v} = \mp 1$, which implies that

$$\begin{aligned} [f(p), f(q)]_{\pm} &= [f(i^k X_1^{u_1} Z_1^{v_1} \dots X_n^{u_n} Z_n^{v_n}), f(i^l X_1^{x_1} Z_1^{z_1} \dots X_n^{x_n} Z_n^{z_n})]_{\pm} \\ &= [i^k f(X_1)^{u_1} f(Z_1)^{v_1} \dots f(X_n)^{u_n} f(Z_n)^{v_n}, \\ &\quad i^l f(X_1)^{x_1} f(Z_1)^{z_1} \dots f(X_n)^{x_n} f(Z_n)^{z_n}]_{\pm} \quad , \text{ by (A)} \\ &= [(-1)^{x \cdot v + u \cdot z} \pm 1] f(X_1)^{x_1} f(Z_1)^{z_1} \dots f(X_n)^{x_n} f(Z_n)^{z_n} \\ &\quad \times f(X_1)^{u_1} f(Z_1)^{v_1} \dots f(X_n)^{u_n} f(Z_n)^{v_n} \quad , \text{ by (B)} \\ &= 0. \end{aligned}$$

3. Let $p = i^k X_1^{u_1} Z_1^{v_1} \dots X_n^{u_n} Z_n^{v_n}$. Then,

$$\begin{aligned} f(i^l p) &= f(i^l \cdot i^k X_1^{u_1} Z_1^{v_1} \dots X_n^{u_n} Z_n^{v_n}) \\ &= i^{l+k} f(X_1)^{u_1} f(Z_1)^{v_1} \dots f(X_n)^{u_n} f(Z_n)^{v_n} \\ &= i^l f(p). \end{aligned}$$

4. Let $p = i^k X_1^{u_1} Z_1^{v_1} \dots X_n^{u_n} Z_n^{v_n}$ and $q = i^l X_1^{x_1} Z_1^{z_1} \dots X_n^{x_n} Z_n^{z_n}$. Then,

$$\begin{aligned} f(pq) &= i^{k+l} (-1)^{x_1 v_1 + \dots + x_n v_n} f(X_1^{u_1+x_1}) f(Z_1^{v_1+z_1}) \dots f(X_n^{u_n+x_n}) f(Z_n^{v_n+z_n}) \\ &= i^{k+l} (-1)^{x_1 v_1 + \dots + x_n v_n} f(X_1)^{u_1+x_1} f(Z_1)^{v_1+z_1} \dots f(X_n)^{u_n+x_n} f(Z_n)^{v_n+z_n} \end{aligned}$$

$$= f(p)f(q).$$

5. Let $p \in \mathcal{P}_n^*$. Then $p = P_1^{(1)} \dots P_n^{(n)}$, where $P^{(i)} \in \{i^{u_i v_i} X^{u_i} Z^{v_i} : u_i, v_i \in \mathbb{F}_2\}$.

Then,

$$\begin{aligned} f(p)^\dagger &= f(i^{u_1 v_1 + \dots + u_n v_n} X_1^{u_1} Z_1^{v_1} \dots X_n^{u_n} Z_n^{v_n})^\dagger \\ &= [i^{u_1 v_1 + \dots + u_n v_n} f(X_1)^{u_1} f(Z_1)^{v_1} \dots f(X_n)^{u_n} f(Z_n)^{v_n}]^\dagger \\ &= (-i)^{u_1 v_1 + \dots + u_n v_n} f(Z_n)^{v_n} f(X_n)^{u_n} \dots f(Z_1)^{v_1} f(X_1)^{u_1} \\ &= i^{u_1 v_1 + \dots + u_n v_n} \prod_{i=1}^n (-1)^{u_i v_i} f(Z_i)^{v_i} f(X_i)^{u_i}. \end{aligned}$$

But $(-1)^{uv} f(Z)^v f(X)^u = f(X)^u f(Z)^v$, by (B). Hence,

$$\begin{aligned} f(p)^\dagger &= i^{u_1 v_1 + \dots + u_n v_n} f(X_1)^{u_1} f(Z_1)^{v_1} \dots f(X_n)^{u_n} f(Z_n)^{v_n} \\ &= f(i^{u_1 v_1 + \dots + u_n v_n} X_1^{u_1} Z_1^{v_1} \dots X_n^{u_n} Z_n^{v_n}) \\ &= f(p). \end{aligned}$$

(\Rightarrow) Assume that f satisfies (1)–(5) of Definition 196. We'll show that f satisfies (A), (B) and (C).

1. Let $f \in \tilde{\mathcal{C}}_n$. Then,

$$f(i^k X_1^{u_1} Z_1^{v_1} \dots X_n^{u_n} Z_n^{v_n}) = i^k f(X_1^{u_1}) f(Z_1^{v_1}) \dots f(X_n^{u_n}) f(Z_n^{v_n}),$$

by homogeneity and multiplicativity. Now, by unitality,

$$\begin{aligned} f(X_i^{u_i}) &= \begin{cases} f(I) = I, & u_i = 0 \\ f(X_i), & u_i = 1 \end{cases} \\ &= f(X_i)^{u_i}. \end{aligned}$$

Likewise, $f(Z_i^{v_i}) = f(Z_i)^{v_i}$. Hence,

$$f(i^k X_1^{u_1} Z_1^{v_1} \dots X_n^{u_n} Z_n^{v_n}) = i^k f(X_1)^{u_1} f(Z_1)^{v_1} \dots f(X_n)^{u_n} f(Z_n)^{v_n}.$$

2. By the (anti-)commutation preserving property,

$$\begin{aligned} \forall i, j, \quad [X_i, X_j] = 0 &\implies [f(X_i), f(X_j)] = 0. \\ \forall i, j, \quad [Z_i, Z_j] = 0 &\implies [f(Z_i), f(Z_j)] = 0. \\ \forall i \neq j, \quad [X_i, Z_j] = 0 &\implies [f(X_i), f(Z_j)] = 0. \\ \forall i, \quad \{X_i, Z_i\} = 0 &\implies \{f(X_i), f(Z_i)\} = 0. \end{aligned}$$

3. Since $X_i, Z_i \in \mathcal{P}_n^*$, it follows from Hermiticity that $f(X_i)^\dagger = f(X_i)$ and $f(Z_i)^\dagger = f(Z_i)$.

□

An implication of Proposition 198 is that the functions $f \in \tilde{\mathcal{C}}_n$ are completely determined by the images of X_i and Z_i under f .

Proposition 199. Let $f \in \tilde{\mathcal{C}}_n$. Then for all $P \in \mathcal{P}_n^*$, there exist $a \in \mathbb{F}_2$ and $Q \in \mathcal{P}_n^*$ such that $f(P) = (-1)^a Q$.

Proof. Let $f \in \tilde{\mathcal{C}}_n$, and $P \in \mathcal{P}_n^*$. Since the range of f is in $\overline{\mathcal{P}}_n$, $f(P) = i^b Q$ for some $b \in \mathbb{Z}_4$ and $Q \in \mathcal{P}_n$. By Hermiticity, $f(P)^\dagger = f(P)$, and hence

$$(-i)^b Q^\dagger = i^b Q.$$

Since $Q^\dagger = Q$ is Hermitian, it follows that $(-1)^b = 1$, which implies that $i^b = \pm 1$. So, $f(P) \in \{\pm Q\}$. But $Q \neq I$ (otherwise, $f(P) = \pm I$, which implies that P commutes with all the elements of \mathcal{P}_n^* , which is a contradiction). Hence, there exist $a \in \mathbb{F}_2$ and $Q \in \mathcal{P}_n^*$ such that $f(P) = (-1)^a Q$. □

Taking $P = X_i, Z_i$ in Proposition 199 gives the following corollary.

Corollary 200. Let $f \in \tilde{\mathcal{C}}_n$. Then, $f(X_i) = (-1)^a A^{(i)}$ and $f(Z_i) = (-1)^b B^{(i)}$ for some $a, b \in \mathbb{F}_2$ and $A^{(i)}, B^{(i)} \in \mathcal{P}_n^*$, i.e. $f(X_i), f(Z_i) \in \pm \mathcal{P}_n^*$.

Next, we prove some lemmas that would be useful for finding sets of generators for $\tilde{\mathcal{C}}_n$.

Lemma 201. Let $\mathcal{U} \in \tilde{\mathcal{C}}_n$ and $s, t \in \mathbb{F}_2^n$. Let

$$\begin{aligned} \mathcal{V} : \overline{\mathcal{P}}_n &\rightarrow \overline{\mathcal{P}}_n \\ P &\mapsto \mathcal{U}(X^s Z^t P Z^t X^s). \end{aligned}$$

Then,

$$\begin{aligned} \mathcal{V}(X_i) &= (-1)^{t_i} \mathcal{U}(X_i), \\ \mathcal{V}(Z_i) &= (-1)^{s_i} \mathcal{U}(Z_i). \end{aligned}$$

Proof. By straightforward calculation,

$$\begin{aligned} \mathcal{V}(X_i) &= \mathcal{U}(X^s Z^t X_i Z^t X^s) \\ &= \mathcal{U}(X^s (-1)^{t_i} X_i X^s) \\ &= \mathcal{U}((-1)^{t_i} X_i) \\ &= (-1)^{t_i} \mathcal{U}(X_i) \quad \text{by homogeneity.} \end{aligned}$$

Similarly,

$$\begin{aligned} \mathcal{V}(Z_i) &= \mathcal{U}(X^s Z^t Z_i Z^t X^s) \\ &= \mathcal{U}(X^s Z_i X^s) \\ &= \mathcal{U}((-1)^{s_i} Z_i) \\ &= (-1)^{s_i} \mathcal{U}(Z_i) \quad \text{by homogeneity.} \end{aligned}$$

□

Lemma 202. Let A, B be matrices over \mathbb{C} . If $A \otimes B = I$, then there exists $a \in \mathbb{C} \setminus \{0\}$

such that

$$A = aI, \tag{A.1}$$

$$B = \frac{1}{a}I. \tag{A.2}$$

Proof. The condition $A \otimes B = I$ is equivalent to the equations

$$a_{ii}B = I, \quad \forall i \tag{A.3}$$

$$a_{ij}B = 0, \quad \forall i \neq j \tag{A.4}$$

From Eq. (A.3), $B \neq 0$. By Eq. (A.4), $a_{ij} = 0$ for all $i \neq j$. Hence,

$$a_{11}B = a_{22}B = \dots = a_{nn}B.$$

Since $B \neq 0$,

$$a_{11} = a_{22} = \dots = a_{nn} := a.$$

Hence, $A = aI$. By Eq. (A.3), $aB = I$, which implies that

$$B = \frac{1}{a}I.$$

□

Next, we show that the operators $U(\cdot)U^\dagger$ and $V(\cdot)V^\dagger$ are equal if and only if $U \sim V$, i.e. U and V differ by a global phase.

Lemma 203. Let $U, V \in \mathcal{U}(2^n)$. Then,

$$U \sim V \iff UPU^\dagger = VPV^\dagger \quad \forall P \in \mathcal{P}_n^*. \tag{A.5}$$

Proof.

(\Rightarrow) If $U \sim V$, then $U = e^{i\theta}V$ for some $\theta \in \mathbb{R}$. Hence,

$$UPU^\dagger = (e^{i\theta}V)P(e^{i\theta}V)^\dagger = VPV^\dagger.$$

(\Leftarrow) Assume that $UPU^\dagger = VPV^\dagger$ for all $P \in \mathcal{P}_n^*$. Then,

$$\text{vec}(UPU^\dagger) = \text{vec}(VPV^\dagger) \quad \text{for all } P \in \mathcal{P}_n^*, \quad (\text{A.6})$$

where $\text{vec} : |u\rangle\langle v| \mapsto |u\rangle \otimes |v\rangle$ is vectorization operation that transforms matrices to column vectors. Applying the identity $\text{vec}(ABC) = (A \otimes C^T)\text{vec}(B)$ (see Eq. 1.132 of [228]) to Eq. (A.6) gives

$$(U \otimes \bar{U})\text{vec}(P) = (V \otimes \bar{V})\text{vec}(P) \quad \text{for all } P \in \mathcal{P}_n^*. \quad (\text{A.7})$$

Also, since U and V are unitary, $UIU^\dagger = VIV^\dagger$, which implies that

$$(U \otimes \bar{U})\text{vec}(I) = (V \otimes \bar{V})\text{vec}(I). \quad (\text{A.8})$$

Combining Eqs. (A.7) and (A.8) gives

$$(U \otimes \bar{U})\text{vec}(P) = (V \otimes \bar{V})\text{vec}(P) \quad \text{for all } P \in \mathcal{P}_n. \quad (\text{A.9})$$

Now, $\{\text{vec}(P) : P \in \mathcal{P}_n\}$ forms a basis for $\mathcal{L}(2^n)$. Hence,

$$U \otimes \bar{U} = V \otimes \bar{V}, \quad (\text{A.10})$$

which implies that

$$UV^\dagger \otimes \overline{UV^\dagger} = I. \quad (\text{A.11})$$

By Lemma 202, there exists $a \in \mathbb{C} \setminus \{0\}$ such that

$$UV^\dagger = aI, \quad \overline{UV^\dagger} = \frac{1}{a}I, \quad (\text{A.12})$$

which implies that

$$\bar{a} = \frac{1}{a}.$$

Hence, $|a|^2 = 1$, i.e. $a = e^{-i\theta}$ for some $\theta \in \mathbb{R}$. Hence, $UV^\dagger = e^{i\theta}I$, which implies that $U = e^{i\theta}V$, i.e. $U \sim V$.

□

Next, we show that if U is generated by a circuit of H , S and CZ gates, then $U(\cdot)U^\dagger$ is a subset of $\tilde{\mathcal{C}}_n$.

Proposition 204. $\{U(\cdot)U^\dagger : \overline{\mathcal{P}}_n \rightarrow \overline{\mathcal{P}}_n | U \in \langle H, S, CZ \rangle^n\} \subseteq \tilde{\mathcal{C}}_n$.

Proof. By Eqs. (2.43), (2.44) and (2.47), $H, S, CZ \in \mathcal{C}_n$. Since \mathcal{C}_n is a group, it is closed under multiplication. Also, it is easy to check that \mathcal{C}_n is closed under taking tensor products. Hence,

$$\langle H, S, CZ \rangle^n \subseteq \mathcal{C}_n. \quad (\text{A.13})$$

Let

$$U(\cdot)U^\dagger \in \{U(\cdot)U^\dagger : \overline{\mathcal{P}}_n \rightarrow \overline{\mathcal{P}}_n | U \in \langle H, S, CZ \rangle^n\}.$$

Hence, there exists $V \in \langle H, S, CZ \rangle^n$ such that $V(\cdot)V^\dagger = U(\cdot)U^\dagger$. By Eq. (A.13), $V \in \mathcal{C}_n$. By Proposition 197, $V(\cdot)V^\dagger \in \tilde{\mathcal{C}}_n$. Hence, $U(\cdot)U^\dagger = V(\cdot)V^\dagger \in \tilde{\mathcal{C}}_n$. □

A.1 The single-qubit case

In the next few results (Lemmas 205–208), we will study the $n = 1$ case, and find a generating set for $\tilde{\mathcal{C}}_1$.

Lemma 205. Let $f \in \tilde{C}_1$. Then there exist $a, c \in \mathbb{F}_2$ and distinct $i, j, k \in \{1, 2, 3\}$ such that

$$f(X) = (-1)^a \sigma_i, \quad (\text{A.14})$$

$$f(Y) = (-1)^b \sigma_j, \quad (\text{A.15})$$

$$f(Z) = (-1)^c \sigma_k, \quad (\text{A.16})$$

where

$$b = \begin{cases} 0 & \text{if } i(-1)^{a+c} \sigma_i \sigma_k \sigma_j = I, \\ 1 & \text{if } i(-1)^{a+c} \sigma_i \sigma_k \sigma_j = -I. \end{cases} \quad (\text{A.17})$$

Proof. By Proposition 199,

$$f(X) = (-1)^a \sigma_i, \quad (\text{A.18})$$

$$f(Y) = (-1)^b \sigma_j, \quad (\text{A.19})$$

$$f(Z) = (-1)^c \sigma_k, \quad (\text{A.20})$$

for some $a, b, c \in \mathbb{F}_2$ and $\sigma_i, \sigma_j, \sigma_k \in \mathcal{P}_n^*$. Since X, Y, Z mutually anticommute with each other, by the anticommutation-preserving property of \tilde{C}_n , $\sigma_i, \sigma_j, \sigma_k$ mutually anticommute with each other. Hence, $i, j, k \in \{1, 2, 3\}$ must be distinct. Now,

$$\begin{aligned} (-1)^b \sigma_j &= f(Y) \\ &= f(iXZ) \\ &= if(X)f(Z) \quad \text{by homogeneity and multiplicativity} \\ &= i(-1)^a \sigma_i (-1)^c \sigma_k \\ &= i(-1)^{a+c} \sigma_i \sigma_k. \end{aligned}$$

Hence,

$$(-1)^b I = i(-1)^{a+c} \sigma_i \sigma_k \sigma_j,$$

which implies that

$$b = \begin{cases} 0 & \text{if } i(-1)^{a+c}\sigma_i\sigma_k\sigma_j = I, \\ 1 & \text{if } i(-1)^{a+c}\sigma_i\sigma_k\sigma_j = -I. \end{cases} \quad (\text{A.21})$$

□

Let $\mathcal{S}_3 = \{(i, j, k) \in \{1, 2, 3\}^3 : i, j, k \text{ distinct}\}$ be the symmetric group of degree 3. In the proof of the next lemma, we will make use of *cycle decomposition notation* (see Chapter 1.3 of [85]). In this notation, the elements of \mathcal{S}_3 are $1, (12), (13), (23) = (12)(13)(12), (123) = (13)(12), (132) = (12)(13)$. Here, the cycle $(a_0, a_1, \dots, a_{s-1})$ refers to the permutation that sends a_i to $a_{i \pmod s}$, for $i \in \mathbb{Z}_s$.

Lemma 206. Let $(i, j, k) \in \mathcal{S}_3$. Then there exist $U \in \langle H, S \rangle$ and $a, b, c \in \mathbb{F}_2$ such that

$$\begin{aligned} UXU^\dagger &= (-1)^a \sigma_i, \\ UYU^\dagger &= (-1)^b \sigma_j, \\ UZU^\dagger &= (-1)^c \sigma_k. \end{aligned}$$

Remark. The idea behind the proof is that H swaps X and Z , and S swaps X and Y . Since any permutation is a product of swaps, any permutation can be achieved by products of S and H . This may be written more formally as follows.

Proof. We use the fact that $\mathcal{S}_3 = \langle (12), (13) \rangle$, i.e. for all $\varpi \in \mathcal{S}_3$, $\varpi = \prod_i g_i$ for some $g_i \in \{(12), (13)\}$. Set $U_{(12)} = S$ and $U_{(13)} = H$ and

$$U_\varpi := U_{\prod_i g_i} = \prod_i U_{g_i}$$

We claim that $U_\varpi \sigma_k U_\varpi^\dagger \in \{\pm \sigma_{\varpi(k)}\}$. To see this, note that when $\varpi = (12)$,

$$\begin{aligned} U_\varpi \sigma_1 U_\varpi^\dagger &= SXS^\dagger = Y = \sigma_2 = \sigma_{\varpi(1)}, \\ U_\varpi \sigma_2 U_\varpi^\dagger &= SYS^\dagger = -X = -\sigma_1 = -\sigma_{\varpi(2)}, \\ U_\varpi \sigma_3 U_\varpi^\dagger &= SZS^\dagger = Z = \sigma_3 = \sigma_{\varpi(3)}, \end{aligned}$$

and when $\varpi = (13)$,

$$\begin{aligned} U_{\varpi}\sigma_1U_{\varpi}^{\dagger} &= HXH = Z = \sigma_3 = \sigma_{\varpi(1)}, \\ U_{\varpi}\sigma_2U_{\varpi}^{\dagger} &= HYH = -Y = -\sigma_2 = -\sigma_{\varpi(2)}, \\ U_{\varpi}\sigma_3U_{\varpi}^{\dagger} &= HZH = x = \sigma_1 = \sigma_{\varpi(3)}. \end{aligned}$$

Finally, when $\varpi = g_1 \dots g_s$,

$$\begin{aligned} U_{\varpi}\sigma_kU_{\varpi}^{\dagger} &= U_{g_1 \dots g_s}\sigma_kU_{g_1 \dots g_s}^{\dagger} \\ &= U_{g_1} \dots U_{g_{s-1}}(U_{g_s}\sigma_kU_{g_s}^{\dagger})U_{g_{s-1}}^{\dagger} \dots U_{g_1}^{\dagger} \\ &= U_{g_1} \dots U_{g_{s-1}}(\pm\sigma_{g_s(k)})U_{g_{s-1}}^{\dagger} \dots U_{g_1}^{\dagger} \\ &= U_{g_1} \dots (\pm\sigma_{g_{s-1}g_s(k)}) \dots U_{g_1}^{\dagger} \\ &\quad \vdots \\ &\in \{\pm\sigma_{g_1g_2 \dots g_s(k)}\} \\ &= \{\pm\sigma_{\varpi(k)}\}, \end{aligned}$$

where the vertical ellipsis describes an inductive process, in which we replace

$$U_{g_{s-u}}(\pm\sigma_{g_{s-u+1} \dots g_s(k)})U_{g_{s-u}}^{\dagger}$$

with $\pm\sigma_{g_{s-u} \dots g_s(k)}$, for $u = 0, 1, \dots, s-1$.

This completes the proof of the claim $U_{\varpi}\sigma_kU_{\varpi}^{\dagger} \in \{\pm\sigma_{\varpi(k)}\}$, and also the lemma. \square

Lemma 207. Let $(i, j, k) \in \mathcal{S}_3$ and $a, c \in \mathbb{F}_2$. Then there exists $U \in \langle H, S \rangle$ such that

$$UXU^{\dagger} = (-1)^a \sigma_i, \tag{A.22}$$

$$UZU^{\dagger} = (-1)^c \sigma_k. \tag{A.23}$$

Remark. Note that this implies that $UYU^{\dagger} = (-1)^b \sigma_j$, where b is given by Eq. (A.17).

Proof. Let $(i, j, k) \in \mathcal{S}_3$. By Lemma 206, there exist $V \in \langle H, S \rangle \subseteq \mathcal{C}_1$ and $e, f \in \mathbb{F}_2$ such that

$$\begin{aligned} VXV^\dagger &= (-1)^e \sigma_i, \\ VZV^\dagger &= (-1)^f \sigma_k. \end{aligned}$$

Set $U = VX^{f+c}Z^{e+a}$. By Eq. (2.49), $X, Z \in \langle H, S \rangle$, and hence $U \in \langle H, S \rangle \subseteq \mathcal{C}_1$. By Lemma 201,

$$\begin{aligned} UXU^\dagger &= (-1)^{e+a} VXV^\dagger = (-1)^a \sigma_i, \\ UZU^\dagger &= (-1)^{f+c} VZV^\dagger = (-1)^c \sigma_k. \end{aligned}$$

□

Lemma 208.

$$\tilde{\mathcal{C}}_1 = \{U(\cdot)U^\dagger : \overline{\mathcal{P}}_1 \rightarrow \mathcal{P}_1 | U \in \langle H, S \rangle\}.$$

Proof. The backward inclusion (\supseteq) was proved in Proposition 204. Hence, it remains to prove the forward inclusion (\subseteq). Let $f : \overline{\mathcal{P}}_1 \rightarrow \overline{\mathcal{P}}_1 \in \tilde{\mathcal{C}}_1$. By Lemma 205, there exist $a, c \in \mathbb{F}_2$ and $(i, j, k) \in \mathcal{S}_3$ such that

$$\begin{aligned} f(X) &= (-1)^a \sigma_i, \\ f(Z) &= (-1)^c \sigma_k. \end{aligned}$$

By Lemma 207, there exists $U \in \langle H, S \rangle$ such that

$$UXU^\dagger = (-1)^a \sigma_i, \tag{A.24}$$

$$UZU^\dagger = (-1)^c \sigma_k. \tag{A.25}$$

Hence, $f(X) = UXU^\dagger$ and $f(Z) = UZU^\dagger$. By Proposition 198, f is completely

determined by $f(X)$ and $f(Y)$. Indeed, for all $i^k X^u Z^v \in \overline{\mathcal{P}}_1$,

$$\begin{aligned} f(i^k X^u Z^v) &= i^k f(X)^u f(Z)^v \\ &= i^k (UXU^\dagger)^u (UZU^\dagger)^v \\ &= U(i^k X^u Z^v)U^\dagger. \end{aligned}$$

Hence, $f(\cdot) = U(\cdot)U^\dagger$ for some $U \in \langle H, S \rangle$. □

A.2 The multi-qubit case

Next, we consider the $n > 1$ case, and generalize Lemma 208 to find a generating set for $\widetilde{\mathcal{C}}_n$.

Lemma 209. Let $n > 1$. Assume that $p, q \in \mathcal{P}_n$ anticommute. Then there exist $U \in \langle \text{SWAP}, H, S \rangle^n$ and $p', q' \in \mathcal{P}_{n-1}$, where $[p', q'] = 0$, such that

$$\begin{aligned} U : p &\rightarrow X \otimes p', \\ q &\rightarrow Z \otimes q'. \end{aligned}$$

Proof. Write $p = p^{(1)} \otimes \dots \otimes p^{(n)}$ and $q = q^{(1)} \otimes \dots \otimes q^{(n)}$. Since $\{p, q\} = 0$, there exists i such that $p^{(i)} \neq q^{(i)}$. (To see this, suppose that $p^{(i)} = q^{(i)}$ for all i . Then, $p = q \implies [p, q] = 0 \implies \{p, q\} \neq 0$ since the Paulis either commute or anticommute, which is a contradiction.)

Let

$$\chi_{1i} = \begin{cases} I, & i = 1, \\ \text{SWAP}_{1i}, & i \neq 1. \end{cases}$$

Then,

$$\begin{aligned} \chi_{1i} p \chi_{1i}^\dagger &= p_1^{(i)} p_2^{(2)} \dots p_i^{(1)} \dots p_n^{(n)} = p^{(i)} \otimes p', \\ \chi_{1i} q \chi_{1i}^\dagger &= q_1^{(i)} q_2^{(2)} \dots q_i^{(1)} \dots q_n^{(n)} = q^{(i)} \otimes q'. \end{aligned}$$

By Lemma 207, there exists $V_1 \in \langle H, S \rangle$ such that

$$\begin{aligned} V_1 X_1 V_1^\dagger &= p_1^{(i)}, \\ V_1 Z_1 V_1^\dagger &= q_1^{(i)}. \end{aligned}$$

Hence, $W_1 = V_1^\dagger \in \langle H, S \rangle$ satisfies

$$\begin{aligned} W_1 p_1^{(i)} W_1^\dagger &= X_1, \\ W_1 q_1^{(i)} W_1^\dagger &= Z_1. \end{aligned}$$

Set $U = W_1 \chi_{1i} \in \langle \text{SWAP}, H, S \rangle^n$. Then

$$\begin{aligned} UpU^\dagger &= W_1 \chi_{1i} p \chi_{1i}^\dagger W_1^\dagger \\ &= W_1 (p^{(i)} \otimes p') W_1^\dagger \\ &= X \otimes p', \\ UqU^\dagger &= W_1 \chi_{1i} q \chi_{1i}^\dagger W_1^\dagger \\ &= W_1 (q^{(i)} \otimes q') W_1^\dagger \\ &= Z \otimes q'. \end{aligned}$$

Now,

$$\begin{aligned} \{p, q\} = 0 &\implies \{UpU^\dagger, UqU^\dagger\} = 0 \\ &\implies 0 = \{X \otimes p', Z \otimes q'\} \\ &= XZ \otimes p'q' + ZX \otimes q'p' \\ &= XZ \otimes (p'q' - q'p') \\ &\implies q'p' = p'q' \\ &\implies [p', q'] = 0. \end{aligned}$$

□

Before we proceed, we will prove two useful identities regarding controlled-Pauli operators.

Lemma 210. Let $P \in \mathcal{P}_1$ and $\sigma^u, \sigma^v \in \mathcal{P}_n$. Then,

$$\begin{aligned} CP_{12} &: X_1 \rightarrow X_1 P_2, \\ Z_1 &\rightarrow Z_1. \end{aligned} \tag{A.26}$$

and

$$C\sigma_{12}^u : \sigma_2^v \rightarrow Z_1^{\langle u,v \rangle} \sigma_2^v. \tag{A.27}$$

Proof. It is easy to verify for $P = I, X, Y, Z$ that Eq. (A.26) holds. Next, Eq. (A.27) may be shown to hold, by direct calculation:

$$\begin{aligned} C\sigma_{12}^u : \sigma_2^v &\rightarrow C\sigma_{12}^u \sigma_2^v C\sigma_{12}^u \\ &= (|0\rangle\langle 0|I + |1\rangle\langle 0|\sigma^u)(I \otimes \sigma^v)(|0\rangle\langle 0|I + |1\rangle\langle 0|\sigma^u) \\ &= |0\rangle\langle 0|\sigma^v + |1\rangle\langle 0|\sigma^u \sigma^v \sigma^u \\ &= |0\rangle\langle 0|\sigma^v + |1\rangle\langle 0|(-1)^{\langle u,v \rangle} \sigma^v \\ &= (|0\rangle\langle 0| + (-1)^{\langle u,v \rangle} |1\rangle\langle 0|) \otimes \sigma^v \\ &= Z^{\langle u,v \rangle} \otimes \sigma^v. \end{aligned}$$

□

Next, we will introduce some notation. For $P, Q \in \mathcal{P}_{n-1}$, define

$$U[P, Q] = H_1 \left(\prod_{k=1}^n CQ_{1k}^{(k)} \right) H_1 \left(\prod_{k=1}^n CP_{1k}^{(k)} \right). \tag{A.28}$$

By Eq. (2.49), $U[P, Q] \in \langle H, S, CZ \rangle^n$.

Lemma 211. Let $n > 1$. Let $P, Q \in \mathcal{P}_{n-1}$ satisfy $[P, Q] = 0$. Then $U = U[P, Q]$

satisfies

$$U : X_1 \rightarrow X \otimes P,$$

$$Z_1 \rightarrow Z \otimes Q.$$

Proof. Write $P = P^{(2)} \otimes \dots \otimes P^{(n)}$ and $Q = Q^{(2)} \otimes \dots \otimes Q^{(n)}$, where $P^{(i)}, Q^{(i)} \in \mathcal{P}_1$ for all $i = 2, \dots, n$. Then, writing Eq. (A.28) as a product of 4 operators

$$U[P, Q] = \underbrace{H_1}_4 \cdot \underbrace{\left(\prod_{k=1}^n CQ_{1k}^{(k)} \right)}_3 \cdot \underbrace{H_1}_2 \cdot \underbrace{\left(\prod_{k=1}^n CP_{1k}^{(k)} \right)}_1,$$

and using Lemma 210, Z_1 and X_1 evolve as follows:

$$\begin{aligned} Z_1 &\xrightarrow{1} Z_1 \\ &\xrightarrow{2} X_1 \\ &\xrightarrow{3} X_1 Q_2^{(2)} \dots Q_n^{(n)} \\ &\xrightarrow{4} Z \otimes Q, \\ X_1 &\xrightarrow{1} X_1 P_2^{(2)} \dots P_n^{(n)} \\ &\xrightarrow{2} Z_1 P_2^{(2)} \dots P_n^{(n)} \\ &\xrightarrow{3} Z_1 Z_1^{(p,q)} P_2^{(2)} \dots P_n^{(n)} \\ &\xrightarrow{4} X_1 X_1^{(p,q)} P_2^{(2)} \dots P_n^{(n)} \\ &= X^{1+(p,q)} \otimes P, \end{aligned}$$

where p and q are defined by $\sigma^q = Q$ and $\sigma^p = P$. Now, $[P, Q] = 0 \implies \langle p, q \rangle = 0$.

Hence,

$$U : X_1 \rightarrow X \otimes P,$$

$$Z_1 \rightarrow Z \otimes Q.$$

□

Lemma 212. Let $n > 1$. Let $\mathcal{W} \in \tilde{\mathcal{C}}_n$. Assume that

$$\begin{aligned}\mathcal{W} : X_1 &\mapsto X \otimes P, \\ Z_1 &\mapsto Z \otimes Q,\end{aligned}$$

for some $P, Q \in \mathcal{P}_{n-1}$. Then for all $i = 2, \dots, n$, there exist $R^{(i)}, S^{(i)} \in \mathcal{P}_{n-1}$ such that

(a) for all $i = 2, \dots, n$,

$$\begin{aligned}U[P, Q]^\dagger : \mathcal{W}(X_i) &\rightarrow I \otimes R^{(i)}, \\ \mathcal{W}(Z_i) &\rightarrow I \otimes S^{(i)},\end{aligned}$$

and

- (i) $[R^{(i)}, R^{(j)}] = [S^{(i)}, S^{(j)}] = 0$ for all $i, j = 2, \dots, n$,
- (ii) $[R^{(i)}, S^{(j)}] = 0$ for all $i \neq j \in \{2, \dots, n\}$,
- (iii) $\{R^{(i)}, S^{(i)}\} = 0$ for all $i \in \{2, \dots, n\}$,

(b) if $\mathcal{V} \in \tilde{\mathcal{C}}_{n-1}$ satisfies

$$\begin{aligned}\mathcal{V} : X_i &\mapsto R^{(i+1)} && \text{for } i = 1, \dots, n-1, \\ Z_i &\mapsto S^{(i+1)} && \text{for } i = 1, \dots, n-1,\end{aligned}\tag{A.29}$$

then

$$\begin{aligned}U[P, Q] \{(I \otimes V)(\cdot)\} U[P, Q]^\dagger : X_i &\mapsto \mathcal{W}(X_i) && \text{for } i = 1, \dots, n-1 \\ Z_i &\mapsto \mathcal{W}(Z_i) && \text{for } i = 1, \dots, n-1.\end{aligned}\tag{A.30}$$

Proof. Write $U = U[P, Q]$.

- (a) For all $i > 1$, X_i and Z_i commute with X_1 and Z_1 , and hence, by the commutation preserving property of $\tilde{\mathcal{C}}_n$, $\mathcal{W}(X_i)$ and $\mathcal{W}(Z_i)$ commute with $\mathcal{W}(X_1) =$

$X \otimes P$ and $\mathcal{W}(Z_i) = Z \otimes Q$. This implies that $U^\dagger(\mathcal{W}(X_i))U$ and $U^\dagger(\mathcal{W}(Z_i))U$ commute with $U^\dagger(X \otimes P)U = X_1$ and $U^\dagger(Z \otimes Q)U = Z_1$.

Since the only single-qubit Pauli operator that commutes with both X and Z is the identity operator, $U^\dagger(\mathcal{W}(X_i))U$ and $U^\dagger(\mathcal{W}(Z_i))U$ must act as the identity on the first qubit, i.e. for $i = 2, \dots, n$,

$$\begin{aligned} U^\dagger(\mathcal{W}(X_i))U &= I \otimes R^{(i)}, \\ U^\dagger(\mathcal{W}(Z_i))U &= I \otimes S^{(i)}, \end{aligned}$$

for some $R^{(i)}, S^{(i)} \in \mathcal{P}_{n-1}$.

- (i) For all i, j , $[\mathcal{W}(X_i), \mathcal{W}(X_j)] = [\mathcal{W}(Z_i), \mathcal{W}(Z_j)] = 0$. This implies $[I \otimes R^{(i)}, I \otimes R^{(j)}] = [I \otimes S^{(i)}, I \otimes S^{(j)}] = 0$, from which it follows that

$$[R^{(i)}, R^{(j)}] = [S^{(i)}, S^{(j)}] = 0.$$

- (ii) For all $i \neq j$, $[\mathcal{W}(X_i), \mathcal{W}(Z_j)] = 0$. This implies $[I \otimes R^{(i)}, I \otimes S^{(j)}] = 0$, from which it follows that

$$[R^{(i)}, S^{(j)}] = 0.$$

- (iii) For all i , $[\mathcal{W}(X_i), \mathcal{W}(Z_i)] = 0$. This implies $[I \otimes R^{(i)}, I \otimes S^{(i)}] = 0$, from which it follows that

$$[R^{(i)}, S^{(i)}] = 0.$$

- (b) Assume that \mathcal{V} satisfies Eq. (A.29). Then for $i = 2, \dots, n$, applying $I \otimes V$ and then $U[P, Q]$ gives

$$\begin{aligned} X_i &\mapsto I \otimes R^{(i)} \mapsto \mathcal{W}(X_i), \\ X_1 &\mapsto X_1 \mapsto X \otimes P = \mathcal{W}(X_1), \end{aligned}$$

$$\begin{aligned} Z_i &\mapsto I \otimes S^{(i)} \mapsto \mathcal{W}(Z_i), \\ Z_1 &\mapsto Z_1 \mapsto Z \otimes Q = \mathcal{W}(Z_1), \end{aligned}$$

which implies that Eq. (A.30) is satisfied.

□

We now generalize Lemma 208 to the case of general n .

Theorem 213. For all integers $n \geq 1$,

$$\tilde{\mathcal{C}}_n = \{U(\cdot)U^\dagger : \overline{\mathcal{P}}_n \rightarrow \overline{\mathcal{P}}_n | U \in \langle H, S, CZ \rangle^n\},$$

where $\langle H, S, CZ \rangle^1 = \langle H, S \rangle$.

Proof. The backward inclusion (\supseteq) was proved in Proposition 204. Hence, it remains to prove the forward inclusion (\subseteq). We will proceed by induction. The base case $n = 1$ is provided by Lemma 208.

Induction hypothesis: Assume that the theorem holds for $n - 1$, i.e.

$$\tilde{\mathcal{C}}_{n-1} \subseteq \{U(\cdot)U^\dagger : \overline{\mathcal{P}}_{n-1} \rightarrow \overline{\mathcal{P}}_{n-1} | U \in \langle H, S, CZ \rangle^{n-1}\}.$$

We want to show that it holds for n . Let $\mathcal{F} \in \tilde{\mathcal{C}}_n$. By Proposition 199, for all $i = 1, \dots, n$, there exist $t_i, s_i \in \mathbb{F}_2$ and $A^{(i)}, B^{(i)} \in \mathcal{P}_n$ such that

$$\begin{aligned} \mathcal{F}(X_i) &= (-1)^{t_i} A^{(i)}, \\ \mathcal{F}(Z_i) &= (-1)^{s_i} B^{(i)}. \end{aligned}$$

Let $\mathcal{H}(\cdot) = \mathcal{F}(X^s Z^t(\cdot) Z^t X^s)$. Then by Lemma 201,

$$\begin{aligned} \mathcal{H}(X_i) &= (-t)^{t_i} \mathcal{F}(X_i) = A^{(i)}, \\ \mathcal{H}(Z_i) &= (-t)^{s_i} \mathcal{F}(Z_i) = B^{(i)}. \end{aligned}$$

Since X_1 and Z_1 anticommute, $\mathcal{F}(X_1)$ and $\mathcal{F}(Z_1)$ anticommute, which implies that $A^{(1)}$ and $B^{(1)}$ anticommute. By Lemma 209, there exist $G \in \langle \text{SWAP}, H, S \rangle^n$ and $P, Q \in \mathcal{P}_{n-1}$, where $[P, Q] = 0$, such that

$$\begin{aligned} G : A^{(1)} &\rightarrow X \otimes P, \\ B^{(1)} &\rightarrow Z \otimes Q. \end{aligned}$$

Let $\mathcal{W}(\cdot) = G\mathcal{H}(\cdot)G^\dagger$. Then,

$$\begin{aligned} \mathcal{W} : X_1 &\xrightarrow{\mathcal{H}} A^{(1)} \xrightarrow{G} X \otimes P, \\ Z_1 &\xrightarrow{\mathcal{H}} B^{(1)} \xrightarrow{G} Z \otimes Q. \end{aligned}$$

By Lemma 212, for all $i = 2, \dots, n$, there exist $R^{(i)}, S^{(i)} \in \mathcal{P}_{n-1}$ such that

$$\begin{aligned} U[P, Q]^\dagger : \mathcal{W}(X_i) &\rightarrow I \otimes R^{(i)}, \\ \mathcal{W}(Z_i) &\rightarrow I \otimes S^{(i)}, \end{aligned}$$

and

- (i) $[R^{(i)}, R^{(j)}] = [S^{(i)}, S^{(j)}] = 0$ for all $i, j = 2, \dots, n$,
- (ii) $[R^{(i)}, S^{(j)}] = 0$ for all $i \neq j \in \{2, \dots, n\}$,
- (iii) $\{R^{(i)}, S^{(i)}\} = 0$ for all $i \in \{2, \dots, n\}$.

Let $\mathcal{V} : \overline{\mathcal{P}}_{n-1} \rightarrow \overline{\mathcal{P}}_{n-1}$ be defined by: $\forall k \in \mathbb{N}, u_i, v_i \in \mathbb{F}_2$,

$$\mathcal{V}(i^k X_1^{u_1} Z_1^{v_1} \dots X_{n-1}^{u_{n-1}} Z_{n-1}^{v_{n-1}}) = i^k \mathcal{V}(X_1)^{u_1} \mathcal{V}(Z_1)^{v_1} \dots \mathcal{V}(X_{n-1})^{u_{n-1}} \mathcal{V}(Z_{n-1})^{v_{n-1}},$$

with

$$\begin{aligned} \mathcal{V} : X_i &\mapsto R^{(i+1)} \quad \text{for } i = 1, \dots, n-1, \\ Z_i &\mapsto S^{(i+1)} \quad \text{for } i = 1, \dots, n-1. \end{aligned} \tag{A.31}$$

(\subseteq) By Eqs. (2.43), (2.44) and (2.47), $H_i, S_i, CZ_{ij} \in \mathcal{C}_n$. Since \mathcal{C}_n is closed under multiplication, tensor products and global phase,

$$\langle H, S, CZ \rangle^n / \mathcal{U}(1) \subseteq \mathcal{C}_n / \mathcal{U}(1).$$

(\supseteq) Let $[U] \in \mathcal{C}_n / \mathcal{U}(1)$. Hence, $U = e^{i\theta} V$ for some $V \in \mathcal{C}_n$. So, $U(\cdot)U^\dagger = V(\cdot)V^\dagger \in \tilde{\mathcal{C}}_n$, which implies that $U(\cdot)U^\dagger = W(\cdot)W^\dagger$ for some $W \in \langle H, S, CZ \rangle^n$. By Lemma 203, $U \sim W$, which implies that $[U] \in \langle H, S, CZ \rangle^n / \mathcal{U}(1)$.

□

Note that the above propositions give the following characterizations of the group $\tilde{\mathcal{C}}_n$.

$$\tilde{\mathcal{C}}_n = \{U(\cdot)U^\dagger : \overline{\mathcal{P}}_n \rightarrow \overline{\mathcal{P}}_n | U \in \mathcal{C}_n\} \quad (\text{A.32})$$

$$= \{U(\cdot)U^\dagger : \overline{\mathcal{P}}_n \rightarrow \overline{\mathcal{P}}_n | [U] \in \mathcal{C}_n / \mathcal{U}(1)\} \quad (\text{A.33})$$

$$= \{U(\cdot)U^\dagger : \overline{\mathcal{P}}_n \rightarrow \overline{\mathcal{P}}_n | [U] \in \langle H, S, CZ \rangle^n / \mathcal{U}(1)\}. \quad (\text{A.34})$$

Next, we show that the group $\tilde{\mathcal{C}}_n$ is isomorphic to the Clifford group $\mathcal{C}_n / \mathcal{U}(1)$.

Corollary 215. $\tilde{\mathcal{C}}_n \cong \mathcal{C}_n / \mathcal{U}(1)$.

Proof. It is easy to check that the map

$$\begin{aligned} \xi : \mathcal{C}_n / \mathcal{U}(1) &\rightarrow \tilde{\mathcal{C}}_n \\ [U] &\mapsto U(\cdot)U^\dagger \end{aligned}$$

is an isomorphism. □

A.3 Enumerating the Clifford group

In this section, we will count, for each integer $n \geq 1$, the number of elements in the n -qubit Clifford group.

Lemma 216. The number of elements in the single-qubit Clifford group is

$$|\tilde{\mathcal{C}}_1| = |\mathcal{C}_1/\mathcal{U}(1)| = 24.$$

Proof. Consider the map

$$\begin{aligned} \xi : \tilde{\mathcal{C}}_1 &\rightarrow \mathbb{F}_2 \times \mathbb{F}_2 \times \mathcal{S}_3 \\ f &\mapsto (a, c, (i, j, k)), \end{aligned} \tag{A.35}$$

where $a, c \in \mathbb{F}_2$ and $(i, j, k) \in \mathcal{S}_3$ are determined by

$$\begin{aligned} f(X) &= (-1)^a \sigma_i, \\ f(Z) &= (-1)^c \sigma_k. \end{aligned}$$

- First, we show that ξ is injective. Let f and g satisfy $\xi(f) = \xi(g)$. Then $(a, c, i, j, k) = (a', c', i', j', k')$, where

$$\begin{aligned} f(X) &= (-1)^a \sigma_i, & g(X) &= (-1)^{a'} \sigma_{i'}, \\ f(Z) &= (-1)^c \sigma_k, & g(Z) &= (-1)^{c'} \sigma_{k'}. \end{aligned}$$

Hence, $f(X) = g(X)$ and $f(Z) = g(Z)$. By Proposition 198, $f = g$, and hence ξ is injective.

- Next, we show that ξ is surjective. Let $(a, c, (i, j, k)) \in \mathbb{F}_2 \times \mathbb{F}_2 \times \mathcal{S}_3$. By Lemma 207, there exists $U \in \langle H, S \rangle$ such that

$$\begin{aligned} UXU^\dagger &= (-1)^a \sigma_i, \\ UZU^\dagger &= (-1)^c \sigma_k. \end{aligned}$$

But $\langle H, S \rangle \subseteq \mathcal{C}_1$. Hence, $U \in \mathcal{C}_1 \implies U(\cdot)U^\dagger \in \tilde{\mathcal{C}}_1 \implies \xi(U(\cdot)U^\dagger) = (a, c, (i, j, k))$. Therefore, ξ is surjective.

Since ξ is a bijection, $|\mathcal{C}_1/\mathcal{U}(1)| = |\mathbb{F}_2 \times \mathbb{F}_2 \times \mathcal{S}_3| = 2 \times 2 \times 3! = 24$. \square

A complete list of all 24 elements of the Clifford group $\mathcal{C}_1/\mathcal{U}(1)$ is given in Table 2.1, where we index the elements of the Clifford group by $\alpha = (-1)^a$, $\gamma = (-1)^c$ and $(i, j, k) \in \mathcal{S}_3$, where a, c, i, j, k are given by $\xi(f) = (a, c, (i, j, k))$ that was defined in Eq. (A.35).

Next, we consider the case when $n \geq 2$.

Lemma 217. For all $n \geq 2$,

$$|\tilde{\mathcal{C}}_n| = 2 \cdot 4^n(4^n - 1)|\tilde{\mathcal{C}}_{n-1}|.$$

Proof. We follow the proof given in [182]. Let $f \in \tilde{\mathcal{C}}_n$. By Proposition 198, f is completely determined by $f(X_1), f(Z_1), f(X_2), f(Z_2), \dots, f(X_n), f(Z_n)$. Consider the last pair $f(X_n), f(Z_n)$. $f(X_n)$ can take any value in $\pm\mathcal{P}_n^*$, and hence, the number of choices for $f(X_n)$ is

$$\#_n^X = |\pm\mathcal{P}_n^*| = 2|\mathcal{P}_n^*| = 2(4^n - 1).$$

$f(Z_n)$ can take any value in $\pm\mathcal{P}_n^*$ that anticommutes with $f(X_n)$, and hence, the number of choices for $f(Z_n)$ is

$$\begin{aligned} \#_n^Z &= \text{number of matrices in } \pm\mathcal{P}_n^* \text{ that anticommute with } f(X_n) \\ &= \text{number of matrices in } \pm\mathcal{P}_n \text{ that anticommute with } f(X_n) \\ &\hspace{15em} (\text{since } \pm I \text{ commute with } f(X_n)) \\ &= \frac{1}{2} |\pm\mathcal{P}_n|, \quad \text{since half of the matrices in } \pm\mathcal{P}_n \text{ anticommute with } f(X_n). \\ &= |\mathcal{P}_n| = 4^n. \end{aligned}$$

The number of choices for $f(X_1), f(Z_1), \dots, f(X_{n-1}), f(Z_{n-1})$ is $\#_{1, \dots, n-1}^{XZ} = |\tilde{\mathcal{C}}_{n-1}|$.

Hence,

$$|\tilde{\mathcal{C}}_n| = \#_n^X \cdot \#_n^Z \cdot \#_{1, \dots, n-1}^{XZ}$$

$$= 2 \cdot 4^n(4^n - 1)|\tilde{\mathcal{C}}_{n-1}|.$$

□

Theorem 218. Let $n \geq 1$. The number of elements in the n -qubit Clifford group is

$$|\mathcal{C}_n/\mathcal{U}(1)| = |\tilde{\mathcal{C}}_n| = 2^{n(n+2)} \prod_{j=1}^n (4^j - 1). \quad (\text{A.36})$$

Proof. By applying Lemma 217 recursively,

$$|\tilde{\mathcal{C}}_n| = \left(\prod_{j=1}^{n-1} 2 \cdot 4^j (4^j - 1) \right) |\tilde{\mathcal{C}}_1|.$$

But from Lemma 216, $|\tilde{\mathcal{C}}_1| = 24 = 2 \cdot 4^1(4^1 - 1)$. Hence,

$$\begin{aligned} |\tilde{\mathcal{C}}_n| &= \prod_{j=1}^n 2 \cdot 4^j (4^j - 1) \\ &= 2^n \cdot 4^{\sum_{j=1}^n j} \prod_{j=1}^n (4^j - 1) \\ &= 2^n \cdot 4^{\frac{n(n+1)}{2}} \prod_{j=1}^n (4^j - 1) \\ &= 2^{n(n+2)} \prod_{j=1}^n (4^j - 1). \end{aligned}$$

□

Note that $1 \leq \prod_{j=1}^n (4^j - 1) \leq \prod_{j=1}^n 4^j = 2^{n(n+1)}$. Hence,

$$2^{n^2+2n} \leq |\tilde{\mathcal{C}}_n| \leq 2^{2n^2+3n}.$$

Hence, $|\tilde{\mathcal{C}}_n| = 2^{\Theta(n^2)}$, i.e. the Clifford group grows exponentially with n . The exponential growth of the Clifford group means that it quickly becomes infeasible to perform algorithms that involve writing down all the elements of the Clifford group. For example, if one wanted to sample uniformly from the Clifford group, it would be

n	$ \tilde{\mathcal{C}}_n = \mathcal{C}_n/\mathcal{U}(1) $	$ \langle H, S, CZ \rangle^n $
1	24	192
2	11520	92160
3	9.289728×10^7	7.4317824×10^8
4	$1.21286688768 \times 10^{13}$	$9.70293510144 \times 10^{13}$
5	$2.54108226784591872 \times 10^{19}$	$2.032865814276734976 \times 10^{20}$
6	$8.52437556169034724016128 \times 10^{26}$	$6.819500449352277792129024 \times 10^{27}$

Table A.1: Number of elements in the n -qubit Clifford group $|\mathcal{C}_n/\mathcal{U}(1)|$ and the group of n -qubit operators generated by H, S, CZ , for $n = 1, 2, \dots, 6$. The cardinalities of these groups are related by $|\tilde{\mathcal{C}}_n| = |\mathcal{C}_n/\mathcal{U}(1)| = \frac{1}{8}|\langle H, S, CZ \rangle^n|$. The sequence $|\langle H, S, CZ \rangle^n|$ is recorded as sequence A003956 in the On-Line Encyclopedia of Integer Sequences (OEIS) [133].

impractical to write down all elements of the group and then pick randomly from the list; for such a task, it is possible to perform much better [147].

The first four values of $|\tilde{\mathcal{C}}_n| = |\mathcal{C}_n/\mathcal{U}(1)|$, for $n \geq 1$, are 24, 11520, 92897280 and 12128668876800 (see Table A.1). As noted in [182], these values are a factor of 8 smaller than the numbers quoted in [61, 133]. This reason for the difference is that the ‘Clifford group’ considered in these references refers to $\langle H, S, CZ \rangle^n$ instead of $\mathcal{C}_n/\mathcal{U}(1)$. A factor of 8 arises because

$$(HS)^3 = e^{i\pi/4}I.$$

and so every equivalence class $[U] \in \mathcal{C}_n/\mathcal{U}(1)$ contains 8 Clifford operators that differ by a global phase: $[U] = \{e^{ik\pi/4}U : k \in \mathbb{Z}_8\}$. Hence,

$$|\langle H, S, CZ \rangle^n| = 8|\tilde{\mathcal{C}}_n| = 2^{n^2+2n+3} \prod_{j=1}^n (4^j - 1).$$

The first four values of $|\langle H, S, CZ \rangle^n|$, for $n \geq 1$, are 192, 92160, 743178240 and 97029351014400 (see Table A.1).

Bibliography

- [1] Scott Aaronson. *Limits on efficient computation in the physical world*. PhD thesis, University of California, Berkeley, 2004.
- [2] Scott Aaronson. Quantum computing, postselection, and probabilistic polynomial-time. In *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, volume 461, pages 3473–3482. The Royal Society, 2005.
- [3] Scott Aaronson. A linear-optical proof that the permanent is $\#P$ -hard. In *Proc. R. Soc. A*, volume 467, pages 3393–3405. The Royal Society, 2011.
- [4] Scott Aaronson. $P \stackrel{?}{=} NP$. In *Open problems in mathematics*, pages 1–122. Springer, 2016 (Updated version at <https://www.scottaaronson.com/papers/pnp.pdf>).
- [5] Scott Aaronson and Alex Arkhipov. The computational complexity of linear optics. In *Proceedings of the forty-third annual ACM symposium on Theory of computing*, pages 333–342. ACM, 2011.
- [6] Scott Aaronson, Adam Bouland, Greg Kuperberg, and Saeed Mehraban. The Computational Complexity of Ball Permutations. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017*, pages 317–327, New York, NY, USA, 2017. ACM.
- [7] Scott Aaronson and Lijie Chen. Complexity-theoretic foundations of quantum supremacy experiments. In *Proceedings of the 32Nd Computational Complexity Conference, CCC '17*, pages 22:1–22:67, Germany, 2017. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.
- [8] Scott Aaronson and Daniel Gottesman. Improved simulation of stabilizer circuits. *Physical Review A*, 70(5):52328, 2004.
- [9] Scott Aaronson et al. Complexity zoo. www.complexityzoo.com.
- [10] Daniel S Abrams and Seth Lloyd. Nonlinear quantum mechanics implies polynomial-time solution for NP-complete and $\#P$ problems. *Physical Review Letters*, 81(18):3992, 1998.

- [11] Leonard M Adleman, Jonathan DeMarrais, and Ming-Deh A Huang. Quantum computability. *SIAM Journal on Computing*, 26(5):1524–1540, 1997.
- [12] Dorit Aharonov. A simple proof that Toffoli and Hadamard are quantum universal. *arXiv preprint quant-ph/0301040*, 2003.
- [13] Dorit Aharonov and Michael Ben-Or. Fault-tolerant quantum computation with constant error. In *Proceedings of the twenty-ninth annual ACM Symposium on Theory of Computing*, pages 176–188. ACM, 1997.
- [14] Dorit Aharonov, Wim Van Dam, Julia Kempe, Zeph Landau, Seth Lloyd, and Oded Regev. Adiabatic quantum computation is equivalent to standard quantum computation. *SIAM review*, 50(4):755–787, 2008.
- [15] DM Appleby. Properties of the extended Clifford group with applications to SIC-POVMs and MUBs. *arXiv preprint arXiv:0909.5233*, 2009.
- [16] Sanjeev Arora and Boaz Barak. *Computational complexity: a modern approach*. Cambridge University Press, 2009.
- [17] Alán Aspuru-Guzik, Anthony D. Dutoi, Peter J. Love, and Martin Head-Gordon. Simulated quantum computation of molecular energies. *Science*, 309(5741):1704–1707, 2005.
- [18] Ryan Babbush, Nathan Wiebe, Jarrod McClean, James McClain, Hartmut Neven, and Garnet Kin-Lic Chan. Low-depth quantum simulation of materials. *Physical Review X*, 8(1):011044, 2018.
- [19] Dave Bacon, Wim Van Dam, and Alexander Russell. Analyzing algebraic quantum circuits using exponential sums. <https://www.cs.ucsb.edu/van-dam/LeastAction.pdf>, 2008.
- [20] Dave Morris Bacon. *Decoherence, control, and symmetry in quantum computers*. PhD thesis, University of California, Berkeley, 2001.
- [21] Boaz Barak, Ankur Moitra, Ryan O’Donnell, Prasad Raghavendra, Oded Regev, David Steurer, Luca Trevisan, Aravindan Vijayaraghavan, David Witmer, and John Wright. Beating the Random Assignment on Constraint Satisfaction Problems of Bounded Degree. In Naveen Garg, Klaus Jansen, Anup Rao, and Jose D. P. Rolim, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2015)*, volume 40 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 110–123, Dagstuhl, Germany, 2015. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.
- [22] Adriano Barenco, Charles H Bennett, Richard Cleve, David P DiVincenzo, Norman Margolus, Peter Shor, Tycho Sleator, John A Smolin, and Harald Weinfurter. Elementary gates for quantum computation. *Physical Review A*, 52(5):3457, 1995.

- [23] Christopher Beck and Russell Impagliazzo. Strong ETH holds for regular resolution. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, pages 487–494. ACM, 2013.
- [24] Richard Beigel and Jun Tarui. On ACC. In *Foundations of Computer Science, 1991. Proceedings., 32nd Annual Symposium on*, pages 783–792. IEEE, 1991.
- [25] Charles H Bennett, Ethan Bernstein, Gilles Brassard, and Umesh Vazirani. Strengths and weaknesses of quantum computing. *SIAM journal on Computing*, 26(5):1510–1523, 1997.
- [26] Piotr Berman and Marek Karpinski. On some tighter inapproximability results. In *International Colloquium on Automata, Languages, and Programming*, pages 200–209. Springer, 1999.
- [27] Juan Bermejo-Vega, Dominik Hangleiter, Martin Schwarz, Robert Raussendorf, and Jens Eisert. Architectures for quantum simulation showing a quantum speedup. *Physical Review X*, 8(2):021010, 2018.
- [28] Bruce C Berndt and Ronald J Evans. Half Gauss sums. *Mathematische Annalen*, 249(2):115–125, 1980.
- [29] Ethan Bernstein and Umesh Vazirani. Quantum complexity theory. *SIAM Journal on computing*, 26(5):141F–1473, 1997.
- [30] Andreas Björklund, Petteri Kaski, and Ryan Williams. Generalized Kakeya sets for polynomial evaluation and faster computation of fermionants. In *LIPICs-Leibniz International Proceedings in Informatics*, volume 89. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2018.
- [31] Elmar Böhler, Christian Glaßer, and Daniel Meister. Error-bounded probabilistic computations between MA and AM. *Journal of Computer and System Sciences*, 72(6):1043–1076, 2006.
- [32] Sergio Boixo, Sergei V Isakov, Vadim N Smelyanskiy, Ryan Babbush, Nan Ding, Zhang Jiang, Michael J Bremner, John M Martinis, and Hartmut Neven. Characterizing quantum supremacy in near-term devices. *Nature Physics*, 14(6):595, 2018.
- [33] Beverley Bolt, TG Room, and GE Wall. On the Clifford collineation, transform and similarity groups. I. *Journal of the Australian Mathematical Society*, 2(1):60–79, 1961.
- [34] Beverley Bolt, TG Room, and GE Wall. On the Clifford collineation, transform and similarity groups. II. *Journal of the Australian Mathematical Society*, 2(1):80–96, 1961.
- [35] Adam Bouland and Scott Aaronson. Generation of universal linear optics by any beamsplitter. *Physical Review A*, 89(6):062316, 2014.

- [36] Adam Bouland, Joseph F. Fitzsimons, and Dax Enshan Koh. Complexity Classification of Conjugated Clifford Circuits. In Rocco A. Servedio, editor, *33rd Computational Complexity Conference (CCC 2018)*, volume 102 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 21:1–21:25, Dagstuhl, Germany, 2018. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.
- [37] Adam Bouland, Laura Mancinska, and Xue Zhang. Complexity Classification of Two-Qubit Commuting Hamiltonians. In Ran Raz, editor, *31st Conference on Computational Complexity (CCC 2016)*, volume 50 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 28:1–28:33, Dagstuhl, Germany, 2016. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.
- [38] P Oscar Boykin, Tal Mor, Matthew Pulver, Vwani Roychowdhury, and Farrokh Vatan. On universal and fault-tolerant quantum computing: a novel basis and a new constructive proof of universality for Shor’s basis. In *Foundations of Computer Science, 1999. 40th Annual Symposium on*, pages 486–494. IEEE, 1999.
- [39] P Oscar Boykin, Tal Mor, Matthew Pulver, Vwani Roychowdhury, and Farrokh Vatan. A new universal and fault-tolerant quantum basis. *Information Processing Letters*, 75(3):101–107, 2000.
- [40] Sergey Bravyi, Dan Browne, Padraic Calpin, Earl Campbell, David Gosset, and Mark Howard. Simulation of quantum circuits by low-rank stabilizer decompositions. *arXiv preprint arXiv:1808.00128*, 2018.
- [41] Sergey Bravyi, Jay M Gambetta, Antonio Mezzacapo, and Kristan Temme. Tapering off qubits to simulate fermionic Hamiltonians. *arXiv preprint arXiv:1701.08213*, 2017.
- [42] Sergey Bravyi and David Gosset. Improved classical simulation of quantum circuits dominated by Clifford gates. *Physical Review Letters*, 116(25):250501, 2016.
- [43] Sergey Bravyi, David Gosset, and Robert Koenig. Quantum advantage with shallow circuits. *Science*, 362(6412):308–311, 2018.
- [44] Sergey Bravyi and Jeongwan Haah. Magic-state distillation with low overhead. *Physical Review A*, 86(5):052329, 2012.
- [45] Sergey Bravyi and Alexei Kitaev. Universal quantum computation with ideal Clifford gates and noisy ancillas. *Physical Review A*, 71(2):022316, 2005.
- [46] Sergey B Bravyi and Alexei Yu Kitaev. Fermionic quantum computation. *Annals of Physics*, 298(1):210–226, 2002.
- [47] Michael J Bremner, Christopher M Dawson, Jennifer L Dodd, Alexei Gilchrist, Aram W Harrow, Duncan Mortimer, Michael A Nielsen, and Tobias J Osborne.

- Practical scheme for quantum computation with any two-qubit entangling gate. *Physical Review Letters*, 89(24):247902, 2002.
- [48] Michael J Bremner, Richard Jozsa, and Dan J Shepherd. Classical simulation of commuting quantum computations implies collapse of the polynomial hierarchy. In *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, page rspa20100301. The Royal Society, 2010.
- [49] Michael J Bremner, Ashley Montanaro, and Dan J Shepherd. Average-case complexity versus approximate simulation of commuting quantum computations. *Physical Review Letters*, 117(8):80501, 2016.
- [50] Michael J Bremner, Ashley Montanaro, and Dan J Shepherd. Achieving quantum supremacy with sparse and noisy commuting quantum computations. *Quantum*, 1, 2017.
- [51] Hans J Briegel, David E Browne, W Dür, Robert Raussendorf, and Maarten Van den Nest. Measurement-based quantum computation. *Nature Physics*, 5(1):19–26, 2009.
- [52] Jean-Luc Brylinski and Ranee Brylinski. Universal quantum gates. *Mathematics of Quantum Computation*, 79, 2002.
- [53] Kaifeng Bu and Dax Enshan Koh. Classical simulation of quantum circuits by half Gauss sums. *arXiv preprint arXiv:1812.00224*, 2018.
- [54] Andrei Bulatov and Martin Grohe. The complexity of partition functions. *Theoretical Computer Science*, 348(2-3):148–186, 2005.
- [55] Andrei A Bulatov. The complexity of the counting constraint satisfaction problem. *Journal of the ACM (JACM)*, 60(5):34, 2013.
- [56] Jin-Yi Cai and Xi Chen. *Complexity Dichotomies for Counting Problems : Volume 1, Boolean Domain*. Cambridge University Press, 2015.
- [57] Jin-Yi Cai, Xi Chen, Richard Lipton, and Pinyan Lu. On tractable exponential sums. In Der-Tsai Lee, Danny Z. Chen, and Shi Ying, editors, *Frontiers in Algorithmics*, pages 148–159, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.
- [58] Jin-Yi Cai, Pinyan Lu, and Mingji Xia. The complexity of complex weighted Boolean $\#CSP$. *Journal of Computer and System Sciences*, 80(1):217 – 236, 2014.
- [59] Chris Calabro, Russell Impagliazzo, and Ramamohan Paturi. The complexity of satisfiability of small depth circuits. In *International Workshop on Parameterized and Exact Computation*, pages 75–85. Springer, 2009.

- [60] A Robert Calderbank, Eric M Rains, Peter W Shor, and Neil JA Sloane. Quantum error correction and orthogonal geometry. *Physical Review Letters*, 78(3):405, 1997.
- [61] A Robert Calderbank, Eric M Rains, PM Shor, and Neil JA Sloane. Quantum error correction via codes over $GF(4)$. *IEEE Transactions on Information Theory*, 44(4):1369–1387, 1998.
- [62] AR Calderbank, RH Hardin, EM Rains, PW Shor, and Neil James Alexander Sloane. A group-theoretic framework for the construction of packings in Grassmannian spaces. *Journal of Algebraic Combinatorics*, 9(2):129–140, 1999.
- [63] Marco L Carmosino, Jiawei Gao, Russell Impagliazzo, Ivan Mihajlin, Ramamoohan Paturi, and Stefan Schneider. Nondeterministic extensions of the strong exponential time hypothesis and consequences for non-reducibility. In *Proceedings of the 2016 ACM Conference on Innovations in Theoretical Computer Science*, pages 261–270. ACM, 2016.
- [64] Jorge Casanova, Carlos Sabin, Juan León, IL Egusquiza, Rene Gerritsma, Christian F Roos, Juan José Garcia-Ripoll, and Enrique Solano. Quantum simulation of the Majorana equation and unphysical operations. *Physical Review X*, 1(2):021018, 2011.
- [65] Xie Chen, Zheng-Cheng Gu, Zheng-Xin Liu, and Xiao-Gang Wen. Symmetry protected topological orders and the group cohomology of their symmetry group. *Physical Review B*, 87:155114, Apr 2013.
- [66] Claude C Chevalley. *The construction and study of certain important algebras*. Math. Soc. Japan, 1955.
- [67] Andrew M Childs. Universal computation by quantum walk. *Physical Review Letters*, 102(18):180501, 2009.
- [68] Peter Clifford and Raphaël Clifford. The classical complexity of boson sampling. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 146–155. SIAM, 2018.
- [69] John H Conway, Ronald H Hardin, and Neil JA Sloane. Packing lines, planes, etc.: Packings in Grassmannian spaces. *Experimental mathematics*, 5(2):139–159, 1996.
- [70] Stephen Cook. The P versus NP problem. *The millennium prize problems*, pages 87–104, 2006.
- [71] Stephen A Cook. The complexity of theorem-proving procedures. In *Proceedings of the third annual ACM symposium on Theory of computing*, pages 151–158. ACM, 1971.

- [72] Nadia Creignou and Miki Hermann. Complexity of generalized satisfiability counting problems. *Information and Computation*, 125(1):1 – 12, 1996.
- [73] W.J.R. Crosby. Solution of the problem 4136. *Amer. Math. Monthly*, 53:103–107, 1946.
- [74] Toby S Cubitt, Ashley Montanaro, and Stephen Piddock. Universal quantum hamiltonians. *Proceedings of the National Academy of Sciences*, 115(38):9497–9502, 2018.
- [75] Alexander M Dalzell. Lower bounds on the classical simulation of quantum circuits for quantum supremacy. Senior Thesis, Massachusetts Institute of Technology, 2017.
- [76] Alexander M Dalzell, Aram W Harrow, Dax Enshan Koh, and Rolando L La Placa. How many qubits are needed for quantum computational supremacy? *arXiv preprint arXiv:1805.05224*, 2018.
- [77] Christopher M Dawson, Andrew P Hines, Duncan Mortimer, Henry L Haselgrove, Michael A Nielsen, and Tobias J Osborne. Quantum computing and polynomial equations over the finite field \mathbb{Z}_2 . *Quantum Information & Computation*, 5(2):102–112, 2005.
- [78] Christopher M Dawson and Michael A Nielsen. The Solovay-Kitaev Algorithm. *Quantum Information & Computation*, 6(1):081–095, 2006.
- [79] Jeroen Dehaene and Bart De Moor. Clifford group, stabilizer states, and linear and quadratic operations over $\text{GF}(2)$. *Physical Review A*, 68:042318, Oct 2003.
- [80] Holger Dell, Thore Husfeldt, Dániel Marx, Nina Taslaman, and Martin Wahlén. Exponential time complexity of the permanent and the Tutte polynomial. *ACM Transactions on Algorithms (TALG)*, 10(4):21, 2014.
- [81] M. Van den Nest. Classical simulation of quantum computation, the Gottesman-Knill theorem, and slightly beyond. *Quantum Information & Computation*, 10(3-4):0258–0271, 2010.
- [82] David Deutsch. Quantum theory, the Church-Turing principle and the universal quantum computer. In *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, volume 400, pages 97–117. The Royal Society, 1985.
- [83] R Di Candia, B Mejia, H Castillo, JS Pedernales, J Casanova, and E Solano. Embedding quantum simulators for quantum computation of entanglement. *Physical Review Letters*, 111(24):240502, 2013.
- [84] David P. DiVincenzo and Peter W. Shor. Fault-tolerant error correction with efficient quantum codes. *Physical Review Letters*, 77:3260–3263, Oct 1996.

- [85] David Steven Dummit and Richard M Foote. *Abstract algebra*, volume 3. Wiley Hoboken, 2004.
- [86] W. Dür, G. Vidal, and J. I. Cirac. Three qubits can be entangled in two inequivalent ways. *Physical Review A*, 62:062314, Nov 2000.
- [87] M. Dyer, L. Goldberg, and M. Jerrum. The complexity of weighted Boolean #CSP. *SIAM Journal on Computing*, 38(5):1970–1986, 2009.
- [88] Martin Dyer, Leslie Ann Goldberg, and Mike Paterson. On counting homomorphisms to directed acyclic graphs. *J. ACM*, 54(6), December 2007.
- [89] Bryan Eastin and Emanuel Knill. Restrictions on transversal encoded quantum gate sets. *Physical Review Letters*, 102(11):110502, 2009.
- [90] Andrzej Ehrenfeucht and Marek Karpinski. The computational complexity of (XOR,AND)-counting problems. Technical report, Proc. 12th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA) (2001), 1990.
- [91] Michael Eisele. Finding room for antilinear terms in the Hamiltonian. *arXiv preprint arXiv:1204.1309*, 2012.
- [92] J Emerson, V Veitch, M Howard, D Gottesman, A Hamed, C Ferrie, and D Gross. Negative quasi-probability, contextuality, quantum magic and the power of quantum computation. Slides of a talk given at UBC, 2013.
- [93] Ronald Evans, Marvin Minei, and Bennet Yee. Incomplete higher-order Gauss sums. *Journal of mathematical analysis and applications*, 281(2):454–476, 2003.
- [94] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. A quantum approximate optimization algorithm. *arXiv preprint arXiv:1411.4028*, 2014.
- [95] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. A quantum approximate optimization algorithm applied to a bounded occurrence constraint problem. *arXiv preprint arXiv:1412.6062*, 2014.
- [96] Edward Farhi, Jeffrey Goldstone, Sam Gutmann, and Michael Sipser. Quantum computation by adiabatic evolution. *arXiv preprint quant-ph/0001106*, 2000.
- [97] Edward Farhi and Aram W Harrow. Quantum supremacy through the quantum approximate optimization algorithm. *arXiv preprint arXiv:1602.07674*, 2016.
- [98] J M Farinholt. An ideal characterization of the Clifford operators. *Journal of Physics A: Mathematical and Theoretical*, 47(30):305303, 2014.
- [99] Bill Fefferman and Christopher Umans. On the power of quantum Fourier sampling. In *11th Conference on the Theory of Quantum Computation, Communication and Cryptography, TQC 2016, September 27-29, 2016, Berlin, Germany*, pages 1:1–1:19, 2016.

- [100] Stephen Fenner, Frederic Green, Steven Homer, and Randall Pruim. Determining acceptance possibility for a quantum computation is hard for the polynomial hierarchy. In *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, volume 455, pages 3953–3966. The Royal Society, 1999.
- [101] José M Fernandez and William A Schneeberger. Quaternionic computing. *arXiv preprint quant-ph/0307017*, 2003.
- [102] Richard P Feynman. Simulating physics with computers. *International journal of theoretical physics*, 21(6-7):467–488, 1982.
- [103] Richard P Feynman, Albert R Hibbs, and Daniel F Styer. *Quantum mechanics and path integrals*. Courier Corporation, 2010.
- [104] Keisuke Fujii, Hirotada Kobayashi, Tomoyuki Morimae, Harumichi Nishimura, Shuhei Tamate, and Seiichiro Tani. Impossibility of classically simulating one-clean-qubit model with multiplicative error. *Physical Review Letters*, 120(20):200502, 2018.
- [105] CF Gauss. *Disquisitiones Arithmeticae*. Fleischer, Leipzig, 1801.
- [106] Vladimir P Gerdt and Vasily M Severyanov. An algorithm for constructing polynomial systems whose solution space characterizes quantum circuits. In *Quantum Informatics 2005*, pages 626406–626406. International Society for Optics and Photonics, 2006.
- [107] Michel X Goemans and David P Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM (JACM)*, 42(6):1115–1145, 1995.
- [108] Leslie Ann Goldberg, Martin Grohe, Mark Jerrum, and Marc Thurley. A complexity dichotomy for partition functions with mixed signs. *SIAM Journal on Computing*, 39(7):3336–3402, 2010.
- [109] Daniel Gottesman. *Stabilizer Codes and Quantum Error Correction*. PhD thesis, California Institute of Technology, 1997.
- [110] Daniel Gottesman. Theory of fault-tolerant quantum computation. *Physical Review A*, 57(1):127, 1998.
- [111] Daniel Gottesman. The Heisenberg representation of quantum computers. *Group22: Proceedings of the XXII International Colloquium on Group Theoretical Methods in Physics*, pages 32–43, 1999.
- [112] Daniel Gottesman. An introduction to quantum error correction and fault-tolerant quantum computation. In *Quantum information science and its contributions to mathematics, Proceedings of Symposia in Applied Mathematics*, volume 68, pages 13–58, 2010.

- [113] Daniel Gottesman and Isaac L Chuang. Demonstrating the viability of universal quantum computation using teleportation and single-qubit operations. *Nature*, 402(6760):390–393, 1999.
- [114] Chris Granade and Ben Criger. QuaEC: Quantum error correction analysis in Python. <http://www.cgranade.com/python-quaec/groups.html#>, 2012. Accessed: 2017-06-01.
- [115] Daniel Grier and Luke Schaeffer. The classification of stabilizer operations over qubits. *arXiv:1603.03999*, 2016.
- [116] Zheng-Cheng Gu and Xiao-Gang Wen. Symmetry-protected topological orders for interacting fermions: Fermionic topological nonlinear σ models and a special group supercohomology theory. *Physical Review B*, 90:115141, Sep 2014.
- [117] Eran Halperin, Dror Livnat, and Uri Zwick. MAX CUT in cubic graphs. *Journal of Algorithms*, 53(2):169–185, 2004.
- [118] Yenjo Han, Lane A Hemaspaandra, and Thomas Thierauf. Threshold computation and cryptographic security. *SIAM Journal on Computing*, 26(1):59–78, 1997.
- [119] Dominik Hangleiter, Juan Bermejo-Vega, Martin Schwarz, and Jens Eisert. Anticoncentration theorems for schemes showing a quantum speedup. *Quantum*, 2:65, May 2018.
- [120] Lucien Hardy. Reformulating and reconstructing quantum theory. *arXiv preprint arXiv:1104.2066*, 2011.
- [121] Daniel Harlow. Jerusalem lectures on black holes and quantum information. *Reviews of Modern Physics*, 88(1):015002, 2016.
- [122] Aram Harrow and Saeed Mehraban. Approximate unitary t -designs by short random quantum circuits using nearest-neighbor and long-range gates. *arXiv preprint arXiv:1809.06957*, 2018.
- [123] Aram W Harrow and Ashley Montanaro. Quantum computational supremacy. *Nature*, 549(7671):203, 2017.
- [124] Juris Hartmanis and Richard E Stearns. On the computational complexity of algorithms. *Transactions of the American Mathematical Society*, 117:285–306, 1965.
- [125] Johan Håstad. Some optimal inapproximability results. *J. ACM*, 48(4):798–859, July 2001.
- [126] Michał Horodecki, Paweł Horodecki, and Ryszard Horodecki. Separability of n -particle mixed states: necessary and sufficient conditions in terms of linear maps. *Physics Letters A*, 283(1):1–7, 2001.

- [127] L-K Hua. *Introduction to number theory*. Springer Science & Business Media, 2012.
- [128] Cupjin Huang, Michael Newman, and Mario Szegedy. Explicit lower bounds on strong quantum simulation, 2018. arXiv:1804.10368v2.
- [129] Marko Huhtanen and Allan Perämäki. Function theory of antilinear operators. *arXiv preprint arXiv:1212.0360*, 2012.
- [130] Marko Huhtanen and Santtu Ruotsalainen. Real linear operator theory and its applications. *Integral equations and operator theory*, 69(1):113–132, 2011.
- [131] Norman E Hurt. Exponential sums and coding theory: a review. *Acta Applicandae Mathematica*, 46(1):49–91, 1997.
- [132] Russell Impagliazzo, Ramamohan Paturi, and Francis Zane. Which problems have strongly exponential complexity? *Journal of Computer and System Sciences*, 63(4):512–530, 2001.
- [133] OEIS Foundation Inc. The On-Line Encyclopedia of Integer Sequences. <http://oeis.org/A003956>.
- [134] Arthur Jaffe and Bas Janssens. Reflection positive doubles. *Journal of Functional Analysis*, 272(8):3506–3557, 2017.
- [135] Arthur Jaffe and Zhengwei Liu. Planar para algebras, reflection positivity. *Communications in Mathematical Physics*, 352(1):95–133, May 2017.
- [136] Arthur Jaffe, Zhengwei Liu, and Alex Wozniakowski. Constructive simulation and topological design of protocols. *New Journal of Physics*, 19(6):063016, 2017.
- [137] Arthur Jaffe, Zhengwei Liu, and Alex Wozniakowski. Holographic software for quantum networks. *Science China Mathematics*, 61(4):593–626, Apr 2018.
- [138] Hamid Jahanjou, Eric Miles, and Emanuele Viola. Local reductions. In *International Colloquium on Automata, Languages, and Programming*, pages 749–760. Springer, 2015.
- [139] Mark Jerrum and Marc Snir. Some exact complexity results for straight-line computations over semirings. *Journal of the ACM (JACM)*, 29(3):874–897, 1982.
- [140] Erich Joos, H Dieter Zeh, Claus Kiefer, Domenico J W Giulini, Joachim Kupsch, and Ion-Olimpiu Stamatescu. *Decoherence and the appearance of a classical world in quantum theory*. Springer Science & Business Media, 2013.
- [141] Stephen Jordan. Quantum Algorithm Zoo. <http://quantumalgorithmzoo.org/>.

- [142] Richard Jozsa and Akimasa Miyake. Matchgates and classical simulation of quantum circuits. In *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, volume 464, pages 3089–3106. The Royal Society, 2008.
- [143] Richard Jozsa and Maarten Van den Nest. Classical simulation complexity of extended Clifford circuits. *Quantum Information & Computation*, 14(7&8):633–648, 2014.
- [144] Julian Kelly. A preview of Bristlecone, Google’s new quantum processor. <https://ai.googleblog.com/2018/03/a-preview-of-bristlecone-googles-new.html>.
- [145] A Yu Kitaev. Quantum computations: algorithms and error correction. *Russian Mathematical Surveys*, 52(6):1191–1249, 1997.
- [146] Emanuel Knill and Raymond Laflamme. Power of one bit of quantum information. *Physical Review Letters*, 81(25):5672, 1998.
- [147] Robert Koenig and John A. Smolin. How to efficiently select an arbitrary Clifford group element. *Journal of Mathematical Physics*, 55(12):122202, 2014.
- [148] Dax Enshan Koh. Further extensions of Clifford circuits and their classical simulation complexities. *Quantum Information & Computation*, 17(3&4):262–282, 2017.
- [149] Dax Enshan Koh, Murphy Yuezhen Niu, and Theodore J Yoder. Quantum simulation from the bottom up: the case of rebits. *Journal of Physics A: Mathematical and Theoretical*, 51(19):195302, 2018.
- [150] Dax Enshan Koh, Mark D Penney, and Robert W Spekkens. Computing quopit Clifford circuit amplitudes by the sum-over-paths technique. *Quantum Information & Computation*, 17(13&14):1081–1095, 2017.
- [151] N. M. Korobov. *Exponential sums and their applications*, volume 80. Springer Science & Business Media, 2013.
- [152] Karl Kraus. *States, effects and operations: fundamental notions of quantum theory*. Springer, 1983.
- [153] Greg Kuperberg. Complexity Zoology Introduction, State Inclusion Diagram. <https://www.math.ucdavis.edu/~greg/zoology/diagram.pdf>.
- [154] Greg Kuperberg. How hard is it to approximate the Jones polynomial? *Theory of Computing*, 11(6):183–219, 2015.
- [155] Raymond Laflamme, Cesar Miquel, Juan Pablo Paz, and Wojciech Hubert Zurek. Perfect quantum error correcting code. *Physical Review Letters*, 77(1):198, 1996.

- [156] Serge Lang. *Algebraic Number Theory*. Addison-Wesley, 1970.
- [157] Benjamin P Lanyon, James D Whitfield, Geoff G Gillett, Michael E Goggin, Marcelo P Almeida, Ivan Kassal, Jacob D Biamonte, Masoud Mohseni, Ben J Powell, Marco Barbieri, et al. Towards quantum chemistry on a quantum computer. *Nature Chemistry*, 2(2):106–111, 2010.
- [158] Clemens Lautemann. BPP and the polynomial hierarchy. *Information Processing Letters*, 17(4):215–217, 1983.
- [159] Derrick Henry Lehmer. Incomplete Gauss sums. *Mathematika*, 23(2):125–135, 1976.
- [160] L Levin. Universal search problems (in Russian). *Problemy Peredachi Informatsii*, 9(3):265–266, 1973.
- [161] Rudolf Lidl and Harald Niederreiter. *Finite Fields*. Cambridge University Press, 1997.
- [162] Daniel Lokshtanov, Ramamohan Paturi, Suguru Tamaki, Ryan Williams, and Huacheng Yu. Beating brute force for systems of polynomial equations over finite fields. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 2190–2202. SIAM, 2017.
- [163] Easwar Magesan, Jay M Gambetta, and Joseph Emerson. Scalable and robust randomized benchmarking of quantum processes. *Physical Review Letters*, 106(18):180504, 2011.
- [164] Yu. I Manin. *Vychislimoe i nevychislimoe*. Moscow: Sov. Radio, 1980.
- [165] Ryan L Mann and Michael J Bremner. On the complexity of random quantum computations and the Jones polynomial. *arXiv preprint arXiv:1711.00686*, 2017.
- [166] Andrea Mari and Jens Eisert. Positive Wigner functions render classical simulation of quantum computation efficient. *Physical Review Letters*, 109(23):230503, 2012.
- [167] Mark D Penney and Robert W Spekkens. Private communication, 2017.
- [168] D. Maslov and M. Roetteler. Shorter stabilizer circuits via Bruhat decomposition and quantum circuit transformations. *IEEE Transactions on Information Theory*, 64(7):4729–4738, 2018.
- [169] Matthew McKague. On the power quantum computation over real Hilbert spaces. *International Journal of Quantum Information*, 11(01):1350001, 2013.
- [170] Matthew McKague, Michele Mosca, and Nicolas Gisin. Simulating quantum systems using real Hilbert spaces. *Physical Review Letters*, 102(2):020505, 2009.

- [171] Ashley Montanaro. Quantum algorithms: an overview. *npj Quantum Information*, 2:15023, 2016.
- [172] Ashley Montanaro. Quantum circuits and low-degree polynomials over \mathbb{F}_2 . *Journal of Physics A: Mathematical and Theoretical*, 2017.
- [173] Tomoyuki Morimae. Hardness of classically sampling the one-clean-qubit model with constant total variation distance error. *Physical Review A*, 96:040302, Oct 2017.
- [174] Tomoyuki Morimae, Keisuke Fujii, and Joseph F Fitzsimons. Hardness of classically simulating the one-clean-qubit model. *Physical Review Letters*, 112(13):130502, 2014.
- [175] Kenichi Morita. Computation-universality of one-dimensional one-way reversible cellular automata. *Information Processing Letters*, 42(6):325–329, 1992.
- [176] Gabriele Nebe, Eric M. Rains, and Neil JA Sloane. The invariants of the Clifford groups. *Designs, Codes and Cryptography*, 24(1):99–122, 2001.
- [177] Gabriele Nebe, Eric M Rains, and Neil James Alexander Sloane. *Self-dual codes and invariant theory*, volume 17. Springer, 2006.
- [178] Alex Neville, Chris Sparrow, Raphaël Clifford, Eric Johnston, Patrick M Birchall, Ashley Montanaro, and Anthony Laing. Classical boson sampling algorithms with superior performance to near-term experiments. *Nature Physics*, 13(12):1153, 2017.
- [179] Xiaotong Ni, Oliver Buerschaper, and Maarten den Nest. A non-commuting stabilizer formalism. *Journal of Mathematical Physics*, 56(5):52201, 2015.
- [180] Michael A Nielsen and Isaac L Chuang. *Quantum computation and quantum information*. Cambridge University Press, 2010.
- [181] Michał Oszmaniec and Zoltán Zimborás. Universal extensions of restricted classes of quantum operations. *Physical Review Letters*, 119(22):220502, 2017.
- [182] Maris Ozols. Clifford group. [http://home.lu.lv/~sd20008/papers/essays/Clifford%20group%20\[paper\].pdf](http://home.lu.lv/~sd20008/papers/essays/Clifford%20group%20[paper].pdf), 2008.
- [183] Károly F Pál and Tamás Vértesi. Efficiency of higher-dimensional Hilbert spaces for the violation of Bell inequalities. *Physical Review A*, 77(4):042105, 2008.
- [184] Ian Parberry. Parallel speedup of sequential machines: A defense of parallel computation thesis. *ACM SIGACT News*, 18(1):54–67, 1986.
- [185] Kenneth G. Paterson. Applications of exponential sums in communications theory. In Michael Walker, editor, *Cryptography and Coding*, pages 1–24, Berlin, Heidelberg, 1999. Springer Berlin Heidelberg.

- [186] Mark D Penney, Dax Enshan Koh, and Robert W Spekkens. Quantum circuit dynamics via path integrals: Is there a classical action for discrete-time paths? *New Journal of Physics*, 19(7):073006, 2017.
- [187] Frank Pollmann and Ari M. Turner. Detection of symmetry-protected topological phases in one dimension. *Physical Review B*, 86:125441, Sep 2012.
- [188] John Preskill. Quantum computing and the entanglement frontier. *arXiv preprint arXiv:1203.5813*, 2012.
- [189] John Preskill. Quantum Computing in the NISQ era and beyond. *Quantum*, 2:79, 2018.
- [190] Pavel Pudlák and Russell Impagliazzo. A lower bound for DLL algorithms for k-SAT, 1999.
- [191] Robert Raussendorf and Hans J Briegel. A one-way quantum computer. *Physical Review Letters*, 86(22):5188, 2001.
- [192] Robert Raussendorf, Daniel E Browne, and Hans J Briegel. Measurement-based quantum computation on cluster states. *Physical Review A*, 68(2):022312, 2003.
- [193] Ran Raz and Avishay Tal. Oracle separation of BQP and PH. *Electronic Colloquium on Computational Complexity (ECCC)*, 25:107, 2018.
- [194] Michael Reck, Anton Zeilinger, Herbert J Bernstein, and Philip Bertani. Experimental realization of any discrete unitary operator. *Physical Review Letters*, 73(1):58, 1994.
- [195] Martin Roetteler, Michael Naehrig, Krysta M. Svore, and Kristin Lauter. Quantum resource estimates for computing elliptic curve discrete logarithms. In Tsuyoshi Takagi and Thomas Peyrin, editors, *Advances in Cryptology – ASIACRYPT 2017*, pages 241–270, Cham, 2017. Springer International Publishing.
- [196] Terry Rudolph and Lov Grover. A 2 rebit gate universal for quantum computing. *arXiv preprint quant-ph/0210187*, 2002.
- [197] Herbert John Ryser. *Combinatorial mathematics*. Mathematical Association of America; distributed by Wiley [New York], 1963.
- [198] Imdad SB Sardharwalla, Toby S Cubitt, Aram W Harrow, and Noah Linden. Universal refocusing of systematic quantum noise. *arXiv:1602.07963*, 2016.
- [199] Adam Sawicki and Katarzyna Karnas. Criteria for universality of quantum gates. *Physical Review A*, 95:062303, Jun 2017.
- [200] Richard Donald Schafer. *An introduction to nonassociative algebras*, volume 22. Courier Corporation, 1966.

- [201] Dan Shepherd and Michael J Bremner. Temporally unstructured quantum computation. In *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, volume 465, pages 1413–1439. The Royal Society, 2009.
- [202] Daniel James Shepherd. *Quantum complexity: restrictions on algorithms and architectures*. PhD thesis, University of Bristol, 2009.
- [203] Yaoyun Shi. Both Toffoli and controlled-NOT need little help to do universal quantum computing. *Quantum Information and Computation*, 3(1):084–092, 2003.
- [204] Ken Shiozaki, Hassan Shapourian, Kiyonori Gomi, and Shinsei Ryu. Many-body topological invariants for fermionic short-range entangled topological phases protected by antiunitary symmetries. *Physical Review B*, 98(3):035151, 2018.
- [205] Peter W Shor. Algorithms for quantum computation: Discrete logarithms and factoring. In *Proceedings 35th annual symposium on foundations of computer science*, pages 124–134. IEEE, 1994.
- [206] Peter W Shor. Fault-tolerant quantum computation. In *Proceedings of 37th Conference on Foundations of Computer Science*, pages 56–65. IEEE, 1996.
- [207] Peter W Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM review*, 41(2):303–332, 1999.
- [208] PW Shor and Neil James Alexander Sloane. A family of optimal packings in Grassmannian manifolds. *Journal of Algebraic Combinatorics*, 7(2):157–163, 1998.
- [209] Igor E Shparlinski. Exponential sums and lattice reduction: Applications to cryptography. In *Finite fields with applications to coding theory, cryptography and related areas*, pages 286–298. Springer, 2002.
- [210] Igor E Shparlinski. Exponential sums in coding theory, cryptology and algorithms. In *Coding Theory and Cryptology*, pages 323–383. World Scientific, 2002.
- [211] Michael Sipser. A complexity theoretic approach to randomness. In *Proceedings of the fifteenth annual ACM symposium on Theory of computing*, pages 330–335. Citeseer, 1983.
- [212] Michael Sipser. *Introduction to the Theory of Computation*. Cengage Learning, 2012.
- [213] John Smolin. The Church of the larger Hilbert space. <https://www.quantiki.org/wiki/church-larger-hilbert-space>.

- [214] Andrew Steane. Multiple-particle interference and quantum error correction. In *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, volume 452, pages 2551–2577. The Royal Society, 1996.
- [215] W Forrest Stinespring. Positive functions on C^* -algebras. *Proceedings of the American Mathematical Society*, 6(2):211–216, 1955.
- [216] Larry Stockmeyer. The complexity of approximate counting. In *Proceedings of the fifteenth annual ACM Symposium on Theory of Computing*, pages 118–126. ACM, 1983.
- [217] Barbara M Terhal and David P DiVincenzo. Adaptive quantum computation, constant depth quantum circuits and Arthur-Merlin games. *Quantum Information & Computation*, 4(2):134–145, 2004.
- [218] Theoretical Computer Science Stack Exchange. Universal sets of gates for $SU(3)$? <https://cstheory.stackexchange.com/questions/11308/universal-sets-of-gates-for-su3>, 2012. Accessed: 2017-08-01.
- [219] Seinosuke Toda. PP is as hard as the polynomial-time hierarchy. *SIAM Journal on Computing*, 20(5):865–877, 1991.
- [220] Seinosuke Toda and Mitsunori Ogiwara. Counting classes are at least as hard as the polynomial-time hierarchy. *SIAM Journal on Computing*, 21(2):316–328, 1992.
- [221] Alan Mathison Turing. On computable numbers, with an application to the entscheidungsproblem. *Proceedings of the London mathematical society*, 2(1):230–265, 1937.
- [222] Leslie G Valiant. The complexity of computing the permanent. *Theoretical computer science*, 8(2):189–201, 1979.
- [223] Leslie G Valiant. Quantum circuits that can be simulated classically in polynomial time. *SIAM Journal on Computing*, 31(4):1229–1254, 2002.
- [224] Maarten Van den Nest. Efficient classical simulations of Quantum Fourier transforms and normalizer circuits over Abelian groups. *Quantum Information and Computation*, 13:1007–1037, 2013.
- [225] Victor Veitch, Nathan Wiebe, Christopher Ferrie, and Joseph Emerson. Efficient simulation scheme for a class of quantum optics experiments with non-negative Wigner representation. *New Journal of Physics*, 15(1):13037, 2013.
- [226] John Watrous. Succinct quantum proofs for properties of finite groups. In *Proceedings 41st Annual Symposium on Foundations of Computer Science*, pages 537–546, 2000.

- [227] John Watrous. Quantum computational complexity. *Encyclopedia of complexity and systems science*, pages 7174–7201, 2009.
- [228] John Watrous. *The theory of quantum information*. Cambridge University Press, 2018.
- [229] Zak Webb. The Clifford group forms a unitary 3-design. *Quantum Information & Computation*, 16(15&16):1379–1400, 2016.
- [230] André Weil. Numbers of solutions of equations in finite fields. *Bull. Amer. Math. Soc.*, 55:497–508, 1949.
- [231] Ryan Williams. A new algorithm for optimal 2-constraint satisfaction and its implications. *Theoretical Computer Science*, 348(2-3):357–365, 2005.
- [232] Ryan Williams and Huacheng Yu. Finding orthogonal vectors in discrete structures. In *Proceedings of the twenty-fifth annual ACM-SIAM symposium on Discrete algorithms*, pages 1867–1877. SIAM, 2014.
- [233] T. Williams. *Advances in the Computational Complexity of Holant Problems*. PhD thesis, University of Wisconsin-Madison, 2015.
- [234] Virginia Vassilevska Williams. Hardness of easy problems: Basing hardness on popular conjectures such as the strong exponential time hypothesis. In *Proc. International Symposium on Parameterized and Exact Computation*, pages 16–28, 2015.
- [235] L. Włodarski. On the equation $\cos(\alpha_1) + \cos(\alpha_2) + \cos(\alpha_3) + \cos(\alpha_4) = 0$. *Ann. Univ. Sci. Budapest. Eötvös Sect. Math*, 12:147, 1969.
- [236] Alan R Woods. Unsatisfiable systems of equations, over a finite field. In *Foundations of Computer Science, 1998. Proceedings. 39th Annual Symposium on*, pages 202–211. IEEE, 1998.
- [237] William K. Wootters. Entanglement of formation of an arbitrary state of two qubits. *Physical Review Letters*, 80:2245–2248, 1998.
- [238] Andrew Chi-Chih Yao. Classical physics and the Church-Turing thesis. *Journal of the ACM (JACM)*, 50(1):100–105, 2003.
- [239] Huangjun Zhu. Multiqubit Clifford groups are unitary 3-designs. *Physical Review A*, 96(6):062336, 2017.