Interactive quantum advantage with noisy, shallow Clifford circuits

Nathan Ju

Abstract

Recent work by Bravyi et al. demonstrates a relation problem separation between *noisy* constant-depth quantum circuits (QNC^0) and bounded fan-in constant-depth classical circuits (NC^0): The former solves a certain relation problem with near certainty (probability $1 - o(1)$) while the latter cannot. We strengthen this result by showing there is a relation problem separation between noisy QNC^0 circuits and unbounded fan-in constant-depth classical circuits (AC^0). We also prove stronger results for interactive problems: Classical machines that solve certain interactive problems on greater than 29/30, 105/106, and 420/421 fractions of all inputs must also be solving NC^1-, w-PBP-, and $\oplus\mathsf{L}$-hard problems, respectively, while noisy QNC^0 circuits solve these problems with near certainty. These results unconditionally separate noisy QNC^0 circuits and $\mathsf{AC}^0[p]$ circuits for all primes $p \geq 2$. Under plausible conjectures, these results suggest an interactive separation between noisy QNC^0 circuits and log-space classical machines. Our main contribution are the classical average-case hardness results, which are worst- to average-case reductions using tools such as low-depth permutation sampling and randomized encodings.

1 Introduction

A major goal in quantum complexity theory is identifying problems which are efficiently solvable by quantum computers and not efficiently solvable by classical computers. If willing to believe certain conjectures, one can be convinced of this separation by the discovery of quantum algorithms that solve classically hard problems. For example, the belief that classical computers cannot efficiently factor integers contrasts with Shor's algorithm for factoring integers on a quantum computer [Sho97]. However, a demonstration of Shor's algorithm on instances that are not efficiently solvable by classical computers would require quantum resources far out of reach of near-term capabilities. This has spurred developments in devising sampling problems that separate efficient, near-term quantum computers and classical computers like IQP circuit sampling [BJS10], BosonSampling [AA11], and random circuit sampling [Boi+18]. However, convincing evidence that *noisy* quantum computers outperform classical computers in these tasks suffers from the necessity of assuming some non-standard complexity-theoretic conjectures that are often native to each proposal.

This tradeoff between provable separation and standard conjectures begs an interesting question: Can we demonstrate separations in computational power without assuming extra conjectures? A recent line of work shows that *noise-free* quantum and classical separations can be *unconditionally* proved if one is willing to pare the setting of comparison down to constant-depth circuits. A breakthrough result by Bravyi, Gosset, and König [BGK18] is a strict separation between constant-depth quantum circuits (QNC^0) and constant-depth classical circuits with bounded fan-in gates (NC^0). They introduced the 2D Hidden Linear Function (HLF) problem and showed that NC^0 circuits cannot solve it with high probability while a QNC^0 circuit solves it with certainty.

Bene Watts, Kothari, Schaeffer, and Tal [Ben+19] extended their results by proving that unbounded fan-in constant-depth classical circuits (AC^0) cannot solve the 2D HLF problem via a reduction from another hard problem for AC^0 circuits that they called Relaxed Parity Halving Problem. 2D HLF and Relaxed Parity Halving Problem are instances of relation problems: A circuit receives some input, and it can output *any* element from an acceptable set of outputs determined by the input. Grier and Schaeffer [GS20] stepped away from this approach and showed that an unconditional separation persists for an interactive problem: A challenger and circuit converse over multiple rounds, and the circuit succeeds if it correctly responds every round. They shows that the interactive tasks are solved by constant-depth quantum circuits on constant-width and poly-width grids, but any classical circuit solving these problem would be solving NC^1-hard and $\oplus\mathsf{L}$-hard problems, respectively.

Each of these constant-depth separations assumes that the quantum circuit is free of errors, or it is noiseless. However, near-term quantum devices are known to be noisy, making the expectation of errorless computation an unreasonable one. Instead, one can ask if these provable separations persist if the quantum device is subject to random noise. As a first step toward answering this question, Bravyi, Gosset, König, and Tomamichel [Bra+20] proved that a separation between QNC^0 and NC^0 circuits persists even if the quantum circuit is subject to *local stochastic noise*. Under this model, random Pauli errors manipulate the state at every step of the circuit.

However, NC^0 circuits are fairly weak because their bounded fan-in property means that each output bit can depend only on a constant number of input bits. For example, the logical

AND function on all inputs cannot be computed by NC^0 circuits. We aim to address this shortcoming by increasing the power of classical circuits in question and probe their power relative to noisy quantum circuits.

1.1 Main results

We show that *noisy* shallow quantum circuits (SQCs) demonstrate unconditional advantage over stronger classical circuit classes than previously known. In particular, we show that a a noisy SQC can solve an interactive problem with probability quickly approaching 1 while $\mathsf{AC}^0[p]$ circuits (for any prime $p \geq 2$) must fail on at least $\frac{1}{30}$ problem instances. Furthermore, assuming plausible complexity-theoretic conjectures, we give evidence that log-space classical machines cannot solve problems that noisy SQCs can solve.

To create the separations, we prove new classical lower bounds that are summarized in Table 1, along with previous results that are relevant for the separations.

Type of problem	Relation	Interactive	Interactive	Interactive
Success rate	$\exp(-n^{\Omega(1)})$	29/30	105/106	420/421
Complexity of classical solution	not in AC^0 [Ben+19]	NC^1-hard Theorem 4.3 [GS20]	w-PBP-hard Theorem 4.4	$\oplus\mathsf{L}$-hard Theorem 4.5

Table 1: Summary of classical hardness results. If a classical machine solves the respective problem with rate at least the value given in the middle row, then its complexity can be described by the bottom row.

As expected, the main difficulty is in proving the classical lower bounds. Noisy quantum circuits cannot be expected to solve problems with absolute certainty, so we must show that our problems are more than worst-case hard for classical machines to make a fair comparison between the two models. If one would like to compare the separations between *noise-free* SQCs and classical machines from Table 1, then (1) the $\oplus\mathsf{L}$ result also promotes the worst-case result by [GS20] to average-case and (2) the w-PBP result shows that larger width grids progressively solve harder problems, assuming larger width in permutation branching programs grants greater power.

Much of the paper is dedicated to proving the w-PBP and $\oplus\mathsf{L}$ average-case hardness. These results are mainly worst- to average-case reductions starting from the interactive protocols in [GS20]. More precisely, these results are basically random self-reductions that work by hiding problem instances from the oracle in the reductions. For the w-PBP result, we achieve this by utilizing an approximate permutation sampling technique introduced by [MV91]. For the $\oplus\mathsf{L}$ result, we utilize a technique from the theory of cryptography, randomized encodings [AIK06], to hide problem instances while in the very weak circuit class NC^0. To show the quantum upper bounds, we rely on a constant-depth, noise-tolerant construction introduced by Bravyi, Gosset, König, and Tomamichel [Bra+20]. We tune their proofs slightly to work for interactive problems.

Result 1. *Let $\mathcal{I}$ be any problem appearing in Table 1. A noisy, constant-depth Clifford circuit can solve $\mathcal{I}$ with probability at least*

$$1 - \exp(-\Omega(\mathsf{polylog}(n)))$$

over all inputs when the noise rate is below a certain constant threshold.

This gives us several corollaries. Let us denote $p_n = 1 - \exp(-\Omega(\mathsf{polylog}(n)))$ for the success rate of the noisy SQC.

Corollary 1.1. *(Noisy AC^0 separation for relation problem) There is a relation problem solved with probability p_n by a noisy SQC over all inputs, but any AC^0 circuit cannot solve the problem with probability exceeding $\exp(-n^\alpha)$ over a uniform input for some $\alpha > 0$.*

Corollary 1.2. *(Noisy $\mathsf{AC}^0[p]$ separation for interactive problem) There is an interactive problem solved with probability p_n by a noisy SQC over all inputs, but any $\mathsf{AC}^0[p]$ circuit (for primes $p \geq 2$) cannot solve the problem with probability exceeding $\frac{29}{30}$ over a uniform input.*

Corollary 1.3. *(Conditional noisy L separation) Assume $\oplus\mathsf{L/poly} \not\subseteq (\mathsf{qAC}^0)^{\mathsf{L}}$. There is an interactive problem solved with probability p_n by a noisy SQC over all inputs, but any log-space machine cannot solve the problem with probability exceeding $\frac{420}{421}$ over a uniform input.*

Let us briefly argue for the $\oplus\mathsf{L/poly} \not\subseteq (\mathsf{qAC}^0)^{\mathsf{L}}$ conjecture. A well-known result in circuit complexity is that the parity function is not in qAC^0. At the same time, it is believed that $\oplus\mathsf{L/poly} \not\subseteq \mathsf{L/poly}$. A YES problem instance for a $\oplus\mathsf{L/poly}$ machine must induce an odd parity of accepting paths in an NL machine, and NO instances must induce even parity of accepting paths. The $\oplus\mathsf{L/poly} \not\subseteq \mathsf{L/poly}$ conjecture says that this ability to calculate the parity of such paths isn't possible in L, and it is likely that qAC^0 circuits do not bridge this gap because they cannot calculate parity.

2 Preliminaries

In this section, we define the classical low-depth circuit classes, delineate relation and interactive problems, and explain the noise model we employ.

2.1 Low-depth circuit classes

Circuits for solving certain problems are generally defined as families of circuits, one for each input size, where the corresponding circuits are

1. NC^i: $\log^i$-depth bounded fan-in AND/OR/NOT circuits
2. AC^i: NC^i circuits with unbounded fan-in gates
3. $\mathsf{AC}^i[p]$: AC^i circuits with MOD_p gates
4. TC^i: AC^i circuits with majority gates

5. $\mathsf{BP}\mathcal{C}$: $\mathcal{C}$ circuits that have access to random bits and two-sided bounded error

6. $\mathsf{q}\mathcal{C}$: $\mathcal{C}$ circuits of $\exp\left(\log^{O(1)} n\right)$ size

In addition, we have the following inclusions that are proven strict ($\subsetneq$) and believed to be strict ($\subset$): $\mathsf{NC}^0 \subsetneq \mathsf{AC}^0 \subsetneq \mathsf{AC}^0[p] \subsetneq \mathsf{TC}^0 \subset \mathsf{NC}^1 \subset \mathsf{L} \subset \oplus\mathsf{L}$.

2.2 Relation and Interactive problems

A relation problem begins with a relation $R \subseteq \{0,1\}^n \times \{0,1\}^m$ where m is polynomially related to n. A computational device is given some input string $x \in \{0,1\}^n$ and is asked to output any $y \in \{0,1\}^m$ such that $(x,y) \in R$. There could be multiple candidates for y, and the device is only judged on whether it is a possible candidate. We use the notation $R(x,y) = 1$ to indicate that (x,y) satisfies the relation problem. Otherwise, we say that $R(x,y) = 0$.

An interactive problem occurs between the question asker (challenger) and computational device over multiple rounds of back-and-forth messages. We focus on a two-round interactive protocol, so the challenger first sends some input to the device, and the device responds with its first answer. Then the challenger sends another round of input and the device outputs another answer. The device output is deemed correct if both rounds of outputs have correct answers.

2.3 Local Stochastic Noise Model

While a noise-free quantum computation can reliably execute a sequence of operations, a noisy quantum computation may have sources of errors that corrupt several key parts of the computation including state initialization, gate execution, and measurement. To capture these sources of error, we consider the *local stochastic quantum noise* model [FGL18; Bra+20]. Under this model, random errors occur at each timestep of the execution of a quantum circuit. For example, a gate error occurs when random noise enters the computation prior to the execution of the gate. Similarly, an erroneous measurement outcome is modeled by random noise affecting the state of the system right before measurement.

The types of random noise that we consider are random Pauli errors of the form $\{I,X,Y,Z\}^{\otimes n}$. For a Pauli error E, we borrow the convention of $\mathrm{Supp}(E) \subseteq [n]$ to denote the subset of indexed qubits for which E acts by a X, Y, or Z. In other words, $\mathrm{Supp}(E)$ is the subset of qubits on which E acts non-trivially. We associate local stochastic noise with a parameter p, which is the noise rate, and arrive at the following definition:

Definition 2.1. *Let $p \in [0,1]$. A random n-qubit Pauli error E is "p-local stochastic" if*

$$\Pr[F \subseteq \mathrm{Supp}(E)] \leq p^{|F|} \quad \forall\, F \subseteq [n] \tag{1}$$

When we say that a layer of local stochastic noise E is sampled with noise rate p, we use the notation $E \sim \mathcal{N}(p)$. There is a property of local stochastic noise that will be useful for our analysis.

Lemma 2.1. *Suppose $E \sim \mathcal{N}(p)$ and E' is another Pauli error such that $Supp(E) \subseteq Supp(E)$ with certainty. Then $E' \sim \mathcal{N}(p)$.*

2.4 The 2-D surface code

The 2D surface code is a CSS-type error correcting code that encodes one logical qubit into m physical qubits on a 2D lattice. For a detailed discussion of the construction that we utilize, we refer the reader to Section IV of [Bra+20]. We will abstract away the physical surface code and henceforth denote the encoded version of a state with a line over the state vector, e.g. the logical $|0\rangle$ state is encoded as the $|\overline{0}\rangle$ state. We follow the same convention when speaking of encoded circuits and measurement observables. If $\mathcal{Y}$ is the physical measurement outcome over multiple codeblocks, we denote $\mathcal{Y} = \mathcal{Y}^1 ... \mathcal{Y}^n$ with each $\mathcal{Y}^i$ the m outcomes of the i'th codeblock. The space of binary physical measurement outcomes on one codeblock forms a linear subspace called the codespace, and we refer to it by $\mathcal{L}$.

A standard quantum computation begins with qubits prepared in a basis state, e.g. multiple copies of $|0\rangle$. However, the surface code must begin with an *encoded* basis state, $|\overline{0}\rangle$, so we require a constant-depth procedure to produce such a state. We can do this, albeit at the cost of extra ancilla qubits and a Pauli recovery operator dependent on the measurement outcomes of the ancillae. We also opt for the encoded Bell state, $|\overline{\Phi}\rangle$, as the starting state.

Lemma 2.2. *(Basis state preparation, Theorem 23 in [Bra+20]) There is a constant-depth Clifford circuit on $2m + m_{anc}$ qubits that measures m_{anc} qubits with measurement outcome $s \in \{0,1\}^{m_{anc}}$ and leaves the remaining $2m$ qubits in the state $Rec(s)\left|\overline{\Phi}\right\rangle$ for some Pauli operator Rec completely determined by s.*

Following basis-state preparation, we would like to perform a constant-depth Clifford circuit on the surface code.

Lemma 2.3. *(Constant-depth Cliffords, Lemma 20 in [Bra+20]) The encoded $\overline{H}$, $\overline{S}$, $\overline{CNOT}$ gates have constant depth implementations on the surface code.*

In particular, an unencoded Clifford circuit can be transformed to an encoded Clifford circuit on the surface code with only constant overhead. If we add local stochastic noise to a Clifford circuit, we can propagate the errors to the end of the circuit.

Lemma 2.4. *(Propagating noise, Theorems 17 and 23 in [Bra+20]) Suppose we have a quantum circuit with noise rate p that creates multiple $Rec(s)\left|\overline{\Phi}\right\rangle$ states using Lemma 2.2. Then suppose that it performs a depth-D Clifford circuit on these states. The state of the system is equivalent to a noiseless computation with only one layer of local stochastic noise, E, following the circuit such that $E \sim \mathcal{N}(O(p^{2^{-O(D)}}))$.*

Suppose that we have finished performing an encoded Clifford circuit on a surface code like Lemma 2.4 describes. We could express the final layer of local stochastic noise as $X(v)Z(w)$ where $X(v)$ is a Pauli X only on qubits with their corresponding bit in v set to 1, and similarly for $Z(w)$. If we measured one surface code, we would have a length-m bitstring $\mathcal{Y}$ that encodes the measurement outcome of a logical qubit. The Dec function decodes the m outcomes to a single bit, and it is able to tolerate noise.

Lemma 2.5. *(Lemma 21 in [Bra+20]) Suppose that there is only one layer of local stochastic noise $X(v) \sim \mathcal{N}(r)$ with $r \leq 0.01$ which occurs right before the measurement of any codeblock. Then*

$$\Pr_{X(v)\sim\mathcal{N}(r)}[Dec(x \oplus v) = Dec(x)] \geq 1 - \exp\left(-\Omega(m^{1/2})\right) \tag{2}$$

for any $x \in \mathcal{L}$.

This lemma says that when the layer of local stochastic noise is below a constant threshold of 0.01, then *Dec* will successfully decode the surface code measurement outcome for any x in the codespace.

3 The noisy extension and AC^0 separation

In this section, we review the noisy extension and its application to any relation problem solved by a constant-depth Clifford circuit. The application of the noisy extension to the Relaxed Parity Halving Problem is straightforward, and we prove both the noisy quantum upper bound and classical lower bound.

3.1 The noisy extension

Suppose that we have a relation problem defined by $R \subseteq \{0,1\}^* \times \{0,1\}^*$ that is solved with certainty by a classically-controlled Clifford circuit C_x that begins with multiple copies of $|\Phi\rangle$ and measures all qubits as output. To make this relation noise-tolerant, we convert R to its *noisy-extended* version. The noisy-extended version is defined using a 2D surface code with desirable properties. Its effect on a relation problem is the following: For some input x, the set of y such that $(x,y) \in R$ is enlarged to the set of $\mathcal{Y}$ that decode to y.

However, recall that our procedure for basis state preparation on the surface code incurs an additional Pauli operator $Rec(s)$. It turns out that the effect of this Pauli operator on the overall quantum computation can be managed by some classical post-processing. To illustrate this, we need to know how $Rec(s)$ propagates through a classically-controlled Clifford circuit $\overline{C_x}$. By the definition of Cliffords, we can define $f(s,x)$ and $h(s,x)$ by

$$X(f)Z(h) \sim \overline{C_x} Rec(s) \overline{C_x}^{\dagger} \tag{3}$$

where $f = f^1 ... f^n$ and each f^i is m bits describing the Pauli X operator on the i'th codeblock of the circuit. We are now ready to define the noisy-extended relation.

Definition 3.1. *The noisy-extended relation problem, R', associated with a relation problem, R, is the following:*

$$R'(x,(\mathcal{Y},s)) = \begin{cases} 1 & \text{if } R(x,y)=1 \text{ for } y_i = Dec(\mathcal{Y}^i \oplus f^i(s,x)) \quad \forall\, i \in [n] \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

It should be the case that if a *noiseless* quantum circuit can solve a certain relation problem with certainty, then another noiseless quantum circuit can solve the related noisy-extended relation problem with certainty.

Suppose that a quantum circuit has prepared the encoded basis state by measuring syndrome outputs $s^1 ... s^n$, resulting in the state $(Rec(s^1) \otimes ... \otimes Rec(s^n)) |\overline{\Phi}^n\rangle = Rec(s) |\overline{\Phi}^n\rangle$ through the procedure of Lemma 2.2. Then it performs a classically-controlled Clifford circuit $\overline{C_x}$ in constant-depth using Lemma 2.3 and measures the output $\mathcal{Y}$ with the property

$$|\langle \mathcal{Y}| \overline{C_x} Rec(s) |\overline{\Phi}^n\rangle|^2 > 0 \tag{5}$$

By propagating the Pauli $Rec(s)$ over Clifford circuits using Equation (3), we have that $|\langle \mathcal{Y}| X(f)Z(h)\overline{C_x}|\overline{\Phi}^n\rangle|^2 > 0$. Z-type Paulis at the end of a circuit have no effect on Z-basis measurement, so we similarly have that

$$|\langle \mathcal{Y} \oplus f(s,x)|\,\overline{C_x}\,|\overline{\Phi}^n\rangle|^2 > 0 \tag{6}$$

Because $\overline{C_x}$ is the encoded version of C_x, then the bit string y defined by

$$y_i = Dec(\mathcal{Y}^i \oplus f^i(s,x))\ \forall i \in [2n] \tag{7}$$

satisfies the original relation R.

If we consider the effect of noise on constant depth Clifford circuits, the noisy extension remains resilient to noise.

Theorem 3.1. *(Noise-tolerance of noisy extension, Theorem 17 in [Bra+20]) Suppose a constant-depth Clifford circuit solves a relation problem R. Let the noisy extension of R be defined as R' in Definition 3.1. Then another constant depth-D Clifford circuit solves R' with probability $\geq 1 - \exp(-\Omega(m^{1/2}))$ when the noise rate is below $\exp(-\exp(O(D)))$.*

3.1.1 Exponential size AC^0 decoding circuit

The noisy extension of Definition 3.1 allows a *noisy* constant-depth quantum circuit to solve a certain modified relation problem with high probability. However, in order to demonstrate a separation between the quantum and classical circuit's capabilities, we need to argue that the modified relation problem remains hard for classical circuits. Our goal is to prove hardness for circuits at least as powerful as AC^0. If the classical circuits could decode the surface code output of the modified relation problem using AC^0-type gates, then they would have output for the original relation problem and remain in the same class of circuits. This is the type of reduction for which we aim.

Recall that the surface code encoded quantum circuit will output $\mathcal{Y}$ and s such that the bitstring y defined by Equation (7) is the output of an unencoded circuit. We show that decoding can be carried out by an $\exp(m, m_{anc})$ size AC^0 circuit[1]. The proof proceeds in two steps: We show that $f^i(s,x)$ for all i can be computed in size $\exp(m_{anc})$, and we compose this with a truth-table circuit for Dec of size $\exp(m)$. The construction is shown in Figure 1 for convenience.

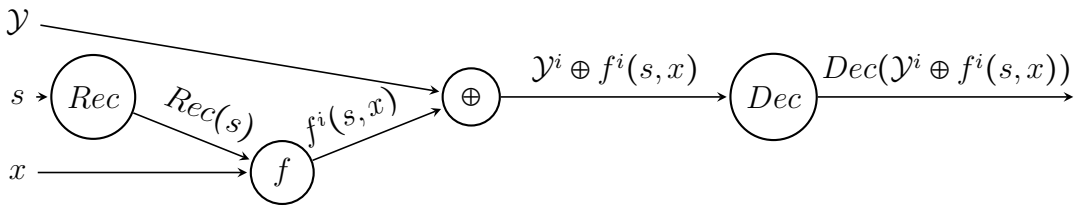


Figure 1: Decoding gadget

[1]In fact, this size is nearly optimal for AC^0 circuits because computing the parity of n bits reduces (using only fan-out) to Dec on $O(n^2)$ bits.

Lemma 3.1. *Let $m_{anc} = \mathsf{polylog}(n)$. Given s and x, $f^i(s,x)$ for every i, defined by Equation (3), can be computed by a constant depth AC^0 circuit of size $\exp(m_{anc})$.*

Proof. Let $s^j \in \{0,1\}^{m_{anc}}$ be the syndrome measurement outcomes for the j'th codeblock. The m-qubit Pauli recovery operators $Rec(s^j)$ for all j can be computed by a constant-depth truth table AC^0 circuit of size $\exp(m_{anc})$. We would like to conjugate the Pauli $Rec(s) = Rec(s^1) \otimes ... \otimes Rec(s^n)$ with the classically-controlled Clifford circuit, $\overline{C_x}$, to obtain $f^i(s,x)$ in Equation (3).

Consider any depth-one Clifford circuit, C, on mn qubits composed of one- and two-qubit gates. Conjugating the Pauli operator $Rec(s)$ by C is locally computable by a polynomial size AC^0 circuit because C only uses one- and two-qubit Clifford gates. Repeating this process for each layer of a constant-depth Clifford circuit $\overline{C_x}$ results in a polynomial size circuit for computing $f^i(s,x)$ from $Rec(s)$. Combining the $\exp(m_{anc})$ size circuit for $Rec(s)$ with the polynomial size conjugation circuit, we finish the construction for the circuit. □

Equipped with a circuit for computing $f^i(s,x)$ for each i, we compose this circuit with a Dec circuit to finish decoding the output.

Lemma 3.2. *Let $m, m_{anc} = \mathsf{polylog}(n)$. Given $\mathcal{Y}$, s, and x, $Dec(\mathcal{Y}^i \oplus f^i(s,x))$ for all codeblocks i can be computed by a constant-depth AC^0 circuit of size $\exp(m_{anc}, m)$.*

Proof. We start with the circuit of Lemma 3.1 to obtain $f^i(s,x)$ for all codeblocks $i \in [n]$. The circuit efficiently computes bitwise $\mathcal{Y}^i \oplus f^i(s,x)$ for all i efficiently. Dec is a function from m bits to 1 bit, so the circuit uses a truth table AC^0 circuit of size $\exp(m)$ to compute $Dec(\mathcal{Y}^i \oplus f^i(s,x))$ for all i. In total, this circuit has size $\exp(m, m_{anc})$. □

In summary, we have computed $Dec(\mathcal{Y}^i \oplus f^i(s,x))$ for each i with a $\exp(m, m_{anc})$ size AC^0 circuit.

3.2 Noise-tolerant AC^0 separation

We begin by reviewing the relevant problem that is solvable by *noiseless* QNC^0 circuits but is hard for AC^0 circuits to solve. By applying the noisy extension introduced in the previous section, we can prove that a separation persists as the quantum circuit is subject to noise.

3.2.1 Noisy-extended Parallel Grid-RPHP

It was shown in [Ben+19] that there is a problem that is solved with certainty by a QNC^0 circuit but is hard for AC^0 circuits to solve. The problem, Parallel Grid-RPHP, is a relation problem with inputs uniformly chosen from a set $P_n \subseteq \{0,1\}^n$ for all n that is solved with certainty by a constant-depth Clifford circuit. In other words, Parallel Grid-RPHP is a promise relation problem.

Lemma 3.3. *(Parallel Grid-RPHP quantum upper bound) There is a constant-depth Clifford circuit that solves Parallel Grid-RPHP over all inputs $x \in P_n$ with certainty.*

The Parallel Grid-RPHP problem is hard for AC^0 circuits with an error rate exponentially close to 1.

Lemma 3.4. *(Theorem 26 in [Ben+19]) Any* AC^0 *circuit of size* s *and depth* d *cannot solve Parallel Grid-RPHP with probability exceeding*

$$\exp\left(\frac{-n^{1/2-o(1)}}{O(\log s)^{2d}}\right) \tag{8}$$

over a uniformly random input $x \in P_n$.

Our goal is to show that the noisy-extended version of Parallel Grid-RPHP can be solved often by noisy QNC^0 circuits but fail often for AC^0 circuits. We will apply the noisy extension of Definition 3.1 directly to the output of Parallel Grid-RPHP. Specifically, we will define an output of the noisy-extended version to be correct if upon decoding matches a correct output of Parallel Grid-RPHP.

Definition 3.2. *(Parallel NoisyGrid-RPHP) Let* $m, m_{anc} = \mathsf{polylog}(n)$. *Given* $x \in P_n$, *output the noisy-extended (defined in Definition 3.1) output of Parallel Grid-RPHP.*

A noisy quantum circuit can solve Parallel NoisyGrid-RPHP with high probability.

Theorem 3.2. *Let* D *be a constant and the local stochastic noise rate* p *be bounded by* $p < p_{th} = \exp(-\exp(O(D)))$. *For all* $x \in P_n$, *a depth* D *Clifford circuit with noise rate* p *can solve Parallel NoisyGrid-RPHP with probability exceeding*

$$1 - \exp(-\Omega(\mathsf{polylog}(n))) \tag{9}$$

Proof. Because the circuit for Parallel Grid-RPHP is a constant-depth, classically-controlled Clifford circuit for a relation problem, this follows directly from Theorem 4.2. □

We now ask how well a classical AC^0 circuit can do on this task. This proof follows from a reduction to the output of Parallel Grid-RPHP.

Theorem 3.3. *Any* AC^0 *circuit of size* s *and depth* d *cannot solve Parallel NoisyGrid-RPHP with probability exceeding*

$$\exp\left(\frac{-n^{1/2-o(1)}}{O(\log(s + \exp(\mathsf{polylog}(n))))^{2d+O(1)}}\right) \tag{10}$$

over the uniformly random $x \in P_n$.

Proof. We decode the output of Parallel NoisyGrid-RPHP to one of Parallel Grid-RPHP using Lemma 3.2. This incurs an extra size overhead of the AC^0 circuit by $\exp(m, m_{anc}) = \exp(\mathsf{polylog}(n))$. By applying this size expansion to Lemma 3.4, we arrive at the desired bound. □

This concludes the proof of separation between noisy QNC^0 and noiseless AC^0 circuits.

4 Interactive hardness

4.1 NC^1-hardness

We now consider an interactive task that a noisy quantum circuit can solve with high probability but if a classical circuit solved would be necessarily deciding a NC^1-hard problem. Roughly speaking, this is how we will show that a $\mathsf{AC}^0[p] \subsetneq \mathsf{NC}^1$ circuit cannot solve such a problem.

A result by Barrington and Thérien is that computing products in non-solvable groups is NC^1-hard [BT88]. Denoting the set of 2-qubit Cliffords modulo 2-qubit Pauli operations $\{\mathcal{C}_2/\mathcal{P}_2\}$, Grier and Schaeffer used this fact to show the following:

Theorem 4.1. *(Corollary 15 in [GS20]) Let $C_1, ..., C_n \in \{\mathcal{C}_2/\mathcal{P}_2\}$. Promised that the product $C_1...C_n \in \{I \otimes I, H \otimes H\}$ modulo Pauli operations, deciding whether $C_1...C_n$ is $I \otimes I$ or $H \otimes H$ is NC^1-hard.*

Distinguishing between $I \otimes I$ or $H \otimes H$ (up to Paulis) straightforwardly recasts to deciding whether $C_1...C_n$ leaves the $|{+}{+}\rangle$ state invariant or transforms it to $|00\rangle$ modulo Pauli operations. A constant-depth quantum circuit certainly cannot perform such a sequence of n Clifford gates serially, or it would have linear depth. Instead, we appeal to *measurement-based quantum computation* (MBQC) to *imperfectly* parallelize the procedure.

4.1.1 Measurement-based Quantum Computation

MBQC begins with a cluster state on a 2D grid [RB01; RBB03]. In our setting, we are interested in the case when the cluster is the brick-like pattern of Figure 2. This pattern, as described in [GS20], is not needed for correctness, but it is a smaller cluster state than those appearing in previous results.

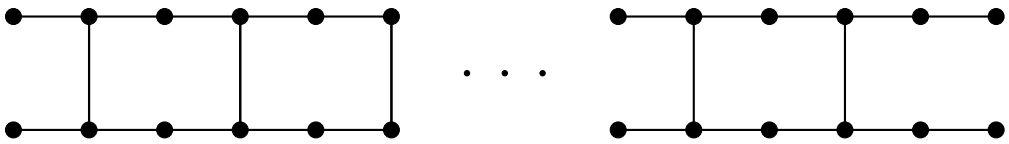


Figure 2: Cluster state $|\mathcal{H}_n\rangle$

We denote the cluster state associated with this pattern as $|\mathcal{H}_n\rangle$. $|\mathcal{H}_n\rangle$ has $16n+4$ vertices split evenly between two layers. Given a sequence of 2-qubit Clifford gates, $C_1, ..., C_n$, one can measure the leftmost $2 \times (8n+1)$ sub-grid in X and Y bases where the measurement basis of a single qubit is dependent on at most a single C_i. MBQC says that with the correct set of measurements, the remaining rightmost qubits are in the state $PC_1...C_n|{+}{+}\rangle$ for some 2-qubit Pauli operator P determined by the measurement outcomes of the MBQC procedure.

This leads to a natural way to prepare the state $PC_1...C_n|{+}{+}\rangle$ in constant depth. One prepares $|\mathcal{H}_n\rangle$ in constant depth, then measures all but a constant number of qubits of the cluster state, resulting in the state $PC_1...C_n|{+}{+}\rangle$. This procedure takes place in the first round of the interactive protocol.

After preparing the state $PC_1...C_n|{+}{+}\rangle$, one can perform 2-qubit Pauli measurements to learn some information about the state. The challenger probes for this information in the second round of the protocol. We first describe the noise-free version of the protocol, prove an average-case hardness of the protocol, and then define its noisy extension.

Problem 1. *(CliffSim[2], Problem 12 in [GS20]) The circuit begins with the cluster state $|\mathcal{H}_n\rangle$, where we define $\{Q_{i,j}\}_{i,j}$ to be the grid of $|\mathcal{H}_n\rangle$ qubits. The challenger has input $A \in \{0,1\}^{2\times(8n+1)}$ and $B \in \{0,1\}^{2\times 9}$. The protocol proceeds in two stages:*

1. *In the first round, the challenger sends A to the circuit. For each $i \leq 2, j \leq 8n+1$, if $A_{i,j} = 1$, then the circuit measures qubit $Q_{i,j}$ in the X-basis. Otherwise it measures $Q_{i,j}$ in the Y-basis. The circuit returns the measurement results to the challenger.*

2. *In the second round, the challenger sends B to the circuit. For each $i \leq 2, j \leq 9$, if $B_{i,j} = 1$, then the circuit measures qubit $Q_{i,(8n+1)+j}$ in the X-basis. Otherwise it measures $Q_{i,(8n+1)+j}$ in the Y-basis. The circuit returns the measurement results to the challenger.*

A noise-free quantum circuit of constant depth can solve this problem with certainty by simply following the protocol exactly. Suppose that a classical machine of some complexity class $\mathcal{C}$ solves the same problem. For many cases, if the interactive protocol can be implemented in $\mathcal{C}$ (with error rate ϵ), then the *rewind oracle*, which is explained in the next paragraph, can be implemented in $\mathcal{C}$ (with error rate ϵ). This is true for several classes of interest including AC^0, $\mathsf{AC}^0[p]$, TC^0, NC^1, L, and $\oplus\mathsf{L}$. There are two ways to query the rewind oracle in analogy to the two rounds of CliffSim[2].

Definition 4.1. *(CliffSim[2] Rewind Oracle, [GS20])*

1. *In the first round, the challenger sends A to the rewind oracle. The rewind oracle returns results consistent with measuring in the given bases.*

2. *In the second round, the challenger sends A, measurement results corresponding to A in the first round, and B to the rewind oracle. The rewind oracle returns results consistent with measuring the remaining qubits in the given bases, conditioned on the given measurement results of A in the first round.*

We will often call the rewind oracle $\mathcal{R}$. An important property of the rewind oracle is the fact that different second round measurements can be performed while conditioning on the same measurement outcomes in the first round. This property leads to the following result:

Lemma 4.1. *(Theorem 24 in [GS20]) There is a procedure carried out by a randomized AC^0 circuit that samples a uniformly random input in $\{\mathcal{C}_2/\mathcal{P}_2\}^n$ to $\mathcal{R}$ in the first round and repeats the second round with six different 2-qubit Pauli measurements. The procedure outputs a uniformly random nonstabilizer or uniformly random nontrivial stabilizer of $C_1...C_n|{+}{+}\rangle$. If $\mathcal{R}$ had no errors in the second round measurements, then it is the former.*

4.1.2 NoisyCliffSim[2] protocol

We are now ready to define the 2-round interactive protocol, NoisyCliffSim[2], solved by a noisy constant-depth quantum circuit with high probability. Recall the noisey extension in Definition 3.1. Though it is defined over relation problems, we can also adapt it to interactive problems in a similar way. The challenger provides the same questions to the prover, and the prover outputs both an encoded output and syndrome measurements. For convenience, we design the protocol to output syndrome measurements in the first round.

Suppose that the quantum circuit has prepared the $\left|\overline{\mathcal{H}_n}\right\rangle$ state with some Pauli recovery operator, $Rec'(s)\left|\overline{\mathcal{H}_n}\right\rangle$, and would like to follow an encoded CliffSim[2] measurement protocol on the state. This requires a constant-depth Clifford circuit, $\overline{C_x}$, which changes measurement bases to follow the challenger's input over two rounds. Following measurement bases changes, the quantum circuit measures the corresponding qubits along the Z-basis over the two rounds.

The Pauli operator $Rec'(s)$ can be propagated beyond $\overline{C_x}$ like the noisy extension.

$$\overline{C_x}Rec'(s)\left|\overline{\mathcal{H}_n}\right\rangle \sim X(f)Z(h)\overline{C_x}\left|\overline{\mathcal{H}_n}\right\rangle \tag{11}$$

where $f = f^1...f^{16n+4}$ and each f^i is over the surface code of the i'th logical qubit of the $\left|\overline{\mathcal{H}_n}\right\rangle$ state. Some of the f^i components of f will belong to surface code qubits of the first round, so we will give them a subscript f^i_1 for convenience. Similarly, we will denote f^i_2 for those in the second round. We are now ready to define NoisyCliffSim[2]. Note that the following description of NoisyCliffSim[2] is simply the noisy extension in Definition 3.1 applied to CliffSim[2], but we describe it in detail to be explicit about the necessary steps of the protocol.

Definition 4.2. (NoisyCliffSim[2]) *Let $m, m_{anc} = \mathsf{polylog}(n)$. Additionally, let $A \in \{0,1\}^{2\times(8n+1)}$ and $B \in \{0,1\}^{2\times 9}$ be binary matrices. Let $s = s^1...s^{16n+4}$ be syndrome measurements from Bell state creation in Lemma 2.2. Then the protocol proceeds in two rounds:*

1. *Challenger sends A to the prover. Prover measures logical qubit (i,j) in $\overline{X}$-basis if $A_{ij} = 0$. Otherwise, prover measures in $\overline{Y}$-basis. Afterwards, the prover sends measurement outcomes $\mathcal{Y}_1 \in \{0,1\}^{[2\times(8n+1)]m}$ and $s \in \{0,1\}^{(16n+4)m_{anc}}$ to the challenger.*

2. *Challenger sends B to the prover. Prover measures logical qubit $(i, j+8n+1)$ in $\overline{X}$-basis if $B_{ij} = 0$. Otherwise, prover measures in $\overline{Y}$-basis. Afterwards, the prover sends measurement outcome $\mathcal{Y}_2 \in \{0,1\}^{(2\times 9)m}$ to the challenger.*

The prover succeeds if and only if (y_1, y_2) is a correct output for Problem 1 where

$$y_{1,i} = Dec(\mathcal{Y}^i_1 \oplus f^i_1(s,x)) \tag{12}$$

$$y_{2,i} = Dec(\mathcal{Y}^i_2 \oplus f^i_2(s,x)) \tag{13}$$

Note that A corresponds to a sequence of n 2-qubit Clifford gates encoded in a MBQC measurement pattern and B corresponds to basis changes for Pauli measurements of 2 qubits. Following the interactive protocol, we define the rewind oracle for NoisyCliffSim[2] as the noisy-extended version of Definition 4.1 with the same decoding properties as NoisyCliffSim[2].

4.1.3 Quantum upper bound

Let us show that a noisy quantum circuit can solve NoisyCliffSim[2]. More generally, consider any protocol that consists of two rounds with a constant-depth Clifford circuit, from which NoisyCliffSim[2] follows.

Theorem 4.2. *Let $\mathcal{I}$ be any interactive problem between a challenger and classically-controlled Clifford circuit on n qubits that requires the circuit to do the following:*

1. *Perform a constant-depth Clifford circuit $\mathcal{C}_{pre}$ on $|0^n\rangle$.*
2. *In the first round, return results that appear with nonzero probability when measuring a subset of qubits in X/Y bases given by the challenger.*
3. *In the second round, return results that appear with nonzero probability when measuring the remaining qubits in X/Y bases given by the challenger, conditioned on the results of the second round.*

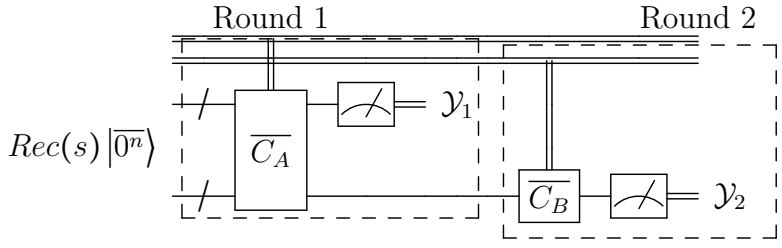
Let $\mathcal{I}'$ be the noisy extension (Definition 3.1) of $\mathcal{I}$ where each logical qubit is encoded in $m \geq \Omega(\mathsf{polylog}(n))$ physical qubits. Then another noisy constant depth-D Clifford circuit can pass $\mathcal{I}'$ with probability

$$\geq 1 - \exp(-\Omega(\mathsf{polylog}(n)))$$

when the noise rate p is bounded by

$$p < p_{th} = \exp(-\exp(O(D)))$$

Proof. Let us define $\mathcal{C}$ to be the classically-controlled Clifford circuit that executes the protocol $\mathcal{I}'$. The circuit $\mathcal{C}$ prepares $Rec(s)\,|\overline{0^n}\rangle$[2], where $Rec(s) := Rec(s^1) \otimes ... \otimes Rec(s^n)$, and performs $\overline{\mathcal{C}_{pre}}$ on this state in constant depth by the basis state preparation in Lemma 2.2 and constant-depth Cliffords in Lemma 2.3. For the first and second rounds of the protocol, $\mathcal{C}$ is asked for X/Y basis measurement results, so we w.l.o.g assume that the circuit performs basis change Cliffords $\overline{\mathcal{C}_A}$ and $\overline{\mathcal{C}_B}$ for the first and second rounds, respectively, and measures in the Z basis. This can be seen in the following figure:



Let us analyze the effect of noise in this process. By propagating noise via Lemma 2.4 and combining noise via Lemma 2.1, the noisy computations are equivalent to single layers of noise before first and second round measurements with noise rate $p^{\exp(-O(D))}$ where D is the depth of $\mathcal{C}$. The measurement success rate given by Lemma 2.5 implies that if the single layer of local

[2]The states $|\mathcal{H}_n\rangle$ and $|0^n\rangle$ are related by a constant-depth Clifford, so we opt for the description using $|\overline{0^n}\rangle$ for simplicity.

stochastic noise before measurement has noise rate below 0.01, then a codeblock outputs a correct measurement outcome with probability exceeding $1-\exp(-\Omega(m^{1/2}))$. We only require that $p < \exp(-\exp(O(D)))$ to achieve the 0.01 noise rate bound before measurements. By noting $m \geq \Omega(\mathsf{polylog}(n))$ and a union bound, we can conclude that all n codeblocks are correct with probability at least $1 - \exp(-\Omega(\mathsf{polylog}(n)))$.

□

Immediately, we get the quantum upper bound.

Corollary 4.1. *There is a quantum circuit of constant depth D that passes the NoisyCliffSim[2] protocol with probability at least $1 - \exp(-\Omega(\mathsf{polylog}(n)))$ for all possible inputs when the noise rate p is bounded by $p < p_{th} = \exp(-\exp(O(D)))$.*

4.1.4 Classical hardness

We first show that a $(\mathsf{BPAC}^0)^{\mathcal{R}}$ circuit can decide a NC^1-hard problem even if the CliffSim[2] rewind oracle $\mathcal{R}$ fails on $\frac{1}{30}$ possible inputs. This is a slight improvement over the $\frac{2}{75}$ present in [GS20]. Then we reduce a NoisyCliffSim[2] rewind oracle's output to CliffSim[2] to prove hardness for NoisyCliffSim[2].

Lemma 4.2. *Let $\mathcal{R}$ be the CliffSim[2] rewind oracle that returns incorrect outputs with probability $\epsilon < \frac{1}{30}$ for a uniformly random input input. That is, it may output incorrect answers for Cliffords in the first round and (six) Pauli measurements in the second round. Then*

$$\mathsf{NC}^1 \subseteq (\mathsf{BPAC}^0)^{\mathcal{R}} \tag{14}$$

Proof. If $\mathcal{R}$ fails for ϵ fraction of all possible inputs, then at most $6\epsilon \cdot |\{\mathcal{C}_2/\mathcal{P}_2\}^n|$ inputs will fail. In other words, at most 6ϵ fraction of inputs in the first round may lead to incorrect output in the second round. By following Lemma 4.1, we have that for any nontrivial stabilizer, S, of $C_1 ... C_n |{+}{+}\rangle$,

$$\Pr[\text{output } S] < 6\epsilon \cdot \frac{1}{3} \tag{15}$$

and for any nonstabilizer N,

$$\Pr[\text{output } N] \geq (1 - 6\epsilon) \cdot \frac{1}{12} \tag{16}$$

By choosing $\epsilon < \frac{1}{30}$, we have that Equation (15) and Equation (16) are bounded above by $1/15$ and below by $1/15$, respectively. In fact, the upper and lower bounds have a constant difference determined by ϵ, so a BPAC^0 circuit can learn the the stabilizer group using $O(\log n)$ samples of Lemma 4.1.

□

If we replace the erroneous CliffSim[2] oracle with an erroneous NoisyCliffSim[2] oracle of equal error rate, then a BPAC^0 circuit can still solve an NC^1-hard problem. The circuit simply decodes the encoded output to have the same guarantees as Lemma 4.2.

Theorem 4.3. *Let $\mathcal{R}$ be the rewind oracle for NoisyCliffSim[2] that outputs a correct answer with probability at least $\frac{29}{30}$ over uniformly random first round input in $\{\mathcal{C}_2/\mathcal{P}_2\}^n$ and one of six Pauli measurements in the second round. Also, let $m, m_{anc} = \mathsf{polylog}(n)$. Then*

$$\mathsf{NC}^1 \subseteq (\mathsf{qBPAC}^0)^{\mathcal{R}} \tag{17}$$

Proof. We use the same circuit as Lemma 4.2 with one modification: Instead of querying the CliffSim[2] rewind oracle, we query the NoisyCliffSim[2] oracle to receive encoded outputs $(\mathcal{Y}_1, \mathcal{Y}_2)$ and syndrome measurements s. Using these outputs, we decode the output following the same procedure as Lemma 3.2 for every codeblock. This reduction requires a $2^{O(\mathsf{polylog}(n))}$ size AC^0 circuit, resulting in a quasipolynomial size decoding circuit for each call to the NoisyCliffSim[2] rewind oracle. The oracle only needs to be called $O(\log n)$ times, so this only expands the circuit by a quasipolynomial size. This places the circuit in the class $(\mathsf{qBPAC}^0)^{\mathcal{R}}$. Because the NoisyCliffSim[2] rewind oracle's output has successful output upon decoding with probability at least $\frac{29}{30}$, then $\mathsf{NC}^1 \subseteq (\mathsf{qBPAC}^0)^{\mathcal{R}}$ by Lemma 4.2. □

qBPAC^0 may seem like an unnatural class of circuits, so we prove the following corollary of Theorem 4.3.

Corollary 4.2. *For any prime p, any $\mathsf{AC}^0[p]$ circuit of depth d and size $\exp(n^{1/2(d+4)})$ cannot pass the noise-tolerant interactive measurement problem with probability at least $\frac{29}{30}$ over a uniform input.*

Proof. Let $\mathsf{AC}^0[p](d, s)$ be the $\mathsf{AC}^0[p]$ circuit of depth d and size s as stated. Suppose for contradiction that such a circuit could pass the interactive protocol with probability $\frac{29}{30}$ over the uniform input distribution. Then there must exist an $\mathsf{AC}^0[p](d, s)$ circuit for the rewind oracle that succeeds with probability at least $\frac{29}{30}$, so $\mathsf{NC}^1 \subseteq \mathsf{BPAC}^0[p](d, s)$. A result by Ajtai and Ben-Or is that BPAC^0 is contained in non-uniform AC^0 with four more layers in depth and polynomial overhead in size [AB84]. Their proof also holds when considering any class of AC^0 circuits of at least polynomial size equipped with MOD_p gates, so $\mathsf{BPAC}^0[p](d, s) \subseteq \mathsf{AC}^0[p](d{+}4, s)$. $\mathsf{NC}^1 \subseteq \mathsf{AC}^0[p](d{+}4, s)$ contradicts the exponential lower bounds by Razborov and Smolensky [Raz87; Smo87], so we conclude that no $\mathsf{AC}^0[p](d, s)$ circuit can pass the interactive measurement problem with average probability $\frac{29}{30}$. □

4.2 Noisy w-PBP-hardness

The main theorem of this section is an average-case w-PBP-hardness of classically simulating w-width 2D grids of constant-depth quantum circuits.

4.2.1 Problem statement and input randomization

Call the set of CNOT gates on m qubits CNOT_m. The problem that we We base the hardness on the following problem:

Problem 2. *(w-Width Cluster Clifford Simulation) Let $v \leq \mathsf{poly}(n)$, $u = O(1)$, and define a 2D grid of qubits $Q_{i,j}$ for $i \in [w]$ and $j \in [v + u]$ to be in the basis state. Let binary matrices $A \in \{0,1\}^{w \times v}$ and $B \in \{0,1\}^{w \times u}$ be input over two rounds of the w-Width Cluster Clifford Simulation problem. Specifically, the circuit must do the following:*

1. *In the first round, the challenger sends A to the circuit. For each $i \le w, j \le v$, if $A_{i,j} = 1$, then the circuit measures qubit $Q_{i,j}$ in the X-basis. Otherwise it measures $Q_{i,j}$ in the Y-basis. The circuit returns the measurement results to the challenger.*

2. *In the second round, the challenger sends B to the circuit. For each $i \le w, j \le u$, if $B_{i,j} = 1$, then the circuit measures qubit $Q_{i,v+j}$ in the X-basis. Otherwise it measures $Q_{i,v+j}$ in the Y-basis. The circuit returns the measurement results to the challenger.*

A constant-depth Clifford circuit can solve this problem by simply following the challenger's requests. In addition, we have the following corollary due to Theorem 4.2

Corollary 4.3. *Let $\mathcal{I}'$ be the noisy extension of Problem 2. A noisy constant depth-D Clifford circuit can solve $\mathcal{I}'$ with probability*

$$\geq 1 - \exp(-\Omega(\mathsf{polylog} n))$$

when the noise rate p is bounded by

$$p < p_{th} = \exp(-\exp(O(D)))$$

The question now is how to turn a classical simulation of Problem 2 to a useful subroutine for deciding a w-PBP-hard problem. From there, we apply the noisy extension to (conditionally) separate noisy quantum and noiseless classical machines using Corollary 4.3. We will use a protocol originating from [GS20], reviewing it in the necessary detail to prove average-case hardness. We refer the reader to [GS20] for proofs of correctness.

Let us define three groups and a set of 3-qubit Paulis that are relevant to the protocol. Let the group G_m consist of single CNOT gates followed by circuits generated by CZ and S gates; i.e. $G_m = \mathrm{CNOT}_m\langle \mathrm{CZ}, \mathrm{S}\rangle_m$; where the binary operation is composition. Let $H_m = \langle \mathrm{CZ}, \mathrm{S}\rangle_m$, where we notice that H_m is a normal subgroup of G_m. Finally, let $H_3^{\oplus}$ be the subgroup of H_m that consists of even numbers of CZ and S gates on the first three qubits. In summary, we have defined the following relationship: $H_3^{\oplus} \leqslant H_m \trianglelefteq G_m$.

Then let

$$I_2 = \{\{XXX, XYY, YXY, YYX\}, \{IYI, XII, XYY, IIY\}, \{IIY, YXY, YII, IXI\}, \{XXX, XII, IIX, IXI\}, \{IYI, IIX, YII, YXY\}\}$$

be a collection of sets of 3-qubit Paulis, and define $\bigstar$ be the union over these sets. Let us define the operation $A \cdot B$ as ABA^{-1}. It can be checked that $|H_3^{\oplus} \cdot \bigstar| = 24$ where $\cdot$ is performed pairwise between the elements of the two sets. Next, we define the protocol R_f:

Algorithm 1: Randomized algorithm, R_f, carried out by $(\mathsf{BPAC}^0)^{\mathcal{R}}$ circuit

Input: $f \in H_3^{\oplus}$, $g_1, ..., g_n \in \mathrm{CNOT}_m$ promised that $\pi = g_1...g_n \in \{C_3, I\}$
Output: A 3-qubit Pauli in $\stackrel{\star}{\star}$
1 Sample $f' \leftarrow H_3^{\oplus}$;
2 Sample $h_1, ..., h_{2n-1} \leftarrow H_m$;
/* The following is Kilian randomization */
3 $(g'_1, ..., g'_{2n}) \leftarrow (f' g_1 h_1, h_1^{-1} g_2 h_2, ..., h_{n-1}^{-1} g_n h_n, h_n^{-1} f g_n h_{n+1}, ..., h_{2n-2}^{-1} g_2 h_{2n-1}, h_{2n-1}^{-1} g_1)$;
4 Input $(g'_1, ..., g'_{2n})$ to $\mathcal{R}$ in first round;
5 **for** *Pauli line* (P_1, P_2, P_3, P_4) *in* I_2 **do**
6 Input (P_1, P_2, P_3, P_4) to $\mathcal{R}$ in the second round;
7 Record measurement outcome of P_i for $i \in [4]$ from $\mathcal{R}$;
8 Rewind $\mathcal{R}$ to beginning of second round;
/* Each 3-qubit Pauli P_i is measured exactly twice in the loop */
9 $P \leftarrow$ any 3-qubit Pauli for which $\mathcal{R}$ returns inconsistent results;
10 **return** $f'^{-1} \cdot P$

The first round input $(g'_1, ..., g'_{2n})$ is uniformly random with the constraints: (1) $g'_i H_m = g_i H_m$ and $g'_{n+i} H_m = g_{n-i+1} H_m$ for $i \in [n]$ and (2) $g'_1 ... g'_{2n} \in H_3^{\oplus}$. Let us denote by $I_1(g_1, ..., g_n)$ the support of this distribution.

If the oracle makes no errors, then the output of R_f will be a sample from the distribution of 3-qubit Paulis $(\pi f \pi^{-1}) \cdot \mathcal{D}_{\mathcal{R}}$ where $\mathcal{D}_{\mathcal{R}}$ is a distribution over the 20 nonstabilizers of $|+^3\rangle$ in $S := H_3^{\oplus} \cdot \stackrel{\star}{\star}$. Let us call this errorless distribution $R_{f,0}$. Otherwise, suppose that the oracle fails on $(g'_1, ..., g'_{2n})$ with probability ϵ for $(g'_1, ..., g'_{2n})$ sampled from $I_1(g_1, ..., g_n)$, where we say the oracle fails for $(g'_1, ..., g'_{2n}) \in I_1(g_1, ..., g_n)$ if it fails for any input in $(g'_1, ..., g'_{2n}) \times I_2$. Then with probability ϵ, R_f samples from any fixed distribution over S, and with probability $1 - \epsilon$, R_f samples from $R_{f,0}$. Let us call this faulty distribution $R_{f,\epsilon}$. Notice that in both cases, $\mathcal{D}_{\mathcal{R}}$ is fixed regardless of the value of f. This leads us to choose f in a useful way for distinguishing the 3-cycle or identity.

Lemma 4.3. *(Theorem 33, [GS20]) For every Pauli $P \in \mathcal{D}_{\mathcal{R}}$, a NC^0 circuit can determine an $f \in H_3^{\oplus}$ such that $f \cdot P$ has no weight in $R_{f,0}$ if $\pi = C_3$.*

Lemma 4.3 becomes useful only after we have a Pauli P on which to apply it. After we have identified such a P, the problem of determining whether π is the 3-cycle or identity becomes tractable for BPAC^0 circuits, and this roughly becomes the idea for learning π.

Lemma 4.4. *Suppose that a BPAC^0 circuit can sample from a distribution $\gamma(g_1, ..., g_n)$ over first round inputs to $\mathcal{R}$ of the form in Line 3 of Algorithm 1 and*

$$\delta := Pr[\mathcal{R} \text{ fails for any } (g'_1, ..., g'_{2n}) \times I_2 \mid (g'_1, ..., g'_{2n}) \leftarrow \gamma(g_1, ..., g_n)] < \frac{1}{21}$$

Then a $(\mathsf{BPAC}^0)^{\mathcal{R}}$ circuit can determine whether $\pi = g_1...g_n$ is the 3-cycle or identity.

Note that the following protocol is similar to that in Theorem 33 of [GS20], except we provide an error analysis that is crucial to the main average-case hardness results.

Proof. The $(\mathsf{BPAC}^0)^{\mathcal{R}}$ circuit proceeds in two phases. In the first phase, the circuit learns information about the distribution $\mathcal{D}_{\mathcal{R}}$. Then in the second phase, the circuit uses this information to deduce the value of π.

In further detail, the first phase of the circuit samples $O(\lg n)$ 3-qubit Paulis from $R_{I,\epsilon}$, where the oracle $\mathcal{R}$ fails with probability $\delta < \frac{1}{21}$ for first round input. With probability $1-\delta$, a sample comes from $R_{f,0} = \mathcal{D}_{\mathcal{R}}$. There is a 3-qubit Pauli $P \in S$ that attains the maximal weight $\mathcal{D}_{\mathcal{R}}(P) \geq \frac{1}{20}$. The Pauli P also has the maximal weight in $R_{I,\delta}$ because $\frac{1}{20}(1-\delta) \geq \delta$. By sampling $O(\lg n)$ Paulis from $R_{I,\delta}$, the circuit recovers P with high probability. Furthermore, by Lemma 4.3, the circuit determines a $f \in H_3^{\oplus}$ such that $f \cdot P$ has no weight in $R_{f,0}$

The second phase of the circuit samples $O(\lg n)$ Paulis from $R_{f,\delta}$. With probability $1-\delta$, a sample will be from the distribution $R_{f,0} = (\pi f \pi^{-1}) \cdot \mathcal{D}_{\mathcal{R}}$. If $\pi = I$, then $R_{f,0} = f \cdot \mathcal{D}_{\mathcal{R}}$, so $f \cdot P$ has equal weight in $R_{f,0}$ as P has weight in $\mathcal{D}_{\mathcal{R}}$. If $\pi = C_3$, then $f \cdot P$ has no weight in $R_{f,0}$ by Lemma 4.3. Due to the previous facts, we have the bound

$$R_{f,\delta:\pi=I}(f \cdot P) \geq (1-\delta)\mathcal{D}_{\mathcal{R}}(P) \geq \frac{1}{21} > \delta \geq R_{f,\delta:\pi=C_3}(f \cdot P)$$

so $R_{f,\delta:\pi=I}(f \cdot P)$ and $R_{f,\delta:\pi=C_3}(f \cdot P)$ are at least a constant difference apart. Thus, the circuit can distinguish $\pi \in \{C_3, I\}$ using the $O(\lg n)$ Pauli samples from $R_{f,\delta}$ with high probability by checking whether $f \cdot P$ appears in significantly more than a δ ratio of all samples. □

Lemma 4.4 gives the circuit a way to deal with the oracle's sufficiently suppressed errors.

4.2.2 Statement of main theorem and reduction

Going further, the main theorem of this section does not deal with $g_1, ..., g_n \in \mathrm{CNOT}_m$. Instead, $g_1, ..., g_n$ will be elements from $\langle \mathrm{SWAP} \rangle_m$, but it is not hard to see that Lemma 4.4 holds for G_m defined using SWAPs instead of CNOTs. Let us state the main theorem.

Theorem 4.4. *Let $w \leq \mathsf{poly}(n)$ and $\mathcal{R}$ be the rewind oracle for $(w+2)$-Width Cluster Clifford Simulation that fails with probability $\epsilon < \frac{1}{106}$. Then*

$$w\text{-}\mathsf{PBP} \subseteq (\mathsf{BPTC}^0)^{\mathcal{R}}$$

From well-known results in complexity theory, we have the following corollaries:

1. $(w = 5)$ $\mathsf{NC}^1 \subseteq (\mathsf{BPAC}^0)^{\mathcal{R}}$. Note that the error bound ϵ is weaker here than X.
2. $(w = \mathsf{poly}(n))$ $\mathsf{L/poly} \subseteq (\mathsf{BPTC}^0)^{\mathcal{R}}$

Additionally, if we denote the noisy-extended version of the protocol by $\mathcal{I}'$, then $\mathcal{I}'$ is solved with probability $1 - \exp(-\Omega(\mathsf{polylog}(n)))$ over all inputs by a noisy SQC due to Theorem 4.2. With the decoding reduction in Lemma 3.2, if the rewind oracle, $\mathcal{R}'$, for $\mathcal{I}'$ fails for $\epsilon < \frac{1}{106}$ of inputs, then

$$w\text{-}\mathsf{PBP} \subseteq (\mathsf{qBPTC}^0)^{\mathcal{R}'}$$

The reduction in Theorem 4.4 utilizes $\mathcal{R}$ to simulate a product of permutations by performing transpositions of qubits via SWAP gates. However, it will be useful to reduce the problem of determining a product of permutations to the problem of deciding whether a permutation is the 3-cycle or identity to leverage Lemma 4.4. Toward that end, we have the following reduction:

Lemma 4.5. *Let $w \leq \mathsf{poly}(n)$ and $\mathcal{O}$ be the oracle that decides whether a sequence of permutations on $w+2$ elements multiplies to C_3 or I. Then*

$$w\text{-}\mathsf{PBP} \subseteq (\mathsf{NC}^0)^{\mathcal{O}}$$

Proof. Let $\pi = \pi_1 ... \pi_n$ be the product of the permutations on w elements induced by the w-PBP. A YES instance implies $\pi = \alpha$ for some fixed non-identity permutation α. Otherwise, a NO instance implies $\pi = I$. Because α is non-identity, then $\alpha(x) = y$ for some $x \neq y \in [w]$. So deciding whether $\beta = (1\ x)\pi(1\ y)$ fixes the first element also decides whether π is α or I.

If we define the permutation $D = \beta(1\ (w+1))\beta^{-1}$, then $D = (1\ (w+1))$ if β fixes the first element. Otherwise, $D = (z\ (w+1))$ for some $z \in \{2, ..., w\}$. If we define the permutation $E = D(1\ (w+2))D(1\ w+2)$, then $E = (1\ (w+2)\ (w+1))$ if $D = (1\ (w+1))$. Otherwise, $E = I$ if $D = (z\ (w+1))$. Therefore, distinguishing E as C_3 or I determines whether β fixes the first element, and hence, decides w-PBP. Note that the only parts necessary in this reduction by a circuit are fan-out and nonuniformity, which are provided by NC^0. □

4.2.3 Input randomization

Equipped with Lemma 4.5, we can use Lemma 4.4 to learn information about the permutation π. It might be tempting to simply let γ be the uniform distribution over $I_1(g_1, ..., g_n)$ and apply Lemma 4.4, but there are two problems with this approach

1. If H_m is simply $\langle \mathrm{CZ}, \mathrm{S} \rangle_m$, as is the case in Algorithm 1, then $\epsilon < \frac{1}{106}$ error is not enough to imply the condition of Lemma 4.4.

2. Sampling a uniform distribution γ while avoiding the problem above is not known (to the best of the author's knowledge) to be possible in TC^0.

Let us more precisely illustrate these problems and provide a roadmap for how we will fix the shortcomings. For the first problem, the input to the protocol, $g_1, ..., g_n$, is uniformly randomized to $g'_1, ..., g'_{2n}$ where one of the constraints on $g'_1, ..., g'_{2n}$ is $g'_i H_m = g_i H_m$ and $g'_{n+i} H_m = g_{n-i+1} H_m$ for all $i \in [n]$. However, for any distinct sets of SWAPs, $(g_1^{(1)}, ..., g_n^{(1)}) \neq (g_1^{(2)}, ..., g_n^{(2)})$ as protocol input, the constraint implies that the supports of the input distributions to the oracle, $(g_1'^{(1)}, ..., g_{2n}'^{(1)})$ and $(g_1'^{(2)}, ..., g_{2n}'^{(2)})$, are disjoint. This is a problem because even if an exponentially small fraction of input to the oracle fails, a weaker assumption than a constant $\epsilon = \frac{1}{106}$ fails, then it could be the case that every input in the distribution of $(g_1'^{(1)}, ..., g_{2n}'^{(1)})$ fails for the oracle, so the procedure cannot reliably determine whether $(g_1^{(1)}, ..., g_n^{(1)})$ amounts to I or C_3.

The lesson is that we require a stronger form of randomization that successfully "hides" the protocol input $(g_1, ..., g_n)$ in the oracle input distribution. The way to "hide" the SWAPs is to introduce random elements from $\langle \mathrm{SWAP} \rangle_m$ as an extra ingredient to H_m. Unfortunately, as the second problem points out, sampling uniformly random permutations is not known to be possible in TC^0. But let us first ignore this problem and consider what happens if we could sample uniform permutations.

Proposition 1. *Let $g_1, ..., g_n \in \langle SWAP \rangle_m$ be a sequence of permutations promised to be the 3-cycle or identity. Then the output of Kilian randomization, Line 3 in Algorithm 1, where $h_1, ..., h_{2n-1}$ are sampled uniformly from $H_m = \langle SWAP, CZ, S \rangle_m$, is uniform over $I_1(g_1, ..., g_n)$.*

Recall that $I_1(g_1, ..., g_n)$ is all $\hat{g_1}, ..., \hat{g_{2n}} \in H_m$ such that $\hat{g_1} ... \hat{g_{2n}} \in H_3^{\oplus}$. Notice that the constraint on the output of Kilian randomization is *independent* of $(g_1, ..., g_n)$, implying that we have successfully hidden the input to the oracle. Fortunately, we can *almost* achieve this in AC^0 by using known *approximate* permutation sampling techniques [MV91]. By using the approximate sampling of $\langle \mathrm{SWAP} \rangle_m$, we essentially fix the second problem, up to some small error. Then we can "hide" the sets of SWAPs in our input distribution to the oracle, similarly to Proposition 1, which fixes the first problem. Toward this solution, let us consider the approximate uniform permutation sampling that we can achieve.

To quantify the distance between two distributions, we define the variation distance:

Definition 4.3. *(Variation distance) Suppose two random variables, $\mathcal{X}$ and $\mathcal{Y}$, take on values over the same set. Then the variation distance between $\mathcal{X}$ and $\mathcal{Y}$ is defined to be*

$$\frac{1}{2}|\mathcal{X} - \mathcal{Y}|_1$$

where $|\mathcal{X} - \mathcal{Y}|_1$ is the 1-norm of entry-wise difference in distributions of $\mathcal{X}$ and $\mathcal{Y}$.

This allows us to quantify the following result:

Lemma 4.6. *([MV91]) A randomized, $\mathsf{poly}(n)$-size AC^0 circuit can sample from a distribution that is $\delta = 2^{-\mathsf{poly}(n)}$ close in variation distance to the uniform distribution over permutations of $[n]$.*

Using this idea, we can approximately sample uniformly random elements from $H_m = \langle \mathrm{SWAP}, \mathrm{CZ}, \mathrm{S} \rangle_m$.

Proposition 2. *A randomized, $\mathsf{poly}(m)$-size AC^0 circuit can sample from a distribution, J, that is $\delta = 2^{-\mathsf{poly}(m)}$ close in variation distance to the uniform distribution over H_m.*

Proof. The circuit samples an element $s \in \langle \mathrm{SWAP} \rangle_m$ using Lemma 4.6, and it composes it with a uniformly random element $d \in \langle \mathrm{CZ}, \mathrm{S} \rangle_m$. The $s \in \langle \mathrm{SWAP} \rangle_m$ is $2^{-\mathsf{poly}(m)}$ close in variation distance to uniform, and d is uniform, so the distribution over sd, J, must be $2^{-\mathsf{poly}(m)}$ close to uniform. □

When randomizing the input $(g_1, ..., g_n)$, the circuit performs Kilian randomization by sampling elements of H_m from Proposition 2 instead of the uniform distribution over H_m. We want to argue that the oracle input distribution is essentially uniform, up to some small error.

Lemma 4.7. *Let $g_1, ..., g_n \in \langle SWAP \rangle_m$ be a sequence of permutations promised to be the 3-cycle or identity. Then the output of Kilian randomization, Line 3 in Algorithm 1, where $h_1, ..., h_{2n-1}$ are sampled by Proposition 2, is $2^{-\mathsf{poly}(n)}$ close in variation distance to the uniform distribution over $I_1(g_1, ..., g_n)$.*

Proof. The uniform distribution over $\hat{g_1}, ..., \hat{g_{2n}} \in H_m$ such that $\hat{g_1}...\hat{g_{2n}} \in H_3^{\oplus}$ is produced by fixing any $f \in H_3^{\oplus}$, sampling $f' \leftarrow U_{H_3^{\oplus}}$ and $h_1, ...h_{2n-1} \leftarrow U_{H_m}$ to produce

$$(\hat{g_1}, ..., \hat{g_{2n}}) = (f'g_1h_1, h_1^{-1}g_2h_2, ..., h_{n-1}^{-1}g_nh_n, h_n^{-1}fg_nh_{n+1}, h_{n+1}^{-1}g_{n-1}h_{n+2}, ..., h_{2n-1}^{-1}g_1) \tag{18}$$

It suffices to show that the variation distance between the distribution over $(\hat{g_1}, ..., \hat{g_{2n}})$ in Equation (18) and $\mathcal{D}_{\mathcal{I}}$ is at most $2^{-\mathsf{poly}(n)}$. The distribution $\mathcal{D}_{\mathcal{I}}$ is created by sampling $f' \leftarrow U_{H_3^{\oplus}}$ and $j_1, ...j_{2n-1} \leftarrow J$ from Proposition 2, and multiplying to get

$$(g'_1, ..., g'_{2n}) = (f'g_1j_1, j_1^{-1}g_2j_2, ..., j_{n-1}^{-1}g_nj_n, j_n^{-1}fg_nj_{n+1}, j_{n+1}^{-1}g_{n-1}j_{n+2}, ..., j_{2n-1}^{-1}g_1) \tag{19}$$

To bound the distance between Equation (18) and Equation (19), it suffices to bound the distance between

$$(g_1h_1, h_1^{-1}g_2h_2, ..., h_{n-1}^{-1}g_nh_n, h_n^{-1}g_nh_{n+1}, h_{n+1}^{-1}g_{n-1}h_{n+2}, ..., h_{2n-2}^{-1}g_2h_{2n-1}) \tag{20}$$

and

$$(g_1j_1, j_1^{-1}g_2j_2, ..., j_{n-1}^{-1}g_nj_n, j_n^{-1}g_nj_{n+1}, j_{n+1}^{-1}g_{n-1}j_{n+2}, ..., j_{2n-2}^{-1}g_2j_{2n-1}) \tag{21}$$

because f' is uniform, f permutes both distributions in the same way, and the final entries in both distributions, $h_{2n-1}^{-1}g_1$ and $j_{2n-1}^{-1}g_1$, are fixed after conditioning on the values of the other entries. In Equation (20), an entry $h_ig_jh_{i+1}$ has the property that h_ig_j is fixed after conditioning on the entries to the left, and a similar property holds for Equation (21). Therefore, it also suffices to bound the distance between

$$(h_1, ..., h_{2n-1}) \tag{22}$$

and

$$(j_1, ..., j_{2n-1}) \tag{23}$$

All h_i are independent from one another (similarly for j_i), and the distance between h_i and j_i is δ. In this case, variation distance is subadditive, so the distance between Equation (22) and Equation (23) is at most $(2n-1)\delta$. Noticing that $\delta \leq 2^{-\mathsf{poly}(n)}$, we get the desired bound. □

4.2.4 Proof of average-case w-PBP-hardness

We are now ready to prove the main theorem of this section. The problem of deciding whether $\pi = g_1, ..., g_n \in \langle \mathrm{SWAP} \rangle_m$ multiplies to the 3-cycle or identity is w-PBP-hard due to Lemma 4.5.

If $\mathcal{R}$ fails for $\frac{1}{106}$ of all input, a constant smaller than $\frac{1}{105}$, then it fails for $\leq \frac{5}{106} < \frac{1}{21}$ of first round input because $|I_2| = 5$. Because first round input is sampled from $\mathcal{D}_{\mathcal{I}}$ which is $2^{-\mathsf{poly}(n)}$-close to $U_{I_1(g_1,...,g_n)}$ by Lemma 4.7, then the circuit samples first round input such that

$$\Pr[\mathcal{R} \text{ fails for any } (g'_1, ..., g'_{2n}) \times I_2 \mid (g'_1, ..., g'_{2n}) \leftarrow \mathcal{D}_{\mathcal{I}}] < \frac{5}{106} + 2^{-\mathsf{poly}(n)} < \frac{1}{21}$$

Combining this with Lemma 4.4, the circuit determines π with high probability.

It remains to check that the input to $\mathcal{R}$ can be made classically-controlled (CC); i.e. each gate that $\mathcal{R}$ simulates should depend on at most one input bit given to it to conform to the definition in Problem 2. The input given to the oracle in I_2 can easily be converted to such a form in NC^0. To convert an element of I_1 to a CC form, we first notice that any $g \in G_m$ can be written as an element of $\langle \mathrm{SWAP} \rangle_m \langle \mathrm{CZ}, \mathrm{S} \rangle_m$. The gates CZ and S are diagonal, so they commute with one another, and $\langle \mathrm{CZ}, \mathrm{S} \rangle_m$ can be made CC in a straightforward way from this fact.

Now we turn to converting a permutation $\sigma = (\sigma(1)...\sigma(m)) \in \langle \mathrm{SWAP} \rangle_m$ to CC form. Any series of local transpositions that sorts $(\sigma(1)...\sigma(m))$ also creates the permutation σ, so it suffices to sort $(\sigma(1)...\sigma(m))$ to create a CC circuit of SWAPs. Using threshold gates, a TC^0 circuit can determine the value of

$$|\{j < i \,:\, \sigma(j) > \sigma(i)\}|$$

for all i in parallel. In fact, this is the only comparison needed to create a CC circuit of SWAPs that performs insertion sort on $(\sigma(1)...\sigma(m))$. Therefore, the TC^0 circuit converts $\sigma \in \langle \mathrm{SWAP} \rangle_m$ to a CC circuit. This proves the claim in Theorem 4.4.

4.3 Noisy $\oplus\mathsf{L}$-hardness

The main theorem of this section is an average-case $\oplus\mathsf{L}$ hardness for a classical simulation of the $\mathsf{poly}(n)$-Width Cluster Clifford Simulation.

4.3.1 A $\oplus\mathsf{L}$-hard problem and statement of main theorem

Let M be the set of $n \times n$ adjacency matrices of monotone graphs[3]. Our hardness result stems from the hardness off the following $\oplus\mathsf{L}$-complete problem related to monotone graphs:

Lemma 4.8. *([Dam90]) Let $A \in M$ be the $n \times n$ adjacency matrix of a monotone graph. Define MGap to be the problem of deciding the parity of paths from 1 to n in A. Then*

$$\oplus\mathsf{L} \subseteq (\mathsf{NC}^0)^{\mathsf{MGap}}$$

The average-case hardness that we are after applies for a very specific promise, and in order to define the promise, we need to define a certain randomized algorithm. It will be useful to describe the randomized algorithm as a composition of three algorithms, $F \circ E \circ D$, that takes as input from M and outputs (randomly) a sequence of elements from $G_m = \mathrm{CNOT}_{m(n)} \langle \mathrm{CZ}, \mathrm{S} \rangle_{m(n)}$ where m is a polynomial. We will describe F, E, D after the statement of the main theorem:

Theorem 4.5. *Let $\mathcal{R}$ be the rewind oracle for $\mathsf{poly}(n)$-Width Cluster Clifford Simulation promised that input is from the image of $F \circ E \circ D(M)$ in the first round and I_2 in the second round. If $\mathcal{R}$ fails on $\epsilon < \frac{1}{421}$ fraction of inputs, then*

$$\oplus\mathsf{L} \subseteq (\mathsf{BPAC}^0)^{\mathcal{R}}$$

[3]A monotone (directed) graph on $V = \{1, ..., n\}$ has no edges from j to i for $j \geq i$

Using previous arguments, we also the following corollary: $\mathcal{I}'$, the noisy-extended version of the protocol, is solved with probability $1-\exp(-\Omega(\mathsf{polylog}(n)))$ over all inputs by a noisy SQC. If the rewind oracle, $\mathcal{R}'$, for $\mathcal{I}'$ fails for $\epsilon < \frac{420}{421}$ of inputs, then

$$\oplus\mathsf{L} \subseteq (\mathsf{qBPAC}^0)^{\mathcal{R}'}$$

In order to achieve average-case hardness, we add an extra layer of randomization to Algorithm 1. To illustrate what we mean by this, recall that in the algorithm construction, we randomize $g_1, ..., g_n \in \mathrm{CNOT}_m$ before querying the oracle with input. Our new layer of randomization occurs even before Algorithm 1 is used.

4.3.2 Input half-randomization and $F \circ E \circ D$ description

Let us first show how input is given to the algorithm, and we will speak on the pre-randomization in more detail. Define $\mathsf{LDAGParity}$ be the problem of determining whether a layered DAG (adjacency matrix) has the parity of paths from a start vertex to end vertex as being odd, and let $\mathsf{CNOTMult}^*$ be the problem of deciding whether $g_1, ..., g_n \in \mathrm{CNOT}_m$ multiplies to the 3-cycle (otherwise, the identity). Then we define the algorithm E:

Lemma 4.9. *(Lemma 44, [GS20]) There is a function E computed by an NC^0 circuit that takes as input an LDAG adjacency matrix A of dimension $n \times n$ and outputs a sequence of $g_1, ..., g_{\mathsf{poly}(n)} \in CNOT_m$ such that*

1. *$A \in \mathsf{LDAGParity}$ iff $(g_1, ..., g_{\mathsf{poly}(n)}) \in \mathsf{CNOTMult}^*$*
2. *E is injective*

The randomized algorithm F will be Kilian randomization from Line 3 in Algorithm 1. Thus, the composed algorithm $F \circ E$ outputs a random sequence of elements from G_m that is given to the oracle as first round input.

The new layer of randomization D is different in the sense that it is a *pre*-randomization of the input to the general $F \circ E$; it occurs *before* the reduction on an $\mathsf{LDAGParity}$ instance occurs. This is why the composition of randomized algorithms $F \circ E \circ D$ is a useful description of the oracle input: D pre-randomizes to produce a random $\mathsf{LDAGParity}$ instance, E reduces from $\mathsf{LDAGParity}$ to $\mathsf{CNOTMult}^*$ instance, and F does the same randomization procedure as in Line 3 in Algorithm 1. The pre-randomizer D is the key ingredient that provides the $\frac{1}{421}$ average-case hardness. We will use r_f and r_d as notation for the random bits of F and D, while noting that E is determinisic.

$F \circ E \circ D$ possesses a special property called *half-randomizing* that we define first. Then we will describe the pre-randomizer D.

Definition 4.4. *Let (Π_{yes}, Π_{no}) be a promise problem. A randomized algorithm $\tilde{f}$ that takes input from $\Pi_{yes} \cup \Pi_{no}$ and outputs from $\mathcal{C}$ is **half-randomizing** if there are two disjoint, equal-sized subsets of $\mathcal{C}$, C_{yes} and C_{no}, such that for $i \in \{yes, no\}$*

$$\forall x \in \Pi_i,\ \tilde{f}(x) \equiv U_{C_i}$$

where U_{C_i} is the uniform distribution over C_i.

There is an immediate consequence of Definition 4.4:

Proposition 3. *If δ fraction of $C_{yes} \cup C_{no}$ have some property, p, then for any $x \in \Pi_{yes} \cup \Pi_{no}$,*

$$Pr[\tilde{f}(x) \text{ has property } p] \leq 2\delta$$

To give some intuition for why Proposition 3 may be useful, consider an oracle that fails for ϵ fraction of inputs. By randomizing its input using $\tilde{f}$, an algorithm that uses this oracle will only have a 2ϵ probability of querying the oracle with a failing input. This is precisely the idea that we use to achieve average-case hardness.[4]

Randomized algorithm D is very similar to the randomized encoding techniques introduced by [AIK06], except we introduce an extra step that an NC^0 circuit can perform. Recall that D takes as input a monotone adjacency matrix from M and converts it to a LDAG adjacency matrix. It will be useful to define $D = D_{adj} \circ D_{re}$ where D_{re} is the randomized encoding map from [AIK06] (to be defined) and D_{adj} is the modification that creates the adjacency matrix of a LDAG. We describe some useful properties related to D_{re}.

Fact 1. *(Fact 4.13, [AIK06]) Let A be an $n \times n$ monotone adjacency matrix, and let L be the $(n-1) \times (n-1)$ top-right submatrix of $A - I$. Then $det(L) \mod 2$ is the parity of the number of paths from 1 to n in A.*

So the determinant of L encodes the answer to a $\oplus\mathsf{L}$-hard problem. Note that L is upper-triangular *except* for -1 on its second diagonal. Furthermore, we can sample uniformly from a certain representation of all matrices that have the same determinant as L and have -1 on their second diagonal. We call the next sampling procedure D_{re}.

Proposition 4. *(D_{re}, Lemma 4.17, [AIK06]) Given A, a randomized NC^0 circuit can sample uniformly $0/1$-values*

$$\{(K_{i,j}^{(1)}, ..., K_{i,j}^{(k)})\}_{1 \leq i \leq j \leq n-1}$$

(where k is polynomially related to n) such that the $(n-1) \times (n-1)$ matrix $K^{\oplus}$ defined by $K_{i,j}^{\oplus} = \oplus_{\ell} K_{i,j}^{(\ell)}$, -1 on its second diagonal, and 0 below it has

$$det(K^{\oplus}) \equiv det(L) \mod 2 \tag{24}$$

Following D_{re}, we let D_{adj} convert the $\{(K_{i,j}^{(1)}, ..., K_{i,j}^{(k)})\}_{1 \leq i \leq j \leq n-1}$ matrix representation to the adjacency matrix of a LDAG. Before doing this, we define the $n \times n$ $0/1$ matrix B: Let the $0/1$ entries above the main diagonal of B be the same $0/1$ entries on or above the main diagonal of $K^{\oplus}$ and let every other entry be 0.

Proposition 5. *B is a uniformly random monotone adjacency matrix such that the parity of paths from 1 to n is the same as in A.*

Proof. Note that B is a valid monotone adjacency matrix, and $K^{\oplus}$ is the $(n-1) \times (n-1)$ top-right submatrix of $B - I$. The entries on or above the main diagonal of $K^{\oplus}$ are uniformly random subject to the determinant constraint in Equation (24), so the entries above the main diagonal of $B - I$, and hence B, are uniformly random subject to the constraint. However, the constraint is equivalent to A and B having the same parity of paths from 1 to n due to Fact 1. □

[4] Of course, the trouble is in making sure that the input to the oracle preserves information about the problem instance, which half-randomizing algorithms do.

We are now ready to define D_{adj}, which takes $\{(K_{i,j}^{(1)}, ..., K_{i,j}^{(k)})\}_{1 \le i \le j \le n-1}$ and deterministically converts it to a LDAG $\mathcal{L}$ that preserves the parity of the number of paths from source to sink as B. $\mathcal{L}$ has n layers labeled N_i for each $i \in [n]$ and $(n-1)$ layers labeled J_i for each $i \in [n-1]$, and the layers are ordered from left to right in the following way: $N_1 J_1 N_2 J_2 ... J_{n-1} N_n$. Each N_i has vertices $\{1, ..., n\}$ and each J_i has vertices $\{1, ..., n\} \cup \{K_{i,j}^{(\ell)}\}_{j,\ell}$. The only edges between layers fall under two classes:

1. (N_i to J_i) For every $q \in [n]$, vertex q in N_i always connects to vertex q in J_i. Vertex i in N_i connects to vertex $K_{i,j}^{(\ell)}$ in J_i iff the value $K_{i,j}^{(\ell)}$ is 1.
2. (J_i to N_{i+1}) For every $q \in [n]$, vertex q in J_i always connects to vertex q in N_{i+1}. Vertex $K_{i,j}^{(\ell)}$ in J_i always connects to vertex j in N_{i+1}.

The D_{adj} procedure can easily be computed in NC^0 from the $\{(K_{i,j}^{(1)}, ..., K_{i,j}^{(k)})\}_{1 \le i \le j \le n-1}$ matrix representation to $\mathcal{L}$ using the description above.

Lemma 4.10. *Given $A \in M$, $\mathcal{L} = D(A)$ is a uniformly random output of the image of $D(M)$ such that the parity of number of paths from vertex 1 in N_1 to vertex n in N_n is equal to the parity of the number of paths from 1 to n in A. Furthermore, D is half-randomizing.*

Proof. For the first part of the claim, it suffices to show that for a given $K = \{K_{i,j}^{(\ell)}\}_{i,j,\ell}$ the parity of number of paths from vertex 1 in N_1 to vertex n in N_n in $\mathcal{L} = D_{adj}(K)$ is equal to the parity of the number of paths from 1 to n in the B associated with K via and due to Proposition 5.

We prove inductively that the parity of paths from vertex i in N_i to vertex n in N_n is the same as i to n in B. Note that the edge i to i between N_i and J_i does not contribute to the parity of paths from $i \in N_i$ to $n \in N_n$. The parity of paths from $i \in N_i$ to $j \in N_{i+1}$ is equal to $K_{i,j}^{\oplus}$, and there is only one path from $j \in N_{i+1}$ to $j \in N_j$. Hence, the paths from $i \in N_i$ to $j \in N_j$ only change the parity if $K_{i,j}^{\oplus} = 1$ and the parity of paths from j to n in B is odd (by assumption). Therefore, the parity of paths from vertex i in N_i to vertex n in N_n is equal to those from i to n in B.

For the second part of the claim, let Π_{yes} be odd (1 to n) path-parity adjacency matrices and Π_{no} be even adjacency matrices. D_{re} is half-randomizing due to Proposition 5, and D_{adj} is injective, so $D = D_{adj} \circ D_{re}$ is half-randomizing. □

This concludes the definition of D. Let E be the injective reduction from $\mathsf{LDAGParity}$ to $\mathsf{CNOTMult}^*$ in Lemma 4.9.

Proposition 6. *Let $A \in M$ be a monotone graph. The output of $E \circ D(A)$, $g_1, ..., g_{\mathsf{poly}(n)} \in CNOT_m$ multiplies to the 3-cycle on the first 3 qubits if A has odd parity of paths from 1 to n. Otherwise, it multiplies to the identity. Furthermore, $E \circ D$ is half-randomizing.*

Proof. Combine Lemma 4.10 with Lemma 4.9 for the first part of the claim. Half-randomizing follows from the fact that E is injective and D is half-randomizing from Lemma 4.10. □

This concludes the definition of E. Let $H_m = \langle \mathrm{CZ}, \mathrm{S} \rangle_m$, and define randomized algorithm F that takes input $g_1, ..., g_n \in \mathrm{CNOT}_m$ and samples as output $g'_1, ..., g'_{2n}$ as in Line 3 of Algorithm 1. Note that the output of F is precisely the input to the oracle in Algorithm 1. A useful property of $F \circ E \circ D$ is the following:

Lemma 4.11. *$F \circ E \circ D$ is half-randomizing.*

Proof. Following the constraints on randomization in Algorithm 1, $g'_1, ..., g'_{2n}$ are uniformly random subject to the constraints (1) $g'_i H_m = g_i H_m$ and $g'_{n+i} = g_{n-i+1} H_m$ for $i \in [n]$ and (2) $g'_1 ... g'_{2n} \in H_3^{\oplus}$. For any $\{g_i^{(1)}\}_i, \{g_i^{(2)}\}_i \leftarrow E \circ D(M)$ and, $\{g_i^{(1)}\}_i \neq \{g_i^{(2)}\}_i$, constraint (1) implies that the distributions $F(\{g_i^{(1)}\}_i)$ and $F(\{g_i^{(2)}\}_i)$ are disjoint. Furthermore, because (1) and (2) are the only constraints, we have that the supports of $F(\{g_i^{(1)}\}_i)$ and $F(\{g_i^{(2)}\}_i)$ have equal size. Combined with Proposition 6, $F \circ E \circ D$ is half-randomizing. □

This concludes the definition of F. If we let the distribution γ in Lemma 4.4 be F, then we get the following result:

Proposition 7. *Let $(g_1, ..., g_n)$ be a sequence of elements from $CNOT_m$ promised to multiply to the 3-cycle or identity and $\mathcal{R}$ be the rewind oracle for $\mathsf{poly}(n)$-Width Cluster Clifford Simulation. If*

$$Pr_{r_f}[\mathcal{R} \text{ fails on any input in } F_{r_f}(g_1, ..., g_n) \times I_2] < \frac{1}{21}$$

then a $(\mathsf{BPAC}^0)^{\mathcal{R}}$ can decide the the value of $g_1 ... g_n$.

Indeed, if the oracle error is sufficiently suppressed, the product of CNOTs is decidable by the $(\mathsf{BPAC}^0)^{\mathcal{R}}$ circuit.

4.3.3 Proof of average-case $\oplus\mathsf{L}$-hardness

We are now ready to prove the main theorem of this section. Given the adjacency matrix of a monotone graph A, it is $\oplus\mathsf{L}$-hard to decide if the parity of paths from 1 to n is even or odd due to Lemma 4.8. By the procedure in Proposition 7, the oracle is given inputs from $F \circ E \circ D(A) \times I_2$, of which $\leq 2\epsilon$ may fail because $F \circ E \circ D$ is half-randomizing by Lemma 4.11 and the error bound in Proposition 3. Furthermore,

$$Pr_{r_f, r_d}[\mathcal{R} \text{ fails on any input in } F_{r_f} \circ E \circ D_{r_d}(A) \times I_2] \leq 10\epsilon$$

by a union bound with $|I_2| = 5$. We immediately get that

$$Pr_{r_d}[Pr_{r_f}[\mathcal{R} \text{ fails on any input in } F_{r_f} \circ E \circ D_{r_d}(A) \times I_2] \geq \frac{1}{21}] \leq 210\epsilon \tag{25}$$

Deciding whether the output of $E \circ D_{r_d}(A)$, $(g_1, ..., g_{\mathsf{poly}(n)})$, multiplies to the 3-cycle or identity also decides the parity of number of paths from 1 to n in A by Proposition 6. Denoting by C the circuit in Proposition 7, we have for any constant $\alpha > 0$

$$\begin{aligned} &Pr_{r_d}[C \text{ fails to decide } (g_1, ..., g_{\mathsf{poly}(n)}) = E \circ D_{r_d}(A) \in \{C_3, I\}] \\ &\leq 210\epsilon + \alpha(1 - 210\epsilon) \\ &< \frac{1}{2} \end{aligned}$$

where the first inequality comes from Proposition 7 and Equation (25) and the final inequality comes from choosing small enough α and $\epsilon < \frac{1}{421}$. To finish, we repeat the process by sampling

$O(\lg n)$ elements from $E \circ D(A)$ and applying circuit C on the sampled instance. By taking a majority vote of outputs over all instances, we recover the parity of the paths from 1 to n in A with high probability. This proves the claim in Theorem 4.5.

References

[AB84] Miklos Ajtai and Michael Ben-Or. "A Theorem on Probabilistic Constant Depth Computations". In: *Proceedings of the Sixteenth Annual ACM Symposium on Theory of Computing*. STOC '84. Association for Computing Machinery, 1984, pp. 471–474.

[Raz87] A. A. Razborov. "Lower bounds on the size of bounded depth circuits over a complete basis with logical addition". In: *Mathematical Notes of the Academy of Sciences of the USSR* 41.4 (1987), pp. 333–338.

[Smo87] Roman Smolensky. "Algebraic methods in the theory of lower bounds for Boolean circuit complexity". In: *Proceedings of the Sixteenth Annual ACM Symposium on Theory of Computing*. STOC '87. Association for Computing Machinery, 1987, pp. 77–82.

[BT88] David A. Mix Barrington and Denis Thérien. "Finite Monoids and the Fine Structure of NC^1". In: *J. ACM* 35.4 (Oct. 1988), pp. 941–952.

[Dam90] Carsten Damm. "Problems complete for $\oplus\mathsf{L}$". In: *International Meeting of Young Computer Scientists* (1990), pp. 130–137.

[MV91] Yossi Matias and Uzi Vishkin. "Converting High Probability into Nearly-Constant Time—with Applications to Parallel Hashing". In: *Proceedings of the Twenty-Third Annual ACM Symposium on Theory of Computing*. STOC '91. New Orleans, Louisiana, USA: Association for Computing Machinery, 1991, pp. 307–316.

[Sho97] Peter W. Shor. "Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer". In: *SIAM J. Comput.* 26.5 (Oct. 1997), pp. 1484–1509.

[RB01] Robert Raussendorf and Hans J. Briegel. "A One-Way Quantum Computer". In: *Phys. Rev. Lett.* 86 (22 May 2001), pp. 5188–5191.

[RBB03] Robert Raussendorf, Daniel E. Browne, and Hans J. Briegel. "Measurement-based quantum computation on cluster states". In: *Phys. Rev. A* 68 (2 Aug. 2003), p. 022312.

[AIK06] Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. "Cryptography in NC^0". In: *SIAM J. Comput.* 36.4 (2006), pp. 845–888.

[BJS10] 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 A: Mathematical, Physical and Engineering Sciences*. 2010, pp. 459–472.

[AA11] Scott Aaronson and Alex Arkhipov. "The Computational Complexity of Linear Optics". In: *Proceedings of the Forty-Third Annual ACM Symposium on Theory of Computing*. STOC '11. Association for Computing Machinery, 2011, pp. 333–342. ISBN: 9781450306911.

[Boi+18] S. Boixo et al. "Characterizing quantum supremacy in near-term devices". In: *Nature Physics* 14.6 (2018), pp. 595–600.

[BGK18] Sergey Bravyi, David Gosset, and Robert Koenig. "Quantum advantage with shallow circuits". In: *Science* 362 (6412 Oct. 2018), pp. 308–311.

[FGL18] Omar Fawzi, Antoine Grospellier, and Anthony Leverrier. "Constant overhead quantum fault-tolerance with quantum expander codes". In: IEEE. 59th Annual Symposium on Foundations of Computer Science (FOCS), 2018, pp. 743–754.

[Ben+19] Adam Bene Watts et al. "Exponential separation between shallow quantum circuits and unbounded fan-in shallow classical circuits". In: *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*. STOC '19. Association for Computing Machinery, 2019.

[Bra+20] Sergey Bravyi et al. "Quantum advantage with noisy shallow circuits". In: *Nature Physics* (2020), pp. 595–600.

[GS20] Daniel Grier and Luke Schaeffer. "Interactive shallow Clifford circuits: quantum advantage against NC^1 and beyond". In: *Proceedings of the Sixteenth Annual ACM Symposium on Theory of Computing*. STOC '20. Association for Computing Machinery, 2020.