Quasi-polynomial Time Approximation of Output Probabilities of Geometrically-local, Shallow Quantum Circuits

Nolan J. Coble[*†], Matthew Coudron[*‡||]
[*]*Joint Center for Quantum Information and Computer Science, University of Maryland*
[†]*Department of Mathematics, University of Maryland*
[‡]*Institute for Advanced Computer Studies and Department of Computer Science, University of Maryland*
[||]*National Institute of Standards and Technology*

***Abstract*—We present a classical algorithm that, for any 3D geometrically-local, polylogarithmic-depth quantum circuit C, and any bit string $x \in \{0,1\}^n$, can compute the quantity $|\langle x|C|0^{\otimes n}\rangle|^2$ to within any inverse-polynomial additive error in quasi-polynomial time. It is known that it is $\#P$-hard to compute this same quantity to within 2^{-n^2} additive error [1], [2], and worst-case hardness results for this task date back to [3]. The previous best known algorithm for this problem used $O(2^{n^{1/3}}\text{poly}(1/\epsilon))$ time to compute probabilities to within additive error ϵ [4]. Notably, that paper, [4], included an elegant polynomial time algorithm for the same estimation task with 2D circuits, which makes a novel use of 1D Matrix Product States carefully tailored to the 2D geometry of the circuit in question. Surprisingly, it is not clear that it is possible to extend this use of MPS to address the case of 3D circuits in polynomial time. This raises a natural question as to whether the computational complexity of the 3D problem might be drastically higher than that of the 2D problem. In this work we address this question by exhibiting a quasi-polynomial time algorithm for the 3D case. In order to surpass the technical barriers encountered by previously known techniques we are forced to pursue a novel approach: constructing a recursive sub-division of the given 3D circuit using carefully designed block-encodings. To our knowledge this is the first use of the block-encoding technique in a purely classical algorithm.**

Our algorithm has a Divide-and-Conquer structure, demonstrating how to approximate the desired quantity via several instantiations of the same problem type, each involving 3D-local circuits on about half the number of qubits as the original. This division step is then applied recursively, expressing the original quantity as a weighted sum of smaller and smaller 3D-local quantum circuits. A central technical challenge is to control correlations arising from entanglement that may exist between the different circuit "pieces" produced this way. We believe that the division step, which makes a novel use of block-encodings [5], together with an Inclusion-Exclusion argument to reduce error in each recursive approximation, may be of independent interest.[1]

***Keywords*-quantum computing; low-depth circuits; lightcone; block encodings;**

I. Introduction

Many schemes for obtaining a quantum computational advantage with near-term quantum hardware are motivated by mathematical results proving the computational hardness of sampling from near-term quantum circuits. In this work we consider quantum circuits which are geometrically local and have polylogarithmic circuit-depth. It is known to be $\#P$-hard to compute output probabilities of n-qubit, geometrically-local, constant-depth quantum circuits to within 2^{-n^2} additive error [1], a result which builds on an extensive line of research focusing on the hardness of sampling from quantum circuits [6]–[10]. It has even been shown, under several computational assumptions, that there is no classical polynomial time algorithm which, given a geometrically-local, constant-depth quantum circuit, K, can produce samples whose distribution lies within a constant, in the ℓ_1 distance, of the output distribution of K in the computational basis [11].

A. Prior work

A series of works on the classical complexity of sampling from near-term quantum circuits, and related tasks, highlights the subtle nature of identifying an actual quantum advantage based on these tasks [12]–[14]. These results frame the significance of the algorithm presented as Theorem 5 in [4], which estimates output probabilites of 2D-local constant depth circuits to inverse polynomial additive error in polynomial time. In fact, the original algorithm in [4], actually estimates quantities of the form $\langle 0^{\otimes n}|\, C^{\dagger}\, (\otimes_{i=1}^{n} P_i)\, C\, |0^{\otimes n}\rangle$, where each $P_i \in \{X,Y,Z,I\}$ is a single-qubit Pauli observable operator. However, it is straightforward to convert their algorithm to compute the quantity $\langle 0^{\otimes n}|\, C^{\dagger}\, (\otimes_{i=1}^{n} |x_i\rangle\langle x_i|)\, C\, |0^{\otimes n}\rangle = |\langle x|\, C\, |0^{\otimes n}\rangle|^2$, $x \in \{0,1\}^n$, instead. Theorem 5 of [4] constitutes a pertinent observation. While it is hard to sample from constant-depth quantum circuits, it is still unresolved whether it is hard to estimate any property of such a circuit which could have been computed using a polynomial number of samples from the output of the quantum circuit itself. In particular: A polynomial number of samples from a 2D-local, constant-depth quantum circuit only allows one to estimate output probabilities of that circuit to inverse polynomial additive error. But, it is shown in Theorem 5 of [4] that this same task can be done in classical polynomial time! One

[1]The full version of this paper is available at arXiv:2012.05460. We refer the reader to the full text for proofs of all Lemmas and Theorems.

might ask: Is there a well-defined Decision problem which can be solved using only a polynomial number of samples from such a quantum circuit, together with classical post-processing, and yet cannot also be efficiently solved using classical computing alone? This is unknown.

We note, at this point, some basic facts about the task of computing the quantity $|\langle 0^{\otimes n}| C |0^{\otimes n}\rangle|^2$ which explain why we can focus on this task WLOG, and may motivate our interest in it:

- If there is an algorithm to estimate the quantity $|\langle 0^{\otimes n}| C |0^{\otimes n}\rangle|^2$, for any 3D-local depth-d quantum circuit C, then that algorithm can be used to estimate $|\langle x| C |0^{\otimes n}\rangle|^2$ for any $x \in \{0,1\}^n$. The reason is that $|\langle x| C |0^{\otimes n}\rangle|^2 = |\langle 0^{\otimes n}| G |0^{\otimes n}\rangle|^2$ where G is taken to be the 3D-local circuit $G \equiv C\,(\otimes_{i=1}^n X^{x_i})$. Here X represents the single qubit Pauli operator σ_X. Note that G is still a depth-$O(d)$ quantum circuit.
- Any such algorithm can also estimate $|\langle 0^{\otimes n}| C Z^n C^\dagger |0^{\otimes n}\rangle|^2$, which is the magnitude of the expected bias of the Parity of the output bits of C, when measured in the computational basis. This is true by virtue of the fact that $CZ^nC^\dagger$ is, itself, a 3D local, depth-$O(d)$ circuit. So, this type of computational problem allows us to study the power of depth-d geometrically-local, quantum circuits combined with certain limited types of classical post-processing, like the Parity function.
- The algorithm we present in this work can easily be modified to approximate marginal probabilities (e.g., the probability that $x_1 = 1$ for $x \in \{0,1\}^n$ sampled from the given circuit, etc). Consequently, it is straightforward to use this algorithm to search for all $x \in \{0,1\}^n$ which have probability at least δ in the output distribution of a given depth-d geometrically-local circuit C. That is, searching for all of the "δ-heavy" strings of C. When $\delta = 1/\mathsf{poly}(n)$ there can be at most $\mathsf{poly}(n)$ such strings and our algorithm can find them all in quasi-polynomial time.

The algorithm for 2D circuits presented in Theorem 5 of [4] makes a novel use of 1D Matrix Product States, carefully tailored to the 2D geometry of the circuit in question. However, the authors of [4] point out that it is not clear that it is possible to extend this use of MPS to address the case of 3D circuits in polynomial time. Instead they provide a sub-exponential time algorithm for the 3D case, which has time complexity $O(2^{n^{1/3}}\mathsf{poly}(1/\epsilon))$ for computing the desired quantity to within additive error ϵ. In this work we introduce a new set of techniques culminating in a divide-and-conquer algorithm which solves the 3D case in quasi-polynomial time.

B. Our contributions

Our algorithm has a divide-and-conquer structure with the goal being to divide the circuit C into pieces, and reduce the original problem to a small number of new 3D-circuit problems involving circuits on only a fraction of the number of qubits as the original. This division step requires the ability to construct Schmidt vectors of the state $C|0^{\otimes n}\rangle$, across a given cut, via a depth-d geometrically-local quantum circuit, so that the new subproblems can be expressed as smaller instantiations of the original problem type. We accomplish this through the use of block-encodings, a technique designed for quantum algorithms [5], but used here as a subroutine of a classical simulation algorithm instead. However, to date, we are only able to construct, as a block-encoding circuit, the *leading* Schmidt vector across certain "heavy" cuts. Due to this restriction we are forced to use a novel division step in our Divide-and-Conquer approach. Instead of dividing about a single cut and constructing many of its Schmidt vectors as depth-d geometrically-local block-encodings, we must divide across many cuts and construct only their leading Schmidt vectors. Interestingly, this process can still lead to low approximation error via an Inclusion-Exclusion style argument, as shown in Lemma 18.

These techniques culminate in a worst-case quasi-polynomial time algorithm for 3D circuits, which is our main result:

Theorem 1. *There exists a classical algorithm which, for any 3D geometrically-local, depth-d quantum circuit C on n qubits, can compute the scalar quantity $|\langle 0^{\otimes n}| C |0^{\otimes n}\rangle|^2$ to within $1/n^{\log(n)}$ additive error in time*

$$T(n) = 2^{d^3\,\mathsf{polylog}(n)} \tag{1}$$

See Algorithm 1 in Section IV for a precise definition of this classical algorithm.

Note that our Theorem statement gives an inverse quasi-polynomial additive error approximation. This is, therefore, asymptotically better than an inverse polynomial additive approximation for any polynomial. There is a more explicit trade-off between runtime and approximation error given in Theorem 28, and in fact, Theorem 1 follows from Theorem 28 with $\delta = 1/n^{\log(n)}$, but we use the above statement here for simplicity. Note also, when the depth, d, is polylogarithmic our algorithm runs in quasi-polynomial time.

C. Organization

We begin with some preliminary definitions in Section II. In Section III we detail the methods for approximating $|\langle x| C |0^{\otimes n}\rangle|^2$ for a geometrically-local quantum circuit by recursively defining smaller geometrically-local circuits. In Section IV we detail the main Algorithm 1 along with the recursive subroutine Algorithm 2. In Subsection IV-A we give an overview for the error and run-time analysis of Algorithm 1. In Subsection IV-B and Subsection IV-C we give an overview of the error and run-time analyses of Algorithm 2, respectively.

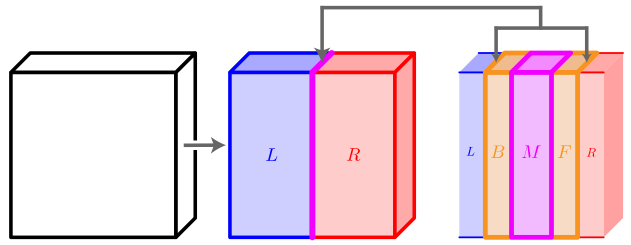

Figure 1. Cutting the Cube: (left) 3D cube of qubits. (center) Choose a location to cut the qubits. The qubits to the left and right of the cut are denoted by L and R, respectively. (right) Within the cut there are three regions: a center region M and regions to the left and right of M, denoted by B and F, respectively.

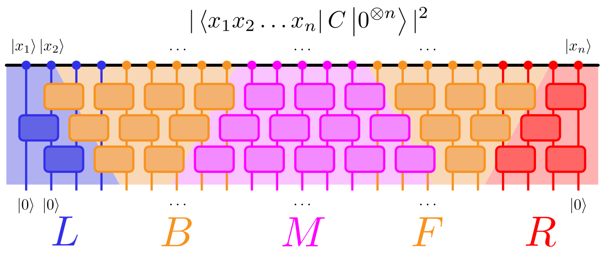
$|\langle x_1x_2\dots x_n| C |0^{\otimes n}\rangle|^2$
$|x_1\rangle$ $|x_2\rangle$ $\dots$ $\dots$ $\dots$ $|x_n\rangle$
$|0\rangle$ $|0\rangle$ $\dots$ $\dots$ $\dots$ $|0\rangle$
L
B
M
F
R

Figure 2. Block depiction of the unitaries defined in Definition 3 and Definition 5 for the case of a 1D geometrically-local constant-depth circuit C. The input to the circuit is at the bottom and the output is at the top. $C_{B\cup M\cup F}$ is defined to be the unitary produced by the gates in the reverse lightcone of M, which are colored magenta in this diagram. C_{wrap} is the unitary consisting of all the orange gates in the diagram. Furthermore, C'_L (resp. C'_R) denote the unitaries consisting of all the blue (resp. red) gates in the diagram. **Note:** we do not have an analogous figure for 3D circuits, which are the main focus of this work, because it would require 4 dimensions to illustrate. However, we believe the reader will gain sufficient intuition from the 1D figure.

II. Preliminaries

A. *Dividing the Cube: Some Notation*

Given a 3D-local, depth-d circuit C, we wish to estimate the quantity $|\langle 0^{\otimes n}| C |0^{\otimes n}\rangle|^2$. To begin our divide-and-conquer approach we will divide the circuit C in half via a cut through the center as shown in Figure 1. The width of the cut is dependent on d, and we will discuss how to select this width below. To begin with, we make the width large enough to have the non-empty sets B, M, and F defined below.

Definition 2 (M, B, F, R, and L (see Figure 1))**.** Let M be the set of all qubits in the “Middle of the cut” (the middle part of the cut which is not in the lightcone of qubits from outside the cut). Let B be the set of all qubits within the cut which are to the left of M. Let F be the set of all qubits within the cut which are to the right of M. We will choose the width of M to be $O(d)$ such that the lightcones of B and F are disjoint. We will choose the widths of B and F to be $O(d)$ such that the lightcone of M is contained in $B \cup M \cup F$. For concreteness we set the width of each of B, M, F to be $10d$. Since C is geometrically-local, this is sufficiently large width to satisfy the above conditions on lightcones.

Let L be all qubits outside the cut which are to the left of the cut (that is, to the left of B). The set L is colored blue. Let R be the set of all qubits outside to the right of the cut (that is, to the right of F). The set R is colored red.

We will now define a 2D geometrically local, depth-d circuit $C_{B\cup M\cup F}$ which can be thought of as the sub-circuit of C which lies within the reverse light-cone of M. Intuitively this circuit captures all of the local information that must be accounted for in the division step across this particular slice in our divide-and-conquer algorithm.

Definition 3 ($C_{B\cup M\cup F}$)**.** Now, let us begin with the all zeroes state on all the qubits $|0\rangle_{L\cup B\cup M\cup F\cup R} = |0_{ALL}\rangle$, and apply the minimum number of gates from the circuit C such that every gate on the qubits within M has been applied. We will call this unitary $C_{B\cup M\cup F}$. Note that this unitary does not act on any qubits outside of $B \cup M \cup F$. This is because the reverse lightcone of M is contained in $B \cup M \cup F$ by Definition 2. Note that $C_{B\cup M\cup F}$ can be thought of as an approximately 2D (not 3D) geometrically-local, depth-d circuit, since the third dimension of the circuit is $O(d)$ which for $d = \mathrm{polylog}(n)$ grows asymptotically slower than $O(n^{1/3})$.

We define $C_{L\cup R}$ to be the unitary composed of the remainder of the gates of C not yet applied in $C_{B\cup M\cup F}$, so that $C = C_{L\cup R} \circ C_{B\cup M\cup F}$. We define C_L (resp. C_R) to be the unitaries composed of the remainder of the gates of C not yet applied in $C_{B\cup M\cup F}$ and which lie to the left (resp. right) of the M. Note that $C_L \circ C_R = C_{L\cup R}$ since none of the circuits $C_L, C_R, C_{L\cup R}$ act non-trivially on M. See Figure 2 for an illustration of these unitaries with a 1D geometrically-local circuit.

The sub-normalized quantum state produced by $C_{B\cup M\cup F}$, defined below, is the state whose Schmidt decomposition we consider in our division step.

Definition 4. Let $|\psi\rangle_{B\cup F} \equiv \langle 0|_M C_{B\cup M\cup F} |0\rangle_{B\cup M\cup F}$.

Note that, $\langle 0|_{\mathrm{ALL}} C_{L\cup R} |0\rangle_{L\cup R} \otimes |\psi\rangle_{B\cup F} = \langle 0|_{\mathrm{ALL}} C |0\rangle_{\mathrm{ALL}}$.

(Throughout this document, the notation $|0_{ALL}\rangle$ will refer to the zero state on all unmeasured qubits for a given state. It's meaning will be clear from context.)

Definition 5 (C_{Wrap})**.** Define a new unitary C_{Wrap} which consists of all the gates from C which are in the light-cone of $B\cup M\cup F$, but not in $C_{B\cup M\cup F}$ itself. That is, let C_{L-Wrap} (resp. C_{R-Wrap}) be the unitary consisting of all the of the gates in C which are in the light-cone of B (resp. F), but not in $C_{B\cup M\cup F}$ itself, and let $C_{Wrap} \equiv C_{L-Wrap} \circ C_{R-Wrap}$. Therefore,

$$C_{Wrap}^{\dagger} \circ C = C'_L \circ C_{B\cup M\cup F} \circ C'_R \tag{2}$$

Where $C'_L \equiv C_{L-Wrap}^{\dagger} \circ C_L$ (see Definition 3 for the definition of C_L) is a unitary acting only or L (the remaining, untouched gates of C within L), and $C'_R \equiv C_{R-Wrap}^{\dagger} \circ C_R$ (see Definition 3 for the definition of C_R) is a unitary acting only on R (the remaining, untouched gates of C within R). Since C is depth-d it is clear that every qubit in the non-trivial support of C_{Wrap} lies within some $O(d)$ distance of M. Let R^{Wrap} (resp. L^{Wrap}) be the subset of qubits in R (resp. L) that lie in the non-trivial support of C_{Wrap}. In other words, R^{Wrap} (resp. L^{Wrap}) is the non-trivial support of C_{R-Wrap} (resp. C_{L-Wrap}). See Figure 2 for an illustration of these unitaries with a 1D geometrically-local circuit.

B. Schmidt vectors and block-encodings

In this section we will show how to construct a geometrically-local, shallow quantum circuit for the largest Schmidt vector of the unnormalized state $\langle 0_M| C |0_{ALL}\rangle$ across the cut M, in the case that the largest Schmidt coefficient is very large. Let us begin, however, by outlining the intuition behind our divide-and-conquer approach, which explains why we are interested in approximating Schmidt vectors via shallow quantum circuits in the first place. Consider expanding the quantity $\langle 0_{ALL}| C |0_{ALL}\rangle = \langle 0|_{\text{ALL}} C_{L\cup R} |0\rangle_{L\cup R} \otimes |\psi\rangle_{B\cup F}$ as a sum over the Schmidt decomposition of $|\psi\rangle_{B\cup F}$ across the cut M. Suppose, that $|\psi\rangle_{B\cup F}$ has almost all of its weight on the top polynomially many Schmidt vectors (In Section III we will show that, in fact, we can restrict this part of the analysis WLOG to cases where $|\psi\rangle_{B\cup F}$ has a large fraction of its weight on λ_1). Then $|\psi\rangle_{B\cup F} \approx \sum_{i=1}^{p(n)} \lambda_i |v_i\rangle_B \otimes |w_i\rangle_F$, and we have:

$$\langle 0|_{\text{ALL}} C_{L\cup R} |0\rangle_{L\cup R} \otimes |\psi\rangle_{B\cup F} \tag{3}$$

$$\approx \sum_{i=1}^{p(n)} \lambda_i \langle 0|_{\text{ALL}} C_{L\cup R} |0\rangle_{L\cup R} \otimes |v_i\rangle_B \otimes |w_i\rangle_F \tag{4}$$

$$= \sum_{i=1}^{p(n)} \lambda_i \langle 0|_{L\cup B} C_L |0\rangle_L \otimes |v_i\rangle_B \langle 0|_{F\cup R} C_R |0\rangle_R \otimes |w_i\rangle_F \tag{5}$$

where $\text{ALL} \equiv L \cup B \cup F \cup R$.

Suppose we could produce approximations for the Schmidt vectors $|v_i\rangle_B$ and $|w_i\rangle_F$ via 2D geometrically-local, shallow quantum circuits. Then, the quantity in Equation 5 would be a sum of polynomially many scalar quantities, each of which is the product of output probabilities of two new 3D geometrically-local circuit problems (C_L and C_R). Furthermore, these new 3D circuit problems involve about half the number of qubits as the original problem we were trying to solve. This leads to a divide-and-conquer recursion which can yield a more efficient runtime for the original problem. The base case in this divide-and-conquer algorithm consists of estimating output probabilities of 3D-local, depth-d quantum circuits which have small width (width at most $w = \text{polylog}(n)$) in one of their dimensions. This base case can be solved efficiently using the algorithm from Theorem 5 of [4], as discussed in Remark 6 below.

Note that this divide-and-conquer approach only works if we can produce explicit approximations for the Schmidt vectors $|v_i\rangle_B$ and $|w_i\rangle_F$ via 2D geometrically-local, shallow quantum circuits. In the case when λ_1 is sufficiently large, it turns out that we can at least produce the top Schmidt vectors $|v_1\rangle_B$ and $|w_1\rangle_F$ in this way. (Note, we will also need to compute λ_1 efficiently, and this can also be done using Theorem 5 of [4], as described in Definition 25 and Remark 6.) However, we do not know how to construct 2D geometrically-local, shallow quantum circuits that approximate $|v_i\rangle_B$ and $|w_i\rangle_F$ for $i > 1$, and so we cannot pursue the divide-and-conquer approach described in Equation 5 verbatim. Nonetheless, we will see in Section III that just approximating the top Schmidt vectors $|v_1\rangle_B$ and $|w_1\rangle_F$ is already sufficient to produce a (more involved) divide-and-conquer algorithm for the whole estimation problem. The complete algorithm is explicitly written out in Section IV (see Algorithms 1 and 2). The key additional insight is to combine the intuition from Equation 5 above, with an additional expansion trick, expressed in Lemma 18.

Remark 6. Theorem 5 of [4] shows that the output probabilities of 2D constant-depth circuits can be computed to inverse polynomial additive error in polynomial time. Technically, this does not exactly cover the base case of our divide-and-conquer approach because our base case will consist of circuits which are 3D, but have a small width in the third dimension. One might say that the base case circuits have a 2D structure with small "thickness" in the third dimension. Fortunately, this extended case is also covered by additional analysis from the [4] paper, in which the authors show, on pages 25 and 26 (of the arXiv version), that a slightly modified version of their algorithm can, in fact, compute output probabilities of 3D-local, depth-d circuits to additive error ϵ in time $n\epsilon^{-2}2^{O(d^2 \cdot w)}$, where w is the width of the third dimension of the circuit. For convenience, throughout the remainder of this paper every reference to Theorem 5 of [4] will refer instead to this modified algorithm which can handle these "small-width" 3D-local, depth-d circuits. Additionally, when we refer to 2D-local circuits we are including, within that definition, 3D-local circuits where the width in the third dimension is $w = \text{polylog}(n)$. The reason that this is a reasonable use of terminology in the context of this paper is that Theorem 5 of [4], and the subsequent discussion, can handle these small-width 3D-local circuits in time exponential in the size of w (which, for $w = \mathsf{polylog}(n)$, is quasi-polynomial).

Our approach for explicitly constructing $|w_1\rangle_F$ is based on a tool called a "block-encoding", which aims to generate a unitary whose top left corner contains the Hermitian matrix $\rho_F \equiv \mathrm{tr}_B(|\psi\rangle\langle\psi|_{B\cup F})$, or the integer powers ρ_F^K for $K = \mathsf{polylog}(n)$. In fact, under an assumption that λ_1 is sufficiently large, $\frac{1}{\lambda_1^K}\rho_F^K$ is already very close to a projector onto $|w_1\rangle_F$ (see Lemma 15 for the explicit scaling).

Lemma 7 (Lemma 45 of [5])**.** *The following is a 2D-local (see Remark 6), depth-d circuit which gives a block encoding for $\rho_F \equiv \mathrm{tr}_B(|\psi\rangle\langle\psi|_{B\cup F})$:*

$$(C^\dagger_{B\cup M\cup F}\otimes I_{F'})(I_{B\cup M}\otimes \mathit{SWAP}_{FF'})(C_{B\cup M\cup F}\otimes I_{F'})$$

We therefore have the following Lemma.

Lemma 8 (Lemma 53 of [5])**.** *For any constant integer $K > 0$, the following is a 2D-local (see Remark 6) quantum circuit which gives a block encoding for ρ_F^K, and has depth $O(dK^2)$:*

$$V_{\rho_F^k} = \prod_{i=1}^{k}\left(C^\dagger_{B_i\cup M_i'\cup F_i'}\otimes I_{M_i\cup F}\right)\cdot \left(I_{B_i}\otimes SWAP_{M_i\cup F, M_i'\cup F_i'}\right)\left(C_{B_i\cup M_i'\cup F_i'}\otimes I_{M_i\cup F}\right). \quad (6)$$

In other words,

$$\rho_F^k = \left(\langle 0|_{\mathcal{B}_k\cup\mathcal{M}_k'\cup\mathcal{F}_k'\cup\mathcal{M}_k}\otimes I_F\right) V_{\rho_F^k}\left(|0\rangle_{\mathcal{B}_k\cup\mathcal{M}_k'\cup\mathcal{F}_k'\cup\mathcal{M}_k}\otimes I_F\right)$$

where $\mathcal{B}_k = B_1\cup B_2\cup\cdots\cup B_k$, $\mathcal{M}_k' = M_1'\cup M_2'\cup\cdots\cup M_k'$, etc.

III. Divide and Conquer: Splitting over Heavy Slices

In this section we will prove a set of results which will allow us to precisely define and analyze the division step in our divide-and-conquer algorithm. The process begins by identifying slices of the depth-d circuit C which are appropriate division points. Those are the slices which have "heavy weight" as defined below.

Consider a set of $O(d)$-width 2D slices $K = \{K_i\}$ of the qubits of C, where each slice K_i is parallel to the cut $B\cup M\cup F$ shown in Figure 1, and is made up of three analogous sections B_i, M_i, F_i (see Figure 3). Let the the slices in K be evenly spaced at an $O(d)$ distance apart, where this value is chosen to be large enough that the light cones of K_i and K_j are disjoint when $i \neq j$. For concreteness we will say that the distance between slices K_i is equal to $10d$. We will also set the width of each of the sections B_i, M_i, F_i to be $10d$, just as discussed in Definition 2. This ensures that the properties stipulated by Definition 2 are satisfied by B_i, M_i, F_i.

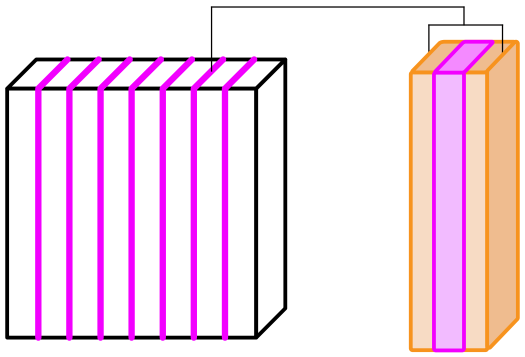

Figure 3. Set of slices $\{K_i\}$

Definition 9. Let $I[M_i = 0]$ be the indicator random variable for the event that all of the qubits in M_i collapse to 0 when measured in the computational basis. Here joint probabilities are defined according the probability distribution p_{total} produced by measuring $C|0^{\otimes n}\rangle$ in the computational basis. Let $p_{M_i=0} := \mathbb{E}_{p_{total}}[I[M_i = 0]]$ be the probability that all of the bits in K_i evaluate to 0 according to the distribution p_{total}.

Lemma 10. *The $I[M_i = 0]$ are independent random variables. Therefore,*

$$p_{total}(M_i = 0\ \forall i) = \prod_i p_{total}(M_i = 0).$$

Note that the variables $I[M_i = 0]$ may well be *conditionally* dependent when conditioned on the outcomes of measuring the qubits in between the K_i slices. Indeed, that's what makes the global problem non-trivial in the first place. But, when measuring the K_i slices alone we see that the $I[M_i = 0]$ are independent as stated in Lemma 10.

Lemma 11. *If $|\langle 0^{\otimes n}|C|0^{\otimes n}\rangle| > |1/q(n)|$, then, for any $0 \le h \le 1$, $h|K|$ of the slices K_i in K have the property that:*

$$p_{total}(M_i = 0) \ge (|1/q(n)|)^{\frac{1}{(1-h)|K|}} \quad (7)$$

We will let K_{heavy} be the subset of K consisting of those K_i satisfying Equation (7).

Definition 12. We define any particular slice K_i as $K_i = B_i\cup M_i\cup F_i$., where B_i, M_i, F_i are the analogous regions to B, M, F (respectively) in Figure 1. These slices are depicted in Figure 3. Let $|\psi\rangle_{B_i\cup F_i}$ be analogous to $|\psi\rangle_{B\cup F} \equiv \langle 0|_M C_{B\cup M\cup F}|0\rangle_{B\cup M\cup F}$. Let L_i and R_i be sets of qubits analogous to the sets L and R. We define the unitaries $C_{B_i\cup M_i\cup F_i}$, C_{wrap_i}, $C'_{L_i} \equiv C^\dagger_{L-Wrap_i}\circ C_{L_i}$, and $C'_{R_i} \equiv C^\dagger_{R-Wrap_i}\circ C_{R_i}$ exactly as given in Definition 5 for the case of a single cut.

Lemma 13. *For any slice $K_i \in K_{heavy}$ satisfying:*

$$p_{total}(M_i = 0) \geq 1 - e(n), \tag{8}$$

the top Schmidt coefficient of $|\psi\rangle_{B_i \cup F_i}$ satisfies $\lambda_1^i \geq 1 - O(e(n))$. (Where the Schmidt decomposition is taken across the partition B_i, F_i.)

The proof of Lemma 13 (See full version) suggests a way to perform a division step, dividing the original computational problem into the product of two new problems, but at the cost of an additive error that scales like $\Theta(e(n))$. But, note that, if we want, say, $1/n^d$ additive error for $d \geq 2$, then this additive error term is way too large (in some cases $e(n)$ scales like $1/\log(n)$). This means that, a priori, we cannot even afford to make use of Lemma 13 one single time! However, Lemma 20 below shows how we can use this type of division step to divide the circuit at Δ different, light-cone separated cuts, K_i, simultaneously, and thereby achieve additive error that scales like $e(n)^{\Delta}$.

Definition 14. For any K_i define the following two operators inspired by the block-encoding approach in Section II-B:

$$P_{F_i}^K \equiv \frac{1}{\lambda_1^K} \left\langle 0^{B_i, M_i, F_i^1, \ldots F_i^k} \middle| V_{P_{F_i}^k} \middle| 0^{B_i, M_i, F_i^1, \ldots F_i^k} \right\rangle \tag{9}$$

and

$$P_{B_i}^K \equiv \frac{1}{\lambda_1^K} \left\langle 0^{F_i, M_i, B_i^1, \ldots B_i^K} \middle| V_{P_{B_i}^k} \middle| 0^{F_i, M_i, B_i^1, \ldots B_i^K} \right\rangle \tag{10}$$

Note that we haven't explicitly defined $V_{P_{F_i}^k}$ and $V_{P_{B_i}^k}$, but the definitions are analogous to that of the quantum circuit given in Lemma 8.

The first equation gives a linear operator on F_i, and the second equation gives a linear operator on B_i. The registers F_i^j (resp. B_i^j) are dummy registers that are used to create K block encodings of the density matrix of the F_i (resp. B_i) register of the state $C_{B_i \cup M_i \cup F_i} \left|0^{F_i, M_i, B_i}\right\rangle$. These K block encodings are then composed (multiplied) with each other in such a manner that they produce the block encoding of the K^{th} power of the density matrix, as described in Section II-B.

Lemma 15. *For any $K_i \in K_{heavy}$,*

$$\|P_{F_i}^K - |w_1\rangle \langle w_1|_{F_i}\|_1 \leq g(n) \tag{11}$$

and

$$\|P_{B_i}^K - |v_1\rangle \langle v_1|_{B_i}\|_1 \leq g(n) \tag{12}$$

where $g(n) \equiv \left(\frac{1-\lambda_1^i}{\lambda_1^i}\right)^K$, and $|w_1\rangle \langle w_1|_{F_i}$, $|v_1\rangle \langle v_1|_{B_i}$ are the projectors onto the top Schmidt vectors of $|\psi\rangle_{B_i \cup F_i}$ in F_i and B_i respectively.

Definition 16. Define $\Pi_{F_i}^K \equiv C_{Wrap_i} P_{F_i}^K C_{Wrap_i}^{\dagger}$.

Note that the operator $\Pi_{F_i}^K$ is in tensor product with $|0_{M_i}\rangle$ (it acts as the identity on the M_i register since C_{Wrap_i} and $P_{F_i}^K$ act trivially on that register).

Definition 17. Let $\sigma \in \mathcal{P}[\Delta] \setminus \emptyset$ where $[\Delta] = \{1, \ldots, \Delta\}$. Define the unormalized states

$$|\Psi_\sigma\rangle = \otimes_{j \in \sigma} \Pi_{F_j}^K \otimes_{i \in [\Delta]} \langle 0_{M_i}| C |0_{ALL}\rangle$$

And,

$$|\Psi_\emptyset\rangle = \otimes_{i \in [\Delta]} \langle 0_{M_i}| C |0_{ALL}\rangle$$

Lemma 18. *Consider a set K_{heavy} of slices such that, for every $K_i \in K_{heavy}$, $|\psi\rangle_{B_i \cup F_i}$ satisfies $\lambda_1^i \geq 1 - e(n)$, and such that for any $K_i, K_j \in K_{heavy}$, the operators $\Pi_{F_i}^K$ and $\Pi_{F_j}^K$ are light-cone separated whenever $i \neq j$. Then, for any set of Δ slices, $\{K_i\}_{i \in [\Delta]} \subseteq K_{heavy}$, we have that:*

$$\left\| \sum_{\sigma \in \mathcal{P}([\Delta])} (-1)^{|\sigma|} |\Psi_\sigma\rangle \langle \Psi_\sigma| \right\| \tag{13}$$

$$= \left\| |\Psi_\emptyset\rangle \langle \Psi_\emptyset| - \sum_{\sigma \in \mathcal{P}([\Delta]) \setminus \emptyset} (-1)^{|\sigma|+1} |\Psi_\sigma\rangle \langle \Psi_\sigma| \right\| \tag{14}$$

$$\leq (2e(n) + 2g(n))^{\Delta}, \tag{15}$$

where $g(n) \equiv \left(\frac{1-\lambda_1^i}{\lambda_1^i}\right)^K$.

Intuition for the statements of Lemmas 18 and 20:: We will show, in Lemma 20 below, that each of the states $|\Psi_\sigma\rangle \langle \Psi_\sigma|$, with $\sigma \neq \emptyset$, is very close to a product state about at least one of the Δ slices. Thus, Lemma 18 gives us a way to approximate $|\Psi_\emptyset\rangle \langle \Psi_\emptyset|$ (which is the original state of interest) by a linear combination of product states $|\Psi_\sigma\rangle \langle \Psi_\sigma|$. Lemma 20 and Definition 14 then provide us with a way of constructing the corresponding product states using low-depth quantum circuits acting on approximately half as many qubits as the original circuit (this process is also further formalized in Definition 23). This combined use of Lemmas 18 and 20 forms the backbone of our divide-and-conquer approach.

In order to state Lemma 20 we now define three new states that are dependent on a particular choice of K_i.

Definition 19. Given a shallow, 3D geometrically local quantum circuit C, and given a slice K_i of C, define the states:

$$|\Omega_i\rangle = \Pi_{F_i}^K \langle 0_{M_i}| C |0_{ALL}\rangle \tag{16}$$

$$|\Xi_{L_i}\rangle = P_{F_i}^K \langle 0_{M_i}| C_{L_i} C_{B_i \cup M_i \cup F_i} |0_{L_i \cup B_i \cup M_i \cup F_i}\rangle \tag{17}$$

$$|\Xi_{R_i}\rangle = P_{B_i}^K \langle 0_{M_i}| C_{R_i} C_{B_i \cup M_i \cup F_i} |0_{R_i \cup B_i \cup M_i \cup F_i}\rangle \tag{18}$$

At this point it is pertinent to state Lemma 20:

Lemma 20. *For any $K_i \in K_{heavy}$ (recall this means that $|\psi\rangle_{B_i \cup F_i}$ satisfies $\lambda_1^i \geq 1 - e(n)$), the state $|\Omega_i\rangle \langle \Omega_i|$ is* *within $6g(n)$ of an unnormalized product state about M_i, described as follows:*

$$\left\| |\Omega_i\rangle\langle\Omega_i| - 1/\lambda_1^i \operatorname{tr}_{F_i}\left(|\Xi_{L_i}\rangle\langle\Xi_{L_i}|\right) \otimes \operatorname{tr}_{B_i}\left(|\Xi_{R_i}\rangle\langle\Xi_{R_i}|\right)\right\| \leq 6g(n) \tag{19}$$

Here $g(n) \equiv \left(\frac{1-\lambda_1^i}{\lambda_1^i}\right)^K \leq \left(\frac{e(n)}{1-e(n)}\right)^K$, as in Lemma 15.

Definition 21 (Synthesis)**.** We say that an unnormalized quantum state ϕ is *synthesized* by a quantum circuit Γ, if Γ has three registers of qubits L, M, N such that:

$$\phi = \phi_{\left(\Gamma,L,M,N\right)} = \operatorname{tr}_{L\cup M}\big(\langle 0_M|\,\Gamma\,|0_{L\cup M\cup N}\rangle \tag{20}$$

$$\langle 0_{L\cup M\cup N}|\,\Gamma^\dagger\,|0_M\rangle\big). \tag{21}$$

In this case we say that the circuit Γ together with a specification of the registers L, M, N constitutes a *synthesis* of ϕ. When ϕ is implicit we will call this collection (Γ, L, M, N) a *synthesis*.

When Γ is a 3D geometrically-local, depth-d circuit, and the register N is one contiguous cubic subset of the qubits that Γ acts on, with L, and M only containing qubits on the "edges", we call (Γ, L, M, N) a 3D geometrically-local, depth-d *synthesis*.

Definition 22. [The Circuits $\Gamma_{i,j}$, $\Gamma_{L,i}$, $\Gamma_{R,j}$] Recall, from Definition 3, that $\Gamma_{B_k\cup M_k\cup F_k}$ ($k \in \{i,j\}$) is defined to be the circuit containing the minimal number of gates of Γ such that every gate acting on M_k is included. Taking this definition for both $k = i$ and $k = j$, we now define $\Gamma_{i,j}$ to be a sub-circuit of Γ consisting of the minimal number of gates of Γ such that $\Gamma_{i,j}\circ\Gamma_{B_i\cup M_i\cup F_i}\circ\Gamma_{B_j\cup M_j\cup F_j}$ contains all of the gates of Γ that lie between M_i and M_j. Similarly define Γ_{L_i} (resp. Γ_{R_j}) to be the sub-circuit of Γ consisting of the minimal number of gates of Γ such that $\Gamma_{L_i}\circ\Gamma_{B_i\cup M_i\cup F_i}$ (resp. $\Gamma_{R_j}\circ\Gamma_{B_j\cup M_j\cup F_j}$) contains all of the gates of Γ that lie between M_i (resp. M_j)and the left-hand side (resp. right-hand side) of the Cube.

Definition 23. Let $S = (\Gamma, G, H, N)$ be a 3D local, depth-d synthesis, and let K_i, K_j ($i < j$) be two slices on the register N, as described in Definition 12. Let M_i, F_i, B_i, and M_j, F_j, B_j be the sub-registers of slices K_i and K_j respectively, as defined in Definition 12. Recall, from Definition 21, that the state synthesized by S is:

$$\phi_S = \operatorname{tr}_{G\cup H}(\langle 0_H|\,\Gamma\,|0_{G\cup H\cup N}\rangle\,\langle 0_{G\cup H\cup N}|\,\Gamma^\dagger\,|0_H\rangle).$$

We define three new pure states as follows:

$$\begin{aligned}
|\varphi_{\mathsf{L},i}\rangle &= (\lambda_1^i)^K P_{F_i}^K \langle 0_{M_i,H}|\,\Gamma_{L_i}\Gamma_{B_i\cup M_i\cup F_i} \\
&\qquad |0_{L_i\cup B_i\cup M_i\cup F_i\cup G\cup H}\rangle \\
|\varphi_{j,\mathsf{R}}\rangle &= (\lambda_1^j)^K P_{B_j}^K \langle 0_{M_j,H}|\,\Gamma_{R_j}\Gamma_{B_j\cup M_j\cup F_j} \\
&\qquad |0_{R_j\cup B_j\cup M_j\cup F_j\cup G\cup H}\rangle \\
|\varphi_{i,j}\rangle &= (\lambda_1^i\lambda_1^j)^K P_{B_i}^K \circ P_{F_j}^K \langle 0_{M_i,M_j,H}|\,\Gamma_{i,j}\circ \\
&\qquad \Gamma_{B_i\cup M_i\cup F_i}\circ\Gamma_{B_j\cup M_j\cup F_j} \\
&\qquad |0_{N_{i,j}\cup B_i\cup M_i\cup F_i\cup B_j\cup M_j\cup F_j\cup G\cup H}\rangle
\end{aligned} \tag{22}$$

Here $P_{F_i}^K, P_{B_j}^K$ are defined as in Definition 14. In the above the notation $N_{i,j}$ is defined to be the sub-register of N containing all of the qubits between F_i and B_j.

From these, we define three new synthesized states (with corresponding syntheses) as follows:

$$\begin{aligned}
\phi_{\mathsf{L},i} &= \operatorname{tr}_{F_i\cup M_i\cup G\cup H}\left(|\varphi_{\mathsf{L},i}\rangle\langle\varphi_{\mathsf{L},i}|\right) \\
\phi_{j,\mathsf{R}} &= \operatorname{tr}_{B_j\cup M_j\cup G\cup H}\left(|\varphi_{j,\mathsf{R}}\rangle\langle\varphi_{j,\mathsf{R}}|\right) \\
\phi_{i,j} &= \operatorname{tr}_{B_i\cup M_i\cup M_j\cup F_j\cup G\cup H}\left(|\varphi_{i,j}\rangle\langle\varphi_{i,j}|\right)
\end{aligned} \tag{23}$$

We can now write out the explicit synthesis for each of these synthesized states as follows:

Recalling Definition 14 we have that the explicit synthesis corresponding to $\phi_{\mathsf{L},i}$ is:

$$\begin{aligned}
S_{\mathsf{L},i} \equiv \Big(&\Gamma_{P_{F_i}^k}\circ\Gamma_{L_i}\circ\Gamma_{B_i\cup M_i\cup F_i}, (F_i\cup G), \\
&(M_i\cup M_i'\cup B_i'\cup F_i^1, ...\cup F_i^K\cup H), \\
&(L_i\cup B_i\cup M_i\cup F_i\cup M_i'\cup B_i'\cup G\cup H\cup F_i^1, ...\cup F_i^K)\Big).
\end{aligned}$$

Symmetrically, the explicit synthesis corresponding to $\phi_{j,\mathsf{R}}$ is:

$$\begin{aligned}
S_{j,\mathsf{R}} \equiv \Big(&\Gamma_{P_{B_j}^k}\circ\Gamma_{R_j}\circ\Gamma_{B_j\cup M_j\cup F_j}, (B_j\cup G), \\
&(M_j\cup M_j'\cup F_j'\cup B_j^1, ...\cup B_j^K\cup H), \\
&R_j\cup B_j\cup M_j\cup F_j\cup M_j'\cup F_j'\cup G\cup H\cup B_j^1, ...\cup B_j^K)\Big).
\end{aligned}$$

Finally, the explicit synthesis for $|\phi_{i,j}\rangle$ can be written as (See Equation 22 for definition of $|\phi_{i,j}\rangle$):

$$\begin{aligned}
S_{i,j} \equiv \Big(&\Gamma_{P_{B_i}^k}\circ\Gamma_{P_{F_j}^k}\circ\Gamma_{i,j}\circ\Gamma_{B_i\cup M_i\cup F_i}\circ\Gamma_{B_j\cup M_j\cup F_j}, \\
&(B_i\cup F_j\cup G), (M_i\cup M_i'\cup F_i'\cup \\
&B_i^1, ...\cup B_i^K\cup M_j\cup M_j'\cup B_j'\cup F_j^1, ...\cup F_j^K\cup H), \\
&(N_{i,j}\cup F_i\cup B_j\cup B_i\cup F_j\cup G\cup M_i\cup M_i'\cup F_i'\cup \\
&B_i^1, ...\cup B_i^K\cup M_j\cup M_j'\cup B_j'\cup F_j^1, ...\cup F_j^K\cup H)\Big),
\end{aligned}$$

where $N_{i,j}$ is defined in the same manner as before: the register containing all of the qubits between K_i and K_j.

Note that we have not explicitly defined $\Gamma_{P^k_{F_i}}$, $\Gamma_{P^k_{B_j}}$, $\Gamma_{P^k_{B_i}}$, and $\Gamma_{P^k_{F_j}}$ here; the explicit definitions can be found in the full version (see Footnote 1). Their definitions are analogous to the circuit $V_{\rho^k_F}$ given in Lemma 8, except that they act on dummy registers instead of the main circuit registers.

Definition 24. Define syntheses

$$\Lambda_1^{j,T} \equiv \Big(\Gamma_{P^T_{B_j}} \circ \Gamma_{B_j \cup M_j \cup F_j}, (B_j \cup F_j), (M_j \cup M'_j \cup F'_j \cup B^1_j, ... \cup B^T_j), B_j \cup M_j \cup F_j \cup M'_j \cup F'_j \cup B^1_j, ... \cup B^T_j)\Big),$$

and

$$Z_j^T \equiv \Big(\Gamma_{P^T_{B_j}}, (B_j), (M'_j \cup F'_j \cup B^1_j, ... \cup B^T_j), (B_j \cup M'_j \cup F'_j \cup B^1_j, ... \cup B^T_j)\Big),$$

Note that these two objects are, in this case, scalars (see Definition 21 to understand why). In fact,

$$Z_j^T = \mathrm{tr}\Big(\rho^T_{B_j}\Big), \quad \text{and} \tag{24}$$

$$\Lambda_1^{j,T} = \mathrm{tr}\Big(\rho^T_{B_j} \left|\psi\right\rangle \left\langle\psi\right|_{B_j \cup M_j \cup F_j} \rho^T_{B_j}\Big) \tag{25}$$

where

$$\rho^K_{B_i} \equiv \left\langle 0^{F_i, M_i, B_i^1, ... B_i^K} \middle| V_{\rho^k_{B_i}} \middle| 0^{F_i, M_i, B_i^1, ... B_i^K} \right\rangle$$

as in Lemma 8.

We write the scalars Z_j^T, $\Lambda_1^{j,T}$ as 2D geometrically-local, shallow depth syntheses to emphasize that they can be computed by any algorithm which computes the probability of zero being output by a 2D geometrically-local, shallow depth synthesis. In the following analysis we will use the 2D algorithm in Theorem 5 of [4] to compute these quantities to inverse polynomial additive error (see Remark 6).

Now we will give a definition of a scalar quantity κ^i_{T,ϵ_2} which is meant to be an approximation for the quantity λ_1^i that we can compute using the "base case" algorithm $\mathcal{B}$ described below. We need this because we want to use λ_1^i to normalize terms in Algorithm 2 below. We will use its approximation, κ^i_{T,ϵ_2}, as a substitute, since it is a quantity that we can compute in quasi-polynomial time (even when $T = \log^c(n)$ and $\epsilon_2 = O(1/n^{\log(n)})$, see Definition 25). The quality of this approximation is the subject of Lemma 26.

Definition 25.

$$\kappa^i_{T,\epsilon_2} \equiv \frac{\mathcal{B}(\Lambda_1^{j,T}, \epsilon_2)}{\mathcal{B}(Z_j^{2T}, \epsilon_2)} = \frac{\mathrm{tr}\Big(\rho^T_{B_j} \left|\psi\right\rangle \left\langle\psi\right|_{B_j \cup M_j \cup F_j} \rho^T_{B_j}\Big) \pm \epsilon_2}{\mathrm{tr}\Big(\rho^{2T}_{B_j}\Big) \pm \epsilon_2}$$

Here the notation $\mathcal{B}(\Lambda_1^{j,T}, \epsilon_2)$ (resp. $\mathcal{B}(Z_j^{2T}, \epsilon_2)$) denotes a use of algorithm $\mathcal{B}$, which we define to be the algorithm from Theorem 5 of [4] (applied according the prescription in Remark 6), to compute the scalar quantity $\Lambda_1^{j,T}$ (resp. Z_j^{2T}) to within additive error ϵ_2. We will elaborate further on this computational task (time complexity, etc) in the analysis of Algorithm 2.

Lemma 26. *If* $\lambda_1^i \geq 1 - e(n)$ *then* $|\kappa^i_{T,\epsilon_2} - \lambda_1^i| \leq O\left(\frac{(e(n))^{2T} + \epsilon_2}{(\lambda_1^i)^{2T+1}}\right)$.

Definition 27. For any natural number Δ, we define $[\Delta] \equiv \{1, ...\Delta\}$. We define $\mathcal{P}([\Delta])$ to be the set of all subsets of $[\Delta]$, that is, the power set of $[\Delta]$. For any set $\sigma \in \mathcal{P}([\Delta])$, we let σ_{max} denote the largest element of σ. We let $|\sigma|$ denote the size of the set σ, and for any $0 < i \leq |\sigma|$ we let $\sigma(i)$ denote the i^{th} smallest element of σ.

IV. Algorithm: Estimating Amplitudes in Quasi-polynomial Time

In this section we define and analyze our algorithm for computing $|\left\langle 0^{\otimes n}\right| C \left|0^{\otimes n}\right\rangle|^2$.

Algorithm 1. $\mathcal{A}_{full}(C, \mathcal{B}, \delta)$: Quasi-Polynomial Time Additive Error Approximation for $|\left\langle 0_{ALL}\right| C \left|0_{ALL}\right\rangle|^2$.

```
1: Input: 3D Geometrically-Local, depth-$d$ circuit $C$, base-case algorithm $\mathcal{B}$, approximation error $\delta$
2: Output: An approximation of $|\langle 0_{ALL}| C |0_{ALL}\rangle|^2$ to within additive error $\delta$.
3: if $\delta \leq 1/n^{\log^2(n)}$ then
4:     return The value $|\langle 0_{ALL}| C |0_{ALL}\rangle|^2$ computed with zero error by a "brute force" $2^{O(n)}$-time algorithm.
5: else if $\delta \geq 1/2$ then
6:     return $1/2$
7: end if
8: Let $N$ be the register containing all of the qubits on which $C$ acts. Since these qubits are arranged in a cubic lattice, one of the sides of the cube $N$ must have length at most $n^{\frac{1}{3}}$. We will call the length of this side the "width" and will now describe how to "cut" the cube $N$, and the circuit $C$, perpendicular to this particular side.
9: Select $\frac{1}{10d}n^{\frac{1}{3}}$ light-cone separated slices $K_i$ of $10d$ width in $N$, with at most $10d$ distance between adjacent slices. Let $h(n) = \log^7(n)$. Use the base case algorithm $\mathcal{B}$ to check if at least $\frac{1}{10d}n^{\frac{1}{3}} - h(n)$ of the slices obey:
       $$\left|\mathrm{tr}\left(\langle 0_{M_i}| C |0_{ALL}\rangle \langle 0_{ALL}| C^\dagger |0_{M_i}\rangle\right)\right| \geq 2^{\frac{\log(\delta)}{h(n)}}.$$
10: OR, there are fewer than $\frac{1}{10d}n^{\frac{1}{3}} - h(n)$ slices that obey:
       $$\left|\mathrm{tr}\left(\langle 0_{M_i}| C |0_{ALL}\rangle \langle 0_{ALL}| C^\dagger |0_{M_i}\rangle\right)\right| \geq 2^{\frac{\log(\delta)}{h(n)}}.$$
```

```
11: if Fewer than $\frac{1}{10d}n^{\frac{1}{3}} - h(n)$ slices obey Line 26 then
12:     return 0
13: else if At least $\frac{1}{10d}n^{\frac{1}{3}} - h(n)$ slices obey Line 26 then
14:     We will denote the set of these slices by $K_{heavy}$. Note
        that the maximum amount of width between any two
        adjacent slices in $K_{heavy}$ is $10d \cdot h(n)$. Furthermore,
        the maximum amount of width collectively between
        $\Delta$ slices in $K_{heavy}$ is $10d\Delta + 10d \cdot h(n)$. Now that the
        set $K_{heavy}$ has been defined, we will use this fixed
        set in the recursive algorithm, Algorithm 2.
15:     Define the geometrically-local, depth-$d$ synthesis $S \equiv$
        $(C, L, M, N)$, where $L = M = \emptyset$, are empty
        registers, and $N$ is the entire input register for the
        circuit $C$.
16:     return $\mathcal{A}(S, \eta = \frac{\log(n)}{3\log(4/3)}, \Delta = \log(n), \epsilon =$
        $\delta 2^{-10\log(n)\log(\log(n)))}, h(n) = \log^7(n), K_{heavy}, \mathcal{B})$
17: end if
```

Algorithm 2. $\mathcal{A}(S, \eta, \Delta, \epsilon, h(n), K_{heavy}, \mathcal{B})$: Recursive Divide-and-Conquer Subroutine for Algorithm 1.

```
1: Input: 3D Geometrically-Local, depth-$d$ synthesis $S$,
   number of iterations $\eta$, number of cuts $\Delta$, positive base-
   case error bound $\epsilon > 0$, base-case algorithm $\mathcal{B}$, a set of
   heavy slices $K_{heavy}$
2: Output: An approximation of the quantity $\langle 0_N | \phi_S | 0_N \rangle$
   where $\phi_S$ is the un-normalized mixed state specified
   by the 3D geometrically-local, depth-$d$ synthesis $S$, and
   $|0_N\rangle$ is the 0 state on the entire $N$ register of that
   synthesis. The approximation error is bounded in the
   analysis below.
3: Given the geometrically-local, depth-$d$ synthesis $S =$
   $(\Gamma, L, M, N)$, let us ignore the registers $L$ and $M$ as
   they have already been measured or traced-out.
4: Let $\ell$ be the width of the $N$ register of the synthesis $S$.
   Define the stopping width $w_0 \equiv 20d(\Delta + h(n) + 2)$.
5: if $\ell < w_0 = 20d(\Delta + h(n) + 2)$ OR $\eta < 1$ then
6:     Use the base-case algorithm $\mathcal{B}$ to compute the quan-
       tity $\langle 0_N | \phi_S | 0_N \rangle$ to within error $\epsilon$.
7:     return $\mathcal{B}(S, \epsilon)$
8: else
9:     We will "slice" the 3D geometrically-local, depth-$d$
       synthesis $S$ in $\Delta$ different locations, as follows:
       Since $N$ is 3D we define a region $Z \subset N$ to be the
       sub-cube of $N$ which has width $10d(\Delta + h(n) + 2)$,
       and is centered at the halfway point of $N$ width-
       wise (about the point $\ell/2$ of the way across $N$).
       Since the maximum amount of width collectively
       between $\Delta$ slices in $K_{heavy}$ is $10d\Delta + 10d \cdot h(n)$ (see
       Algorithm 1), we are guaranteed that the region $Z$
       will contain at least $\Delta$ slices, $K_1, K_2, \dots, K_\Delta$, from
       $K_{heavy}$. For any two slices $K_i, K_j \in K_{heavy}$, let the
       un-normalized states $|\varphi_{\mathsf{L},i}\rangle, |\varphi_{i,j}\rangle, |\varphi_{j,\mathsf{R}}\rangle$, and corre-
       sponding sub-syntheses $S_{\mathsf{L},i}, S_{i,j}, S_{j,\mathsf{R}}$ be as defined
       in Definition 23, with $K = \log^3(n)$. We will use these
       to describe the result of our division step below.
       For each $K_i \in K_{heavy}$ pre-compute the quan-
       tity $\kappa^i_{T,\epsilon_2}$, with $T = \log^3(n)$, and $\epsilon_2 =$
       $\delta 2^{-10\log(n)\log(\log(n)))}$.
10:    return
       $$\sum_{i=1}^{\Delta} \mathcal{A}^{(1)}_i - \sum_{i=1}^{\Delta}\sum_{j=i+1}^{\Delta} \mathcal{A}^{(2)}_{i,j} + \sum_{i=1}^{\Delta}\sum_{j=i+2}^{\Delta} \mathcal{A}^{(3)}_{i,j} \quad (26)$$
       The quantities $\mathcal{A}^{(1)}_i, \mathcal{A}^{(2)}_{i,j}$, and $\mathcal{A}^{(3)}_{i,j}$ are defined below.
11: end if
```

The return quantities are defined as:

$$\mathcal{A}^{(1)}_i \equiv \frac{1}{(\kappa^i_{T,\epsilon_2})^{4K+1}} \mathcal{A}(S_{L,i}, \eta - 1) \cdot \mathcal{A}(S_{i,R}, \eta - 1), \tag{27}$$

$$\mathcal{A}^{(2)}_{i,j} \equiv \frac{1}{(\kappa^i_{T,\epsilon_2}\kappa^j_{T,\epsilon_2})^{4K+1}} \mathcal{A}(S_{L,i}, \eta - 1) \cdot \mathcal{B}(S_{i,j}, \epsilon) \cdot \mathcal{A}(S_{j,R}, \eta - 1), \tag{28}$$

and

$$\mathcal{A}^{(3)}_{i,j} \equiv \frac{1}{(\kappa^i_{T,\epsilon_2}\kappa^j_{T,\epsilon_2})^{4K+1}} \mathcal{A}(S_{L,i}, \eta - 1) \cdot \mathcal{A}(S_{j,R}, \eta - 1) \cdot \left[\sum_{\sigma \in \mathcal{P}(\{i+1,\dots,j-1\})\setminus\emptyset} (-1)^{|\sigma|+1} \mathcal{B}\Big(\left(\otimes_{k\in\sigma} \Pi^K_{F_k} \langle 0_{M_k}|\right) \phi_{i,j} \left(\otimes_{k\in\sigma} |0_{M_k}\rangle \Pi^K_{F_k}\right), \frac{\epsilon}{2^\Delta} \Big) \right]. \tag{29}$$

Note: $\mathcal{B}\left(\left(\otimes_{k\in\sigma}\Pi^K_{F_k} \langle 0_{M_k}|\right) \phi_{i,j} \left(\otimes_{k\in\sigma} |0_{M_k}\rangle \Pi^K_{F_k}\right), \frac{\epsilon}{2^\Delta}\right)$ denotes an $\frac{\epsilon}{2^\Delta}$ approximation of the quantity $\left(\langle 0_{ALL}| \otimes_{k\in\sigma} \Pi^K_{F_k}\right) \phi_{i,j} \left(\otimes_{k\in\sigma}\Pi^K_{F_k} |0_{ALL}\rangle\right)$ obtained via base case Algorithm $\mathcal{B}$. Note that for brevity it is implied that $\mathcal{A}(S, \eta) = \mathcal{A}(S, \eta, \Delta, \epsilon, h(n), K_{heavy}, \mathcal{B})$.

A. *Run-Time and Error Analysis for Algorithm 1*

Theorem 28. *Let C be any depth-d, 3D geometrically local quantum circuit on n qubits. Algorithm 1, $\mathcal{A}_{full}(C, \mathcal{B}, \delta)$, where $\mathcal{B}$ is the base case algorithm specified in Theorem 5 of [4], will produce the scalar quantity $|\langle 0^{\otimes n}| C |0^{\otimes n}\rangle|^2$ to within δ error in time*

$$T(n) = \delta^{-2} 2^{d^3 \mathsf{polylog}(n) (1/\delta)^{1/\log^2(n)}} \tag{30}$$

Proof outline of Theorem 28: The proof proceeds in two parts, the first bounding the approximation error obtained by the algorithm, and the second bounding the runtime. We will only give a brief summary here. A complete proof is given in Section 5 of the full version of the paper (referenced in Footnote 1). See Parameter Table in the full version for a brief summary of the parameters used throughout Algorithms 1 and 2.

The analysis of the approximation error and runtime obtained by $\mathcal{A}_{full}(C, \mathcal{B}, \delta)$ can be broken into four cases according to the IF statements on Lines 3, 5, 11, and 13 of Algorithm 1. The first three cases are easily verified to return δ approximations of $|\langle 0_{ALL}| C |0_{ALL}\rangle|^2$ within the desired runtime.

The fourth case occurs when the IF statement on Line 11 of Algorithm 1 is not satisfied, so that the IF statement of Line 13 must be satisfied, by definition. In that case Algorithm 1 returns the quantity:

$$\mathcal{A}(S, \eta, \Delta, \epsilon, h(n), K_{heavy}, \mathcal{B})$$

where $\eta = \frac{\log(n)}{3\log(4/3)}, \Delta = \log(n), \epsilon = \delta \cdot 2^{-10\log(n)\log(\log(n))}$, and $h(n) = log^7(n)$. By definition, this quantity is an $f(S, \eta, \Delta, \epsilon)$-additive error approximation of $|\langle 0_{ALL}| C |0_{ALL}\rangle|^2$. Through standard asymptotic analysis combined with Lemma 32 we show

$$\begin{aligned} f(S, \eta, \Delta, \epsilon) &\le \eta 20^\eta \Delta^{2\eta} \Big(E_3(n, K, T, \epsilon_2, \epsilon, \Delta) \\ &\quad + (2e(n) + 2g(n))^\Delta \Big) \\ &\le o(1) \cdot \delta, \end{aligned} \tag{31}$$

proving that $\mathcal{A}(S, \eta, \Delta, \epsilon, h(n), K_{heavy}, \mathcal{B})$ is a δ-approximation of $|\langle 0_{ALL}| C |0_{ALL}\rangle|^2$, as desired. This computation takes $T(n) < \delta^{-2} 2^{d^3 \mathsf{polylog}(n)}$ time to compute, which follows directly from the runtime bound on Algorithm 2, which is given in Theorem 33 of Subsection IV-C.

■

B. Error Analysis for Algorithm 2

The main result of this subsection is Theorem 32, the error bound on the recursive divide-and-conquer Algorithm 2. We will briefly detail the methods here. The output of Algorithm 2 is the scalar quantity:

$$\mathcal{A}(S, \eta) \equiv \sum_{i=1}^{\Delta} \mathcal{A}_i^{(1)} - \sum_{i=1}^{\Delta} \sum_{j=i+1}^{\Delta} \mathcal{A}_{i,j}^{(2)} + \sum_{i=1}^{\Delta} \sum_{j=i+2}^{\Delta} \mathcal{A}_{i,j}^{(3)} \tag{32}$$

where $\mathcal{A}_i^{(1)}, \mathcal{A}_{i,j}^{(2)}$, and $\mathcal{A}_{i,j}^{(3)}$ are defined in Equations 27, 28, and 29, respectively.

Note that the terms $\mathcal{A}_i^{(1)}, \mathcal{A}_{i,j}^{(2)}$, and $\mathcal{A}_{i,j}^{(3)}$ correspond to divisions at one cut, exactly two cuts, and strictly more than two cuts, respectively. As shown in Figure 4, these terms are approximations of the quantities:

$$\left\langle 0_{ALL} \middle| \Psi_{\{i\}} \right\rangle \left\langle \Psi_{\{i\}} \middle| 0_{ALL} \right\rangle \tag{33}$$

$$\left\langle 0_{ALL} \middle| \Psi_{\{i,j\}} \right\rangle \left\langle \Psi_{\{i,j\}} \middle| 0_{ALL} \right\rangle \tag{34}$$

$$\left\langle 0_{ALL} \middle| \Psi_{\{i,j\}\cup\sigma} \right\rangle \left\langle \Psi_{\{i,j\}\cup\sigma} \middle| 0_{ALL} \right\rangle, \tag{35}$$

respectively.

We know, by Lemma 18 that,

$$\begin{aligned} &\left\| \sum_{\sigma \in \mathcal{P}([\Delta])} (-1)^{|\sigma|} |\Psi_\sigma\rangle \langle \Psi_\sigma| \right\| \\ &= \left\| |\Psi_\emptyset\rangle \langle \Psi_\emptyset| - \sum_{\sigma \in \mathcal{P}([\Delta]) \setminus \emptyset} (-1)^{|\sigma|+1} |\Psi_\sigma\rangle \langle \Psi_\sigma| \right\| \\ &\le (2e(n) + 2g(n))^\Delta \end{aligned} \tag{36}$$

In other words, the summation

$$\sum_{\sigma \in \mathcal{P}([\Delta]) \setminus \emptyset} (-1)^{|\sigma|+1} \langle 0_{ALL}|\Psi_\sigma\rangle \langle \Psi_\sigma|0_{ALL}\rangle$$

is a $(2e(n)+2g(n))^\Delta$ approximation of our desired quantity

$$\langle 0_{ALL}|\Psi_\emptyset\rangle \langle \Psi_\emptyset|0_{ALL}\rangle = |\langle 0_{ALL}| C |0_{ALL}\rangle|^2.$$

Combining this along with the triangle inequality, we see that

$$\begin{aligned} &f(S, \eta, \Delta, \epsilon) \le \left\| \langle 0_{ALL}|\Psi_\emptyset\rangle \langle \Psi_\emptyset|0_{ALL}\rangle - \mathcal{A} \right\| \\ &\le (2e(n) + 2g(n))^\Delta \\ &+ \sum_{i=1}^{\Delta} \left\| \mathcal{A}_i^{(1)} - \left\langle 0_{ALL} \middle| \Psi_{\{i\}} \right\rangle \left\langle \Psi_{\{i\}} \middle| 0_{ALL} \right\rangle \right\| \\ &+ \sum_{i=1}^{\Delta} \sum_{j=i+1}^{\Delta} \left\| \mathcal{A}_{i,j}^{(2)} - \left\langle 0_{ALL} \middle| \Psi_{\{i,j\}} \right\rangle \left\langle \Psi_{\{i,j\}} \middle| 0_{ALL} \right\rangle \right\| \\ &+ \sum_{i=1}^{\Delta} \sum_{j=i+2}^{\Delta} \left\| \mathcal{A}_{i,j}^{(3)} - \left\langle 0_{ALL} \middle| \Psi_{\{i,j\}\cup\sigma} \right\rangle \left\langle \Psi_{\{i,j\}\cup\sigma} \middle| 0_{ALL} \right\rangle \right\|. \end{aligned} \tag{37}$$

The first term represents the error in approximating $|\langle 0_{ALL}| C |0_{ALL}\rangle|^2$ by summing over terms like in Equations 33, 34, 35 (the content of Lemma 18). The second, third, and fourth terms represent the total error in approximating one cut, two cuts, and more than two cuts, respectively, each via recursive calls to Algorithm 2.

We can bound the remaining three quantities as follows:

Lemma 29.

$$\begin{aligned} &\left\| \mathcal{A}_i^{(1)} - \left\langle 0_{ALL} \middle| \Psi_{\{i\}} \right\rangle \left\langle \Psi_{\{i\}} \middle| 0_{ALL} \right\rangle \right\| \\ &\le E_1(n, K, T, \epsilon_2) + 2f(S, \eta - 1, \Delta, \epsilon), \end{aligned}$$

where $E_1(n, K, T, \epsilon_2) \equiv 10K(e(n)^{2T} + 6g(n) + \epsilon_2)$.

Lemma 30.

$$\begin{aligned} &\left\| \mathcal{A}_{i,j}^{(2)} - \left\langle 0_{ALL} \middle| \Psi_{\{i,j\}} \right\rangle \left\langle \Psi_{\{i,j\}} \middle| 0_{ALL} \right\rangle \right\| \\ &\le E_2(n, K, T, \epsilon_2, \epsilon) + 2f(S, \eta - 1, \Delta, \epsilon), \end{aligned} \tag{38}$$

where $E_2(n, K, T, \epsilon_2, \epsilon) \equiv 10K(e(n)^{2T} + 6g(n) + \epsilon_2) + \epsilon$

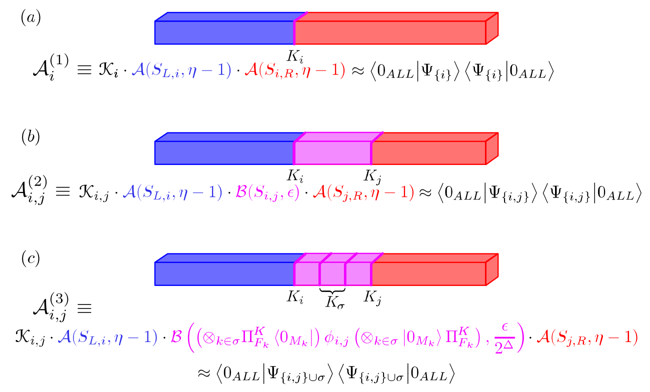
(a)
K_i
$\mathcal{A}_i^{(1)} \equiv \mathcal{K}_i \cdot \mathcal{A}(S_{L,i}, \eta - 1) \cdot \mathcal{A}(S_{i,R}, \eta - 1) \approx \left\langle 0_{ALL} \middle| \Psi_{\{i\}} \right\rangle \left\langle \Psi_{\{i\}} \middle| 0_{ALL} \right\rangle$
(b)
K_i
K_j
$\mathcal{A}_{i,j}^{(2)} \equiv \mathcal{K}_{i,j} \cdot \mathcal{A}(S_{L,i}, \eta - 1) \cdot \mathcal{B}(S_{i,j}, \epsilon) \cdot \mathcal{A}(S_{j,R}, \eta - 1) \approx \left\langle 0_{ALL} \middle| \Psi_{\{i,j\}} \right\rangle \left\langle \Psi_{\{i,j\}} \middle| 0_{ALL} \right\rangle$
(c)
K_i
K_σ
K_j
$\mathcal{A}_{i,j}^{(3)} \equiv$
$\mathcal{K}_{i,j} \cdot \mathcal{A}(S_{L,i}, \eta - 1) \cdot \mathcal{B}\left(\left(\otimes_{k\in\sigma} \Pi_{F_k}^K \left\langle 0_{M_k} \right| \right) \phi_{i,j} \left(\otimes_{k\in\sigma} \left| 0_{M_k} \right\rangle \Pi_{F_k}^K \right), \frac{\epsilon}{2^\Delta} \right) \cdot \mathcal{A}(S_{j,R}, \eta - 1)$
$\approx \left\langle 0_{ALL} \middle| \Psi_{\{i,j\}\cup\sigma} \right\rangle \left\langle \Psi_{\{i,j\}\cup\sigma} \middle| 0_{ALL} \right\rangle$

Figure 4. Depiction of the types of terms that appear in Equation 32: (a) those involving exactly one cut, $\{i\}$, (b) those involving exactly two cuts, $\{i,j\}$, and (c) those involved three or more of the Δ cuts, $\{i,j\} \cup \sigma$, for $\sigma \in \mathcal{P}(\{i+1, \ldots, j-1\}) \setminus \emptyset$. For brevity we have denoted $\mathcal{K}_i \equiv \frac{1}{(\kappa_{T,\epsilon_2}^i)^{4K+1}}$ and $\mathcal{K}_{i,j} \equiv \frac{1}{(\kappa_{T,\epsilon_2}^i \kappa_{T,\epsilon_2}^j)^{4K+1}}$.

Lemma 31.

$$\left\| \mathcal{A}_{i,j}^{(3)} - \left\langle 0_{ALL} \middle| \Psi_{\{i,j\}\cup\sigma} \right\rangle \left\langle \Psi_{\{i,j\}\cup\sigma} \middle| 0_{ALL} \right\rangle \right\| \le E_3(n, K, T, \epsilon_2, \epsilon, \Delta) + 16 f(S, \eta - 1, \Delta, \epsilon), \tag{39}$$

where

$$E_3(n, K, T, \epsilon_2, \epsilon, \Delta) \equiv O\left(2^\Delta (6g(n)) + 2^\Delta K \left(e(n)^{2T} + \epsilon_2\right) + \epsilon\right).$$

The first two Lemmas are a direct result of Lemma 20 and Lemma 26. The third Lemma is a result of Lemma 18, Lemma 20, and Lemma 26 and its proof can be found in Appendix A of the full text.

These three results combined with Equation 37 allow us to prove:

Lemma 32. *The error function $f(S, \eta, \Delta, \epsilon)$ obeys the following bound:*

$$f(S, \eta, \Delta, \epsilon) \le \eta 20^\eta \Delta^{2\eta} \Big(E_3(n, K, T, \epsilon_2, \epsilon\Delta) + (2e(n) + 2g(n))^\Delta \Big) \tag{40}$$

C. Run-Time Analysis for Algorithm 2

The main result of this subsection is Theorem 33. We will only briefly summarize the main ideas here. Recall that ℓ is defined, in Algorithm 2, to be the width of the N register for our input synthesis S. We use $T(\ell)$ to denote the run-time bound for our algorithm on a synthesis with an N register of width ℓ.

Theorem 33. *Suppose $\eta = \frac{\log(n)}{3\log(4/3)}$ and $\Delta = \log n$. Given these values, the run-time for Algorithm 2 will be bounded by*

$$T(n) < \delta^{-2} 2^{d^3 \mathsf{polylog}(n)} \tag{41}$$

The main time costs for Algorithm 2 come from the **return** line of the algorithm, which makes recursive calls to Algorithm 2 on a variety of smaller subproblems, as well as calls to the base case algorithm $\mathcal{B}$, and computing the κ_{T,ϵ_2}^i quantities. All of the steps performed in Algorithm 2 before the **return** line (selecting the region Z, etc), can easily be done in (lesser) polynomial time.

*Recursive calls in the **return** line of Algorithm 2:* The **return** line of Algorithm 2 makes 2Δ distinct recursive calls to itself, which each only need to be computed once, even though they are reused multiple times in Equations 27, 28, and 29. Furthermore, the width of the active register of the synthesis input to each of the recursive calls to Algorithm 2 is at most $\frac{\ell+|Z|}{2} \le \frac{3}{4}\ell$ (where the inequality follows because, in the context of Algorithm 2, the relationship $|Z| \le \frac{\ell}{2}$ is enforced by lines 5 and 9). Thus, the runtime accrued by recursive calls will be

$$2\Delta T\left(\frac{3}{4}\ell\right). \tag{42}$$

*Uses of $\mathcal{B}$ in the **return** line of Algorithm 2:* There are at most $2\Delta^2$ calls to the base case algorithm $\mathcal{B}$ in the **return** line of Algorithm 2, which all occur in Equations 28 and 29 of the **return** line. The terms in Equation 28 have the form $\mathcal{B}(S_{i,j}, \epsilon)$, and the terms in Equation 29 have the form $\mathcal{B}\left(\left(\otimes_{k\in\sigma} \Pi_{F_k}^K \left\langle 0_{M_k} \right| \right) \phi_{i,j} \left(\otimes_{k\in\sigma} \left| 0_{M_k} \right\rangle \Pi_{F_k}^K \right), \frac{\epsilon}{2^\Delta} \right)$. By Remark 6 we know that, for any 3D geometrically-local, depth-D synthesis S (on n qubits, total) we can use the algorithm from Theorem 5 of [4] to compute the quantity $\mathcal{B}(S, \epsilon) = |\left\langle 0_{ALL} \right| \phi_S \left| 0_{ALL} \right\rangle|^2 \pm \epsilon$ in time $n\epsilon^{-2} 2^{O(D^2 \cdot w)}$, where w is the width of the "active register", N, of the synthesis S in the third dimension (see Definition 21 for the definitions of syntheses S, and the register N). Using Remark 6 in this way, we see that the quantities $\mathcal{B}(S_{i,j}, \epsilon) = |\left\langle 0_{ALL} \right| \phi_{i,j} \left| 0_{ALL} \right\rangle|^2 \pm \epsilon$ from Equation 28 can each be computed in time $n\epsilon^{-2} 2^{O((dK^2)^2 \cdot 10d(\Delta + h(n) + 2))}$, because the synthesis $S_{i,j}$ has depth at most $O(dK^2)$. Combining these results along with our particular parameter choices leads to a total accrued runtime of

$$\delta^{-2} 2^{d^3 \mathsf{polylog}(n)}. \tag{43}$$

*Computation of quantities κ_{T,ϵ_2}^i in the **return** line of Algorithm 2:* The task of computing the κ_{T,ϵ_2}^i values in the **return** line of Algorithm 2 requires using the base case algorithm $\mathcal{B}$ as specified and discussed in Definition 25 (via the usual prescription in Remark 6). Following a similar analysis to that of the previous calls to $\mathcal{B}$, we see this computation has a total runtime of

$$\delta^{-2} 2^{d^3 \mathsf{polylog}(n)}. \tag{44}$$

Combining these runtime results, we have total runtime:

$$T(\ell) <= 2\Delta T\left(\frac{3}{4}\ell\right) + \delta^{-2}2^{d^3\mathsf{polylog}(n)}. \tag{45}$$

Note that Equation 45 is a recursive run-time whereby, at each level, we have at most 2Δ subproblems, each with size at most $\frac{3}{4}$ of the original problem. This is a common formula, and we can use the Master Theorem for divide-and-conquer algorithms to determine an upper bound for our run-time. We can ultimately determine an upper bound:

$$T(n^{1/3}) < (2\Delta)^{\eta}\left[T\left(\left(\frac{3}{4}\right)^{\eta} n^{\frac{1}{3}}\right) + 2\Delta\delta^{-2}2^{d^3\mathsf{polylog}(n)}\right]. \tag{46}$$

where η is the depth of our recursive calls. Note that the reason we start our recursion at $n^{1/3}$ instead of at n is because of the technical definition of $T(\ell)$. Recall that, we use $T(\ell)$ to denote the run-time bound for our algorithm on a synthesis with an N register of width ℓ. The starting point of our recursion is a cube of n qubits, which has side length $N^{1/3}$ in each dimension, and this is the reason that the total runtime for the original problem is bounded by $T(n^{1/3})$.

The proof of Theorem 33 is immediate from this result and our choice of parameters in Algorithm 2.

Acknowledgment

MC thanks David Gosset and Sergey Bravyi for helpful discussions. MC thanks students Samuel Tan, Suchetan Dontha, and Stephen Smith for correcting an error in the original writing of Lemma 8 (the rest of the paper is unchanged by this correction). Part of this work was completed while MC was attending the Simons Institute Quantum Wave in Computing workshop.

References

[1] R. Movassagh, "Quantum supremacy and random circuits," QIP, 2020. [Online]. Available: https://arxiv.org/pdf/1909.06210.pdf

[2] Y. Kondo, R. Mori, and R. Movassagh, "Fine-grained analysis and improved robustness of quantum supremacy for random circuit sampling," 2021.

[3] B. M. Terhal and D. P. DiVincenzo, "Adaptive quantum computation, constant depth quantum circuits and arthur-merlin games," *Quantum Inf. Comput.*, vol. 4, no. 2, pp. 134–145, 2004. [Online]. Available: https://doi.org/10.26421/QIC4.2-5

[4] S. Bravyi, D. Gosset, and R. Movassagh, "Classical algorithms for quantum mean values," QIP, 2020. [Online]. Available: https://arxiv.org/abs/1909.11485

[5] A. Gilyén, Y. Su, G. H. Low, and N. Wiebe, "Quantum singular value transformation and beyond: exponential improvements for quantum matrix arithmetics," STOC, 2019. [Online]. Available: https://arxiv.org/pdf/1806.01838.pdf

[6] S. Aaronson and A. Arkhipov, "The computational complexity of linear optics," in *Proceedings of the Forty-Third Annual ACM Symposium on Theory of Computing*, ser. STOC '11. New York, NY, USA: Association for Computing Machinery, 2011, p. 333–342. [Online]. Available: https://doi.org/10.1145/1993636.1993682

[7] M. J. Bremner, R. Jozsa, and D. J. Shepherd, "Classical simulation of commuting quantum computations implies collapse of the polynomial hierarchy," *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 467, no. 2126, p. 459–472, Feb 2011.

[8] M. J. Bremner, A. Montanaro, and D. J. Shepherd, "Achieving quantum supremacy with sparse and noisy commuting quantum computations," *Quantum*, vol. 1, p. 8, Apr 2017. [Online]. Available: http://dx.doi.org/10.22331/q-2017-04-25-8

[9] A. Neville, C. Sparrow, R. Clifford, E. Johnston, P. M. Birchall, A. Montanaro, and A. Laing, "Classical boson sampling algorithms with superior performance to near-term experiments," *Nature Physics*, vol. 13, no. 12, p. 1153–1157, Dec 2017.

[10] A. Bouland, B. Fefferman, C. Nirkhe, and U. Vazirani, "On the complexity and verification of quantum random circuit sampling," *Nature Physics*, vol. 15, no. 2, p. 159–163, Feb 2019.

[11] J. Bermejo-Vega, D. Hangleiter, M. Schwarz, R. Raussendorf, and J. Eisert, "Architectures for quantum simulation showing a quantum speedup," *Phys. Rev. X*, vol. 8, p. 021010, Apr 2018. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevX.8.021010

[12] A. M. Dalzell, A. W. Harrow, D. E. Koh, and R. L. La Placa, "How many qubits are needed for quantum computational supremacy?" *Quantum*, vol. 4, p. 264, May 2020. [Online]. Available: http://dx.doi.org/10.22331/q-2020-05-11-264

[13] C. Huang, F. Zhang, M. Newman, J. Cai, X. Gao, Z. Tian, J. Wu, H. Xu, H. Yu, B. Yuan, M. Szegedy, Y. Shi, and J. Chen, "Classical simulation of quantum supremacy circuits," 2020.

[14] J. Napp, R. L. L. Placa, A. M. Dalzell, F. G. S. L. Brandao, and A. W. Harrow, "Efficient classical simulation of random shallow 2d quantum circuits," 2020.