2022 IEEE International Conference on Quantum Computing and Engineering (QCE) | 978-1-6654-9113-6/22/$31.00 ©2022 IEEE | DOI: 10.1109/QCE53715.2022.00073

Approximate Quantum Circuit Reconstruction

Daniel Chen*, Betis Baheri†, Vipin Chaudhary*, Qiang Guan†, Ning Xie‡, Shuai Xu*
* Department of Computer and Data Science
Case Western Reserve University, Cleveland, Ohio 33549
Email: {txc461, vxc204, sxx214}@case.edu
† Department of Computer Science
Kent State University, Kent, Ohio 44240
Email: {bbaheri, qguan}@kent.edu
‡ Knight Foundation School of Computing and Information Sciences
Florida International University, Miami, FL 33199
Email: nxie@cis.fiu.edu

***Abstract*—Current and imminent quantum hardware lacks reliability and applicability due to noise and limited qubit counts. Quantum circuit cutting — a technique dividing large quantum circuits into smaller subcircuits with sizes appropriate for the limited quantum resource at hand — is used to mitigate these problems. However, classical postprocessing involved in circuit cutting generally grows exponentially with the number of cuts and quantum counts. This article introduces the notion of approximate circuit reconstruction. Using a sampling-based method like Markov Chain Monte Carlo (MCMC), we probabilistically select bit strings of high probability upon reconstruction. This avoids excessive calculations when reconstructing the full probability distribution. Our results show that such a sampling-based postprocessing method holds great potential for fast and reliable circuit reconstruction in the NISQ era and beyond.**

I. Introduction

Quantum computers are believed to enable polynomial or even exponential speed-up over classical computers for many classes of problems [1]. However, current quantum computing hardware, also known as NISQ computers, is characterized by its lack of scalability and reliability [2]. To effectively handle noise, the quantum algorithm community has leveraged classical resources to aid computation, which led to the emergence of quantum-classical hybrid algorithms [3]. In particular, variational quantum circuits has so far received great attention [4], with well-known examples like quantum approximate optimization algorithm (QAOA) [5] and variational quantum eigensolver (VQE) [6] [7]. Such variational circuits perform expensive calculations (such as solving NP classical optimization or simulating Hamiltonians) using a quantum computer and update the circuit parameters using classical optimizers like COBYLA [8] or gradient descent [9].

Hybrid algorithms mentioned above have enjoyed some success in the NISQ era, showing their robustness against noisy computation [10]. At the same time, escaping the issue of vanishing gradients, or barren plateaus, induced by noise, remains a problem [11]. Also, the size of NISQ computers poses barriers for most real applications. Thus, Peng *et al.* [12] developed a method for dividing any quantum circuit into multiple smaller subcircuits that can be executed independently. Then, one could use the information gathered from each subcircuit to reproduce the theoretical outcome from running the full, uncut circuit. In addition to making evaluation of large circuits possible, it is also shown that evaluating smaller circuits improves fidelity while being more robust to noise [13] [14] [15].

Despite the theoretical success, reconstructing the probability distribution of the full circuit is time-consuming. More specifically, the classical post-processing procedures generally scale exponentially with respect to the size of the circuits. If one desires the classical probability distribution of the full circuit, there will be an additional exponential factor with respect to the circuit size. One way to avoid this scaling is to design algorithms that reduce the number of cuts needed [16]. However, we take a more direct approach in attempting to lower the reconstruction time complexity. Similar to how probabilistic methods are used to provide approximate solutions to problems that are difficult to solve exactly, we propose the concept of an *approximate circuit reconstruction*. Instead of exploring exponentially growing state-space via brute force, we probabilistically explore the space of bit-strings and sample the points that are of higher likelihood. This procedure is asymptotically correct as the number of samples taken increases. We also found a significant reduction in run time, shedding light on the potential utility of applying this technique for larger, more practical problems.

The contribution of this work can be summarized as follows:

- We introduce the notion of *approximate circuit reconstruction* using Monte Carlo methods for reconstructing measurement outcomes of subcircuits

978-1-6654-9113-6/22/$31.00 ©2022 IEEE
DOI 10.1109/QCE53715.2022.00073

Authorized licensed use limited to: Kent State University Libraries. Downloaded on June 15,2023 at 20:54:54 UTC from IEEE Xplore. Restrictions apply.

after cutting.

- Our model puts no explicit restriction on the run time: one can decide the trade-off between speed and accuracy depending on one's resources.
- The naive implementation was able to reconstruct the circuit outcome at speed an order of magnitude faster without losing much accuracy, motivating further development and optimization of this method.

II. Quantum Circuit Cutting

In short, the idea of quantum circuit cutting is separating circuits by applying a set of measurements and state preparation such that, when combined, results in a trivial action. Consider the Pauli matrices.

$$\sigma_x = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}, \quad \sigma_y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \quad \sigma_z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \tag{1}$$

Together with $I = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$, these four matrices form an orthonormal basis over the set of 2×2 complex matrices. We can apply a combination of measurements and initializations in different Pauli bases such that the state itself is unchanged. For the formalism presented in this paper, readers should refer to [14].

We will begin by considering the case that there is one cut. Suppose there is an m-qubit quantum circuit C, which can be cut into subcircuits A, B (c.f. Figure 1). Subcircuit A has n qubits whose measurement outcome in the computational basis will be represented as $\hat{b}_1 \dots \hat{b}_{n-1}\hat{a}$. We can compactly store the set of outcomes of subcircuit A with a rank-3 tensor $p_A^{(\hat{b}_1 \dots \hat{b}_{n-1}, \hat{a}, \beta)}$. The first index denotes the measurement outcome of the first $n-1$ qubits. The second index refers to the extra "connection" bit $\hat{a} \in \{0, 1\}$. There is always this extra bit upstream of the cut that will be measured in different bases. It naturally exists as part of the subcircuit and does not directly correspond to the circuit output. The third index refers to the measurement basis, $\beta \in \mathcal{B} = \{\sigma_x, \sigma_y, \sigma_z\}$.

Similarly, subcircuit B is a $m - n + 1$ qubit circuit with measurement outcomes $\hat{b}_n \dots \hat{b}_m$. This can again be written compactly as a rank-3 tensor $p_B^{(\hat{b}_n \dots \hat{b}_m, e, \beta)}$. The first index represents the measurement outcome of subcircuit B. The second and third indices are related: e denotes the eigenbasis, with respect to $\beta \in \mathcal{B}$, that the qubit immediately downstream of the cut is initialized to. For example, if $e = 0$ and $\beta = \sigma_x$, then the qubit is initialized to the $|+\rangle$ state (the eigenvector of β corresponding to eigenvalue 1).

The probability distribution P corresponding to the outcome can simply be written as a vector of 2^m bins, each corresponding to a measurement outcome. Finding the probability of an outcome $\hat{b}_1 \hat{b}_2 \dots \hat{b}_m$ from its subcircuits is represented in the following equation, which one can think of as a tensor contraction over indices $\hat{a}, e, \beta$.

$$P^{(\hat{b}_1 \dots \hat{b}_{n-1}, \hat{b}_n \dots \hat{b}_m)} = \sum_{\hat{a}, e, \beta} \gamma^{(\hat{a}, e, \beta)} p_A^{(\hat{b}_1 \dots \hat{b}_{n-1}, \hat{a}, \beta)} p_B^{(\hat{b}_n \dots \hat{b}_m, e, \beta)} \tag{2}$$

and the rank-3 tensor γ is defined as follows.

$$\gamma^{(\hat{a}, e, \beta)} = \begin{cases} 2\delta_{\hat{a}, e} - 1 & \text{if } \beta = \sigma_x \text{ or } \sigma_y \\ 2\delta_{\hat{a}, e} & \text{if } \beta = \sigma_z \end{cases} \tag{3}$$

The same formalism can also be extended to the case of multiple cuts with additional bookkeeping. Let there be K cuts and $K + 1$ subcircuits. For each cut, there will be an associated triple $(\hat{a}_k, e_k, \beta_k)$ representing the connection bit, the eigenvalue, and measurement/initialization basis respectively. Furthermore, each subcircuit tensor will have its respective elements, namely, $\hat{a}_k$ or e_k, depending on whether it is the upstream or downstream circuit. The matrix-product state formalism gives a concise representation for generalized circuit cutting (cf. [14], [17], [18]).

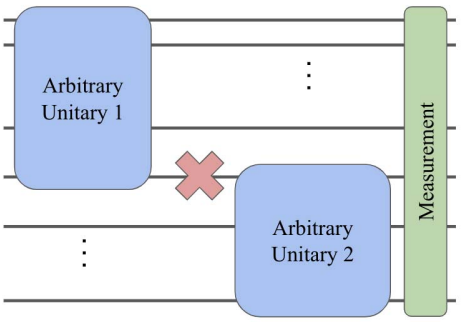
This contraction involves summing over 12 terms — there are two possibilities each for $\hat{a}$ and e, three for β — which means that simulating the probability of obtaining one particular bit string takes only constant time. However, the naive method for constructing the output of the full distribution requires querying the probability of each bit string, which there are exponentially many, one by one. Also, since the representation above is only exact given completely accurate state tomography, the realistic recovered "probability distribution" might contain negative probabilities or are not normalized due to finite-shot noise and hardware noise (if ran on a physical device). Moreover, the time complexity of reconstructing the full distribution also grows exponentially with respect to the number of cuts as three additional indices are to be contracted for each additional cut. These unfavorable scaling factors inhibit circuit cutting from being used at its full potential.

III. Approximate Circuit Reconstruction

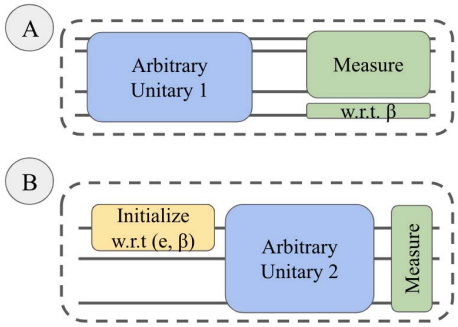
This section introduces the idea of an *approximate circuit reconstruction* that seeks to avoid exponential scaling. For ease of exposition, we will demonstrate the reconstruction for the case of a single cut and two cuts; generalizing to arbitrary cuts is straightforward but tedious.

The high-level idea is as follows: our goal is to build an approximate distribution with polynomially many queries from an exponentially growing state space. Since the weight of each bit string is unknown a priori, there is not a good deterministic strategy for approximating the

Authorized licensed use limited to: Kent State University Libraries. Downloaded on June 15,2023 at 20:54:54 UTC from IEEE Xplore. Restrictions apply.



(a) Circuit prior to cutting. The cut location is specified at the red cross. Upon performing the cut, the circuit naturally separates into two independent subcircuits that can be run in parallel.



(b) Resulting subcircuits. (top) The last qubit of the upstream subcircuit, denoted by $\hat{a}$, requires measurements in basis β. The first qubit of downstream subcircuit (bottom) requires preparing the eigenstate of β corresponding to eigenvalue e.

Fig. 1: A pictorial representation of the circuit cutting procedure.

distribution. Thus, we introduce a naive version of the Metropolis-Hastings (MH) algorithm, which is a type of Markov Chain Monte Carlo (MCMC) method. Let $f(\hat{x})$ be a probability distribution, $\hat{x} \in \{0,1\}^n$, that we would like to sample from. Moreover, the unnormalized likelihood ratio between adjacent points in the support of the distribution can be computed efficiently. To avoid computing $f(\hat{x})$ on every x in the support, we perform random walks on a Markov chain defined on $\{0,1\}^n$ with transition kernel $g(\hat{y}|\hat{x})$. Then, at each step of the random walk, we move to state $\hat{y}$ from state $\hat{x}$ probabilistically with the probability of transition increasing if the ratio of likelihoods increases. For details of MH algorithms and Markov chain mixing, readers are referred to [19], [20], [21], [22].

A. Example: one-cut case

Consider the case that a single cut has been made, producing outcomes p_A and p_B from the respective subcircuits where equation (2) describes the relationship between the subcircuits and the full circuit. In our case, each state $\hat{x}$ represents a possible output from the full quantum circuit and the distribution we would like to sample from is the one described in equation (2). The complete procedure is described in Algorithm 1.

Here, the transition kernel is one that flips one randomly-chosen bit from the conditioned bit string. This procedure is easily parallelizable in the case of one cut. One can establish multiple *chains*: implementation of the same algorithm with different initial conditions running in parallel. This comes at the advantage of not only time complexity, but also prevents improper chain mixing and guarantees convergence in the sample size limit.

Algorithm 1: Metropolis-Hastings algorithm for the one-cut case

Input: N: number of samples, BI: burn-in
Output: S: histogram of accepted bit strings

$S \leftarrow \{\}, \quad n \leftarrow 0$
$\hat{x} \leftarrow$ random bit string of length m
while $n < N$ **do**
  $\hat{y} \leftarrow$ bit string of length m that is different with x in only 1 entry
  Let $r = \min\{1, P^{(\hat{y})}/P^{(\hat{x})}\}$ (cf. eq. (2))
  if $u \sim$ *Uniform(0,1)* $< r$ **then**
    $\hat{x} \leftarrow \hat{y}$
  if $n >$ *BI* $\cdot N$ **then**
    $S[\hat{x}] \leftarrow S[\hat{x}] + 1$
  $n \leftarrow n + 1$
return S

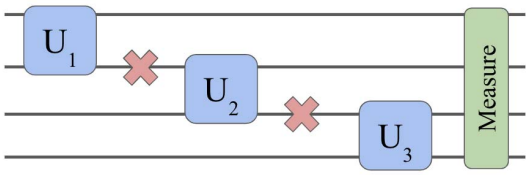


Fig. 2: Example of a circuit with two cuts. Subcircuits A, B, and C can be arbitrary.

B. Example: two-cut case

Now consider the case where two cuts are made, resulting in three subcircuits A, B, and C of size n_1, n_2, n_3 respectively. For concreteness, suppose the cuts follow the cascade-like structure depicted in Figure

Authorized licensed use limited to: Kent State University Libraries. Downloaded on June 15,2023 at 20:54:54 UTC from IEEE Xplore. Restrictions apply.

2. We will refer to the cut between subcircuit A and B as the first cut, and the one between B and C as the second. Subcircuit A is upstream of B, meaning that it will need additional measurements at the location of the cut, denoted by indices $\hat{a}_1$ and β_1. Subcircuit C is downstream of B, so it will need additional initializations at the location of the cut, represented by the indices e_2, β_2. Subcircuit B is downstream of A and upstream of C, which means that it will contain information about both subcircuits, meaning that p_B will be a rank-5 tensor with indices $e_1, \beta_1, \hat{a}_2, \beta_2$ along with the measurement bit strings.

We begin to realize the exponential scaling starting in the two-cut case. The exponential growth comes from the fact that, for each cut, we produce a pair of measurement-initialization combinations. Quantum information must be passed through each cut, which means that we must sum over all measurement-initialization pairs for every cut, resulting in roughly $2^{\text{poly}K}$ in runtime complexity. Here, we will propose two types of approximate reconstruction. The first obeys the exponential scaling with K to gain accuracy at the cost of efficiency. It involves running six chains, one for each index pair (e_1, β_1), as shown in Algorithm 2.

Algorithm 2: Metropolis-Hastings algorithm for the two-cut case

Input: N: number of samples, BI: burn-in
Output: S: histogram of accepted bit strings

$p_{BC} \leftarrow \texttt{zeros}^{(\hat{b}_1 \ldots \hat{b}_{n_2+n_3-1}, e_1, \beta_1)}$
$n \leftarrow [0,0,0,0,0,0]$
$[\hat{x}_1, \ldots, \hat{x}_6] \leftarrow$ random bit strings of length $n_2 + n_3 - 1$
while $\min(n) < N$ **do**
  for $(e_1, \beta_1) \in \{0,1\} \times \mathcal{B}$ **do**
    $\hat{y} \leftarrow$ bit string of length $n_2 + n_3 - 1$ that is different with $\hat{x}$ in only 1 entry
    Let $r = \min\{1, P^{(\hat{y})}/P^{(\hat{x})}\}$
    if $u \sim$ *Uniform(0,1)* $< r$ **then**
      $\hat{x}_i \leftarrow y$
    if $n >$ *BI* $\cdot N$ **then**
      $p_{BC}^{(\hat{x}, e_1, \beta_1)} \leftarrow p_{BC}^{(\hat{x}, e_1, \beta_1)} + 1$
    $n[i] \leftarrow n[i] + 1$
return $\texttt{reconstructOneCut}(p_A, p_{BC})$

The function `reconstructOneCut` calls Algorithm 1. Here, we simply avoided the exponential scaling in the space of bit strings. However, the scaling with respect to the number of cuts will be preserved as more indices are added into the middle for-loop. As an attempt to avoid this scaling as well, we propose to select which index to contract uniformly at random. The algorithm is presented in Algorithm 3

Algorithm 3: Metropolis-Hastings algorithm for the two-cut case with randomized indexing

Input: N: number of samples, BI: burn-in
Output: S: histogram of accepted bit strings

$p_{BC} \leftarrow \texttt{zeros}^{(\hat{b}_1 \ldots \hat{b}_{n_2+n_3-1}, e_1, \beta_1)}$
$n \leftarrow [0,0,0,0,0,0]$
$[\hat{x}_1, \ldots, \hat{x}_6] \leftarrow$ random bit strings of length $n_2 + n_3 - 1$
while $\min(n) < N$ **do**
  $e_1 \leftarrow 0, 1$ uniformly at random
  $\beta \leftarrow \sigma_x, \sigma_y, \sigma_z$ uniformly at random
  $\hat{y} \leftarrow$ bit string of length $n_2 + n_3 - 1$ that is different with $\hat{x}$ in only 1 entry
  Let $r = \min\{1, P^{(\hat{y})}/P^{(\hat{x})}\}$
  if $u \sim$ *Uniform(0,1)* $< r$ **then**
    $\hat{x}_i \leftarrow y$
  if $n >$ *BI* $\cdot N$ **then**
    $p_{BC}^{(\hat{x}, e_1, \beta_1)} \leftarrow p_{BC}^{(\hat{x}, e_1, \beta_1)} + 1$
  $n[i] \leftarrow n[i] + 1$
return $\texttt{reconstructOneCut}(p_A, p_{BC})$

The asymptotic property is preserved even if we randomly choose which index to contract over. However, the rate at which it converges is slowed, which is the usual trade-off between efficiency and accuracy.

C. Generalizing to an arbitrary number of cuts

Implementing a general circuit cutting routine is a rather tedious task. The *tensor network* formalism becomes convenient for consistent book-keeping [12]. For each quantum circuit, we can map it to a corresponding directed graph: the vertices are quantum gates and an initial set of qubits. Two vertices (gates) are connected if it is directly connected by a qubit wire in the circuit. An example is demonstrated in Figure 3.

The circuit cut then corresponds to removing one edge. And by the graph formalism, we can immediately identify the gates and qubits those gates operate on upstream and downstream of the cut. A subcircuit is produced only when the resulting graph is no longer in one component and this procedure might take multiple cuts (removal of edges) to achieve. For each subcircuit, the associated tensor will be of rank $2k+1$ where k is the number of cuts the subcircuit is involved in. For example, in the two-cut example (Figure 1b), subcircuit A can be stored into a rank-3 tensor because it is upstream of one cut. On the other hand, subcircuit B is sandwiched between two cuts, so there are $2 * 2 + 1 = 5$ indices. To combine two subcircuits, one can use Algorithm 1, 2, or 3 depending on the situation, so long as the right indices are being contracted.

Authorized licensed use limited to: Kent State University Libraries. Downloaded on June 15,2023 at 20:54:54 UTC from IEEE Xplore. Restrictions apply.

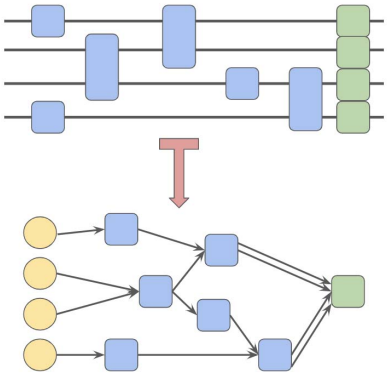


Fig. 3: Example mapping from quantum circuit to a directed graph. Each gate is a vertex and an edge connects two vertices if the two gates are connected directly by a qubit wire.

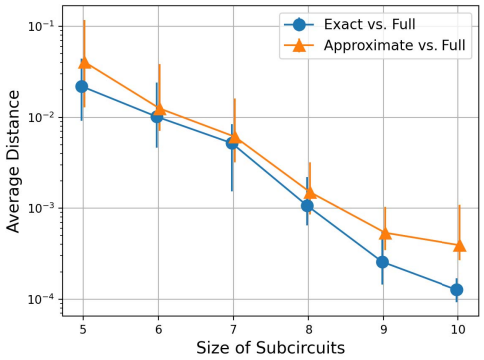


Fig. 4: The average distance (in log scale) with respect to the size of each subcircuit in the case of one cut. The error bars show the 25-th and 75-th quantiles.

IV. Experiments

We give preliminary empirical results for the effectiveness of our proposed approximate circuit cutting scheme. The experiments were run on a personal computer equipped with Intel i5 processors. The algorithms implemented follow exactly as sketched in the pseudocode presented in the previous section. We will show that, despite the simplicity of the program, our method scales well with circuit size both in terms of time complexity and performance.

To compare two distributions, we propose the following *average variational distance* metric: for distribution $p(x)$ and $q(x)$ over the same sample space $\mathcal{X}$, the average variational distance is define by the following:

$$\mathcal{D}(p;q) = \sum_{x\in\mathcal{X}} \left(p(x) - q(x)\right)\mu(x) \tag{4}$$

where $\mu(x)$ is the average probability distribution over $\mathcal{X}$ of p and q:

$$\mu(x) = \frac{p(x) + q(x)}{2} \tag{5}$$

Intuitively, this represents the difference in likelihood functions with respect to the average distribution. This metric effectively captures the goal of approximate circuit cutting: estimating bit strings of large probability. If our estimate distribution has the same shape as the intended distribution, that is, the two distributions share the same high-probability bit strings, then $\mathcal{D}(p;q)$ will not punish the discrepancy for bit strings in the low likelihood regime.

A. Case of One Cut

For the one-cut case, we generated random subcircuits of the same sizes using Qiskit [23], then connect the two subcircuits using a CNOT gate. We compared the distributions resulting from the exact reconstruction and the approximate reconstruction against a run of the full circuit without any cuts. This process is repeated for 30 trials, and the results are shown in Figure 4. We can see that the exact reconstruction method consistently outperformed the approximate one, which was expected by virtue of probabilistic methods. However, the discrepancy is not large and overall was close to that of the exact reconstruction. The main advantage of the approximate reconstruction is the computational complexity, which was measured empirically and displayed in Figure 5. Here, we linearly increased the number of samples taken with respect to the size of each subcircuit. As a result, the time needed for computation only grows linearly. However, performing exact reconstruction naively will require exponential time. Moreover, the time recorded did not account for time for normalizing the distribution. This shows another advantage of approximate reconstruction: it requires no extra post-processing for normalization. It is important to note that, like all Monte Carlo methods, the more samples are taken, the better the performance is. One could choose to scale the number of samples taken to be more than linear with respect to the size of the circuit if one can afford the time and computational resources.

B. Case of Two Cuts

We repeat a similar experiment for the case of two cuts. We generate subcircuits of equal size randomly and connect each subcircuit using a CNOT gate, structured like one depicted in Figure 2. And since circuit cutting generally improves the fidelity [13], we will compare the exact reconstruction method with the two proposed approximate methods: one that loops over all indices, and the other randomly selects the index. Again, for

Authorized licensed use limited to: Kent State University Libraries. Downloaded on June 15,2023 at 20:54:54 UTC from IEEE Xplore. Restrictions apply.

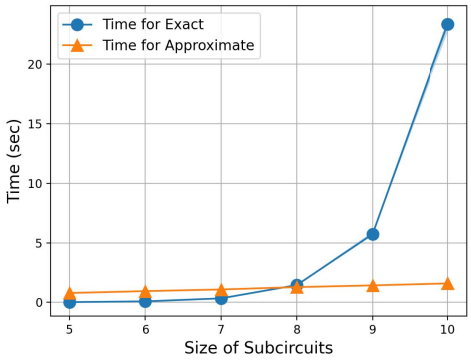


Fig. 5: Comparison of running time with respect to the size of subcircuits in the case of one cut.

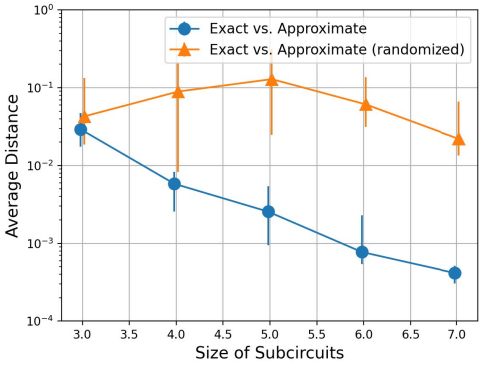


Fig. 6: The average distance (in log scale) with respect to the size of each subcircuit for the case of two cuts. The error bars show the 25-th and 75-th quantiles.

each size of subcircuits, the same experiment was repeated 30 times. The results are shown in Figure 6. The approximate method without randomized indexing behaved similarly to the case of one cut, showing that the approximate reconstruction procedure can be applied sequentially without significant deterioration in the quality of reconstruction. However, randomly choosing indices resulted in poor accuracy. It is important to keep in mind that randomized indexing was used to escape the exponential scaling in the number of cuts. We suspect that the accuracy of the randomized indexing method can be improved with much larger sample sizes. However, for only a few cuts, sacrificing such magnitude of accuracy is inappropriate. Meanwhile, exponential runtime is again mitigated by controlling the inflation of sample sizes, as in Figure 7.

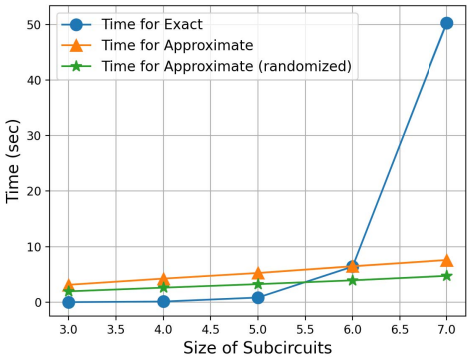


Fig. 7: Comparison of running time with respect to the size of subcircuits in the case of two cuts.

V. Conclusion and Future Directions

In this paper, we proposed the concept of *approximate circuit reconstruction*, which aims at piecing together the measurement outcome of subcircuits in a randomized way to avoid unfavorable scaling both in terms of the circuit size and the number of cuts. The experimental results show that this is a promising direction. Below, we list two particular directions to explore.

a) Improved Algorithm Design: The current implementation uses the simplest implementation of the Metropolis-Hastings algorithm. Furthermore, no assumptions of quantumness were made throughout. As a result, there is significant room for designing more sophisticated sampling methods that take advantage of existing structures.

b) Bayesian Reconstruction: Traditionally, MCMC methods are used for Bayesian computation to neglect an intractable normalizing factor that is produced in invoking Bayes' theorem. Here, sampling was used to escape the imperfect normalizing factor from statistical shot noise. Yet, there is potential for further exploiting the properties of MCMC algorithms. For example, integrating priors into reconstructions might be beneficial for reducing noise while keeping computation tractable. However, one would need a probabilistic model for classical distributions generated by quantum circuits, which will also be left as a future direction.

Acknowledgement

This work was partially supported by NSF PPoSS under award number 2216923. N.X. was partially supported by the U.S. Army Research Office (ARO) under award number W911NF1910362.

References

[1] E. Bernstein and U. Vazirani, "Quantum complexity theory," *SIAM Journal on computing*, vol. 26, no. 5, pp. 1411–1473, 1997.

Authorized licensed use limited to: Kent State University Libraries. Downloaded on June 15,2023 at 20:54:54 UTC from IEEE Xplore. Restrictions apply.

[2] J. Preskill, "Quantum computing in the nisq era and beyond," *Quantum*, vol. 2, p. 79, 2018.
[3] K. Bharti, A. Cervera-Lierta, T. H. Kyaw, T. Haug, S. Alperin-Lea, A. Anand, M. Degroote, H. Heimonen, J. S. Kottmann, T. Menke *et al.*, "Noisy intermediate-scale quantum algorithms," *Reviews of Modern Physics*, vol. 94, no. 1, p. 015004, 2022.
[4] M. Cerezo, A. Arrasmith, R. Babbush, S. C. Benjamin, S. Endo, K. Fujii, J. R. McClean, K. Mitarai, X. Yuan, L. Cincio *et al.*, "Variational quantum algorithms," *Nature Reviews Physics*, vol. 3, no. 9, pp. 625–644, 2021.
[5] E. Farhi, J. Goldstone, and S. Gutmann, "A quantum approximate optimization algorithm," *arXiv preprint arXiv:1411.4028*, 2014.
[6] A. Peruzzo, J. McClean, P. Shadbolt, M.-H. Yung, X.-Q. Zhou, P. J. Love, A. Aspuru-Guzik, and J. L. O'brien, "A variational eigenvalue solver on a photonic quantum processor," *Nature communications*, vol. 5, no. 1, pp. 1–7, 2014.
[7] A. Kandala, A. Mezzacapo, K. Temme, M. Takita, M. Brink, J. M. Chow, and J. M. Gambetta, "Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets," *Nature*, vol. 549, no. 7671, pp. 242–246, 2017.
[8] M. J. Powell, "A direct search optimization method that models the objective and constraint functions by linear interpolation," in *Advances in optimization and numerical analysis*. Springer, 1994, pp. 51–67.
[9] M. Schuld, V. Bergholm, C. Gogolin, J. Izaac, and N. Killoran, "Evaluating analytic gradients on quantum hardware," *Physical Review A*, vol. 99, no. 3, p. 032331, 2019.
[10] C. Xue, Z.-Y. Chen, Y.-C. Wu, and G.-P. Guo, "Effects of quantum noise on quantum approximate optimization algorithm," *Chinese Physics Letters*, vol. 38, no. 3, p. 030302, 2021.
[11] S. Wang, E. Fontana, M. Cerezo, K. Sharma, A. Sone, L. Cincio, and P. J. Coles, "Noise-induced barren plateaus in variational quantum algorithms," *Nature communications*, vol. 12, no. 1, pp. 1–11, 2021.
[12] T. Peng, A. W. Harrow, M. Ozols, and X. Wu, "Simulating large quantum circuits on a small quantum computer," *Physical Review Letters*, vol. 125, no. 15, p. 150504, 2020.
[13] W. Tang, T. Tomesh, M. Suchara, J. Larson, and M. Martonosi, "Cutqc: using small quantum computers for large quantum circuit evaluations," in *Proceedings of the 26th ACM International Conference on Architectural Support for Programming Languages and Operating Systems*, 2021, pp. 473–486.
[14] T. Ayral, F.-M. Le Régent, Z. Saleem, Y. Alexeev, and M. Suchara, "Quantum divide and compute: Hardware demonstrations and noisy simulations," in *2020 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*. IEEE, 2020, pp. 138–140.
[15] M. A. Perlin, Z. H. Saleem, M. Suchara, and J. C. Osborn, "Quantum circuit cutting with maximum-likelihood tomography," *npj Quantum Information*, vol. 7, no. 1, pp. 1–8, 2021.
[16] Z. H. Saleem, T. Tomesh, M. A. Perlin, P. Gokhale, and M. Suchara, "Quantum divide and conquer for combinatorial optimization and distributed computing," *arXiv preprint arXiv:2107.07532*, 2021.
[17] U. Schollwöck, "The density-matrix renormalization group in the age of matrix product states," *Annals of physics*, vol. 326, no. 1, pp. 96–192, 2011.
[18] R. Orús, "A practical introduction to tensor networks: Matrix product states and projected entangled pair states," *Annals of physics*, vol. 349, pp. 117–158, 2014.
[19] C. P. Robert and G. Casella, "The metropolis—hastings algorithm," in *Monte Carlo statistical methods*. Springer, 1999, pp. 231–283.
[20] A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin, *Bayesian data analysis*. Chapman and Hall/CRC, 1995.
[21] S. Brooks, A. Gelman, G. Jones, and X.-L. Meng, *Handbook of markov chain monte carlo*. CRC press, 2011.
[22] D. A. Levin and Y. Peres, *Markov chains and mixing times*. American Mathematical Soc., 2017, vol. 107.
[23] A. Cross, "The ibm q experience and qiskit open-source quantum computing software," in *APS March meeting abstracts*, vol. 2018, 2018, pp. L58–003.

Authorized licensed use limited to: Kent State University Libraries. Downloaded on June 15,2023 at 20:54:54 UTC from IEEE Xplore. Restrictions apply.