Qubit-Reuse Compilation with Mid-Circuit Measurement and Reset

Matthew DeCross,[*] Eli Chertkov,[†] Megan Kohagen,[‡] and Michael Foss-Feig[§]

Quantinuum, 303 South Technology Court, Broomfield, Colorado 80021, USA

(Received 7 April 2023; revised 18 August 2023; accepted 1 November 2023; published 22 December 2023)

A number of commercially available quantum computers, such as those based on trapped-ion or superconducting qubits, can now perform mid-circuit measurements and resets. In addition to being crucial for quantum error correction, this capability can help reduce the number of qubits needed to execute many types of quantum algorithms by measuring qubits as early as possible, resetting them, and reusing them elsewhere in the circuit. In this work, we introduce the idea of qubit-reuse compilation, which takes as input a quantum circuit and produces as output a compiled circuit that requires fewer qubits to execute due to qubit reuse. We present two algorithms for performing qubit-reuse compilation: an exact constraint programming optimization model and a greedy heuristic. We introduce the concept of *dual circuits*, obtained by exchanging state preparations with measurements and vice versa and reversing time, and show that optimal qubit-reuse compilation requires the same number of qubits to execute a circuit as its dual. We illustrate the performance of these algorithms on a variety of relevant near-term quantum circuits, such as one-dimensional and two-dimensional time-evolution circuits, and numerically benchmark their performance on the quantum adiabatic optimization algorithm (QAOA) applied to the MaxCut problem on random three-regular graphs. To demonstrate the practical benefit of these techniques, we experimentally realize an 80-qubit QAOA MaxCut circuit on the 20-qubit Quantinuum H1-1 trapped-ion quantum processor using qubit-reuse compilation algorithms.

DOI: 10.1103/PhysRevX.13.041057 Subject Areas: Quantum Physics, Quantum Information

I. INTRODUCTION

Current quantum computers are limited by the number of qubits available for computation. To conclusively demonstrate the computational advantage of these devices compared to their classical counterparts on a variety of practical applications, researchers need to make efficient use of qubits.

Generally, the process of submitting a quantum circuit to solve a problem of interest consists of devising a quantum algorithm, generating the associated quantum circuit, performing circuit optimization for a quantum device of interest, submitting to the quantum device, and retrieving the results. The step of circuit optimization is key on noisy intermediate-scale quantum (NISQ) devices. Circuit optimization takes many different forms but typically consists of modifying the gate structure of a quantum circuit to increase the fidelity of results. Some examples of circuit optimizations include reducing gate count by simplifying Clifford subcircuits [1–4], removing redundant gates [3–5], and using the Cartan decomposition of two-qubit gates [6], or mapping circuits to a specific quantum device architecture to decrease potential transport or SWAP gate costs [3,4,7,8].

An underexplored area of quantum circuit design is that of qubit reuse. The ability of a quantum computer to perform mid-circuit measurements and resets enables the reuse of qubits after resets. Qubit reuse is an essential ingredient of scalable quantum error correction protocols [9], which require repeated mid-circuit measurements and resets of ancilla qubits to measure error syndromes, and is already available in trapped-ion [10] and superconducting [11] qubit architectures. Recently, qubit-reuse techniques have been used to experimentally prepare and time evolve large tensor network states on trapped-ion quantum computers [12], to study a nonequilibrium phase transition [13], to extract critical scalings of quantum systems near a phase transition [14], to build quantum machine learning models for sequence classification [15], and to perform entanglement spectroscopy of quantum systems [16].

In this paper, we investigate the qubit-reuse compilation problem: the task of converting a quantum circuit into an equivalent circuit that uses fewer qubits via qubit reuse. We demonstrate two algorithms for performing qubit-reuse

[*]matthew.decross@quantinuum.com
[†]eli.chertkov@quantinuum.com
[‡]megan.l.kohagen@quantinuum.com
[§]michael.feig@quantinuum.com

Published by the American Physical Society under the terms of the Creative Commons Attribution 4.0 International license. Further distribution of this work must maintain attribution to the author(s) and the published article's title, journal citation, and DOI.

compilation. The first is an exact algorithm that uses a constraint programming and satisfiability (CP-SAT) model. Since this algorithm uses a general-purpose type of model for solving combinatorial optimization problems, it does not scale well with qubit number but provides a useful benchmark for other methods at low qubit number. The second algorithm is a greedy heuristic that runs quickly up to large numbers of qubits and scales polynomially with qubit number.

We benchmark these algorithms numerically on the quantum approximate optimization algorithm (QAOA) MaxCut circuits on random three-regular graphs. We also investigate several examples of highly structured circuits, including 1D and 2D time-evolution circuits and certain quantum tensor networks, and solve the qubit-reuse compilation problem analytically for these cases. Using our algorithms, we are able to compress an 80-qubit QAOA circuit into a 20-qubit circuit, which we then experimentally execute on the Quantinuum H1-1 trapped-ion quantum computer [17], which has high-fidelity and low-crosstalk mid-circuit measurement and reset capabilities [10], as well as the recently announced [18] support for arbitrary angle two-qubit rotation gates.

Techniques for executing large quantum circuits on a smaller number of qubits have been developed previously in, for example, Refs. [19–24]. Many of these methods are based on cutting the original circuits into smaller circuits and specifying a protocol for "knitting" together the results obtained from the subcircuits. Depending on the way in which this is accomplished, there can be a number of drawbacks to this approach. For instance, circuits corresponding to graph problems can be cut by approximately factorizing the graph by removing special edges. However, the resulting measurements cannot sample from fully entangled states and correspond only to approximate solutions even in the limit of very deep circuits [25]. Methods based on inserting randomized measurements or Pauli operators circumvent this problem but generally proliferate the number of circuits exponentially in the number of cut wires or require distributed networks of quantum computers to run the resulting cut circuits. See also Refs. [28,29] for work on compressing the MaxCut problem by assigning multiple graph vertices to different Pauli operators acting on the same qubit. Broadly speaking, while circuit cutting has general applicability to any circuit, one should expect its most effective use to be restricted to circuits that nearly factorize, with only a small number of gates coupling different subsets of qubits.

Our work is novel in that it provides a one-to-one mapping of a quantum circuit into a single equivalent circuit that uses fewer qubits and samples from the identical measurement distribution as the original circuit. The key underlying technique is not based on circuit cutting but rather takes advantage of the causal structure of the circuit, by executing the *causal cones* of output qubits in a prescribed order. The causal cone of a measurement operation consists of the set of operations in the circuit that influence the distribution from which that measurement samples; see Fig. 2. Causal-cone techniques have been utilized before in the context of the MaxCut problem [30,31] to run large graph instances on tensor network simulators with distributed computing networks and/or supercomputers. The concept of reusing physical qubits after they are no longer needed during a computation has also been discussed before, for example, in Ref. [32], which selected the next qubit to reuse based on how many qubits were required to measure it. This work provides a compilation protocol for quantum programs in the circuit model rather than classical tensor networks and, more importantly, details sophisticated algorithms that specify optimized orders in which to execute causal cones. We also use this technique to execute the full QAOA optimization protocol on hardware, demonstrating the near-term practical application of qubit-reuse compilation. In Supplemental Material [33], we provide visual examples of the effect of qubit-reuse compilation on some sample application circuits, which demonstrates what the sequential execution of causal cones looks like as a complete quantum circuit. Similar to their use in the execution of classical tensor networks, we expect causal-cone techniques for qubit reuse to be most effective in shallow and wide circuits, and we anticipate qubit reuse to be a useful general-purpose tool for efficiently using qubit resources in that setting. This lies in contrast to the general applicability of circuit cutting to any circuit but is also similar to circuit cutting in the sense that it is most effective or resource efficient for circuits with a particular gate structure. Nonetheless, we emphasize that qubit reuse and circuit cutting are not exclusive techniques but may potentially be employed in conjunction with each other to obtain even greater qubit resource reductions.

Two relevant works were posted concurrently with this work, Refs. [34,35], which also focus on cominimizing SWAP gate count along with required qubit resources on IBM systems. Reference [34] presents an algorithm for qubit reuse that appears functionally identical to our greedy heuristic, which we show in this work is nonoptimal by explicit computation of alternative orderings of causal-cone execution (see Fig. 3 for a simple example of where a greedy heuristic is not globally optimal). In contrast to Ref. [34], this work is centrally focused on reducing qubit resource overhead independent of considerations of the architecture. In architectures with limited qubit connectivity, such as all current superconducting architectures, maximally reducing the number of required qubits may increase the SWAP gate overhead. Reference [35] also presents a general-purpose algorithm for reducing qubit resources from a different perspective. The algorithm in Ref. [35] starts with an initial quantum circuit and iteratively chooses a qubit to reuse, reducing the required number of qubits by one each time. This algorithm is a greedy method that attempts to reduce the size of a circuit until it fits within a desired user-specified limit. Our work can be viewed as additionally providing methods to specify the order in which qubits should be iteratively chosen for reuse such that the qubit resource savings can be maximized, although we take a perspective in which the qubit-reuse compiled circuit is built up through sequential execution of causal cones rather than obtained through iterative reduction of the original circuit.

We emphasize that, although our experimental demonstration is performed on the Quantinuum H1-1 trapped-ion quantum processor, the algorithms described in this work are not architecture-specific. Our qubit-reuse compilation can be applied to quantum programs targeting any architecture with any connectivity for the purpose of minimizing required qubit number. In the case of limited connectivity, the compilation simply must be applied before SWAP gate insertion and other routing compilation passes.

Although our techniques are designed with the limitations of current architectures in mind, we note that qubit-reuse compilation is not limited to the NISQ era. Owing to the large overhead of physical qubits required to encode a single logical qubit in quantum error correction, early fault-tolerant quantum computers are also starved for logical qubits. Qubit-reuse compilation works equally well in that setting, with physical qubits and gates replaced by their logical equivalents. In any case, regardless of the physical or logical qubit number supported by architectures in the future, we envision that qubit reuse can generally provide benefits analogous to the use of causal structure in classical tensor network techniques to exchange time and memory overheads. That is, at any fixed qubit count, qubit reuse can potentially enable the solution of larger problems than supported on hardware.

The rest of this paper is organized as follows. In Sec. II, we introduce the concept of qubit-reuse compilation using causal cones. In Sec. III, we present exact and heuristic algorithms for determining the order in which to implement causal cones in a quantum circuit, to attempt to minimize the number of qubits needed to implement the compiled circuit. We also introduce the concept of a *dual circuit*, show that the same number of qubits are required to implement the optimal compression of a circuit and its dual, and describe how this can be used to potentially improve heuristic algorithms for qubit reuse. In Sec. IV, we study certain structured quantum circuits analytically, providing exact results for the number of qubits required to optimally compress these circuits. In Sec. V, we benchmark the run-time and performance of our qubit-reuse algorithms, applied to MaxCut QAOA circuits at different depths as well as to computing local correlation functions of some circuits that we study analytically in the previous section. In Sec. VI, we experimentally demonstrate the practical value of qubit-reuse compilation applied to MaxCut QAOA by solving an $N = 80$ MaxCut problem using the 20 physical qubits available on the Quantinuum H1-1 trapped-ion quantum computer. We conclude with remarks in Sec. VII, in particular, focusing on the impacts of noise in qubit-reuse compiled circuits.

II. QUBIT REUSE VIA MID-CIRCUIT MEASUREMENT

The key principle underlying the ability to measure and reuse qubits to reduce qubit number overhead is the fact that in many cases only partial execution of a circuit encoding a quantum program is required to measure a given qubit, at which point that qubit can potentially be reset to play the role of a different logical qubit elsewhere in the circuit. Figure 1 demonstrates how a qubit that is no longer required to implement the remainder of a circuit can be measured, reset, and reused as a different logical qubit.

Specifically, the measurement of a given qubit's quantum state depends only on operations in its past causal cone (defined in the previous section). Exploiting the causal structure of quantum circuits in quantum simulation is a technique frequently utilized in classical tensor network methods, such as those based on the multiscale entanglement renormalization *Ansatz* (MERA) [36], and has been recently applied to numerous quantum extensions

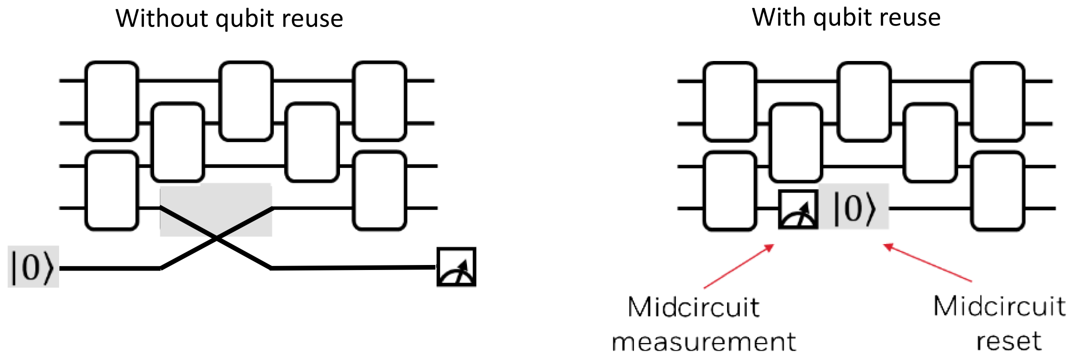


FIG. 1. On the left, the lowermost qubit is not used until after crossing the shaded gray region, while the adjacent qubit is no longer used after that region. The overall circuit can be executed using one fewer qubit using mid-circuit measurement and reset, as depicted on the right.

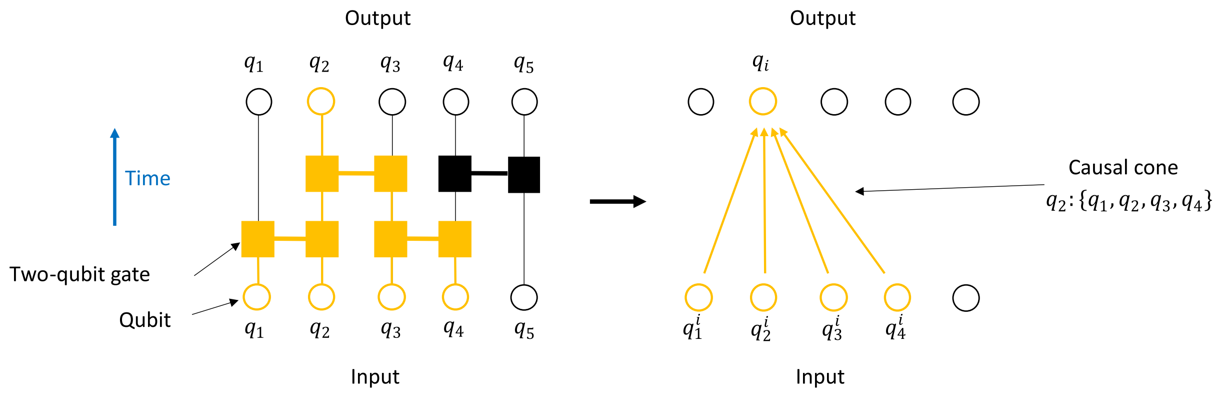


FIG. 2. Identification of the causal cone of the output qubit q_2 in the displayed quantum circuit, consisting of the set of operations that influence the distribution sampled by measurements of q_2. Measuring and resetting q_2 requires only gates executed between the four input qubits $\{q_1, q_2, q_3, q_4\}$, after which q_2 can be reset and reused as q_5.

of tensor-network methods for both many-body physics and machine learning [12,13,37–43] and to several examples of variational quantum algorithms in Refs. [44–46].

However, to the best of our knowledge, previous examples of compression based on qubit reuse executed on hardware have proceeded by inspection of circuits with particularly simple structure, without a general framework for automating the compilation of a circuit to run on fewer qubits. In this work, we develop algorithms that determine an order for measuring and reusing qubits based on their causal cones within a quantum circuit. Figure 2 illustrates how a given output qubit depends causally on a restricted subset of input qubits.

The process of compiling a given quantum circuit into a circuit with fewer qubits is performed in two steps. (i) We determine an order in which to measure and reset the qubits in the original circuit. This order can be determined numerically by the algorithms we describe below, which aim to globally minimize the number of qubits required to run the final circuit, or can be determined analytically. (ii) We use the determined measurement order to rewrite the original circuit into a smaller circuit by measuring and resetting the qubits in the desired order. We emphasize that *any* measurement order can be used in (ii), so one does not need to restrict oneself to minimizing the qubit number but can instead determine measurement orders that optimize more complicated objectives, e.g., objectives that strike a balance between minimizing hardware errors and minimizing qubits on a noisy qubit-limited device. Both (i) and (ii) are fully automated and integrated into a software package used to produce the numerical results discussed in this work.

We note that a circuit $\mathcal{C}$ and any version of that circuit obtained by compression via qubit-reuse compilation, $\mathcal{R}(\mathcal{C})$, do not generate equivalent operators on a set of qubits (they do not even act on qubit sets of the same size). Rather, their equivalence is in the restricted sense that they produce measurement data drawn from identical distributions, which we denote as $\mathcal{R}(\mathcal{C}) \cong \mathcal{C}$. This equivalence is fairly strong, in that no classical postprocessing of the circuit output can distinguish $\mathcal{R}(\mathcal{C})$ from $\mathcal{C}$. However, quantum postprocessing of the state generated prior to measurement in $\mathcal{C}$ (i.e., extending the circuit $\mathcal{C}$ to include more gates) cannot generally be carried out in an equivalent fashion on $\mathcal{R}(\mathcal{C})$, since $\mathcal{R}(\mathcal{C})$ never actually produces the quantum state generated by $\mathcal{C}$ immediately prior to measurement.

We also point out that a given quantum circuit $\mathcal{C}$ and its compressed version $\mathcal{R}(\mathcal{C})$ contain an identical set of gates only in the case that the target device architecture supports interactions between arbitrary qubits, i.e., is fully connected. In architectures that support only nearest-neighbor or similar interactions, both the circuit and its compressed version generally require the insertion of SWAP gates to execute, and the location and number of these SWAP gates might not be the same in both the circuit and its compressed version. See Refs. [34,35] for some considerations of SWAP-aware qubit reuse.

III. ALGORITHMS FOR OPTIMIZING QUBIT REUSE

A. Exact solution by constraint programming

The problem of minimizing the required number of qubits to execute a given quantum program can be formulated in the language of a CP-SAT problem. The optimization version of CP-SAT is formulated as the minimization of an objective function subject to a set of equality and inequality constraints as well as (potentially) a set of conditional constraints. By formulating the qubit-reuse compilation problem as a CP-SAT problem, we can use existing solvers to find globally optimal solutions. While in the worst case these solvers run in time superpolynomial in the problem size, they can be used to obtain exact solutions for small quantum circuits or to find approximate solutions for larger circuits by constraining the run-time.

Suppose that we are given a quantum circuit defined on N qubits and wish to determine a measurement order of the N qubits that allows us to execute the program on (possibly) fewer than N qubits by measuring, resetting, and reusing certain qubits. We define the following problem input data:

$q \in Q$: qubits in the circuit;

$t \in T$: indexes the order in which the qubit causal cones are executed and measured;

C_q: set of qubits initialized in the causal cone of the measurement of qubit q.

Here, $q \in Q = \{1, \ldots, N\}$ indexes the original set of N qubits Q, while $t \in T = \{1, \ldots, N\}$ indexes the order in which each of the causal cones of the measurements of the N original qubits are implemented. At each step t, one qubit is measured and reset. Note that t is not necessarily related to any notion of time in quantum circuits like the number of layers of two-qubit gates applied to the input state. To "run the circuit up to step t" means to apply only the operations in the causal cones of all measurements performed in steps $1, 2, \ldots, t$.

The optimization model for the qubit-reuse problem is defined in terms of two sets of N^2 binary variables and an integer-valued cost function C to be minimized:

$m_{qt} \in \{0, 1\}$: qubit q is measured at step t or not;

$c_{qt} \in \{0, 1\}$: qubit q is initialized in a causal cone of any measurement performed at $t' \leq t$ and qubit q has not been measured yet before step t; otherwise not

$C \in \mathbb{Z}^+$: maximum number of qubits used to execute the circuit.

The variable m_{qt} tracks when qubits are measured. The variable c_{qt} keeps track of which qubits are required to execute the circuit up to step t. The above descriptions of the variables m_{qt} and c_{qt} and the cost function C are enforced by the constraints in the model.

In words, the constraints are

(C.1) The number of qubits required to run the circuit is the maximum over the number of qubits required to execute the causal cones up to any given step.

(C.2) If qubit q is measured at step t, then all of the qubits in the causal cone of q must be active at or before step t.

(C.3) If a qubit is required to run the circuit up to step $t - 1$, it is either measured at step $t - 1$ or remains an active qubit at step t.

(C.4) If qubit q is measured at step t, it is initialized in at least one causal cone of a measurement performed at or before step t [47].

(C.5) If qubit q is measured at step t, then it is no longer needed to run the circuit subsequently [48].

(C.6) Each qubit is measured exactly once.

(C.7) Only one qubit is measured at each step.

The technical statement of each of these constraints can be found in Supplemental Material [33]. We emphasize that, without the constraints, the binary variables m_{qt} and c_{qt} have no meaning; the purpose of the constraints is to provide a set of equality, inequality, and conditional constraints on these variables that correspond to the physical problem of interest. For example, constraint (C.1) defines the cost function C in terms of a certain sum of c_{qt} variables that corresponds to the number of qubits you need to implement the circuit up to a certain step t. The cost function of interest is the number of qubits you need to run the circuit, which, in turn, is the size of the worst-case (maximum) number of qubits needed to run the circuit at any step.

We solve the model specified by minimization of the objective function subject to the constraints (C.1)–(C.7) using the CP-SAT solver provided by Google's open-source OR-Tools package [49]. This solver employs constraint programming techniques combined with SAT solving techniques to solve the problem exactly and is well suited for problems with binary variables [50]. It is also possible for the solver to return the best value of the objective found respecting the constraints within a given time limit and to seed the solver with a "hint" solving the constraints at a nonoptimal value of the objective. However, our numerical experiments indicate that for this particular problem the search space is sufficiently large that typically either (a) the solver produces an exact solution or (b) the solver cannot find a feasible solution or improve upon a hint in a reasonable amount of time, depending on the size of the problem instance. Using the output of the CP-SAT model, it is straightforward to extract the prescribed measurement order of the output qubits by examination of the variables m_{qt}: At step t (index t in the measurement order), measure the unique qubit q for which $m_{qt} = 1$.

B. A local greedy heuristic algorithm

Although the CP-SAT model furnishes an exact solution to the qubit-reuse compilation problem, it scales poorly with qubit number, as it involves $\mathcal{O}(N^2)$ variables and $\mathcal{O}(N^4)$ constraints with nontrivial structure, which becomes prohibitive in the $N \approx 100$–1000 qubit regime that will be a near-term proving ground for quantum computers. For practical purposes, it is, therefore, important to develop heuristic algorithms that can produce useful approximate solutions in a reasonable amount of time. In this section, we describe a local greedy algorithm that accomplishes this. Later, we show that this greedy heuristic is extremely effective (and often exact) for many circuit structures of near-term relevance.

Using the variables defined in the previous section, the local greedy algorithm proceeds as follows:

(1) First, measure the qubit q with the fewest number of input qubits in its causal cone C_q.

(2) Next, measure the qubit q' whose causal cone $C_{q'}$ adds the fewest *new* input qubits to C_q.

(3) Repeat step (2), successively choosing the next qubit q'' to measure as the qubit whose causal cone $C_{q''}$ adds the fewest new input qubits to the union of all causal cones of qubits measured so far.

A simple improvement of this heuristic algorithm that we also study employs a brute-force search over the possible choices of first qubit, at the expense of a multiplicative $\mathcal{O}(N)$ time complexity. That is, instead of first choosing the qubit with the fewest number of input qubits in its causal cone, consider all possible choices of initial qubit. For each possible initial qubit, use steps (2) and (3) above to iteratively construct the full measurement order. Then, select the ordering that required the fewest number of total qubits to execute the program. As numerical results substantiate, this small modification significantly improves the amount of compression achieved by qubit reuse with only a modest performance penalty. Though we do not explore it in this work, one could also perform a brute-force search over the first k qubits in the measurement order, which would multiply the run-time overhead by $\mathcal{O}(N^k)$.

C. Dual circuits

Consider any circuit $\mathcal{C}$ that involves quantum gates, state preparations, and measurements, where the state preparations and measurements can be interspersed arbitrarily within the circuit (qubits can also be traced out at the end of the circuit, which we regard as measurement with the result discarded). A circuit $\mathcal{C}^\star$ that can be written by replacing all state preparations in $\mathcal{C}$ with measurements and all measurements in $\mathcal{C}$ by state preparations, and then reading the circuit from right to left, is referred to as *dual* to $\mathcal{C}$. In this section, we describe an additional potential optimization that can be applied on top of any heuristic algorithm for causal-cone execution order, using the dual circuit to a given circuit. This optimization is optional, and the previously described algorithms are independent of its utilization. Namely, for any quantum circuit, one can apply any given algorithm for qubit reuse to both a circuit as well as its dual. In some cases, including the case in Fig. 3, it is possible for better compression of the dual circuit to be achieved, and, since dualization commutes with compression, this provides an ordering of the causal-cone execution that yields a better compression of the original circuit.

The compression of $\mathcal{C}$ via qubit reuse into $\mathcal{R}(\mathcal{C})$ can be understood as the following procedure: (i) move all state

$$\mathcal{R}(\mathcal{C}) \cong \mathcal{R}(\mathcal{C}^\star)^\star.$$

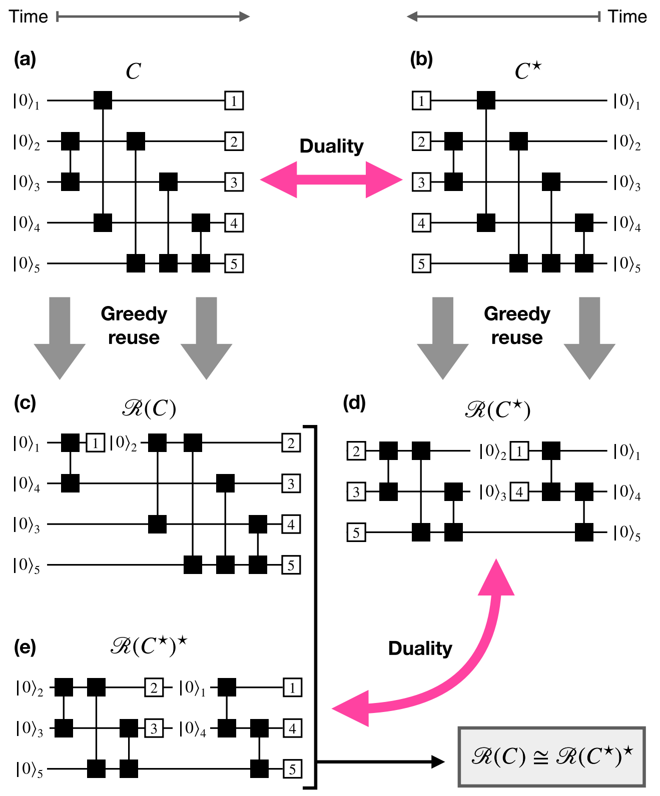


FIG. 3. (a) A five-qubit quantum circuit C. (b) The dual circuit $C^\star$ of circuit C, with time flowing in the opposite direction and with exchanged measurements and resets. (c) The compressed version $\mathcal{R}(C)$ of circuit C, after applying the greedy qubit-reuse heuristic, which involves four qubits. (d) The (greedily) compressed version $\mathcal{R}(C^\star)$ of circuit $C^\star$. (e) The dual circuit of $\mathcal{R}(C^\star)$, which involves only three qubits as opposed to the four in $\mathcal{R}(C)$ [see (c)].

preparation operations as late in time as possible, (ii) move all measurement operations as early in time as possible, and (iii) for each measurement, optionally connect the associated qubit wire to a state preparation occurring later in time, thus reducing the number of required qubits. Note that the only rule one must obey in this procedure is to never move two operations past each other in time if they have shared support. Performing this compression from $\mathcal{C}$ to $\mathcal{R}(\mathcal{C})$ using the greedy heuristic is shown as Figs. 3(a)–3(c). We can also compress the dual circuit $\mathcal{C}^\star$ into $\mathcal{R}(\mathcal{C}^\star)$ [Figs. 3(b)–3(d)], at which point we are free to replace all measurements and state preparations in $\mathcal{R}(\mathcal{C}^\star)$ according to the replacements mapping $\mathcal{C}^\star \to \mathcal{C}$ in order to generate a new circuit $\mathcal{R}(\mathcal{C}^\star)^\star$ [Fig. 3(e)]. Note that $\mathcal{R}(\mathcal{C}^\star)^\star$ *could have been* obtained directly from $\mathcal{C}$ by obeying the compression rules, since respecting time ordering allows all the same rearrangements and rewiring for either a circuit or its dual. As a result, we have the general relation

$$\mathcal{R}(\mathcal{C}) \cong \mathcal{R}(\mathcal{C}^\star)^\star.$$

Note that this equivalence implies that the optimally compressed version of $\mathcal{C}$ must always contain the same number of qubits as the optimally compressed version of $\mathcal{C}^\star$. However, nonoptimal techniques, such as the greedy heuristic described above, need not necessarily return the same level of compression on a circuit and its dual, as evidenced by comparing Figs. 3(c) and 3(d). This implies an immediate optimization for any heuristic: One should always apply it to both the circuit in question and its dual, and if its dual is more compressible, one should accept $\mathcal{R}(\mathcal{C}^\star)^\star$ [Fig. 3(e)] over the less efficient direct compression $\mathcal{R}(\mathcal{C})$ [Fig. 3(c)].

IV. ANALYTIC RESULTS FOR QUBIT-REUSE COMPILATION

For certain classes of circuits, it is possible to analytically write down measurement orders that achieve significant compression of the circuit by exploiting its underlying structure and symmetries. We emphasize that these measurement orders can, in principle, achieve better compression than the approximate numerical algorithms discussed above in certain cases. For several of the circuits discussed below, the analytical measurement order is identical to that produced by the local greedy heuristic and can be shown to be optimal. This demonstrates that in many cases of practical interest the greedy heuristic is exact and quantifies the scaling (with qubit number) of the compression it achieves.

A. Local brickwork circuits in 1D

The first class of circuits we consider are one-dimensional brickwork circuits, composed of k alternating even-odd layers of two-qubit gates acting with periodic boundary conditions on N qubits, shown in Fig. 4. These circuits appear naturally when simulating local 1D Hamiltonians, and qubit reuse in such linear circuits has been explored previously in the context of quantum matrix product states [37,41,51–53] and their time evolution [12].

For this particular circuit, we show that the optimal measurement order is the simple linear order $m_{qt} = \delta_{q,(q_0+t) \bmod N}$ with an arbitrary initial qubit q_0. After k layers (where here each "layer" includes successive application of both a row of "even" gates and a row of "odd" gates), each qubit's causal cone is of size $4k$, as illustrated in Fig. 4 for $k = 3$. For example, the causal cone of qubit $q = 2k$ includes input qubits $1, \ldots, 4k$. For the linear measurement order with $q_0 = 2k$, we first measure qubits $q = 2k$ and $q = 2k + 1$, which participate in the same final gate and have the same past causal cone consisting of the $4k$ qubits $q = 1, \ldots, 4k$. As already mentioned, measuring *any single output qubit* requires $4k$ qubits (to initialize the past causal cone of any single output), so the circuit cannot be compressed to less than $4k$ qubits. It is now easy to show that a linear measurement order uses exactly $4k$ qubits, implying optimality: After measurement of $q = 2k$ and $q = 2k + 1$, we can reset and reuse these two qubits as input qubits $4k + 1$ and $4k + 2$. Then, we are free to execute the green gates in Fig. 4 and then measure qubits $q = 2k + 2$ and $q = 2k + 3$ at the output. We then reset and reuse them as input qubits $4k + 3$ and $4k + 4$, execute the purple gates, and proceed this way until we measure qubits $q = 2k - 2$ and $2k - 1$, never needing additional qubits beyond the initial $4k$ used to measure the first two qubits. Note that we assume that $4k < N$, since otherwise

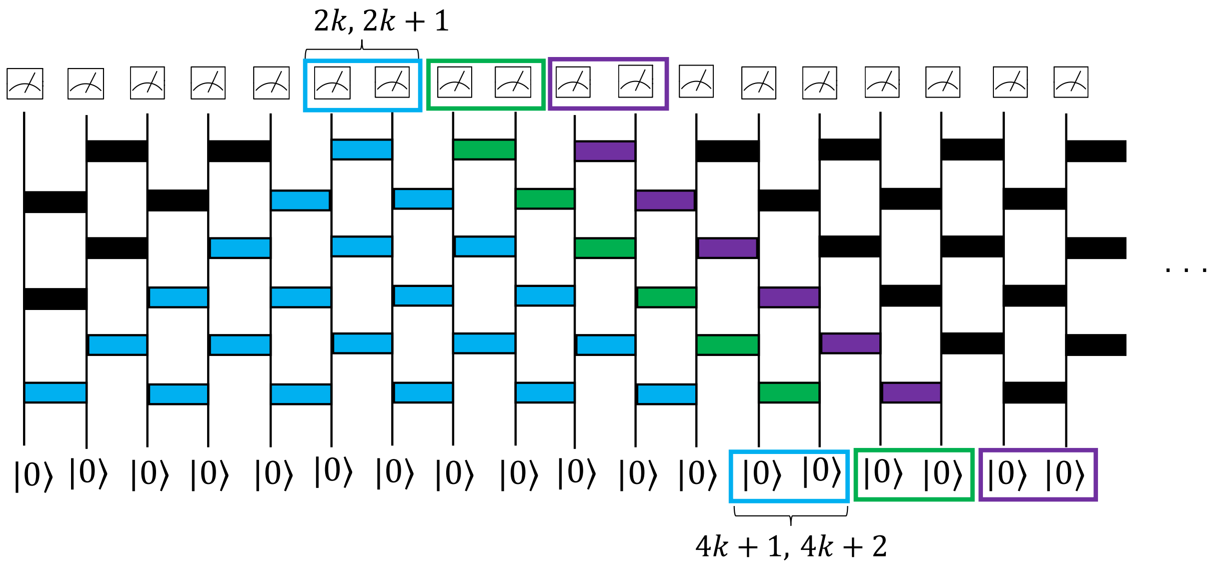


FIG. 4. Qubit reuse in one-dimensional brickwork circuit with $k = 3$ layers, where time flows upward by convention. The output qubits labeled $2k$ and $2k + 1$ (blue) are measured first and reused as input qubits $4k + 1$ and $4k + 2$. Those input qubits are subsequently used to implement the green causal cone and measure the green output qubits $2k + 2$ and $2k + 3$, which are reused as the purple input qubits, and so on.

each qubit's causal cone spans the entire chain and it is not possible to reuse qubits. We assume periodicity in this section to enforce translation symmetry, which allows the initial qubit q_0 to be chosen arbitrarily. However, there is no substantial change in the result by relaxing this symmetry; the initial qubit in this case must be chosen at one edge of the linear chain.

In general, for the one-dimensional brickwork circuits, there are many degenerate measurement orders that produce the same result of $4k$ qubits required to execute the compressed circuit. This is because, for any pair of qubits i and $i+1$ that are measured, the next causal cone can be chosen to measure $i+2$ and $i+3$ or $i-1$ and $i-2$, proceeding to the right or left down the linear chain, respectively. We expect that this choice can have an impact on the memory errors incurred on qubits which have not yet been measured. For example, proceeding linearly down the chain results in qubits at the beginning remaining live for the entire duration of the circuit, whereas alternating back and forth measuring qubit pairs on either side of the initial pair avoids this but slightly increases the average active time of each qubit. Examining the consequences of such trade-offs is a subject of future study.

B. Local brickwork circuits in 2D

It is instructive to understand how the compression of local brickwork circuits generalizes to interactions in higher dimensions. We now examine two-dimensional local brickwork circuits, that is, brickwork circuits where gates are applied between adjacent qubits laid out on a square grid. The analysis in two dimensions demonstrates a general result that $\mathcal{O}(N^{d-1})$ qubits are required to execute a circuit that is originally defined on $\mathcal{O}(N^d)$ qubits in d dimensions, which becomes substantially harder to visualize and analyze in $d > 2$.

Consider a two-dimensional brickwork circuit composed of k layers of gates staggered horizontally and vertically with periodic boundary conditions on an $N \times N$ grid of qubits as shown in Fig. 5. Here, as in the one-dimensional case, a "layer" refers to the $2d = 4$ total rows of odd + even gates in each dimension. For this circuit, the causal cone of each output qubit can be interpreted as a pyramid containing $4k \times 4k = 16k^2$ input qubits (Fig. 5). Here, we describe a particular measurement order that performs the optimal qubit reuse for the 2D brickwork circuit. For simplicity, we assume that N is even and that $4k < N$; we explain how to generalize the result to nonsquare lattices without this bound on k at the end.

Again, due to translational symmetry, the first qubit q_0 to be measured is arbitrary. Furthermore, it is straightforward to check that the qubits on the lattice can be partitioned into groups of four that all share the same causal cone (Fig. 6). Consequently, a greedy approach again is optimal, since adjacent qubits either share identical causal cones or their causal cones are simply geometrically shifted along the

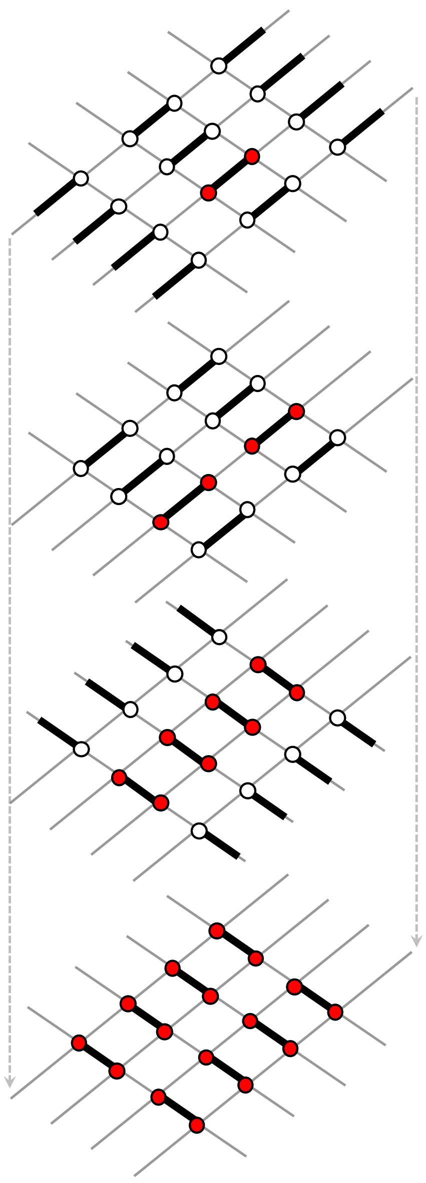


FIG. 5. The causal cone of a single qubit after $k = 1$ layer of local gates is applied in two dimensions. Read with time flowing from top to bottom, the red coloring of the qubits indicates how causal influence propagates between qubits after each row or column of two-qubit gates is applied. After k layers, the causal cone of the initial qubit expands to a size of $16k^2$ at the bottom.

lattice and heavily overlapping. The most straightforward strategy to count the number of qubits required to implement the greedy algorithm in this case is to tile the lattice column by column, by measuring blocks of four adjacent input qubits. After completing a column, one measures a block of four output qubits in the adjacent column and proceeds to tile that column. This process repeats until the entire lattice has been measured; see Fig. 6 for a clarifying depiction.

We now proceed to count the number of qubits required to implement the brickwork circuit on the 2D lattice according to the process detailed above. As discussed,

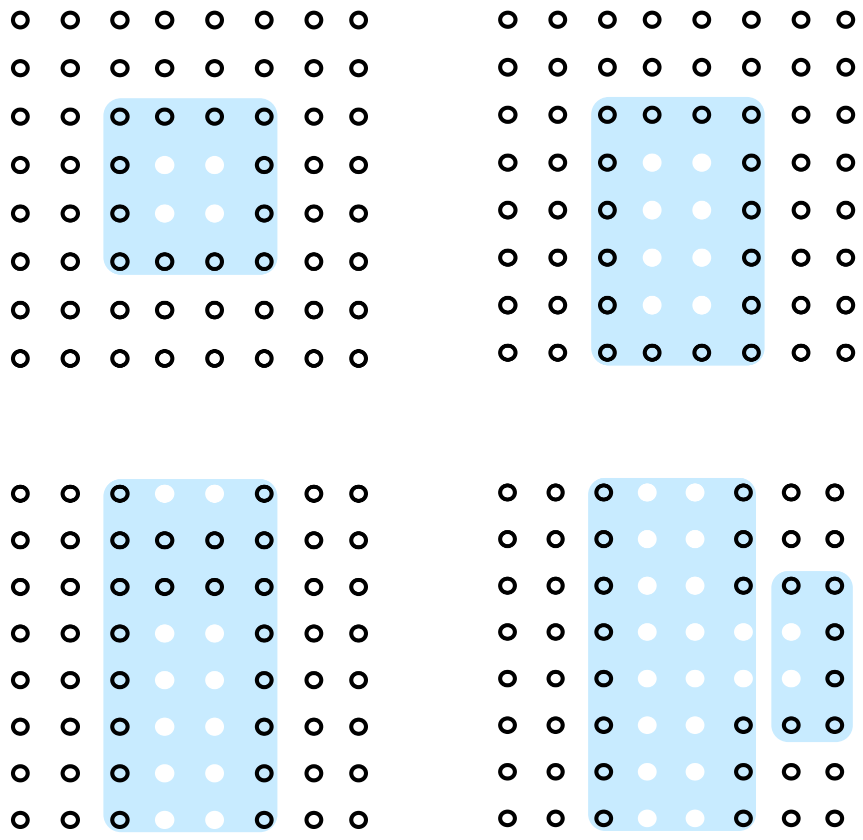


FIG. 6. Top left: the 16 input qubits in the shared causal cone (light blue) of the four white qubits, for $k = 1$ layer of gates. Top right: The greedy algorithm dictates that the next four output qubits to be measured are the four white qubits directly below the first four. Their causal cone extends the required set of input qubits downward by two (light blue). Bottom left: The four qubits in the middle of the last causal cone in a complete column can be measured for free without any additional qubits. Bottom right: Extending to the next column over requires eight new qubits (light blue, right-hand side), but eight qubits are available for reuse from the previous column.

the first causal cone requires $16k^2$ input qubits to measure the first four output qubits (Fig. 6, top left). The next causal cone reclaims another four output qubits and requires an additional $8k$ inputs, of which four are reused from the previous step (Fig. 6, top right). This process repeats until an entire column has been tiled by causal cones, and the last sets of four output qubits in the column are obtained for free, since their causal cones fully overlap with causal cones of previously measured qubits (Fig. 6, bottom left). After the very first causal cone, one requires $N/2 - 2k$ additional causal cones to tile a single column before the subsequent causal cones are completely overlapping. Consequently, the last $2k - 1$ causal cones each reclaim four qubits for free, which yields $4(2k - 1) + 4 = 8k$ qubits available for reuse in the next column after completing a single column, where the extra 4 comes from the last causal cone before overlaps occur.

In total, therefore, measuring the $2N$ output qubits corresponding to a single column of causal cones requires the following number of simultaneously active input qubits:

$$16k^2 + (8k - 4)(N/2 - 2k) = (4k - 2)N + 8k. \quad (1)$$

When we proceed to the next column and repeat this process, the next causal cone requires $8k$ input qubits (Fig. 6, bottom right). This conveniently exactly matches the number of qubits reclaimed for reuse at the end of the previous column, so no new qubits are required. Furthermore, this causal cone reclaims four output qubits, and each subsequent causal cone in the new column requires only four additional input qubits, not $8k$ (see Fig. 6, bottom right—the reason is that the causal cones now overlap on two sides). Therefore, the entire column can be measured without use of any additional input qubits, as can all subsequent columns. The last few columns overlap, but this can only reduce the required overhead.

In summary, therefore, the number of qubits required to execute the circuit containing only the causal cones that tile the first column is the same as the number of qubits required to execute the entire circuit. This number is, from Eq. (1), $(4k - 2)N + 8k$.

In the slightly more general case that the lattice is not square and is instead $N_x \times N_y$ with both N_x, N_y even, the same argument applies where N in Eq. (1) should be replaced with the dimension that you first choose to tile with causal cones. To minimize the overhead, therefore, you should choose the shorter dimension first, replacing N with $\min(N_x, N_y)$ above.

In the case that $4k > \min(N_x, N_y)$, the counting arguments above are slightly modified, since the size of each causal cone already spans one or more dimensions. Performing the same counting procedure taking this into account yields the following general formula for the compiled qubit number:

$$\begin{array}{ll} (4k-2)\ \min(N_x, N_y) + 8k, & 4k < \min(N_x, N_y) \\ N_x N_y, & 4k > \max(N_x, N_y) \\ 4k \min(N_x, N_y), & \text{otherwise.} \end{array}$$

C. Matrix product state preparation

We continue by studying circuits implementing quantum tensor networks, which provide broadly useful *Ansätze* for representing various types of entangled quantum states. The simplest example of applying qubit reuse to a quantum tensor network circuit occurs in the context of matrix product state (MPS) preparation.

As shown in Ref. [54], by restructuring an N-site MPS into a suitable canonical form, it can always be interpreted as a quantum circuit in which a register with Hilbert space dimension χ (representing the bond space of the MPS) sequentially interacts with N qubits (representing the physical legs of the MPS). Therefore, an MPS can always be constructed using $N + \lceil \log_2 \chi \rceil$ qubits, as displayed in Fig. 7. As discussed in Refs. [39–41], owing to the sequential nature of the interactions between the ancilla register and the N physical qubits, the full output of an MPS can be sampled with only $1 + \lceil \log_2 \chi \rceil$ qubits by resetting and reusing a single physical qubit after each local measurement.

D. Tree tensor networks

A more complex generalization of the one-dimensional entanglement generated by an MPS is provided by the tree

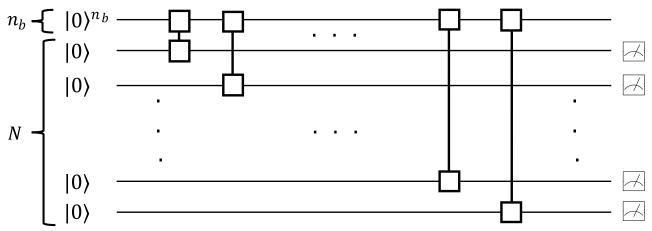


FIG. 7. The MPS preparation circuit on N qubits, using an MPS with $n_b = \lceil \log_2 \chi \rceil$ bond qubits, where χ is the bond dimension. Since the state preparation circuit consists of sequential gates applied to each of the physical qubits, the circuit can be rewritten using only $1 + n_b$ qubits by sequentially measuring and resetting each physical qubit.

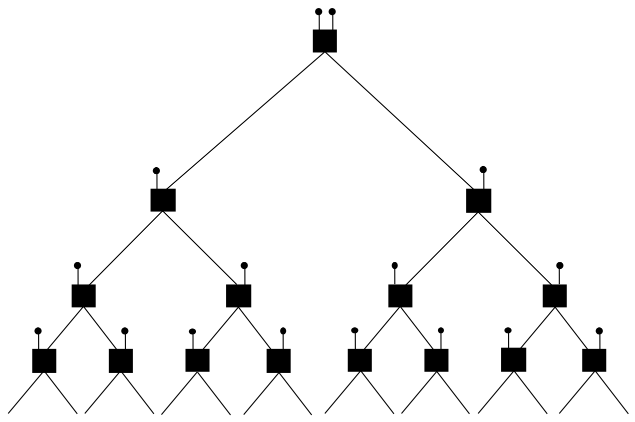


FIG. 8. A depth-4 TTN generating a state on 16 output qubits, read top to bottom. Each of the two legs of the topmost tensor connects to a depth-3 TTN as referenced in the text. Dots correspond to qubits initialized to $|0\rangle$ and squares to unitary gates.

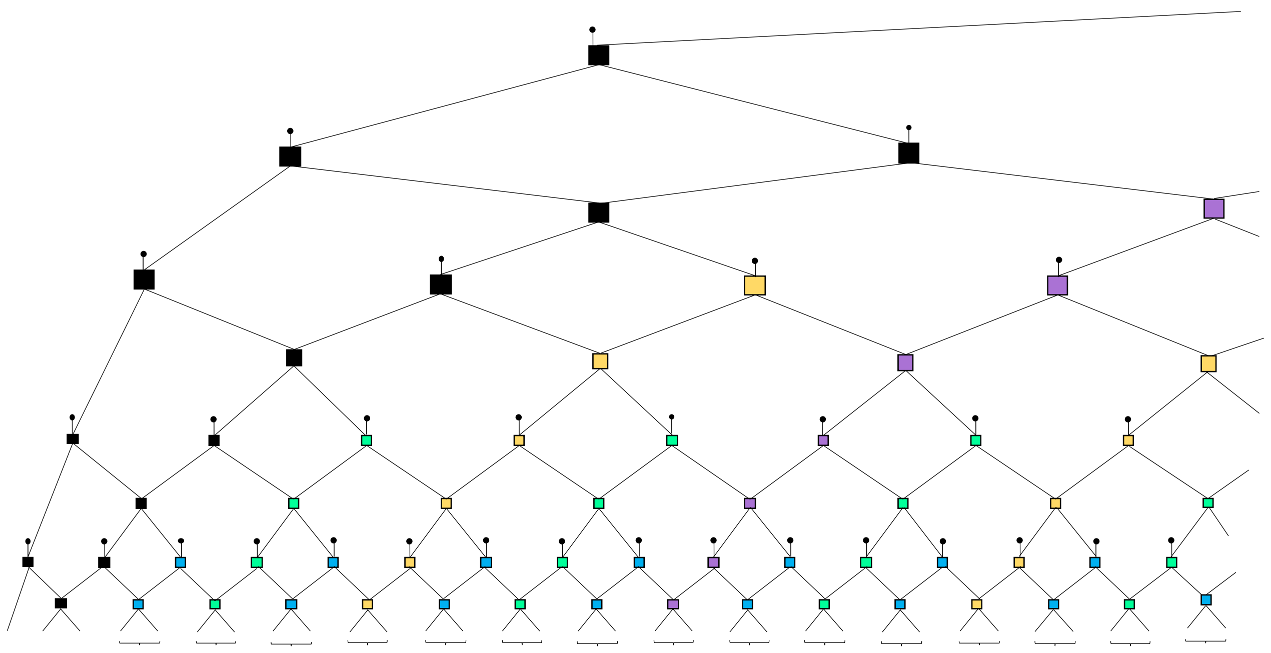
tensor network (TTN), defined by placing isometries on the nodes of a binary tree, shown in Fig. 8. A depth-D binary TTN generates a state of 2^D qubits and can be written as a circuit by embedding the isometries into unitaries. To determine how many qubits are required to sample the output of a general binary TTN, we suppose that N qubits are required to sample the output of a depth D TTN and proceed inductively, as suggested in Ref. [39]. A depth $D+1$ TTN is built by introducing a new top tensor and attaching a depth D TTN to each of its two outputs. Assuming for simplicity a bond-dimension of 2 (though this argument can extended to higher bond dimension), the output of the full depth $D+1$ TTN can be sampled by first inputting two qubits into the unitary representing the top tensor, after which point one qubit is idly waiting at the top of both depth D TTNs sitting immediately below it. By assumption, the output of one depth D TTN can be sampled with $N-1$ more qubits, since one qubit is already provided by the output of the top tensor. After sampling one of the two depth D TTNs, the same $N-1$ qubits can be reused to sample the output of the second. Thus, the full depth $D+1$ TTN can be sampled using $2 + N - 1 = N + 1$ qubits. Since a $D = 1$ TTN can be sampled using two qubits, by induction it follows that the output of a depth D TTN can be sampled using $D + 1$ qubits.

Any TTN circuit can also be viewed as a coarse-graining procedure that converts an N-qubit input into a small number of qubits at the top of the network; in this mode of operation, TTNs form natural classifiers [39]. For classical (product state) input data, the TTN classifier circuit is dual to the TTN generator circuit and, thus, can also be executed using $D + 1$ qubits.

E. Multiscale entanglement renormalization *Ansatz*

The MERA extends the treelike architecture of a TTN by adding unitary disentanglers between neighboring branches of the tree. Read from top to bottom as shown in Fig. 9 and interpreting the isometries as unitaries acting on additional initialized input qubits, one can view a MERA as a circuit that takes in N qubits initialized in $|0\rangle$ and outputs the N-qubit state at the bottom of the MERA. It is known to be a compact and efficiently contractible TN ansatz for critical states of matter [36].

For a MERA with open boundary conditions, the greedy measurement order is optimal for compressing the circuit. To see this, begin by executing the past causal cone of the leftmost qubit, which requires only $D+1$ input qubits and returns one to the "reuse queue" of qubits that have already been measured and are available for reset and reuse. The next step in a greedy approach is to implement the non-evaluated portion of the causal cone of the next two qubits from the left, which requires only $D-1$ qubits, one of which is borrowed from the reuse queue. In this way, the first three outputs can be measured using $2D-1$ qubits, leaving two qubits in the reuse queue. It can be checked that any other choice of the first measured qubit would have required at least $2D-1$ qubits. Sequential qubits are measured in pairs going down the line, which we index starting with the fourth and fifth qubit being P_1, and we denote the queue size at prior to measuring P_1 by Q_1. If one proceeds greedily from left to right evaluating the causal cones of pairs $P_{n=2,3,\ldots}$, the minimal required set of isometries for each pair is shown in Fig. 9 using color coding. We denote the size of the set of isometries required to measure P_n by S_n, which sets the number of qubits required to measure that pair. The sequence S is known as the ruler function [55], and each value S_n can be interpreted as the positions of the most significant bit (reading the least significant as the first) that gets flipped when incrementing from the binary representation of $n-1$ to that of n. Denote by Δ_n the number of qubits added to the reuse queue by measuring all pairs up to the one corresponding to the first instance of $n+1$ in the sequence S. For example, Δ_1 is obtained by executing the causal cone of pair P_1, which requires one qubit, and then returning P_1 to the reuse queue, such that $\Delta_1 = -1 + 2 = 1$. One can show that Δ_n obeys the recursion relation $\Delta_n = 2(\Delta_n + 1) - n$, which is satisfied (with proper boundary condition $\Delta_1 = 1$) by the



Pair label	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Needed (S_n)	1	2	1	3	1	2	1	4	1	2	1	3	1	2	1

Queue:

Q_n	2	3	3	4	3	4	4	5	3	4	4	5	4	5	5

FIG. 9. A section of an open boundary MERA circuit, with time going from top to bottom. Dots correspond to qubits initialized to $|0\rangle$ and squares to unitary gates (gates with an incoming $|0\rangle$ are isometries, while the others are disentanglers). Different colors of gates correspond to different causal cones of the output qubits at the bottom of the diagram. Output qubits come in pairs, which we label from left to right, with the fourth and fifth qubits from the left corresponding to pair P_1. The S_n values indicate how many new isometries (and, therefore, new $|0\rangle$ qubits) need to be added to produce the causal cone of the qubits in pair P_n. The Q_n values specify how many qubits are available for reuse from previously prepared causal cones.

solution $\Delta_n = n$. Given that $Q_1 = 2$ qubits are in the queue prior to the evaluation of P_1's causal cone, this ensures that there will always be $n+1$ qubits in the reuse queue at the first occurrence of n in S (e.g., $Q_{2^{j-1}} = j+1$, black arrows in the figure), such that we always have enough qubits in the reuse queue to execute the next causal cone without adding new qubits. As we approach the right boundary of the MERA, there are times when fewer qubits are needed than suggested by this argument, but that only works in our favor. Thus, we can sample the full output with only the $2D-1$ qubits required to measure the first three qubits.

F. Quantum convolutional neural networks (QCNN)

By embedding each isometry into a unitary and either postselecting on or discarding the padded outputs, any MERA can be read from bottom to top as a circuit that coarse grains an N qubit state, outputting a small number of qubits at the top. This view lends itself naturally to the problem of classifying quantum data and forms a natural quantum generalization of the convolutional neural network [56]. Note that the QCNN, as originally defined, involves classical feed-forward on measurements of padded qubits exiting the isometries; this does not change the causal structure of the MERA and can be ignored for our purposes. More generally, the conditional probabilities for measurement outcomes at the top of the QCNN do not change if we replace the measurement conditioned gates with quantum controlled gates, compile those gates into the unitary embedding of the isometry, and trace out the control qubit(s). Assuming product-state inputs, this construction of a QCNN is exactly the dual circuit of the corresponding MERA and, thus, can also be executed using $2D-1$ qubits in the worst case. However, we point out that the practical use of the QCNN to classify quantum data typically assumes the initial existence of a global many-body quantum state defined on $N = 2^D$ qubits rather than product-state inputs; in this case, the extent of qubit compression that can be accomplished depends on the entanglement structure of the input state.

G. Bernstein-Vazirani algorithm

The Bernstein-Vazirani (BV) algorithm solves the following problem: Given a function $f\colon\{0,1\}^N \to \{0,1\}$ which is defined by a *hidden bit string* s by $f(x) = s\cdot x$ (mod 2), find the bit string s. The algorithm relies on the ability to query an oracle for the function f, so that classically N function calls (on the N unique bit strings which are zero everywhere except for one register) are required to learn f. In the quantum case, the physical implementation of the unitary operator encoding the oracle typically requires knowledge of the hidden bit string in advance, in which case the BV algorithm implemented as such is not a practical demonstration of quantum supremacy but rather a theoretical demonstration that given access to a true oracle only one query suffices in the quantum case (see Fig. 10).

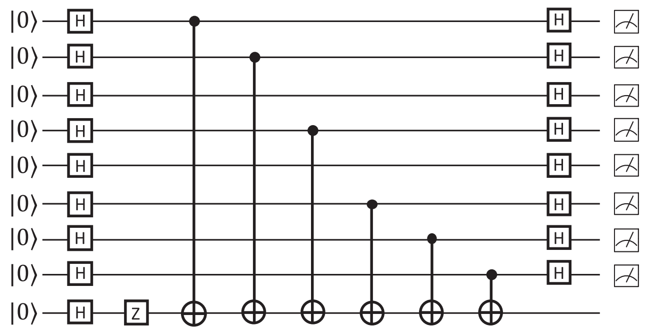


FIG. 10. The Bernstein-Vazirani circuit on eight qubits, with the unitary oracle explicitly decomposed for the example of the hidden bit string 11101011. Note that, following the standard convention for encoding measurement bit strings, the bottommost qubit represents the leftmost bit in the string.

It is well known [11,57] that the BV algorithm can be implemented using only two qubits using mid-circuit measurement and reuse. In this section, we simply state how this known result follows from the causal-cone framework presented in this paper. Namely, none of the register qubits interact in the BV algorithm in the typical realization of the unitary oracle. Consequently, the causal cone of every register qubit contains input qubits that consist only of itself and of the shared ancilla qubit. Consequently, the BV algorithm defined on any number of qubits can always be executed using only two physical qubits. One physical qubit is used as the ancilla qubit and persists for the duration of the computation, while the other physical qubit represents a single register qubit. After the unitary oracle function is applied to the two qubits, the register qubit can subsequently be measured and reused as the next register until all registers are exhausted.

The analytic results of this section are summarized in Table I. The general effect of qubit-reuse compilation can be described as a sort of generalized holographic quantum dynamics for local physics and tensor network circuits, in

TABLE I. The minimum number of qubits required to execute certain structured quantum circuits after qubit-reuse compilation. For the k-layer brickwork circuits, we assume that $4k$ is smaller than all dimensions.

Algorithm	Original qubit no.	Compiled qubit no.
k-layer brickwork 1D	N	$4k$
k-layer brickwork 2D	N^2	$(4k-2)N + 8k$
MPS, bond dim. χ [41]	$N + \lceil \log_2 \chi \rceil$	$1 + \lceil \log_2 \chi \rceil$
Depth-D binary TTN	2^D	$D+1$
Depth-D binary MERA	2^D	$2D-1$
Depth-D binary QCNN	2^D	$2D-1$
Bernstein-Vazirani	N	2

that the scaling of the compiled qubit number is parametrically less than that of the original circuit.

V. NUMERICAL EXPERIMENTS AND BENCHMARKING

We expect many practical examples of circuits on which qubit reuse may be helpful to lie outside of the analytically tractable categories discussed in the previous section. To this end, in this section, we numerically study the amount of compression that can be achieved in the QAOA. We study the algorithmic time complexity (run-time) and solution quality (compiled qubit number) for QAOA MaxCut circuits at various depths on random unweighted three-regular (U3R) graphs. MaxCut on random U3R graphs is a standard benchmarking case for QAOA on near-term quantum devices (see Refs. [58–60] for some recent investigations of QAOA on hardware), since the number of gates matches the number of edges in the graph and scales only linearly with the number of vertices. In general, MaxCut QAOA is an ideal near-term algorithm for qubit reuse, as it is a shallow, wide circuit with fairly sparse gate connectivity, so the typical size of causal cones can be expected to be small relative to the size of the original set of qubits.

The QAOA unitary [61] takes the form of alternating applications of a mixing unitary $U_B(\beta_n) = e^{-i\beta_n H_B}$ and a phase-splitting cost unitary $U_C(\gamma_n) = e^{-i\gamma_n H_C}$:

$$U(\vec{\beta},\vec{\gamma}) = \prod_{n=1}^{p} U_B(\beta_n) U_C(\gamma_n), \tag{2}$$

where $H_B = \sum_i X_i$ and H_C encodes the cost Hamiltonian of the combinatorial problem. The unitary depends on $2p$ parameters $\beta_1, \ldots, \beta_p$ and $\gamma_1, \ldots, \gamma_p$ and the product is conventionally ordered so that the terms with β_1 and γ_1 are applied first. For the MaxCut problem, the cost Hamiltonian H_C has a standard encoding as a quadratic unconstrained binary optimization (QUBO) Hamiltonian, taking the form

$$H_C = \frac{1}{2} \sum_{(i,j)\in E} w_{ij}(1 - Z_i Z_j), \tag{3}$$

where the coefficient w_{ij} is the weight of edge (i,j). Since we take the graph to be unweighted in all cases, all coefficients $w_{ij} = 1$.

The QAOA protocol finds the parameters $\vec{\beta}$ and $\vec{\gamma}$ by variational minimization. A classical optimization algorithm is used to search for parameters $\vec{\beta}^*$ and $\vec{\gamma}^*$ that minimize

$$\langle H_C \rangle = \langle \psi_0 | U(\vec{\beta},\vec{\gamma})^\dagger H_C U(\vec{\beta},\vec{\gamma}) | \psi_0 \rangle \tag{4}$$

with $|\psi_0\rangle = |+\rangle^{\otimes N}$ by convention. For the purposes of studying qubit reuse, the circuits we consider evaluate $U(\vec{\beta},\vec{\gamma})|+\rangle^{\otimes N}$ and measure all qubits at the end.

Throughout this section, we are primarily concerned with comparing three different algorithms for compiling circuits with qubit reuse: the local greedy algorithm, the local greedy algorithm with an additional brute-force search over the first measured qubit, and the exact CP-SAT solution where it is viable. Depending on the circuit structure and the amount of compression that is possible, it becomes impractical to execute the CP-SAT model at around $N \sim 30$–50 qubits in the original circuit, so we also compare only the heuristic algorithms at significantly larger N. The time benchmarking presented in this section is evaluated on a small personal laptop, effectively as a model of the realistic computation time that can be expected by a typical user desiring to use qubit-reuse compilation for algorithm development; it is possible that high-performance computing techniques can slightly extend the regime in which the algorithms presented herein are feasible.

We begin by examining the performance of the three algorithms for $p = 1$ (Figs. 11 and 12) and $p = 2$ (Figs. 13 and 14) MaxCut QAOA. For each fixed qubit number, we evaluate the qubit-reuse algorithms on 100 random U3R graphs generated using the NetworkX package [62].

As expected, it is clear from Fig. 11 that the CP-SAT model is superpolynomially scaling and rapidly becomes impractical for circuits of more than a few dozen qubits. Furthermore, from Fig. 12, it is apparent that the performance of the heuristics is only slightly worse than optimal at these small qubit numbers, especially when the additional brute-force search is employed on the first qubit measured.

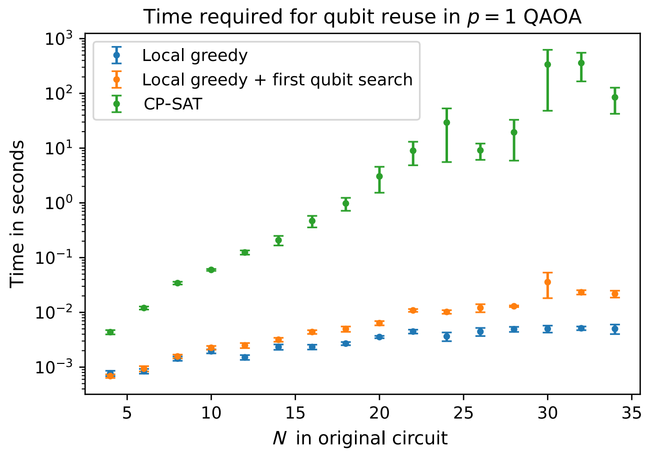


FIG. 11. The time required in seconds to execute each qubit-reuse algorithm as a function of the number of qubits in the original circuit, averaged over 100 instances of random U3R graph MaxCut $p = 1$ QAOA circuits. The plotted uncertainties correspond to the error on the mean in this and all below plots.

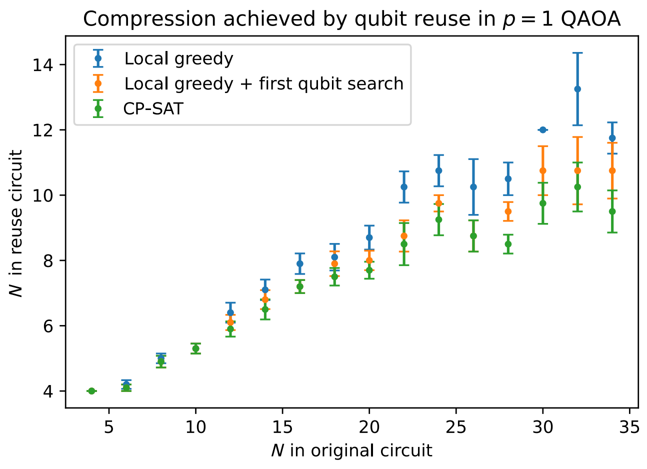


FIG. 12. The compiled qubit number as a function of the number of qubits in the original circuit, averaged over 100 instances of random U3R graph MaxCut $p = 1$ QAOA circuits.

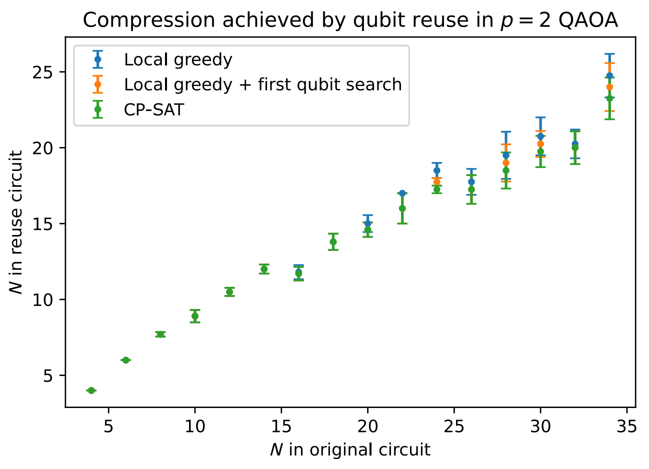


FIG. 14. The compiled qubit number as a function of the number of qubits in the original circuit, averaged over 100 instances of random U3R graph MaxCut $p = 2$ QAOA circuits.

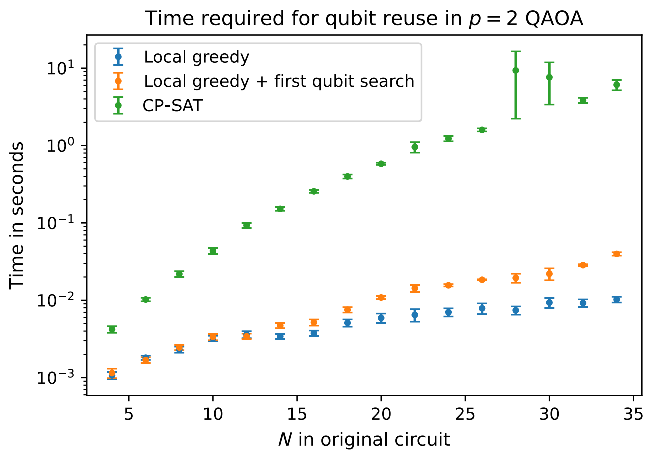


FIG. 13. The time required in seconds to execute each qubit-reuse algorithm as a function of the number of qubits in the original circuit, averaged over 100 instances of random U3R graph MaxCut $p = 2$ QAOA circuits.

In Figs. 13 and 14, one can see that, for $p = 2$, although the run-time of the heuristic algorithms slightly increases from $p = 1$, the CP-SAT model runs about an entire order of magnitude more quickly. This can be explained by the behavior of the classical optimization techniques underlying the CP-SAT solver, which iteratively rule out sections of the solution search space. Because the causal cones in $p = 2$ QAOA are larger than that of $p = 1$ QAOA, the CP-SAT model is more highly constrained [see (C.2)] in the $p = 2$ case, allowing more of the search space to be ruled out faster. We further see that all three algorithms perform nearly comparably and achieve significantly less compression of the original circuit in the $p = 2$ case.

The CP-SAT model also permits seeding with an initial "hint" solution that satisfies the provided constraints, attempting to improve upon this solution. This suggests an obvious method to improve upon the heuristic solutions at larger N by handing off the heuristic solution as input to the CP-SAT model. In Fig. 15, we compare the amount of compression achieved in $p = 1$ MaxCut QAOA by the local greedy algorithm to that produced by the CP-SAT model both with and without the hint. To realistically bound the amount of time available for attempting qubit-reuse compilation, the CP-SAT model is limited to 10 min to attempt to find a solution both with and without the hint.

Figure 15 displays several different regimes owing to the time limit placed on the CP-SAT model. At sufficiently small N, the CP-SAT model solves the qubit-reuse problem exactly in under 10 min, so the hint provides no added value. Around $N \sim 40$ there is a transition, where 10 min is no longer sufficient for CP-SAT to optimally solve the qubit-reuse problem. In this case, seeding with the hint may produce a better solution than either the heuristic or

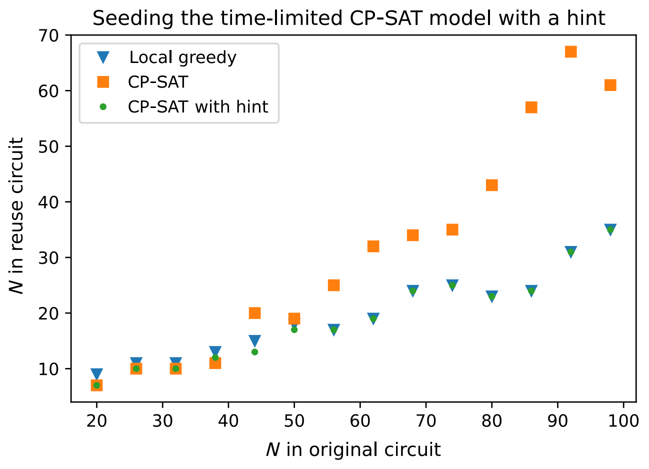


FIG. 15. The compiled qubit number as a function of the number of qubits in the original circuit, for one instance of random U3R graph MaxCut $p = 1$ QAOA at each point.

CP-SAT solutions without the hint, though the point at $N = 38$ shows that it is also possible to get stuck in a local optimum when starting from the hint that does not beat the unseeded model. Finally, at $N \gtrsim 60$, 10 min is clearly insufficient for the CP-SAT model to find an optimal solution, since it is beaten by the local greedy algorithm, and it is also insufficient for the CP-SAT model to improve on the heuristic solution.

Looking ahead, it is of interest to determine how well qubit-reuse compilation performs when algorithms are scaled to several hundred or several thousand qubits in size. To this end, we study the performance of the two heuristic algorithms outside of the regime in which the CP-SAT model is practical. The results (Figs. 16 and 17) show that enhancing the local greedy algorithm with the brute-force search is viable out to at least $N \sim 500$, where the qubit-reuse compilation takes about a minute for the improved local greedy algorithm and fractions of a second for the local greedy algorithm without this improvement. Furthermore, the brute-force search over the first qubit results in a 13% reduction on average in the number of qubits required to execute the circuit compared to the greedy algorithm with no brute-force search, requiring only about one-quarter of the original number of qubits on average. The upshot is that, for near-term applications with dozens or at most hundreds of qubits, it is essentially always preferable to include this improvement when executing the local greedy algorithm.

It is prudent to test the limits of the local greedy algorithm (with no brute-force search improvement) in terms of both time expenditure and how well it scales with deeper circuits. To this end, in Figs. 18 and 19, we display

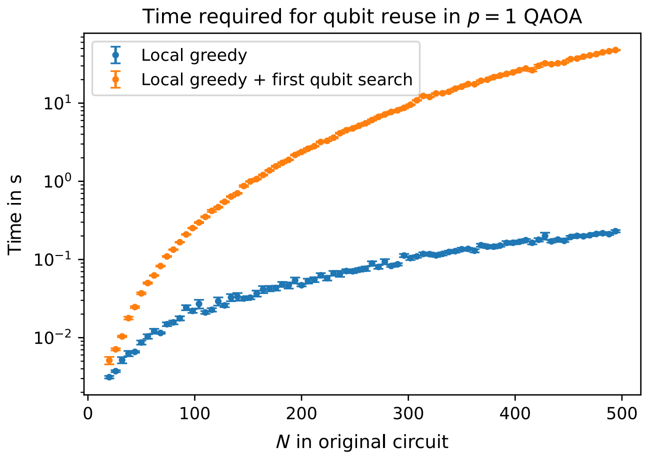


FIG. 16. The time required in seconds to execute each qubit-reuse algorithm as a function of the number of qubits in the original circuit for random U3R graph MaxCut $p = 1$ QAOA circuits. The data are averaged over 100 instances for $N < 100$ and 20 instances for $N > 100$.

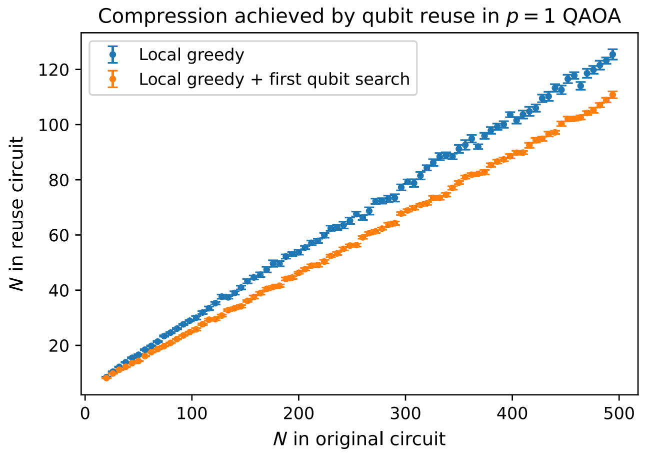


FIG. 17. The compiled qubit number as a function of the number of qubits in the original circuit for random U3R graph MaxCut $p = 1$ QAOA circuits. The data are averaged over 100 instances for $N < 100$ and 20 instances for $N > 100$.

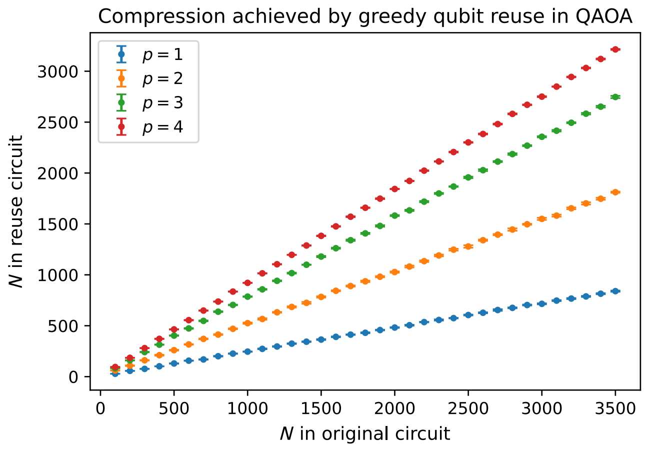


FIG. 18. The compiled qubit number as a function of the number of qubits in the original circuit, averaged over ten instances of random U3R graph MaxCut QAOA circuits at different depths p.

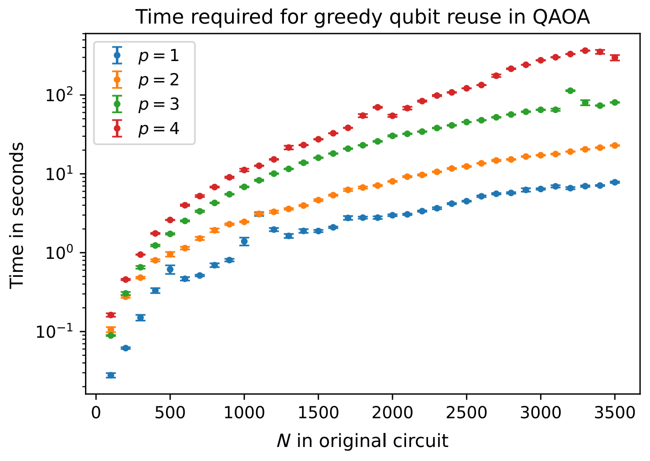


FIG. 19. The time required in seconds to execute the local greedy algorithm as a function of the number of qubits in the original circuit, averaged over ten instances of random U3R graph MaxCut QAOA circuits at different depths p.

the performance and efficiency of the local greedy algorithm out to several thousand qubits in the original circuit. The results show that the local greedy algorithm scales quite well, running efficiently in under a minute on circuits with several thousand qubits. The amount of time required also increases with the circuit depth (number of iterations p), although we expect both the time and amount of achievable compression to plateau at sufficiently large p, once the causal cones are the size of the entire set of qubits. In particular, we expect no compression of the circuit to be achievable at sufficiently large depth, and already by $p = 4$ there is relatively little savings in the overhead required to execute the circuit achieved by the compilation. We note that the causal-cone size for depth p MaxCut QAOA on a d-regular graph is bounded by d^p and that, as long as d^p is smaller than N, some number of qubits can always be saved with qubit reuse. With more substantial overlap between different causal cones, one can obtain a greater savings in qubit number, since each causal-cone execution makes available another qubit for subsequent reuse while requiring only a number of new qubits that depends on the degree of overlap with previously implemented causal cones. Thus, while the amount of compression is relatively limited at $p = 4$ in Fig. 18 owing to the larger causal-cone size of 3^4, this is still fairly small compared to several thousand, and one can still save a few hundred qubits owing to the amount of overlap between the various causal cones.

The analysis thus far in this section primarily focuses on MaxCut QAOA, as stated, because it is a shallow, wide circuit with a sparse graph of interactions. However, we note that, in fact, dynamics on d-regular graphs such as the case of QAOA are in a sense as difficult as possible for qubit-reuse techniques. In general, for these graphs, the typical causal-cone size grows to the size of the entire set of qubits after only circuit depth $\mathcal{O}(\log N)$, which is as fast as possible. Consequently, MaxCut QAOA essentially quantifies worst-case behavior for applications in which we expect qubit reuse to be useful.

Nonetheless, it is important also to emphasize the benefit of our automated qubit-reuse compilation strategy even for the structured applications studied analytically in the previous section. Therefore, in Figs. 20–22, we utilize compilation via our local greedy heuristic to compress 1D brickwork circuits, 2D brickwork circuits, and MERA circuits, respectively. To illustrate a variety of settings, we choose to display results for very deep 1D circuits with $k = 100$ circuit layers, relatively shallow 2D circuits with $k = 3$, and a variety of MERA sizes. The agreement with analytics for these structured applications illustrates the power of the greedy heuristic, which can also be applied automatically to more complex structured applications that would be laborious to derive by hand, such as complicated lattice geometries or higher bond dimension MERA or QCNN.

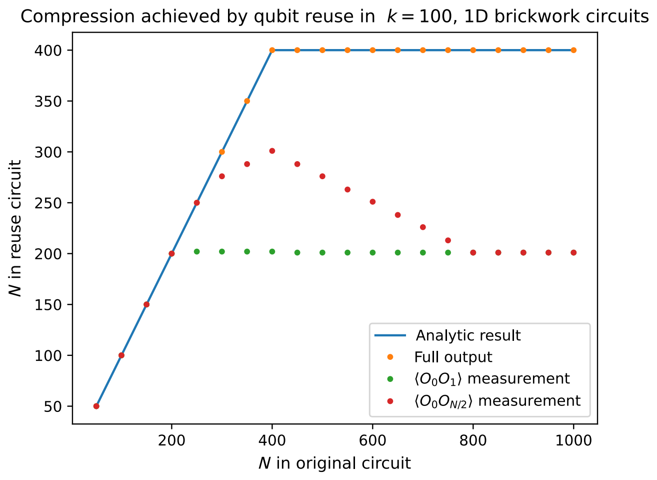


FIG. 20. The compiled qubit number as a function of the number of qubits in the original circuit for one instance at each point of a 1D brickwork circuit with depth 100.

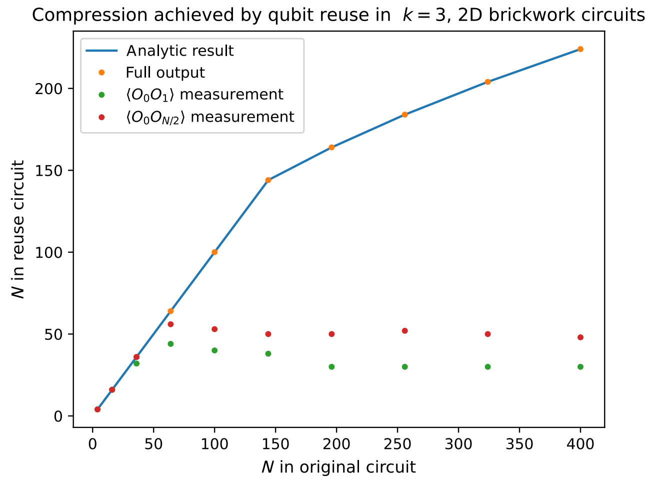


FIG. 21. The compiled qubit number as a function of the number of qubits in the original circuit for one instance at each point of a 2D brickwork circuit with depth 3.

Throughout this work, we assume that the reader is generally interested in sampling the full output of a given quantum circuit. Figures 20–22 display the qubit resource overhead associated with measuring the full output for these three applications and demonstrate that the results obtained by compilation with the greedy heuristic agree precisely with the scalings obtained analytically. In many circumstances, one is interested only in calculating a few-body correlation function of the output qubits. In this case, a further reduction in the number of qubits required to execute the circuit in order to measure this restricted set of output qubits is typically possible. One simply constructs the modified circuit consisting of only the gates belonging to the causal cones of the desired output qubits and utilizes the same algorithms for qubit reuse on this restricted circuit. In Figs. 20–22, we also display the qubit resource

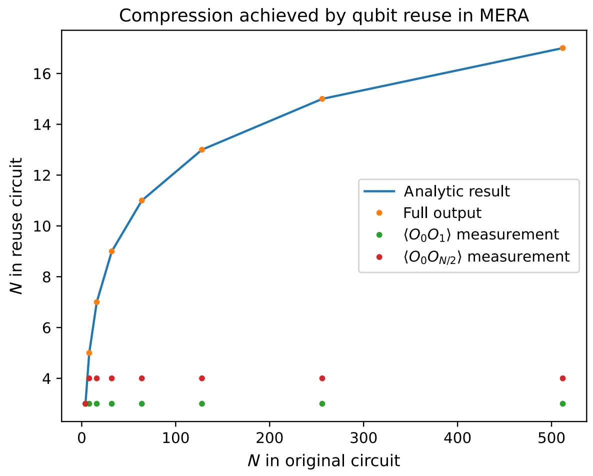


FIG. 22. The compiled qubit number as a function of the number of qubits in the original circuit for one instance at each point of a MERA circuit.

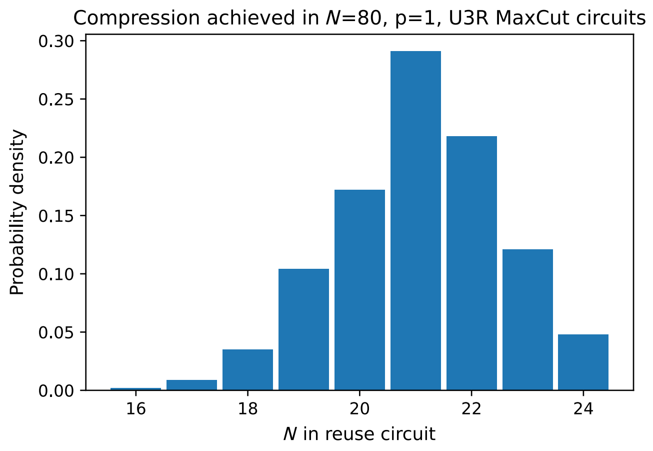


FIG. 23. The number of qubits required to execute $N = 80$, $p = 1$ MaxCut circuits for 1000 instances of random U3R graphs, using the local greedy heuristic algorithm with brute-force search on the first qubit.

overhead associated with measuring just the output of two adjacent qubits q_0 and q_1, as well as with measuring the two most distant qubits q_0 and $q_{N/2}$. In the limit of large N, this overhead effectively scales as a constant, after a short transient period where competing effects like the size of the causal cones compared to the total system size as well as possible intersection between the causal cones exchange dominance.

VI. EXPERIMENTAL DEMONSTRATION

We now experimentally demonstrate the practical functionality of qubit reuse by solving an 80-qubit combinatorial problem with QAOA, using circuits executed on only 20 qubits. We study a specific U3R MaxCut instance by generating a random U3R graph on 80 vertices using the NetworkX PYTHON package, for which the corresponding direct QUBO encoding requires 80 qubits. We expect qubit reuse to be most effective in shallower circuits, so $p = 1$ iteration of QAOA is chosen for this demonstration. A histogram of the typical amount of compression that can be achieved for such 80-qubit $p = 1$ QAOA circuits is displayed in Fig. 23 for 1000 instances of random U3R graphs, which demonstrates that the typical 80-qubit U3R MaxCut instance can be expected to compress to a circuit with 21.1 ± 1.5 qubits on average. In particular, 32% of instances compress to 20 qubits or less, so this amount of compression is a reasonable expectation for a typical instance. As a result, the first random U3R graph that was sampled allowed for compression to the Quantinuum H1-1 hardware limitation of 20 qubits. In order to compute the minimal number of qubits required to execute these circuits, we execute the local greedy algorithm with the additional improvement of brute-force search over the first qubit as described above.

For this random U3R graph, we generated the $p = 1$ QAOA circuit as a function of parameters β_1 and γ_1 that are initialized at a random point. Owing to the expense of numerically approximating derivatives with many function calls, we choose a derivative-free optimizer that performs the best comparatively in simulations, the BOBYQA optimizer (bound optimization by quadratic approximation) [63] as implemented in the PY-BOBYQA package [64]. This optimizer computes interpolating points to generate a quadratic approximation to the objective function within a trust region of a particular size. We use the out-of-the-box settings for this optimizer, wherein the convergence criterion is taken to be the size of the trust region becoming smaller than 10^{-8} [65].

Starting from the random initialization we choose, the full QAOA optimization procedure is run on the Quantinuum H1-1 trapped-ion quantum computer [17]. Each circuit is run for 100 shots, for a total of roughly 3 min per circuit of machine time. This number of shots is chosen by simulation of the optimization procedure as sufficiently large enough to statistically measure decreases in the objective function while remaining small enough to make the machine time feasible. A total of 78 circuits are run before the convergence criterion of the optimizer is met.

In Fig. 24, we show the progress of the objective as evaluated at the best point selected by the optimizer so far. As in Eq. (4), the expectation value is used as the objective function for the optimizer, but we also plot the progress of the best sampled cut value among all of the shots taken.

In Fig. 25, we display the full data taken from all circuits evaluated on the machine. After a short period of exploration in the parameter space, the optimizer converges rapidly around the tenth circuit evaluation. The subsequent circuits, during which the size of the trust region is decreased until meeting the convergence criterion of the optimizer, essentially end up providing extra samples at near-optimal

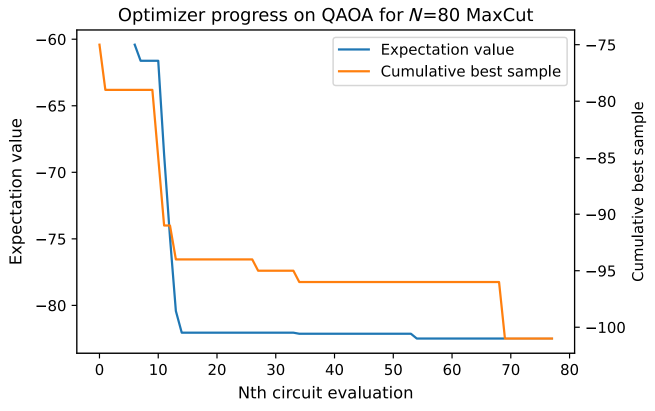


FIG. 24. Progress of the objective function over the course of the BOBYQA optimization (blue), overlaid on the cumulative best sample for all circuits submitted so far (orange). The first six circuits are used to seed the quadratic approximation in the optimizer, which does not record progress before that point.

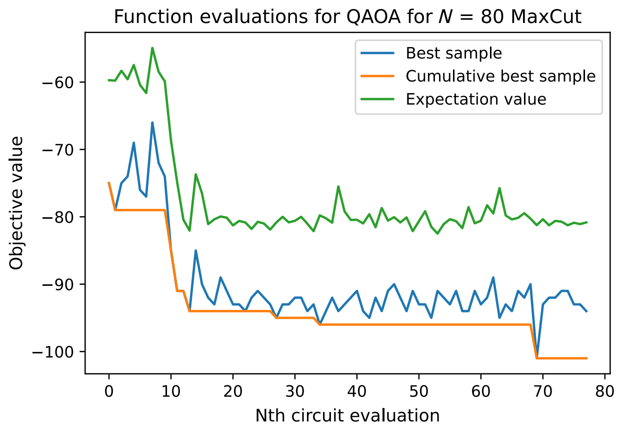


FIG. 25. Full experimental data for all circuits submitted to the H1-1 machine. These data differ from Fig. 24 in that data for all circuits are displayed, not merely those that improve the objective.

parameters, increasing the probability of a high-quality solution (Fig. 26). In Fig. 27, we confirm this interpretation with the full optimization trace overlaid on the energy landscape for this QAOA problem, generated by both noiseless and noisy simulation of the circuits over a lattice of parameter values.

The best cut value measured in any shot throughout the optimization procedure is 101. For comparison, the exact MaxCut value is computed to be 110 using the Gurobi solver [66] built into the QISKIT Optimization package [3]. This exact solver takes only 0.12 s to execute on a laptop, unsurprisingly since $N = 80$ is well within the purview of exact classical solvers. Given this exact value, the approximation ratio achieved by the QAOA run is 91.8%, which requires about 11 h of wall-clock time to execute on H1-1. In Fig. 26, we study the probability of obtaining a cut value of 101 to assess how this value compares to the typical cut

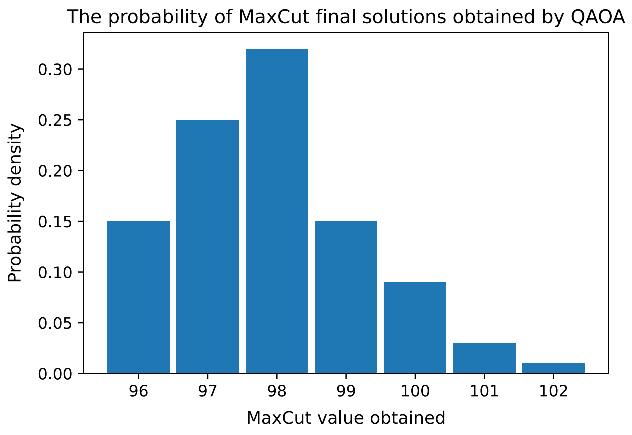


FIG. 26. The probability of obtaining a particular cut value as the final QAOA solution, extracted from 100 circuits with 5000 shots each.

value that could be expected from a repetition of the experiment. As mentioned, from Figs. 24, 25, and 27, it is apparent that the last 50 circuits essentially serve only to provide extra samples at nearly optimal parameters, for a total of 5000 additional shots. Therefore, we take 100 circuits evaluated with 5000 shots each as a proxy for the full optimization procedure and in Fig. 26 display the cut value obtained for the best shot for each circuit. It is clear that, although 101 is slightly better than the average expected result, the distribution of results is fairly narrow and indeed the most likely cut value, 98, is only marginally worse.

We also implement the best-known classical approximation algorithm for MaxCut, the poly-time semidefinite-programming-based Goemans-Williamson (GW) algorithm [67], using the cvxpy solver [68,69]. This algorithm provides a more scalable quantum-classical comparison, since at sufficiently large N exact classical methods time out and one must rely on approximate algorithms to obtain the best possible solution in a reasonable time frame. The GW algorithm achieves a cut value of 104, or an approximation ratio of 94.5%, and requires approximately one second of computation time on a laptop. Note that the GW algorithm guarantees a lower bound for the approximation ratio of 87.8% [70].

VII. DISCUSSION

In this work, we present a method (“qubit-reuse compilation”) for executing quantum circuits using fewer qubits than naively required using mid-circuit measurements and resets. We detail several algorithms that attempt to minimize the number of qubits required in the new circuit and benchmark their performance both numerically on MaxCut QAOA and analytically for certain structured circuits. Finally, we demonstrate the practical application and automated software implementation of these techniques by compressing and

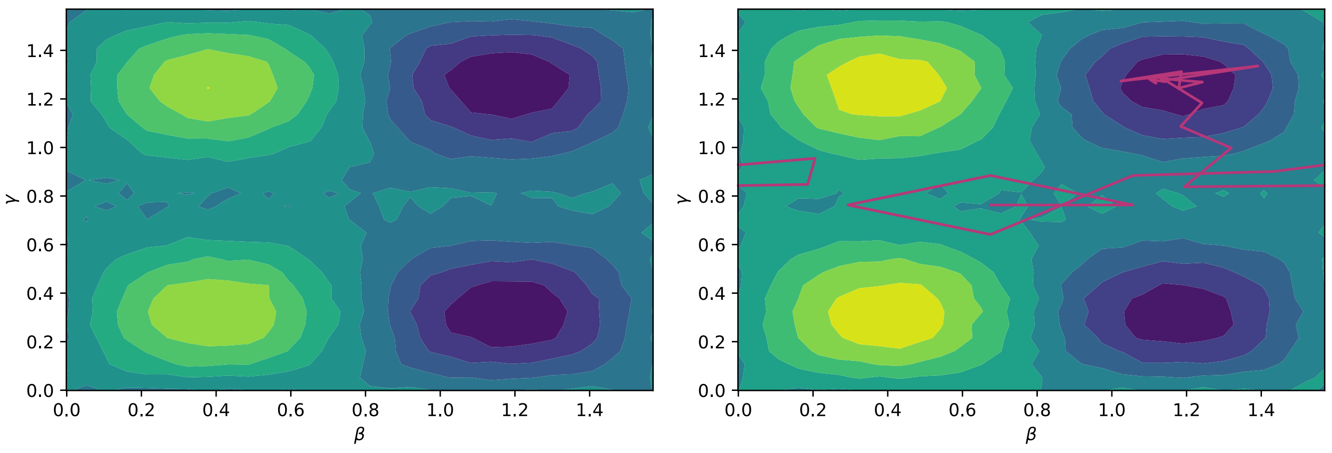


FIG. 27. The energy landscape for the $N = 80$, $p = 1$ MaxCut instance discussed in the main text as a function of (β, γ), where the optimum corresponds to minimum energy (indigo). Left: the noiseless landscape, generated using $30 \times 30 = 900$ ideal simulations of the QAOA circuit. Right: the noisy landscape, generated using $30 \times 30 = 900$ simulations of the QAOA circuit with an appropriate error model for the Quantinuum H1-1 device. The actual optimization trajectory from the experimental data is overlaid in pink.

experimentally running the full optimization procedure for an $N = 80$, $p = 1$ MaxCut QAOA instance on the Quantinuum H1-1 device using 20 qubits.

Since qubit reuse preserves the total number of gates in the original circuit, compressing a circuit using qubit-reuse compilation generally increases circuit depth. Consequently, some qubits in the compressed circuit may experience more memory errors by the time they are measured, since they may be active for the duration of a larger number of gates. Therefore, it is most likely beneficial to compress circuits only down to the number of qubits required to execute the circuit on a desired target device. A simple way to accomplish this when iteratively constructing a compiled circuit from a given measurement order is to prioritize using new qubits over reusing measured qubits at the beginning of the procedure, until the desired minimum number of qubits has been used, and has already been implemented in software.

Because both the total number of gates and their causal structure remain unchanged before and after compressing a circuit, the impact of gate errors is identical between the two (although on superconducting architectures, the subsequent qubit placement onto the connectivity of the architecture may have different impacts in terms of number of SWAP gate insertions; see Refs. [34,35]). Regardless of architecture, there are two new potential sources of error in the compiled circuit: The first is mid-circuit measurement and reset crosstalk, while the second, as mentioned above, is increased memory error associated to the larger depth circuit. On the Quantinuum H1-1 device used in this work, the typical mid-circuit measurement and reset crosstalk is on the order of 10^{-5} or less, suppressed by at least 2 orders of magnitude compared to two-qubit gate errors (2×10^{-3}) which are presently the primary limiting factor for overall circuit fidelity [72]. Architectures with more difficult-to-control mid-circuit measurement crosstalk may not be able to take full advantage of qubit reuse. On the H1-1 device, the typical memory error per qubit in a given circuit layer ("depth-1 circuit") is currently approximately 2×10^{-4}, suppressed by an order of magnitude relative to two-qubit gate errors. If there is a substantial increase in circuit depth after qubit-reuse compilation, this contribution to the error may become appreciable.

The H1-1 device can execute five gates in parallel, so, at least for sufficiently large circuits where some degree of parallelizability remains after qubit reuse, it is possible that there is little change in the overall circuit time and the corresponding memory error. In the worst case, with no parallelizability remaining in the compiled circuit, we expect this to lead to a factor of about 5 times longer for total circuit time. Comparing the memory and gate errors, we see that the net effect in this worst case should result in a memory error contribution roughly comparable to the two-qubit gate error. One should expect this to lead to a measurable but not overwhelming impact on the overall circuit fidelity, depending on the application of interest. This finding is substantiated by, e.g., Refs. [14,15], which employ our qubit-reuse compilation strategies to execute useful and interesting experiments without introducing intolerable amounts of noise. The former, Ref. [14], uses the strategy discussed above to reduce the impact of memory errors by compressing MERA circuits down only to the hardware limitation of 20 qubits on the Quantinuum H1-1 device, which increases the amount of parallelization possible in the compiled circuit.

It is important to note the sense in which a compressed circuit produced by qubit-reuse compilation is equivalent to the original circuit. Experiments performed on the compressed circuit (in the absence of errors) produce bit string samples from the same probability distribution as the original circuit. This is true even though the compressed circuit may have mid-circuit measurements, while the original circuit does not, and that the compressed circuit does not implement the same overall operator and, in fact, acts on a Hilbert space of different dimension. While in a particular experimental realization a mid-circuit measurement collapses the quantum state to a particular quantum trajectory dependent on the measurement outcome, upon averaging over such realizations the estimated reduced density matrices on the measured qubits are equivalent to the reduced density matrices produced by the original circuits. This is because, before any qubit is measured in the compressed circuit, all of the operations in that qubit's causal cone are executed, and, therefore, all of the necessary information for producing that qubit's reduced density matrix has been prepared. All of the other gates in the circuit not in this causal cone do not affect this qubit's density matrix and so can be delayed into a later section of the compressed circuit after this qubit has been reset and reused. Another perspective of this explanation comes from considering the tensor network formulation of the circuit. In this case, the qubit-reuse compiled circuit can be obtained from the original circuit simply from diagrammatic manipulations of the tensor network reinterpreting the locations of input and output qubits, so they must sample from the same distributions.

In many cases of interest, the original circuit is not unique but can be expressed using a different ordering and possibly different set of gates. One of the simplest possible examples of this occurs in both QAOA and in Hamiltonian simulation of Ising models, where many commuting $ZZ(\theta)$ gates are applied in a nonunique order that determines the resulting causal structure of the circuit. For this commuting gates problem, it is always possible to implement the block of commuting gates in a number of layers that is at most one more than the maximal degree of the interaction graph, by mapping the problem to edge coloring and using Vizing's theorem [73,74]. Preliminary study indicates that minimizing the original circuit depth in this way also improves the amount of compression obtainable by qubit reuse, but, in general, the effects of the original circuit depth and gate ordering on causal structure merit further consideration.

ACKNOWLEDGMENTS

The authors thank many people within the broader Quantinuum team for their contributions toward this work. We acknowledge useful conversations with David Hayes, Steve Ragole, Jason Dominy, Reza Haghshenas, Mattia Fiorentini, David Amaro, Marcello Benedetti, and Ross Duncan. The experimental data in this work were produced by the Quantinuum H1-1 trapped-ion quantum computer, powered by Honeywell, and we thank Brian Neyenhuis and the Quantinuum H1-1 operations team for their support and assistance.

[1] S. Bravyi, R. Shaydulin, S. Hu, and D. Maslov, *Clifford circuit optimization with templates and symbolic Pauli gates*, Quantum **5**, 580 (2021).
[2] S. Aaronson and D. Gottesman, *Improved simulation of stabilizer circuits*, Phys. Rev. A **70**, 052328 (2004).
[3] M. Anis, A. Mitchell, H. Abraham, A. Offei, R. Agarwal, G. Agliardi, M. Aharoni, V. Ajith, I. Y. Akhalwaya, G. Aleksandrowicz, T. Alexander, M. Amy, S. Anagolum, Anthony-Gandon, E. Arbel *et al.*, *QISKIT: An open-source framework for quantum computing*, https://zenodo.org/records/8190968.
[4] S. Sivarajah, S. Dilkes, A. Cowtan, W. Simmons, A. Edgington, and R. Duncan, $t|\mathrm{ket}\rangle$*: A retargetable compiler for NISQ devices*, Quantum Sci. Technol. **6**, 014003 (2020).
[5] W. Jang, K. Terashi, M. Saito, C. Bauer, B. Nachman, Y. Iiyama, T. Kishimoto, R. Okubo, R. Sawada, and J. Tanaka, *Quantum gate pattern recognition and circuit optimization for scientific applications*, EPJ Web Conf. **251**, 03023 (2021).
[6] N. Khaneja and S. Glaser, *Cartan decomposition of* $SU(2^n)$, *constructive controllability of spin systems and universal quantum computing*, arXiv:quant-ph/0010100.
[7] W.-H. Lin, B. Tan, M. Niu, J. Kimko, and J. Cong, *Domain-specific quantum architecture optimization*, arXiv:2207.14482.
[8] A. Zulehner, A. Paler, and R. Wille, *An efficient methodology for mapping quantum circuits to the IBM QX architectures*, IEEE Trans. Computer-Aided Design Integrated Circ. Syst. **38**, 1226 (2019).
[9] C. Ryan-Anderson, N. C. Brown, M. S. Allman, B. Arkin, G. Asa-Attuah, C. Baldwin, J. Berg, J. G. Bohnet, S. Braxton, N. Burdick, J. P. Campora, A. Chernoguzov, J. Esposito, B. Evans, D. Francois *et al.*, *Implementing fault-tolerant entangling gates on the five-qubit code and the color code*, arXiv:2208.01863.
[10] J. M. Pino, J. M. Dreiling, C. Figgatt, J. P. Gaebler, S. A. Moses, M. S. Allman, C. H. Baldwin, M. Foss-Feig, D. Hayes, K. Mayer, C. Ryan-Anderson, and B. Neyenhuis, *Demonstration of the trapped-ion quantum CCD computer architecture*, Nature (London) **592**, 209 (2021).
[11] P. Nation, *Dynamic Bernstein-Vazirani using midcircuit reset and measurement*, https://nonhermitian.org/posts/2021/2021-10-27-dynamic_BV.html.
[12] E. Chertkov, J. Bohnet, D. Francois, J. Gaebler, D. Gresh, A. Hankin, K. Lee, D. Hayes, B. Neyenhuis, R. Stutz, A. C. Potter, and M. Foss-Feig, *Holographic dynamics simulations with a trapped-ion quantum computer*, Nat. Phys. **18**, 1074 (2022).
[13] E. Chertkov, Z. Cheng, A. C. Potter, S. Gopalakrishnan, T. M. Gatterman, J. A. Gerber, K. Gilmore, D. Gresh, A. Hall, A. Hankin, M. Matheny, T. Mengle, D. Hayes, B. Neyenhuis, R. Stutz, and M. Foss-Feig, *Characterizing a non-equilibrium phase transition on a quantum computer*, Nat. Phys. (2023), 10.1038/s41567-023-02199-w.
[14] R. Haghshenas, E. Chertkov, M. DeCross, T. M. Gatterman, J. A. Gerber, K. Gilmore, D. Gresh, N. Hewitt, C. V. Horst, M. Matheny, T. Mengle, B. Neyenhuis, D. Hayes, and M. Foss-Feig, *Probing critical states of matter on a digital quantum computer*, arXiv:2305.01650.
[15] C. Harvey, R. Yeung, and K. Meichanetzidis, *Sequence processing with quantum tensor networks*, arXiv:2308.07865.

[16] J. Yirka and Y. Subaşı, *Qubit-efficient entanglement spectroscopy using qubit resets*, Quantum **5**, 535 (2021).

[17] *Quantinuum, Quantinuum H1-1*, https://www.quantinuum.com/.

[18] *Quantinuum, Quantinuum sets new record with highest ever quantum volume*, https://www.quantinuum.com/news/quantinuum-sets-new-record-with-highest-ever-quantum-volume.

[19] A. Lowe, M. Medvidović, A. Hayes, L. J. O'Riordan, T. R. Bromley, J. M. Arrazola, and N. Killoran, *Fast quantum circuit cutting with randomized measurements*, Quantum **7**, 934 (2023).

[20] J. Li, M. Alam, and S. Ghosh, *Large-scale quantum approximate optimization via divide-and-conquer*, IEEE Trans. Computer-Aided Design Integrated Circ. Syst. **42**, 1852 (2023).

[21] Z. H. Saleem, T. Tomesh, M. A. Perlin, P. Gokhale, and M. Suchara, *Divide and conquer for combinatorial optimization and distributed quantum computation*, arXiv:2107.07532.

[22] W. Tang and M. Martonosi, *Scaleqc: A scalable framework for hybrid computation on quantum and classical processors*, arXiv:2207.00933.

[23] T. Peng, A. W. Harrow, M. Ozols, and X. Wu, *Simulating large quantum circuits on a small quantum computer*, Phys. Rev. Lett. **125**, 150504 (2020).

[24] C. Piveteau and D. Sutter, *Circuit knitting with classical communication*, IEEE Trans. Inf. Theory (2023), 10.1109/TIT.2023.3310797.

[25] Although Ref. [20] observes that it is possible to design policies that reconstruct the full state from the partial state measurements while minimizing the effect of entanglement between the partitioned subsystems. See also Refs. [26,27] for interesting approaches that recombine solutions for cut subcircuits by casting the recombination process as an optimization problem.

[26] Z. Zhou, Y. Du, X. Tian, and D. Tao, *Qaoa-in-qaoa: Solving large-scale maxcut problems on small quantum machines*, Phys. Rev. Appl. **19**, 024027 (2023).

[27] G. Uchehara, T. M. Aamodt, and O. D. Matteo, *Rotation-inspired circuit cut optimization*, in *Proceedings of the 2022 IEEE/ACM Third International Workshop on Quantum Computing Software (QCS)* (IEEE, New York, 2022), pp. 50–56.

[28] T. L. Patti, J. Kossaifi, A. Anandkumar, and S. F. Yelin, *Variational quantum optimization with multibasis encodings*, Phys. Rev. Res. **4**, 033142 (2022).

[29] B. Fuller, C. Hadfield, J. R. Glick, T. Imamichi, T. Itoko, R. J. Thompson, Y. Jiao, M. M. Kagele, A. W. Blom-Schleiber, R. Raymond, and A. Mezzacapo, *Approximate solutions of combinatorial problems via quantum relaxations*, arXiv:2111.03167.

[30] C. Huang, M. Szegedy, F. Zhang, X. Gao, J. Chen, and Y. Shi, *Alibaba cloud quantum development platform: Applications to quantum algorithm design*, arXiv:1909.02559.

[31] D. Lykov, R. Schutski, A. Galda, V. Vinokur, and Y. Alexeev, *Tensor network quantum simulator with step-dependent parallelization*, in *Proceedings of the 2022 IEEE International Conference on Quantum Computing and Engineering (QCE)* (IEEE, New York, 2022), pp. 582–593.

[32] A. Paler, R. Wille, and S. J. Devitt, *Wire recycling for quantum circuit optimization*, Phys. Rev. A **94**, 042337 (2016).

[33] See Supplemental Material at http://link.aps.org/supplemental/10.1103/PhysRevX.13.041057 for visual examples of the effect of qubit-reuse compilation on several application circuits and technical statements of the CP-SAT constraints.

[34] M. Sadeghi, S. Khadirsharbiyani, and M. T. Kandemir, *Quantum circuit resizing*, arXiv:2301.00720.

[35] F. Hua, Y. Jin, Y. Chen, S. Vittal, K. Krsulich, L. S. Bishop, J. Lapeyre, A. Javadi-Abhari, and E. Z. Zhang, *Caqr: A compiler-assisted approach for qubit reuse through dynamic circuit*, in *Proceedings of the 28th ACM International Conference on Architectural Support for Programming Languages and Operating Systems*, Volume 3, ASPLOS 2023 (Association for Computing Machinery, New York, 2023), pp. 59–71.

[36] G. Vidal, *Class of quantum many-body states that can be efficiently simulated*, Phys. Rev. Lett. **101**, 110501 (2008).

[37] I. H. Kim, *Holographic quantum simulation*, arXiv:1702.02093.

[38] I. H. Kim, *Noise-resilient preparation of quantum many-body ground states*, arXiv:1703.00032.

[39] W. Huggins, P. Patil, B. Mitchell, K. B. Whaley, and E. M. Stoudenmire, *Towards quantum machine learning with tensor networks*, Quantum Sci. Technol. **4**, 024001 (2019).

[40] J.-G. Liu, Y.-H. Zhang, Y. Wan, and L. Wang, *Variational quantum eigensolver with fewer qubits*, Phys. Rev. Res. **1**, 023025 (2019).

[41] M. Foss-Feig, D. Hayes, J. M. Dreiling, C. Figgatt, J. P. Gaebler, S. A. Moses, J. M. Pino, and A. C. Potter, *Holographic quantum algorithms for simulating correlated spin systems*, Phys. Rev. Res. **3**, 033002 (2021).

[42] X. Yuan, J. Sun, J. Liu, Q. Zhao, and Y. Zhou, *Quantum simulation with hybrid tensor networks*, Phys. Rev. Lett. **127**, 040501 (2021).

[43] M. L. Wall, P. Titum, G. Quiroz, M. Foss-Feig, and K. R. A. Hazzard, *Tensor-network discriminator architecture for classification of quantum data on quantum computers*, Phys. Rev. A **105**, 062439 (2022).

[44] M. Benedetti, M. Fiorentini, and M. Lubasch, *Hardware-efficient variational quantum algorithms for time evolution*, Phys. Rev. Res. **3**, 033083 (2021).

[45] D. Amaro, C. Modica, M. Rosenkranz, M. Fiorentini, M. Benedetti, and M. Lubasch, *Filtering variational quantum algorithms for combinatorial optimization*, Quantum Sci. Technol. **7**, 015021 (2022).

[46] O. Shehab, I. H. Kim, N. H. Nguyen, K. Landsman, C. H. Alderete, D. Zhu, C. Monroe, and N. M. Linke, *Noise reduction using past causal cones in variational quantum algorithms*, arXiv:1906.00476.

[47] Although this implies a subcase of (C.2), it is an independent constraint. The constraint (C.2) implies that a qubit q measured at step t is itself active at some point possibly *before* t, while (C.4) is the stronger statement that q is also active at step t.

[48] This does not mean the circuit is independent of q later; rather, it means that the operations involving q are already accounted for.

[49] L. Perron and V. Furnon, *Or-tools*, https://developers.google.com/optimization/.

[50] P. J. Stuckey, *Search is dead, long live proof!*, in *Proceedings of the 15th International Symposium on Principles and Practice of Declarative Programming, Universidad Complutense de Madrid, Madrid, Spain* (2013).

[51] F. Barratt, J. Dborin, M. Bal, V. Stojevic, F. Pollmann, and A. G. Green, *Parallel quantum simulation of large systems on small NISQ computers*, npj Quantum Inf. **7**, 79 (2021).

[52] D. Niu, R. Haghshenas, Y. Zhang, M. Foss-Feig, Garnet Kin-Lic Chan, and A. C. Potter, *Holographic simulation of correlated electrons on a trapped-ion quantum processor*, PRX Quantum **3**, 030317 (2022).

[53] Y. Zhang, S. Jahanbani, D. Niu, R. Haghshenas, and A. C. Potter, *Qubit-efficient simulation of thermal states with quantum tensor networks*, Phys. Rev. B **106**, 165126 (2022).

[54] C. Schön, E. Solano, F. Verstraete, J. I. Cirac, and M. M. Wolf, *Sequential generation of entangled multiqubit states*, Phys. Rev. Lett. **95**, 110503 (2005).

[55] OEIS Foundation Inc., The ruler function: Exponent of the highest power of 2 dividing 2n. equivalently, the 2-adic valuation of 2n, https://oeis.org/A001511.

[56] I. Cong, S. Choi, and M. D. Lukin, *Quantum convolutional neural networks*, Nat. Phys. **15**, 1273 (2019).

[57] *Honeywell, Get to know Honeywell's latest quantum computer system model H1*, https://www.honeywell.com/us/en/news/2020/10/get-to-know-honeywell-s-latest-quantum-computer-system-model-h1.

[58] L. Zhou, S.-T. Wang, S. Choi, H. Pichler, and M. D. Lukin, *Quantum approximate optimization algorithm: Performance, mechanism, and implementation on near-term devices*, Phys. Rev. X **10**, 021067 (2020).

[59] M. P. Harrigan, K. J. Sung, M. Neeley, K. J. Satzinger, F. Arute, K. Arya, J. Atalaya, J. C. Bardin, R. Barends, S. Boixo *et al.*, *Quantum approximate optimization of non-planar graph problems on a planar superconducting processor*, Nat. Phys. **17**, 332 (2021).

[60] S. Ebadi, A. Keesling, M. Cain, T. T. Wang, H. Levine, D. Bluvstein, G. Semeghini, A. Omran, J.-G. Liu, R. Samajdar, X.-Z. Luo, B. Nash, X. Gao, B. Barak, E. Farhi *et al.*, *Quantum optimization of maximum independent set using Rydberg atom arrays*, Science **376**, 1209 (2022).

[61] E. Farhi, J. Goldstone, and S. Gutmann, *A quantum approximate optimization algorithm*, arXiv:1411.4028.

[62] A. A. Hagberg, D. A. Schult, and P. J. Swart, *Exploring network structure, dynamics, and function using networkx*, in *Proceedings of the 7th Python in Science Conference (SciPy2008), Pasadena, CA*, edited by G. Varoquaux, T. Vaught, and J. Millman (2008), p. 11.

[63] M. Powell, *The BOBYQA algorithm for bound constrained optimization without derivatives*, technical report, Department of Applied Mathematics and Theoretical Physics, 2009.

[64] C. Cartis, J. Fiala, B. Marteau, and L. Roberts, *Improving the flexibility and robustness of model-based derivative-free optimization solvers*, ACM Trans. Math. Softw. **45**, 1 (2019).

[65] This was more demanding than necessary, since the variational angles $\vec{\beta}$ and $\vec{\gamma}$ cannot be implemented on hardware with fidelity matching this precision.

[66] *Gurobi Optimization, LLC, Gurobi Optimizer Reference Manual*, https://www.gurobi.com/.

[67] M. X. Goemans and D. P. Williamson, *Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming*, J. ACM **42**, 1115 (1995).

[68] S. Diamond and S. Boyd, *CVXPY: A Python-embedded modeling language for convex optimization*, J. Machine Learn. Res. **17**, 1 (2016), https://dl.acm.org/doi/10.5555/2946645.3007036.

[69] A. Agrawal, R. Verschueren, S. Diamond, and S. Boyd, *A rewriting system for convex optimization problems*, J. Control Decision **5**, 42 (2018).

[70] This can be improved to 93.2% for the specific case of U3R graphs by a series of classical local postprocessing steps [71]. However, this postprocessing can also be applied to the quantum approximate result.

[71] E. Halperin, D. Livnat, and U. Zwick, *Max cut in cubic graphs*, J. Algorithms **53**, 169 (2004).

[72] *Quantinuum, Quantinuum system model h1 product data sheet*, https://www.quantinuum.com/hardware/h1.

[73] V. Vizing, *Critical graphs with given chromatic class*, Met. Diskret. Analiz. **5**, 9 (1965).

[74] J. Misra and D. Gries, *A constructive proof of Vizing's theorem*, Inf. Proc. Lett. **41**, 131 (1992).