SIAM J. COMPUT.
Vol. 38, No. 3, pp. 963–981

SIMULATING QUANTUM COMPUTATION BY CONTRACTING TENSOR NETWORKS*

IGOR L. MARKOV† AND YAOYUN SHI†

Abstract. The treewidth of a graph is a useful combinatorial measure of how close the graph is to a tree. We prove that a quantum circuit with T gates whose underlying graph has a treewidth d can be simulated deterministically in $T^{O(1)}\exp[O(d)]$ time, which, in particular, is polynomial in T if $d = O(\log T)$. Among many implications, we show efficient simulations for log-depth circuits whose gates apply to nearby qubits only, a natural constraint satisfied by most physical implementations. We also show that *one-way quantum computation* of Raussendorf and Briegel (*Phys. Rev. Lett.*, 86 (2001), pp. 5188–5191), a universal quantum computation scheme with promising physical implementations, can be efficiently simulated by a randomized algorithm if its quantum resource is derived from a small-treewidth graph with a constant maximum degree. (The requirement on the maximum degree was removed in [I. L. Markov and Y. Shi, preprint:quant-ph/0511069].)

Key words. quantum computation, computational complexity, treewidth, tensor network, classical simulation, one-way quantum computation

AMS subject classifications. 81P68, 68Q05, 68Q10, 05C83, 68R10

DOI. 10.1137/050644756

1. Introduction.

The recent interest in quantum circuits is motivated by several complementary considerations. Quantum information processing is rapidly becoming a reality as it allows the manipulation matter on an unprecedented scale. Such manipulations may create particular entangled states or implement specific quantum evolutions—they find uses in atomic clocks, ultra-precise metrology, high-resolution lithography, optical communication, etc. On the other hand, engineers traditionally simulate new designs before implementing them. Such simulation may identify subtle design flaws and save both costs and effort. It typically uses well-understood host hardware, e.g., one can simulate a quantum circuit on a commonly-used conventional computer.

More ambitiously, quantum circuits compete with conventional computing and communication. Quantum-mechanical effects may potentially lead to computational speed-ups, more secure or more efficient communication, better keeping of secrets, etc. To this end, one seeks new circuits and algorithms with revolutionary behavior as in Shor's work on number-factoring, or provable limits on possible behaviors. While proving abstract limitations on the success of unknown algorithms appears more difficult, a common line of reasoning for such results is based on simulation. For example, if the behavior of a quantum circuit can be faithfully simulated on a conventional computer, then the possible speed-up achieved by the quantum circuit is limited by the cost of simulation. Thus, aside from sanity-checking new designs for quantum information-processing hardware, more efficient simulation can lead to sharper bounds on all possible algorithms.

*Received by the editors November 10, 2005; accepted for publication (in revised form) November 27, 2007; published electronically June 25, 2008.
http://www.siam.org/journals/sicomp/38-3/64475.html

†Department of Electrical Engineering and Computer Science, University of Michigan, 2260 Hayward Street, Ann Arbor, MI 48109-2121 (imarkov@eecs.umich.edu, shiyy@eecs.umich.edu). The first author's research was supported in part by NSF 0208959, the DARPA QuIST program, and the Air Force Research Laboratory. The second author's research was supported in part by NSF 0323555, 0347078, and 0622033.

Since the outcome of a quantum computation is probabilistic, we shall clarify our notion of simulation. By a randomized simulation, we mean a classical randomized algorithm whose output distribution on an input is identical to that of the simulated quantum computation. By a deterministic simulation, we mean a classical deterministic algorithm which, on a given pair of input x and output y of the quantum computation, outputs the probability that y is observed at the end of the quantum computation on x.

To simulate a quantum circuit, one may use a naïve brute-force calculation of quantum amplitudes that has exponential overhead. Achieving significantly smaller overhead in the generic case appears hopeless—in fact, this observation lead Feynman to suggest that quantum computers may outperform conventional ones in some tasks. Therefore, only certain restricted classes of quantum circuits were studied in existing literature on simulation.

Classes of quantum circuits that admit efficient simulation are often distinguished by a restricted "gate library," but do not impose additional restrictions on how gates are interconnected or sequenced. A case in point is the seminal Gottesman–Knill theorem [12] and its recent improvement by Aaronson and Gottesman [1]. These results apply only to circuits with stabilizer gates—controlled-NOT, Hadamard, phase, and single-qubit measurements in the so-called Clifford group. Another example is given by *match gates* defined and studied by Valiant [33], and extended by Terhal and DiVincenzo [31].

A different way to impose a restriction on a class of quantum circuits is to limit the amount of entanglement in intermediate states. Jozsa and Linden [16], as well as Vidal [36], demonstrate efficient classical simulation of such circuits and conclude that achieving quantum speed-ups requires more than a bounded amount of entanglement.

In this work we pursue a different approach to efficient simulation and allow the use of arbitrary gates. More specifically, we assume a general quantum circuit model in which a gate is a general quantum operation (so-called *physically realizable operators*) on a constant number of qubits. This model, proposed and studied by Aharonov, Kitaev, and Nisan [2], generalizes the standard quantum circuit model, defined by Yao [40], where each gate is unitary and measurements are applied at the end of the computation. We also assume that (i) the computation starts with a fixed unentangled state in the computational basis, and (ii) at the end each qubit is either measured or *traced-out*.

Our simulation builds upon the framework of *tensor network contraction*. Being a direct generalization of matrices, tensors capture a wide range of linear phenomena including vectors, operators, multilinear forms, etc. They facilitate convenient and fundamental mathematical tools in many branches of physics such as fluid and solid mechanics, and general relativity [14]. More recently, several methods have been developed to simulate quantum evolution by contracting variants of tensor networks, under the names of *matrix product states (MPS)*, *projected entangled pairs states (PEPS)*, etc. [36, 37, 34, 41, 35, 24]. Under this framework, a quantum circuit is regarded as a network of tensors. The simulation contracts edges one by one and performs the convolution of the corresponding tensors, until there is only one vertex left. Having degree 0, this vertex must be labeled by a single number, which gives the final measurement probability sought by simulation. In contrast with other simulation techniques, we do not necessarily simulate individual gates in their original order—in fact, a given gate may even be simulated *partially* at several stages of the simulation.

While tensor network contraction has been used in previous work, little was known

about optimal contraction orders. We prove that the minimal cost of contraction is determined by the *treewidth* $\mathrm{tw}(G_C)$ of the circuit graph G_C. Moreover, existing constructions that approximate optimal *tree-decompositions* (e.g., [28]) produce near-optimal contraction sequences. We shall define the concepts of treewidth and tree decompositions in section 2. Intuitively, the smaller a graph's treewidth is, the closer it is to a tree, and a tree decomposition is a drawing of the graph to make it look like a tree as much as possible. Our result allows us to leverage the extensive graph-theoretical literature dealing with the properties and computation of treewidth.

THEOREM 1.1. *Let C be a quantum circuit with T gates and whose underlying circuit graph is G_C. Then C can be simulated deterministically in time $T^{O(1)}\exp[O(\mathrm{tw}(G_C))]$.*

A rigorous restatement of the above theorem is Theorem 4.6. By this theorem, given a function computable in polynomial time by a quantum algorithm but not classically, any polynomial-size quantum circuit computing the function must have super-logarithmic treewidth. The following corollary is an immediate consequence.

COROLLARY 1.2. *Any polynomial-size quantum circuit of a logarithmic treewidth can be simulated deterministically in polynomial time.*

Quantum formulas defined and studied by Yao [40] are quantum circuits whose underlying graphs are trees. Roychowdhury and Vatan [30] showed that quantum formulas can be efficiently simulated deterministically. Since every quantum formula has treewidth 1, Corollary 1.2 gives an alternative efficient simulation.

Our focus on the *topology* of the quantum circuit allows us to accommodate arbitrary gates, as long as their qubit-width (number of inputs) is limited by a constant. In particular, Corollary 1.2 implies efficient simulation of some circuits that create the maximum amount of entanglement in a partition of the qubits, e.g., a layer of two-qubit gates. Therefore, our results are not implied by previously published techniques.

We now articulate some implications of our main result to classes of quantum circuits, in terms of properties of their underlying graphs. The following two classes of graphs are well-studied, and their treewidths are known. The class of *series parallel graphs* arises in electric circuits, and such circuits have treewidth ≤ 2. Planar graphs G with n vertices are known to have treewidth $\mathrm{tw}(G) = O(\sqrt{|V(G)|})$ [4].

COROLLARY 1.3. *Any polynomial size parallel serial quantum circuit can be simulated deterministically in polynomial time.*

COROLLARY 1.4. *A size T planar quantum circuit can be simulated deterministically in $\exp[O(\sqrt{T})]$ time.*

Another corollary deals with a topological restriction representative of many physical realizations of quantum circuits. Let $q \geq 1$ be an integer. A circuit is said to be q-local-interacting if under a linear ordering of its qubits, each gate acts only on qubits that are at most q distance apart. A circuit is said to be local-interacting if it is q-local interacting with a constant q independent of the circuit size. Such *local-interaction* circuits generalize the restriction of qubit couplings to nearest-neighbor qubits (e.g., in a spin-chain) commonly appearing in proposals for building quantum computers, where qubits may be stationary and cannot be coupled arbitrarily. To this end, we observe that the treewidth of any local-interaction circuit of logarithmic depth is at most logarithmic.

COROLLARY 1.5. *Let C be a quantum circuit of size T and depth D and q-local-interacting. Then C can be simulated deterministically in $T^{O(1)}\exp[O(qD)]$ time. In particular, if C is a polynomial-size local-interacting circuit with a logarithmic depth,*

then it can be simulated deterministically in polynomial time.

Yet another important application of our approach is the simulation of *one-way quantum computation*. In two influential papers [7, 25], Briegel and Raussendorf introduced the concept of *graph states*—quantum states derived from graphs,—and show that an arbitrary quantum circuit can be simulated by *adaptive*, *single-qubit measurements* on the *graph state* derived from the grid graph. Note that the graph state for a one-way quantum computation does not depend on the quantum circuit to be simulated (except that its size should be large enough) and that for most physical implementations single-qubit measurements are much easier to implement than multiqubit operations. Hence it is conceivable that graph states would be manufactured by a technologically more advanced party, then used by other parties with lesser quantum-computational power in order to facilitate universal quantum computing. This makes one-way quantum computation an attractive scheme for physical implementations of universal quantum computation. An experimental demonstration of one-way quantum computation appeared in a recent Nature article [38].

A natural question about one-way computation is to characterize the class of graphs whose graph states are universal for quantum computation. We call a family of quantum states $\phi = \{|\phi_1\rangle, |\phi_2\rangle, \ldots, |\phi_n\rangle, \ldots\}$ *universal for one-way quantum computation* if (a) the number of qubits in $|\phi_n\rangle$ is bounded by a fixed polynomial in n; (b) any quantum circuit of size n can be simulated by a one-way quantum computation on $|\phi_n\rangle$. On the other hand, ϕ is said to be efficiently simulatable if any one-way quantum computation on $|\phi_n\rangle$ can be efficiently simulated classically for all sufficiently large n. Note that the class of universal families and that of efficiently simulatable families are disjoint if and only if efficient quantum computation is indeed strictly more powerful than efficient classical computation. We show that it is necessary for graphs with a constant maximum degree to have high treewidth so that the corresponding graph states are not efficiently simulatable.

THEOREM 1.6. *Let G be a simple undirected graph with the maximum degree $\Delta(G)$. Then a one-way quantum computation on the respective graph state can be simulated by a randomized algorithm in time $|V(G)|^{O(1)} \exp[O(\Delta(G)\mathrm{tw}(G))]$.*

The above result was improved in [19] so that the simulation time does not depend exponentially in $\Delta(G)$. Our simulation can be made deterministic with a better upper bound on time complexity if the one-way computation satisfies additional constraints, such as those in [25]. We shall elaborate on this improvement in section 6.

An important limitation of our techniques is that a circuit family with sufficiently fast-growing treewidth may require super-polynomial resources for simulation. In particular, this seems to be the case with known circuits for modular exponentiation. Therefore, there is little hope to efficiently simulate number-factoring algorithms using tree decompositions. As an extreme example to illustrate the limitation of our technique, we give a depth-4 circuit—including the final measurement as the fourth layer—that has large treewidth.

THEOREM 1.7. *There exists a depth-4 quantum circuit on n qubits using only one- and two-qubit gates such that its treewidth is $\Omega(n)$.*

Note that a circuit satisfying the assumption in the above theorem must have $O(n)$ size. Our construction is based on expander graphs, whose treewidth must be linear in the number of vertices (Lemma 5.2).

This finding is consistent with the obstacles to efficient simulation that are evident in the results of Terhal and DiVincenzo [32], later extended by Fenner et al. [13]. In contrast, we are able to efficiently simulate any depth-3 circuit *deterministically* while

the simulation in [32] is probabilistic.

THEOREM 1.8. *Assuming that only one- and two-qubit gates are allowed, any polynomial-size depth-3 quantum circuit can be simulated deterministically in polynomial time.*

Our simulation algorithm is related to algorithms for other tasks in that its runtime depends on the treewidth of a graph derived from the input. Bodlaender wrote an excellent survey [8] on this subject. Particularly relevant are algorithms based on "vertex eliminations," e.g., the *bucket elimination* algorithm for Bayesian inference [11]. Another parallel can be made with the work by Broering and Lokam [10], which solves circuit-SAT in time exponential in the treewidth of the graph of the given circuit. However, to the best of our best knowledge, we are the first to relate the treewidth of a quantum circuit to its classical simulation.

Our results are applicable to the simulation of classical probabilistic circuits, which can be modeled by matrices, similarly to quantum circuits. Such simulation has recently gained prominence in the literature on the reliability of digital logic [17], and is particularly relevant to satellite-based and airborne electronics which experience unpredictable particle strikes at higher rates.

The rest of this paper is organized as follows. After introducing notation, we describe how quantum circuits and their simulation can be modeled by tensor networks. The runtime of such simulation depends on the graph parameter that we call the *contraction complexity*. We then relate the contraction complexity to treewidth, and apply the simulation to restricted classes of graphs and to one-way quantum computation. Finally, we discuss possible directions for future investigations with a brief survey on the subsequent development since the announcement of our results.

2. Notation and definitions. For integer $n \geq 1$, define $[n] \stackrel{\text{def}}{=} \{1, 2, \ldots, n\}$. An ordering π of an n-element set is denoted by $\pi(1)$, $\pi(2)$, ..., $\pi(n)$. Unless otherwise stated, graphs in this paper are undirected and may have multiple edges or loops. Edges connecting the same pair of vertices are called parallel edges. If G is a graph, then its vertex set is denoted by $V(G)$ and its edge set by $E(G)$. When it is clear in this context, we use $V = V(G)$ and $E = E(G)$. The *degree* of a vertex v, denoted by $d(v)$, is the number of edges incident to it. In particular, a loop counts as 1 edge. The maximum degree of a vertex in G is denoted by $\Delta(G)$.

Treewidth of a graph. Let G be a graph. A *tree decomposition* of G [27] is a tree $\mathcal{T}$, together with a function that maps each vertex $w \in V(\mathcal{T})$ to a subset $B_w \subseteq V(G)$. These subsets B_w are called *bags* (of vertices). In addition, the following conditions must hold.

(T1) $\bigcup_{v \in V(\mathcal{T})} B_v = V(G)$, i.e., each vertex must appear in at least one bag.
(T2) $\forall\ \{u, v\} \in E(G)$, $\exists w \in V(\mathcal{T})$, $\{u, v\} \subseteq B_w$, i.e., for each edge, at least one bag must contain both of its endvertices.
(T3) $\forall\ u \in V(G)$, the set of vertices $w \in V(\mathcal{T})$ with $u \in B_w$ form a connected subtree, i.e., all bags containing a given vertex must be connected in $\mathcal{T}$.

The *width* of a tree decomposition is defined by $\max_{w \in V(\mathcal{T})} |B_w| - 1$. The *treewidth* of G is the minimum width over its tree decompositions. For example, all trees have treewidth 1 and single cycles of length at least 3 have treewidth 2. Figure 1 shows an example of tree decomposition. Intuitively, a tree decomposition $\mathcal{T}$ is a way of drawing a graph to look like a tree, which may require viewing sets of vertices (bags) as single vertices. The less a graph looks like a tree, the larger the bags become. The notion of tree decomposition has been useful in capturing the complexity of constraint

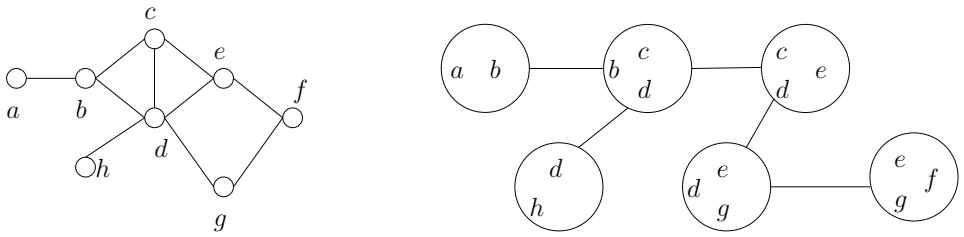

FIG. 1. *A graph and its decomposition of width* 2 *with* 6 *bags.*

satisfaction problems, Bayesian networks, and other combinatorial phenomena represented by graphs. In further writing, we may refer to a vertex in $\mathcal{T}$ by its bag when the context is clear.

Treewidth can be defined in several seemingly unrelated ways, e.g., as the minimum k for which a given graph is a *partial k-tree*, as the *induced width* (also called the *dimension*), or as the *elimination width* [29, 5]. An *elimination ordering* π of a graph G is an ordering of $V(G)$. The *induced width of a vertex* $v \in V(G)$ in the ordering is the number of its neighbors at the time it is being removed in the following process: Start with $\pi(1)$, add an edge for each pair of its neighbors that were previously not adjacent, remove $\pi(1)$, then repeat this procedure with the next vertex in the ordering. The *width of* π is the maximum induced width of a vertex, and the *induced width of* G is the minimum width of an elimination ordering. It is known that the induced width of a graph is precisely its treewidth [5].

It follows straightforwardly from the definition of treewidth that if G is obtained from G' by removing a degree 1 vertex, then $\mathrm{tw}(G) = \mathrm{tw}(G')$, unless G' has only 1 edge, in which case $\mathrm{tw}(G) = 0$ and $\mathrm{tw}(G') = 1$. We will also use the following well-known and simple fact, a proof for which is provided in the appendix.

PROPOSITION 2.1. *Let G be a simple undirected graph, and w be a degree* 2 *vertex. Then removing w and connecting its two adjacent vertices does not change the treewidth.*

Quantum circuits. We review some basic concepts of quantum mechanics and quantum computation. For a more detailed treatment, we refer the readers to the book by Nielsen and Chuang [23].

The state space of one qubit is denoted by $\mathcal{H} \stackrel{\text{def}}{=} \mathbb{C}^2$. We fix an orthonormal basis for $\mathcal{H}$ and label the basis vectors with $|0\rangle$ and $|1\rangle$. The space of operators on a vector space V is denoted by $\mathbf{L}(V)$. The identity operator on V is denoted by I_V, or by I if V is implicit from the context. A density operator, or a mixed state, of n qubits is a positive semidefinite operator $\rho \in \mathbf{L}(\mathcal{H}^{\otimes n})$ with $\mathrm{trace}\rho = 1$. For a binary string $x = x_1 x_2 \cdots x_n \in \{0,1\}^n$, let $\rho_x \stackrel{\text{def}}{=} \bigotimes_{i=1}^n |x_i\rangle\langle x_i|$ be the density operator of the state $|x\rangle \stackrel{\text{def}}{=} \otimes_{i=1}^n |x_i\rangle$.

In this paper, a quantum gate with a input qubits and b output qubits is a superoperator $Q : \mathbf{L}(\mathcal{H}^{\otimes a}) \to \mathbf{L}(\mathcal{H}^{\otimes b})$. There are certain constraints that Q must satisfy in order to represent a physically realizable quantum operation. We need not be concerned about those constraints as our simulation method does not depend on them. In existing applications one typically has $a \geq b$ and often $a = b$, though a density operator can also be regarded as a gate with $a = 0$. The ordering of inputs and outputs is, in general, significant. If Q is a *traced out* operator, then $b = 0$, and $Q(|x\rangle\langle y|) = \langle x|y\rangle$ for all $x, y \in \{0,1\}^a$. We denote by $Q[A]$ the application of Q to an

ordered set A of a qubits.

The information in a quantum state is retrieved through the application of measurements. A POVM (positive operator-valued measure) $\mathcal{M}$ on n qubits is a set $\mathcal{M} = \{M_1, M_2, \ldots, M_k\}$, where each M_i is called a POVM element, and is a positive semidefinite operator in $\mathbf{L}(\mathcal{H}^{\otimes n})$ such that $\sum_{i=1}^{k} M_i = I$. The single-qubit measurement in the computational basis is $\{|0\rangle\langle 0|, |1\rangle\langle 1|\}$.

We assume that the maximum number of qubits on which a quantum gate can act is bounded by a constant (often two or three). A quantum circuit of size T with n input-qubits and m output-qubits consists of the following:

(1) A sequence of n input-wires, each of which represents one input-qubit, i.e., a qubit which is not the output qubit of any gate.
(2) A sequence of T quantum gates g_1, g_2, ..., g_T, each of which is applied to some subset of the wires.
(3) A sequence of m output-wires, each of which represents an output-qubit, i.e., a qubit which is not the input qubit of any gate.

Note that by the above definition, a quantum circuit C defines a function $C : \mathbf{L}(\mathcal{H}^{\otimes n}) \to \mathbf{L}(\mathcal{H}^{\otimes m})$. In most applications, a circuit C is applied to an input state $\rho_x \stackrel{\text{def}}{=} \otimes_{i=1}^{n} |x_i\rangle\langle x_i|$, for some binary string $x = x_1 \cdots x_n \in \{0,1\}^n$, and at the end of the computation, measurements in the computational basis are applied to a subset of the qubits. We shall restrict our discussions to such a case, though our results can be extended to more general cases.

The graph of a quantum circuit C, denoted by G_C, is obtained from C as follows. Regard each gate as a vertex, and for each input/output wire add a new vertex to the open edge of the wire.[1] Each wire segment can now be represented by an edge in the graph.

3. Tensors and tensor networks. Tensors, commonly used in physics, are multidimensional matrices that generalize more traditional tools from linear algebra, such as matrix products. Here we focus on features of tensors that are relevant to our work.

DEFINITION 3.1. *A rank-k tensor in an m-dimension space $g = [g_{i_1,i_2,\ldots,i_k}]_{i_1,i_2,\ldots,i_k}$ is an m^k-dimensional array of complex numbers $g_{i_1,i_2,\ldots,i_k}$, indexed by k indices, i_1, i_2, ..., i_k, each of which takes m values. When the indices are clear we omit them outside the bracket.*

For example, a rank-0 tensor is simply a complex number, and a rank-1 tensor is a dimension-m complex vector. We focus on dimension-4 tensors, and set the range of each index to be $\Pi \stackrel{\text{def}}{=} \{|b_1\rangle\langle b_2| : b_1, b_2 \in \{0,1\}\}$. We fix the following tensor representation of a density operator and a superoperator.

DEFINITION 3.2. *Let ρ be a density operator on a qubits. The tensor of ρ is $[\rho_{\sigma_1,\sigma_2,\ldots,\sigma_a}]_{\sigma_1,\sigma_2,\ldots,\sigma_a \in \Pi}$, where*

$$\rho_{\sigma_1,\ldots,\sigma_a} \stackrel{\text{def}}{=} tr(\rho \cdot (\otimes_{i=1}^{a} \sigma_i)^\dagger).$$

Let Q be a superoperator acting on a input qubits and b output qubits. The tensor of Q is

$$Q_{\sigma_1,\sigma_2,\ldots,\sigma_a,\tau_1,\tau_2,\ldots,\tau_b}]_{\sigma_1,\ldots,\sigma_a,\tau_1,\ldots,\tau_b \in \Pi},$$

[1] These vertices are going to represent input states, as well as measurements and trace-out operators at the end of the computation.

where

$$Q_{\sigma_1,\sigma_2,\ldots,\sigma_a,\tau_1,\tau_2,\ldots,\tau_b} \stackrel{\text{def}}{=} tr(Q(\otimes_{i=1}^{a}\sigma_i)\cdot(\otimes_{j=1}^{b}\tau_j)^\dagger).$$

We shall use the same notation for a density operator (or a superoperator) and its tensor. We now define the central object of this paper.

DEFINITION 3.3. *A* tensor network *is a collection tensors, each index of which may be used by either one or two tensors.*

A rank-k tensor g can be graphically represented as a vertex labeled with g, and connected to k open wires, each of which is labeled with a distinct index. We may represent a tensor network by starting with such graphical representations of its tensors, and then connecting wires corresponding to the same index. Note that now each wire corresponds to a distinct index. Also, an index that appears in one tensor corresponds to an open wire, and an index that appears in two tensors corresponds to an edge connecting two vertices. Parts (a) and (b) in Figure 2 give an example of the graphical representation of a tensor and a tensor network. In the tensor g_Q, we call the σ_i wires, $1 \le i \le a$, input wires, and the τ_j wires, $1 \le j \le b$, the output wires.

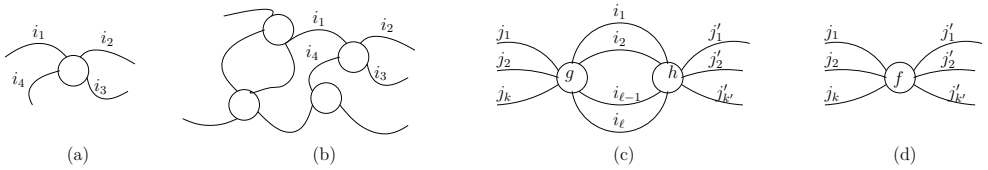

FIG. 2. *A rank-4 tensor is illustrated in* (a)*, and a tensor network with four tensors is shown in* (b)*. Contraction of two tensors is illustrated in* (c) *and* (d).

Suppose in a tensor network there are ℓ parallel edges i_1, i_2, …, i_ℓ between two vertices $g = [g_{i_1,\ldots,i_\ell,j_1,\ldots,j_k}]$ and $h = [h_{i_1,\ldots,i_\ell,j'_1,\ldots,j'_{k'}}]$. We may contract those edges by first removing them, then merging v_g and v_h into a new vertex v_f, whose tensor is $f = [f_{j_1,\ldots,j_k,j'_1,\ldots,j'_{k'}}]$, and

$$f_{j_1,\ldots,j_k,j'_1,\ldots,j'_{k'}} \stackrel{\text{def}}{=} \sum_{i_1,i_2,\ldots,i_\ell} g_{i_1,\ldots,i_\ell,j_1,\ldots,j_k} \cdot h_{i_1,\ldots,i_\ell,j'_1,\ldots,j'_{k'}}. \tag{1}$$

Parts (c) and (d) in Figure 2 illustrate the above contraction. Note that a tensor network with k open wires can be contracted to a single tensor of rank k, and the result does not depend on the order of contractions. The following example is instructive.

Example 1. Let ρ be an a-qubit density operator and Q be a superoperator with a input qubits and b output qubits. Consider the tensor network that connects all wires of the tensor ρ to the input wires of the tensor Q. Then contracting this tensor network gives the tensor of the density operator $Q(\rho)$. Figure 3 illustrates this example.

A quantum circuit C can be naturally regarded as a tensor network $N(C)$: each gate is regarded as the corresponding tensor. The qubit lines are wires connecting the tensors, or open wires that correspond to the input and output qubits. Figure 4 illustrates the concept.

Let C be a quantum circuit with n input qubits and m output qubits. Suppose that C is applied to the initial state ρ_x, for some $x \in \{0,1\}^n$. We are interested in

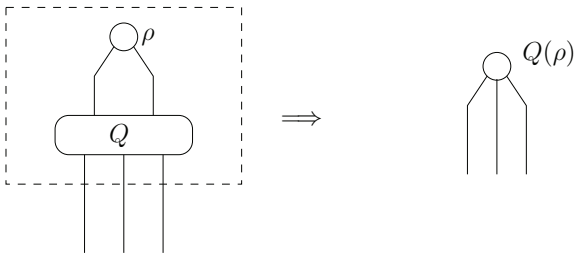

FIG. 3. *Contracting the wires connecting the tensors for a density operator ρ and a gate Q results in the tensor for $Q(\rho)$.*

knowing the probability of observing some particular outcome when some single-qubit measurements are applied to a subset of the qubits. The setting can be described by a measurement scenario defined as follows.

DEFINITION 3.4. *Let $m \geq 1$ be an integer. A* measurement scenario *on m qubits is a function $\tau : [m] \to \mathbf{L}(\mathbb{C}^2)$, such that $\tau(i)$ is a single-qubit POVM measurement element.*

Note that if a qubit i is not to be measured, then we can set $\tau(i) = I$.

To compute the probability that τ is realized on $C(\rho_x)$, we build a tensor network $N(C; x, \tau)$ from $N(C)$ by attaching to each input open wire i the tensor for $|x_i\rangle\langle x_i|$, and attaching to each open wire for the output qubit i the tensor for $\tau(i)$. When $x = 0^n$, we abbreviate $N(C; x, \tau)$ as $N(C; \tau)$. Figure 4 illustrates the concept of $N(C)$ and $N(C; \tau)$.

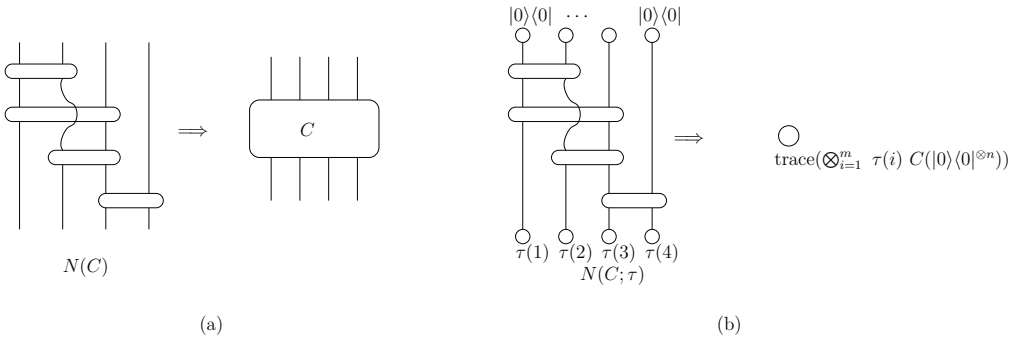

FIG. 4. *In* (a)*, a circuit C can be naturally regarded as a tensor network $N(C)$. Contracting $N(C)$ gives the tensor for the operator that C realizes. Part* (b) *illustrates the tensor network $N(C; \tau)$, contracting which gives the rank-0 tensor whose value is precisely the probability that the measurement scenario τ is realized on $C(|0\rangle\langle 0|^{\otimes n})$.*

PROPOSITION 3.5. *Let C be a quantum circuit, x be a binary string, and τ be a measurement scenario. Contracting the tensor network $N(C; x, \tau)$ to a single vertex gives the rank-0 tensor which is the probability that τ is realized on $C(\rho_x)$.*

Proof. Let $\rho^t \stackrel{\text{def}}{=} g_t g_{t-1} \cdots g_1(\rho_x)$, $1 \leq t \leq T$, and $\rho^0 = \rho_x$. By the definitions of tensors for density operators and superoperators and tensor contraction, contracting wires connecting the tensor of a superoperator Q and the tensors for a density operator ρ gives the tensor of $Q(\rho)$. Thus sequentially contracting input wires of $g_1, \ldots, g_t$ gives the tensor for ρ^t, and contracting the remaining wires gives the tensor for $\tau(\rho^T)$, which is the probability of realizing τ on $\rho^T = C(\rho_x)$. ∎

We remark that $N(C; x, \tau)$ is not the only tensor network for which the previous proposition holds.

Although the ordering of the edges in the contraction process does not affect the final tensor, it may significantly affect space and time requirements.

PROPOSITION 3.6. *Given a tensor network N of a size T quantum circuit, and a contraction process specified by an ordering of wires in N, let d be the maximum rank of all the tensors that appear in the process. Then the contraction takes $O(T \exp[O(d)])$ time.*

Proof. Note that the size of N is $\Theta(T)$. The algorithm stores the tensors of each vertex. When contracting an edge, it computes the new tensor according to (1) and updates the tensor accordingly. This takes $\exp[O(d)]$ time. Hence the total runtime is $O(T \exp[O(d)])$. ∎

In the next section we will investigate near-optimal orderings for simulation and ways to find them. While traditional simulation of quantum circuits proceeds in the same order in which the gates are applied, it appears that an optimal ordering may not have any physical meaning. Therefore, we formalize this optimization using abstract graph contractions.

4. Contraction complexity and treewidth. Let G be a graph with vertex set $V(G)$ and edge set $E(G)$. Recall that the contraction process discussed in the previous section removes parallel edges in one step because contracting one edge at a time can create multiple loops. However, for future convenience we prefer the latter simulation and therefore allow loops to remain uncontracted, counting toward the degree of a vertex. Note that if a "parallel" contraction contracts ℓ edges between two vertices u and v of degrees $\ell + k$ and $\ell + k'$, respectively, the corresponding "one-edge-at-a-time" contraction would create vertices of degrees $k+k'+\ell-1$, $k+k'+\ell-2$, ..., $k+k'$, each of which is $\leq d(u) + d(v)$. Thus the one-edge-at-a-time contraction process can emulate the parallel contraction, while increasing the maximum vertex degree observed by no more than twofold. We make the definition of this new contraction process precise below.

DEFINITION 4.1. *The* contraction of an edge *e removes e and replaces its end vertices (or vertex) with a single vertex. A* contraction ordering *π is an ordering of all the edges of G, $\pi(1)$, $\pi(2)$, ..., $\pi(|E(G)|)$. The complexity of π is the maximum degree of a merged vertex during the contraction process. The* contraction complexity *of G, denoted by $\mathrm{cc}(G)$, is the minimum complexity of a contraction ordering.*

Since only the degrees of the merged vertices are considered in defining the contraction complexity, $\mathrm{cc}(G)$ could be strictly larger than $\Delta(G)$. For example, if G is a path, then $\mathrm{cc}(G) = 1$ and $\Delta(G) = 2$.

Note that sequentially contracting all $\pi(i)$, $1 \leq i \leq |E(G)|$, reduces G to a single vertex (or an empty graph of several vertices). Also, for any graph G, $\mathrm{cc}(G) \leq |E(G)| - 1$, since any merged vertex would be incident to no more than $|E(G)| - 1$ number of edges. Furthermore, $\mathrm{cc}(G) \geq \Delta(G) - 1$, since when an edge incident to a vertex of degree $\Delta(G)$ is removed, the resulting merged vertex is incident to at least $\Delta(G) - 1$ edges.

The nature of $\mathrm{cc}(G)$ becomes clearer once we consider the *line graph* of G, denoted by G^*. That is, the vertex set of G^* is $V(G^*) \stackrel{\text{def}}{=} E(G)$, and the edge set is

$$E(G^*) \stackrel{\text{def}}{=} \{e_1, e_2\} \subseteq E(G) : e_1 \neq e_2, \exists v \in V(G)$$

such that e_1 and e_2 are both incident to v.

PROPOSITION 4.2. *For any graph $G = (V, E)$, $\mathrm{cc}(G) = \mathrm{tw}(G^*)$. Furthermore, given a tree decomposition of G^* of width d, there is a deterministic algorithm that outputs a contraction ordering π with $\mathrm{cc}(\pi) \leq d$ in polynomial time.*

Computing the treewidth of an arbitrary graph is NP-hard [6], but we do not know if this remains true for the special class of graphs G^*. Nevertheless, this is not critical in our work since the constant-factor approximation due to Robertson and Seymour [28] suffices for us to prove our key results.

THEOREM 4.3 (see Robertson and Seymour [28]). *There is a deterministic algorithm that, given a graph G, outputs a tree decomposition of G of width $O(\mathrm{tw}(G))$ in time $|V(G)|^{O(1)} \exp[O(\mathrm{tw}(G))]$.*

Proof of Proposition 4.2. There is a one-to-one correspondence of the contraction of an edge in G and the elimination of a vertex in G^*, and the degree of the merged vertex resulting from contracting an edge e in G is the same as the degree of e being eliminated in G^*. Thus $\mathrm{cc}(G) = \mathrm{tw}(G^*)$.

To prove the second part of the statement, denote the tree decomposition by $\mathcal{T}$. Repeat the following until the tree decomposition becomes an empty graph. Choose a leaf ℓ in $\mathcal{T}$. If ℓ is the single vertex of $\mathcal{T}$, then output vertices (of G^*) in B_ℓ in any order. Otherwise, let ℓ' be its parent. If $B_\ell \subseteq B_{\ell'}$, then remove ℓ and repeat this process. Otherwise, let $e \in B_\ell - B_{\ell'}$. Then, output e, remove it from the tree decomposition, and continue the process until all vertices of the tree decomposition are removed. The number of steps in this process is polynomial in the size of the tree decomposition.

Note that each output e appears in only one bag in the tree decomposition. Therefore, all (current) neighbors of e must appear in the same bag. Hence its induced width is at most d, by the one-to-one correspondence of the vertex elimination in G^* and the contraction process in G, $\mathrm{cc}(\pi) \leq d$. ∎

Before we complete the description of our simulation algorithm, we relate the treewidth of G to that of G^*. This is useful for reasoning about quantum circuits C when the graph G_C is easier to analyze than its line graph G_C^*. In such cases one hopes to bound the runtime of the simulation algorithm in terms of parameters of G rather than G^*. Fortunately, since G_C is of bounded degree, the treewidths of G_C and G_C^* are asymptotically the same.

LEMMA 4.4. *For any graph G of maximum degree $\Delta(G)$,*

$$(\mathrm{tw}(G) - 1)/2 \leq \mathrm{tw}(G^*) \leq \Delta(G)(\mathrm{tw}(G) + 1) - 1.$$

Proof. From a tree decomposition $\mathcal{T}$ of G of width d we obtain a tree decomposition $\mathcal{T}^*$ of G^* of width $(d+1) \cdot \Delta(G) - 1$ by replacing each vertex $v \in V(G)$ with all edges e incident to v. This guarantees that every edge of G^* is in some bag, i.e., (T1) is true. Item (T2) is true since if e_1 and e_2 are both incident to a vertex u in G, then any bag in $\mathcal{T}$ containing u contains both e_1 and e_2 in $\mathcal{T}^*$. To verify item (T3), suppose that e connects u and v in $V(G)$. Take two bags a and b that both contain e. Then, in $\mathcal{T}$, both bags a and b must have either u or v. If they contain the same vertex, then a and b are connected, by (T3). Otherwise, there must be a bag c that contains both u and v, by (T2). So a and b are connected through c. Therefore we have proved that $\mathrm{tw}(G^*) \leq \Delta(G)(\mathrm{tw}(G) + 1) - 1$.

Now to prove $\mathrm{tw}(G) \leq 2\mathrm{tw}(G^*) + 1$, we start with a tree decomposition $\mathcal{T}^*$ of G^* of width d, and replace every e by its two endvertices in $V(G)$. The verification of (T1) through (T3) can be accomplished in a similar way. ∎

Note that the above bounds are asymptotically tight, since for an m-ary tree (of which each nonroot internal vertex has degree $m + 1$), the treewidth is 1 and the

contraction complexity is m. We summarize the previous finding in the following theorem.

THEOREM 4.5. *Let $d \geq 1$ be an integer. For any family of graphs G_n, $n \in \mathbb{N}$, such that $\Delta(G_n) \leq d$ for all n. Then*

$$(\mathrm{tw}(G_n) - 1)/2 \leq \mathrm{cc}(G_n) = \mathrm{tw}(G_n^*) \leq d(\mathrm{tw}(G_n) + 1) - 1 \quad \forall n \in \mathbb{N}.$$

We are now ready to put everything together to prove the following restatement of Theorem 1.1.

THEOREM 4.6. *Let C be a quantum circuit of size T and with n input and m output qubits, $x \in \{0,1\}^n$ be an input, and $\tau : [m] \to \mathbf{L}(\mathbb{C}^2)$ be a measurement scenario. Denote by G_C the underlying circuit graph of C. Then the probability that τ is realized on $C(\rho_x)$ can be computed deterministically in time $T^{O(1)} \exp[O(\mathrm{cc}(G_C))] = T^{(1)} \exp[O(\mathrm{tw}(G_C))]$.*

Proof. The following algorithm computes the desired probability.

(1) Construct $N = N(C; x, \tau)$.
(2) Apply the Robertson–Seymour algorithm to compute a tree decomposition $\mathcal{T}$ of N^* of width $w = O(\mathrm{tw}(N^*))$ (Theorem 4.3).
(3) Find a contraction ordering π from $\mathcal{T}$ (Proposition 4.2) of width w.
(4) Contract N using π, and output the desired probability from the final (rank-0) tensor (Proposition 3.5).

The runtime bottlenecks are steps (2) and (4), which can be combined, and both take time $T^{O(1)} \exp(O[\mathrm{tw}(N^*)])$, which by Theorem 4.5 is $T^{O(1)} \exp[O(\mathrm{cc}(G_C))] = T^{O(1)} \exp[O(\mathrm{tw}(G_C))]$; for the sake of clarity we separate both steps . ∎

5. Treewidth and quantum circuits. In this section we prove the implications of Theorem 1.1 stated in the introduction. A number of tight bounds for the treewidth of specific families of graphs have been published, including those for planar and series-parallel graphs. However, similar results for graphs derived from quantum circuits are lacking. To this end, we strengthen Corollary 1.5 as follows.

PROPOSITION 5.1. *Let C be a quantum circuit in which each gate has an equal number of input and output qubits, and whose qubits are index by $[n]$, for an integer $n \geq 1$. Suppose that the size of C is T, and r is the minimum integer so that for any i, $1 \leq i \leq n-1$, no more than r gates act on some qubits j and j' with $j \leq i < j'$. Then C can be simulated deterministically in time $T^{O(1)} \exp[O(r)]$.*

Corollary 1.5 follows since $r = O(qD)$ under its assumption.

Proof of Proposition 5.1. Assume without loss of generality that $\mathrm{tw}(G_C) \geq 2$. Let G be the graph obtained from G_C by removing degree 1 vertices and contracting edges incident to degree 2 vertices. Then $\mathrm{tw}(G) = \mathrm{tw}(G_C)$, by Proposition 2.1 and the observation stated before it. Then each vertex in G corresponds to a multiqubit gate in C.

We now construct a tree decomposition $\mathcal{T}$ for G that forms a path of $n-1$ vertices B_1–B_2–$\cdots$–B_{n-1}. The bag B_i of the ith vertex ($1 \leq i \leq n-1$) consists of multiqubit gates (vertices) that act on some qubits j and j' with $j \leq i < j'$. Hence $|B_i| \leq r$ by the assumption. If u acts on qubits $i_1, i_2, \ldots, i_k$, $i_1 < i_2 < \cdots < i_k$, then $u \in B_i$ for all i, $i_1 \leq i \leq i_k$. Thus (T1) and (T3) are true. If a wire segment corresponding to the qubit i connects two gates u and v, then the bag B_i contains both u and v. Thus (T2) is true. Therefore $\mathcal{T}$is a tree decomposition for G with width $r-1$. Hence $\mathrm{tw}(G_C) = \mathrm{tw}(G) = O(r)$, which by Theorem 1.1 implies that C can be simulated in $T^{O(1)} \exp[O(r)]$ time. ∎

We now turn to quantum circuits of bounded depth. To prove Theorem 1.7 we will make use of the following observation that relates expander graphs to contraction complexity. Let d be a constant and $\{G_n\}_{n\in\mathbb{N}}$ be a family of d-regular graphs, and $\epsilon > 0$ be a universal constant. Recall that $\{G_n\}$ is called a family of expander graphs with expansion parameter ϵ if, for any subset $S \subseteq V(G_n)$ with $|S| \le |V(G_n)|/2$, there are no less than $\epsilon|S|$ edges connecting vertices in S with vertices in $V(G) - S$.

LEMMA 5.2. *For an expander graph G_n with the expansion parameter ϵ,* $\mathrm{cc}(G_n) \ge \epsilon|V(G_n)|/4$.

Proof. Fix a contraction ordering of G_n. Let v be the first merged vertex so that k_v, the number of vertices in $V(G_n)$ that were eventually merged to v, is at least $|V(G_n)|/4$. Then $k_v \le |V(G_n)|/2$, and v must have a degree $\epsilon|V(G_n)|/4$. ∎

The following graph is shown to be an expander by Lubotzky, Phillips, and Sarnak [18]. Let $p > 2$ be a prime, and G_n be the graph with $V(G_p) \stackrel{\mathrm{def}}{=} \mathbb{Z}_p \cup \{\infty\}$. Every vertex x is connected to $x+1$, $x-1$, and x^{-1} ($\infty \pm 1$ are defined to be ∞). Note that G_p is a 3-regular graph.

Proof of Theorem 1.7. By Lemma 5.2, $\mathrm{cc}(G_p) = \Omega(p)$. Since G_p is a 3 regular graph, $\mathrm{tw}(G_p) = \Theta(\mathrm{cc}(G_p)) = \Omega(p)$, by Theorem 4.5. Let G'_p be the graph obtained from G_p by removing the vertex ∞ and the edge $\{0, p-1\}$. This would only decrease $\mathrm{tw}(G_p)$ by at most constant. Hence $\mathrm{tw}(G'_p) = \Omega(p)$. Therefore to prove the theorem, it suffices to construct a quantum circuit C on p qubits so that G'_p is a minor of G^*_C.

Each qubit of C corresponds to a distinct vertex in $V(G'_p)$. Observe that edges in $E(G'_p)$ can be partitioned into three vertex-disjoint subsets: (1) $\{x, x^{-1}\}$; (2) $\{x, x+1\}$ for even x, $0 \le x \le p-3$; (3) the remaining edges. Each subset gives a layer of two-qubit gates in C. In G^*_C, contracting all of the vertices that correspond to the same qubit gives a graph of which G'_p is a minor. Hence $\mathrm{tw}(C) = \Theta(\mathrm{tw}(G^*_C)) = \Omega(p)$. ∎

Proof of Theorem 1.8. By Theorem 4.5, it suffices to prove that $\mathrm{cc}(G_C) = O(1)$ for any depth-2 circuit. Observe that for any such circuit, after contracting the input and output vertices (those are of degree 1, hence contracting them will not increase the contraction complexity), every vertex in G_C has degree either 1 or 2. Hence the edges can be decomposed into disjoint paths and cycles, which can be contracted without increasing the degree. Hence $\mathrm{cc}(G_C) \le 2$. ∎

6. Simulating one-way quantum computation. This section revisits the notions of *graph states* and *one-way quantum computation*. We simulate one-way computation with an algorithm whose complexity grows exponentially with the contraction complexity of the underlying graph.

Let $G = (V, E)$ be a simple undirected graph with $|V| = n$. For a subset $V' \subseteq V$, denote by $e(V')$ the number of edges in the subgraph induced by V'. We associate a qubit with each vertex $v \in V$ and refer to it by qubit v. For a subset $V' \subseteq V$, we identify the notation $|V'\rangle$ with the computational basis $|x\rangle$, for $x \in \{0,1\}^n$ being the characteristic vector of V' (i.e., the ith bit of x is 1 if and only if the ith vertex under some fixed ordering is in V'). The graph state $|G\rangle$ is the following n-qubit quantum state [7]

$$|G\rangle \stackrel{\mathrm{def}}{=} \frac{1}{\sqrt{2^n}} \sum_{V' \subseteq V} (-1)^{e(V')} |V'\rangle.$$

Note that $|G\rangle$ can be created from $|0^n\rangle$ by first applying Hadamard gates to all qubits, followed by the controlled-phase gate $\Lambda(\sigma^z) = \sum_{b_1,b_2\in\{0,1\}} (-1)^{b_1 \cdot b_2} |b_1, b_2\rangle\langle b_1, b_2|$ on each pair of qubits u and v with $\{u, v\} \in E$. Since all of the $\Lambda(\sigma^z)$ operators commute, the order of applying them does not affect the result.

A basic building block of our simulation algorithm is the following.

LEMMA 6.1. *Let $G = (V, E)$ be a graph with n vertices, and let τ be a measuring scenario (defined in Definition 3.4) on n qubits. Then the probability p that τ is realized on $|G\rangle$ can be computed deterministically in time $O(|V|^{O(1)} \exp[O(\mathrm{cc}(G))])$.*

Proof. Fix a circuit C_G that creates $|G\rangle$ from $|0\rangle\langle 0|^{\otimes n}$. Let $\{u, v\} \in E$ and $g = g_{u^+,u^-,v^+,v^-}$ be a tensor in $N(C_G; \tau)$ corresponding to $\Lambda(\sigma^z)[u, v]$. The wires representing the qubit u (or v) before and after the gate are labeled u^+ (or v^+) and u^- (or v^-), respectively. We replace g by two tensors $g^u = g^u_{u^+,u^-,t^+,t^-}$ and $g^v = g^v_{v^+,v^-,t^+,t^-}$, which share two labels t^+ and t^- and are defined as follows. For a wire segment with a label a, denote by $\mathbf{L}_a$ the 4-dimensional space of linear operators associated with this wire segment. Set g^u to be the identity superoperator that maps $\mathbf{L}_{u^+} \otimes \mathbf{L}_{t^-} \to \mathbf{L}_{t^+} \otimes \mathbf{L}_{u^-}$, and g^v to be the tensor for a $\Lambda(\sigma^z)$ that maps $\mathbf{L}_{t^+} \otimes \mathbf{L}_{v^+} \to \mathbf{L}_{t^-} \otimes \mathbf{L}_{v^-}$. By their definitions, contracting g^u and g^v gives precisely g. We call the inserted wires labeled with t^+ and t^- *transition wires*. See Figure 5 for an illustration.

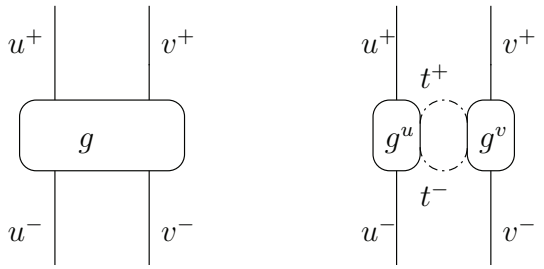

FIG. 5. *Replacing a tensor g corresponding to $\sigma_z[u, v]$ by two tensors g^u and g^v.*

Denote by $N'(C_G; \tau)$ the tensor network obtained from $N(C_G; \tau)$ by applying the above replacement procedure for each edge in E. Let G' be the underlying graph of $N'(C_G; \tau)$. Note that G' has the maximum degree 4 and the number of vertices is $O(|E|)$. See Figure 6 for an illustration. Thus p can be computed by contracting $N'(C_G; \tau)$ in time $O(|V|^{O(1)} \exp[O(\mathrm{cc}(G'))])$, according to Theorem 4.6.

We now prove that $\mathrm{cc}(G') = O(\mathrm{cc}(G))$. This can be seen by contracting all wire segments corresponding to the same qubit in G', while leaving the transition wires untouched. Since contracting the edge incident to an input or output vertex results in a new vertex of degree 3, and contracting the rest of the wires for a qubit v results in a new vertex of degree $2d(v)$, the maximum degree of a merged vertex in this process is $\max\{3, 2\Delta(G)\}$. The one-to-one correspondence between the resulting vertex set and V induces naturally a one-to-one correspondence between the pairs of transition wires and E. Thus a contraction ordering of G gives a contraction ordering of G' (of this stage) with at most twice the contraction complexity. Therefore

$$\mathrm{cc}(G') \le \max\{3, 2\Delta(G), 2\mathrm{cc}(G)\} = O(\mathrm{cc}(G) + 1).$$

Thus p can be computed deterministically in time $O(|V|^{O(1)} \exp[O(\mathrm{cc}(G'))])$ $= O(|V|^{O(1)} \exp[O(\mathrm{cc}(G))])$. ∎

A *one-way* computation on a quantum state $|\phi\rangle$ consists of a sequence of adaptive single-qubit measurements and single-qubit unitary operations applied to $|\phi\rangle$. The description of each measurement or unitary operation, including the index of the qubit that it acts on, can be computed by a deterministic and efficient (polynomial

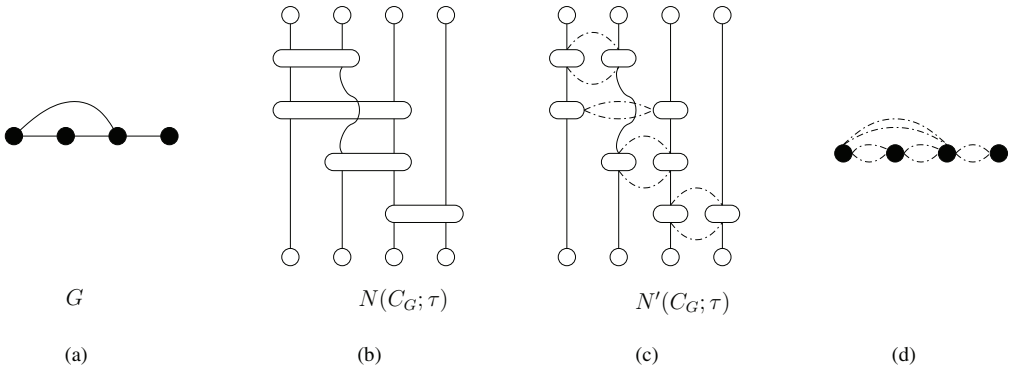

FIG. 6. *For a graph G in* (a)*, the tensor network $N(C_G;\tau)$ is shown in* (b)*. Input vertices are at the top, and output vertices are at the bottom. Each box is a tensor corresponding to a $\Lambda(\sigma_z)$ applied to qubits adjacent in G. In* (c)*, each $\Lambda(\sigma_z)$ tensor is replaced by two tensors and two wires connecting them, as described in Figure* 5*. Contracting all solid lines in* (c) *produces the graph in* (d)*, which is precisely G with each edge doubled.*

time) algorithm from previous operations and their measurement outcomes. In our discussion we treat this computation time as a constant. We call a one-way quantum computation *oblivious* if before the last measurement (which produces the outcome of the computation), different computational paths involve the same number of measurements, take place with the same probability, and result in an identical state. Note that the one-way computation of Raussendorf and Briegel [25] is oblivious.

We point out that allowing single-qubit unitary operations in the definition is for the convenience of discussion only, since each single-qubit unitary can be combined with a future measurement on the same qubit (should there be one). To see this fact, let us call two quantum states *LU-equivalent* (where LU stands for local unitary), if there exists a set of single-qubit unitary operations applying which maps one state to the other. A one-way computation with unitary operators always has an almost identical one-way computation without unitary operations: the measurements are in one-to-one correspondence with identical outcome distributions, and the states after corresponding measurements are LU-equivalent. Therefore, when we are only interested in the distribution of the measurement outcomes, we may assume without loss of generality that a one-way computation does not involve any unitary operation.

We now derive a simulation algorithm whose complexity depends on the contraction complexity.

THEOREM 6.2. *A one-way quantum computation on a graph $G = (V, E)$ can be simulated by a randomized algorithm in time $O(|V|^{O(1)} \exp[O(\mathrm{cc}(G))])$. If the one-way computation is oblivious, then the simulation can be made deterministic.*

Theorem 1.6 follows immediately from the combination of the above theorem with Theorem 4.5.

Proof of Theorem 6.2. Let T be the number of measurements during the one-way computation. Assume without loss of generality that no single-qubit unitary operation is applied. The simulation consists of T steps, one for each single-qubit measurement. It maintains a data structure $r = (\tau, p)$, where τ is a measurement scenario, and p is the probability that τ is realized on $|G\rangle$. Denote by $r_t = (\tau_t, p_t)$ the value of r when t measurements have been simulated. Initially $\tau_0(i) = I$ for all i, $1 \le i \le n$, and $p_0 = 1$.

Suppose we have simulated the first $t-1$ measurements, $1 \le t \le T-1$.

(1) Based on the one-way algorithm, compute from τ_{t-1} the description of the tth measurement $P_t = \{P_t^0, P_t^1\}$ and the qubit a_t that it acts on. Denote by τ_t^0 the measurement scenario identical to τ_{t-1}, except that $\tau_t^0(a_t) = P_t^0$.
(2) Compute p_t^0, the probability of realizing τ_t^0. By Lemma 6.1, this step takes $O(|V|^{O(1)} \exp[O(\mathrm{cc}(G))])$ time.
(3) Flip a coin that produces 0 with probability p_t^0/p_{t-1}, resulting in an outcome $b_t \in \{0,1\}$. Set τ_t to be identical to τ_{t-1}, except that $\tau(a_t) = p_t^{b_t}$. Set $p_t = (1-b_t)p_t^0 + b_t(p_{t-1} - p_t^0)$. Continue the simulation until $t = T$.

By construction, the output distribution is identical to that of the one-way computation. The complexity of the algorithm is $O(|V|^{O(1)} \exp[\mathrm{cc}(G)])$.

If the one-way computation is oblivious, then there is no need to adaptively simulate the first $T-1$ measurements, as all of them lead to the same state with the same probability p_{T-1}. Let τ_{T-1} (τ_T) be the measurement scenario corresponding to the first $T-1$ (T, respectively) measurements giving the outcome 0. We compute the probabilities p_{T-1} and p_T that τ_{T-1} and τ_T are realized. Then the probability that the one-way computation produces 0 is precisely p_T/p_{T-1}. The computation is deterministic and takes $|V|^{O(1)} \exp[O(\mathrm{cc}(G))]$ time by Lemma 6.1. ∎

7. Discussion. In this work we studied quantum circuits regardless of the types of gates they use, but with a focus on how the gates are connected. We have shown that quantum circuits that look too similar to trees do not offer significant advantage over classical computation. More generally, when solving a difficult classical problem on a quantum computer, one encounters an *inherent* trade-off between the treewidth and the size of quantum circuits for solving it—the smaller the quantum circuit, the more topologically sophisticated it must be. Investigating such trade-offs for specific problems of interest is an entirely open and very attractive avenue for future research. Similar considerations may apply to classical circuits. We conjecture that there are simple functions, such as modular exponentiation, whose circuit realizations require large treewidth.

Furthermore, our work raises an intriguing possibility that the treewidth of some quantum circuits may be systematically reduced by restructuring the circuit, while preserving the final result of the entire computation. Perhaps future research in this direction can clarify the limits to efficient quantum computation, while the tools developed in this context will be useful for practical tasks.

The preprint of this paper [19] has lead to several follow-up results. Jozsa [15] and Aharonov, Landau, and Makowsky [3] gave alternative proofs for some of our theorems. Furthermore, Aharonov, Landau, and Makowsky [3], and Yoran and Short [39] pointed out that quantum Fourier transform (QFT) over $\mathbb{Z}_n$ admits approximate circuit realizations that, viewed as tensor networks, have small treewidth. Given the central role of QFT in known quantum algorithms, their results are somewhat unexpected and their implications are yet to be fully explored. For example, what type of circuits would remain efficiently simulatable when interleaved with QFT circuits? In general, as implied by Theorems 1.7 and 1.8, the treewidth of a circuit may increase dramatically under composition. Yoran and Short [39] have shown that this drawback may be avoided in some cases. Extending their result would deepen our understanding of quantum speed-ups.

As mentioned earlier, Theorem 1.6 was improved in [19] so that any one-way quantum computation on a graph state $|G\rangle$ can be simulated deterministically in $|V(G)|^{O(1)} \exp(\mathrm{tw}(G))$ time. The proof relies on a graph-theoretical result from [20]:

for any graph G there is a graph G' such that $\Delta(G') \leq 3$, $\mathrm{tw}(G') \leq \mathrm{tw}(G) + 1$, and that G can be restored from G' by contracting a subset of edges forming a forest. A small modification turns G' into a graph G'' with $\Delta(G'') \leq 3$, $\mathrm{tw}(G'') \leq \mathrm{tw}(G) + 1$, and $|G\rangle$ can be constructed from $|G''\rangle$ through an efficient one-way computation. Since $\mathrm{cc}(G'') = O(\mathrm{tw}(G'')) = O(\mathrm{tw}(G))$ (Theorem 4.5), this implies, by Theorem 6.2, that any one-way computation on $|G\rangle$ can be simulated deterministically in time $|V(G)|^{O(1)} \exp(O(\mathrm{tw}(G)))$. The interested reader is referred to [19, 20] for details.

The important question of characterizing quantum states that are universal (or efficiently simulatable) for one-way quantum computation remains unsolved. In another follow-up thread, van den Nest et al. [22, 21] defined additional width-based parameters of quantum states and demonstrated results for those parameters similar to Theorem 1.6. It is unlikely that the set of quantum states with small width-based parameters includes all efficiently simulatable states because a set of simulatable states of high widths was identified recently by Bravyi and Raussendorf [9]. Nevertheless, it remains plausible that those width-based results and their further extensions may be part of a classification theorem that gives a complete characterization of efficiently simulatable states.

Appendix. Proof of Proposition 2.1. Recall that a minor of a graph G is a graph obtained from a subgraph of G by contracting edges. A basic property of treewidth is that it does not increase under taking minors [26].

Proof of Proposition 2.1. Let G' be the graph resulting from the contractions. Since G' is a minor of G, $\mathrm{tw}(G') \leq \mathrm{tw}(G)$ [26]. If $\mathrm{tw}(G') = 1$, then G' is a nonempty forest (otherwise, G has a triangle minor, thus $\mathrm{tw}(G) \geq 2$). Thus G is also a nonempty forest and $\mathrm{tw}(G) = 1 = \mathrm{tw}(G')$. Suppose $\mathrm{tw}(G') \geq 2$. Let $\mathcal{T}$ be a tree decomposition for G'. We obtain a tree decomposition $\mathcal{T}'$ for G by inserting a bag containing $\{u, w, v\}$, and connecting it to a bag that contains $\{u, v\}$. One can verify directly that the three conditions $(T_1 - T_3)$ that define tree decompositions hold for $\mathcal{T}'$. Since the width of $\mathcal{T}'$ is no more than that of $\mathcal{T}$, we have $\mathrm{tw}(G) \leq \mathrm{tw}(G')$. Therefore, $\mathrm{tw}(G) = \mathrm{tw}(G')$. ∎

Acknowledgments. We thank George Viamontes and John Hayes for motivating this study and helpful discussions. We are grateful to Guifré Vidal, Frank Verstraete, Ashwin Nayak, Tzu-Chieh Wei, and the anonymous reviewers for many valuable comments, including pointing out relevant previous works. Yaoyun Shi is grateful to Alexei Kitaev for introducing him to the general concept of tensor networks, and to Peng-Jun Wan for useful comments. Igor Markov is grateful to Ike Chuang for useful discussions.

REFERENCES

[1] S. Aaronson and D. Gottesman, *Improved simulation of stabilizer circuits*, Physical Rev. A, 70 (2004), p. 052328-1–052328-14.

[2] D. Aharonov, A. Kitaev, and N. Nisan, *Quantum circuits with mixed states*, in Proceedings of the 30th Annual ACM Symposium on the Theory of Computation (STOC), Dallas, TX, ACM Press, New York, 1998, pp. 20–30.

[3] D. Aharonov, Z. Landau, and J. Makowsky, *The quantum FFT can be classically simulated*, preprint: quant-ph/0611156.

[4] J. Alber, H. L. Bodlaender, H. Fernau, T. Kloks, and R. Niedermeier, *Fixed parameter algorithms for dominating set and related problems on planar graphs*, Algorithmica, 33 (2002), pp. 461–493.

[5] S. Arnborg, *Efficient algorithms for combinatorial problems on graphs with bounded decomposability—a survey*, BIT, 25 (1985), pp. 2–23.

[6] S. Arnborg, D. G. Corneil, and A. Proskurowski, *Complexity of finding embeddings in a k-tree*, SIAM J. Algebraic and Discrete Methods, 8 (1987), pp. 277–284.
[7] H. J. Briegel and R. Raussendorf, *Persistent entanglement in arrays of interacting particles*, Phys. Rev. Lett., 86 (2001), pp. 910–913.
[8] H. L. Bodlaender, *Treewidth: Characterizations, Applications, and Computations*, Technical report UU-CS-2006-041, Universiteit Utrecht, Utrecht, The Netherlands, 2006.
[9] S. Bravyi and R. Raussendorf, *On measurement-based quantum computation with the toric code states*, preprint: quant-ph/0610102.
[10] E. Broering and S. Lokam, *Width-based algorithms for SAT and Circuit-SAT (extended abstract)*, in Proceedings of the Sixth International Conference on Theory and Applications of Satisfiability Testing (SAT 2003), Santa Margherita, Italy, Lecture Notes in Comput. Sci. 2919, Springer-Verlag, 2004, pp. 162–171.
[11] R. Dechter, *Bucket elimination: A unifying framework for reasoning*, Artificial Intelligence, 113 (1999), pp. 41–85.
[12] D. Gottesman, *The Heisenberg representation of quantum computers*, in Proceedings of the 22nd International Colloquium on Group Theoretical Methods in Physics, Group 22: S. P. Corney, R. Delbourgo, and P. D. Jarvis, eds., International Press, Cambridge, MA, 1999, pp. 32–43. Long version: quant-ph/9807006.
[13] S. A. Fenner, F. Green, S. Homer, and Y. Zhang, *Bounds on the power of constant-depth quantum circuits*, in Proceedings of the 15th International Symposium on Fundamentals of Computation Theory (FCT), Lübeck, Germany, 2005, Lecture Notes in Comput. Sci. 3623, M. Liskiewicz and R. Reischuk, eds., Springer, Berlin, 2005, pp. 44–55.
[14] A. W. Joshi, *Matrices and Tensors in Physics*, Halsted Press [John Wiley & Sons], New York-London-Sydney, 1975.
[15] R. Jozsa, *On the simulation of quantum circuits*, preprint: quant-ph/0603163.
[16] R. Jozsa and N. Linden, *On the role of entanglement in quantum computational speed-up*, Lond. Proc. R. Soc. Ser. A Math. Phys. Eng. Sci., 459, 2003, pp. 2011–2032.
[17] S. Krishnaswamy, G. F. Viamontes, I. L. Markov, and J. P. Hayes, *Accurate reliability evaluation and enhancement via probabilistic transfer matrices*, in Proceedings of the Design Automation and Test in Europe (DATE'05), Munich, Germany, 2005, pp. 282–287.
[18] A. Lubotzky, R. Phillips, and P. Sarnak, *Ramanujan graphs*, Combinatorica, 8 (1988), pp. 261–277.
[19] I. L. Markov and Y. Shi, *Simulating quantum computation by contracting tensor networks*, preprint: quant-ph/0511069.
[20] I. L. Markov and Y. Shi, *Constant Degree Graph Expansions and Treewidth*, manuscript.
[21] M. van den Nest, W. Dür, G Vidal, and H J. Briegel, *Classical simulation versus universality in measurement based quantum computation*, Phys. Rev. A, 75 (2007), p. 012337.
[22] M. van den Nest, A. Miyake, W. Dür, and H. J Briegel, *Universal resources for measurement-based quantum computation*, Phys. Rev. Lett., 97 (2006), p. 150504.
[23] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information*, Cambridge University Press, Cambridge, UK, 2000.
[24] D. Porras, F. Verstraete, and J. I. Cirac, *Renormalization algorithm for the calculation of spectra of interacting quantum systems*, preprint: cond-mat/0504717.
[25] R. Raussendorf and H. J. Briegel, *A one-way quantum computer*, Phys. Rev. Lett., 86 (2001), pp. 5188–5191.
[26] N. Robertson and P. D. Seymour, *Graph minors.* II. *Algorithmic aspects of tree-width*, J. Algorithms, 7 (1986), pp. 309–322.
[27] N. Robertson and P. D. Seymour, *Graph minors.* III. *Planar tree-width*, J. Comb. Theory Series B, 36 (1984), pp. 49–64.
[28] N. Robertson and P. D. Seymour, *Graph minors.* X. *Obstructions to tree-decomposition*, Journal of Comb. Theory Ser. B, 52 (1991), pp. 153–190.
[29] D. J. Rose, *Triangulated graphs and the elimination process*, J. Math. Anal. Appl., 32 (1970), pp. 597–609.
[30] V. P. Roychowdhury and F. Vatan, *Quantum formulas: A lower bound and simulation*, SIAM J. Comput., 31 (2001), pp. 460–476.
[31] B. M. Terhal and D. P. DiVincenzo, *Classical simulation of noninteracting-fermion quantum circuits*, Phys. Rev. A, 65 (2002), pp. 32325–32334.
[32] B. M. Terhal and D. P. DiVincenzo, *Adaptive quantum computation, constant depth quantum circuits and Arthur-Merlin games*, Quantum Inf. Comput., 4 (2004), pp. 134–145.
[33] L. G. Valiant, *Quantum circuits that can be simulated classically in polynomial time*, SIAM J. Comput., 31 (2002), pp. 1229–1254.

[34] F. Verstraete and J. I. Cirac, *Renormalization algorithms for Quantum-Many Body Systems in two and higher dimensions*, preprint: cond-mat/0407066.

[35] F. Verstraete, J. J. Garcia-Ripoll, and J. I. Cirac, *Matrix product density operators: Simulation of finite-temperature and dissipative systems*, Phys. Rev. Lett., 93 (2004), p. 207204.

[36] G. Vidal, *Efficient classical simulation of slightly entangled quantum computations*, Phys. Rev. Lett., 91 (2003), p. 147902.

[37] G. Vidal, *Efficient simulation of one-dimensional quantum many-body systems*, Phys. Rev. Lett., 93 (2004), p. 040502.

[38] P. Walther, K. J. Resch, T. Rudolph, E. Schenck, H. Weinfurter, V. Vedral, M. Aspelmeyer, and A. Zeilinger, *Experimental one-way quantum computing*, Nature, 434 (2005), pp. 169–176.

[39] N. Yoran and A. Short, *Efficient classical simulation of the approximate quantum Fourier transform*, Phys. Rev. A, 76 (2007), p. 042321.

[40] A. Yao, *Quantum circuit complexity*, in Proceedings of the 34th Annual Symposium on Foundations of Computer Science, Palo Alto, CA, 1993, IEEE Comput. Soc. Press, Los Alamitos, CA, 1993, pp. 352–361.

[41] M. Zwolak and G. Vidal, *Mixed-State dynamics in one-dimensional quantum lattice systems: A time-dependent superoperator renormalization algorithm*, Phys. Rev. Lett., 93 (2004), p. 207205.