On the depth overhead incurred when running quantum algorithms on near-term quantum computers with limited qubit connectivity

Steven Herbert

1 Department of Applied Mathematics and Theoretical Physics, University of Cambridge, UK

July 6, 2018

Abstract

This paper addresses the problem of finding the depth overhead that will be incurred when running quantum algorithms on near-term quantum computers. Specifically, it is envisaged that near-term quantum computers will have low qubit connectivity: each qubit will only be able to interact with a subset of the other qubits, a reality typically represented by a qubit interaction graph in which a vertex represents a qubit and an edge represents a possible direct 2-qubit interaction (gate). Thus the depth overhead is unavoidably incurred by introducing swap gates into the quantum circuit to enable general qubit interactions. In near-term quantum computers it is reasonable to assume that all qubits will be available for use in the algorithm, termed the full-width setting. This paper proves that there exist quantum circuits where a depth overhead of $\Omega(\log n)$ must necessarily be incurred when running quantum algorithms on full-width quantum computers whose qubit interaction graph has finite degree, but that such a depth overhead is achievable.

1 Introduction

It has been shown that the quantum circuit model, in which the component qubits undergo single and 2-qubit gates from a universal set, is sufficient to express general quantum algorithms [1–3]. As quantum computers move from theory to reality, it is becoming apparent that the envisaged physical limitations on the connectivity of the component qubits in near-term quantum computers presents a significant challenge to the implementation of quantum algorithms. So it follows that finding a method to efficiently run quantum algorithms on near-term quantum computers with limited connectivity is one of the foremost priorities of designers of such devices.

The connectivity of quantum computers can be represented as a simple graph, with each vertex representing a single qubit, and each edge corresponding to a universal set of 2-qubit quantum gates, i.e., the quantum computer can operate on ‘connected’ (geometrically local) qubits (see Fig. 1). It follows that the physical limitations on the connectivity of the component qubits is manifested as the qubit connectivity graph having low degree, for example the architecture of Google’s quantum computer is a rectangular lattice of connected qubits (degree 4) [4], as is that of IBM [5, Fig. 4], whilst Rigetti’s proposed 19-qubit architecture has maximum degree 3 [6, Fig. 1]. NQIT’s quantum computer uses photonic links [7] which enables much greater connectivity than the previously mentioned architectures, but ultimately will also not be completely connected. These proposed quantum computer architectures conflict with the quantum circuit model which allows general 2-qubit interactions and thus implicitly assumes a completely connected graph, however Beals et al. demonstrate the use of swap gates, which can be inserted into the quantum circuit to enable it to be run on a quantum computer with limited qubit connectivity [8].

Therefore the central question is how to insert these swaps such that the algorithm will run on a quantum computer with limited connectivity, whilst minimising the adverse effect on the algorithm’s performance (e.g., its run-time). To answer this question, it is often helpful to consider the quantum algorithm to be decomposed into successive layers of disjoint interactions (see Fig. 2). In the literature, the typical approach
Figure 1: Fragment of quantum computer with rectangular lattice of qubits. In this case, for example, qubit Q1 and qubit Q2 can interact, however qubit Q1 cannot interact directly with qubit Q5 – such an interaction can only be achieved indirectly by first inserting a swap.

Figure 2: The beginning of a quantum circuit decomposed into four layers (with single qubit gates included for illustration). Note that decomposing a quantum circuit into layers implies that the gates have been most efficiently packed into layers.

is to use swaps to permute the qubits such that an arrangement is reached in which all of the 2-qubit gates required at the current layer can be performed on connected pairs of qubits (and repeat until the algorithm is completed). Thus the question of how to run quantum algorithms on near-term quantum computers with limited qubit connectivity is typically reduced to the question of how to emulate a completely-connected graph on a finite-degree regular graph. In particular, architectures where the graph is embedded in a finite number of spatial dimensions (a more restrictive condition than the fixed degree of the graph considered herein) has been widely addressed, for example by Fowler, Devitt and Hollenberg [9], Maslov [10] and Kutin [11] for one dimension and Pham and Svore [12] for two dimensions. More generally, Cheung, Maslov and Severini present the result that a graph embedded in k dimensional space can emulate the completely connected graph in depth $O(\sqrt[\kappa]{n})$ for an n-qubit device [13].

---

1 As explicitly addressed in Section 2, this is actually not, in general sufficient.

2 See Section 2, for specific definition of graph emulation – Definition 2.4.
However, this constitutes just one way to implement an algorithm on a quantum computer with limited connectivity: there is no requirement that the algorithm need be compiled onto the quantum computer by modifying the quantum circuit such that it consists of successively repeating swapping and gating blocks. Therefore a more general approach to bounding the performance of limited connectivity quantum computers is required to account for the possibility of performing gates at any time, which is the subject-matter of this paper. The analysis in this paper is restricted to noiseless synchronous machines, i.e., where each qubit pair simultaneously undergoes a gate or a swap in unit time—or, in other words, a layer takes unit time (note that qubits can only interact with one neighbour at a time). Additionally, only full-width quantum circuits are considered, i.e., all qubits are logical qubits used in the random algorithm. This constitutes a physically reasonable and important scenario in which any qubits associated with error correction are ‘abstracted away’, leaving just the $n$ logical qubits—all of which are available to the algorithm. Such a setting is reasonable for near-term devices, and moreover, is consistent with that considered in the aforementioned literature [9–12] and thus constitutes a useful benchmark to gauge the impact that limited qubit connectivity will have on the overall performance of quantum algorithms.

2 Definitions

To formalise the notion of the impact on the performance of a quantum algorithm by the insertion of swaps, raised in Section 1, it is necessary first to define the ‘dimensions’ of a quantum circuit.

**Definition 2.1.** The dimensions of a quantum circuit are its ‘width’, the number of component qubits, denoted $n$; and its ‘depth’, the number of layers of disjoint 2-qubit interactions, denoted $m$.

This enables the depth overhead to be formally defined:

**Definition 2.2.** The depth overhead, denoted $D$, is the maximum multiplicative factor increase in the number of time steps (layers) required to run any random quantum algorithm on a quantum computer with limited qubit connectivity with any initial qubit arrangement, compared to that of a theoretical completely connected quantum computer.

In general, the depth overhead is a function of three terms: the graph of the qubit connectivity in the quantum computer, the quantum algorithm, and the way that the quantum algorithm is compiled to run on the quantum computer with limited qubit connectivity. The fully connected graph emulation complexity is closely related to the depth overhead, as defined by Brierley [14]:

**Definition 2.3.** The fully connected graph emulation complexity, denoted $T$, is the number of time steps required to emulate the completely connected graph using a sequence of local gates, that is to send the qubit at node $a$ to $\pi(a)$ for all $a = 1, \ldots, n$ and any permutation $\pi : [1,n] \to [1,n]$.

As noted in Section 1 an achievable fully connected graph emulation complexity is typically assumed to imply an achievable depth overhead, however a further condition is required:

**Condition 2.4.** An achievable fully connected graph emulation complexity is sufficient to imply an achievable depth overhead if the graph can be decomposed into $\lfloor n/2 \rfloor$ disjoint connected pairs of qubits.

This condition explicitly requires that the qubits can be permuted such that all interactions in a layer can be undertaken simultaneously, even for the extreme case where a layer consists of $\lfloor n/2 \rfloor$ 2-qubit interactions. In general a less tight condition may apply, however this is sufficient for the analysis in this paper.

3 Main results

Let $G_{n,r}$ be the set of all $r$-regular graphs with $n$ nodes, $\zeta$ be a quantum circuit and $\chi$ be a description for how to compile any $\zeta$ to run on $G \in G_{n,r}$.

**Theorem 3.1.** $\forall r \geq 4 \forall \zeta \exists (G \in G_{n,r}, \chi), D(G, \zeta, \chi) \in \mathcal{O}(\log n)$
The parenthesised \((G \in \mathcal{G}_{n,r}, \chi)\) is used to denote the fact that there exists a graph \(G\) with the required connectivity, and an associated description for how to compile any quantum circuit. Therefore, the theorem states that, for any quantum computer in which each qubit can connect to four others, there exists a qubit connectivity graph and method to compile quantum algorithms such that the depth overhead grows at most logarithmically with the number of qubits (the circuits width).

**Proof.** Brierley [14, Theorem 1], gives a procedure for the Cyclic Butterfly graph, which achieves \(T(G) \in \mathcal{O}(\log n)\) with a width overhead of 2 (i.e., each qubit is equipped with an ancilla). Thus to prove Theorem 3.1, Brierley’s procedure is adjusted such that no ancillas are required (thus satisfying the full-width requirement), and such that Condition 2.5 is satisfied, in order that \(T(G) \in \mathcal{O}(\log n) \implies D(G) \in \mathcal{O}(\log n)\). Full details of the adjusted procedure are given in Appendix A. \(\square\)

Whilst Theorem 3.1 provides an important achievable depth overhead, regular random graph theory suggests that it may well be the case that *almost all* regular random graphs (with degree greater than two) can achieve the same asymptotic rate. To formally state this possibility as a conjecture, it is first necessary to consider that the component qubits of any graph, \(G \in \mathcal{G}_{n,r}\) are arranged in one of \(n!\) permutations (from here on denoted *network states*), thus for each, \(G\) there exists a regular graph, denoted \(G'\), with \(n!\) vertices, each corresponding to one network state in \(G\), and with connectivity corresponding to which network states are connected in a single time step (i.e., with a set of simultaneous swaps). Note that \(G'\) is regular because the connectivity of network states is a function of the permutations of edges of the underlying graph \(G\) and not the current locations of the qubits (i.e., the network state – a vertex of \(G'\)). Let \(r'\) be the degree of \(G'\), using the previous notation, \(\mathcal{G}_{n,r'}^{\prime}\) denotes the set of all \(r'\)-regular graphs with \(n!\) vertices, and let the subset containing all \(G'\) be denoted \(\mathcal{G}_{n!r'}^{\prime}\) \(^3\) This is the only unproven assumption in the following argument.

**Conjecture 3.2.** \(\forall r \geq 3 \ \forall \zeta \text{ almost all } G \in \mathcal{G}_{n,r} \exists \chi, D(G, \zeta, \chi) \in \mathcal{O}(\log n)\)

That is, rather than explicitly constructing a graph to achieve logarithmic depth overhead, it is conjectured that such a depth overhead can be achieved on *almost all* \(r\)-regular random graphs. The validity of the conjecture is conditional on an upper-bound on the diameter of *almost all* \(r'\)-regular graphs with \(n!\) vertices (i.e., *almost all* members of the set \(\mathcal{G}_{n!,r'}^{\prime}\)) also applying to *almost all* members of the subset \(\mathcal{G}_{n!,r'}^{\prime}\) (itself a measure-0 subset of \(\mathcal{G}_{n!,r'}^{\prime}\)). This is the only unproven assumption in the following argument.

**Argument.** The conjecture relies on the equivalence \(\forall G \in \mathcal{G}_{n,r}, T(G) \equiv d_{G'}\), where \(d\) is the diameter of the graph. That is, in order to emulate the fully connected graph using \(G\), it is necessary traverse at most the diameter of \(G'\).

The diameter of a random regular graph, \(G' \in \mathcal{G}_{n!,r'}^{\prime}\), is, *almost surely*, the least integer \(d_{G'}\) satisfying \((r' - 1)^{(d_{G'}-1)} \geq (2 + \epsilon)r'n! \log n!\), where \(\epsilon > 0\) \([15, \text{Section 2.4}]\), i.e., for large graphs:

\[
(r' - 1)^{(d_{G'}-1)} \geq (2 + \epsilon)r'n! \log n!
\]

\[
\implies d_{G'} - 1 \geq \frac{\log_2((2 + \epsilon)r'n! \log n!)}{\log_2(r' - 1)} \tag{1}
\]

\(^3\)Note that *almost all* \(r'\)-regular graphs having some property is also referred to as a \(r'\)-regular random graph *almost surely* having that property.

\(^4\)The requirement of no correlation is rather strong, and may not hold in reality, however the conjecture will still hold with myriad weaker conditions, the strong condition merely stated for simplicity of exposition.
Given that \( d_{G'} \) is the least integer to satisfy \( 1 \):

\[
d_{G'} - 1 \leq \frac{\log_2((2 + \epsilon)r' \log_2 n!)}{\log_2(r' - 1)} + 1
\]

\[
\implies d_{G'} \leq \frac{\log_2((2 + \epsilon)r' + \log_2 n! + \log_2 \log_2 n!)}{\log_2(r' - 1)} + 2
\]

\[
= \log_2 r' + n \log_2 n + A
\]

\[
= \frac{n \log_2(r + 1) + n \log_2 n + A}{\log_2(2^{n/4} - 1)}
\]

\[
\leq \frac{\frac{n}{4} \log_2 2}{\log_2 2}
\]

\[
\in O(\log n),
\tag{2}
\]

where \( A \in O(n) \) and \( r' \) is upper- and lower-bounded in the numerator and denominator using Lemmas B.1 and B.2 respectively, and Stirling’s approximation \( \log_2 n! = n \log_2 n - n \log_2 e + A', \) where \( A' \in O(\log_2 n) \) is also used.

To complete the argument, notice that almost all \( r \)-regular random graphs are Hamiltonian connected \( 17 \), therefore it is possible to separate the graph into disjoint connected pairs by the simple procedure of ‘pairing off’ around a Hamiltonian cycle. Thus, Condition 2.5 is satisfied, so \( T(G) \in O(\log n) \implies D(G) \in O(\log n) \), completing the argument. \( \square \)

Graph theoretic results, concerning the reachability of certain vertices from other distant vertices, are useful for proving achievable upper-bounds on the depth overhead, however this approach is insufficient for proving a lower-bound, as it neglects the possibility (at least in principle) of performing qubit interactions at any time (rather than always sorting such that an entire layer can be completed simultaneously). Furthermore, attempting to marginalise out all possible interactions throughout the process leads to an infeasible complex problem. Nevertheless, it may appear that a lower-bound of \( D(G) \in \Omega(\log n) \) is obviously true, given that the diameter of the graph must grow at least as \( \log n \). However this is not, in fact necessarily the case, because prior qubit swapping could have been such that qubits are always ‘close’ by the time they need to interact. Therefore a more general method, allowing for more general qubit ordering algorithms, is required to lower-bound the depth overhead. One such method is to treat the network as a disordered system, that must be ordered (to an extent) by performing swaps to achieve a gate, and gates in turn have a mixing effect on the network. This approach, it turns out, is sufficiently general to lower-bound the depth overhead in \( \Omega(\log n) \).

**Theorem 3.3.** \( \forall (G \in G_{n,r}, \chi) \exists \zeta, D(G, \zeta, \chi) \in \Omega(\log n) \)

That is, for any proposed method of achieving a depth overhead that grows less than logarithmically with the quantum circuit width, there is a way to construct a quantum circuit that violates this less than logarithmic depth overhead, and thus invalidates the claim that the method always achieves less than logarithmic depth overhead.

**Proof.** To prove this theorem, it is necessary to formalise the notion of gates mixing the system and swaps introducing order. A suitable description for the network is to index all vertices (which are fixed), and also index all qubits (which can be located at any vertices), and to label all qubits with their target qubit (i.e., the next qubit with which they will interact, that is jointly undergo a two-qubit gate), or with a ‘null’ label, to signify that the qubit has completed all of its interactions required in the quantum circuit. WLOG, let \( n \) be even, and consider the ensemble of quantum circuits of depth \( m \) in which all \( n \) qubits are involved in an interaction at each layer. There are \( n!2^{-n/2} \) permutations of interactions in each layer, therefore the process of re-labelling an entire layer has entropy (all entropies are expressed in bits):

\[
\Delta S_L = \log_2 n! - \frac{n}{2},
\tag{3}
\]
from which the average entropy added when a single qubit is re-labelled can be expressed:

$$\langle \Delta S_Q \rangle = \frac{1}{n} \left( \log_2 n! - \frac{n}{2} \right),$$  \hspace{1cm} (4)

which in turn can be used to express the average entropy added to the system for each interaction, noting that each interaction re-labels exactly two qubits the first of which can be at any vertex, and the second can be at any of $r$ vertices:

$$\langle \Delta S_I \rangle \geq 2 \langle \Delta S_Q \rangle - \log_2 n - \log_2 r$$

$$= \frac{2}{n} \left( \log_2 n! - \frac{n}{2} \right) - \log_2 n - \log_2 r,$$  \hspace{1cm} (5)

the terms $-\log_2 n - \log_2 r$ correspond to the aforementioned order required for an interaction to occur; specifically, the inequality arises from the minimum additional entropy case where the interaction occurs at each permissible pair of vertices with equal frequency. Considering now the system as a whole, the final state of the system is such that all of the qubits are labelled with the null label, and can be at any vertex, therefore the final entropy of the system can be bounded:

$$S_{final} \leq \log_2 n!,$$  \hspace{1cm} (6)

where the inequality again arises from the bounding case where the final arrangement of the qubits is maximally mixed. Let $N_I = mn$ be the number of interactions in each circuit, and the initial state of the system have entropy $S_{init}$, and consider the whole circuit:

$$S_{final} = S_{init} + \sum_{i=1}^{N_I} \Delta S_I^{(i)} + \Delta S(\chi).$$  \hspace{1cm} (7)

where $-\Delta S(\chi)$ is the entropy reduction achieved by the description for how to compile a circuit, hereafter referred to as ordering process for consistency with the entropic formulation. According to Definition 2.2, the depth overhead is defined for any initial qubit arrangement, therefore the ensemble must be expanded to include the initial arrangement of qubits as uniformly distributed over all $n!$ permutations. Additionally contributing to $S_{init}$ is the initial target label for each qubit, $n!/2^{-n/2}$ permutations as previously stated, thus:

$$S_{init} = 2\log_2 n! - n.$$  \hspace{1cm} (8)

Rearranging (6), recalling that $N_I = mn$, substituting in (8) and using inequalities (5) and (6):

$$-\Delta S(\chi) = \sum_{i=1}^{N_I} \Delta S_I^{(i)} + S_{init} - S_{final}$$

$$\geq (m - 1) (2\log_2 n! - n - n \log_2 n - n \log_2 r) + (2\log_2 n! - n) - \log_2 n!$$

$$= (m - 1) \left( 2n \log_2 n - 2n \log_2 e + \hat{A} - n - n \log_2 n - n \log_2 r \right) + n \log_2 n - n \log_2 e + \hat{A}'$$

$$= mn \log_2 n + \hat{A}'',$$  \hspace{1cm} (9)

using Sterling’s approximation [16], where $|\hat{A}|, |\hat{A}'| \in O(\log_2 n)$ and $|\hat{A}''| \in O(n)$. Also note that the $(m - 1)$ factor arises, as opposed to $m$, as in the final layer all the qubits are re-labelled with the null label, and so no entropy is added.

It follows that it is necessary to consider the extent to which the ordering process, $\chi$, can be compressed into time-steps. That is, to lower-bound the number of time-steps required to achieve the necessary reduction in entropy $\Delta S(\chi)$. To do this, consider the maximum entropy that can be removed from the system in the $j^{th}$ time step, denoted $-\Delta S_S^{(j)}$. This can be upper-bounded by considering the scenario in which each of the $nr/2$ edges is independently set into a specified state (i.e., intentionally swapped or not) thus achieving a reduction in entropy of $(nr/2)$ bits (each edge decision adds one bit of order, as there are two states). Thus,

$$-\Delta S_S^{(j)} \leq \frac{nr}{2},$$  \hspace{1cm} (10)
where any effect of double counting the entropy decrease of successive steps ‘overlapping’ to an extent in entropy reduction is absorbed into the inequality. It follows that the minimum number of time-steps required for the ordering process, \( \chi \), (for the ensemble average) can be lower-bounded:

\[
-\Delta S(\chi) = \sum_{j=1}^{N_S} \Delta S^{(j)}_S \leq N_S \frac{nr}{2},
\]

Equating (9) and (11), for \( m \geq 1 \):

\[
N_S \geq \frac{2mn \log_2 n + \hat{A}''}{nr} \Rightarrow \frac{N_S}{m} \geq \frac{2\log_2 n + (\hat{A}''/nm)}{r} \in \Omega(\log n).
\]

At least one of the quantum circuits in the ensemble must take the full \( N_S \) steps, and thus have a depth overhead \( \in \Omega(\log n) \), thus proving Theorem 3.3.

**Corollary 3.4.** \( \forall r(n) \in O(\log^a n) \) \( \forall (G \in G_{n,r,\chi}) \exists \zeta, D(G, \zeta, \chi) \in \Omega(\log^{1-a} n) \)

**Proof.** The proof follows directly by substituting in \( r \) as a function of \( n \) into (12) and following the remaining analysis.

In general, it may be of interest to understand how the depth overhead varies if the degree is not finite, but is itself allowed to grow with \( n \). Corollary 3.4 corresponds to one such instance, and holds for all \( a \), however the bound is only useful for \( a < 1 \).

**4 Discussion and open problems**

Theorem 3.1 proves that a depth overhead in \( O(\log n) \) is achievable, and Theorem 3.3 provides the converse, that quantum circuits can be constructed such that the depth overhead must be at least \( \log n \). Thus together Theorem 3.1 and 3.3 constitute a tight bound, in terms of the order of the asymptotic growth of the depth overhead. The existence of the converse confirms that the procedure and graph proposed by Brierley [14] (with the adjusted procedure detailed in Appendix A) is asymptotically efficient. Intriguingly, however, Conjecture 3.2 indicates that such performance may be achievable on almost all \( r \)-regular graphs.

One feature of the analysis in this paper is that the depth overhead is defined as being for any initial qubit arrangement (Definition 2.2). However, in practical systems, it may well be the case that the quantum algorithm compiler is free to select a particular initial qubit arrangement (i.e., one which is advantageous) for negligible cost, thus making the depth overhead defined herein a slight overestimate. Informally, it can be argued that, by considering sufficiently deep quantum circuits, the effect of the initial qubit arrangement can be made negligible. Formally, it is a little harder to account for this scenario – it is not merely a case of setting \( S_0 \geq 0 \) (i.e., to allow for initial order) as the subsequent interactions (re-labelling process) could be correlated with the initial qubit arrangement (as is the possible effect of an initial ordering), and thus the analysis would be invalid. Crucially, however, the analysis herein is valid with the random initial qubit arrangement scenario used, even for an ordering process which may make its decisions taking into account interactions far into the future (and thus not corresponding to the current labelling). This is because ordering a pair of qubits (as in a decision to swap or not) can only ever input 1 bit of order at most, regardless of the labelling convention; furthermore, considering the qubits to be relabelled after an interaction leads to exactly the same input of entropy in total as if a multiple-label model were used.

Also on a practical note, the analysis in this paper is significant for informing the design of near-term quantum computers, as well as the process of ordering qubits to achieve gates therein. In particular, the qubit interaction graphs used to achieve depth overhead \( O(\log n) \) in both Theorem 3.1 and Conjecture 3.2
cannot be embedded in $k$-dimensional space. The best known achievable depth overhead for a qubit interaction graph that is embeddable in $k$-dimensional space is in $O(\sqrt[3]{n})$, and thus there may be good reason to favour quantum computer designs which do not constrain the qubit interaction graph in this manner, such as NQIT’s proposed quantum computer, in which the graph edges are given physical reality as photonic links [7].

Another interesting question prompted by the analysis in this paper is whether removing the requirement of running only full-width algorithms can lead to a situation where depth overhead can potentially be exchanged for width overhead. In particular, allowing width overhead introduces the possibility of using entanglement as a resource, and thus teleporting states around the quantum computer. Rosenbaum [18] addresses such a scenario, however this work does not explicitly include the quantum circuitry required to continually generate and distribute the Bell pairs around the network, rendering the analysis somewhat incompatible with that which would constitute a natural extension of this paper.

A Achieving $D \in O(\log n)$

![Diagram](image)

Figure 3: Demonstrating swap of Q1 and Q3 in three steps, without disturbing Q2 – i.e., Q3 acts as the ancilla, into which Q1 is loaded.

Let there be a cyclic butterfly network with $n/2$ vertices.

1. Use the first alteration in [14, Section IV] to make a graph with degree 3.

2. For each vertex, connect another vertex (i.e., yielding a graph with $n$ vertices, half of which are of degree 4, half of which are of degree 1). These newly added vertices, along with their originally connected existing vertex are referred to as ‘pairs’ below.

3. Add arbitrary extra edges to make the graph $r$-regular (where $r \geq 4$) as specified. These edges are unused.

4. For all pairs of vertices, if the paired qubits are to interact in the current layer then do the interaction, and remove from the layer. **1 time step maximum**

5. Choose a random vertex from the set of vertices in the original graph, label the qubit at this vertex ‘A’. Label the qubit at the paired vertex (i.e., as added in step 2) ‘B’. If ‘A’ is due to interact in the current layer, label its interacting qubit ‘A’, likewise if ‘B’ is due to interact in the current layer, label its interacting qubit ‘B’. Label the paired qubit of the newly labelled ‘A’ as ‘B’, and label the paired qubit of the newly labelled ‘B’ as ‘A’. Continue until the cycle terminates, either because both ends of the string have reached qubits which do not interact with any other qubit in the current layer, or because a loop is formed. Chose another random starting point and repeat – continue until all qubits are labelled.

6. The network is now labelled such that each pair has exactly one A and one B. Furthermore, all As are to interact with other As if they have a interaction in the current layer (and likewise for Bs).
7. Order the network so that all of the As are in vertices in the original graph, and Bs are in the additional paired vertices. **1 time step maximum**

8. Let second of each pair of logical qubits (i.e., the one with only one connection to the first logical qubit) act as the ancilla for the subsequent sorting. Note that in the method detailed by Brierley [14] ‘moving a qubit into a neighbours ancilla’ is an identical operation to swapping an ancilla. Thus, this can be achieved by the process shown in Fig. 3. This takes three time steps, and at worst has to occur three times to achieve the required swap with the ancilla for all qubits (i.e., label qubits alternately as odds and evens in each string of ancilla swaps, do all of the odds and then all of the evens, and at worst one further step will be required in the case of qubits connected in odd length cycles). **A maximum of 27 log n time steps**

9. The vertices of the original graph are arranged in loops of four, which means that the original graph can be divided into \( \frac{n}{4} \) connected pairs of vertices: Condition 2.5 is satisfied, and the sort in the previous step can be such that all of the interactions of qubits labelled A can be conducted.

10. Notice that the ordering is such that a qubit labelled A is at each of the original nodes, this means that a qubit labelled B is at each of the extra vertices introduced in step 2. Thus a swap is conducted at each pair such that the Bs now occupy the original vertices, and steps 8 – 9 are repeated with the qubits labelled B, thus completing the layer. **1 time step**

The procedure requires a maximum of \( 3 + 2 \times 27 \log n \) time steps, i.e., noticing that step 8 is conducted twice, and that the original algorithm proposed by Brierley requires \( 3 \log n \) time steps [14]. Therefore the number of time steps remains \( O(\log n) \).

B Lemmas

**Lemma B.1.** \( r' \leq (r + 1)^n \)

*Proof.* This inequality can be seen by considering the upper-bounding case where each of the \( n \) nodes is independently selected to either permute with one of \( r \) neighbours, or to remain inactive for the time step, i.e., \( r + 1 \) possibilities. Therefore each state can transfer to at most \( (r + 1)^n \) other states, which is the degree of \( G' \in \mathcal{G}_{n,r} \) by definition.

**Lemma B.2.** \( r' \geq 2^{n/4} \)

*Proof.* Consider the set of edges of \( G \), which has size \( nr/2 \), a subset of which can be constructed of size \( n/4 \) consisting of edges that can be independently permuted (or not) by the following method: Select an edge, remove all edges connected to either connecting vertex from future consideration, repeat until all edges are either selected or removed from future consideration. At step two, at most \( 2r - 2 \) edges are removed from future consideration (i.e., if non of the connecting edges have already been neglected), therefore each step occupies (either by selecting or neglecting) at most \( 2r - 1 < 2r \) edges. Therefore a set of size greater than \( (nr/2)/2r = n/4 \) has been constructed, containing combinations of edges which can be independently permuted or left, thus there are at least \( 2^{n/4} \) possible state transitions available in each state of \( G' \), and Lemma B.3 follows from the definition of \( G' \).

References

[1] A. C.-C. Yao, “Quantum circuit complexity,” in Proceedings of 1993 IEEE 34th Annual Foundations of Computer Science, Nov 1993, pp. 352–361.

[2] “Quantum computational networks,” Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences, vol. 425, no. 1868, pp. 73–90, 1989. [Online]. Available: http://rspa.royalsocietypublishing.org/content/425/1868/73
[3] A. Barenco, C. H. Bennett, R. Cleve, D. P. DiVincenzo, N. Margolus, P. Shor, T. Sleator, J. A. Smolin, and H. Weinfurter, “Elementary gates for quantum computation,” *Phys. Rev. A*, vol. 52, pp. 3457–3467, Nov 1995. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevA.52.3457

[4] “A preview of Bristlecone, Googles new quantum processor.” [Online]. Available: https://research.googleblog.com/2018/03/a-preview-of-bristlecone-googles-new.html

[5] M. Steffen, D. P. DiVincenzo, J. M. Chow, T. N. Theis, and M. B. Ketchen, “Quantum computing: An IBM perspective,” *IBM Journal of Research and Development*, vol. 55, no. 5, pp. 13:1–13:11, Sept 2011.

[6] “The quantum processing unit (rigetti).” [Online]. Available: http://pyquil.readthedocs.io/en/latest/qpu.html

[7] N. H. Nickerson, J. F. Fitzsimons, and S. C. Benjamin, “Freely scalable quantum technologies using cells of 5-to-50 qubits with very lossy and noisy photonic links,” *Phys. Rev. X*, vol. 4, p. 041041, Dec 2014. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevX.4.041041

[8] R. Beals, S. Brierley, O. Gray, A. W. Harrow, S. Kutin, N. Linden, D. Shepherd, and M. Stathier, “Efficient distributed quantum computing,” *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, vol. 469, no. 2153, 2013. [Online]. Available: http://rspa.royalsocietypublishing.org/content/469/2153/20120686

[9] A. G. Fowler, S. J. Devitt, and L. C. L. Hollenberg, “Implementation of shor’s algorithm on a linear nearest neighbour qubit array,” *eprint arXiv:quant-ph/0402196*, Feb. 2004.

[10] D. Maslov, “Linear depth stabilizer and quantum fourier transformation circuits with no auxiliary qubits in finite-neighbor quantum architectures,” vol. 76, no. 5, p. 052310, Nov. 2007.

[11] S. A. Kutin, “Shor’s algorithm on a nearest-neighbor machine,” *eprint arXiv:quant-ph/0609001*, Aug. 2006.

[12] P. Pham and K. M. Svore, “A 2D Nearest-Neighbor Quantum Architecture for Factoring in Polylogarithmic Depth,” *ArXiv e-prints*, Jul. 2012.

[13] D. Cheung, D. Maslov, and S. Severini, “Translation techniques between quantum circuit architectures,” 2007.

[14] S. Brierley, “Efficient implementation of Quantum circuits with limited qubit interactions,” *ArXiv e-prints*, Jul. 2015.

[15] B. Bollobas, *Random Graphs*, 2nd ed., ser. Cambridge Studies in Advanced Mathematics. Cambridge University Press, 2001.

[16] J. Dutka, “The early history of the factorial function,” *Archive for History of Exact Sciences*, vol. 43, no. 3, pp. 225–249, Sep 1991. [Online]. Available: https://doi.org/10.1007/BF00389433

[17] R. W. Robinson and N. C. Wormald, “Almost all regular graphs are hamiltonian,” *Random Struct. Algorithms*, vol. 5, no. 2, pp. 363–374, Apr. 1994. [Online]. Available: http://dx.doi.org/10.1002/rsa.3240050209

[18] D. Rosenbaum, “Optimal Quantum Circuits for Nearest-Neighbor Architectures,” *ArXiv e-prints*, Apr. 2012.