Synthesis of Quantum Logic Circuits

Vivek V. Shende[1] vshende@umich.edu

Stephen S. Bullock[2] stephen.bullock@nist.gov

Igor L. Markov[1] imarkov@eecs.umich.edu

[1]Dept. of Electrical Engineering and Computer Science,
The University of Michigan, Ann Arbor, MI 48109-2212, USA

[2]Mathematical and Computational Sciences Division,
Natl. Inst. of Standards and Technology, Gaithersburg, MD 20899-8910, USA

Abstract

The pressure of fundamental limits on classical computation and the promise of exponential speedups from quantum effects have recently brought quantum circuits [10] to the attention of the Electronic Design Automation community [18, 28, 7, 27, 17]. We discuss efficient quantum logic circuits which perform two tasks: (i) implementing generic quantum computations and (ii) initializing quantum registers. In contrast to conventional computing, the latter task is nontrivial because the state-space of an n-qubit register is not finite and contains exponential superpositions of classical bit strings. Our proposed circuits are asymptotically optimal for respective tasks and improve earlier published results by at least a factor of two.

The circuits for generic quantum computation constructed by our algorithms are the most efficient known today in terms of the number of difficult gates (quantum controlled-NOTs). They are based on an analogue of the Shannon decomposition of Boolean functions and a new circuit block, quantum multiplexor, that generalizes several known constructions. A theoretical lower bound implies that our circuits cannot be improved by more than a factor of two. We additionally show how to accommodate the severe architectural limitation of using only nearest-neighbor gates that is representative of current implementation technologies. This increases the number of gates by almost an order of magnitude, but preserves the asymptotic optimality of gate counts.

1 Introduction

As the ever-shrinking transistor approaches atomic proportions, Moore's law must confront the small-scale granularity of the world: we cannot build wires thinner than atoms. Worse still, at atomic dimensions we must contend with the laws of quantum mechanics. For example, suppose one bit is encoded as the presence or the absence of an electron in a small region.[1] Since we know very precisely where the electron is located, the Heisenberg uncertainty principle dictates that we cannot know its momentum with high accuracy. Without a reasonable upper bound on the electron's momentum, there is no alternative but to use a large potential to keep it in place, and expend significant energy during logic switching. A quantitative analysis of these phenomena leads experts from NCSU, SRC and Intel [36] to derive fundamental limitations on the scalability of any computing device which moves electrons.

Yet these same quantum effects also facilitate a radically different form of computation [13]. Theoretically, *quantum* computers could outperform their classical counterparts when solving certain discrete

[1] Most current computing technologies use electron charges to store information; exceptions include spintronics-based techniques, e.g., magnetic RAM.

problems [16]. For example, a successful large-scale implementation of Shor's integer factorization [29] would compromise the RSA cryptosystem used in electronic commerce. On the other hand, quantum effects may also be exploited for public-key cryptography [4]. Indeed, such cryptography systems, based on single-photon communication, are commercially available from MagiQ Technologies in the U.S. and IdQuantique in Europe.

Physically, a quantum bit might be stored in one of a variety of quantum-mechanical systems. A broad survey of these implementation technologies, with feasibility estimates and forecasts, is available in the form of the ARDA quantum computing roadmap [1]. Sample carriers of quantum information include top-electrons in hyperfine energy levels of either trapped atoms or trapped ions, tunneling currents in cold superconductors, nuclear spin polarizations in nuclear magnetic resonance, and polarization states of single photons. A collection of n such systems would comprise an n-qubit register, and quantum logic gates (controlled quantum processes) would then be applied to the register to perform a computation. In practice, such gates might result from rotating the electron between hyperfine levels by shining a laser beam on the trapped atom/ion, tuning the tunneling potential by changing voltages and/or current in a super-conducting circuit, or perhaps passing multiple photons through very efficient nonlinear optical media.

The logical properties of qubits also differ significantly from those of classical bits. Bits and their manipulation can be described using two constants (0 and 1) and the tools of boolean algebra. Qubits, on the other hand, must be discussed in terms of vectors, matrices, and other linear algebraic constructions. We will fully specify the formalism in Section 2, but give a rough idea of the similarities and differences between classical and quantum information below.

1. A readout (observation, measurement) of a quantum register results in a classical bit-string.

2. However, identically prepared quantum states may yield different classical bit-strings upon observation. Quantum physics only predicts the probability of each possible readout, and the readout probabilities of different bits in the register need not be independent.

3. After readout, the state "collapses" onto the classical bit string observed. All other quantum data is lost.

These differences notwithstanding, *quantum logic circuits*, from a high level perspective, exhibit many similarities with their classical counterparts. They consist of quantum gates, connected (though without fanout or feedback) by quantum wires which carry quantum bits. Moreover, logic synthesis for quantum circuits is as important as for the classical case. In current implementation technologies, gates that act on three or more qubits are prohibitively difficult to implement directly. Thus, implementing a quantum computation as a sequence of two-qubit gates is of crucial importance. Two-qubit gates may in turn be decomposed into circuits containing one-qubit gates and a standard two-qubit gate, usually the quantum controlled-not (CNOT). These decompositions are done by hand for published quantum algorithms (e.g., Shor's factorization algorithm [29] or Grover's quantum search [16]), but have long been known to be possible for arbitrary quantum functions [12, 3]. While CNOTs are used in an overwhelming majority of theoretical and practical work in quantum circuits, their implementations are orders of magnitude more error-prone than implementations of single-qubit gates and have longer durations. Therefore, the cost of a quantum circuit can be realistically calculated by counting CNOT gates. Moreover, it has been shown previously that if CNOT is the only two-qubit gate type used, the number of such gates in a sufficiently large irredundant circuit is lower-bounded by approximately 20% [27].

The first quantum logic synthesis algorithm to so decompose an arbitrary n-qubit gate would return a circuit containing $O(n^3 4^n)$ CNOT gates [3]. The work in [9] interprets this algorithm as the QR decomposition, well-known in matrix algebra. Improvements on this method have used clever circuit transformations and/or Gray codes [20, 2, 31] to lower this gate count. More recently, different techniques [21] have led to circuits with CNOT-counts of $4^n - 2^{n+1}$. The exponential gate count is not unexpected: just as the exponential number of n-bit Boolean functions ensures that the circuits computing them are generically large, so too in the quantum case. Indeed, it has been shown that n-qubit operators generically require $\lceil \frac{1}{4}(4^n - 3n - 1) \rceil$ CNOTs [27]. Similar exponential lower bounds existed earlier in other gate libraries [20].

Existing algorithms for n-qubit circuit synthesis remain a factor of four away from lower bounds and fare poorly for small n. These algorithms require at least 8 CNOT gates for $n = 2$, while three CNOT gates are necessary and sufficient in the worst case [27, 34, 33]. Further, a simple procedure exists to produce two-qubit circuits with minimal possible number of CNOT gates [25]. In contrast, in three qubits the lower bound is 14 while the generic n-qubit decomposition of [21] achieves 48 CNOTs and a specialty 3-qubit circuit of [32] achieves 40.

In this work, we focus on identifying useful quantum circuit blocks. To this end, we analyze quantum conditionals and define *quantum multiplexors* that generalize CNOT, Toffoli and Fredkin gates. Such quantum multiplexors implement if-then-else conditionals when the controlling predicate evaluates to a coherent superposition of $|0\rangle$ and $|1\rangle$. We find that quantum multiplexors prove amenable to recursive decomposition and vastly simplify the discussion of many results in quantum logic synthesis (cf. [8, 31, 21]). Ultimately, our analysis leads to a quantum analogue of the Shannon decomposition, which we apply to the problem of quantum logic synthesis.

We contribute the following key results.

- An arbitrary n-qubit quantum state can be prepared by a circuit containing no more than $2^{n+1} - 2n$ CNOT gates. This lies a factor of four away from the theoretical lower bound.

- An arbitrary n-qubit operator can be implemented in a circuit containing no more than $(23/48) \times 4^n - (3/2) \times 2^n + 4/3$ CNOT gates. This improves upon the best previously published work by a factor of two and lies less than a factor of two away from the theoretical lower bound.

- In the special case of three qubits, our technique yields a circuit with 20 CNOT gates, whereas the best previously known result was 40.

- The architectural limitation of permitting only nearest-neighbor interactions, common to physical implementations, does not change the asymptotic behavior of our techniques.

In addition to these technical advances, we develop a theory of quantum multiplexors that parallels well-known concepts in digital logic, such as Shannon decomposition of Boolean functions. This new theory produces short and intuitive proofs of many results for n-qubit circuits known today.

The remainder of the paper is organized as follows. In §2, we define quantum bits, quantum logic, and quantum circuits, and we introduce the necessary mathematical formalism for manipulating them. In §3, we introduce a novel circuit block, the *quantum multiplexor*, which immediately allows radical notational simplifications of the statements and proofs of previously known results. In §4, we give a novel, asymptotically-optimal algorithm for register initialization and indicate its applications to more general problems in quantum logic synthesis. In §5, we use the *Cosine-Sine decomposition*, along with a novel decomposition of single-select-bit quantum multiplexors, to derive a functional decomposition for quantum logic that can be applied recursively. We obtain quantum circuits to simulate any unitary operator (quantum evolution) U and present competitive gate counts. In §6, we show that our techniques adapt well to severe implementation constraints representative of many quantum-circuit technologies. Our results are summarized in §7, which concludes the paper. Additionally, two highly-technical aspects of our work required to achieve the best gate counts are described in the Appendix.

2 Background and Notation

The notion of a *qubit* formalizes the logical properties of an ideal quantum-mechanical system with two basis states. The two states are labeled $|0\rangle$ and $|1\rangle$. They can be distinguished by quantum measurement of the qubit, which yields a single classical bit of information, specifying which state the qubit was observed in. However, the state of an isolated (in particular, unobserved) qubit must be modeled by vector in a two-dimensional complex[2] vector space $\mathcal{H}_1$ which is spanned by the basis states.

$$\mathcal{H}_1 = \mathrm{span}_{\mathbb{C}}\{|0\rangle, |1\rangle\} \tag{1}$$

We identify $|0\rangle$ and $|1\rangle$ with the following column vectors.

$$|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \qquad |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \tag{2}$$

Thus, an arbitrary state $|\phi\rangle \in \mathcal{H}_1$ can be written in either of the two equivalent forms given below.

$$|\phi\rangle = \alpha_0 |0\rangle + \alpha_1 |1\rangle = \begin{pmatrix} \alpha_0 \\ \alpha_1 \end{pmatrix} \tag{3}$$

The entries of the state vector determine the readout probabilities: if we measure a qubit whose state is described by $|\phi\rangle$, we should expect to see $|0\rangle$ with probability $|\alpha_0|^2$ and $|1\rangle$ with probability $|\alpha_1|^2$. Since these are the only two possibilities, α_0 and α_1 are required to satisfy $|\alpha_0|^2 + |\alpha_1|^2 = 1$.

2.1 Qubit Registers

By a *register of qubits*, we shall simply mean a logical qubit array with a fixed number of qubits in a fixed order. A readout of a qubit register amounts to readouts of each component qubit; thus a readout of an n-qubit register might take the form $|b_0\rangle |b_1\rangle \dots |b_{n-1}\rangle$ for each $b_j \in \{0,1\}$. We shall abbreviate this to $|b_0 b_1 \dots b_{n-1}\rangle$, and call it a *bitstring state*. Just as for a single qubit, the state of an isolated qubit register is modeled by a vector in the complex vector space spanned by the bitstring states.

$$\mathcal{H}_n = \mathrm{span}_{\mathbb{C}}\{|b\rangle; b \text{ a bitstring of length n}\} \tag{4}$$

Writing $\mathbb{B}^n$ for the set of length n bitstrings, an arbitrary vector $|\psi\rangle \in \mathcal{H}_n$ may be expressed as $\sum_{b \in \mathbb{B}^n} \alpha_b |b\rangle$, or as the column vector whose b-th entry is α_b. As for a single qubit, $|\alpha_b|^2$ represents the probability that a readout of $|\psi\rangle$ yields the bitstring $|b\rangle$; thus the α_b are subject to the relation $\sum_b |\alpha_b|^2 = 1$.

Suppose we concatenate a ℓ-qubit register L and an m-qubit register M to form an $\ell + m = n$-qubit register N. Assuming L and M have not previously interacted (and remain independent), we may describe them by state vectors $|\psi_L\rangle \in \mathcal{H}_\ell$ and $|\psi_M\rangle \in \mathcal{H}_m$.

$$|\psi_L\rangle = \sum_{b \in \mathbb{B}^\ell} \beta_b |b\rangle \qquad |\psi_M\rangle = \sum_{b' \in \mathbb{B}^m} \gamma_{b'} |b'\rangle \tag{5}$$

To describe the state of N, we must somehow obtain from $|\psi_L\rangle$ and $|\psi_M\rangle$ a state vector $|\psi_N\rangle \in \mathcal{H}_n$. Quantum mechanics demands that we use a natural generalization of bitstring concatenation called the *tensor product*. To compute the tensor product of two states, we write $|\psi_N\rangle = |\psi_L\rangle |\psi_M\rangle$, and expand it using the distributive law.

[2] Complex rather than real coefficients are required in most applications. For example, in certain optical implementations [22, §7.4.2] real and imaginary parts encode both the presence and phase of a photon.

$$|\psi_L\rangle\,|\psi_M\rangle = \sum_{b\in\mathbb{B}^\ell, b'\in\mathbb{B}^m} \beta_b\gamma_{b'}\,|b\rangle\,|b'\rangle \tag{6}$$

Let $\cdot$ denote concatenation; then $|b\;\rangle\,|b'\rangle$ and $|b\cdot b'\rangle$ represent the same bitstring state. As $b\cdot b'\in\mathbb{B}^n$, we have $|\psi_L\rangle\,|\psi_M\rangle\in\mathcal{H}_n$, as desired.

Perhaps counter-intuitively, the quantum-mechanical state of N cannot in general be specified only in terms of the states of L and M. Indeed, $\mathcal{H}_k$ is a 2^k dimensional vector space, and for $n\gg 2$ we observe $2^n\gg 2^m+2^\ell$. For example, three independent qubits can be described by three two-dimensional vectors, while a generic state-vector of a three-qubit system is eight-dimensional. Much interest in quantum computing is driven by this exponential scaling of the state space, and the loss of independence between different subsystems is called quantum entanglement.

2.2 Quantum Logic Gates

By a *quantum logic gate*, we shall mean a closed-system evolution (transformation) of the n-qubit state space $\mathcal{H}_n$. In particular, this means that no information is gained or lost during this evolution, thus a quantum gate has the same number of input qubits as output qubits. If $|\psi\rangle$ is a state vector in $\mathcal{H}_n$, the operation of an n-qubit quantum logic gate can be represented by $|\psi\rangle\mapsto U\,|\psi\rangle$ for some *unitary* $2^n\times 2^n$ matrix U. To define unitarity, we first introduce the *adjoint* of a matrix.

Notation. Let M be an $n\times m$ matrix. By $M^\dagger$, we will mean the $m\times n$ matrix whose (i,j)-th entry is the complex conjugate of the (j,i)-th entry of M. In other words, $M^\dagger$ is the conjugate transpose of M.

A square matrix M is *unitary* iff $M^\dagger M = I_\ell$ for I_ℓ an $\ell\times\ell$ identity matrix. This is the matrix equation for a symmetry: M is unitary iff the vector images of M have the same complex inner products as the original vectors. Thus, (a) identity matrices are unitary, (b) a product of unitary matrices is unitary, and (c) the inverse of a unitary matrix, given by the adjoint, is also unitary. These may be restated in terms of quantum logic. The quantum logic operation of "doing nothing" is modeled by the identity matrix, serial composition of gates is modeled by the matrix products, and every quantum gate is reversible.

We shall often define quantum gates by simply specifying their matrices. For example, the following matrix specifies a quantum analogue of the classical inverter: it maps $|0\rangle\mapsto|1\rangle$ and $|1\rangle\mapsto|0\rangle$.

- The inverter $\sigma_x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$

Many quantum gates are specified by time-dependent matrices that represent the evolution of a quantum system (e.g., an RF pulse affecting a nucleus) that has been "turned on" for time θ. For example, the following families of gates are the one-qubit gates most commonly available in physical implementations of quantum circuits.

- The x-axis rotation $R_x(\theta) = \begin{pmatrix} \cos\theta/2 & i\sin\theta/2 \\ i\sin\theta/2 & \cos\theta/2 \end{pmatrix}$
- The y-axis rotation $R_y(\theta) = \begin{pmatrix} \cos\theta/2 & \sin\theta/2 \\ -\sin\theta/2 & \cos\theta/2 \end{pmatrix}$
- The z-axis rotation $R_z(\theta) = \begin{pmatrix} \mathrm{e}^{-i\theta/2} & 0 \\ 0 & \mathrm{e}^{i\theta/2} \end{pmatrix}$

An arbitrary one-qubit computation can be implemented as a sequence of at most three R_z and R_y gates. This is due to the *ZYZ decomposition*:[3] given any 2×2 unitary matrix U, there exist angles $\Phi, \alpha, \beta, \gamma$ satisfying the following equation.

$$U = \mathrm{e}^{i\Phi} R_z(\alpha) R_y(\beta) R_z(\gamma) \tag{7}$$

The nomenclature R_x, R_y, R_z is motivated by a picture of one-qubit states as points on the surface of a sphere of unit radius in $\mathbb{R}^3$. This picture is called the *Bloch sphere* [22], and may be obtained by expanding an arbitrary two-dimensional complex vector as below.

$$|\psi\rangle \;=\; \alpha_0 |0\rangle + \alpha_1 |1\rangle \;=\; r\mathrm{e}^{it/2} \left[\mathrm{e}^{-i\varphi/2} \cos\frac{\theta}{2} |0\rangle + \mathrm{e}^{i\varphi/2} \sin\frac{\theta}{2} |1\rangle \right] \tag{8}$$

The constant factor $r\mathrm{e}^{it/2}$ is physically undetectable. Ignoring it, we are left with two angular parameters θ and φ, which we interpret as spherical coordinates $(1, \theta, \varphi)$. In this picture, $|0\rangle$ and $|1\rangle$ correspond to the north and south poles, $(1, 0, 0)$ and $(1, \pi, 0)$, respectively. The $R_x(\theta)$ gate (resp. $R_y(\theta)$, $R_z(\theta)$) corresponds to a counterclockwise rotation by θ around the x (resp. y, z) axis. Finally, just as the point given by the spherical coordinates $(1, \theta, \varphi)$ can be moved to the north pole by first rotating $-\varphi$ degrees around the z-axis, then $-\theta$ degrees around the y axis, so too the following matrix equations hold.

$$\begin{gathered} R_y(-\theta) R_z(-\varphi) |\psi\rangle = r\mathrm{e}^{it/2} |0\rangle \\ R_y(\theta - \pi) R_z(\pi - \varphi) |\psi\rangle = r\mathrm{e}^{i(t-\pi)/2} |1\rangle \end{gathered} \tag{9}$$

2.3 Quantum Circuits

A combinational *quantum logic circuit* consists of quantum gates, interconnected by quantum wires – carrying qubits – without fanout or feedback. As each quantum gate has the same number of inputs and outputs, any cut through the circuit crosses the same number of wires. Fixing an ordering on these, a quantum circuit can be understood as representing the sequence of quantum logic operations on a quantum register. An example is depicted in Figure 1, and many more will appear throughout the paper.

Figure 1 contains 12 one- and two-qubit gates applied to a three-qubit register. Observe that the state of a three-qubit register is described by a vector in $\mathcal{H}_3$ (an 8-element column), whereas one- and two-qubit gates are described by unitary operations on $\mathcal{H}_2$ and $\mathcal{H}_1$ (given by 4×4 and 2×2 matrices, respectively). In order to reconcile the dimensions of various state-vectors and matrices, we introduce the tensor product operation.

Consider an $\ell + m = n$-qubit register, on which an ℓ-qubit gate V acts on the top ℓ qubits, with an m-qubit gate W acting on the remainder. We expand the state $|\psi\rangle \in \mathcal{H}_n$ of the n-qubit register, as follows.

$$|\psi\rangle = \sum_{b \in \mathbb{B}^n} \alpha_b |b\rangle = \sum_{b \in \mathbb{B}^\ell, b' \in \mathbb{B}^m} \alpha_{b \cdot b'} \left| b \right\rangle \left| b' \right\rangle \tag{10}$$

Then, denoting by $V \otimes W$ the operation performed on the register as a whole,

$$V \otimes W |\psi\rangle = \sum_{b \in \mathbb{B}^\ell, b' \in \mathbb{B}^m} \alpha_{b \cdot b'} \left(V \left| b \right\rangle \right) \left(W \left| b' \right\rangle \right) \tag{11}$$

Here, $V\left| b \right\rangle \in \mathcal{H}_\ell$ and $W\left| b' \right\rangle \in \mathcal{H}_m$ are to be concatenated, or tensored, as per Equation 6. It can be deduced from Equation 11 that the $2^n \times 2^n$ matrix of $V \otimes W$ is given by

$$(V \otimes W)_{r \cdot r', c \cdot c'} = V_{r,c} W_{r',c'} \qquad \text{for} \qquad r, c \in \mathbb{B}^\ell, \quad r', c' \in \mathbb{B}^m \tag{12}$$

[3] This decomposition is well known and finds many proofs in the literature, e.g., [3]. We shall derive another as a corollary in Section 4.

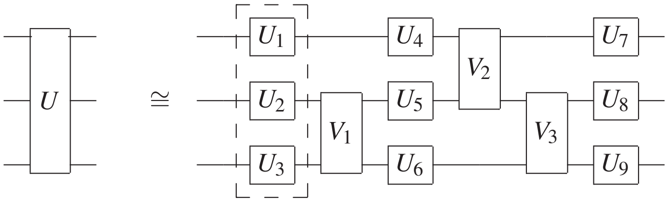
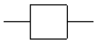


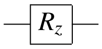
Figure 1: A typical quantum logic circuit. Information flows from left to right, and the higher wires represent higher order qubits. The quantum operation performed by this circuit is $(U_7 \otimes U_8 \otimes U_9)(I_2 \otimes V_3)(V_2 \otimes I_2)(U_4 \otimes U_5 \otimes U_6)(I_2 \otimes V_1)(U_1 \otimes U_2 \otimes U_3)$, and the last factor is outlined above. Note that when the matrix $A \cdot B$ is applied to vector $\vec{v}$, this is equivalent to applying the matrix B first, followed by the matrix A. Therefore, the formulas describing quantum circuits must be read right to left.

2.4 Circuit Equivalences

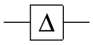
Rather than begin the statement of every theorem with "let $U_1, U_2, \ldots$ be unitary operators...," we are going to use diagrams of quantum logic circuits and *circuit equivalences*. An equivalence of circuits in which all gates are fully specified can be checked by multiplying matrices. However, in addition to fully specified gates, our circuit diagrams will contain the following *generic*, or *under-specified* gates:

Notation. An equivalence of circuits containing generic gates will mean that for any specification (i.e., parameter values) of the gates on one side, there exists a specification of the gates on the other such that the circuits compute the same operator. Generic gates used in this paper are limited to the following:

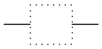

A generic unitary gate.



An R_z gate without a specified angular parameter; conventions for R_x, R_y are similar.

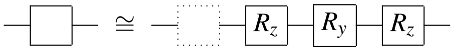


A generic diagonal gate.


A generic scalar multiplication (*uncontrolled* gate implemented by "doing nothing.")

We may restate Equation 7 as an equivalence of generic circuits.

Theorem 1 : The ZYZ decomposition [3].

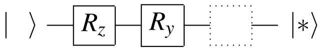


Similarly, we also allow underspecified states.

Notation. We shall interpret a circuit with underspecified states and generic gates as an assertion that for any specification of the underspecified input and output states, some specification of the generic gates circuit that performs as advertised. We shall denote a completely unspecified state as $|\ \ \rangle$, and an unspecified bitstring state as $|*\rangle$.

For example, we may restate Equation 9 in this manner.

Theorem 2 : Preparation of one-qubit states.



We shall use a backslash to denote that a given wire may carry an arbitrary number of qubits (quantum bus). In the sequel, we seek backslashed analogues of Theorems 1 and 2.

3 Quantum Conditionals and the Quantum Multiplexor

Classical conditionals can be described by the `if-then-else` construction: `if` the predicate is true, perform the action specified in the `then` clause, if it is false, perform the action specified in the `else` clause. At the gate level, such an operation might be performed by first processing the two clauses in parallel, then multiplexing the output. To form the quantum analogue, we replace the predicate by a qubit, replace true and false by $|1\rangle$ and $|0\rangle$, and demand that the actions corresponding to clauses be unitary. The resulting "quantum conditional" operator U will then be unitary. In particular, when selecting based on a coherent superposition $\alpha_0 |0\rangle + \alpha_1 |1\rangle$, it will generate a linear combination of the `then` and `else` outcomes. Below, we shall use the term *quantum multiplexor* to refer to the circuit block implementing a quantum conditional.

Notation. We shall say that a gate U is a quantum multiplexor with *select* qubits S if it preserves any bitstring state $|b\rangle$ carried by S. In this case, we denote U in quantum logic circuit diagrams by "□" on each select qubit, connected by a vertical line to a gate on the remaining *data* (read-write) qubits.

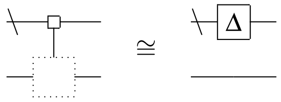
In the event that a multiplexor has a single select bit, and the select bit is most significant, the matrix of the quantum multiplexor is block diagonal.

$$U = \begin{pmatrix} U_0 & \\ & U_1 \end{pmatrix} \tag{13}$$

The multiplexor will apply U_0 or U_1 to the data qubits according as the select qubit carries $|0\rangle$ or $|1\rangle$. To express such a block diagonal decomposition, we shall use the notation $U = U_0 \oplus U_1$ that is standard in linear algebra. More generally, let V be a multiplexor with s select qubits and a d-qubit wide data bus. If the select bits are most significant, the matrix of V will be block diagonal, with 2^s blocks of size $2^d \times 2^d$. The j-th block V_j is the operator applied to the data bits when the select bits carry $|j\rangle$.

In general, a gate depicted as a quantum multiplexor need not read or modify as many qubits as indicated on a diagram. For example, a multiplexor which performs the same operation on the data bits regardless of what the select bits carry can be implemented as an operation on the data bits alone. We give a less trivial example below: a multiplexor which applies a different scalar multiplication for each value of the select bits can be implemented as a diagonal operator applied to the select bits.

Theorem 3 : Recognizing diagonals.



Indeed, both circuits represent diagonal matrices in which each diagonal entry is repeated (at least) twice. In the former case, the repetition is due to a multiplexed scalar acting on the least significant qubit, and in the latter there is no attempt to modify the least significant qubit.

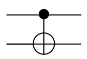
We now clarify the meaning of multiplexed generic gates in circuit diagrams, like that in the above circuit equivalence.

Notation. Let G be a generic gate. A specification U of a multiplexed-G gate can be any quantum multiplexor which effects a potentially different specification of G on the data qubits for each bitstring appearing on the select qubits. Of course, select qubits may carry a superposition of several bitstring states, in which case the behavior of the multiplexed gate is defined by linearity.

3.1 Quantum Multiplexors on Two Qubits

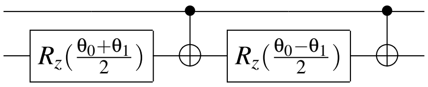
Perhaps the simplest quantum multiplexor is the *Controlled-NOT* (CNOT) gate.

$$\mathrm{CNOT} = I \oplus \sigma_x = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} = \qquad (14)$$



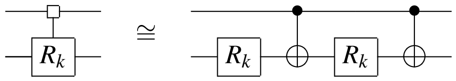
On bitstring states, the CNOT flips the second (data) bit if the first (select) bit is $|1\rangle$, hence the name Controlled-NOT. The CNOT is so common in quantum circuits that it has its own notation: a "•" on the select qubit connected by a vertical line to an "⊕" on the data qubit. This notation is motivated by the characterization of the CNOT by the formula $|b_1\rangle\,|b_2\rangle \mapsto |b_1\rangle\,|b_1 \text{ XOR } b_2\rangle$. Several CNOTs are depicted in Figure 3.

The CNOT, together with the one-qubit gates defined in §2, forms a universal gate library for quantum circuits.[4] In particular, we can use it as a building block to help construct more complicated multiplexors. For example, we can implement the multiplexor $R_z(\theta_0) \oplus R_z(\theta_1)$ by the following circuit.

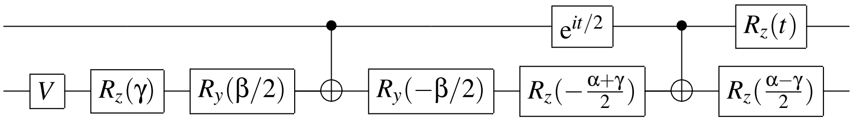

$R_z(\frac{\theta_0+\theta_1}{2})$
$R_z(\frac{\theta_0-\theta_1}{2})$

In fact, the exact same statement holds if we replace R_z by R_y (this can be verified by multiplying four matrices). We summarize the result with a circuit equivalence.

Theorem 4 : Demultiplexing a singly-multiplexed R_y or R_z.

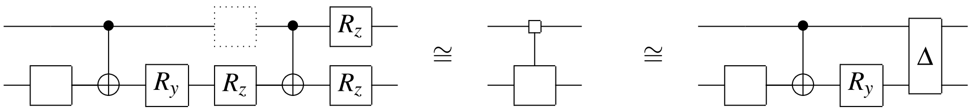


A similar decomposition exists for any $U \oplus V$ where U, V are one-qubit gates. The idea is to first unconditionally apply V on the less significant qubit, and then apply $A = UV^\dagger$, conditioned on the more significant qubit. Decompositions for such controlled-A operators are well known [3, 9]. Indeed, if we write $A = e^{it} R_z(\alpha) R_y(\beta) R_z(\gamma)$ by Theorem 1, then $U \oplus V$ is implemented by the following circuit.



Since V is a generic unitary, it can absorb adjacent one-qubit boxes, simplifying the circuit. We re-express the result as a circuit equivalence.

Theorem 5 : Decompositions of a two-qubit multiplexor [3]



Proof. The first equivalence is just a re-statement of what we have already seen; the second follows from it by applying a CNOT on the right to both sides and extracting a diagonal operator. ■

[4] This was first shown in [12]. The results in the present work also constitute a complete proof.

3.2 The Multiplexor Extension Property

The theory of n-qubit quantum multiplexors begins with the observation that whole circuits and even circuit equivalences can be multiplexed. This observation has non-quantum origins and can be exemplified by comparing two expressions involving conditionals in terms of a classical bit s.

- if (s) $A_0 \cdot B_0$ else $A_1 \cdot B_1$

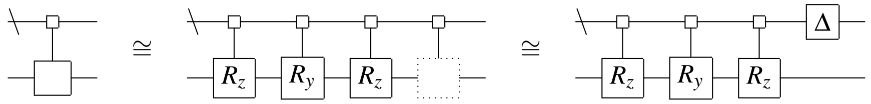
- $A_s \cdot B_s$. Here A_s means if (s) A_0 else A_1, with the syntax and semantics of (s?A_0:A_1) in the C programming language.

Indeed, one can either make a whole expression conditional on s or make each term conditional on s — the two behaviors will be identical. Similarly, one can multiplex a whole equation (with two different instantiations of every term) or multiplex each of its terms. The same applies to quantum multiplexing by linearity.

Multiplexor Extension Property (MEP). Let $C \equiv D$ be an equivalence of quantum circuits. Let C' be obtained from C by adding a wire which acts as a multiplexor control for every generic gate in C, and let D' be obtained from D similarly. Then $C' \equiv D'$.

Consider the special case of quantum multiplexors with a single data bit, but arbitrarily many select bits. We seek to implement such multiplexors via CNOTs and one-qubit gates, beginning with the following decomposition.

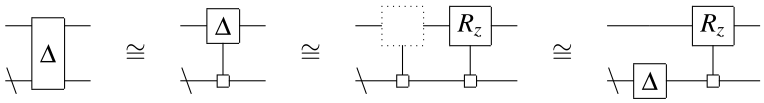
Theorem 6 : ZYZ decomposition for single-data-bit multiplexors.



Proof. Apply the MEP to Theorem 1, and Theorem 3 to the result. ■

The diagonal gate appearing on the right can be recursively decomposed.

Theorem 7 : Decomposition of diagonal operators [8].



Proof. The first equivalence asserts that any diagonal gate can be expressed as a multiplexor of diagonal gates. This is true because diagonal gates possess the block-diagonal structure characteristic of multiplexors, with each block being diagonal. The second equivalence amounts to the MEP applied to the obvious fact that a one-qubit gate given by a diagonal matrix is a scalar multiple of an R_z gate. The third follows from Theorem 3. ■

It remains to decompose the other gates appearing on the right in the circuit diagram of Theorem 6. We shall call these gates *multiplexed R_z (or R_y) gates*,[5] as, e.g., the rightmost would apply a different R_z gate to the data qubit for each classical configuration of the select bits. While efficient implementations are known [8, 21], the usual derivations involve large matrices and Gray codes.

[5] Other authors have used the term *uniformly-controlled rotations* to describe these gates [21].

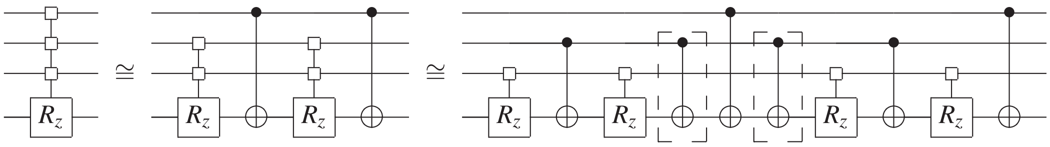
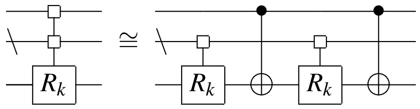


Figure 2: The recursive decomposition of a multiplexed R_z gate. The boxed CNOT gates may be canceled.

Theorem 8 : Demultiplexing multiplexed R_k gates, $k = y,z$ [8, 21].



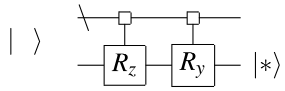
Proof. Apply the MEP to Theorem 4. ■

It is worth noting that since every gate appearing in Theorem 8 is symmetric, the order of gates in this decomposition may be reversed. Recursive application of Theorem 8 can decompose any multiplexed rotation into basic gates. In the process, some CNOT gates cancel, as is illustrated in Figure 2. The final CNOT count is 2^k, for k select bits.

4 The Preparation of Quantum States

We present an asymptotically-optimal technique for the initialization of a quantum register. The problem has been known for some time in quantum computing, and it was considered in [11, 20, 26] after the original formulation [10] of the quantum circuit model. It is also a computational primitive in designing larger quantum circuits.

Theorem 9 : Disentangling a qubit. *An arbitrary $(n+1)$-qubit state can be converted into a separable (i.e., unentangled) state by a circuit shown below. The resulting state is a tensor product involving a desired basis state ($|0\rangle$ or $|1\rangle$) on the less significant qubit.*



Proof. We show how to produce $|0\rangle$ on the least significant bit; the case of $|1\rangle$ is similar. Let $|\psi\rangle$ be an arbitrary $(n+1)$-qubit state. Divide the 2^{n+1}-element vector $|\psi\rangle$ into 2^n contiguous 2-element blocks. Each is to be interpreted as a two-dimensional complex vector, and the c-th is to be labeled $|\psi_c\rangle$. We now determine $r_c, t_c, \varphi_c, \theta_c$ as in Equation 9.

$$R_z(-\varphi_c)R_y(-\theta_c)\,|\psi_c\rangle = r_c \mathrm{e}^{it_c}\,|0\rangle \tag{15}$$

Let $|\psi'\rangle$ be the n-qubit state given by the 2^n-element row vector with c-th entry $r_c \mathrm{e}^{it_c}$, and let U be the block diagonal sum $\bigoplus_c R_y(-\theta_c)R_z(-\varphi_c)$. Then $U\,|\phi\rangle = |\phi'\rangle\,|0\rangle$, and U may be implemented by a multiplexed R_z gate followed by a multiplexed R_y. ■

We may apply Theorem 8 to implement the $(n+1)$-bit circuit given above with 2^{n+1} CNOT gates. A slight optimization is possible given that the gates on the right-hand size in Theorem 8 can be optionally reversed, as explained above. Indeed, if we reverse the decomposition of the multiplexed R_y gate, its first gate (CNOT) will cancel with the last gate (CNOT) from the decomposed multiplexed R_z gates. Thus, only $2^{n+1}-2$ CNOT gates are needed.

Applying Theorem 9 *recursively* can reduce a given n-qubit quantum state $|\psi\rangle$ to a scalar multiple of a desired bitstring state $|b\rangle$; the resulting circuit C uses $2^{n+1}-2n$ CNOT gates. To go from $|b\rangle$ to $|\psi\rangle$, apply the gates of C in reverse order and inverted. We shall call this the inverse circuit, $C^\dagger$.

The state preparation technique can be used to decompose an arbitrary unitary operator U. The idea is to construct a circuit for $U^\dagger$ by iteratively applying state preparation. Indeed, an operator is entirely determined by its behavior on basis vectors. To this end, each iteration needs to implement the correct behavior on a new basis vector while preserving the behavior on previously processed basis vectors. This idea has been tried before [20, 31], but with methods less efficient than Theorem 9. We outline the procedure below.

- At step 0, apply Theorem 9 to find a circuit C_0 that maps $U|0\rangle$ to a scalar multiple of $|0\rangle$. Let $U_1 = C_0 U$.

- At step j, apply Theorem 9 to find a circuit C_j that maps $U|j\rangle$ to a scalar multiple of $|j\rangle$. Importantly, the construction of C_j and the previous steps of the algorithm ensure $C_j|i\rangle = |i\rangle$ for all $i < j$. Define $U_{j+1} = C_j U_j$.

- U_{2^n-1} will be diagonal, and may be implemented by a circuit D via Theorem 7.

- Finally, $U = C_0{}^\dagger C_1{}^\dagger \ldots C_{2^n-2}{}^\dagger D$

Thus 2^n-1 state preparation steps and 1 diagonal operator are used. The final CNOT count is $2\times 4^n - (2n+3)\times 2^n + 2n$. For $n > 2$, we improve upon the best previously published technique to decompose unitary operators column by column [31], as can be seen in Table 1.

5 A Functional Decomposition for Quantum Logic

Below we introduce a decomposition for quantum logic that is analogous to the well-known Shannon decomposition of Boolean functions ($f = x_i f_{x_i=1} + \bar{x}_i f_{x_i=0}$). It expresses an arbitrary n-qubit quantum operator in terms of $(n-1)$-qubit operators (cofactors) by means of quantum multiplexors. Applying this decomposition recursively yields a synthesis algorithm, for which we compute gate counts.

5.1 The Cosine-Sine Decomposition

We recall the Cosine-Sine Decomposition from matrix algebra.[6] It has been used explicitly and regularly to build quantum circuits [30, 21] and has also been employed inadvertently [32, 6].

The CSD states that an even-dimensional unitary matrix $U \in \mathbb{C}^{\ell\times\ell}$ can be decomposed into smaller unitaries A_1, A_2, B_1, B_2 and real diagonal matrices C, S such that $C^2 + S^2 = I_{\ell/2}$.

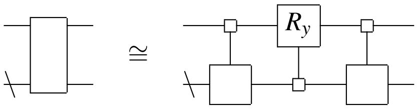
$$U = \begin{pmatrix} A_1 & \\ & B_1 \end{pmatrix} \begin{pmatrix} C & -S \\ S & C \end{pmatrix} \begin{pmatrix} A_2 & \\ & B_2 \end{pmatrix}$$

For 2×2 matrices U, we may extract scalars out of the left and right factors to recover Theorem 1. For larger U, the left and right factors $A_j \oplus B_j$ are quantum multiplexors controlled by the most significant qubit which determines whether A_j or B_j is to be applied to the lower order qubits. The central factor has the same structure as the R_y gate. A closer inspection reveals that it applies a different R_y gate to the most

[6] Source code for computing the CSD can be obtained from Matlab by typing “which gsvd” at a Matlab command prompt. On most laptops this numerical computation scales to ten-qubit quantum operators, i.e., 1024 × 1024 matrices.

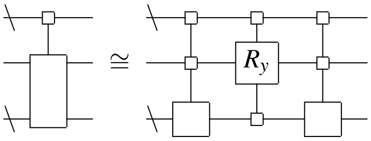
significant bit for each classical configuration of the low order bits. Thus the CSD can be restated as the following equivalence of generic circuits.

Theorem 10 : The Cosine-Sine Decomposition [15, 24].



It has been observed that this theorem may be recursively applied to the side factors on the right-hand side [30]. Indeed, this can be achieved by adding more qubits via the MEP, as shown below.

Theorem 11 : A multiplexed Cosine-Sine Decomposition [30].



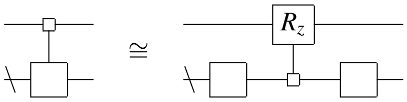
We may now outline the best previously published generic quantum logic synthesis algorithm [21]. Iterated application of Theorem 11 to the decomposition of Theorem 10 gives a decomposition of an arbitrary unitary operator into single-data-bit QMUX gates, some of which are already multiplexed R_y gates. Those which are not can be decomposed into multiplexed rotations by Theorem 6, and then all the multiplexed rotations can be decomposed into elementary gates by Theorem 8.

One weakness of this algorithm is that it cannot readily take advantage of hand-optimized generic circuits on low numbers of qubits [34, 33, 27, 25]. This is because it does not recurse on generic operators, but rather on multiplexors.

5.2 Demultiplexing Multiplexors, and the Quantum Shannon Decomposition

We now give a novel, simpler decomposition of single-select-bit multiplexors whose two cofactors are generic operators. As will be shown later, it leads to a more natural recursion, with known optimizations in end-cases [34, 33, 27, 25].

Theorem 12 : Demultiplexing a multiplexor.



Proof. Let $U = U_0 \oplus U_1$ be the multiplexor of choice; we formulate and solve an equation for the unitaries required to implement U in the manner indicated above. We want unitary V, W and unitary diagonal D satisfying $U = (I \otimes V)(D \oplus D^\dagger)(I \otimes W)$. In other words,

$$\begin{pmatrix} U_1 & \\ & U_2 \end{pmatrix} = \begin{pmatrix} V & \\ & V \end{pmatrix} \begin{pmatrix} D & \\ & D^\dagger \end{pmatrix} \begin{pmatrix} W & \\ & W \end{pmatrix} \tag{16}$$

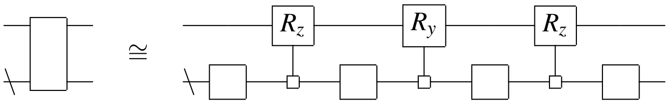
Multiplying the expressions for U_1 and U_2, we cancel out the W-related terms and obtain $U_1 {U_2}^\dagger = VD^2V^\dagger$. Using this equation, one can recover D and V from $U_1 {U_2}^\dagger$ by a standard computational primitive called diagonalization. Further, $W = DV^\dagger U_2$. It remains only to remark that for D diagonal, the matrix $D \oplus D^\dagger$ is in fact a multiplexed R_z gate acting on the most significant bit in the circuit. ■

	Number of qubits and gate counts							
Synthesis Algorithm	1	2	3	4	5	6	7	n
Original QR decomp. [3, 9]	———							$O(n^3 4^n)$
Improved QR decomp. [20]	———							$O(n4^n)$
Palindrome transform [2]	———							$O(n4^n)$
QR [31, Table I]	0	4	64	536	4156	22618	108760	$O(4^n)$
QR (Theorem 9)	0	8	62	344	1642	7244	30606	$2\times 4^n-(2n+3)\times 2^n+2n$
CSD [21, p. 4]	0	8	48	224	960	3968	16128	$4^n-2\times 2^n$
QSD ($l=1$)	0	6	36	168	720	2976	12096	$(3/4)\times 4^n-(3/2)\times 2^n$
QSD ($l=2$)	0	3	24	120	528	2208	9024	$(9/16)\times 4^n-(3/2)\times 2^n$
QSD ($l=2$**, optimized)**	0	**3**	**20**	**100**	**444**	**1868**	**7660**	$(23/48)\times 4^n-(3/2)\times 2^n+4/3$
Lower bounds [27]	0	3	14	61	252	1020	4091	$\lceil\frac{1}{4}(4^n-3n-1)\rceil$

Table 1: A comparison of CNOT counts for unitary circuits generated by several algorithms (best results are in bold). We have labeled the algorithms by the matrix decomposition they implement. The results of this paper are boldfaced, including an optimized QR decomposition and three algorithms based on the Quantum Shannon Decomposition (QSD). Other rows represent previously published algorithms. Gate counts are not given for algorithms whose performance is not (generically) asymptotically optimal.

Using the new decomposition, we now demultiplex the two side multiplexors in the Cosine-Sine Decomposition (Theorem 10). This leads to the following decomposition of generic operators that can be applied recursively.

Theorem 13 : The Quantum Shannon Decomposition.



Hence an arbitrary n-qubit operator can be implemented by a circuit containing three multiplexed rotations and four generic $(n-1)$-qubit operators, which can be viewed as cofactors of the original operator.

5.3 Recursive Gate Counts for Universal Circuits

We present gate counts for the circuit synthesis algorithm implicit in Theorem 13. An important issue which remains is to choose the level at which to cease the recursion and handle end-cases with special purpose techniques.

Thus, let c_j be the least number of CNOT gates needed to implement a j-qubit unitary operator using some known quantum circuit synthesis algorithm. Then Theorem 13 implies the following.

$$c_j \;\leq\; 4c_{j-1}+3\times 2^{j-1} \tag{17}$$

One can now apply the decomposition of Theorem 13 recursively, which corresponds to iterating the above inequality. If ℓ-qubit operators may be implemented using $\leq c_\ell$ CNOT gates, one can prove the following inequality for c_n by induction.

$$c_n \;\leq\; 4^{n-\ell}(c_\ell+3\times 2^{\ell-1})-3\times 2^{n-1} \tag{18}$$

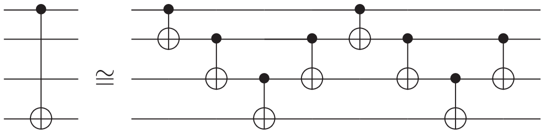


Figure 3: Implementing a long-range CNOT gate with nearest-neighbor CNOTs.

We have recorded in Table 1 the formula for c_n with recursion bottoms out at one-qubit operators ($l = 1$ and $c_l = 0$), or two-qubit operators ($l = 2$ and $c_l = 3$ by [27, 34, 33]). In either case, we improve on the best previously published algorithm (cf. [21]). However, to obtain our advertised CNOT-count of $(23/48) \times 4^n - (3/2) \times 2^n + 4/3$ we shall need two further optimizations. Due to their more technical nature, they are discussed in the Appendix.

Note that for $n = 3$, only 20 CNOTs are needed. This is the best known three-qubit circuit at present (cf. [32]). Thus, our algorithm is the first efficient n-qubit circuit synthesis routine which also produces a best-practice circuit in a small number of qubits.

6 Nearest-Neighbor Circuits

A frequent criticism of quantum logic synthesis (especially highly optimized circuits which nonetheless must conform to large theoretical lower bounds on the number of gates) is that the resulting circuits are physically impractical. In particular, naïve gate counts ignore many important physical problems which arise in practice. Many such are grouped under the topic of quantum architectures [5, 23], including questions of (1) how best to arrange the qubits and (2) how to adapt a circuit diagram to a particular physical layout. A *spin chain*[7] is perhaps the most restrictive architecture: the qubits are laid out in a line, and all CNOT gates must act only on adjacent (nearest-neighbor) qubits. As spin-chains embed into two and three dimensional grids, we view them as the most difficult architecture from the perspective of layout. The work in [14] shows how to adapt Shor's algorithm to spin-chains without asymptotic increase in gate counts. However, it is not yet clear if generic circuits can be adapted similarly.

As shown next, our circuits adapt well to the spin-chain limitations. Most CNOT gates used in our decomposition already act on nearest neighbors, e.g., those gates implementing the two-qubit operators. Moreover, Fig. 2 shows that only 2^{n-k} CNOT gates of length k (where the length of a local CNOT is 1) will appear in the circuit implementing a multiplexed rotation with $(n-1)$ control bits. Figure 3 decomposes a length k CNOT into $4k-4$ length 1 CNOTs. Summation shows that $9 \times 2^{n-1} - 8$ nearest-neighbor CNOTs suffice to implement the multiplexed rotation. Therefore restricting CNOT gates to nearest-neighbor interactions increases CNOT count by at most a factor of nine.

7 Conclusions and Future Work

Our approach to quantum circuit synthesis emphasizes simplicity, a well-pronounced top-down structure, and practical computation via the Cosine-Sine Decomposition. By introducing the quantum multiplexor and optimizing its singly-controlled version, we derived a quantum analogue of the well-known Shannon decomposition of Boolean functions. Applying this decomposition recursively to quantum operators leads to a circuit synthesis algorithm in terms of quantum multiplexors. As seen in Table 1, our techniques

[7] The term arises since the qubit is also commonly thought of as an abstract particle with quantum spin $1/2$.

achieve the best known controlled-not counts, both for small numbers of qubits and asymptotically. Our approach has the additional advantage that it co-opts all results on small numbers of qubits – e.g., future specialty techniques developed for three-qubit quantum logic synthesis can be used as terminal cases of our recursion. We have also discussed various problems specific to quantum computation, specifically initialization of quantum registers and mapping to the nearest-neighbor gate library.

Acknowledgements. We are grateful to Professors Dianne O'Leary from the Univ. of Maryland and Joseph Shinnerl from UCLA for their help with computing the CS decomposition in Matlab; to Gavin Brennen at NIST and Jun Zhang at UC Berkeley for their helpful comments, and the authors of quant-ph/0406003, whose package Qcircuit.tex produced almost all figures.

This work is funded by the DARPA QuIST program and an NSF grant. SSB is supported by an NRC postdoctoral fellowship. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing official policies or endorsements of employers and funding agencies. Certain commercial equipment or instruments may be identified in this paper to specify experimental procedures. Such identification is not intended to imply recommendation or endorsement by the National Institute of Standards and Technology.

Appendix A: Additional Circuit Optimizations

Section 5 shows that recursively applying the Quantum Shannon Decomposition until only ℓ-qubit operators remain produces circuits with at most $4^{n-\ell}(c_\ell + 3 \times 2^{\ell-1}) - 3 \times 2^{n-1}$ CNOT gates.

To obtain our advertised CNOT-count, we apply additional optimizations below that reduce $(4^{n-\ell} - 1)/3$ CNOTs in general, and an additional $4^{n-2} - 1$ in the case $\ell = 2, c_\ell = 3$. This results in the following, final CNOT count.

$$c_n \;\leq\; (23/48) \times 4^n - (3/2) \times 2^n + 4/3 \tag{19}$$

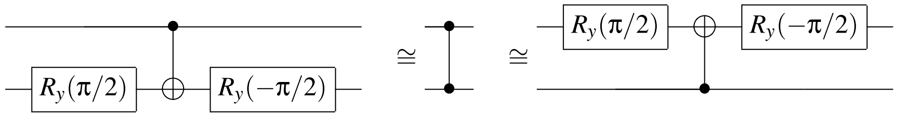
Observe that the leading term is slightly below $4^n/2$, whereas the leading term in the lower bound from [27] is $4^n/4$. Thus, our result cannot be improved by more than a factor of two.

A.1 Implementing Multiplexed-R_y with Controlled-Z

Recall the two-qubit *controlled-Z* gate, given by the following matrix.

$$\text{Controlled-Z} = \begin{pmatrix} 1 & & & \\ & 1 & & \\ & & 1 & \\ & & & -1 \end{pmatrix} \tag{20}$$

The controlled-Z gate is commonly denoted by a "•" on each qubit, connected by a vertical line, as shown in the diagram below. This gate can be implemented using a single CNOT with the desired orientation, and one-qubit gates (whose physical realizations are typically simpler).

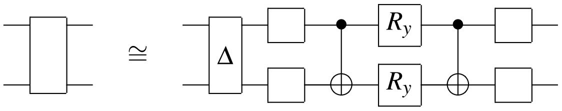


The statements and proofs of Theorem 8 and Figure 2 still hold for multiplexed R_y gates if all CNOTs are replaced with controlled-Z gates. Thus the central multiplexed R_y in the Cosine-Sine decomposition may be implemented with 2^n controlled-Z gates, of which one is initial (or terminal). As the initial controlled-Z gate is diagonal, it may be absorbed into the neighboring generic multiplexor. This saves one gate *at each step of the recursion*, for the total savings of $(4^{n-\ell} - 1)/3$ CNOT gates.

A.2 Extracting Diagonals to Improve Decomposition of Two-Qubit Operators

Terminate the recursion when only two-qubit operators remain; there will be 4^{n-2} of them. These two-qubit operators all act on the least significant qubits and are separated by the controls of multiplexed rotations. To perform better optimization, we recite a known result on the decomposition of two-qubit operators.

Theorem 14 : Decomposition of a two-qubit operator [27].



We use Theorem 14 to decompose the rightmost two-qubit operator; migrate the diagonal through the select bits of the multiplexor to the left, and join it with the two-qubit operator on the other side. Now we decompose this operator, and continue the process. Since we save one CNOT in the implementation of every two-qubit gate but the last, we improve the $l = 2$, $c_l = 3$ count by $4^{n-2} - 1$ gates.

References

[1] The ARDA Roadmap For Quantum Information Science and Technology, `http://qist.lanl.gov.`

[2] A. V. Aho and K. M. Svore. Compiling quantum circuits using the palindrome transform. e-print, quant-ph/0311008.

[3] A. Barenco, C. 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*, 52:3457, 1995.

[4] C. H. Bennett and G. Brassard. Quantum cryptography: Public-key distribution and coin tossing. In *Proceedings of IEEE International Conference on Computers, Systems, and Signal Processing*, page 175179, Bangalore, India, 1984. IEEE Press.

[5] G.K. Brennen, D. Song, and C.J. Williams, Quantum-computer architecture using nonlocal interactions. *Phys. Rev. A.(R)*, 67:050302, 2003.

[6] S.S. Bullock, Note on the Khaneja Glaser decomposition. *Quant. Info. and Comp.* 4:396, 2004.

[7] S. S. Bullock and I. L. Markov. An elementary two-qubit quantum computation in twenty-three elementary gates. In *Proceedings of the 40th ACM/IEEE Design Automation Conference*, pages 324–329, Anaheim, CA, June 2003. Journal: *Phys. Rev. A* 68:012318, 2003.

[8] S. S. Bullock and I. L. Markov, Smaller circuits for arbitrary n-qubit diagonal computations. *Quant. Info. and Comp.* 4:27, 2004.

[9] G. Cybenko: "Reducing Quantum Computations to Elementary Unitary Operations", *Comp. in Sci. and Engin.*, March/April 2001, pp. 27-32.

[10] D. Deutsch, Quantum Computational Networks, *Proc. R. Soc. London A* 425:73, 1989.

[11] D. Deutsch, A. Barenco, A. Ekert, Universality in quantum computation. *Proc. R. Soc. London A* 449:669, 1995.

[12] D. P. DiVincenzo. Two-bit gates are universal for quantum computation. *Phys. Rev. A* 15:1015, 1995.

[13] R. P. Feynman. Quantum mechanical computers. *Found. Phys.*, 16:507–531, 1986.

[14] A. G. Fowler, S. J. Devitt, L. C. L. Hollenberg, “Implementation of Shor’s Algorithm on a Linear Nearest Neighbour Qubit Array”, *Quant. Info. Comput.* 4, 237-251 (2004).

[15] G.H. Golub and C. vanLoan, *Matrix Computations*, Johns Hopkins Press, 1989.

[16] L. K. Grover. Quantum mechanics helps with searching for a needle in a haystack. *Phys. Rev. Let.*, 79:325, 1997.

[17] W. N. N. Hung, X. Song, G. Yang, J. Yang, and M. Perkowski. Quantum logic synthesis by symbolic reachability analysis. In *Proceedings of the 41st Design Automation Conference*, San Diego, CA, June 2004.

[18] K. Iwama, Y. Kambayashi, and S. Yamashita. Transformation rules for designing cnot-based quantum circuits. In *Proceedings of the 39th Design Automation Conference*, pages 419–425, 2002.

[19] R. Jozsa and N. Linden, On the role of entanglement in quantum computational speed-up. e-print, quant-ph/0201143.

[20] E. Knill, Approximation by quantum circuits. LANL report LAUR-95-2225.

[21] M. Möttönen, J. J. Vartiainen, V. Bergholm, and M. M. Salomaa. Quantum circuits for general multiqubit gates. *Phys. Rev. Let.*, 93:130502, 2004.

[22] M. A. Nielsen and I. L. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, 2000.

[23] M. Oskin, F.T. Chong, I. Chuang, and J. Kubiatowicz, Building quantum wires: the long and the short of it. In 30^{th} *Annual International Symposium on Computer Architecture (ISCA)*, June 2003.

[24] C. C. Paige and M. Wei. History and generality of the CS decomposition. *Linear Alg. and App.*, 208:303, 1994.

[25] V. V. Shende, S. S. Bullock, and I. L. Markov. Recognizing small-circuit structure in two-qubit operators. *Phys. Rev. A*, 70:012310, 2004.

[26] V. V. Shende and I. L. Markov. Quantum Circuits for Incompletely Specified Two-Qubit Operators *Quant. Inf. and Comput.*, vol.5, no.1, pp. 49-58, January 2005.

[27] V. V. Shende, I. L. Markov, and S. S. Bullock. Smaller two-qubit circuits for quantum communication and computation. In *Design, Automation, and Test in Europe*, pages 980–985, Paris, France, February 2004. Journal: *Phys. Rev. A*, 69:062321, 2004.

[28] V. V. Shende, A. K. Prasad, I. L. Markov, and J. P. Hayes. Synthesis of reversible logic circuits. *IEEE Transactions on Computer Aided Design*, 22:710, 2003.

[29] P. Shor. Polynomial-time algorithms for prime factorization and discrete logarithm on a quantum computer. *SIAM Journal on Computing*, 26(5):1484–1509, 1997.

[30] R. R. Tucci, A Rudimentary Quantum Compiler. e-print, `quant-ph/9805015`.

[31] J. J. Vartiainen, M. Möttönen, and M. M. Salomaa. Efficient decomposition of quantum gates. *Phys. Rev. Let.*, 92:177902, 2004.

[32] F. Vatan and C. Williams. Realization of a general three-qubit quantum gate. e-print, quant-ph/0401178.

[33] F. Vatan and C. Williams. Optimal quantum circuits for general two-qubit gates. *Phys. Rev. A*, 69:032315, 2004.

[34] G. Vidal and C. M. Dawson. A universal quantum circuit for two-qubit transformations with three CNOT gates. *Phys. Rev. A*, 69:010301, 2004.

[35] J. Zhang, J. Vala, S. Sastry, and K. B. Whaley. Exact two-qubit universal quantum circuit. *Phys. Rev. Let.*, 91:027903, 2003.

[36] V. V. Zhirnov, R. K. Cavin, J. A. Hutchby, and G. I. Bourianoff. Limits to binary logic switch scaling — a gedanken model. *Proceedings of the IEEE*, 91(11):1934–1939, November 2003.