An Arbitrary Two-qubit Computation In 23 Elementary Gates Or Less *

Stephen S. Bullock and Igor L. Markov
The University of Michigan, Ann Arbor
stephnsb@math.lsa.umich.edu imarkov@eecs.umich.edu

Abstract

Quantum circuits currently constitute a dominant model for quantum computation [14]. Our work addresses the problem of constructing quantum circuits to implement an arbitrary given quantum computation, in the special case of two qubits. We pursue circuits without ancilla qubits and as small a number of elementary quantum gates [1, 9] as possible. Our lower bound for worst-case optimal two-qubit circuits calls for at least 17 gates: 15 one-qubit rotations and 2 CNOTs. To this end, we constructively prove a worst-case upper bound of 23 elementary gates, of which at most 4 (CNOTs) entail multi-qubit interactions. Our analysis shows that previously known synthesis algorithms, although more general, entail much larger quantum circuits than ours in the special case of two qubits. One such algorithm [4] has a worst case of 61 gates of which 18 may be CNOTs.

Our techniques rely on the *KAK* decomposition from Lie theory as well as the polar and spectral (symmetric Shur) matrix decompositions from numerical analysis. They are related to the canonical decomposition of a two-qubit gate with respect to the "magic basis" of phase-shifted Bell states [11, 12]. We extend this decomposition in terms of elementary gates for quantum computation.

Categories: A.1 General Literature, Introductory and Survey. B.2.2 Arithmetic and Logic Structures, Performance Analysis and Design Aids, Worst-case Analysis. B.6.3 Logic Design, Design Aids, Automatic Synthesis and Optimization. F.1 Theory of Computing, Computation by Abstract Devices, Models of Computation.

Terms: Design, Algorithms, Performance.

Keywords: Quantum Circuits, Synthesis, Optimization, Elementary Gates, CNOT, Qubit, Algorithms, Lower Bounds, Lie Theory, Circuit Decomposition.

1 Introduction

Quantum Computing gained popularity due to predictions that Moore's law will soon end, in that future improvements in commercial semiconductors will be slower and less far-reaching. Indeed, quantum computation has been demonstrated at much smaller scales than contemporary semiconductor devices. In liquid NMR technologies [14], demonstrated by IBM and MIT researchers, a small organic molecule can store quantum information. Quantum computation can be performed by a series of RF pulses. In ion trap technologies developed at NIST and the University of Michigan, chains of ions suspended in a magnetic field store quantum states, and quantum computation is performed by laser pulses. Another technology, currently under development at Michigan State and Los Alamos, exploits single electrons floating over the surface of liquid helium. Quantum interactions are initiated through electrodes that terminate below a vessel containing liquid helium. Various other technologies are being pursued, including (a) many based on solid-state physics, some silicon-based, (b) optical, where photons carry quantum states, and (c) utilizing quantized currents in superconductors. Quantum computations enabled by these technologies can be described by a unified mathematical notation based on matrices. Indeed, the axioms of Quantum Mechanics describe the dynamics of quantum states in terms of linear operators on Hilbert space. Matrices represent such linear operators.

We point out that quantum computers are not merely miniaturized versions of classical computers. In order to evaluate the power of quantum computation, one uses the formalism of quantum circuit schematics that captures computations possible with the implementation technologies above. Currently quantum circuits are the dominant model of quantum computation [14]. While being relevant to the work in experimental physics, they also enable rigorous and relevant results in mathematics and computer science. In particular, it has been shown that some quantum algorithms run asymptotically faster than their classical counterparts. The most striking example is Shor's number-factoring algorithm [14], which can break the widely used RSA cryptosystem in polynomial time. This is now infeasible using the fastest modern computers.

Another reason to study quantum computing is its universality. Classical computation is commonly used to model numerous aspects of the physical world — from sending mail and reproducing paintings to simulating natural disasters and armed conflicts. However, as first suggested by Richard Feynman, classical computation is fundamentally limited in its ability to simulate quantum-mechanical effects. Nevertheless, performing such simulations is extremely important to advance our understanding theoretical and astro-physics. Additionally, a number of important applications are driven by quantum effects, such as radioactive decay, nuclear fusion and photosynthesis. Being able to efficiently simulate those at the atomic and elementary-particle scale may require quantum computers in the future.

Quantum effects have already been used in quantum cryptography, which is based on the fact that a quantum

*Partially supported by the University of Michigan Mathematics department VIGRE summer stipend and the DARPA QuIST program. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing official policies of endorsements, either expressed or implied, of respective funding institutions.

measurement of a secret transmission will alter the contents of the transmission. By tracking check-sums, the intended recipient will be aware of the breach and can close the communication channel. A number of quantum key distribution (QKD) protocols exploit this effect, starting with the famous BB84 and BB92 [14]. Those protocols have been successfully implemented with support from DARPA and NSA, as well as in Europe and Australia. Recent research in the field shows that a powerful quantum computer could, in principle, break the security of such QKD protocols. This further motivates research in quantum computing.

Our work is analogous to the development of algorithms for automated synthesis of classical logic circuits [6], which started in the 1960s. We focus on the synthesis of quantum circuits from functional specifications.

Quantum computations can be described by unitary matrices. In order to effect a quantum computation on a quantum computer, one must decompose such a matrix into a quantum circuit, which consists of elementary quantum gates connected by Kronecker (tensor) and matrix products. Those connections are often represented using quantum circuit schematics. The matrix and graph-based notations for quantum circuits are analogous to how classical logic circuits are modeled by Boolean polynomials, e.g., during logic synthesis. Classical logic synthesis relies on algorithms for factoring polynomials, and we observe that quantum synthesis algorithms benefit from matrix factorizations. This is confirmed by our research, and motivates one to take a deeper look at the algebraic theory behind matrix factorizations (*SVD*, *QR*, polar and others) and possible applications to quantum circuit synthesis.

It is well-known that any one-qubit computation can be implemented using *three* one-qubit elementary gates (rotations) or less [1]. Our work answers a similar question about arbitrary two-qubit computations assuming that CNOT gates can be used in addition to single-qubit rotations, without ancilla qubits. Our lower bound that calls for at least *seventeen* elementary gates: *fifteen* rotations and *two* CNOTs. We also constructively prove that *twenty three* elementary gates suffice to implement an arbitrary two-qubit computation. At most *four* of those are CNOTs and the rest are single-qubit gates. In comparison, a previously known construction [1, 4] implies *sixty-one* gates of which *eighteen* are CNOTs. While this construction is more general than ours, for two-qubit computations our algorithm generates far fewer gates in the worst (generic) case. The savings in the number of multi-qubit gates (CNOTs) are particularly dramatic.

The techniques we developed rely on the *KAK* decomposition from Lie theory [10] as well as the polar and spectral (symmetric Shur) matrix decompositions from numerical analysis [7] and operator theory. They are related to the canonical decomposition of a two-qubit gate with respect to the "magic basis" of phase-shifted Bell states [11, 12]. We further extend this decomposition in terms of elementary gates for quantum computation.

Our work can be compared to the GQC online "quantum compiler" [8].[1] That program inputs a 4×4 unitary U and returns a "canonical decomposition" which is not, in a strict sense, a circuit in terms of elementary gates. It also returns a circuit that computes CNOT using U and one-qubit gates. When U is used only once, this easily yields a circuit decomposition of U in terms of elementary gates. However, not all input matrices can be processed successfully.[2]

[1] We point out that the term "compiler" in classical computing means "translator from a high-level description to a register-transfer level (RTL) description, e.g., machine codes". The task of producing circuits with given function is commonly referred to as "circuit synthesis". In this context, digital circuits are called "logic circuits".

Algorithm	decomp.	# elem. gates	# CNOTs	# var 1-qubit gates
[4]	*QR*	61	18	39
Our #1	u. *KAK*	23	4	19
Our #2	u. *KAK*	28	8	15(sharp)
Our lower bounds		17	2	15

Table 1: A summary of our quantum circuit decomposition results for two-qubit computations in terms of elementary gates.

Our research emphasizes several technical ideas for synthesis of quantum circuits.

- Continued use of matrix decompositions from numerical analysis and Lie theory: *polar*, *spectral* and *KAK*.
- Focus on matrix decompositions that are intrinsic to unitary matrices, e.g., *KAK* of $SU(2^n)$, and include multiple non-trivial unitary factors.
- Using entanglement in the course of synthesis by considering computations mapping unentangled basis states into highly entangled basis states, with the purpose of recognizing quantum computations implementable with one-qubit gates only.
- Incremental reduction of existing quantum circuits by local optimization; exploiting degrees of freedom in circuit synthesis may be useful to expose additional reductions.

These ideas are covered in more depth in the technical discussion below and lead to quantum circuit decompositions for two-qubit computations summarized in Table 1.

The remainder of the paper is organized as follows. The necessary background and relevant prior work are covered in Section 2. Our contributions to two-qubit synthesis are presented in Section 3, followed by an account of their empirical validation in Section 4. Conclusions and ongoing work are discussed in 5.

2 Background

Quantum states and quantum circuits are governed by the laws of quantum mechanics: k-qubit states are 2^k-dimensional vectors, i.e., complex linear combinations of 0-1 bit-strings of length k. A quantum computation acting on k qubits (k inputs and k outputs) is modelled by a unitary $2^k\times 2^k$-matrix [14]. We denote such matrices by $U(2^k)=\{M\in(2^k\times 2^k)\text{-matrices}|MM^*=\mathbf{1}\}$. $O(2^k)$ represents those matrices from $U(2^k)$ with real entries. $SU(2^k)$ and $SO(2^k)$ are the respective subsets with determinant one. Below, we will consider two generic elements of $SU(2)$: $A=\underline{\alpha}E_{11}+(-\beta)E_{12}+\bar{\beta}E_{21}+\bar{\alpha}E_{22}$ and $B=\gamma E_{11}+(-\delta)E_{12}+\bar{\delta}E_{21}+\bar{\gamma}E_{22}$ with $1=|\alpha|^2+|\beta|^2=|\gamma|^2+|\delta|^2$. Such a parameterization of $SU(2)$ can be verified directly. We largely ignore the effects of quantum measurement that is typically performed after a quantum circuit is applied, but we use the fact that any measurement is invariant under a global phase change. In mathematical terms, this means that any computation in $U(2^k)$ can be represented in normalized form by a matrix from $SU(2^k)$.

[2] As of December 2002, the quantum compiler [8] fails on $exp\left(i\begin{bmatrix}0&1&2&3\\1&0&4&5\\2&4&0&6\\3&5&6&0\end{bmatrix}\right)$. The authors are aware of this problem.

2.1 Quantum circuits and elementary gates for quantum computation

Our work focuses on combinational quantum circuits, which are directed acyclic graphs where every vertex represents a gate. An output of a gate can be connected to exactly one input of another gate or one circuit output. A similar restriction applies to gate inputs. Examples of quantum circuits are shown in Figures 1 and 2.

Following [1], we attempt to express arbitrary computations using as small numbers of elementary gates as possible. In order to write matrix elements of particular gates, we order the elements of the computational basis lexicographically [14]. The computation implemented by several gates acting independently on different qubits can be described by the Kronecker (tensor) product $\otimes$ of their matrices. In the usual computational basis $|00\rangle, |01\rangle, |10\rangle, |11\rangle$ ordered in the dictionary order, the matrix in $U(4)$ representing $A \otimes B$ (for A and B defined above) is

$$(A \otimes B) = \begin{pmatrix} \alpha B & -\beta B \\ \bar{\beta} B & \bar{\alpha} B \end{pmatrix} \tag{1}$$

As an illustration of tensor product, consider the well-known Hadamard gate, which transforms the computational basis states $|0\rangle$ and $|1\rangle$ to superposition states $(|0\rangle + |1\rangle)/\sqrt{2}$ and $(|0\rangle - |1\rangle)/\sqrt{2}$ respectively. It is given by $H = \frac{\sqrt{2}}{2}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$. Then in terms of the ordered two-qubit computational basis $|00\rangle$, $|01\rangle$, $|10\rangle$, and $|11\rangle$, the two-qubit computation $H \otimes H$ is given by

$$H \otimes H = \frac{1}{2}\begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix} \tag{2}$$

Composition of multiple quantum computations is described by the matrix product. However, as most circuit diagrams are read left-to-right, the order in respective matrix expressions is reversed. For example, the expression $(A \otimes B)(C \otimes D)$ corresponds to a two-qubit circuit where C acts on the top line and D on the bottom line, followed by A acting on the top line and B on the bottom line. Since the two lines do not interact, the same computation is performed by AC acting on the top line and BD acting on the bottom line independently, i.e., $(A \otimes B)(C \otimes D) = (AC \otimes BD)$. Sometimes this identity allows one to simplify quantum circuits and reduce their gate counts.

We distinguish two versions of the `CNOT` gate, `topCNOT` and `botCNOT` conditioned on the top and bottom lines respectively: (i) `botCNOT` exchanges $|01\rangle \leftrightarrow |11\rangle$, i.e. `CNOT` controlled by the top line, and (ii) `topCNOT` exchanges $|10\rangle \leftrightarrow |11\rangle$. Those gates can be represented by matrices:

$$\texttt{topCNOT} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \quad \texttt{botCNOT} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix} \tag{3}$$

Gate library. If every gate in a circuit permutes elements of the computational basis, the circuit performs *classical* computation [17, 15]. In order to achieve what cannot be achieved by AND-OR-NOT circuits, some gates must create superpositions. We consider the following library of *elementary* one- and two-qubit gates [1]:

- $R_y(\theta) = \begin{pmatrix} \cos\theta/2 & \sin\theta/2 \\ -\sin\theta/2 & \cos\theta/2 \end{pmatrix}$ for all $0 \le \theta < 2\pi$;
- $R_z(\alpha) = \begin{pmatrix} e^{-i\alpha/2} & 0 \\ 0 & e^{i\alpha/2} \end{pmatrix}$ for all $0 \le \alpha < 2\pi$;
- The CNOT gate, conditioned on either line.

A given gate may, in principle, be applied to different lines. We do not restrict to which lines the above gates may be applied. Note that the gate library we use generates $U(4)$ up to global phase [4]. The relative costs of different gates depend on the implementation technology. In many technologies (solid- and liquid-state NMR) gates that involve multiple qubits, such as `CNOT`s, are much more expensive than one-qubit rotations. However, in other technologies (ion traps) one-qubit rotations are significantly more expensive than `CNOT`s.

An arbitrary one-qubit quantum computation can be implemented, up to phase, by *three* elementary gates using [1, Lemma 4.1], which decomposes a 2×2 unitary into

$$U = \begin{pmatrix} e^{i\delta} & 0 \\ 0 & e^{i\delta} \end{pmatrix} \begin{pmatrix} e^{-i\alpha/2} & 0 \\ 0 & e^{i\alpha/2} \end{pmatrix} \times \begin{pmatrix} \cos\theta/2 & \sin\theta/2 \\ -\sin\theta/2 & \cos\theta/2 \end{pmatrix} \begin{pmatrix} e^{-i\beta/2} & 0 \\ 0 & e^{i\beta/2} \end{pmatrix} \tag{4}$$

Observe that the first matrix represents a global phase shift and can be ignored. To recover the non-δ parameters, we divide U by its determinant. The resulting matrix $\tilde{U}$ has $\delta = 0$, and

$$\tilde{U}^t \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \tilde{U} = \begin{pmatrix} -e^{-i\beta}\sin\theta & \cos\theta \\ \cos\theta & e^{i\beta}\sin\theta \end{pmatrix} \tag{5}$$

The off-diagonal elements yield the value of θ, and the value of β can be computed subsequently. We routinely ignore global phase because it does not affect the result of quantum measurement, which is the last step in quantum algorithms. A particular one-qubit computation, the Hadamard gate H, can be implemented, up to global phase, using two elementary gates as follows:

$$H = \frac{\sqrt{2}}{2}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} = \frac{\sqrt{2}}{2}\begin{pmatrix} -i & 0 \\ 0 & -i \end{pmatrix}\begin{pmatrix} i & 0 \\ 0 & -i \end{pmatrix}\begin{pmatrix} 1 & 1 \\ -1 & 1 \end{pmatrix} \tag{6}$$

Similarly, the `NOT` gate (also known as Pauli-X) costs two elementary gates up to global phase:

$$X = \texttt{NOT} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} = \begin{pmatrix} -i & 0 \\ 0 & -i \end{pmatrix}\begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}\begin{pmatrix} i & 0 \\ 0 & -i \end{pmatrix} \tag{7}$$

2.2 Prior Work

As shown above, quantum circuits can be modelled by matrix formulas that decompose the overall computation (one large unitary matrix) into matrix products and tensor products of elementary gates (smaller unitary matrices). This suggests the use of matrix decomposition theorems from numerical analysis and Lie theory [10]. Prior work as well as our work use these decompositions: *SVD, polar, symmetric Shur (spectral), QR* [7] and *KAK* [10]. Additionally, (i) a block-2×2 version of the *SVD* called the *CS* decomposition [7, pp.77-79] was used for circuit synthesis in [16], and (ii) the *LU* decomposition [7] was used to analyze CNOT-based circuits. Most of those decompositions can be computed with existing softare `LAPACK`, downloadable from

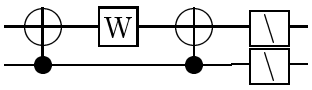

Figure 1: Any 4×4 diagonal unitary $D=\mathrm{diag}(z_1,z_2,z_3,z_4)$ can be decomposed into up to five elementary gates. We set $\mathrm{e}^{-i\phi}=z_1z_2^{-1}z_3^{-1}z_4$ and define $W=\mathrm{diag}(\mathrm{e}^{i\phi/4},\mathrm{e}^{-i\phi/4})$. The two one-qubit unitaries on the right are diagonal. Since the inverse of a diagonal matrix is also diagonal, the form of this circuit can be reversed for any given matrix.

`http://www.netlib.org`. Lie group theory [10] offers a number of more sophisticated decompositions that we plan to study in the future.

The unitary matrix of a quantum computation can be analogized with the truth table of a classical logic circuit. Logic minimization aside [6], it is trivial to come up with a classical `AND-OR-NOT` circuit implementing a given truth table. Each line of the truth table is implemented using `AND` and `NOT` gates, then all lines are connected by `OR` gates. The algorithm proposed in [4] solves a quantum version of this task.[3] While careful gate counts are not given in [4], our detailed analysis of this algorithm shows that 61 gates will be required in the generic (worst) case, and 18 of those will be `CNOT`s. Of course, it is desirable to improve gate counts.

A recent work [11] on time-optimal control of spin systems presents a holistic view of circuit-related optimizations, which is based on the Lie group theory. However, their approach is not as detailed as previously published circuit synthesis algorithms, and comparisons in terms of gate counts are not straightforward.

3 Circuit Constructions and Synthesis Algorithms

3.1 Circuits for diagonal matrices

Figure 1 shows a circuit diagram with which we can implement any diagonal unitary 4x4-matrix. We have developed an algorithm that, given such a matrix, computes required parameters of the variable gates in Figure 1.

For a diagonal matrix $D\in U(4)$, we have $D=\mathrm{diag}(z_1,z_2,z_3,z_4)$ with $z_i\overline{z_i}=1, i=1\ldots 4$. The coordinates or their product can be normalized by choosing the global phase. In contrast, the quantity $z_1z_2^{-1}z_3^{-1}z_4$ is invariant.

Proposition: i) A diagonal matrix $D=\mathrm{diag}(z_1,z_2,z_3,z_4)$ in $U(4)$ may be written as a tensor product of diagonal elements of $U(2)$ iff $z_1/z_2=z_3/z_4$. ii) Any gate which is diagonal when written in the computation basis may be implemented up to phase in five elementary gates or less.

Proof: i) The identity $\mathrm{diag}(\eta_1,\eta_2)\otimes\mathrm{diag}(\eta_3,\eta_4)=\mathrm{diag}(\eta_1\eta_3,\eta_1\eta_4,\eta_2\eta_3,\eta_2\eta_4)$ implies the forward implication. For the reverse implication, we adjust the global phase so that $z_1=1$ and assume that $\eta_1=\eta_3=1$. Thus $z_2=z_4/z_3$, $\eta_4=z_4/z_3$, $\eta_2=z_3$. The global phase can then be applied to either one-qubit gate.

ii) Consider the computation of Figure 1. For a fixed $D=\mathrm{diag}(z_1,z_2,z_3,z_4)$, put $\mathrm{e}^{-i\phi}=z_1z_2^{-1}z_3^{-1}z_4$. Now note the leftmost three gates enact

$$\begin{cases} |00\rangle\mapsto \mathrm{e}^{i\phi/4}|00\rangle \\ |01\rangle\mapsto \mathrm{e}^{-i\phi/4}|01\rangle \\ |10\rangle\mapsto \mathrm{e}^{-i\phi/4}|00\rangle \\ |11\rangle\mapsto \mathrm{e}^{i\phi/4}|00\rangle \end{cases} \tag{8}$$

[3] We note that the work in [4] to a large extent relies on results in [1].

Thus by Equation 8 and part one of the present proposition, the difference between D and the leftmost three gates is a pair of single elementary gates which are diagonal elements of $U(1)\oplus U(1)$ on each line. □

3.2 Entanglers and disentanglers

The second type of simple circuits we designed implement two specific rather than parameterized computations, which can be thought of as complex gates. These are important to our synthesis algorithms. The *entangler* gate maps the ordered computational basis $|00\rangle,|01\rangle,|10\rangle,|11\rangle$ into the "magic basis", which we introduce below. Together with its inverse — the *disentangler* — the entangler gate is useful for breaking down arbitrary two-qubit computations into elementary gates. With such uses in mind, we implement the entangler and disentangler by elementary gates.

The "magic basis" [12] provides an elegant way of thinking about tensor products of one-qubit gates.[4] The magic basis of phase shifted Bell states is given by

$$\begin{cases} |\mathsf{m1}\rangle &=& (|00\rangle+|11\rangle)/\sqrt{2} \\ |\mathsf{m2}\rangle &=& (i|00\rangle-i|11\rangle)/\sqrt{2} \\ |\mathsf{m3}\rangle &=& (i|01\rangle+i|10\rangle)/\sqrt{2} \\ |\mathsf{m4}\rangle &=& (|01\rangle-|10\rangle)/\sqrt{2} \end{cases} \tag{9}$$

Note that each state is maximally entangled. E has the following matrix:

$$E=\frac{\sqrt{2}}{2}\begin{pmatrix} 1 & i & 0 & 0 \\ 0 & 0 & i & 1 \\ 0 & 0 & i & -1 \\ 1 & -i & 0 & 0 \end{pmatrix} \tag{10}$$

One finds that E can be realized up to global phase by seven elementary gates, as shown in Figure 2. This is most easily verified by multiplying the appropriate 4×4 matrices. In particular, Equation 3 writes `topCNOT` and `botCNOT` as permutation matrices. Then

$$E=\mathtt{botCNOT}\circ\mathtt{topCNOT}\circ\frac{\sqrt{2}}{2}\begin{pmatrix} 1 & 1 & 0 & 0 \\ 1 & -1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & -1 \end{pmatrix}\circ$$

$$\mathtt{botCNOT}\circ\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & i & 0 \\ 0 & 0 & 0 & i \end{pmatrix}\circ\mathtt{botCNOT} \tag{11}$$

Note that the circuit diagram in Figure 2 travels right to left, so gate matrices are multiplied in reverse. $S=\mathrm{diag}(1,i)$ is an elementary gate up to global phase $\mathrm{e}^{-i\pi/4}$, and the Hadamard gate H can be implemented, up to global phase, with two elementary gates as shown in Equation 2.1.

In summary, E requires four CNOT gates and three one-qubit rotations. Similarly, E^* may be implemented in seven elementary gates by writing the inverse of each gate of figure 2 in reverse order.

3.3 Arbitrary two-qubit computations

The main result of our work is an algorithm that synthesizes a circuit for an arbitrary 2-qubit computation. On average, our algorithm produces much better results than the algorithm from [4].

[4] Stated in terms of the Lie algebra of $U(4)$, this involves the isomorphism $\mathfrak{u}(2)\oplus\mathfrak{u}(2)\cong\mathfrak{o}(4)$ [10, p. 370].

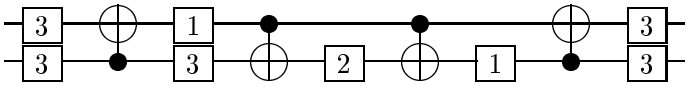

Figure 3: The decomposition of a generic 2-qubit quantum computation into up to 23 gates. Four generic one-qubit rotations are marked with "3" as they require up to three elementary gates. Computations requiring two or one elementary gates are also labelled.

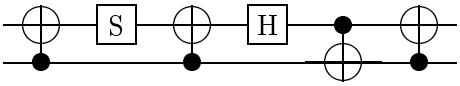

Figure 2: Implementing E by elementary gates. Here $S = \mathrm{diag}(1, i)$ counts as one elementary gate and the Hadamard gate H counts as two.

First, we decompose an arbitrary 4x4-unitary into $U = (U_1 \otimes U_2) \circ \texttt{botCNOT} \circ \texttt{topC}{-}U_3 \circ (\mathbf{1} \otimes U_4) \circ \texttt{botCNOT} \circ (U_5 \otimes U_6)$ where $U_1, \ldots, U_6$ are one-qubit gates. The algorithm relies on the *KAK* matrix decomposition and also involves the spectral and the polar matrix decompositions. Algebraic details are available online at `http://xxx.lanl.gov/quant-ph/abs/0211002` and are not reproduced here due to space limitations.

The gate count yields:

- three elementary rotations for each of five one-qubit gates U_1, U_2, U_3, U_5 and U_6,
- two `botCNOT` gates,
- eight elementary gates to implement the `topC`$-U_4$ gate, according to [4, Figure 7].

The total gate count of 25 can be further reduced, given the structure of the `topC`$-V$ circuit proposed in [4]. Indeed, that circuit can be written symbolically as `topC`$-U_3 = (\mathbf{1} \otimes C) \circ \texttt{topCNOT} \circ (\mathbf{1} \otimes B) \circ \texttt{topCNOT} \circ (D \otimes A)$. C and D are elementary gates up to phase, but A and B require up to two elementary gates [4]. We outline the structure of circuits produced and provide gate counts.

Since `topC`$-U_3$ is next to $(\mathbf{1} \otimes U_4)$ our structural decomposition, we can reduce $(D \otimes A) \circ (\mathbf{1} \otimes U_4)$ to $(D \otimes U_7)$ where $U_7 = AU_4$. By merging the computation A with the generic one-qubit computation U_4 that may require up to three elementary gates, one reduces the overall circuit by two elementary gates.

The overall circuit decomposition can be described algebraically as follows:

$$U = (U_1 \otimes U_2) \circ \texttt{botCNOT} \circ (D \otimes U_7) \circ \texttt{topCNOT} \circ (\mathbf{1} \otimes B) \circ \texttt{topCNOT} \circ (\mathbf{1} \otimes C) \circ \texttt{botCNOT} \circ (U_5 \otimes U_6) \qquad (12)$$

It is illustrated in Figure 3, where gate counts are shown.

Our circuit decomposition requires at most four CNOTs, while other gates are elementary one-qubit rotations. Such a small number of non-one-qubit gates may be desired in practical implementations where multi-qubit interactions are more difficult to implement.

It is understood that Figure 3 and our gate counts refer to the worst case. Specific computations may require only some of those gates. We have considered a number of examples and found that our algorithm produces reasonably-sized circuits, even compared to best known circuits from the literature. In those examples, our algorithm is able to capture the structure of the given quantum computation. Unlike previously known circuit synthesis algorithms, ours can always implement $A \otimes B$ without using `CNOT` gates.

For example, our algorithm synthesized a circuit for 2-qubit Quantum Fourier Transform (QFT) that consists of 14 elementary gates of which 4 are `CNOT`s (details in `http://xxx.lanl.gov/quant-ph/abs/0211002`). The standard circuit for QFT has 12 gates, but 5 of them are `CNOT`s. Thus, the standard circuit is less desirable in quantum implementation technologies where `CNOT` gates are more costly than one-qubit rotations.

Our algorithm can be viewed as a constructive proof that any two-qubit quantum computation can be implemented in 23 elementary gates (or fewer), of which at most 4 are `CNOT`s and remaining gates are one-qubit rotations. We also provide a non-constructive proof that there exists a two-qubit computation such that any circuit implementing it in terms of elementary gates consists of at least 17 gates. In particular, 15 one-qubit rotations are required and two `CNOT`s, and a geometric dimensionality argument imply that almost every quantum computation will require at least 17 elementary gates.

Trying to close the gap between the upper and lower bounds, we observe that the 19 non-constant one-qubit rotations in Figure 3 seem redundant as only 15 rotations are required for reasons of dimension. To this end, we offer another generic gate decomposition for arbitrary 2-qubit computations that entails no more than 15 non-constant one-qubit rotations, at the price of some constant rotations and significantly more `CNOT` gates than used by our main decomposition in Figure 3. It also stems from the structural decomposition of 4x4 unitary matrices $U = (U_1 \otimes U_2) \circ (EDE^*) \circ (U_3 \otimes U_4)$ where $U_1, \ldots, U_4$ are one-qubit gates and D is a diagonal unitary. In this context, we use circuit decompositions for E, E^* and D given earlier. The matrix D is controlled by 3 real parameters (4 diagonal unitaries modulo global phase). It is implemented in Figure 1 using 3 one-qubit rotations and 2 `CNOT`s. The entangler E and disentangler E^* are fixed matrices and require no parameters. The implementation of E in Figure 2 requires 3 constant rotations and 4 `CNOT`s.

Adding the gate counts, we see that $U_1, \ldots, U_4$ may require up to 12 elementary gates altogether. D counts for 5, while E and E^* count for 7 each, for a total of 31. However, upon inspection of the Figures 1 and 2, one notes that the circuit EDE^* has two canceling `botCNOT` gates. Moreover, since the inverse of D is also a diagonal unitary matrix, we can flip the asymmetric circuit for D in Figure 1. This allows us to merge a constant rotations from E with a variable rotation from D. The resulting circuit decomposition is shown in Figure 4 and requires up to 28 elementary gates, of which 15 are variable one-qubit rotations, 5 are constant rotations and 8 are `CNOT`s. The slight asymmetry in Figure 4 is explained by the asymmetric circuit for D in Figure 1.

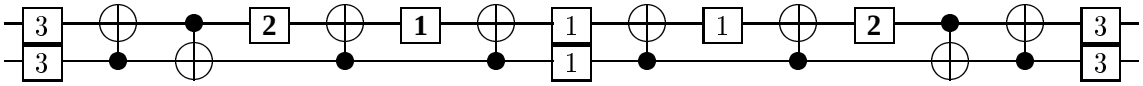

Figure 4: The overall structure entailed by our circuit decomposition #2. Four generic one-qubit rotations are marked with "3" because they are worth up to three elementary gates. Two Hadamard gates are marked with "2" because they are worth two elementary gates. Constant gates are in bold.

4 Empirical Validation

We implemented our twenty-three gate synthesis algorithm in C++ without the use of numerical or matrix libraries, and tested it on various quantum computations. Synthesis runtimes are negligibly small on a PC and are not reported. Algebraic primitives used by our implementation include finding all eigenvalues and real eigenvectors of a given matrix, computing matrix square-roots and exponentials, etc.

The first experiment tests the synthesis algorithm on arbitrary twenty-three qubit matrices, which were generated as follows. A random number generator produces an arbitrary 4×4 complex Hermitian matrix H so that computing $\exp(iH)$ produces a unitary matrix which may be interpreted as a quantum computation. For simplicity, (1) the entries of H are chosen to have real and complex part no greater than five, and (2) we exponentiate iH by adding 70 terms of the Taylor series. After applying our synthesis algorithm to such unitary matrices, we verify the result by multiplying out the gates and comparing the result to the original matrix. The purpose of this experiment is to check that generic circuits could be correctly synthesized with no more than twenty-three gates. This is indeed the case.

5 Conclusions and Ongoing Work

The main result of our work is a technique for implementing an arbitrary two-qubit computation in 23 gates or less. We observe that it always produces an even number of CNOTs. Thus, an interesting problem is to find non-trivial quantum computations such that the smallest possible number of CNOTs in their implementations is odd. The line-swap is a likely candidate as it can be implemented in three CNOTs and requires at least three elementary gates.

Our on-going work indicates that 18 elementary gates are actually required for worst-case two-qubit computations, and yet 18 gates are sufficient. Of those at most 3 are CNOTs. In particular, we found smaller entangler circuits. More details will be posted in a pre-print at http://xxx.lanl.org/find/quant-ph

The reported constructive methods for quantum circuit synthesis can be analogized with library-less synthesis and technology mapping in classical logic-synthesis. Matrix decompositions represent library-less synthesis and are applicable with a variety of gate libraries. In the quantum technology mapping step, one synthesizes matrices of the various matrix factors in a given gate library.

Generalizing our techniques to N-qubit computations is a significant challenge. This seems to require more sophisticated techniques from Lie theory and more subtle algorithmic optimizations. From the physics perspective, a better understanding of entanglement is needed. Having formulated these grand challenges, we already made the first step of synthesizing asymptotically optimal circuits for arbitrary N-qubit diagonal computations [3].

Possible generalizations of our work to three qubits echo recent work in physics. In particular, there have been attempts to describe $SU(8)/\otimes_1^3 SU(2)$ geometrically. The studies of "maximal" entanglement on three qubits represented by the so-called $|\mathrm{GHZ}\rangle$ states is also relatively new [13]. More generally, the concept of "entanglement type" was proposed and seems relevant to our work [5].

References

[1] A. Barenco et al., "Elementary Gates For Quantum Computation," *Physical Rev. A* (52), 1995, 3457-3467.

[2] M. J. Bremner et al., "A Practical Scheme For Quantum Computation With Any Two-qubit Entangling Gate," 2002. http://xxx.lanl.gov/abs/quant-ph/0207072

[3] S. S. Bullock and I. L. Markov, "Smaller Circuits for Arbitrary n-qubit Diagonal Computations," http://xxx.lanl.gov/abs/quant-ph/0303039

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

[5] W. Dür, G. Vidal and J.I. Cirac, "Three qubits can be entangled in two inequivalent ways," *Physical Review A*, vol 62, 062314, 2000.

[6] G. Hachtel, F. Somenzi, *Synthesis and Verification of Logic Circuits*, 3rd ed., Kluwer 2000.

[7] G. H. Golub, C. F. Van Loan, *Matrix Computations*, Johns Hopkins Press, Baltimore, 1996.

[8] C. Dawson and A. Gilchrist, "GQC: A Quantum Compiler", 2002. http://www.physics.uq.edu.au/gqc/.

[9] G. Song and A. Klappenecker, "Optimal Realizations of Controlled Unitary Gates," 2003, http://xxx.lanl.gov/abs/quant-ph/0301078

[10] A. W. Knapp, *Lie Groups Beyond an Introduction*, Progress in Mathematics, vol. 140, Birkhäuser, 1996.

[11] N. Khaneja, R. Brockett and S. J. Glaser, "Time Optimal Control In Spin Systems," 2001 http://xxx.lanl.gov/abs/quant-ph/0006114.

[12] M. Lewenstein, B. Kraus, P. Horodecki and I. Cirac, "Characterization of Separable States and Entanglement Witnesses," *Physical Review A (3)*, vol. 63, no. 4, 2001, pp. 044304-7, http://xxx.lanl.gov/abs/quant-ph/0011050.

[13] R. B. Lockhart, M. J. Steiner and K. Gerlach, "Geometry and Product States", *Quantum Information and Computation*, vol. 2 no. 5, Sept. 2002, pp. 333-347.

[14] M. Nielsen and I. Chuang, *Quantum Computation and Quantum Information*, Cambridge Univ. Press, 2000.

[15] K. N. Patel, I. L. Markov, J. P. Hayes, "Efficient Synthesis of Linear Reversible Circuits", 2003. http://xxx.lanl.gov/abs/quant-ph/0302002

[16] R. Tucci, "A Rudimentary Quantum Compiler", 1999, http://xxx.lanl.gov/abs/quant-ph/9902062

[17] V. V. Shende, A. K. Prasad, I. L. Markov and J. P. Hayes, "Reversible Logic Synthesis", to appear in *IEEE Trans. on Computer-Aided Design*, 2002, http://xxx.lanl.gov/abs/quant-ph/0207001

[18] G. F. Viamontes, M. Rajagopalan, I. L. Markov and J. P. Hayes, "Gate-level Simulation of Quantum Circuits", *Proc. Asia and South-Pacific Design Automation Conf.*, Kitakyushu, Japan, January 2003. http://xxx.lanl.gov/abs/quant-ph/0208003