Quantum Information and Computation, Vol. 6, No. 4&5 (2006) 436–454
© Rinton Press

EFFICIENT CIRCUITS FOR EXACT-UNIVERSAL COMPUTATION WITH QUDITS

G.K. BRENNEN[a]
Atomic Physics Division, National Institute of Standards and Technology
Gaithersburg, Maryland, 20899-8420

S.S. BULLOCK
Center for Computing Sciences, Institute for Defense Analyses
Bowie, MD 20715-4300

D.P. O'LEARY
Department of Computer Science and Institute for Advanced Computer Studies, University of Maryland
College Park, Maryland, 20742
Mathematical and Computational Sciences Division, National Institute of Standards and Technology
Gaithersburg, MD 20899-8910

Received October 4, 2005
Revised March 20, 2006

This paper concerns the efficient implementation of quantum circuits for qudits. We show that controlled two-qudit gates can be implemented without ancillas and prove that the gate library containing arbitrary local unitaries and one two-qudit gate, CINC, is exact-universal. A recent paper [S.Bullock, D.O'Leary, and G.K. Brennen, Phys. Rev. Lett. **94**, 230502 (2005)] describes quantum circuits for qudits which require $O(d^n)$ two-qudit gates for state synthesis and $O(d^{2n})$ two-qudit gates for unitary synthesis, matching the respective lower bound complexities. In this work, we present the state-synthesis circuit in much greater detail and prove that it is correct. Also, the $\lceil (n-2)/(d-2) \rceil$ ancillas required in the original algorithm may be removed without changing the asymptotics. Further, we present a new algorithm for unitary synthesis, inspired by the *QR* matrix decomposition, which is also asymptotically optimal.

Keywords:

Communicated by: B Kane & M Mosca

1 Introduction

A qudit is a d-level generalization of a qubit, i.e. the one-qudit Hilbert space splits orthogonally as

$$\mathcal{H}(1,d) \;=\; \mathbb{C}\{|0\rangle\} \oplus \mathbb{C}\{|1\rangle\} \oplus \cdots \oplus \mathbb{C}\{|d-1\rangle\} \tag{1}$$

while the n-qudit state-space is $\mathcal{H}(n,d) = [\mathcal{H}(1,d)]^{\otimes n}$. Thus for $N = d^n$, closed-system evolutions of n qudits are modeled by $N \times N$ unitary matrices. Qudit circuit diagrams then factor such unitaries into two-qudit operations $I_{d^{n-2}} \otimes V$ where V is a $d^2 \times d^2$ unitary matrix, or more generally into similarity transforms of such gates by particle-swaps. The algorithmic complexity of an evolution may then be

[a] Present address: Institute for Quantum Optics and Quantum Information of the Austrian Academy of Sciences, A-6020, Innsbruck, Austria.

thought of as the number of two-qudit gates required to build it. A degree of freedom argument [1] leads one to guess that exponentially many gates are required for most unitary evolutions, since the space of all $N \times N$ unitary matrices is d^{2n}-dimensional. Indeed, this space of evolutions is a manifold so the argument may be made rigorous using smooth topology, and thus $\Omega(d^{2n})$ gates are required for exact-universality. Yet until quite recently the best qudit circuits contained $O(n^2 d^{2n})$ gates [2]. In contrast, $O(4^n)$ gates were known to suffice for qubits ($d = 2$) [3], presenting the possibility that qudits are genuinely less efficient for d not a power of two.

Recently, an explicit $O(d^{2n})$ construction was achieved [4]. It uses the spectral decomposition of the unitary matrix desired and also a new *state-synthesis circuit* [1, 5, 6, 9]. Given a $|\psi\rangle \in \mathcal{H}(n,d)$, a state-synthesis circuit for $|\psi\rangle$ realizes some unitary U such that $U|\psi\rangle = |0\rangle^{\otimes n}$. We also sometimes use the term to refer to the inverse problem of constructing $W = U^\dagger$ with $|\psi\rangle = W|0\rangle^{\otimes n}$. There are $2d^n - 2$ real degrees of freedom in a normalized state ket $|\psi\rangle$, which may be used to prove that circuits for generic states cost $\Omega(d^n)$ two-qudit gates. This is in sharp contrast to the case of classical logic, where $O(n)$ inverters may produce any bit-string. The most recent qudit state-synthesis circuit [4] contains $(d^n - 1)/(d - 1)$ two-qudit gates, and in fact each is a singly-controlled one-qudit operator $\wedge_1(V) = I_{d^2-d} \oplus V$.

There are two ways to employ an asymptotically optimal state-synthesis circuit in order to obtain asymptotically optimal unitary circuits. The first is to exploit the spectral decomposition, which involves a three part circuit for each eigenstate of the unitary: building an eigenstate [1, 4], applying a conditional phase to one logical basis ket, and unbuilding the eigenstate. We here introduce a second option, the **Triangle** algorithm, which uses the state-synthesis circuit with extra controls to reduce the unitary to upper triangular form. Recursive counts of the number of control boxes show that it is also asymptotically optimal (Cf. [3].) Although these algorithms are unlikely to be used to implement general unitary matrices, they can be usefully applied to improving subblocks of larger circuits (peephole optimization).

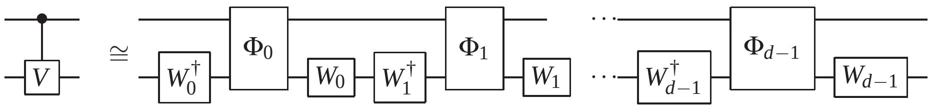
Finally, this work also addresses two further topics in which qudit circuits lag behind qubit circuits. First, to date the smallest gate library for exact universality with qudits uses arbitrary locals complemented by a continuous one parameter two-qudit gate [10]. In contrast, it is well known [5] that any computation on qubits can be realized using gates from the library $U(2)^{\otimes n} \sqcup \{\texttt{CNOT}\}$, where the symbol $\sqcup$ denotes the disjoint union. We prove that the library $U(d)^{\otimes n} \sqcup \{\texttt{CINC}\}$ is exactly universal, where CINC controlled-increment) is the qudit generalization of the CNOT gate. Second, the first asymptotically optimal qubit quantum circuit exploited a single ancilla qubit [3] and current constructions require none [9, 6], while qudit diagrams tend to suppose $\lceil (n-2)/(d-2) \rceil$ ancilla qudits. Here we present methods which realize a k-controlled operation $\wedge_k(V) = I_{d^{k+1}-d} \oplus V$ in $O[(k+2)^{2+\log_2 d}]$ gates without the need for any ancilla. This makes all qudit asymptotics competitive with their qubit counterparts. We note that the state of the art circuit design for exact-universal computation with qubits use a variant of the the Cosine-Sine Decomposition (CSD) [6, 7]. For $d = 2$, the CSD circuits attain a generic unitary with number of CNOT gates that is less than a factor of two over the lower bound of $O(4^n/4)$. Quite recently, a CSD circuit for d-level systems has been developed [8]; however, the gate library used in this circuit uses certain multi-qudit circuit blocks as primitives. The cost of these blocks in terms of two-qudit gates is unclear.

The paper is organized as follows. §2 improves on earlier constructions of $\wedge_1(V)$ gates, which are ubiquitous in later sections. §3 presents a new circuit for a qudit $\wedge_k(V)$ gate which is later used to produce the first $O(d^{2n})$ gate unitary circuits without ancilla. §4 details the recent state-synthesis algorithm as an iteration over a new ♣-sequence and exploits the new constructions to prove it is correct. §5 presents a new asymptotically optimal unitary circuit inspired by the QR matrix factorization and compares it with a previous algorithm based on spectral decomposition. §6 discusses two applications of the state synthesis algorithm.

2 Optimizing singly-controlled one-qudit unitaries

Several operators $\wedge_1(V)$ appear in later circuits. Thus, it is worthwhile to optimize this computation in our gate libraries. For qubits, CNOT-optimal circuits for $\wedge_1(V)$ are known [11]. The qudit case is open. Here we improve the $\wedge_1(V)$ circuit in that work. Given that any $d^n \times d^n$ unitary may be constructed using $\wedge_1(V)$ [2], a corollary is that $U(d)^{\otimes n} \sqcup \{\mathtt{CINC}, \mathtt{CINC}^{-1}\}$ is exact-universal. Since $\mathtt{CINC}^{-1} = \mathtt{CINC}^{d-1}$, this also demonstrates that $U(d)^{\otimes n} \sqcup \{\mathtt{CINC}\}$ is exact-universal. This is a smaller universal library than that presented in earlier work [10].

Thus, consider the question of factoring $\wedge_1(V)$. Let $\{|\psi_k\rangle\}_{k=0}^{d-1}$ be the eigenvectors of V with eigenvalues $\{e^{i\theta_k}\}_{k=0}^{d-1}$. Let W_k be some one-qudit unitary with $W_k|0\rangle = |\psi_k\rangle$, e.g. the appropriate one-qudit Householder reflection (See §4.1.) Finally, let Φ_k be a controlled one-qudit phase unitary given by $\Phi_k = \wedge_1[I_d + (e^{i\theta_k} - 1)|0\rangle\langle 0|]$. Then note that $V = \prod_{k=0}^{d-1} W_k[I_d + (e^{i\theta_k} - 1)|0\rangle\langle 0|]W_k^\dagger$. Thus $\wedge_1(V)$ can be implemented by the following circuit:



(2)

(In the above diagram, we denote control on state $|d-1\rangle$ by a solid dot.) Thus, we have reduced the question to building Φ_k in terms of $U(d)^{\otimes 2}$ and $\mathtt{CINC}$.

Building Φ_k requires some preliminary remarks. Suppose we have $\xi \in \mathbb{C}$, $|\xi| = 1$. Consider the diagonal unitary of the corresponding geometric sequence: $D = \sum_{j=0}^{d-1} \xi^j |j\rangle\langle j|$. Recall that $\mathtt{INC}$ is the increment permutation, i.e. $\mathtt{INC}|j\rangle = |(j+1) \bmod d\rangle$. Thus permuting the diagonal entries, $\mathtt{INC}\, D\, \mathtt{INC}^{-1} = \xi^{d-1}|0\rangle\langle 0| + \sum_{j=1}^{d-1} \xi^{j-1}|j\rangle\langle j|$. Hence

$$\mathtt{INC}\, D\, \mathtt{INC}^{-1}\, D^{-1} = \xi^{d-1}|0\rangle\langle 0| + \xi^{-1}\sum_{j=1}^{d-1}|j\rangle\langle j| = (\xi^{-1} I_d)\big(\xi^d |0\rangle\langle 0| + \sum_{j=1}^{d-1}|j\rangle\langle j|\big). \tag{3}$$

We next generalize a standard trick from $d = 2$[12, Lemma 5.2] to arbitrary d. Note that

$$\wedge_1(\xi I_d) = \big(\sum_{j=0}^{d-2}|j\rangle\langle j| + \xi|d-1\rangle\langle d-1|\big) \otimes I_d, \tag{4}$$

so that a controlled uniform-phase is in fact a local operation. Hence taking $\xi = e^{i\theta_k/d}$, we obtain in particular an expression for Φ_k of Equation 2 in terms of $\mathtt{CINC}$ and $\mathtt{CINC}^{-1}$:

$$\begin{aligned} \Phi_k &= \wedge_1(\xi I_d)\, \mathtt{CINC}\, (I_d \otimes D)\, \mathtt{CINC}^{-1}\, (I_d \otimes D^{-1}) \\ &= \big[\big(\textstyle\sum_{j=0}^{d-2}|j\rangle\langle j| + \xi|d-1\rangle\langle d-1|\big) \otimes I_d\big]\, \mathtt{CINC}\, (I_d \otimes D)\, \mathtt{CINC}^{-1}\, (I_d \otimes D^{-1}). \end{aligned} \tag{5}$$

Hence, $\wedge_1(V)$ may be realized using gates from $U(d)^{\otimes 2}$ along with d copies of $\mathtt{CINC}$ and d copies of $\mathtt{CINC}^{-1}$.

Recall that these circuits may be expanded into circuits in terms of $\wedge_1(\sigma_x \oplus I_{d-2})$. Indeed, when viewed as permutations, $\mathtt{INC}$ and $\mathtt{INC}^{-1}$ factor into d flips. To see this, consider $0 \leq j < k \leq d-1$ and let (jk) denote the flip permutation $j \leftrightarrow k$ of $\{0, 1, \dots, d-1\}$. Then

$$\mathtt{INC} \; = \; (01) \circ (12) \circ \cdots \circ (d-2\; d-1). \tag{6}$$

Since $\wedge_1[(jk)]$ is equivalent to $\wedge_1(\sigma_x \oplus I_{d-2})$ up to permutations within $U(d)^{\otimes n}$, we see that $\mathtt{CINC}$ and $\mathtt{CINC}^{-1}$ may be implemented using $d-1$ copies of the controlled-flip. Thus, $\wedge_1(V)$ may also be realized using $2d(d-1)$ copies of the $\wedge_1(\sigma_x \oplus I_{d-2})$ gate.

Remark: Note that the controlled-flip is also equivalent to $\wedge_1(I_{d-2} \oplus \sigma_z)$, making blockwise use of the 2×2 matrix identity $H\sigma_x H = \sigma_z$ for $H = \frac{1}{\sqrt{2}} \sum_{j,k=0}^{1} (-1)^{jk} |k\rangle \langle j|$. Thus, the above also realizes $\wedge_1(V)$ in roughly $2d^2$ controlled-π phase gates. This is half the roughly $4d^2$ gates of earlier work [10], even after including the arbitrary relative phase $e^{i\theta}$ allowed there.

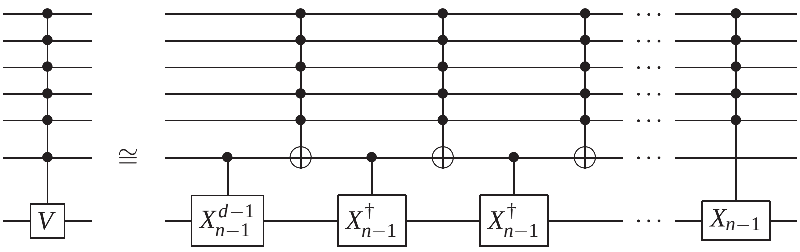
3 Qudit control without ancillas

In this section we simulate a $\wedge_{n-1}(V)$ gate for $V \in U(d)$ using $O[(n+1)^{\log_2 d+2}]$ singly-controlled one qudit gates without ancilla. The method parallels the techniques used in Ref. [12] for universal computation with qubits.

First we decompose a $\wedge_{n-1}(V)$ gate using a sequence of gates with a smaller number of controls. As a first step, notice that

$$\wedge_{n-1}(V) \; = \; \wedge_{n-2}(X_{n-1})[\wedge_{n-2}(\mathtt{INC})\wedge_1(X_{n-1}^{\dagger})]^{d-1}\wedge_{n-2}(\mathtt{INC})\wedge_1(X_{n-1}^{d-1}), \tag{7}$$

where $X_{n-1} = V^{1/d}$. For example, for $n = 7$, we have the following circuit:

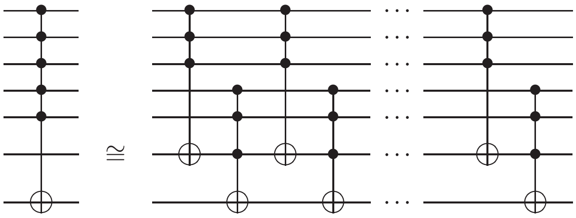


(8)

All control operations are conditioned on the control qudits being in state $|d-1\rangle$. The circuit is designed to cycle over each possible dit value of the control qudit in the $\wedge_1(X_{n-1}^{\dagger})$ gates. The entire construction then follows by recursive application of Equation 7 to the last gate. In theory, this construction is an exact implementation of $\wedge_{n-1}(V)$. Yet in practice, the sequence of matrices X_j obtained by taking the d-th root of X_{j+1} (with $X_n = V$) quickly converges to the identity matrix as j decreases. Hence, an approximate implementation results if the recursion is terminated early.

As an example of Equation 8, consider the generalized Toffoli gate $\wedge_2(\mathtt{INC})$. This breaks into $(d+1)$ variants of singly-controlled $\wedge_1(W)$ gates along with d extra $\mathtt{CINC}$ gates. Hence $(d+1)d+d$ $\mathtt{CINC}$ gates along with $(d+1)d$ $\mathtt{CINC}^{-1}$ gates and sundry gates from $U(d)^{\otimes n}$ suffice to emulate $\wedge_2(\mathtt{INC})$.

Note that the size of the circuit for $\wedge_{n-2}(\mathtt{INC})$ that is analogous to the above grows exponentially in n. However, it is possible to simulate $\wedge_{n-2}(\mathtt{INC})$ more efficiently using a sequence of $\wedge_{\lceil (n-1)/2 \rceil}(\mathtt{INC})$ and $\wedge_{\lfloor (n-1)/2 \rfloor}(\mathtt{INC})$ gates, proceeding recursively down to $\wedge_2(\mathtt{INC})$. The argument is analogous to that used for qubits in Lemma 7.3 in Ref. [12] for $n \geq 5$. The following circuit illustrates the method for $n = 7$:



(9)

Ignoring which qudits are controlled or targeted, the circuit sequence is $\wedge_{n-2}(\texttt{INC}) = [\wedge_{\lfloor (n-1)/2 \rfloor}(\texttt{INC})\, \wedge_{\lceil (n-1)/2 \rceil}(\texttt{INC})]^d$.

For the remainder of this section, we use a tilde to distinguish a count for $\texttt{CINC}^{-1}$ from a $\texttt{CINC}$ count. Thus, we let b_{n-2} be the total number of $\texttt{CINC}$ gates required to emulate $\wedge_{n-2}(\texttt{INC})$, and $\tilde{b}_{n-2}$ be the similar count for $\texttt{CINC}^{-1}$. For Circuit 9,

$$\begin{array}{rcl} b_{n-2} & = & d(b_{\lceil (n-1)/2 \rceil} + b_{\lfloor (n-1)/2 \rfloor}), \\ \tilde{b}_{n-2} & = & d(\tilde{b}_{\lceil (n-1)/2 \rceil} + \tilde{b}_{\lfloor (n-1)/2 \rfloor}). \end{array} \tag{10}$$

A quick induction shows that each sequence is increasing, and thus $b_{n-2} \leq 2db_{\lceil (n-1)/2 \rceil}$ and $\tilde{b}_{n-2} \leq 2d\tilde{b}_{\lceil (n-1)/2 \rceil}$. Moreover, by the analysis of $\wedge_2(\texttt{INC})$ above $b_2 = d^2 + 2d$ and $\tilde{b}_2 = d^2 + d$. Recalling $(\log_d n)(\log_2 d) = \log_2 n$, we obtain the following:

$$\begin{array}{rcccl} b_{n-2} & \leq & (d^2+2d)(2d)(2d)^{\log_2 n} & = & (d^2+2d)(2d)n^{1+\log_2 d}, \\ \tilde{b}_{n-2} & \leq & (d^2+d)(2d)(2d)^{\log_2 n} & = & (d^2+d)(2d)n^{1+\log_2 d}. \end{array} \tag{11}$$

Note that these counts assume that the emulation of $\wedge_{n-2}(\texttt{INC})$ is done on a system with n qudits. Combining this circuit with Circuit 8 allows for an ancilla-free implementation of $\wedge_{n-1}(V)$.

Thus, let c_{n-1} be the number of $\texttt{CINC}$ gates required to emulate $\wedge_{n-1}(V)$, not counting an additional $\tilde{c}_{n-1}$ $\texttt{CINC}^{-1}$ gates. Using Circuit 8,

$$\begin{array}{rcl} c_{n-1} & = & db_{n-2} + c_{n-2} + d^2, \\ \tilde{c}_{n-1} & = & d\tilde{b}_{n-2} + \tilde{c}_{n-2} + d^2. \end{array} \tag{12}$$

We may then overestimate c_{n-1} and $\tilde{c}_{n-1}$ using integral comparison and $c_2 = d^2 + 2d$, $\tilde{c}_2 = d^2 + d$, obtaining

$$\begin{array}{rcl} c_{n-1} & = & d\left(\sum_{j=2}^{n-2} b_j\right) + c_2 + (n-3)d^2 \\ & \leq & d[(d^2+2d)(2d)] \int_4^{n+1} t^{1+\log_2 d}\, dt + 2d + (n-2)d^2 \\ & = & \frac{(2d^2)(d^2+2d)}{2+\log_2 d}\left[(n+1)^{2+\log_2 d} - 4d^2\right] + (n-2)d^2 + 2d. \end{array} \tag{13}$$

We may similarly overestimate $\tilde{c}_{n-1}$:

$$\tilde{c}_{n-1} \leq \frac{(2d^2)(d^2+d)}{2+\log_2 d}\left[(n+1)^{2+\log_2 d} - 4d^2\right] + (n-2)d^2 + d. \tag{14}$$

Hence c_{n-1}, $\tilde{c}_{n-1}$ are both bounded by $O[(n+1)^{2+\log_2 d}]$. This can be used to show that the earlier spectral algorithm [4] is asymptotically optimal even when ancilla qudits are absent.

If we disallow $\mathtt{CINC}^{-1}$ and rather emulate $\mathtt{CINC}^{-1} = \mathtt{CINC}^{d-1}$, then the overall CINC count for $\wedge_{n-1}(V)$ would be $c_{n-1} + (d-1)\tilde{c}_{n-1}$. Note that if the gate library contains the two qudit gate $\wedge_1(\sigma_x \oplus I_{d-2})$ rather than CINC, a naïve application of the above argument would imply a linear overhead with a factor of $d-1$. However Circuits 8 and 9 can be adapted by replacing the $\wedge_k(\mathtt{INC})$ gates with gates locally equivalent to $\wedge_1(\sigma_x \oplus I_{d-2})$, resulting in a smaller overhead.

4 Asymptotically optimal qudit state-synthesis

State-synthesis is an important problem in quantum circuit design [5, 1]. This section expands upon the earlier account [4] of an asymptotically optimal state-synthesis circuit for qudits. The earlier circuit used only $O(d^n)$ two-qudit gates, while a dimension-based argument [4] shows that no fewer number of gates may achieve qudit state-synthesis.

Our description of the algorithm relies upon two tools: a sequence that we call the ♣-sequence (§4.3), and a Householder reflection matrix. The ♣-sequence's utility is two-fold:

- The ♣-sequence specifies the order in which state amplitudes are zeroed while (de)constructing the target state. It substitutes for the Gray code ordering [3] used for the case $d = 2$.
- Using the ♣-sequence, we prove that the state-synthesis algorithm functions as asserted. The ♣-sequence simplifies the careful accounting of which amplitudes have and have not been zeroed after applying each $\wedge_1(V)$ gate.

The zeroing of $d-1$ amplitudes at a time is accomplished by a Householder matrix, which we define in the next subsection.

4.1 One-qudit Householder reflection matrices

Earlier universal $d = 2$ circuits [12] relied on a QR factorization to write any unitary U as a product of *Givens rotations*, realized in the circuit as k-controlled unitaries [13]. In the multi-level case, we instead use *Householder reflection matrices* [14, §5.1]. Thus, suppose $|\psi\rangle \in \mathcal{H}(1,d)$, perhaps not normalized. Householder matrices solve the one-qudit case of the inverse state-synthesis problem. Given that $|\psi\rangle$ might not be normalized, the appropriate formulas are:

$$\left\{ \begin{array}{lcl} |\eta\rangle & = & |\psi\rangle - \sqrt{\langle\psi|\psi\rangle}\,\frac{\langle 0|\psi\rangle}{|\langle 0|\psi\rangle|}\,|0\rangle\,, \\ W & = & I_d - (2/\langle\eta|\eta\rangle)\; |\eta\rangle\,\langle\eta|\,. \end{array} \right. \tag{15}$$

Then $W|\psi\rangle$ is a multiple of $|0\rangle$. Geometrically, W is that unitary matrix which reflects across a plane lying between $|0\rangle$ and $|\psi\rangle$. For any given vector, then, a Householder matrix can be constructed for which the matrix-vector product has zeros except in a single position.

In general, the quantum circuits for $d^n \times d^n$ Householder matrices are not simple. However, if V is a $d \times d$ Householder matrix then $\wedge_1(V)$ is a Householder matrix. The n-qudit state-synthesis circuit is built from such gates.

4.2 The ♣-sequence

The ♣-sequence is a sequence of words of length n in the letters $\{0,1,2,\ldots,d-1\} \sqcup \{♣\}$; see Table 1. To define it, we might associate ♣ to an artificial dit value d and list all words in $\{0,1,2,\ldots,d\}^n$ in lexicographic order. Deleting those words in which a d character occurs before a lesser character produces the sequence.

The ♣-sequence may be recursively generated by Algorithm 1. In it, we prepend all possible dits to the $(n-1)$-long dit sequence to generate all but one term of the n-dit sequence, and then end with the term $\clubsuit^n$.

To each word in the ♣-sequence we associate an n-qudit controlled rotation, where the leftmost ♣ defines the target. For example, the Householder matrix corresponding to 000♣ puts zeros in positions 000ℓ, while that corresponding to 12♣♣ zeros $12\ell0$, and the Householder matrix for 2♣♣♣ zeros $2\ell00$ ($\ell = 1, \ldots, d-1$).

It is easy to convince oneself that applying the ♣-sequence of Householder matrices zeros all elements except $0\ldots0$. It is a bit more difficult to show how to avoid ruining zeros that have already been achieved, and we concentrate in the next subsections on showing that a small set of controls is sufficient to achieve this.

n	♣-sequence, $d = 3$
1	♣
2	0♣, 1♣, 2♣, ♣♣
3	00♣, 01♣, 02♣, 0♣♣, 10♣, 11♣, 12♣, 1♣♣, 20♣, 21♣, 22♣, 2♣♣, ♣♣♣
4	000♣, 001♣, 002♣, 00♣♣, 010♣, 011♣, 012♣, 01♣♣, 020♣, 021♣, 022♣, 02♣♣, 0♣♣♣ 100♣, 101♣, 102♣, 10♣♣, 110♣, 111♣, 112♣, 11♣♣, 120♣, 121♣, 122♣, 12♣♣, 1♣♣♣ 200♣, 201♣, 202♣, 20♣♣, 210♣, 211♣, 212♣, 21♣♣, 220♣, 221♣, 222♣, 22♣♣, 2♣♣♣, ♣♣♣♣

Fig. 1. Sample ♣-sequences for $d = 3$, i.e. qutrits.

Algorithm 1: $\{s_1, \ldots, s_p\}$ = **Make-♣-sequence**(d, n)

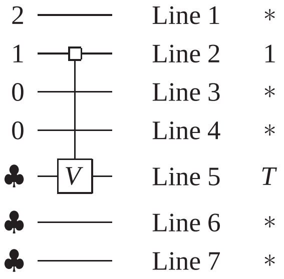
```
% We return a sequence of $p = (d^n - 1)/(d-1)$ terms, with n letters each,
% drawn from the alphabet $\{0, 1, \ldots, d-1, \clubsuit\}$.
Let $\{\tilde{s}_j\}_{j=1}^{\tilde{p}}$ = Make-♣-sequence ($d$,$n-1$).
for $q = 0, 1, \ldots, d-1$ do
    The next $(d^{n-1} - 1)/(d-1)$ terms of the sequence are formed by prefixing the letter $q$ to each
    term of the sequence $\{\tilde{s}_j\}$.
end for
The final term of the sequence is $\clubsuit^n$.
```

4.3 Inserting zeroes using Householder matrices in ♣-sequence order

The ♣-sequence determines the appropriate controls as well as the targets for a Householder matrix V. To be precise, we extend the earlier notation $\wedge_k(V)$. Namely, we write $\wedge(C,V)$ where V is a $d \times d$ unitary and C is a word in $\{0, 1, 2, \ldots, d-1\} \sqcup \{T\} \sqcup \{*\}$, of which at most one letter is T. The $*$ denotes a dropped control, so that $\wedge(C,V)$ changes a computational basis state $|j\rangle$ if and only if the dits of the d-ary expansion of j agree with similarly placed dits of C. The unitary V is then applied to the target qudit, whose position is denoted by T.

Each term of the clubsuit sequence corresponds to a $\wedge(C,V)$, in fact a singly-controlled Householder matrix. We illustrate with an example. The two-qudit gate below corresponds to the ♣-word

2100♣♣♣ for seven qudits.



```
Algorithm 2:   $\wedge(C,V)$ = **Single-♣Householder** (♣ term $t = t_1t_2\dots t_n$,   $n$-qudit state $|\psi_j\rangle$)
    Initialize $C = **\cdots*$
    % Set the target:
    Let $\ell$ be the index of the leftmost ♣ and set $C_\ell = T$.
    % Set a single control if needed:
    if $t$ contains numeric values greater than 0,
        Let $q$ be the index of the rightmost such value and set $C_q = t_q$.
    end if
    Extract from $|\psi_j\rangle$ the unnormalized $d$-component vector $|\varphi\rangle$ whose components have $d$-ary
    indices $t_1t_2\dots t_{\ell-1} * 00\dots 0$. As a formula, if $|\psi_j\rangle = \sum_{k=0}^{d^n-1} \alpha(k)\,|k\rangle$, then
    $\varphi = \sum_{q=0}^{d} \alpha(t_1t_2\dots t_{\ell-1}q00\dots 0)\,|q\rangle$. Form $V$ as a one-qudit Householder matrix such that
    $V\,|\varphi\rangle = |0\rangle$.
```

To construct the control word C, place the V-target symbol T on the leftmost club. So in this case $\wedge(C,V)$ targets line 5. Place *a single active control* on the least significant line carrying a nonzero prior to the target. (If there is no nonzero, then no control is necessary.) In our case this places a control corresponding to the 1 on line 2. We denote the control word by $C = *1**T**$, and the open box in the gate denotes the single control (which, as this example illustrates, is not necessarily on state $|d-1\rangle$). We complete the gate $\wedge(C,V)$ by choosing the one-qudit Householder matrix V to use element 2100000 to zero α_k for $k = 2100\ell 00$ and $\ell = 1,\dots,d-1$. The formal definition of $\wedge(C,V)$ as a function of a word in the ♣-sequence and a state vector $|\phi_j\rangle$, is detailed in Algorithm 2.

Our construction allows only a single active control. Hence each controlled operation is in fact a two-qudit gate, and since there are $O(d^n)$ words in our ♣-sequence, this would imply that two-qudit gates suffice for building the n-qudit state-synthesis unitary W. We can provide three equivalent descriptions of our algorithm for generating the gates of W with $W|\psi\rangle = |0\rangle^{\otimes n}$. Colloquially, the first uses terms of the ♣-sequence to construct $\wedge(C,V)$ which zero $|\psi\rangle$ sequentially, updating $|\psi\rangle$ as we go. More precisely, set $|\psi\rangle = |\psi_1\rangle$ and suppose inductively $|\psi_t\rangle = \prod_{k=p-t+2}^{p} \wedge[C(p-k+1),V(p-k+1)]\,|\psi\rangle$. Then we zero $d-1$ new entries by forming $|\psi_{t+1}\rangle = \wedge[C(t),V(t)]\,|\psi_t\rangle$ for $\wedge[C(t),V(t)]$ arising from **Single-♣Householder**, term t of the ♣-sequence, and $|\psi_t\rangle$. The second description is given in Algorithm 3, which implicitly includes the generation of the ♣-sequence within its structure of nested loops. (Cf. [4].) This algorithm produces a sequence of $(d^n-1)/(d-1)$ two-qudit gates factoring W. The nested loops generate the ♣-sequence terms inline. Using Algorithm 2, each term is used to form a $\wedge(C,V)$ zeroing $d-1$ amplitudes of the current state $|\psi_t\rangle$. A third way of considering the sequence of two-qudit gates and the zeroes they introduce within the entries of $|\psi\rangle$ (actually $|\psi_t\rangle$) is by considering a depth-first search of the tree of Figure 2. Atop each box vertex is

```
Algorithm 3:  ♣Householder($|\psi_1\rangle, d, n$)
% Reduce $|\psi_1\rangle$ onto $|0\rangle^{\otimes n}$.
$t = 0$
for $c_1 = 0 : d-1$
 for $c_2 = 0 : d-1$
   ⋱
    for $c_{n-2} = 0 : d-1$
     for $c_{n-1} = 0 : d-1$
      $t = t+1$
      Use Single-♣Householder($c_1 \ldots c_{n-1}\clubsuit, |\psi_t\rangle$) to zero $\{\langle c_1 c_2 \ldots c_{n-1} j_n | \psi_t\rangle, j_n > 0\}$
     end
     $t = t+1$
     Use Single-♣Householder($c_1 \ldots c_{n-2}\clubsuit\clubsuit, |\psi_t\rangle$) to zero $\{\langle c_1 c_2 \ldots c_{n-2} j_{n-1} 0 | \psi_t\rangle, j_{n-1} > 0\}$
    end
   ⋰
  end
 $t = t+1$
 Use Single-♣Householder($c_1 \clubsuit \ldots \clubsuit, |\psi_t\rangle$) to zero $\{\langle c_1 j_2 0 \ldots 0 | \psi_t\rangle, j_2 > 0\}$
end
$t = t+1$
Use Single-♣Householder($\clubsuit \ldots \clubsuit, |\psi_t\rangle$) to zero $\{\langle j_1 0 \ldots 0 | \psi_t\rangle, j_1 > 0\}$.
```

the ♣-term generating $\wedge(C,V)$ by Algorithm 2, while the amplitudes of all but the top ditstring of each box are zeroed by $\wedge(C,V)$.

4.4 Modification for $W|\psi\rangle = |j\rangle$

We have presented state-synthesis Algorithm 3, which maps $|\psi\rangle \mapsto |0\rangle^{\otimes n}$. However, the algorithm is easily adapted to realize $|\psi\rangle \mapsto |j\rangle$ for any $j = d_1 d_2 \ldots d_n$. Indeed, single qudit gates may permute the elements to put j in position 0, so that a similarity transform of **Single-♣Householder** suffices. To see this, let $j = d_1 d_2 d_3 \ldots d_n$ be a d-ary expansion of some j, $0 \leq j \leq d^n - 1$. Then $|j\rangle = \otimes_{k=1}^{n} \mathtt{INC}^{d_k} |0\rangle$. Further, for a generic control word C, define a new j-dependent control word $\tilde{C}$ by

$$\tilde{C}_k = \begin{cases} *, & C_k = * \\ T, & C_k = T \\ (C_k + d_k) \bmod d, & C_k \in \{0, 1, \ldots, d-1\} \end{cases} \tag{16}$$

Suppose also that $C_m = T$. Then noting that $[\mathtt{INC}^{d_m}]^\dagger = \mathtt{INC}^{d-d_m}$, we have the similarity relation

$$[\otimes_{k=1}^{n} \mathtt{INC}^{d_k}] \wedge (C,V) [\otimes_{k=1}^{n} \mathtt{INC}^{d-d_k}] \;=\; \wedge[\tilde{C},\ \mathtt{INC}^{m}\, V\, \mathtt{INC}^{d-d_m}]. \tag{17}$$

Hence mapping an arbitrary state $|\psi\rangle$ to any separable state $|j\rangle$ requires the same overhead in terms of two-qudit gates as does the mapping to the fiducial state $|0\rangle^{\otimes n}$.

4.5 Proof that ♣Householder achieves $W|\psi\rangle = |0\rangle^{\otimes n}$

Given n, $p(n) = (d^n - 1)/(d-1)$ is the number of elements of the ♣-sequence. It would suffice to prove (i) that each operator $\wedge[C(j), V(j)]$ guarantees $d-1$ new zeroes in the state $|\psi_{j+1}\rangle$ not guaranteed in $|\psi_j\rangle$ and (ii) moreover that $\wedge[C(j), V(j)]$ does not act on previously guaranteed zeroes. The assertion (i) is straightforward and left to the reader; see Figure 2 caption. However, the second

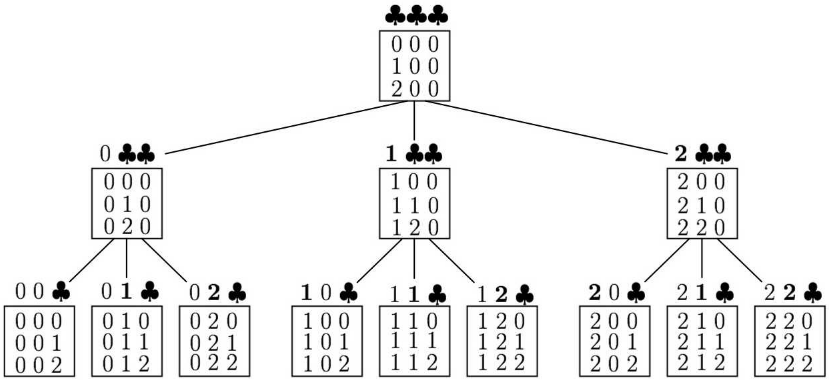


Fig. 2. Using the ♣-sequence for $d = 3$, $n = 3$ to generate Householder matrices to reduce $|\psi\rangle$ to a multiple of $|0\rangle$. Each node is labeled by a ♣-term and represents a gate $\wedge(C,V)$. The control is indicated by the boldface entry in the label. As the tree is traversed in a depth-first search, each node indicates a $\wedge(C,V)$ that zeroes the components of the last two indices in each node using the component of the top entry. See also Figure 1 of [4].

assertion is *false*. Rather, the controlled one-qudit operators do act on previously zeroed entries, but always replace them with a zero result. We next make this assertion precise and prove it.

Define the index set $S = \{0, 1, \dots, d^n - 1\}$ and introduce two new sets of dit-strings:

- $S_*(j)$ is the set of dit-strings for which the corresponding amplitude of $|\psi_j\rangle$ is not guaranteed zero by some $\wedge[C(k), V(k)]$, $k < j$.

- $S[C(j)]$ is the set of dit-strings on which the control $C(j)$ is active.

Also, define ℓ to be the index of the target symbol in $C(j)$: $C(j)_\ell = T$. Now there is a group action of $\mathbb{Z}/d\mathbb{Z}$ on the index set S corresponding to addition mod d on the ℓ^{th} dit:

$$c \bullet_\ell c_1 c_2 \dots c_n \;=\; c_1 c_2 \dots c_{\ell-1}(c_\ell + c \text{ mod d}) c_{\ell+1} \dots c_n. \tag{18}$$

Since the operator $V(j)$ is applied to qudit ℓ, the amplitudes (components) of $|\psi_{j+1}\rangle$ are either equal to the corresponding amplitude of $|\psi_j\rangle$ or else are linear combinations of the $|\psi_j\rangle$-amplitudes whose indices lie in the $\mathbb{Z}/d\mathbb{Z}$ orbit contained in $S[C(j)]$. To establish the correctness of ♣**Householder**, we will prove the following Proposition.

Proposition 1 *$|\psi_{j+1}\rangle$ has at least $d-1$ more guaranteed zero amplitudes than $|\psi_j\rangle$.*

Since ♣**Householder** sets $j = 1, \dots, (d^n - 1)/(d-1)$, this means that the final $|\psi_j\rangle$ has a single nonzero element corresponding to $|0\rangle$ and state synthesis has been achieved. We prove this result using three lemmas. To state them, we write $S_*(j)$ as the union of the three sets $R_1(j)$, $R_2(j)$, and $R_3(j)$ which we now define.

Definition 1 *Suppose the j^{th} term of the ♣-sequence is given by $c_1c_2\dots c_{\ell-1}♣\dots♣$. We have $C(j)$ the corresponding control word, with $C(j)_\ell = T$. Consider the following three sets, noting $R_1(j)$ may be empty.*

$$\begin{aligned}
R_1(j) &= \bigsqcup_{q=0}^{\ell-2}\left\{ c_1c_2\dots c_qk00\cdots 0\ ;\ k<c_{q+1}, k\in\{0,1,\dots,d-1\}\right\}\\
R_2(j) &= \left\{ c_1\cdots c_{\ell-1}k00\dots 0\ ;\ k\in\{0,1,\dots,d-1\}\right\} \qquad (19)\\
R_3(j) &= \left\{ f_1\cdots f_{\ell-1}k_\ell k_{\ell+1}\dots k_n\ ;\ f_1f_2\dots f_{\ell-1}>c_1c_2\cdots c_{\ell-1}, k_*\in\{0,1,\dots,d-1\}\right\}.
\end{aligned}$$

These sets may be interpreted in terms of Figure 2. Recall the figure recovers the ♣-sequence by doing a depth-first search of the tree. In this context, $S_*(j)$ is the set of possibly nonzero components of $|\psi_j\rangle$ at the j^{th} node. The subset $R_3(j)$ results from indices that lie in nodes not yet traversed, loosely above the present node in the tree or to the right. The set $R_2(j)$ is precisely the set of indices in the current node, node j. The set $R_1(j)$ is the set of indices of elements that have been previously used to zero other elements and still might remain nonzero themselves; it is the set of indices of elements that were always at the top of nodes already traversed in the depth-first search. Thus, $R_1(j)$ is loosely a set of entries within nodes to the left and perhaps below node j. The first lemma, along with the third, is used to show that the algorithm does not harm previously-introduced zeroes.

Lemma 1 *Suppose the ℓ^{th} letter of $C(j)$ is the target symbol T, and label $\tilde{S}_*(j) = R_1(j)\sqcup R_2(j)\sqcup R_3(j)$. Then*

$$(\mathbb{Z}/d\mathbb{Z})\ \bullet_\ell\ \tilde{S}_*(j)\cap S[C(j)]\ \subseteq\ \tilde{S}_*(j)\cap S[C(j)]. \qquad (20)$$

More colloquially, the $\mathbb{Z}/d\mathbb{Z}$ action which determines which amplitudes are mixed by the jth two-qudit gate respects the set of dit-strings $\tilde{S}_*(j)$. We require the tilde since we do not yet know that the union of the $R_*(j)$ is actually the set of unzeroed entries.

Proof: Due to the choice of a single control on a dit to the right of position ℓ in the appropriate term of the ♣-sequence, $R_1(j)\cap S[C(j)] = \emptyset$. On the other hand, a direct computation verifies that $(\mathbb{Z}/d\mathbb{Z})\bullet_\ell R_2(j)\subset R_2(j)$ and also that $R_2(j)\cap S[C(j)] = R_2(j)$.

Finally, we argue that $(\mathbb{Z}/d\mathbb{Z})\bullet_\ell R_3(j)\subset R_3(j)$. However, the following partition is in general nontrivial:

$$R_3(j)\ =\ \{R_3(j)\cap S[C(j)]\}\sqcup\{R_3(j)\cap\big(S-S[C(j)]\big)\}. \qquad (21)$$

Should $C(j)$ admit no control, we are done. If not, let $m<\ell$ be the control qudit. Then

$$R_3(j)\cap S[C(j)] = \left\{ f_1\cdots f_{\ell-1}k_\ell k_{\ell+1}\dots k_n\ ;\ \mathbf{f_m}=\mathbf{c_m}, f_1\dots f_{\ell-1}>c_1c_2\cdots c_{\ell-1}, k_*\in\{0,1,\dots,d-1\}\right\}. \qquad (22)$$

Hence the $\mathbb{Z}/d\mathbb{Z}$ action respects the partition of Equation 21 as well. □

The second lemma shows that the algorithm produces $d-1$ newly guaranteed zeroes at each step. In particular, $\mathcal{Z}$ is the set of elements zeroed by $\wedge[C(j),V(j)]$.

Lemma 2 *Let $C(j)$ result from $c_1c_2\dots c_{\ell-1}\clubsuit\dots\clubsuit\clubsuit$ of the $\clubsuit$-sequence as in §4.3. Consider the following set of dit-strings:*

$$\mathcal{Z} \;=\; \{c_1c_2\dots c_{\ell-1}k00\dots 0\;;\; k\in\{1,2,\dots,d-1\}\}. \tag{23}$$

Then $R_1(j)\sqcup R_2(j)\sqcup R_3(j) = R_1(j+1)\sqcup R_2(j+1)\sqcup R_3(j+1)\sqcup\mathcal{Z}$.

Proof: We break our argument into two cases based on the value of $c_{\ell-1}$.

Case $c_{\ell-1}<d-1$: Considering the lexicographic order and ignoring inappropriate words, the $(j+1)^{\text{st}}$ term of the $\clubsuit$-sequence is is given by $c_1c_2\dots(c_{\ell-1}+1)00\dots0\clubsuit$. Note that for leaves of the tree, the buffering sequence of zeroes is vacuous.

$$\begin{array}{lcl} R_1(j+1) & = & R_1(j)\sqcup R_2(j)-\mathcal{Z},\\ R_2(j+1)\sqcup R_3(j+1) & = & R_3(j). \end{array} \tag{24}$$

Hence $R_1(j)\sqcup R_2(j)\sqcup R_3(j) = R_1(j+1)\sqcup R_2(j+1)\sqcup R_3(j+1)\sqcup\mathcal{Z}$.

Case $c_{\ell-1}=d-1$: Suppose instead the j^{th} $\clubsuit$-sequence term is $c_1c_2\dots c_{\ell-2}(d-1)\clubsuit\clubsuit\dots\clubsuit$, so that the $(j+1)^{\text{st}}$ term is $c_1c_2\dots c_{\ell-2}\clubsuit\clubsuit\clubsuit\dots\clubsuit$. We note that $\{c_0c_1\dots c_{\ell-2}(d-1)0\dots0\}\in R_2(j)\cap R_2(j+1)$ [b]. Then

$$\begin{array}{lcl} R_1(j) & = & R_1(j+1)\sqcup R_2(j+1)-\{c_0c_1\dots c_{\ell-2}(d-1)0\dots0\},\\ R_2(j) & = & \mathcal{Z}\sqcup\{c_0c_1\dots c_{\ell-2}(d-1)0\dots0\},\\ R_3(j) & = & R_3(j+1). \end{array} \tag{25}$$

From the first two, $R_1(j)\sqcup R_2(j) = R_1(j+1)\sqcup R_2(j+1)\sqcup\mathcal{Z}$. Hence $R_1(j)\sqcup R_2(j)\sqcup R_3(j) = R_1(j+1)\sqcup R_2(j+1)\sqcup R_3(j+1)\sqcup\mathcal{Z}$. □

The third lemma shows that the set we considered in Lemma 1 is indeed the set of guaranteed zeros. After which, the main result follows immediately by the first of the three lemmas.

Lemma 3 *$S_*(j) = R_1(j)\sqcup R_2(j)\sqcup R_3(j)$ is the set of guaranteed zero amplitudes (components) of a generic $|\psi_j\rangle$.*

Proof: The proof is by induction. For $j=1$, we have

$$R_1(1)=\emptyset,\quad R_2(1)=\{00\dots0*\},\quad R_3(1)=\{c_1c_2\dots c_{n-1}*\;;\;\text{some } c_j>0\}. \tag{26}$$

Hence the entire index set $S=S_*(1)=R_1(1)\sqcup R_2(1)\sqcup R_3(1)$.

Hence, we suppose by way of induction that $S_*(j)=R_1(j)\sqcup R_2(j)\sqcup R_3(j)$ and attempt to prove the similar statement for $j+1$. Now $\wedge[C(j),V(j)]$ will add new zeroes to the amplitudes (components) with indices $\mathcal{Z}$ by Lemma 2. On the other hand, $\wedge[C(j),V(j)]$ will not destroy any zero amplitudes existing in $S_*(j)$ due to the induction hypothesis and Lemma 1. Thus $S_*(j+1)=R_1(j+1)\sqcup R_2(j+1)\sqcup R_3(j+1)$. □

Proof of 1: The main result now follows after combining our three lemmas. □

5 Unitary synthesis by reduction to triangular form

In this section, we present an asymptotically optimal unitary circuit not found in [4]. It leans heavily on the optimal state-synthesis of $\clubsuit$Householder. Since this state-synthesis circuit can likewise clear any length d^n vector using fewer than d^n single controls, the asymptotic is perhaps unsurprising. Yet

[b] So in the application, the amplitude (component) of this index is the single amplitude not zeroed by $\wedge[C(j),V(j)]$, but it is immediately afterwards zeroed by $\wedge[C(j+1),V(j+1)]$.

the unitary circuit requires highly-controlled one-qudit unitary operators when clearing entries near the diagonal. Optimality persists since these are used sparingly. Two themes should be made clear at the outset:

- We process the size $d^n \times d^n$ unitary V in subblocks of size $d^{n-1} \times d^{n-1}$.

- Due to rank considerations, at least one block in each block-column of size $d^n \times d^{n-1}$ must remain full rank throughout.

Hence, we cannot carelessly zero subcolumns. One solution is to triangularize the $d^{n-1} \times d^{n-1}$ matrices on the block diagonal, recursively. Given that strategy, the counts below show only $O(n^2 d^n)$ fully $(n-1)$ controlled one-qudit operations appear in the algorithm. This is allowed when working towards an asymptotic of $O(d^{2n})$ gates.

The organization for the algorithm is then as follows. Processing (triangularization) of V moves along block-columns of size $d^n \times d^{n-1}$ from left to right. In each block-column, we first triangularize the block $d^{n-1} \times d^{n-1}$ block-diagonal element, perhaps adding a control on the most significant qudit to a circuit produced by recursive triangularization. After this recursion, we zero the blocks below the block-diagonal element one column at a time. For each column j, $0 \leq j \leq d^{n-1}-1$, the zeroing process is to collapse the $d^{n-1} \times 1$ subcolumns onto their j^{th} entries, again adding a control on the most significant qudit to prevent destroying earlier work. These subcolumn collapses produce the bulk of the zeroes and are done using ♣**Householder**. After this, fewer than d entries remain to be zeroed in the column below the diagonal. These are eliminated using a controlled reflection containing $n-1$ controls and targeting the top line.

We now give a formal statement of the algorithm. We emphasize the addition of controls when previously generated circuits are incorporated into the universal circuit (i.e. recursively telescoping control.)

Algorithm 4: **Triangle**(U,d,n)

```
if $n = 1$ then
    Triangularize $U$ using a $QR$ reduction.
else
    Reduce top-left $d^{n-1} \times d^{n-1}$ subblock using Triangle$(*,d,n-1)$, (writing output to bottom
      $n-1$ circuit lines)
    for $m = 0,1,\ldots,d-1$ do     % Block-column iteration
        for columns $j = md^{n-1},\ldots,[(m+1)d^{n-1}-1]$ do
            for $\ell = (m+1),\ldots,(d-1)$ do % Block-row iterate
                Use ♣Householder to zero the column entries $(m+\ell)d^{n-1},\ldots,[(m+\ell+1)d^{n-1}-1]$,
                  leaving a nonzero entry at $(m+\ell)c_2\ldots c_n$ for $j = c_1 c_2 \ldots c_n$ and
                  adding $|m+\ell\rangle$- control on the most significant qudit.
            end for
            Clear the remaining nonzero entries below diagonal using one $\bigwedge(Tc_2\ldots c_n, V)$.
        end for     % All subdiagonal entries zero in block-col
        Use Triangle$(*,d,n-1)$ on the $d^{n-1} \times d^{n-1}$ matrix at the $(m+1)^{\text{st}}$ block diagonal
          adding $|m+1\rangle$- control to the most significant qudit.
    end for
end if-else
```

To generate a circuit for a unitary operator U, we use **Triangle** to reduce U to a diagonal operator $W = \sum_{j=0}^{d^n-1} \mathrm{e}^{i\phi_j} |j\rangle\langle j|$. Now V and $U = WV$ would be indistinguishable if a von Neumann measurement $\{|j\rangle\langle j|\}_{j=0}^{d^n-1}$ were made after each computation. However, the diagonal is important if U is a computation corresponding to a subblock of the circuit of a larger computation with other trailing, entangling interactions. In this case, the diagonal unitary can be simulated with $d^n \wedge_{n-1}(V)$ gates. Writing j in its d-ary expansion, $j = j_0 j_1 \dots j_{n-1}$ we have $W = \prod_{j=0}^{d^n-1} \otimes_{k=1}^{n} \mathtt{INC}_k^{j_k} \wedge_{n-1} (e^{i\phi_j |d-1\rangle\langle d-1|}) \otimes_{k=1}^{n}$ $\mathtt{INC}_j^{-j_k}$. By the argument in §3, the gate count for such a simulation is $O[d^n(n-1)^{2+\log_2 d}]$. This is asymptotically irrelevant compared to the lower bound.

5.1 Counting gates and controls

Let $h(n,k)$ be the number of k-controls required in the **Single-♣Householder** reduction of some $|\psi\rangle \in \mathcal{H}(n,d)$. Then $h(n,k) = 0$ for $k \geq 2$. Moreover, each 0-control results from an element of the ♣-sequence of the form $00\dots0$♣♣$\dots$♣, and there are n such sequences. Thus, since the number of elements of the ♣-sequence is $(d^n-1)/(d-1)$, we see that

$$\left\{ \begin{array}{rcr} h(n,1) & = & (d^n-1)/(d-1) - n \\ h(n,0) & = & n. \end{array} \right. \tag{27}$$

We next count controls in the matrix algorithm **Triangle**. We break the count into two pieces: g for the work outside the main diagonal blocks and f for the total work.

Let $g(n,k)$ be the number of k-controls applied in operations in each column that zero the matrix below the block diagonal; this is the total work in the **for** j loops of **Triangle**. We use **Single-♣Householder** $d(d-1)d^{n-1}/2$ times since there are $d(d-1)/2$ blocks of size $d^{n-1} \times d^{n-1}$ below the block diagonal, and we add a single control to those counted in h. The last statement in the loop is executed $d^n - d^{n-1}$ times. Therefore, letting δ_j^k be the Kronecker delta, the counts are

$$g(n,k) = \delta_k^{n-1}(d^n - d^{n-1}) + \frac{1}{2}d(d-1)d^{n-1}h(n-1,k-1). \tag{28}$$

Supposing $n \geq 3$, then we see that

$$g(n,k) = \left\{ \begin{array}{rr} d^n - d^{n-1}, & k = n-1 \\ 0, & n-1 \leq k \leq 3 \\ \frac{1}{2}d^n(d^{n-1}-1) - \frac{1}{2}d^n(d-1)(n-1), & k=2 \\ \frac{1}{2}d^n(d-1)(n-1), & k=1 \\ 0, & k=0. \end{array} \right. \tag{29}$$

Finally, let $f(n,k)$ be the total number of k-controlled operations in the **Triangle** reduction, including the block diagonals. This work includes that counted in g, plus a recursive call to **Triangle** before the **for** m loop, plus $(d-1)$ calls within the k loop, for a total of

$$f(n,k) = g(n,k) + f(n-1,k) + (d-1)f(n-1,k-1), \tag{30}$$

with $f(n,0) = 1$ and $f(1,k) = 0$ for $n,k > 0$.

Using the recursive relation of Equation 30 and the counts of Equation 29, we next argue that **Triangle** has no more than $O(d^{2n})$ controls. The following lemma is helpful.

Lemma 4 *For sufficiently large n, we have* $f(n,k) \leq d^{2n-k+4}$.

Proof: By inspection of Equation 29, we see that $g(n,k) \leq (1/2)d^{2n-k+2}$ for all k and n large. Now $f(n,0) = 1$, which we take as an inductive hypothesis while supposing $f(n-1,\ell) \leq d^{2n-2-\ell+4} = d^{2n-\ell+2}$. Thus, using the recursion relation of Equation 30,

$$\begin{array}{rcl} f(n,k) & \leq & \frac{1}{2}d^{2n-k+2} + d^{2n-k+2} + (d-1)d^{2n-k+3} \\ & = & d^{2n-k+4}\left(\frac{1}{2d^2} + \frac{1}{d^2} + 1 - \frac{1}{d}\right). \end{array} \tag{31}$$

Now since $d > 3/2$, we must have $\frac{1}{d} > \frac{3}{2d^2}$, whence an inductive proof of the result. □

By the results from §3, each k-controlled single-qudit unitary operator costs $c_k = O[(k+2)^{2+\log_2(d)}]$ `CINC` and $\texttt{CINC}^{-1}$ gates without ancillas. The expected number of `CINC` gates ℓ_T for the algorithm **Triangle** is then given by the weighted sum for the k-control gates in the diagonalization and the d^n instances of $n-1$-controlled phase gates for emulation of the diagonal:

$$\begin{array}{rcl} \ell_T & = & d^n c_{n-1} + \sum_{k=0}^{n-1} c_k f(n,k) \\ & \leq & 2(n+1)^{2+\log_2(d)} d^{n+4} + d^{8+2n} \sum_{k=0}^{n-1} d^{-k} k^{2+\lceil \log_2 d \rceil} \\ & \leq & 2(n+1)^{2+\log_2(d)} d^{n+4} + d^{8+2n} \mathrm{Li}_{-(2+\lceil \log_2 d \rceil)}(1/d) \\ & \leq & 2(n+1)^{2+\log_2(d)} d^{n+4} + 26 d^{8+2n}. \end{array} \tag{32}$$

In the third line we have used the fact that for the Polylogarithm function, $\mathrm{Li}_{-(2+\lceil \log_2 d \rceil)}(1/d) \leq \mathrm{Li}_{-3}(1/2) = 26$.

5.2 Comparison with the spectral algorithm

In an earlier work [4], we described an different algorithm for unitary synthesis. That algorithm relied on a spectral decomposition of the unitary and was also shown to be asymptotically optimal. For a circuit without ancillas, the `CINC` gate count ℓ_S using the spectral algorithm is:

$$\ell_S \leq 2d^{n+1}[(d^n - 1)/(d-1) - n] + (n+1)^{2+\log_2 d} d^{n+4}. \tag{33}$$

Our tests show that the spectral algorithm outperforms **Triangle** when the number of qudits is greater than two. The general d^{2n} scaling of both algorithms is shown in Figure 3.

Yet there are several situations where **Triangle** may be preferred over the spectral algorithm. First, consider two-qudit circuits. For **Triangle** each $\wedge_1(V)$ gate is chosen to zero a maximum number of elements below the diagonal of the unitary resulting from previous controlled operations. On average, $d-1$ components are zeroed by each controlled unitary gate, hence the number of gates needed to bring the unitary U to diagonal form is the number of zeros below the diagonal divided by $d-1$ or $k = (d^4 - d^2)/[2(d-1)]$. The first of these transformations can be implemented by a local gate but all subsequent gates must be controlled to avoid introducing new nonzero elements. For example, if $d = 3$, then the elements below the main diagonal are zeroed at the time steps indicated in the following matrix (x's indicate values that change during the gate sequence):

$$\begin{bmatrix} x & x & x & x & x & x & x & x & x \\ 1 & x & x & x & x & x & x & x & x \\ 1 & 1 & x & x & x & x & x & x & x \\ 4 & 5 & 8 & x & x & x & x & x & x \\ 2 & 7 & 8 & 11 & x & x & x & x & x \\ 2 & 5 & 10 & 11 & 11 & x & x & x & x \\ 4 & 6 & 9 & 13 & 14 & 16 & x & x & x \\ 3 & 7 & 9 & 12 & 15 & 16 & 18 & x & x \\ 3 & 6 & 10 & 12 & 14 & 17 & 18 & 18 & x \end{bmatrix}.$$

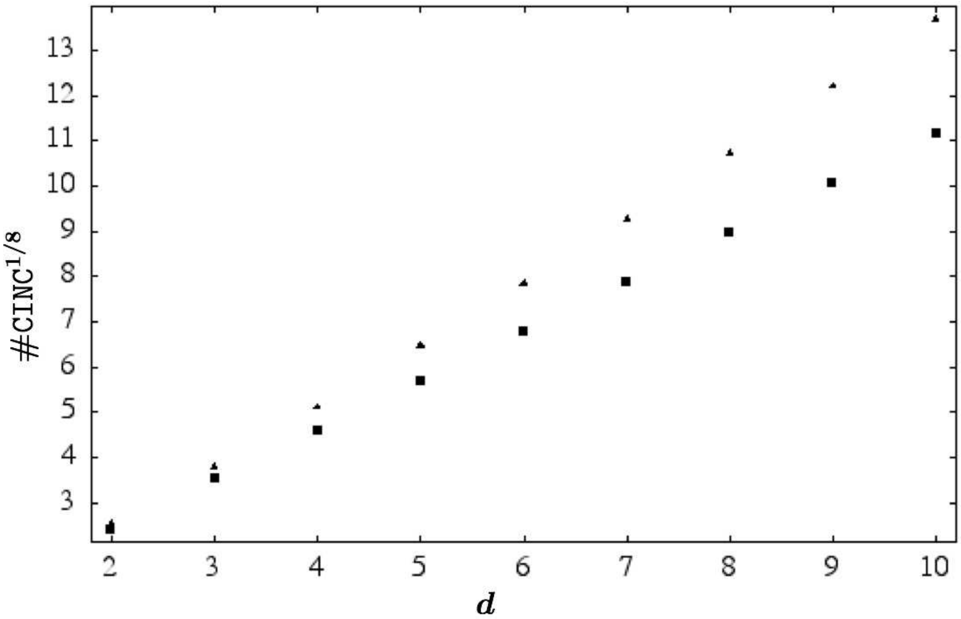


Fig. 3. Performance comparison of the two algorithms for unitary synthesis on $n = 4$ qudits as a function of qudit dimension d. Triangles (boxes) indicate `CINC` gate counts for the **Triangle** (spectral) algorithm.

The final diagonal unitary $D = \sum_{j,k=0}^{d} e^{i\phi_{jk}|j\rangle\langle j|\otimes|k\rangle\langle k|}$ can be simulated using d controlled single qudit gates as evident by the decomposition

$$D = \prod_{j=0}^{d-1}[e^{i\frac{\pi}{2}(|j\rangle\langle d-1|+|d-1\rangle\langle j|)} \otimes 1_d] \wedge_1 (\sum_{k=0}^{d-1} e^{i\phi_{jk}|k\rangle\langle k|})[e^{-i\frac{\pi}{2}(|j\rangle\langle d-1|+|d-1\rangle\langle j|)} \otimes 1_d].$$

The $\wedge_1(V)$ gate count is then $\ell_T = d^2(d+1)/2 + d$ gates. In contrast, the gate count using the spectral algorithm is $\ell_S = 2d^3 - d^2$.

Second, in certain circumstances it may only be necessary to build a unitary up to measurement in a logical basis. Then the simulation of the final diagonal unitary in **Triangle** can be dropped. Third, the spectral algorithm requires a classical diagonalization of the unitary U which requires $O(d^{3n})$ steps. For matrices of large size, particularly when there are degenerate eigenstates, numerical stability can be an issue. The classical computations involved in **Triangle** also scale like $O(d^{3n})$ but are carried out directly in the logical basis of the qudits. Finally, in zeroing entries in analogy to a QR decomposition, **Triangle** has more direct precedents among earlier quantum circuit designs in the literature [12, 1, 13, 2] and in particular the first asymptotically optimal $d = 2$ construction [3].

6 Two applications of state-synthesis

A primary motivation for describing state-synthesis circuits is to utilize them as subcircuits for unitary synthesis as in §5. Yet there are also independent applications for the algorithm. We present two examples.

6.1 Computing expected values

First, consider the problem of computing the expectation value of a Hermitian operator $A \in \mathcal{H}(n,d)$ i.e. $A \in \mathrm{End}[\mathcal{H}(n,d)] \cong \mathbb{C}^{d^n \times d^n}$ with $A^\dagger = A$. For a system in the possibly mixed state ρ of n qudits, the the expectation of an operator A is $\langle A \rangle = \mathrm{Tr}[A\rho]$. In some cases there does not exist a physically realistic direct measurement of A. However, one may infer the expectation value by a suitably weighted set of von Neumann measurements as follows. By the spectral theorem, any normal operator A may be diagonalized by a unitary transformation U: $A = U^\dagger D U$ where $D = \sum_{j=0}^{d^n-1} \lambda_j |j\rangle\langle j|$ and $\{\lambda_j\}_{j=0}^{d^n-1}$ are the eigenvalues of A. Then

$$\langle A \rangle \;=\; \mathrm{Tr}[A\rho] \;=\; \mathrm{Tr}[DU\rho U^\dagger] \;=\; \sum_{j=0}^{d^n-1} \lambda_j \mathrm{Tr}\big[\, |j\rangle\langle j| U\rho U^\dagger \,\big]. \tag{34}$$

Hence we may compute $\langle A \rangle$, given an *arbitrary* state ρ, by performing two steps: first, enact the unitary evolution U on ρ, and second perform the computational-basis von Neumann measurement on the resulting state, extracting all populations of the basis states $|j\rangle\langle j|$.

In some instances one may want to know the weight of a quantum state on a subspace of the operator A, i.e. $\langle P_S A P_S \rangle$ where P_S is some projection operator onto a subspace $\mathcal{H}_S \subseteq \mathcal{H}(n,d)$. In particular, consider the case of a k dimensional subspace diagonal in the eigenbasis $\{|u_j\rangle\}_{j=0}^{d^n-1}$ of A. We wish to compute $\mathrm{Tr}[\sum_{j=1}^{k} \lambda_j |u_j\rangle\langle u_j| \rho]$ where $k < d^n$ and the eigenvalues of A have been reordered accordingly. Then we can rewrite the projection $P_S A P_S = \sum_{j=1}^{k} \lambda_j W(u_j)|j\rangle\langle j| W(u_j)^\dagger$ where $W(u_j)$ is a unitary extension of the mapping $|j\rangle \to |u_j\rangle$. The operator $W(u_j)$ is the unitary obtained in the state-synthesis algorithm. The expectation value is then

$$\langle P_S A P_S \rangle \;=\; \sum_{j=1}^{k} \lambda_j \mathrm{Tr}\big[\, |j\rangle\langle j| W(u_j)^\dagger \rho W(u_j) \,\big]. \tag{35}$$

The expectation value can be measured as before but now one need only implement the state-synthesis operator k times on each state ρ of an ensemble of identically prepared states.

The above argument may in fact be generalized to compute the expectation value of any operator A. First decompose the operator as $A = A_h + A_a$ with $A_h = (A + A^\dagger)/2$ the Hermitian part and $A_a = (A - A^\dagger)/2$ the anti-Hermitian part of A. Both A_h and A_a are normal operators and therefore can be diagonalized. Hence, the expectation value can be computed by evaluating the weighted sum as per Eq. 34 and summing.

6.2 The general state-synthesis problem

Both **Triangle** and the spectral algorithm are well adapted to the general state-synthesis problem. This problem demands synthesizing any unitary extension of the many state mapping $\{|j\rangle \to |\psi_j\rangle \mid 0 \leq j \leq \ell \ll d^n\}$ [1]. It is unclear what sorts of applications might arise when the states are arbitrary, requiring exponentially expensive circuits to build each. Nonetheless, less generic unitaries of this form have been used in quantum error correction to encode a few logical qudits into many physical qudits [15].

Triangle provides one solution to this problem. Start with a matrix containing $|\psi_j\rangle$ in its jth column, with “don’t care” entries in columns after column ℓ. Ignore any operations on the “don’t care” entries, and discard any gates meant to place zeros among them.

The spectral algorithm provides an alternative solution. Note that the matrix U formed from the product of the ℓ Householder matrices necessary to reduce the $d^n \times \ell$ matrix $[|\psi_1\rangle \dots |\psi_\ell\rangle]$ to diagonal form has $d^n - \ell$ eigenvalues equal to 1, so the spectral algorithm needs to build an eigenstate, apply a conditional phase to one logical basis ket, and unbuild the eigenstate only ℓ times.

7 Conclusions

This work concerns asymptotically optimal quantum circuits for qudits. By asymptotically optimal, we mean that the circuits require $O(d^n)$ gates of (no more than) two qudits for constructing arbitrary states and $O(d^{2n})$ gates for unitary evolutions. Contributions of this work are the following:

- We provide the first argument that both asymptotics survive even when no ancilla (helper) qudits are allowed.
- We present the state-synthesis circuit in much more detail than previously published, in particular describing it in terms of iterates over a $\clubsuit$-sequence. Using the $\clubsuit$ sequence, we provide the first proof that the state-synthesis circuits actually achieve $U|0\rangle = |\psi\rangle$.
- We present **Triangle**, a new asymptotically optimal quantum circuit for qudit unitaries which is inspired by QR matrix factorization. Since it leans more heavily on QR than on spectral decomposition, the gate parameters of **Triangle** require less classical pre-processing than the spectral algorithm. Moreover, **Triangle** more closely resembles earlier quantum circuit design techniques [12, 3] than other asymptotically optimal qudit unitary circuits.
- §2 provides a constructive proof that $\{\texttt{CINC}\} \sqcup U(d)^{\otimes n}$ is exact-universal for qudits.

Some open questions remain. The $\wedge_1(V)$ gates are much better than earlier practice but not provably optimal, as is the case with qubits [11]. Moreover, the current best-practice n-qubit circuits exploit the cosine-sine decomposition (CSD). The tensor product structure for qudits [16] is not as well adapted to a single CSD as in the $d = 2$ case, and it is unclear whether a CSD circuit for qudits could be made asymptotically optimal in the $\{\texttt{CINC}\} \sqcup U(d)^{\otimes n}$ library.

Acknowledgements

DPO received partial support from the National Science Foundation under Grants CCR-0204084 and CCF-0514213. GKB was supported in part by a grant from DARPA/QUIST. SSB was supported by a National Research Council postdoctoral fellowship. We are grateful to the referees for their careful reading and their suggestions to improve the exposition.

References

1. E. Knill, *Approximation by Quantum Circuits*, http://arxiv.org/abs/quant-ph/9508006.
2. A. Muthukrishnan and C.R.Stroud Jr., *Multivalued Logic Gates for Quantum Computation*, Phys. Rev. A, **62**, 052309 (2000).
3. J.J.Vartiainen, M.Möttönen, M.M.Salomaa, *Efficient Decomposition of Quantum Gates*, Phys. Rev. Lett., **92**, 177902 (2004).
4. S.S. Bullock, D.P. O'Leary, and G.K. Brennen, *Asymptotically Optimal Quantum Circuits for d-level Systems*, Phys. Rev. Lett., **94**, 230502 (2005).
5. D. Deutsch, A. Barenco, A. Ekert, *Universality in Quantum Computation*, Proc. R. Soc. London A, **449**, 669 (1995).
6. V. Shende, S. Bullock, and I. Markov, *Synthesis of Quantum Logic Circuits*, Proceedings on Asia and South Pacific Design Automation Conference, 272 (2005), quant-ph/0406176.

7. R. Tucci, A Rudimentary Quantum Compiler, http://arxiv.org/abs/quant-ph/9805015.
8. F.S. Khan and M. Perkowski, *Synthesis of multi-qudit Hybrid and d-valued Quantum Logic Circuits by Decomposition*, http://arxiv.org/abs/quant-ph/0511019.
9. V. Bergholm, J. Vartiainen, M.Möttönen, and M. Salomaa, *Quantum Circuits with Uniformly Controlled one-qubit Gates*, Phys. Rev. A, **71** 052330 (2005).
10. G.K. Brennen, D.P. O'Leary, and S.S. Bullock, *Criteria for Exact Qudit Universality*, Phys. Rev A, **71** 052318 (2005).
11. G. Song and A. Klappenecker, *Optimal Realizations of Controlled Unitary Gates*, Quantum Inf. Comput., **3(2)** 139 (2003).
12. A. Barenco, C. Bennett, R. Cleve, D. DiVincenzo, N. Margolus, P. Shor, T. Sleator, J. Smolin, and H. Weinfurter, *Elementary Gates for Quantum Computation*, Phys. Rev A, **52**, 3457 (1995).
13. G. Cybenko, *Reducing Quantum Computations to Elementary Unitary Operations*, Comp. in Sci. and Eng., **3**, March/April, 27 (2001).
14. G.H. Golub and C. van Loan, *Matrix Computations*, Johns Hopkins Press (Baltimore), (1989).
15. M. Grassl, M. Roetteler, T. Beth, *Efficient Quantum Circuits for Non-Qubit Quantum Error-Correcting Codes*, International Journal of Foundations of Computer Science, **14**, pp. 757 (2003).
16. K.G.H. Vollbrecht and R.F. Werner, *Why two qubits are special*, J. Math. Phys. **41**, 6772 (2000).