Transition-Aware Decomposition of Single-Qudit Gates

Denis A. Drozhzhin,[1] Evgeniy O. Kiktenko,[1] Aleksey K. Fedorov,[1] and Anastasiia S. Nikolaeva[1]

[1] *National University of Science and Technology "MISIS", Moscow 119049, Russia*

Quantum computation with d-level quantum systems, also known as qudits, benefits from the possibility to use a richer computational space compared to qubits. However, for arbitrary qudit-based hardware platform the issue is that a generic qudit operation has to be decomposed into the sequence of native operations – pulses that are adjusted to the transitions between two levels in a qudit. Typically, not all levels in a qudit are simply connected to each other due to specific selection rules. Moreover, the number of pulses plays a significant role, since each pulse takes a certain execution time and may introduce error. In this paper, we propose a resource-efficient algorithm to decompose single-qudit operations into the sequence of pulses that are allowed by qudit selection rules. Using the developed algorithm, the number of pulses is at most $d(d-1)/2$ for an arbitrary single-qudit operation. For specific operations the algorithm could produce even fewer pulses. We provide a comparison of qudit decompositions for several types of trapped ions, specifically $^{171}Yb^{+}$, $^{137}Ba^{+}$ and $^{40}Ca^{+}$ with different selection rules, and also decomposition for superconducting qudits. Although our result deal with single-qudit operations, the proposed approach is important for realizing two-qudit operations since they can be implemented as a standard two-qubit gate that is surrounded by efficiently implemented single-qudit gates.

I. INTRODUCTION

The development of a large-scale quantum computer [1–3] that is able to outperform devices based on classical principles in practically relevant problems, such as simulating complex (quantum) systems [4], combinatorial optimization [5], or prime factorization [6], represents an outstanding challenge. Various computational models, for example, digital quantum computing [1, 2], variational quantum algorithms [7, 8], or programmable quantum simulation [9], and diverse physical platforms, such as superconducting circuits [10, 11], semiconductor quantum dots [12–14], optical systems [15, 16], neutral atoms [9, 17–19], and trapped ions [20–22], are under study on the way to useful quantum computers. The inherent idea is to find suitable conditions for scaling quantum devices with respect to the number of available information carriers without significantly degrading the quality of the control. In the most well-developed digital model of quantum computing, information carriers are qubits [1–3], which are two-level quantum counterparts of classical bits. The execution of quantum algorithms requires the realization of single- and two-qubit operations under a register of qubits, so that the combination of the number of qubits and quality of quantum operations (gates) is the crucial parameter.

Underlying physical platforms, such as trapped ions or neutral atoms, however, almost always have Hilbert spaces of higher dimensionalities, which gives interesting possibilities [23].Computing models based on the use of additional degrees of freedom of physical systems, which makes them qudits (where d indicates that the dimension of the Hilbert space may be larger than two), attract great interest since they allow scalability of quantum computing devices without the need for an increase in the number of physical carriers [24–62]. There are several approaches to how qudits can be used for more efficient quantum processors. First, one can think about a multilevel system with 2^n levels as a set of qubits: for example, a ququart is equivalent to a two-qubit system [63–65]. At the same time, implementing a two-qubit operation with a single ququart (strictly speaking, this is a single-qudit operation) would not require physical interaction between physical objects, so the fidelity of such operation can be as high as the fidelity of single-qubit operation. The second approach is the use of additional qudit levels to substitute ancilla qubits in multiqubit gate decompositions [66] (for example, for the Toffoli gate [36, 38, 40, 43, 67–73]). However, the efficiency of these methods and their combination depends on the mapping, i.e. the way in which qubits are encoded in qudits [63, 65].

As in the case of conventional qubit-based processors, the choice of a specific physical platform is also important for qudit setups. Recently, multiqudit quantum processors [74–77], including systems based on superconducting transmon qubits [74] and photons [75] as well as $^{40}Ca^{+}$ [76] and $^{171}Yb^{+}$ [77] ion qudits, have been demonstrated. It is clear that in the case of atomic qubits (ions or neutral atoms), qudit encoding is natural due to multilevel structures, and it is straightforward to address more than two levels using a single laser with acousto-optic modulators (AOMs) [76, 77]. Superconducting [78, 79] and photonic [80] systems are also promising for qudit-based quantum computing. Recent progress in quantum computing with qudits also includes the demonstration of a significant improvement in the realization of multiqubit gates [81, 82] and quantum algorithms with qudits [39, 78]. However, various details of resource-efficient implementation of quantum algorithms with qudits are the subject of research.

In this work, we address a specific problem of the realization of single-qudit operations, taking into account selection rules of physical systems. The objective is to minimize the number of pulses in order to realize a single-qudit operation, since such a minimization would result

in shorter execution time and lower amount of errors. We propose a resource-efficient algorithm to decompose qudit operations into the sequence of pulses that are allowed by qudit selection rules. As we demonstrate, in the proposed algorithm the number of pulses is at most $d(d-1)/2$ for an arbitrary single-qudit operation, or even fewer pulses for specific single-qudit operations. We provide a comparison of qudit decompositions for several types of trapped ions, specifically $^{171}Yb^{+}$, $^{137}Ba^{+}$ and $^{40}Ca^{+}$ with different selection rules, and also decomposition for superconducting qudits.

II. QUDIT COMPUTING

A common quantum device relies on operations with *qubits*, i.e. the set of transformations of two-level systems in the superposition states:

$$|\psi\rangle_{\mathbf{qb}} = \psi_0 |0\rangle + \psi_1 |1\rangle . \tag{1}$$

In the case of *qudits*, the structure of the superposition state of the d-level quantum system is more complex:

$$|\psi\rangle_{\mathbf{qd}} = \psi_0 |0\rangle + \cdots + \psi_{d-1} |d-1\rangle = \sum_{n=0}^{d-1} \psi_n |n\rangle , \tag{2}$$

where d is the dimension of the quantum state.

The evolution of a qudit state can be described using unitary operators, similarly to the case of qubit operators. For example, if we interpret qubit Pauli σ_x as an operation that swaps two levels, then we obtain:

$$\mathsf{SWAP}^{ij}_{\mathbf{qd}} = |i\rangle\langle j| + |j\rangle\langle i| + \sum_{n\neq i,j} |n\rangle\langle n| . \tag{3}$$

On the other hand, σ_x can be generalized to modulo d increment:

$$\mathsf{X}_{\mathbf{qd}} = \sum_n |n{+}1 \bmod d\rangle\langle n| , \tag{4}$$

$$\mathsf{X}^k_{\mathbf{qd}} = \sum_n |n{+}k \bmod d\rangle\langle n| . \tag{5}$$

Also, one can use phase gates $\mathsf{Z}_{\mathbf{qd}}$ and the d-nary Hadamard transform $\mathsf{H}_{\mathbf{qd}}$, which is equivalent to the Quantum Fourier Transform on a single qudit:

$$\mathsf{Z}_{\mathbf{qd}} = \sum_n \omega^n |n\rangle\langle n| , \tag{6}$$

$$\mathsf{H}_{\mathbf{qd}} = \sum_{n,m} \frac{\omega^{nm}}{\sqrt{d}} |n\rangle\langle m| , \tag{7}$$

where $\omega = e^{\iota 2\pi/d}$ is the d^{th} root of unity.

In general, any single-qudit evolution can be described using elements of the unitary group $\mathsf{U}_{\mathbf{qd}} \in U(d)$. This group is isomorphic to the group of $d{\times}d$ matrices with complex entries that satisfy the unitary condition:

$$\mathsf{U}_{\mathbf{qd}} \equiv \begin{Vmatrix} \mathcal{U}_{0,0} & \mathcal{U}_{0,1} & \cdots & \mathcal{U}_{0,d-1} \\ \mathcal{U}_{1,0} & \mathcal{U}_{1,1} & \cdots & \mathcal{U}_{1,d-1} \\ \vdots & \vdots & \ddots & \vdots \\ \mathcal{U}_{d-1,0} & \mathcal{U}_{d-1,1} & \cdots & \mathcal{U}_{d-1,d-1} \end{Vmatrix} \tag{8}$$

$$\mathsf{U}_{\mathbf{qd}}{}^{\dagger}\mathsf{U}_{\mathbf{qd}} = \mathsf{Id} \equiv \sum_{k=0}^{d-1} \mathcal{U}^*_{k,n}\, \mathcal{U}_{k,m} = \delta_{n,m} \tag{9}$$

The knowledge of how to efficiently implement arbitrary single-qudit evolution with existing qudit hardware is the crucial step towards the universal qudit computer. Especially, for the systems with high dimensionality [78, 79, 83–85]

Another important part of computation with qudits is entangling operations. Using the extension of Pauli operations to qudits, we obtain the analogues of qubit controlled gates, particularly CX and CZ:

$$\mathsf{CU}_{\mathbf{qd}} = \sum_{m=0}^{d-1} |m\rangle\langle m| \otimes \mathsf{U}^m_{\mathbf{qd}}, \tag{10}$$

$$\mathsf{CX}_{\mathbf{qd}} = \sum_{m,n} |m, n{+}m\rangle\langle m, n| , \tag{11}$$

$$\mathsf{CZ}_{\mathbf{qd}} = \sum_{m,n} \omega^{mn} |m, n\rangle\langle m, n| . \tag{12}$$

Also, on several trapped-ion platforms the main two-qudit operation is Mølmer-Sørensen gate [76, 86]:

$$\mathsf{MS}(\chi) = \exp\left\{-\iota\frac{\chi}{2}\left(\sigma_x^{01}\otimes 1 + 1\otimes\sigma_x^{01}\right)^2\right\}, \tag{13}$$

which is equivalent up to global phase to qubit RXX operation on the subspace $\{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$:

$$\mathsf{RXX}(\chi) = \exp\{-\iota\chi\ \sigma_x\otimes\sigma_x\} . \tag{14}$$

Although we do not consider qudit entangling operations within the scope of this paper, however, we are able to embed n-qubit gates into a single-qudit operation if $d \geq 2^n$.

III. QUDIT HARDWARE OPERATIONS

From the experimental point of view, the qudit system can be implemented using different physical systems: trapped ions, neutral atoms, superconducting circuits, photonic circuits, etc.

A qudit state can be encoded into ions' or neutral atoms' degrees of freedom, e.g. states of the electron on the outer shell [84–86]. In this model, each state $|n\rangle$ has the corresponding energy E_n. After applying the laser field with a frequency $\frac{1}{\hbar}|E_i - E_j|$, the Rabi oscillation

of the electron between states $|i\rangle$ and $|j\rangle$ is stimulated if this transition is permitted by selection rules.

Superconducting circuits are able to store qudit states as Fock states of Copper pairs in the potential well [78, 79]. Transitions between states are also stimulated via Rabi pulses between neighboring states $|n\rangle$ and $|n{+}1\rangle$. The potential well should provide some degree of anharmonicity so that each transition can be addressed individually.

Photonic circuits can represent a qudit state using d-wire schemes with a single photon [75, 80]. An index of the basis qudit state represents the index of the wire. Using the beam-splitter with a given amplitude reflection rate r (it is usually set $r{=}1/\sqrt{2}$ for a half-transparent mirror), we can permute photons in two neighboring wires. Also, we can perform a phase-shift of a single wire at an arbitrary angle.

We can note that every mentioned qudit platform supports two main types of operations. The first is the phase shift on the level $|k\rangle$, which we denote $\mathsf{P}^k(\theta)$. The second is a two-level transition $\mathsf{R}^{ij}(\theta,\phi)$ between levels $|i\rangle$ and $|j\rangle$. These operations have the following definitions:

$$\mathsf{P}^k(\theta) = \exp\left\{\iota\theta\,|k\rangle\langle k|\right\}, \tag{15}$$

$$\mathsf{R}^{ij}(\theta,\phi) = \exp\left\{-\iota\frac{\theta}{2}\sigma_\phi^{ij}\right\}, \tag{16}$$

where $\sigma_\bullet^{ij}$ is the qudit analogue of qubit Pauli operators acting on the two-level subspace $\{|i\rangle, |j\rangle\}$ of qudit:

$$\sigma_x^{ij} = |j\rangle\langle i| + |i\rangle\langle j|, \tag{17}$$

$$\sigma_y^{ij} = \iota\,|j\rangle\langle i| - \iota\,|i\rangle\langle j|, \tag{18}$$

$$\sigma_\phi^{ij} = e^{+\iota\phi}\,|j\rangle\langle i| + e^{-\iota\phi}\,|i\rangle\langle j|. \tag{19}$$

Thus, R^{ij} operation can be viewed as $2{\times}2$ unitary matrix R that acts on subspace spanned by $\{|i\rangle, |j\rangle\}$:

$$\mathsf{R}(\theta,\phi) = \begin{Vmatrix} \cos\left(\frac{\theta}{2}\right) & -\iota e^{-\iota\phi}\sin\left(\frac{\theta}{2}\right) \\ -\iota e^{+\iota\phi}\sin\left(\frac{\theta}{2}\right) & \cos\left(\frac{\theta}{2}\right) \end{Vmatrix}. \tag{20}$$

Several platforms may lack two-parametric transition operations, e.g. for photonic circuits there is only a single parameter r in the beam-splitter operation:

$$\mathsf{T}(r) = \begin{Vmatrix} r & -\sqrt{1-r^2} \\ \sqrt{1-r^2} & \sqrt{r} \end{Vmatrix}. \tag{21}$$

We may always apply phase shifts and transform this operation into R^{ij} operation:

$$\mathsf{R}^{ij}(\theta,\phi) = \mathsf{P}^j\left(\phi-\frac{\pi}{2}\right)\circ\mathsf{T}^{ij}\left(\cos\frac{\theta}{2}\right)\circ\mathsf{P}^j\left(\frac{\pi}{2}-\phi\right). \tag{22}$$

We also consider arbitrary selection rules for a given qudit physical implementation that permit applicable R^{ij} operations. In the following sections, we demonstrate that any qudit operation U can be decomposed into the sequence of allowed R^{ij} and P^k operations. Moreover, for any selection rules, we guarantee at most $d(d{-}1)/2$ transition operations R^{ij} in the final decomposition.

IV. UNITARY DECOMPOSITION FOR TRAPPED-ION AND SUPERCONDUCTING HARDWARE

Existing decomposition schemes utilize superconducting selection rules: transitions are allowed only for neighboring states. Two distinct schemes are provided in [87] (*Row-by-row* scheme for superconducting circuits) and in [88] (*Square* scheme for photonic circuits). Both schemes perform unitary decomposition by eliminating non-diagonal entries from unitary matrix U using a sequence of two-level R operations.

In these decompositions, we initialize the algorithm with a unitary $d{\times}d$ matrix. After right-applying a two-level operation $\mathsf{R}^{01}(-\theta_1,\phi_1)$ to U, elements of the first two columns are modified in the following way:

$$\begin{Vmatrix} \mathcal{U}'_{k,0} \\ \mathcal{U}'_{k,1} \end{Vmatrix} = \mathsf{R}(-\theta_1,\phi_1)^T \begin{Vmatrix} \mathcal{U}_{k,0} \\ \mathcal{U}_{k,1} \end{Vmatrix}. \tag{23}$$

By choosing proper values for parameters θ_1 and ϕ_1, we can set a single element of the first column $\mathcal{U}'_{d-1,0}$ to 0:

$$\cos\left(\frac{\theta_1}{2}\right)\mathcal{U}_{d-1,0} + \iota e^{\iota\phi_1}\sin\left(\frac{\theta_1}{2}\right)\mathcal{U}_{d-1,1} = 0, \tag{24}$$

$$\theta_1 = 2\arctan\left(\frac{|\mathcal{U}_{d-1,0}|}{|\mathcal{U}_{d-1,1}|}\right), \tag{25}$$

$$\phi_1 = \frac{\pi}{2} + \arg(\mathcal{U}_{d-1,0}) - \arg(\mathcal{U}_{d-1,1}). \tag{26}$$

In the next steps, we apply operations $\mathsf{R}^{n-1,n}$ in order to eliminate element $\mathcal{U}_{d-1,n-1}$ using element $\mathcal{U}_{d-1,n}$. After $d{-}1$ steps, all elements in a row $d{-}1$ are eliminated. Due to the unitary property, all elements in a column $d{-}1$ are also eliminated and the diagonal element has absolute value 1. With the phase gate $\mathsf{P}^{d-1}(\alpha_{d-1} = \arg(\mathcal{U}_{d-1,d-1}))$ we obtain the following decomposition:

$$\begin{aligned} \mathsf{U}_d = &\begin{Vmatrix} \mathsf{U}_{d-1} & 0 \\ 0 & 1 \end{Vmatrix} \circ \\ &\circ \mathsf{P}^{d-1}(\alpha_{d-1}) \circ \mathsf{R}^{d-2,d-1}(\theta_{d-1},\phi_{d-1}) \circ \\ &\circ \cdots \circ \\ &\circ \mathsf{R}^{1,2}(\theta_2,\phi_2) \circ \mathsf{R}^{0,1}(\theta_1,\phi_1). \end{aligned} \tag{27}$$

After eliminating the row, we can repeat the same algorithm on the lesser $(d{-}1){\times}(d{-}1)$ unitary matrix U_{d-1}. The full row elimination scheme for superconducting qudit is depicted in Figure 1a.

If we want to implement the similar algorithm on the trapped-ion quantum computer [86] with $d{=}4$ levels, we

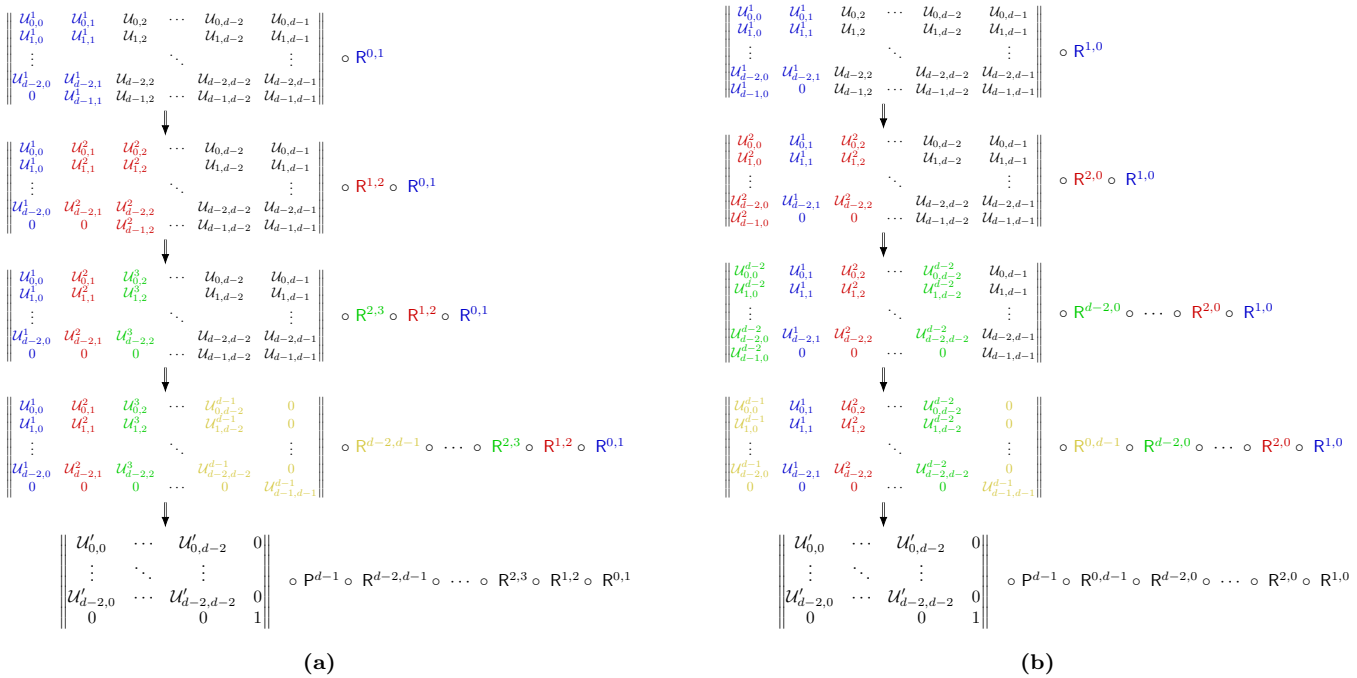


Figure 1. Row elimination schemes for qudit unitary matrix with given allowed transitions. **(a)**, Decomposition for superconducting qudit with allowed transitions $\mathsf{R}^{n,n\pm1}$. **(b)**, Decomposition for trapped-ion qudit with allowed transitions $\mathsf{R}^{0,n}$ and $\mathsf{R}^{n,0}$.

face the following issue. Consider the device operating with transitions 01, 02, 03, whereas the given decomposition scheme operates with 01, 12, 23. The naive approach requires using level-swap operations that transform any transition R^{ij} into the allowed transition R^{0i}:

$$\begin{gathered}\mathsf{R}^{ij}(\theta,\phi) = \mathsf{S}^{0i} \circ \mathsf{R}^{0j}(\theta,\phi) \circ \mathsf{S}^{i0}, \\ \text{or} \\ \mathsf{R}^{ij}(\theta,\phi) = \mathsf{S}^{0j} \circ \mathsf{R}^{i0}(\theta,\phi) \circ \mathsf{S}^{j0},\end{gathered} \tag{28}$$

$$\mathsf{S}^{ij} = \mathsf{R}^{ij}\left(\pi, \frac{\pi}{2}\right) \equiv \begin{Vmatrix} 0 & -1 \\ 1 & 0 \end{Vmatrix}. \tag{29}$$

However, using this strategy for arbitrary unitary, we obtain 12 trapped-ion native transitions, instead of 6 as for superconducting qudit. The best we can get using level-swaps is 8 native transitions. This number will grow with d more rapidly, than $\leq d(d-1)/2$ estimation for a superconducting qudit.

Hence, we should deduce the most optimal decomposition scheme for trapped-ion qudits, which typically have a star-like level transition graph. In general, we consider d levels trapped-ion qudit with transitions $\mathsf{R}^{0,n}$ or $\mathsf{R}^{n,0}$, where $0 < n < d$. The main idea of our algorithm is to eliminate all row elements $\mathcal{U}_{d-1,n}$ for $0 < n < d$ using the zeroeth element $\mathcal{U}_{d-1,0}$. Then, apply $\mathsf{R}^{0,d-1}$ to eliminate the zeroeth element as well. The full row elimination scheme for trapped-ion qudit unitary is depicted in Figure 1b.

Row elimination for $d\times d$ unitary requires at most $(d-1)\times$ R and $1\times$ P gates. In total, after decomposing all lesser matrices, we obtain $d(d-1)/2\times$ R and $d\times$ P gates in unitary decomposition. Though some θ_n parameters could be equal 0 (if eliminated elements are already $= 0$), hence these operations can be omitted in the hardware execution. Also, on both considered platforms, phase gates P can be performed virtually, so their amount in the decomposition does not affect execution time as much as transition gates R.

V. UNITARY DECOMPOSITION FOR QUDIT WITH ARBITRARY SELECTION RULES

Our goal is to generalize the considered decomposition schemes to the case of arbitrary selection rules. A specific decomposition pattern could be spotted in the earlier proposed schemes. We apply $d-1$ operations in order to eliminate all non-diagonal terms. Each R operation requires an additional non-zero row element to perform elimination. This pattern to eliminate a row r could be described in a more formal way:

- On each step we apply $\mathsf{R}^{z_i p_i}$ gate, where z_i is the index of eliminated element in row and p_i is the index of pivot element that used to eliminate z_i;
- z_i, p_i must be allowed transition;
- Indices z_i are non-repetitive and take all levels $\{0, 1, \ldots, r-1\}$;

- Indices p_i could be arbitrary levels, except those that have already been eliminated:

$$\forall i < r, j < i \Rightarrow p_i \neq z_j. \tag{30}$$

In fact, we are not restricted to eliminating only the last row in unitary matrix. For a given sequence for eliminated rows $\{r_n\}_{n=0}^{d-1}$ (elimination from top r_{d-1} to bottom r_0), we just have to modify 3rd rule in the pattern. On k^{th} step, where row r_k is eliminated, we should run through indices z_i that are non-repetitive and take all levels $\{r_0, r_1, \ldots, r_{k-1}\}$;

In total, the decomposition scheme consists of indices:

- r_k – the order of eliminated rows;
- z_{ki} – the order of eliminated elements on k^{th} step;
- p_{ki} – pivot elements on k^{th} step.

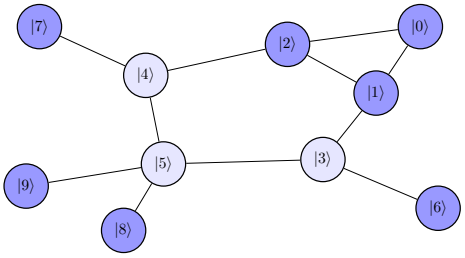
Using defined rules, we could produce a decomposition scheme for all arbitrary selection rules. The existence of such decomposition is not obvious yet but will be proved during the rest of the paper. Though indices for some selection rules, e.g. for superconducting qudit and trapped-ion qudit, are already obtained in the previous section and depicted in Table I.

Supperconducting qudit d=4		$r_3 = 3$ $z_3 = \{0,1,2\}$ $p_3 = \{1,2,3\}$ $r_2 = 2$ $z_2 = \{0,1\}$ $p_2 = \{1,2\}$ $r_1 = 1$ $z_1 = \{0\}$ $p_1 = \{1\}$
$^{171}\mathbf{Yb}^{+}$ **trapped-ion qudit** d=4		$r_3 = 3$ $z_3 = \{1,2,0\}$ $p_3 = \{0,0,3\}$ $r_2 = 2$ $z_2 = \{1,0\}$ $p_2 = \{0,2\}$ $r_1 = 1$ $z_1 = \{0\}$ $p_1 = \{1\}$
$^{171}\mathbf{Yb}^{+}$ **trapped-ion qudit** d=5		$r_4 = 1$ $z_4 = \{0,4,3,2\}$ $p_4 = \{2,1,1,1\}$ $r_3 = 4$ $z_3 = \{3,2,0\}$ $p_3 = \{0,0,4\}$ $r_2 = 3$ $z_2 = \{2,0\}$ $p_2 = \{0,3\}$ $r_1 = 2$ $z_1 = \{0\}$ $p_1 = \{2\}$ $r_4 = 4$ $z_4 = \{3,2,1,0\}$ $p_4 = \{0,0,4,4\}$ $r_3 = 3$ $z_3 = \{2,1,0\}$ $p_3 = \{0,3,3\}$ $r_2 = 1$ $z_2 = \{0,2\}$ $p_2 = \{2,1\}$ $r_1 = 2$ $z_1 = \{0\}$ $p_1 = \{2\}$

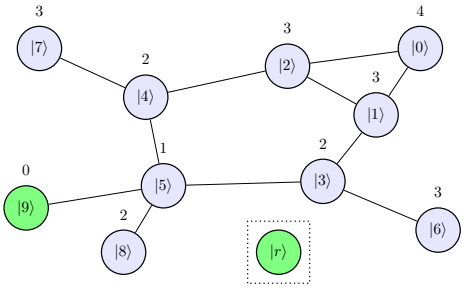
Table I. Decomposition scheme indices for several qudit platforms: superconducting with 4 levels and trapped-ion with 4 and 5 levels.

We note that these rules are not sufficient to get a unique decomposition scheme. For example, in the trapped-ion case we can eliminate first $k-1$ elements in any order and still get the row eliminated. Also, trapped-ion qudit with 5 levels can utilize two different schemes with different row orders. This degree of freedom can be useful, since we are allowed to select preferable sequence of operations according to physical heuristics of levels and transitions of the given qudit platform.

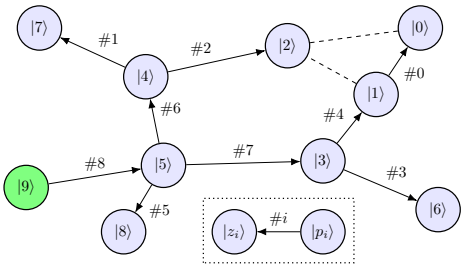
An arbitrary single-qudit system could be described as a connected undirected graph $\mathcal{G}$, where nodes $\mathcal{N}$ represent qudit levels and edges $\mathcal{E}$ represent allowed transitions. To determine decomposition indices for a given graph $\mathcal{G}$, we propose the following algorithm:



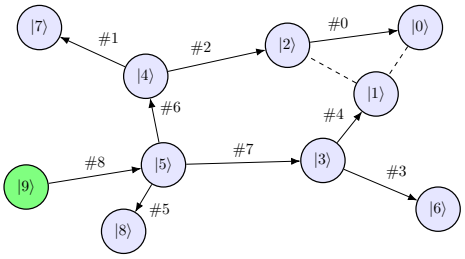
(a) Select $|r\rangle$ candidates that can be removed without breaking connectivity of the graph.



(b) Split the transition graph into sets $\{F_l\}$ using breadth-first search. $\forall l, |n\rangle \in F_l \rightarrow \text{dist}(|r\rangle, |n\rangle) = l$.



(c) Add indices $|z_i\rangle$ and $|p_i\rangle$ into the decomposition scheme for row $|r\rangle$.



(d) Degree of freedom is presented in this algorithm. Qudit physical properties could be used to choose the best transition sequence.

Figure 2. Algorithm that determines indices for qudit decomposition scheme. It utilizes the graph structure of selection rules and the breadth-first search algorithm.

- On the step k (runs from $d-1$ to 1), $\mathcal{N}$ has $k+1$ nodes/levels. We select levels that, if removed, do not break connectivity of the graph. One of these levels can be chosen as $|r_k\rangle$ to eliminate the row r_k of qudit unitary matrix (Figure 2a). Here we can use physical heuristics to eliminate less stable levels earlier;
- Each level $|i\rangle \in \mathcal{G}$ is assigned the shortest dis-

tance from the level $|r_k\rangle$. This could be done using breadth-first search, which has time complexity $O(|\mathcal{N}| + |\mathcal{E}|)$. We can split all levels into sets $\{F_l\}$, where F_l contains levels at distance l from $|r_k\rangle$ (Figure 2b);

- Let L be the maximum distance from $|r_k\rangle$ in the $\mathcal{G}$. Each level $|z_{L,i}\rangle \in F_L$ is connected by a transition to some element $|p_{L,i}\rangle \in F_{L-1}$. We add these indices into the decomposition scheme, then repeat for all F_l in descending order of l (Figure 2c). There could be several levels $\in F_{L-1}$ connected to $|z_{L,i}\rangle$ (Figure 2d). Using physical properties of a qudit, we can select the least noisy transition to reduce average error of hardware execution.

- Continue the algorithm on the step $k-1$ for the transition graph $\mathcal{G}' = \mathcal{G}/\{|r_k\rangle\}$ until $k = 0$.

Produced decomposition indices satisfy rules required for correct row elimination:

- Indices $z_{l,i}$ are taken from each $\{F_l\}_{l=1}^{L}$. These are non-intersecting sets, so $\{|z_i\rangle\} = \mathcal{G}/\{|r_k\rangle\}$, i.e. all elements are eliminated except diagonal one;

- Indices $p_{l,i}$ do not intersect already eliminated elements, since they have distance $> l$.

Note that on each step transition graph remains connected. This property is crucial for correct unitary decomposition. If we have a graph with 2 or more disconnected components, then we can only obtain a non-zero element per component. Each of these non-zero elements could not be eliminated since they are not connected via any R path.

The similar graph algorithm for the transition graph of ^{86}Rb atom qudit was mentioned in [83]. Though it describes the general technique, we propose the algorithm that is able to minimize physical execution error and reduce the number of transitions for sparse unitaries.

The *static* decomposition scheme is obtained once for a given qudit platform. Then, this scheme can be used for any qudit unitary matrix that should be executed on a given qudit. However, for some sparse matrices with zero non-diagonal elements, better optimized decomposition may be performed using *adaptive* decomposition scheme (Figure 3). Adaptive decomposition runs through the same step as static decomposition for each unitary matrix. The difference is that a level is excluded from the graph during decomposition if it corresponds to a zero element and does not break graph connectivity. Also, the order of eliminated rows is determined by the matrix sparsity pattern: rows with fewer non-zero elements are eliminated earlier. To sum up, *adaptive* decomposition scheme is only valid for a given matrix and contains the least amount of operations, whereas *static* decomposition scheme can be applied to an arbitrary matrix.

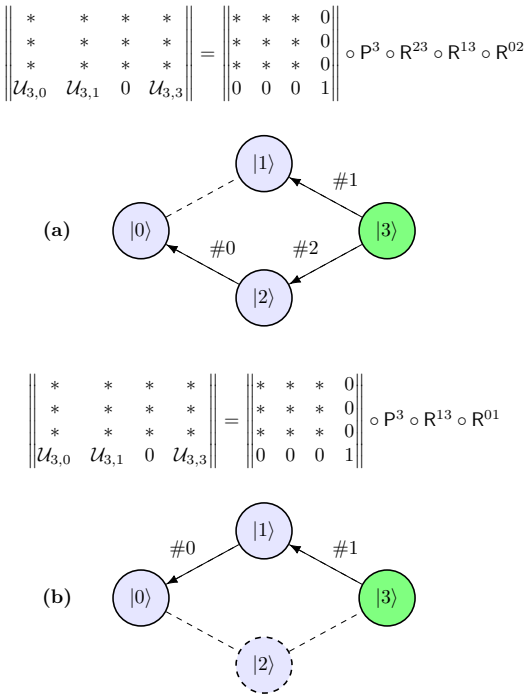


Figure 3. Decomposition of a matrix with zero non-diagonal elements: **(a)** *Static* and **(b)** *Adaptive* schemes.

VI. COMPARISON WITH EXISTING DECOMPOSITION METHODS

To show the advantage of the developed algorithm, we provide the comparison with other approaches that provide the functionality of decomposing arbitrary qudit unitary operations into the sequence of native pulses. All of them provide Python packages that help execute and compare results on a $d \times d$ Python matrix of the type `np.ndarray`. We compare existing methods with our developed decompositions in Tables II and III.

The first approach is *BQSKIT* framework that provides a generic method for numerical synthesis of unitary operations. It supports both single- or multi- qubit/qudit unitary synthesis. There exist several methods in *BQSKIT* that allow one to decompose qudit unitary operations. First is *QSearchPass* method that constructs the target unitary matrix numerically using the provided gate set. We can obtain desired decomposition by declaring custom parametric qudit gates, which correspond to allowed transition R^{ij} and phase shift P^k operations. Notably, this method is not deterministic, so we run each experiment 20 times and take the median value for decomposition length. The second method is called *QSweepPass* [87], which is not part of *BQSKIT* but relies on its transpilation workflow. It uses numerical and analytical results to construct qudit unitary operation in terms of two-level operations. However, it does not support arbitrary selection rules and only targets superconducting transition topology.

Method	X_d^{+1}			X_d^{-1}			H_d			U_d			Two-qubit gate					
	d=4	d=5	d=6	d=4	d=5	d=6	d=4	d=5	d=6	d=4	d=5	d=6	RXX	RZZ	CZ	CX	CH	SWAP
QSearchPass	3	4	5	3	4	5	7	11	17	7	12	16	6	0	0	3	3	1
QSweepPass	5	6	7	3	4	5	7	11	16	7	11	16	6	2	2	5	5	3
LocQRPass	3	4	5	3	4	5	6	10	15	6	10	15	6	0	0	3	3	1
LocAdaPass	3	4	5	3	4	5	6	**TO**	**TO**	7	**TO**	**TO**	4	0	0	2	2	1
TAQR	3	4	5	3	4	5	6	10	15	6	10	15	6	0	0	3	3	1
TAQR Adaptive	3	4	5	3	4	5	6	10	15	6	10	15	6	0	0	3	3	1

(a) Line transition graph.

Method	X_d^{+1}			X_d^{-1}			H_d			U_d			Two-qubit gate					
	d=4	d=5	d=6	d=4	d=5	d=6	d=4	d=5	d=6	d=4	d=5	d=6	RXX	RZZ	CZ	CX	CH	SWAP
QSearchPass	3	4	5	3	4	5	6	12	16	7	11	16	4	0	0	3	3	3
LocQRPass	7	10	13	7	10	13	16	28	43	16	28	43	16	0	0	9	9	3
LocAdaPass	4	6	7	4	5	6	8	15	24	8	15	24	3	0	0	2	2	2
TAQR	5	6	9	3	4	5	5	10	15	6	10	15	4	0	0	3	3	3
TAQR Adaptive	3	4	5	3	4	5	6	10	15	6	10	15	4	0	0	3	3	3

(b) Star transition graph.

Method	X_d^{+1}			X_d^{-1}			H_d			U_d			Two-qubit gate					
	d=4	d=5	d=6	d=4	d=5	d=6	d=4	d=5	d=6	d=4	d=5	d=6	RXX	RZZ	CZ	CX	CH	SWAP
QSearchPass	3	4	6	3	4	5	7	12	13	7	11	16	2	0	0	1	1	1
LocQRPass	7	10	13	7	10	13	14	26	41	14	26	41	14	0	0	7	7	1
LocAdaPass	4	6	7	4	5	6	9	14	21	9	14	21	2	0	0	1	1	1
TAQR	3	6	7	3	4	5	6	10	14	6	10	15	2	0	0	1	1	1
TAQR Adaptive	3	4	5	3	4	5	6	10	14	6	10	15	2	0	0	1	1	1

(c) Bipartite transition graph.

Table II. Comparison of decompositions produced by given methods in terms of transition count. **TO** means that the execution exceeded the 1-minute timeout.

The second approach is provided by *MQT.Qudits* [89] framework for mixed-dimensional quantum computing, which is part of the *Munich Quantum Toolkit*. It is possible to pass qudit transition graph with corresponding weights into the *LocQRPass* method and obtain the sequence of allowed gates as decomposition. This framework also supports adaptive gate transpilation mode via the *LocAdaPass* method that can reduce the length of the decomposition.

In the Table II, our method is called *Transition-Aware QR decomposition*, or *TAQR* for simplicity, for static decomposition and *TAQR Adaptive* for adaptive decomposition. In our comparison we are targeting 3 types of selection rules given as an undirected graph for different qudit platforms. First considered platform has *line transition graph* (Table IIa), where only transitions on neighboring levels are allowed. Superconducting qudits and photonic circuits have this type of selection rules. We can extend line transition graph to any number of levels d. Second platform is trapped-ion qudits. Overall, this type of qudit can have any arbitrary transitions selected by atomic physics rules. Though, particular trapped-ion devices provide *star transition graph* (Table IIb) or *bipartite transition graph* (Table IIc) of the size d. *Star transition graph* is the special case of *bipartite transition graph* with partition p=1. *Bipartite transition graph* with partition p is the graph with two disjoint subsets with sizes p and $d-p$, that is, every node in the first subset is connected via an edge to each node in the second subset. We use bipartite graph with p=2 in our experiments.

Each method produces decomposition for each given platform for one of the following single-qudit operations:

- State number increment/decrement gate X_d^{+1} and X_d^{-1}, which have the form as in eq. (4);
- Quantum Fourier Transform with the size d that is denoted H_d in eq. (7);
- Uniformly distributed (using Haar measure procedure) unitary matrix U_d. Since the matrix is random, we run each decomposition on 100 generated matrices, then take median value for decomposition length;
- Two-qubit matrix $\in U(4)$ that represents the action on two qubits embedded into the qudit with d=4. Chosen gates are: RXX $(\pi/2)$, RZZ $(\pi/2)$, CZ, CX, CH, SWAP.

As a result, *Transition-Aware QR* decomposition performs as good as any other qudit synthesis framework for superconducting or photonic platforms. For the majority of qudit unitary operations, it produces the decomposition with the same length in a reasonable time (refer Table III in the Appendix for execution time comparison). Thus, it can become a drop-in replacement for the qudit transpiler pass for single-qudit operations for this type of platform.

Next, our method provides the shortest decomposition for trapped-ion platforms (with any arbitrary selection rules). Our result outperforms the average numerical decomposition using *QSearch* method that produces random decomposition on each invocation and takes significantly greater execution time. It also outperforms *MQT.Qudits* compilation passes, which produce more transitions and level-swaps for the same unitary. For generic unitary, our method is guaranteed to produce $d(d-1)/2$ transitions, which is the theoretical upper bound (the number of lower non-diagonal elements in unitary).

And lastly, we have shown the potential benefits of using either *static* or *adaptive* decomposition. Static decomposition is computed once for each platform, which reduces decomposition time during transpilation process. On the other hand, adaptive decomposition considers the sparsity pattern of the given unitary matrix and produces a sequence with fewer transitions for several cases, e.g. for X_d operation.

Notably, while *LocAdaPass* produces shorter decomposition for two-qubit gates embedded into the ququart, it also permutes levels virtually. This fact makes sense for whole qudit circuit optimization. Though, we should permute rows virtually or using level-swaps gates to obtain proper unitary. Potentially, our method can also benefit from virtual level-swaps when it is a part of the whole qudit circuit workflow.

VII. CONCLUSION

In this paper we have proposed the method that converts d-level qudit unitary operation into the sequence of the allowed transitions between qudit levels. It considers arbitrary selection rules of the qudit system and produces the optimal decomposition sequence in terms of transition amount. Moreover, we show that the decomposition consists of no more than $d(d-1)/2$ transitions, which is the theoretical upper bound for an arbitrary unitary matrix $d\times d$.

Decomposition could be done once for each qudit of interest before the transpilation, then applied to any unitary matrix to obtain parameters of transitions. This is the essence of the *static* decomposition. On the other hand, we have proposed the *adaptive* modification that utilizes the sparsity pattern of the matrix and the transition graph simultaneously. Such treatment could reduce the transition amount even greater for some operations, such as level permutation.

As we have shown, the crucial advantage of our approach is that it does not distinguish between qudit platforms and produces similar optimal decomposition for any transition graph. Most trapped-ion qudit platforms could benefit from utilizing this approach due to complex selection rules on operated levels.

ACKNOWLEDGMENTS

The work of A.S.N. and D.A.D. was supported by RSF Grant No. 24-71-00084 (developing methods of quantum algorithms transpilation for the acceleration of their execution; Sections IV, V and VI). The work of E.O.K. and A.K.F. was supported by the Priority 2030 program at the NIST "MISIS" under the project K1-2022-027 (exploring existing qudit gates; Sections II and III).

[1] G. Brassard, I. Chuang, S. Lloyd, and C. Monroe, Quantum computing, Proceedings of the National Academy of Sciences **95**, 11032 (1998).
[2] T. D. Ladd, F. Jelezko, R. Laflamme, Y. Nakamura, C. Monroe, and J. L. O'Brien, Quantum computers, Nature **464**, 45 (2010).
[3] A. K. Fedorov, N. Gisin, S. M. Beloussov, and A. I. Lvovsky, Quantum computing at the quantum advantage threshold: a down-to-business review (2022).
[4] S. Lloyd, Universal quantum simulators, Science **273**, 1073 (1996).
[5] E. Farhi, J. Goldstone, and S. Gutmann, A quantum approximate optimization algorithm (2014), arXiv:1411.4028 [quant-ph].
[6] P. Shor, Algorithms for quantum computation: discrete logarithms and factoring, in *Proceedings 35th Annual Symposium on Foundations of Computer Science* (1994) pp. 124–134.
[7] M. Cerezo, A. Arrasmith, R. Babbush, S. C. Benjamin, S. Endo, K. Fujii, J. R. McClean, K. Mitarai, X. Yuan, L. Cincio, and P. J. Coles, Variational quantum algorithms, Nature Reviews Physics **3**, 625 (2021).
[8] K. Bharti, A. Cervera-Lierta, T. H. Kyaw, T. Haug, S. Alperin-Lea, A. Anand, M. Degroote, H. Heimonen, J. S. Kottmann, T. Menke, W.-K. Mok, S. Sim, L.-C. Kwek, and A. Aspuru-Guzik, Noisy intermediate-scale quantum (nisq) algorithms (2021), arXiv:2101.08448 [quant-ph].
[9] S. Ebadi, T. T. Wang, H. Levine, A. Keesling, G. Semeghini, A. Omran, D. Bluvstein, R. Samajdar, H. Pichler, W. W. Ho, S. Choi, S. Sachdev, M. Greiner, V. Vuletić, and M. D. Lukin, Quantum phases of matter on a 256-atom programmable quantum simulator, Nature **595**, 227 (2021).
[10] F. Arute, K. Arya, R. Babbush, D. Bacon, J. C. Bardin, R. Barends, R. Biswas, S. Boixo, F. G. S. L. Brandao, D. A. Buell, B. Burkett, Y. Chen, Z. Chen,

B. Chiaro, R. Collins, W. Courtney, A. Dunsworth, E. Farhi, B. Foxen, A. Fowler, C. Gidney, M. Giustina, R. Graff, K. Guerin, S. Habegger, M. P. Harrigan, M. J. Hartmann, A. Ho, M. Hoffmann, T. Huang, T. S. Humble, S. V. Isakov, E. Jeffrey, Z. Jiang, D. Kafri, K. Kechedzhi, J. Kelly, P. V. Klimov, S. Knysh, A. Korotkov, F. Kostritsa, D. Landhuis, M. Lindmark, E. Lucero, D. Lyakh, S. Mandrà, J. R. McClean, M. McEwen, A. Megrant, X. Mi, K. Michielsen, M. Mohseni, J. Mutus, O. Naaman, M. Neeley, C. Neill, M. Y. Niu, E. Ostby, A. Petukhov, J. C. Platt, C. Quintana, E. G. Rieffel, P. Roushan, N. C. Rubin, D. Sank, K. J. Satzinger, V. Smelyanskiy, K. J. Sung, M. D. Trevithick, A. Vainsencher, B. Villalonga, T. White, Z. J. Yao, P. Yeh, A. Zalcman, H. Neven, and J. M. Martinis, Quantum supremacy using a programmable superconducting processor, Nature **574**, 505 (2019).

[11] Y. Wu, W.-S. Bao, S. Cao, F. Chen, M.-C. Chen, X. Chen, T.-H. Chung, H. Deng, Y. Du, D. Fan, M. Gong, C. Guo, C. Guo, S. Guo, L. Han, L. Hong, H.-L. Huang, Y.-H. Huo, L. Li, N. Li, S. Li, Y. Li, F. Liang, C. Lin, J. Lin, H. Qian, D. Qiao, H. Rong, H. Su, L. Sun, L. Wang, S. Wang, D. Wu, Y. Xu, K. Yan, W. Yang, Y. Yang, Y. Ye, J. Yin, C. Ying, J. Yu, C. Zha, C. Zhang, H. Zhang, K. Zhang, Y. Zhang, H. Zhao, Y. Zhao, L. Zhou, Q. Zhu, C.-Y. Lu, C.-Z. Peng, X. Zhu, and J.-W. Pan, Strong quantum computational advantage using a superconducting quantum processor, Phys. Rev. Lett. **127**, 180501 (2021).

[12] X. Xue, M. Russ, N. Samkharadze, B. Undseth, A. Sammak, G. Scappucci, and L. M. K. Vandersypen, Quantum logic with spin qubits crossing the surface code threshold, Nature **601**, 343 (2022).

[13] M. T. Madzik, S. Asaad, A. Youssry, B. Joecker, K. M. Rudinger, E. Nielsen, K. C. Young, T. J. Proctor, A. D. Baczewski, A. Laucht, V. Schmitt, F. E. Hudson, K. M. Itoh, A. M. Jakob, B. C. Johnson, D. N. Jamieson, A. S. Dzurak, C. Ferrie, R. Blume-Kohout, and A. Morello, Precision tomography of a three-qubit donor quantum processor in silicon, Nature **601**, 348 (2022).

[14] A. Noiri, K. Takeda, T. Nakajima, T. Kobayashi, A. Sammak, G. Scappucci, and S. Tarucha, Fast universal quantum gate above the fault-tolerance threshold in silicon, Nature **601**, 338 (2022).

[15] H.-S. Zhong, H. Wang, Y.-H. Deng, M.-C. Chen, L.-C. Peng, Y.-H. Luo, J. Qin, D. Wu, X. Ding, Y. Hu, P. Hu, X.-Y. Yang, W.-J. Zhang, H. Li, Y. Li, X. Jiang, L. Gan, G. Yang, L. You, Z. Wang, L. Li, N.-L. Liu, C.-Y. Lu, and J.-W. Pan, Quantum computational advantage using photons, Science **370**, 1460 (2020).

[16] L. S. Madsen, F. Laudenbach, M. F. Askarani, F. Rortais, T. Vincent, J. F. F. Bulmer, F. M. Miatto, L. Neuhaus, L. G. Helt, M. J. Collins, A. E. Lita, T. Gerrits, S. W. Nam, V. D. Vaidya, M. Menotti, I. Dhand, Z. Vernon, N. Quesada, and J. Lavoie, Quantum computational advantage with a programmable photonic processor, Nature **606**, 75 (2022).

[17] P. Scholl, M. Schuler, H. J. Williams, A. A. Eberharter, D. Barredo, K.-N. Schymik, V. Lienhard, L.-P. Henry, T. C. Lang, T. Lahaye, A. M. Läuchli, and A. Browaeys, Quantum simulation of 2d antiferromagnets with hundreds of rydberg atoms, Nature **595**, 233 (2021).

[18] L. Henriet, L. Beguin, A. Signoles, T. Lahaye, A. Browaeys, G.-O. Reymond, and C. Jurczak, Quantum computing with neutral atoms, Quantum **4**, 327 (2020).

[19] T. M. Graham, Y. Song, J. Scott, C. Poole, L. Phuttitarn, K. Jooya, P. Eichler, X. Jiang, A. Marra, B. Grinkemeyer, M. Kwon, M. Ebert, J. Cherek, M. T. Lichtman, M. Gillette, J. Gilbert, D. Bowman, T. Ballance, C. Campbell, E. D. Dahl, O. Crawford, N. S. Blunt, B. Rogers, T. Noel, and M. Saffman, Multi-qubit entanglement and algorithms on a neutral-atom quantum computer, Nature **604**, 457 (2022).

[20] J. Zhang, G. Pagano, P. W. Hess, A. Kyprianidis, P. Becker, H. Kaplan, A. V. Gorshkov, Z. X. Gong, and C. Monroe, Observation of a many-body dynamical phase transition with a 53-qubit quantum simulator, Nature **551**, 601 (2017).

[21] R. Blatt and C. F. Roos, Quantum simulations with trapped ions, Nature Physics **8**, 277 (2012).

[22] C. Hempel, C. Maier, J. Romero, J. McClean, T. Monz, H. Shen, P. Jurcevic, B. P. Lanyon, P. Love, R. Babbush, A. Aspuru-Guzik, R. Blatt, and C. F. Roos, Quantum chemistry calculations on a trapped-ion quantum simulator, Phys. Rev. X **8**, 031022 (2018).

[23] E. O. Kiktenko, A. S. Nikolaeva, and A. K. Fedorov, Colloquium: Qudits for decomposing multiqubit gates and realizing quantum algorithms, Rev. Mod. Phys. **97**, 021003 (2025).

[24] E. Farhi and S. Gutmann, Analog analogue of a digital quantum computation, Phys. Rev. A **57**, 2403 (1998).

[25] A. R. Kessel' and V. L. Ermakov, Multiqubit spin, Journal of Experimental and Theoretical Physics Letters **70**, 61 (1999).

[26] A. R. Kessel' and V. L. Ermakov, Physical implementation of three-qubit gates on a separate quantum particle, Journal of Experimental and Theoretical Physics Letters **71**, 307 (2000).

[27] A. R. Kessel and N. M. Yakovleva, Implementation schemes in nmr of quantum processors and the deutsch-jozsa algorithm by using virtual spin representation, Phys. Rev. A **66**, 062322 (2002).

[28] A. Muthukrishnan and C. R. Stroud, Multivalued logic gates for quantum computation, Phys. Rev. A **62**, 052309 (2000).

[29] M. A. Nielsen, M. J. Bremner, J. L. Dodd, A. M. Childs, and C. M. Dawson, Universal simulation of hamiltonian dynamics for quantum systems with finite-dimensional state spaces, Phys. Rev. A **66**, 022317 (2002).

[30] X. Wang, B. C. Sanders, and D. W. Berry, Entangling power and operator entanglement in qudit systems, Phys. Rev. A **67**, 042323 (2003).

[31] A. B. Klimov, R. Guzmán, J. C. Retamal, and C. Saavedra, Qutrit quantum computer with trapped ions, Phys. Rev. A **67**, 062313 (2003).

[32] E. Bagan, M. Baig, and R. Muñoz Tapia, Minimal measurements of the gate fidelity of a qudit map, Phys. Rev. A **67**, 014303 (2003).

[33] A. Y. Vlasov, Algebra of quantum computations with higher dimensional systems, in *First International Symposium on Quantum Informatics*, Vol. 5128, edited by Y. I. Ozhigov, International Society for Optics and Photonics (SPIE, 2003) pp. 29 – 36.

[34] A. D. Greentree, S. G. Schirmer, F. Green, L. C. L. Hollenberg, A. R. Hamilton, and R. G. Clark, Maximizing the hilbert space for a finite number of distinguishable quantum states, Phys. Rev. Lett. **92**, 097901 (2004).

[35] D. P. O'Leary, G. K. Brennen, and S. S. Bullock, Parallelism for quantum computation with qudits, Phys. Rev. A **74**, 032334 (2006).

[36] T. C. Ralph, K. J. Resch, and A. Gilchrist, Efficient toffoli gates using qudits, Phys. Rev. A **75**, 022313 (2007).

[37] B. P. Lanyon, T. J. Weinhold, N. K. Langford, J. L. O'Brien, K. J. Resch, A. Gilchrist, and A. G. White, Manipulating biphotonic qutrits, Phys. Rev. Lett. **100**, 060504 (2008).

[38] R. Ionicioiu, T. P. Spiller, and W. J. Munro, Generalized toffoli gates using qudit catalysis, Phys. Rev. A **80**, 012312 (2009).

[39] M. Neeley, M. Ansmann, R. C. Bialczak, M. Hofheinz, E. Lucero, A. D. O'Connell, D. Sank, H. Wang, J. Wenner, A. N. Cleland, M. R. Geller, and J. M. Martinis, Emulation of a quantum spin with a superconducting phase qudit, Science **325**, 722 (2009).

[40] B. P. Lanyon, M. Barbieri, M. P. Almeida, T. Jennewein, T. C. Ralph, K. J. Resch, G. J. Pryde, J. L. O'Brien, A. Gilchrist, and A. G. White, Simplifying quantum logic using higher-dimensional hilbert spaces, Nature Physics **5**, 134 (2009).

[41] S. S. Ivanov, H. S. Tonchev, and N. V. Vitanov, Time-efficient implementation of quantum search with qudits, Phys. Rev. A **85**, 062321 (2012).

[42] B. E. Mischuck, S. T. Merkel, and I. H. Deutsch, Control of inhomogeneous atomic ensembles of hyperfine qudits, Phys. Rev. A **85**, 022302 (2012).

[43] A. Fedorov, L. Steffen, M. Baur, M. P. da Silva, and A. Wallraff, Implementation of a toffoli gate with superconducting circuits, Nature **481**, 170 (2012).

[44] B. Li, Z.-H. Yu, and S.-M. Fei, Geometry of quantum computation with qutrits, Scientific Reports **3**, 2594 (2013).

[45] E. Svetitsky, H. Suchowski, R. Resh, Y. Shalibo, J. M. Martinis, and N. Katz, Hidden two-qubit dynamics of a four-level josephson circuit, Nature Communications **5**, 5617 (2014).

[46] M. J. Peterer, S. J. Bader, X. Jin, F. Yan, A. Kamal, T. J. Gudmundsen, P. J. Leek, T. P. Orlando, W. D. Oliver, and S. Gustavsson, Coherence and decay of higher energy levels of a superconducting transmon qubit, Phys. Rev. Lett. **114**, 010501 (2015).

[47] E. O. Kiktenko, A. K. Fedorov, O. V. Man'ko, and V. I. Man'ko, Multilevel superconducting circuits as two-qubit systems: Operations, state preparation, and entropic inequalities, Phys. Rev. A **91**, 042312 (2015).

[48] E. Kiktenko, A. Fedorov, A. Strakhov, and V. Man'ko, Single qudit realization of the deutsch algorithm using superconducting many-level quantum circuits, Physics Letters A **379**, 1409 (2015).

[49] J. Braumüller, J. Cramer, S. Schlör, H. Rotzinger, L. Radtke, A. Lukashenko, P. Yang, S. T. Skacel, S. Probst, M. Marthaler, L. Guo, A. V. Ustinov, and M. Weides, Multiphoton dressing of an anharmonic superconducting many-level quantum circuit, Phys. Rev. B **91**, 054523 (2015).

[50] C. Song, S.-L. Su, J.-L. Wu, D.-Y. Wang, X. Ji, and S. Zhang, Generation of tree-type three-dimensional entangled states via adiabatic passage, Phys. Rev. A **93**, 062321 (2016).

[51] A. Frydryszak, L. Jakóbczyk, and P. Ługiewicz, Determining quantum correlations in bipartite systems - from qubit to qutrit and beyond, Journal of Physics: Conference Series **804**, 012016 (2017).

[52] C. Godfrin, A. Ferhat, R. Ballou, S. Klyatskaya, M. Ruben, W. Wernsdorfer, and F. Balestro, Operating quantum states in single magnetic molecules: Implementation of grover's quantum algorithm, Phys. Rev. Lett. **119**, 187702 (2017).

[53] A. Bocharov, M. Roetteler, and K. M. Svore, Factoring with qutrits: Shor's algorithm on ternary and metaplectic quantum architectures, Phys. Rev. A **96**, 012306 (2017).

[54] M. Kues, C. Reimer, P. Roztocki, L. R. Cortés, S. Sciara, B. Wetzel, Y. Zhang, A. Cino, S. T. Chu, B. E. Little, D. J. Moss, L. Caspani, J. Azaña, and R. Morandotti, On-chip generation of high-dimensional entangled quantum states and their coherent control, Nature **546**, 622 (2017).

[55] P. Gokhale, J. M. Baker, C. Duckering, N. C. Brown, K. R. Brown, and F. T. Chong, Asymptotic improvements to quantum circuits via qutrits, in *Proceedings of the 46th International Symposium on Computer Architecture*, ISCA '19 (Association for Computing Machinery, New York, NY, USA, 2019) pp. 554–566.

[56] Y.-H. Luo, H.-S. Zhong, M. Erhard, X.-L. Wang, L.-C. Peng, M. Krenn, X. Jiang, L. Li, N.-L. Liu, C.-Y. Lu, A. Zeilinger, and J.-W. Pan, Quantum teleportation in high dimensions, Phys. Rev. Lett. **123**, 070505 (2019).

[57] P. J. Low, B. M. White, A. A. Cox, M. L. Day, and C. Senko, Practical trapped-ion protocols for universal qudit-based quantum computing, Phys. Rev. Research **2**, 033128 (2020).

[58] Z. Jin, W.-J. Gong, A.-D. Zhu, S. Zhang, Y. Qi, and S.-L. Su, Dissipative preparation of qutrit entanglement via periodically modulated rydberg double antiblockade, Opt. Express **29**, 10117 (2021).

[59] R. Sawant, J. A. Blackmore, P. D. Gregory, J. Mur-Petit, D. Jaksch, J. Aldegunde, J. M. Hutson, M. R. Tarbutt, and S. L. Cornish, Ultracold polar molecules as qudits, New Journal of Physics **22**, 013027 (2020).

[60] P. J. Low, B. M. White, A. A. Cox, M. L. Day, and C. Senko, Practical trapped-ion protocols for universal qudit-based quantum computing, Phys. Rev. Research **2**, 033128 (2020).

[61] A. Pavlidis and E. Floratos, Quantum-fourier-transform-based quantum arithmetic with qudits, Phys. Rev. A **103**, 032417 (2021).

[62] P. Rambow and M. Tian, Reduction of circuit depth by mapping qubit-based quantum gates to a qudit basis (2021).

[63] A. S. Nikolaeva, E. O. Kiktenko, and A. K. Fedorov, Efficient realization of quantum algorithms with qudits, EPJ Quantum Technology **11**, 1 (2024).

[64] A. S. Nikolaeva, E. O. Kiktenko, and A. K. Fedorov, Universal quantum computing with qubits embedded in trapped-ion qudits, Physical Review A **109**, 022615 (2024).

[65] D. A. Drozhzhin, A. S. Nikolaeva, E. O. Kiktenko, and A. K. Fedorov, Transpiling quantum assembly language circuits to a qudit form, Entropy **26**, 1129 (2024).

[66] A. Barenco, C. H. Bennett, R. Cleve, D. P. DiVincenzo, N. Margolus, P. Shor, T. Sleator, J. A. Smolin, and H. Weinfurter, Elementary gates for quantum computation, Phys. Rev. A **52**, 3457 (1995).

[67] W.-Q. Liu, H.-R. Wei, and L.-C. Kwek, Low-cost fredkin gate with auxiliary space, Phys. Rev. Applied **14**, 054057

(2020).
[68] J. M. Baker, C. Duckering, and F. T. Chong, Efficient quantum circuit decompositions via intermediate qudits, in *2020 IEEE 50th International Symposium on Multiple-Valued Logic (ISMVL)* (2020) pp. 303–308.
[69] E. O. Kiktenko, A. S. Nikolaeva, P. Xu, G. V. Shlyapnikov, and A. K. Fedorov, Scalable quantum computing with qudits on a graph, Phys. Rev. A **101**, 022304 (2020).
[70] W.-Q. Liu, H.-R. Wei, and L.-C. Kwek, Universal quantum multi-qubit entangling gates with auxiliary spaces, Advanced Quantum Technologies **5**, 2100136 (2022).
[71] W.-Q. Liu, H.-R. Wei, and L.-C. Kwek, Implementation of cnot and toffoli gates with higher-dimensional spaces (2021).
[72] A. Galda, M. Cubeddu, N. Kanazawa, P. Narang, and N. Earnest-Noble, Implementing a ternary decomposition of the toffoli gate on fixed-frequency transmon qutrits (2021), arXiv:2109.00558 [quant-ph].
[73] X. Gu, J. Allcock, S. An, and Y.-x. Liu, Efficient multi-qubit subspace rotations via topological quantum walks (2021).
[74] A. D. Hill, M. J. Hodson, N. Didier, and M. J. Reagor, Realization of arbitrary doubly-controlled quantum phase gates (2021).
[75] Y. Chi, J. Huang, Z. Zhang, J. Mao, Z. Zhou, X. Chen, C. Zhai, J. Bao, T. Dai, H. Yuan, M. Zhang, D. Dai, B. Tang, Y. Yang, Z. Li, Y. Ding, L. K. Oxenløwe, M. G. Thompson, J. L. O'Brien, Y. Li, Q. Gong, and J. Wang, A programmable qudit-based quantum processor, Nature Communications **13**, 1166 (2022).
[76] M. Ringbauer, M. Meth, L. Postler, R. Stricker, R. Blatt, P. Schindler, and T. Monz, A universal qudit quantum processor with trapped ions, Nature Physics **18**, 1053–1057 (2022).
[77] M. A. Aksenov, I. V. Zalivako, I. A. Semerikov, A. S. Borisenko, N. V. Semenin, P. L. Sidorov, A. K. Fedorov, K. Y. Khabarova, and N. N. Kolachevsky, Realizing quantum gates with optically addressable $^{171}Yb^{+}$ ion qudits, Phys. Rev. A **107**, 052612 (2023).
[78] E. Champion, Z. Wang, R. W. Parker, and M. S. Blok, Efficient control of a transmon qudit using effective spin-7/2 rotations, Physical Review X **15**, 10.1103/vbh4-lysv (2025).
[79] Z. Wang, R. W. Parker, E. Champion, and M. S. Blok, High-E_J/E_C transmon qudits with up to 12 levels, Phys. Rev. Appl. **23**, 034046 (2025).
[80] Y. Yu, Y. Chi, C. Zhai, J. Huang, Q. Gong, and J. Wang, Simulating hamiltonian dynamics in a programmable photonic quantum processor using linear combinations of unitary operations (2022), arXiv:2211.06723 [quant-ph].
[81] A. S. Nikolaeva, E. O. Kiktenko, and A. K. Fedorov, Decomposing the generalized toffoli gate with qutrits, Phys. Rev. A **105**, 032621 (2022).
[82] A. S. Nikolaeva, I. V. Zalivako, A. S. Borisenko, N. V. Semenin, K. P. Galstyan, A. E. Korolkov, E. O. Kiktenko, K. Y. Khabarova, I. A. Semerikov, A. K. Fedorov, and N. N. Kolachevsky, Scalable improvement of the generalized toffoli gate realization using trapped-ion-based qutrits (2024), arXiv:2407.07758 [quant-ph].
[83] G. Brennen, D. O'Leary, and S. Bullock, Criteria for exact qudit universality, Physical Review A **71**, 10.1103/physreva.71.052318 (2005).
[84] P. J. Low, B. White, and C. Senko, Control and readout of a 13-level trapped ion qudit, npj Quantum Information **11**, 85 (2025), arXiv:2306.03340 [quant-ph].
[85] P. J. Low, N. C. F. Zutt, G. A. Tathed, and C. Senko, Quantum logic operations and algorithms in a single 25-level atomic qudit (2025), arXiv:2507.15799 [quant-ph].
[86] I. V. Zalivako, A. S. Nikolaeva, A. S. Borisenko, A. E. Korolkov, P. L. Sidorov, K. P. Galstyan, N. V. Semenin, V. N. Smirnov, M. A. Aksenov, K. M. Makushin, E. O. Kiktenko, A. K. Fedorov, I. A. Semerikov, K. Y. Khabarova, and N. N. Kolachevsky, Towards multiqudit quantum processor based on a $^{171}Yb^{+}$ ion string: Realizing basic quantum algorithms, Quantum Reports **7**, 10.3390/quantum7020019 (2025).
[87] E. Younis and N. Goss, Qsweep: Pulse-optimal single-qudit synthesis (2023), arXiv:2312.09990 [quant-ph].
[88] W. R. Clements, P. C. Humphreys, B. J. Metcalf, W. S. Kolthammer, and I. A. Walmsley, Optimal design for universal multiport interferometers, Optica **3**, 1460 (2016).
[89] K. Mato, M. Ringbauer, L. Burgholzer, and R. Wille, Mqt qudits: A software framework for mixed-dimensional quantum computing (2024), arXiv:2410.02854 [quant-ph].

Appendix A: Comparison with existing decomposition methods in terms of execution time

Method	X_d^{+1}			X_d^{-1}			H_d		
	d=4	d=5	d=6	d=4	d=5	d=6	d=4	d=5	d=6
QSearchPass	3182	14078	10513	3160	7046	14449	3712	6264	16764
QSweepPass	61.95	96.84	157.01	13.02	17.37	24.71	145.45	246.6	427.24
LocQRPass	0.227	0.294	0.328	0.239	0.298	0.552	0.421	0.785	0.789
LocAdaPass	0.706	0.93	1.223	1.263	3.23	8.81	1427.2	**TO**	**TO**
TAQR	0.28	0.347	0.487	0.281	0.425	0.468	0.435	0.688	1.009
TAQR Adaptive	1.261	2.318	4.456	1.253	2.341	4.146	1.553	2.854	4.878

Method	U_d			Two-qubit gate					
	d=4	d=5	d=6	RXX	RZZ	CZ	CX	CH	SWAP
QSearchPass	3993	7324	16364	10378	2554	2204	2891	3170	2550
QSweepPass	155.2	323.21	599.99	57.8	48.01	39.79	52.21	74.06	44.62
LocQRPass	0.396	0.553	0.998	0.361	0.097	0.083	0.229	0.237	0.147
LocAdaPass	1089.1	**TO**	**TO**	0.929	0.179	0.15	0.472	0.478	0.249
TAQR	0.494	0.692	1.035	0.407	0.117	0.105	0.265	0.282	0.166
TAQR Adaptive	1.67	2.857	4.861	1.502	0.597	0.594	1.638	1.185	0.859

(a) Line transition graph.

Method	X_d^{+1}			X_d^{-1}			H_d		
	d=4	d=5	d=6	d=4	d=5	d=6	d=4	d=5	d=6
QSearchPass	2821	3067	3742	2929	3057	4241	4147	7187	17974
LocQRPass	0.392	0.896	0.625	0.361	0.514	0.654	0.842	1.361	2.036
LocAdaPass	1.052	2.222	2.545	1.005	1.509	2.144	4.606	14.07	36.08
TAQR	0.663	0.472	0.623	0.274	0.367	0.459	0.397	1.09	0.99
TAQR Adaptive	1.359	2.204	2.374	1.059	1.631	2.355	1.379	2.335	3.578

Method	U_d			Two-qubit gate					
	d=4	d=5	d=6	RXX	RZZ	CZ	CX	CH	SWAP
QSearchPass	3949	7785	17719	3461	2602	2200	3068	3107	3051
LocQRPass	1.186	1.173	1.947	0.713	0.101	0.083	0.407	0.408	0.208
LocAdaPass	4.181	13.41	39.36	1.254	0.168	0.143	0.769	0.711	0.396
TAQR	0.431	0.735	1.048	0.314	0.113	0.104	0.254	0.279	0.268
TAQR Adaptive	1.38	2.33	3.581	1.128	0.398	0.353	1.118	1.009	1.249

(b) Star transition graph.

Method	X_d^{+1}			X_d^{-1}			H_d		
	d=4	d=5	d=6	d=4	d=5	d=6	d=4	d=5	d=6
QSearchPass	2446	2835	4177	2524	2949	4469	3159	7087	11866
LocQRPass	0.352	0.489	0.651	0.371	0.553	0.775	0.761	1.27	1.823
LocAdaPass	0.872	1.65	2.437	0.953	1.565	2.339	3.658	13.0	37.24
TAQR	0.286	0.439	0.576	0.262	0.387	0.47	0.441	0.736	0.966
TAQR Adaptive	1.203	1.952	3.11	1.213	1.821	2.542	1.57	2.568	3.822

Method	U_d			Two-qubit gate					
	d=4	d=5	d=6	RXX	RZZ	CZ	CX	CH	SWAP
QSearchPass	3525	7992	19952	2276	2289	2267	2239	2229	2268
LocQRPass	0.658	1.152	1.933	0.619	0.099	0.086	0.354	0.369	0.15
LocAdaPass	826.7	12.02	49.2	1.102	0.171	0.138	0.625	0.606	0.254
TAQR	0.438	0.677	1.032	0.21	0.116	0.108	0.161	0.169	0.163
TAQR Adaptive	1.574	2.594	3.909	1.027	0.418	0.405	0.93	0.83	0.987

(c) Bipartite transition graph.

Table III. Comparison of decompositions produced by given methods in terms of execution time (in milliseconds). **TO** means that the execution exceeded the 1-minute timeout.

Appendix B: Python code to execute decomposition

We use the following Python code to perform a comparison of different methods for qudit unitary decomposition. Each method produces qudit circuit format that contains only allowed transition gates and phase-shift gates.

```python
import numpy as np
import networkx as nx

from bqskit.ir import Circuit
from bqskit.ir.gates import EmbeddedGate, U1Gate, U1qGate
from bqskit.ir.opt.cost import HilbertSchmidtCostGenerator
from bqskit.ir.opt.minimizers import LBFGSMinimizer
from bqskit.ir.opt.instantiaters import Minimization
from bqskit.qis import UnitaryMatrix
from bqskit.compiler import MachineModel, Compiler, Workflow
from bqskit.passes import SetTargetPass, SetModelPass
from bqskit.passes import QSearchSynthesisPass, SingleQuditLayerGenerator
from bqskit.passes import ScanningGateRemovalPass


class QSearch:
    def decompose(self, utry: np.ndarray, trans: nx.Graph) -> Circuit:
        assert len(utry.shape) == 2 and utry.shape[0] == utry.shape[1]
        d = utry.shape[0]

        target = UnitaryMatrix(utry, [d])

        model = MachineModel(
            1,
            None,
            [EmbeddedGate(U1Gate(), [d], [0, k]) for k in range(1, d)]
            + [EmbeddedGate(U1qGate(), [d], [i, j]) for i, j in trans.edges],
            [d],
        )

        workflow = Workflow(
            [
                SetTargetPass(target),
                SetModelPass(model),
                QSearchSynthesisPass(
                    max_layer=d**2,
                    layer_generator=SingleQuditLayerGenerator(
                        None, allow_repeats=False
                    ),
                    instantiate_options={
                        "method": Minimization(
                            cost_fn_gen=HilbertSchmidtCostGenerator(),
                            minimizer=LBFGSMinimizer(),
                        )
                    },
                ),
                ScanningGateRemovalPass(
                    instantiate_options={
                        "multistarts": 2,
                        "max_iters": 1000,
                        "min_iters": 10,
                    }
                ),
            ]
        )

        with Compiler(num_workers=8, runtime_log_level=40) as compiler:
            circuit = compiler.compile(Circuit(1, [d]), workflow=workflow)
        return circuit
```

Listing (1) Invoke *QSearchPass* decomposition

```python
import numpy as np

from bqskit.ir import Circuit
from bqskit.ir.gates import U1Gate, U1qGate
from bqskit.qis import UnitaryMatrix
from QSweep.qsweep import QSweepPass


class QSweep:
    def decompose(self, utry: np.ndarray) -> Circuit:
        assert len(utry.shape) == 2 and utry.shape[0] == utry.shape[1]
        d = utry.shape[0]

        target = UnitaryMatrix(utry, [d])
        qsweep_pass = QSweepPass({U1Gate(), U1qGate()})
        return qsweep_pass.synthesize_non_async(target)
```

Listing (2) Invoke *QSweepPass* decomposition

```python
import numpy as np
import networkx as nx

from mqt.qudits.core import LevelGraph
from mqt.qudits.compiler import QuditCompiler
from mqt.qudits.quantum_circuit import QuantumCircuit
from mqt.qudits.simulation import MQTQuditProvider
from mqt.qudits.simulation.backends.backendv2 import Backend


class FakeBackend(Backend):
    def __init__(self, trans: nx.Graph, dim: int):
        super().__init__(provider=MQTQuditProvider(), name="FakeBackend")

        graph = LevelGraph(
            edges=trans.edges,
            nodes=list(range(dim)),
            nodes_physical_mapping=list(range(dim)),
            initialization_nodes=[0],
        )
        self._energy_level_graphs: list[LevelGraph] = [graph]

    @property
    def energy_level_graphs(self) -> list[LevelGraph]:
        return self._energy_level_graphs

    def run(self, circuit, **options):
        pass


class MqtQudits:
    def __init__(self, method: str):
        assert method == "LocQRPass" or method == "LocAdaPass"
        self.method = method

    def decompose(self, utry: np.ndarray, trans: nx.Graph) -> QuantumCircuit:
        assert len(utry.shape) == 2 and utry.shape[0] == utry.shape[1]
        d = utry.shape[0]

        circuit = QuantumCircuit(1, [d])
        circuit.cu_one(0, utry)

        backend = FakeBackend(trans, d)
        return QuditCompiler().compile(backend, circuit, [self.method])
```

Listing (3) Invoke *MQT.Qudits* decomposition