Compilation of Entangling Gates for High-Dimensional Quantum Systems

Kevin Mato* Martin Ringbauer+ Stefan Hillmich† Robert Wille*‡

* Chair for Design Automation, Technical University of Munich, Germany
+Institute for Experimental Physics, University of Innsbruck, Austria
†Institute for Integrated Circuits, Johannes Kepler University Linz, Austria
‡ Competence Center Hagenberg (SCCH) GmbH, Austria
kevin.mato@tum.de,martin.ringbauer@uibk.ac.at,stefan.hillmich@jku.at,robert.wille@tum.de
https://www.cda.cit.tum.de/research/quantum/

ABSTRACT

Most quantum computing architectures to date natively support multi-valued logic, albeit being typically operated in a binary fashion. Multi-valued, or qudit, quantum processors have access to much richer forms of quantum entanglement, which promise to significantly boost the performance and usefulness of quantum devices. However, much of the theory as well as corresponding design methods required for exploiting such hardware remain insufficient and generalizations from qubits are not straightforward. A particular challenge is the compilation of quantum circuits into sets of native qudit gates supported by state-of-the-art quantum hardware. In this work, we address this challenge by introducing a complete workflow for compiling any two-qudit unitary into an arbitrary native gate set. Case studies demonstrate the feasibility of both, the proposed approach as well as the corresponding implementation (which is freely available at *github.com/cda-tum/qudit-entanglement-compilation*).

CCS CONCEPTS

• **Hardware → Quantum computation**; **Electronic design automation**; **Emerging languages and compilers**.

ACM Reference Format:
Kevin Mato, Martin Ringbauer, Stefan Hillmich, and Robert Wille. 2023. Compilation of Entangling Gates for High-Dimensional Quantum Systems. In *Proceedings of Asia and South Pacific Design Automation Conference (ASP-DAC '23).* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3566097.3567930

1 INTRODUCTION

In the future, quantum computers are expected to solve industrial and scientific problems with a reduced consumption of resources and greater algorithmic efficiency. Current quantum devices can host up to hundreds of noisy *quantum bits* (qubits) and support a limited number of logical operations on these qubits. For this reason, they are referred to as *Noisy Intermediate-Scale Quantum* (NISQ) devices [1]. A variety of technology platforms have now implemented such NISQ devices, including superconducting circuits [2], trapped ions [3], and single photons [4]. Notably, while these devices thus far almost exclusively work with two-level qubits, the underlying hardware almost always natively supports encoding multi-valued logic in high-dimensional *quantum digits* (qudits).

The research on qudit design and computation has a long history, with efforts primarily focusing on conceptual studies of algorithms for idealized qudits and their comparison to qubits [5]. Fundamentally, a qudit can not only store and process more information per quantum particle, but also features a richer set of logical operations [6] that make processing more efficient. Proof-of-principle demonstrations [7]–[10] have shown that qudits enable improvements in circuit complexity and algorithmic efficiency for a wide class of problems. These results inspired proposals for and demonstration of qudit basic control in numerous physical platforms such as trapped ions [7], [11], photonic systems [8], [12]–[14], superconducting circuits [15], [16], Rydberg atoms [17], [18], nuclear spins [19], cold atoms [20], [21], and nuclear magnetic resonance systems [9]. More recently, efforts have intensified with the demonstration of universal qudit quantum processors with error rates that are competitive to qubit systems [7], [22].

However, a key task for using qudit quantum processors remains the efficient compilation of circuits for this hardware. Compared to qubits, where every entangling gate is equivalent to the *CNOT* [23] gate, qudits offer much richer possibilities in the form of many inequivalent entangling gates. The flip side of these opportunities, however, are challenges in finding suitable gate sets that are native to the hardware and, then, compiling algorithms to these gate sets. Thus far, much of the theory required to address these challenges is insufficient and, accordingly, no corresponding design methods are available for this purpose yet—making the compilation of entangling gates a mostly manual task and, thus, preventing the further exploitation of high-dimensional quantum systems.

In this work, we introduce a workflow for the compilation of arbitrary two-qudit entangling gates in any high-dimensional system. The core idea is to map the target unitary operating on two qudits to a single-qudit unitary in an appropriately larger space. The latter can then be decomposed into two-level couplings using established techniques [24]. The resulting decomposition will feature at most d^2 two-level entangling gates of two standardized types. For decomposing these two standard gates into any hardware-native gate set, we developed an offline optimization routine, which we demonstrate on a recently developed gate set for trapped ion qudit quantum processors [7], [25]. In typical experimental scenarios, where the native gates act only on a subspace of the full Hilbert space, the cost of this pre-computation is independent of the size of the target unitary. The resulting circuit will express the target unitary using only native gates. Overall, this results in a methodology that, for the first time, enables compiling entangling gates for high-dimensional quantum systems in a fully automatic fashion. The proposed method can be applied to any two-qudit unitary and is computationally efficient, since the numerical optimization step is done offline. Case studies demonstrate the feasibility of the proposed method and a corresponding implementation is available in open-source via *github.com/cda-tum/qudit-entanglement-compilation*, along with other components of the Munich Quantum

ASP-DAC '23, January 16–19, 2023, Tokyo, Japan
© 2023 Association for Computing Machinery.
ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/3566097.3567930

Toolkit (MQT). The proposed tool is only a component of a complete compiler for high-dimensional and mixed-dimensional systems, that realizes a fully automated and computationally scalable workflow for the compilation of entangling operations. Therefore the results proposed are not necessarily efficient to the point of practical applicability on a quantum system, both in terms of gate count and possible requirement of additional circuit optimization.

The remainder of this paper is structured as follows: Section 2 briefly reviews the basics of quantum information processing (QIP) and entanglement. Section 3 motivates the problem of entanglement compilation for qudits. Section 4 describes the proposed approach to tackle entanglement compilation. Section 5 provides case studies on the feasibility of the proposed approach. Finally, Section 6 concludes the paper.

2 BACKGROUND

In this work, we provide the basis for efficient compilation of entangling operations for high-dimensional systems. To this end, this section first briefly reviews the fundamentals of QIP (with a focus on high-dimensional quantum logic) and entanglement.

2.1 Quantum Information Processing

The fundamental unit of classical information is the *bit* (binary digit), which can exclusively take the values 0 or 1. Generalizing this concept to quantum computers gives rise to the *qubit* as the corresponding unit of quantum information. The crucial difference to the classical case, however, is that qubits can be in any linear combination of the basis states $|0\rangle$ and $|1\rangle$ (using Dirac's bra-ket notation).

Since there are no ideal two-level systems in nature, qubits are usually constructed by restricting the natural multi-level structure of the underlying physical carriers of quantum information. These systems, therefore, natively support multi-level logic with the fundamental unit of information termed a *qudit*. A qudit is the quantum equivalent of a d-ary digit with $d \geq 2$, whose state can be described as a vector in a d-dimensional complex Hilbert space $\mathcal{H}_d$. The quantum state $|\psi\rangle$ of a qudit can thus be written as a linear combination $|\psi\rangle = \alpha_0 \cdot |0\rangle + \alpha_1 \cdot |1\rangle + \ldots + \alpha_{d-1} \cdot |d-1\rangle$, or simplified as vector $|\psi\rangle = [\, \alpha_0 \; \alpha_1 \; \ldots \; \alpha_{d-1} \,]^{\mathrm{T}}$, where $\alpha_i \in \mathbb{C}$ are the amplitudes relative to the computational basis of the Hilbert space—given by the vectors $|0\rangle, |1\rangle, |2\rangle, \ldots, |d-1\rangle$. The squared magnitude of an amplitude $|\alpha_i|^2$ gives the probability with which the corresponding basis state i would be observed when measuring the qudit in the computational basis. Normalization of probabilities further requires that $\sum_{i=0}^{d-1} |\alpha_i|^2 = 1$.

Example 2.1. Consider a single qudit with three levels (also referred to as *qutrit*).

The quantum state $|\psi\rangle = \sqrt{1/3} \cdot |0\rangle + \sqrt{1/3} \cdot |1\rangle + \sqrt{1/3} \cdot |2\rangle$ is a state with equal probability of measuring each basis. A different notation for the same quantum state may be $\sqrt{1/3} \cdot [\, 1 \; 1 \; 1 \,]^{\mathrm{T}}$.

Two key properties that distinguish quantum computing from classical computing are superposition and entanglement. A qudit is said to be in a *superposition* of states in a given basis when at least two amplitudes are non-zero relative to this basis. *Entanglement*, instead, describes a powerful form of non-classical correlation born from interactions in multi-qudit systems. Quantum computer operations are represented by unitary matrices U, satisfying $U^{\dagger}U = UU^{\dagger} = I$.

2.2 Entanglement Structures

Classically, the state of a bipartite system can always be written as a product of the state of the individual systems (or a mixture of such products). Entangled quantum states encode information in a non-local way, such that it can only be extracted from the full system, but not from the constituent qudits. More precisely, two or more quantum systems are in an *entangled* state, whenever their state cannot be written as a product of states of the individual subsystems (or a convex mixture of such products) [23]. While systems of two qubits are quite well understood, entanglement in multi-partite or higher dimensional systems still present a range of open questions [26].

The central scope of the work are quantum logic operations generating quantum entanglement, with a special focus on bipartite entangling gates, which enable the core aspect of error correction. Generalizations to multipartite entangling operations will be tackled where appropriate. The first notable observation is the much richer entanglement structure of qudits compared to qubits.

Specifically, for qubits, all entangling operations are related to the controlled-*NOT* (*CNOT*) gate via local operations on the subsystems [23]. Hence, for qubit systems, it is sufficient to compile the *CNOT* gate to the native operations of the quantum hardware. For qudit systems, instead, this is no longer true and they can be entangled in many inequivalent ways. Consequently, while any single entangling gate is sufficient for universal quantum computation [27], not all entangling gates are equally useful for any given application.

Example 2.2. Consider, the controlled-exchange gate *CEX* [7] defined by

$$CEX_{c,t_1,t_2} : \begin{cases} |c, t_1\rangle \leftrightarrow |c, t_2\rangle \\ |j, k\rangle \longrightarrow |j, k\rangle \quad \text{for } j \neq c, k \neq t_1, t_2. \end{cases} \tag{1}$$

This qudit-embedded version of the *CNOT* gate generates qubit-level entanglement in a high-dimensional Hilbert space. However, there are gates that directly generate *genuine* qudit entanglement, such as the controlled-SUM gate defined by

$$CSUM : |i, j\rangle \mapsto |i, i \oplus j\rangle, \tag{2}$$

where $\oplus$ denotes addition modulo d. These are just two examples of a more general theme in qudit systems, where entangling gates differ in their *entangling power* or *structure*, that corresponds to the amount of entanglement generated by an operation. The reasons why genuine qudit gates produce more entanglement than the first one are more intuitively explained in Ref. [25]. While the *CEX* gate has a low entangling power, since it only generates two-level entanglement, it is very flexible and can be used to intuitively build up any entanglement structure. In turn, however, it is quite inefficient for constructing highly entangling unitaries such as the *CSUM* gate out of two-level entangling operations. Too much entangling power, however, is not always helpful either. It may happen that a qudit circuit requires the generation of entanglement only within a small subspace of the Hilbert space, in that case applying *CSUM* becomes disadvantageous and the *CEX* is preferable.

In practice, of course, neither of the gates from Example 2.2 might be natively supported by the quantum hardware. However, potent qudit quantum devices are expected to support several primitive entangling gates, with a range of entangling powers. Hence, it becomes crucial to be able to compile arbitrary qudit unitaries into the native set of operations, while making the best use of the available resources.

3 MOTIVATION

Based on the general setting introduced above, the compilation of entangling operations into native gate sets is of central importance for efficient QIP. Since the general problem is NP-hard already for qubits [28], quantum computers critically depend on efficient, if not optimal, compilation methods. Given the more complicated entanglement structure of qudits, compared to their binary counterpart, the compilation problem becomes much more challenging in high-dimensional spaces.

3.1 Problem Formulation

Qubit circuits may be written using several entangling gates, all equivalent to the *CNOT* gate [29]. For qudit systems, the situation is very different. With the structure of entanglement becoming much richer, there is not just more freedom in designing quantum circuits, but also more opportunities for optimizing their efficiency with the right set of operations. The challenge for qudit compilation is then to understand and to harness this entanglement structure for efficient decompositions while at the same time, keeping the problem computationally tractable.

Example 3.1. In order to make the problem more intuitive for the qubit expert, the use of a metaphor can help. Using color coding as an analogy: Information encoded in one qubit is like a single grayscale pixel. Information encoded in a *qutrit* ($d = 3$) is like an RGB pixel that can combine 3 different colors and all their possible shades inbetween. Now, adding a second pixel, the qubits will only have two parameters, where qutrits have 6 parameters for leveraging the full potential of the two pixels. This highlights the drastically higher information density in qutrits (and subsequently qudits of even higher dimensions). However, the qutrit, just like the color pixel, requires a vastly more complex control system, to efficiently use the capability.

The problem of entanglement compilation can now be formulated as follows: *Given a unitary U representing an interaction between two qudits of dimension d, find a decomposition of U into arbitrary local unitaries and a pre-defined set of entangling gates, in a way that is as close to the optimum as possible.* In an application setting, the native gate set will be defined by the used quantum hardware, which will also set the cost of each component of the decomposition. While it is in general not known how complex the optimal solution is, the general figure of merit will be the gate fidelity given the experimental noise sources for the various gates. Here, we will focus on two-qudit gates, although the methods we present generally also apply to the multi-qudit setting, at the cost of decomposing the tensor product in two-qudit unitaries.

3.2 State of the Art

In this section, we briefly review existing approaches to compile entangling operations in high-dimensional Hilbert space. While some results exist for special cases [24], [30], these are of limited use in actual quantum hardware, which typically does not natively support the types of interactions we might want to work with.

More precisely, in pioneering work, Ref. [24] introduced a synthesis algorithm for qudit entangling gates that produces decompositions in terms of controlled-householder rotations. They prove a lower and a constructive upper bound on the number of two-qudit controlled-rotation gates required for an arbitrary two-qudit unitary. These results, however, only apply to the specific controlled rotations used, without considering the constraints or indeed opportunities of physical qudit quantum processors. In Ref. [31], a compilation algorithm for generic qudit unitaries is presented. The final goal of the mathematical procedure is to reduce the number of gates that use magic state injection protocols. The entanglement complexity of qudit systems is ignored, and although an elegant solution is found, the restricted scalability of the algorithm and the final output, too far from being applied to a physical quantum processor, make the work still theoretical. Other previous works [32] try to lay out routines for the synthesis of qudit circuits, in this particular case for qutrits only.

Related to this, Ref. [30] presents a method to construct circuits for generating different forms of qudit entanglement using a specific gate set including the controlled-exchange gate. While the goal is the generation of specific quantum states from a fixed input, it is unclear to what extent these methods can be transferred to the more complicated compilation of entangling unitaries.

In Ref. [33], the challenge of multi-qubit compilation is tackled. The authors use parametrized quantum circuits and numerical optimization of the fidelity function, for solving an incremental structure of alternating local and entangling gate layers. More precisely, the work focuses on the compilation of global entangling gates based on an ansatz made of alternating global entangling gates and specific equatorial rotations; the approach cannot be directly applied to qudit entanglement compilation, due to the inefficient search strategy. Moreover, the solution provided in [33] is unable to express the richer variety of local and partial entangling operations available in this new setting. Finally, in Ref. [34] arbitrary controlled unitaries are proved to be physically implemented using auxiliary levels in the qudit Hilbert space, regardless of their form.

4 EFFICIENT COMPILATION OF ENTANGLEMENT

Motivated by the challenges described in Section 3, we now propose a workflow that enables the compilation of any two-qudit unitary into any target gate set. The proposed solution consists of two steps. First, a synthesis method, which takes the target two-qudit unitary and returns a high-level circuit comprised of only two types of two-level entangling gates, regardless of the initial unitary. This step is crucial for making the technique scalable to arbitrary dimensions. Second, a compilation step that uses parametrized circuits and numerical optimization to decompose the two standardized two-level entangling gates into any target gate set. Although this step can be computationally intensive, it can be pre-computed, since the structure from the first step is fixed. Hence, this step does not affect the scalability of the method.

4.1 Step 1: QR Decomposition of Entangling Operations

Qudit quantum hardware typically exploits two-level couplings between the various qudit levels, despite having access to a full high-dimensional Hilbert space. It has been shown that this is sufficient for implementing arbitrary single-qudit operations with a cost that is at most quadratic in the size of the Hilbert space [24]. In the simplest instance, the resulting sequence of operations is composed of two-level *Givens*-rotations between adjacent sites V_i and a diagonal phase matrix Θ in $U = V_k \cdot V_{k-1} \cdot \ldots \cdot V_1 \cdot \Theta$ [24]. This algorithm is universally valid. However, it scales quadratically in the Hilbert space dimension and, due to the rigid structure, it can introduce redundant operations. Hence, for efficiently compiling local qudit unitaries, more advanced algorithms, that take into account the structure of the qudits and experimental constraints, can be beneficial. For the purpose of the present work, however, it is precisely the standardized structure of the QR decomposition that is beneficial.

In more detail, we start by interpreting the target two-qudit unitary U as a single qudit unitary in dimension d^2. This is achieved by mapping the two-qudit states to a single qudit as $|i, j\rangle \mapsto |d \cdot i + j\rangle$. We then apply the QR decomposition algorithm to that higher-dimensional local unitary. Without loss of generality, we thereby assume a ladder-type coupling, where each of the virtual single-qudit states is coupled to its neighboring states $|j\rangle \leftrightarrow |j \pm 1\rangle$. The QR decomposition then applies successive rotations between adjacent state in the virtual single qudit. The result of the decomposition is a set of two-level rotations of the form

$$R(\theta,\phi) = \begin{bmatrix} \cos\frac{\theta}{2} & \sin\frac{\theta}{2}(-i\cos\phi - \sin\phi) \\ \sin\frac{\theta}{2}(-i\cos\phi + \sin\phi) & \cos\frac{\theta}{2} \end{bmatrix}$$

followed by a phase matrix, which can be represented as a sequence of phase rotations on neighboring states, which again can be decomposed into two-level rotations using the identity $Z(\theta) = R(\frac{\pi}{2},0)\cdot R(\theta,\frac{\pi}{2})\cdot R(-\frac{\pi}{2},0)$.

As a consequence of our choice of ladder-type coupling, the two-level rotations all occur between adjacent states and are thus embedded into the d^2 dimensional Hilbert space as

$$\hat{R}_i(\theta,\phi) = \begin{bmatrix} 1 & \cdots & 0 \\ 0 & \cdots [R(\theta,\phi)]_{i,i+1} \cdots & 0 \\ 0 & \cdots & 1 \end{bmatrix},$$

where the subscript $i, i+1$ denotes the affected states. The resulting matrix is a two-level rotation $\hat{R}$ embedded in a higher-dimensional space that translates to an entangling operation in the original two-qudit system. The next challenge is now decomposing these entangling gates into the native gate set of the target hardware.

A straightforward approach would be to simply try and compile each of the $O(d^2)$ entangling gates in the sequence into the target gate set, although computationally very costly. Instead, one can notice that only two different gates need to be compiled. This is most easily seen at a simple example with two qutrits. By design, each of the two-level rotation R_i is represented as 2×2 block matrices in a $d^2 \times d^2$ identity matrix as in Eq. (3).

$$\left[\begin{array}{ccc|ccc|ccc} \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & cR & cR & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & cR & cR & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \hline \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & pS & pS & \cdot & \cdot \\ \hline \cdot & \cdot & \cdot & \cdot & \cdot & pS & pS & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \end{array}\right] \tag{3}$$

Here, the horizontal and vertical lines indicate the tensor product structure of the original two-qutrit unitaries. We note that whenever the rotation R_i acts on levels that do not cross these lines, it physically corresponds to a controlled-rotation gate *cRot* in the two-qudit system (indicated above as *cR*). When R_i does cross a boundary, it corresponds to a partial Swap operation *pSWAP* in the bipartite system (indicated above as *pS*). While mathematically very similar, these operations are fundamentally different in their complexity, so they must be treated separately. Fortunately, it is sufficient to compile just one example of each, since the other cases (for different qudit indices) can be obtained simply by permuting the states of one operation of the same type. For simplicity and generality, we focus in the following on gates that act in the 0-1 subspace of the qudit Hilbert space.

After the generation of the abstract sequence of rotations, the compiler simplifies the complexity of the problem by applying permutation gates to every *cRot* and *pSwap* gate in order to express the sequence in terms of only two entangling operations, $cRot_{1;0,1}$, $pSwap_{0,1}$, and permutation gates. This way, every operation can be compiled at the expense of compiling efficiently only one *cRot* and one *pSwap*. The two operations cannot, in general, be implemented directly on the hardware, therefore one further step is needed before concluding the synthesis.

Since $cRot_{1;0,1}$, $pSwap_{0,1}$ are parametrized in θ and ϕ, it would require to compile these default gates for every different value of the two variables. The default gates chosen will be further decomposed into a sequence of local operations, that will encode the variables θ and ϕ for the equatorial rotations, plus a static entangling gate that will provide the necessary entangling strength and that will act on a two-level subspace.

In this regard, the compiler will follow the decomposition of the controlled rotation $cRot_{1;0,1}$ (control on the level 1 of the first qudit, target the subspace 0-1 of the second qudit) as,

$$\begin{aligned} cRot_{1;0,1}(\theta,\phi) = {} & [I\otimes R(\pi/2, -\phi-\pi/2)]\cdot \\ & CEX\cdot[I\otimes Z(\theta/2)]\cdot \\ & CEX\cdot[I\otimes(Z(-\theta/2)\cdot R(-\pi/2,-\phi-\pi/2)), \end{aligned} \tag{4}$$

where the decomposition uses the $CEX1;0,1$ that will act on the 0,1 subspace of the second qudit, while the partial swap:

$$\begin{aligned} pSwap_{0,1}(\theta,\phi) = {} & [H\otimes H]\cdot CEX\cdot[H\otimes H] \\ & [I\otimes R(\pi/2,-\phi-\pi/2)]\cdot CEX\cdot \\ & [I\otimes Z(\theta/2)]\cdot CEX\cdot[I\otimes Z(-\theta/2)]\cdot \\ & [I\otimes R(-\pi/2,-\phi-\pi/2)]\cdot \\ & [H\otimes H]\cdot CEX\cdot[H\otimes H] \end{aligned} \tag{5}$$

the $CEX_{1;0,1}$ is used once again in the *pSwap* operation.

Example 4.1. In order to better understand the intermediary compilation step, the operations involved, and the new encoding, let us consider the case where two qutrits are interacting. The controlled rotation on levels $|7\rangle$ and $|8\rangle$ appears in the decomposition, which correspond to $|21\rangle$ and $|22\rangle$ in the two-qutrit system. Since the method requires the controlled rotation to be applied to the 0-1 subspace, we have to permute the levels of the two-qutrit system. We add local rotations with angle π and phase $\pi/2$ that will act as permutations and route $|21\rangle$ to $|10\rangle$ and $|22\rangle$ to $|11\rangle$.

The decomposed sequence will be:

$$\begin{aligned} cRot_{2;1,2}(\theta,\phi) = {} & (P_{0,2}\cdot P_{1,2}\otimes P_{0,1}.P_{1,2})^{\dagger}\cdot \\ & cRot_{1;0,1}(\theta,\phi)\cdot \\ & (P_{0,2}\cdot P_{1,2}\otimes P_{0,1}.P_{1,2}), \end{aligned}$$

where $P_{i,j}$ denotes a permutation of levels i and j.

Note that after executing the controlled-rotation, the sequence finishes with the uncomputation of the permutation sequence, in order to restore the original encoding for the subsequent operations.

At this point the synthesis returns a sequence of local operations that will encode the angles of rotation and the permutation of the states, interleaved with the chosen *CEX* gate. This is the default choice made in the project, but the user could provide a different one that could act on a different subspace. The compiler will now proceed with the last step: the compilation of the *CEX* gate with an entangling gate input by the user and compatible with the target machine.

4.2 Step 2: Layered Compilation

This section describes the second step of the compiler and the lowest abstraction level before getting to a physical implementation. The software component takes as input the target unitary to compile and the target entangling gate for the decomposition. The entangling gates could be a primitive entangling gate of the target quantum hardware, or it could be a higher-level abstract gate. The software outputs a compiled sequence made of arbitrary local two-level rotations and the chosen gate.

At the core of the second component are parametrized quantum circuits, that solved for different gates, lead to different decompositions. More precisely, as Figure 1 shows, results present different depths and noise characteristics depending on the noise and power of the single entangling operation.

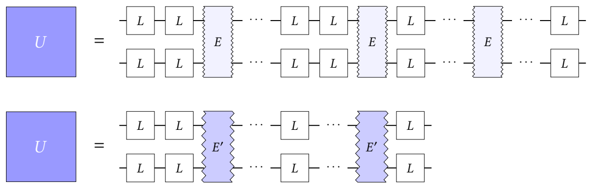

Figure 1: Given a two-qudit unitary U, compilation results depend on the structure and the noise inherent to the chosen gate E; new local (L) gates will match the gate E.

The section will continue with a brief overview of parametrized quantum circuits, followed by a detailed explanation on the construction of the problem representation and its resolution by the layered compiler.

4.2.1 Parametrized Quantum Circuits (PQCs). PQCs consist of a series of quantum gates dependent on a set of continuous or discrete parameters θ_i that can be optimized. The particular initial choice of the sequence of gates is called an *ansatz*. This circuit is then optimized, typically by a classical algorithm, to minimize a cost function that encodes the solution of interest.

This approach can be used for a range of computational questions [35], like Hamiltonian ground-state energy estimation [36] or compilation [33]. In the following we discuss the important design choices during the ansatz construction, its resolution and how the ansatz evolves during the optimization loop.

4.2.2 Construction of Ansatz and Expressibility. The first step consists of choosing the building blocks for our ansatz and then incrementally composing them in order to create the final circuit to be solved. There are two types of operations involved in the quantum circuit: local operations and entangling operations.

The initial objective is to find a representation for single-qudit operations that is parametric and that achieves maximal expressibility [37] with the minimal number of parameters. Here, expressibility refers to the circuit's ability to generate a large fraction of two-qudit unitaries. The proposed solution exploits theoretical results [38] already present in literature. The single qudit lives in a d-dimensional ($d \geq 2$) complex Hilbert space $\mathcal{H} = \mathbb{C}^d$ spanned by the orthonormal basis $|0\rangle, \ldots, |d-1\rangle$. Local qudit unitaries will be written as in [38], where elements of the special unitary group $\mathcal{SU}(d)$ are parametrized as:

$$U = \left[\prod_{m=0}^{d-2}\left(\prod_{n=m+1}^{d-1}\exp(iZ_{m,n}\lambda_{n,m})\exp(iY_{m,n}\lambda_{m,n})\right)\right]\cdot \left[\prod_{l=1}^{d-1}\exp(iZ_{l,d}\lambda_{l,l})\right]$$

using d^2-1 real parameters $\{\lambda_{m,n}\}$ in the interval between 0 and π for $m > n$, $\pi/2$ for $m < n$, as well as 2π for $m = n$.

On this space, Z is a diagonal trace-less operator, a generalized form of the Pauli σ_z applied to the subspace spanned by $|m\rangle$ and $|n\rangle$, while Y is the generalized form of the Pauli σ_y applied to the subspace on the subspace spanned by $|m\rangle$ and $|n\rangle$.

$$Z_{m,n} = |m\rangle\langle m| - |n\rangle\langle n| \qquad \text{for } 0 \leq m < n \leq d-1$$
$$Y_{m,n} = -i|m\rangle\langle n| + i|n\rangle\langle m| \qquad \text{for } 0 \leq m < n \leq d-1$$

This formulation comes with the promise of being expressible for the single qudits and it has the minimum number of parameters required for representing the special unitary group.

The second objective is an efficient use of user-specified entangling gates in the ansatz compilation. The usage of multiple species of qudit entangling gates inside the same ansatz is future research. Although the user can specify the preferred choice for the entangling primitive, here, we focus on using a single entangling gate, choosing between two gates commonly used in the trapped-ion platform: the Molmer-Sorenson (MS) gate in Eq. (6) [7] and the generalized light-shift (LS) in Eq. (7), a genuine qudit entangling gate [25], [39].

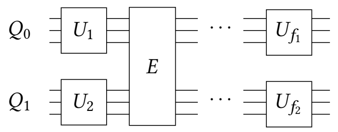

Figure 2: Example of an ansatz. Each layer contains an entangling gate, preceded by a generic unitary U on each one of the qudits and followed by two more generic unitaries; in this way, every layer is universal.

$$MS(\theta) = e^{-i\frac{\theta}{4}\cdot(I\otimes I+\sigma_{x_{01}}\otimes\sigma_{x_{01}})} \tag{6}$$

$$LS(\theta) = e^{-i\theta\cdot\sum_{i=0}^{d-1}|ii\rangle\langle ii|} \tag{7}$$

These are both well-established operations in quantum computing hardware with well-characterized noise characteristics. Moreover, they operate in a distinct way (the light-shift acts on phases, the MS acts on populations) with different entangling power [25] when applied to qudit systems.

Example 4.2. The compiler solves an ansatz like the one in Figure 2.— Consider a two-qudit entangling unitary U. Figure 1 illustrates two different compilation results: (1) The top result uses multiple pre-selected entangling gates with low entangling power but potentially less noise. (2) The bottom result uses two pre-selected entangling gates with high entangling power but more noise per gate.— The selection of the best result heavily depends on the performance of the gates, i.e., specifics of the underlying hardware. Both cases will always produce a correct result, as more entanglement can be built up with additional gates, and entanglement generation can also be reduced through judicious use of single qudit gates between entangling layers.

4.2.3 Solution of the Ansatz. The optimization of the ansatz works similar to a variational quantum algorithm. In each step, the full circuit is simulated by multiplying the gate matrices for the chosen parameters. This is computationally feasible for two-qudit gates in dimensions of practical relevance given the capabilities of current and future quantum hardware. For multi-qudit gates, the applicability of this approach is expected to be limited. The resulting unitary matrix in each optimization step is then compared to the target matrix according to an objective function. Here, the choice was the fidelity between the unitaries [40], for two-d-dimensional systems:

$$Fidelity(A,B) = \frac{1}{d^2}\operatorname{Tr}\langle A^\dagger\ , B\ \rangle$$

For the purpose of this work, the fidelity has several desirable properties that make it desirable for physical applications, compared to other commonly used objective functions for matrix optimization.

The optimization loop is performed by iteratively simulating the ansatz and verifying the fidelity achieved by the current solution of optimization algorithm.

4.2.4 Optimizer. The used optimization method is the dual annealing method, a combination of classical and fast simulated annealing, coupled with the L-BFGS [41] algorithm on boxed constraints, which is a local search method applied on the neighborhood of the solutions found by the global method in the high dimensional landscape of the problem to optimize. We refer the reader to Ref. [42] for details on this algorithm. Alternatively, the use of gradient-based methods is left for future work, due to the unclear trainability of qudit PQCs.

4.2.5 Binary Search. Although the cost of a circuit depends on its depth and gates structures, the final resulting circuit should have the least number of layers that can compile with the desired fidelity the entangling gate. The maximum number of layers for the search is heuristically chosen as $2 \cdot d^2$. This number was found to be sufficient for generating maximally entangled qudit states from the least powerful operation, acting only on two fixed physical levels. The compiler uses a binary search and the number of layers is decreased every time the gate can be decomposed for a certain number of layers under a time period heuristically chosen, otherwise if the target fidelity is not reached or the optimizer does not converge before the end of a timer, the number of layers is increased.

5 CASE STUDIES

The considerations made so far in the paper already showed that the problem of entanglement compilation in high-dimensional systems is complex. The solution proposed above is supposed to provide a step forward in the field of compilation for qudit systems. The implementation is available freely under the MIT licence at *github.com/cda-tum/qudit-entanglement-compilation* as part of the Munich Quantum Toolkit (MQT). It is completely written in Python 3.8, with exception of the external dependencies Numpy [43] and Scipy [42]. This section provides corresponding case studies, in order to demonstrate the feasibility of the workflow and its usefulness in compiling any multi-level entangling operation. The use cases focus on the compilation of the controlled-sum gate *CSUM*, a characteristic operation in qudit systems as discussed in the previous sections.

Although constituted by two compilation steps, the computational complexity of the workflow is dictated on a high level by three routines. The first routine is the QR decomposition applied on the unitary to compile; the algorithm has quadratic complexity in the number of dimension of the two-qudit interaction with size D, therefore $O(D^2)$. The result is a sequence of, in the worst case, D^2 operations. The second routine is the translation of every output operation in term of local operations and *CEX*; this algorithm has complexity $O(D^2)$, because linear in the number of output operations of the QR. Finally, the last routine of the workflow is the solution by optimization of the ansatz, which has complexity of $O(\log D)$. In fact, the last compilation step is a binary search, with every step of duration linear in the number of dimensions of the original unitary.

The evaluations were performed on a server running GNU/Linux using an AMD Ryzen 9 3950X and 128 GiB main memory. The layered compilation step is performed under a time limit in hours heuristically chosen as $d/4$ (e.g., 4 h for a two-ququarts unitary), at each step in the binary search. The compiler has minimum target infidelity ($1 - \textit{Fidelity}$) of 10^{-3}. Several runs of the optimization algorithm could lead to better results and, beyond dimension 16, the computational time and the fidelity are affected.

The considered example follows the default strategies encoded in the compiler. Every controlled rotation and partial swap is transformed by local rotations into $cRot_{1;0,1}$ and for the partial swaps $pSwap_{0,1}$, while the automatic choice for generating entanglement is the controlled exchange gates *CEX*. More precisely, $CEX_{1;0,1}$ has control on the first level of the first qudit and the targets on the first two levels of the second qudit.

The designer can decide to follow a different path and impose a decomposition for which they have a specific implementation in terms of the entangling gate involved, or compile it directly for the machine.

In the studies, we considered the implementation of the *CSUM* gate for qutrits and ququarts. Table 1 shows the results. Between the near-term usable physical qudit dimensions [7], the solution for ququarts proves an already difficult and useful design case. The columns denote the dimension ("Dim."), followed by the number of controlled rotations, and partial swaps, as well as the total number of *CEX*, *MS*, and *LS* gates required for the decomposition, respectively. The last column gives a bound on the achieved infidelity (i.e., $1 - \textit{Fidelity}$). In a first phase the synthesis of *CSUM* for 9 dimensions (two qutrits) requires 20 controlled rotations and 6 partial swaps, while the synthesis on two ququarts (dimension 16) outputs 42 controlled rotations and 6 partial swaps. These decompositions are, by construction, not necessarily optimal. The controlled rotations are further decomposed into 2 *CEX*s and the partial swap into 4 *CEX*s, with the resulting sequence being general for all dimensions.

Table 1: Results of the workflow on CSUM for dimension 9 and dimension 16

System	Dim.	*cRot*	*pSwap*	CEX_{tot}	MS_{tot}	LS_{tot}	$(1-F)$
two qutrits	9	44	24	184	1472	368	$< 10^{-4}$
two ququarts	16	92	36	328	9184	10168	$10^{-3} \sim 10^{-4}$

In order to show to the adaptability of the workflow, in the smaller case we compile the entangling gate in terms of two native gates used in the latest ion trap qudit processors [7] [25] and introduced before, the *MS* and *LS* gate. The manual decomposition of the *CEX* gate dedicated for the two-qutrit case by itself is non-trivial, even for an experienced quantum information specialists. The inherent difficulty arises not only from the amount of intuition needed, but also from the inability to easily generalize the single result found to higher dimensions. The pen-and-paper optimized sequence for *CEX* in dimension 9 is made of two *LS* gates with angle π, while two *MS* gates are also required for performing the same *CEX* at any dimension, if the sequence is allowed to exploit auxiliary levels [7].

The layered compiler performs a similar decomposition with the same number of π-*LS* gates autonomously. Without auxiliary levels, the compiler outputs a sequence of 8 *MS* gates. For both the compilation results, the infidelity reached is below 10^{-4}. In the case of two-ququarts, optimized sequences could not be achieved, while the layered compiler autonomously decomposes the *CEX* with infidelity between 10^{-3} and 10^{-4}, at the cost of 28 *MS* gates and in alternative of 31 *LS* gates. The number of gates required for a two-ququarts circuit gives a feeling of the increased complexity of the problem compared to two-qutrits.

The results are in line with the expected *automation overhead* in a first automatic solution. In fact, it was predictable that the used ansatz would ask for an over-parametrization [44], or using more layers than necessarily, which allows for a great simplification of the landscapes by eliminating spurious local minima. Although the results are not expected be efficient to the point of practical applicability on a quantum system, this is only one component of a complete compiler for high-dimensional and mixed-dimensional systems; future developments will comprise optimization routines[45]–[49].

In summary, the results show the feasibility of the proposed workflow to deliver decompositions efficiently for any two-qudit unitary. Further, the results provide evidence compilation of entangling operations for multi-level systems is not easy, especially compared to binary systems. This work investigates the first key step towards full automation for qudits. The implementation is available under the MIT license at *github.com/cda-tum/qudit-entanglement-compilation* as part of the Munich Quantum Toolkit (MQT).

6 CONCLUSION

The challenges for efficient and reliable application of entangling gates inside qudit circuits arise because of the complexity in understanding the amount of entanglement generated by an operation as well as the corresponding affected levels—commonly referred to as structure. Consequently, the operations are mostly considered as black-boxes with the question of whether it is possible to reach an implementation for a certain qudit platform and with a given gate-set. So far, the study of feasibility and the implementation of entangling operations for specific qudit technologies was performed by quantum information specialists manually, without the promise of re-utilizing a particular decomposition for a system certain number of dimension, to those of greater dimensionality.

In this paper, we introduced a complete workflow for compiling any two-qudit unitary into any target gate set. While the resulting gate sequences are typically not optimal in terms of gate count, the method is computationally efficient, since the only step involving numerical optimization can be pre-computed. The case studies confirm the feasibility of the workflow as well, for the first time, the automated results of compilation of generic entangling operations to any dimensionality. The implementation is available under the MIT license at *github.com/cda-tum/qudit-entanglement-compilation* under the ensemble of the Munich Quantum Toolkit (MQT). In the future, the proposed approach may be improved up on by utilizing auxiliary qudit levels, alternative ansatz designs, compression of synthesis results, and circuit optimization in post-processing.

ACKNOWLEDGMENTS

This work has received funding from the European Union's Horizon 2020 research and innovation programme under the ERC Consolidator Grant (agreement No 101001318), the Marie Skłodowska-Curie Grant (agreement No 840450), and the NeQST Grant (agreement No 101080086). It is part of the Munich Quantum Valley, which is supported by the Bavarian state government with funds from the Hightech Agenda Bayern Plus and was partially supported by the BMK, BMDW, and the State of Upper Austria in the frame of the COMET program (managed by the FFG).

REFERENCES

[1] J. Preskill, "Quantum computing in the nisq era and beyond," *Quantum*, vol. 2, p. 79, 2018.
[2] F. Arute, K. Arya, *et al.*, "Quantum supremacy using a programmable superconducting processor," *Nature*, vol. 574, no. 7779, pp. 505–510, 2019.
[3] I. Pogorelov, T. Feldker, *et al.*, "Compact Ion-Trap Quantum Computing Demonstrator," *PRX Quantum*, vol. 2, p. 020 343, 2021.
[4] F. Flamini, N. Spagnolo, and F. Sciarrino, "Photonic quantum information processing: A review," *Reports Prog. Phys.*, vol. 82, p. 016 001, 2019.
[5] Y. Wang, Z. Hu, B. C. Sanders, and S. Kais, "Qudits and high-dimensional quantum computing," *Front. Phys.*, vol. 8, 2020.
[6] M. Huber and J. I. de Vicente, "Structure of multidimensional entanglement in multipartite systems," *Phys. Rev. Lett.*, vol. 110, p. 030 501, 2013.
[7] M. Ringbauer, M. Meth, *et al.*, "A universal qudit quantum processor with trapped ions," 2021. arXiv: 2109.06903.
[8] B. P. Lanyon, M. Barbieri, *et al.*, "Simplifying quantum logic using higher-dimensional Hilbert spaces," *Nature Physics*, vol. 5, pp. 134–140, 2008.
[9] Z. Gedik, I. A. Silva, *et al.*, "Computational speed-up with a single qudit," *Sci. Rep.*, vol. 5, p. 14 671, 2015.
[10] X. Zhan, J. Li, *et al.*, "Linear optical demonstration of quantum speed-up with a single qudit," *Optics Express*, vol. 23, p. 18 422, 14 2015. [Online]. Available: https://www.osapublishing.org/abstract.cfm?URI=oe-23-14-18422.
[11] X. Zhang, M. Um, *et al.*, "State-independent experimental test of quantum contextuality with a single trapped ion," *Phys. Rev. Lett.*, vol. 110, p. 070 401, 2013.
[12] M. Ringbauer, T. R. Bromley, *et al.*, "Certification and quantification of multilevel quantum coherence," *Phys. Rev. X*, vol. 8, p. 041 007, 2018.
[13] X.-M. Hu, Y. Guo, *et al.*, "Beating the channel capacity limit for superdense coding with entangled ququarts," *Sci. Adv.*, vol. 4, no. 7, eaat9304, 2018.
[14] M. Malik, M. Erhard, *et al.*, "Multi-photon entanglement in high dimensions," *Nature Photonics*, vol. 10, pp. 248–252, 2016.
[15] M. Kononenko, M. Yurtalan, *et al.*, "Characterization of control in a superconducting qutrit using randomized benchmarking," *Phys. Rev. Res.*, vol. 3, no. 4, p. L042007, 2021.
[16] A. Morvan, V. V. Ramasesh, *et al.*, "Qutrit randomized benchmarking," *Phys. Rev. Lett.*, vol. 126, p. 210 504, 2021.
[17] J. Ahn, T. Weinacht, and P. Bucksbaum, "Information storage and retrieval through quantum phase," *Science*, vol. 287, no. 5452, pp. 463–465, 2000.
[18] J. R. Weggemans, A. Urech, *et al.*, "Solving correlation clustering with QAOA and a Rydberg qudit system: a full-stack approach," 2021. arXiv: 2106.11672.
[19] C. Godfrin, A. Ferhat, *et al.*, "Operating quantum states in single magnetic molecules: Implementation of Grover's quantum algorithm," *Phys. Rev. Lett.*, vol. 119, p. 187 702, 2017.
[20] B. E. Anderson, H. Sosa-Martinez, *et al.*, "Accurate and robust unitary transformations of a high-dimensional quantum system," *Phys. Rev. Lett.*, vol. 114, p. 240 401, 2015.
[21] V. Kasper, D. González-Cuadra, *et al.*, "Universal quantum computation and quantum error correction with ultracold atomic mixtures," 2020. arXiv: 2010.15923.
[22] Y. Chi, J. Huang, *et al.*, "A programmable qudit-based quantum processor," *Nature Communications*, vol. 13, p. 1166, 1 2022. [Online]. Available: https://www.nature.com/articles/s41467-022-28767-x.
[23] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information: 10th Anniversary Edition*, 10th. New York, USA: Cambridge University Press, 2011.
[24] S. Bullock, D. O'Leary, and G. Brennen, "Asymptotically optimal quantum circuits ford-level systems," *Phys. Rev. Lett.*, vol. 94, no. 23, 2005.
[25] P. Hrmo, B. Wilhelm, *et al.*, "Native qudit entanglement in a trapped ion quantum processor," 2022. arXiv: 2206.04104.
[26] R. Horodecki, P. Horodecki, M. Horodecki, and K. Horodecki, "Quantum entanglement," *Rev. Mod. Phys.*, vol. 81, no. 2, p. 865, 2009.
[27] G. K. Brennen, S. S. Bullock, and D. P. O'Leary, "Efficient circuits for exact-universal computations with qudits," 2005. arXiv: quant-ph/0509161.
[28] A. Botea, A. Kishimoto, and R. Marinescu, "On the complexity of quantum circuit compilation," in *Symp. on Combinatorial Search*, 2018.
[29] S. Bullock and I. Markov, "An arbitrary two-qubit computation in 23 elementary gates or less," in *Design Automation Conf.*, 2003, pp. 324–329.
[30] K. N. Smith and M. A. Thornton, "Entanglement in higher-radix quantum systems," in *Int'l Symp. on Multi-Valued Logic*, IEEE Computer Society, 2019, pp. 114–119.
[31] L. E. Heyfron and E. T. Campbell, "A quantum compiler for qudits of prime dimension greater than 3.," *arXiv: 1902.05634*, 2019.
[32] A. Glaudell, "Quantum compiling methods for fault-tolerant gate sets of dimension greater than two," Ph.D. dissertation, University of Maryland, College Park, 2019.
[33] E. A. Martinez, T. Monz, *et al.*, "Compiling quantum algorithms for architectures with multi-qubit gates," *New Journal of Physics*, vol. 18, no. 6, p. 063 029, 2016.
[34] N. Friis, V. Dunjko, W. Dür, and H. J. Briegel, "Implementing quantum control for unknown subroutines," *Phys. Rev. A*, vol. 89, no. 3, 2014.
[35] K. Bharti, A. Cervera-Lierta, *et al.*, "Noisy intermediate-scale quantum algorithms," *Rev. Mod. Phys.*, vol. 94, p. 015 004, 1 2022.
[36] M. Cerezo, A. Arrasmith, *et al.*, "Variational quantum algorithms," *Nature Reviews Physics*, vol. 3, no. 9, pp. 625–644, 2021.
[37] S. Sim, P. D. Johnson, and A. Aspuru-Guzik, "Expressibility and entangling capability of parameterized quantum circuits for hybrid quantum-classical algorithms," *Advanced Quantum Technologies*, vol. 2, no. 12, p. 1 900 070, 2019.
[38] C. Spengler, M. Huber, and B. C. Hiesmayr, "Composite parameterization and haar measure for all unitary and special unitary groups," *Journal of Mathematical Physics*, vol. 53, no. 1, p. 013 501, 2012.
[39] B. C. Sawyer and K. R. Brown, "Wavelength-insensitive, multispecies entangling gate for group-2 atomic ions," *Phys. Rev. A*, vol. 103, p. 022 427, 2021.
[40] X. Wang, C.-s. Yu, and X. X. Yi, "An alternative quantum fidelity for mixed states of qudits," *Physics Letters A*, vol. 373, pp. 58–60, 2008.
[41] D. C. Liu and J. Nocedal, "On the limited memory BFGS method for large scale optimization," *Mathematical Programming*, vol. 45, no. 1-3, pp. 503–528, 1989.
[42] P. Virtanen, R. Gommers, *et al.*, "SciPy 1.0: Fundamental algorithms for scientific computing in python," *Nature Methods*, vol. 17, pp. 261–272, 2020.
[43] C. R. Harris *et al.*, "Array programming with NumPy," *Nature*, vol. 585, no. 7825, pp. 357–362, 2020.
[44] M. Larocca, N. Ju, *et al.*, "Theory of overparametrization in quantum neural networks," 2021. arXiv: 2109.11676.
[45] R. D. da Silva, E. Pius, and E. Kashefi, "Global quantum circuit optimization," *arXiv preprint arXiv:1301.0351*, 2013.
[46] R. Duncan, A. Kissinger, S. Perdrix, and J. van de Wetering, "Graph-theoretic simplification of quantum circuits with the ZX-calculus," *Quantum*, vol. 4, p. 279, 2020. [Online]. Available: https://doi.org/10.22331%2Fq-2020-06-04-279.
[47] A. Kissinger and J. van de Wetering, "Reducing the number of non-clifford gates in quantum circuits," *Physical Review A*, vol. 102, no. 2, 2020. [Online]. Available: https://doi.org/10.1103%2Fphysreva.102.022406.
[48] Q. Wang, *Qufinite zx-calculus: A unified framework of qudit zx-calculi*, 2021. [Online]. Available: https://arxiv.org/abs/2104.06429.
[49] K. Mato, M. Ringbauer, S. Hillmich, and R. Wille, "Adaptive compilation of multi-level quantum operations," *arXiv preprint arXiv:2206.03842*, 2022.