Synthesis of and compilation with time-optimal multi-qubit gates

P. Baßler[1], M. Zipper[1], C. Cedzich[1], M. Heinrich[1], P. H. Huber[2], M. Johanning[2], and M. Kliesch[1,3]

[1] Institute for Theoretical Physics, Heinrich-Heine-Universität Düsseldorf, Germany

[2] Department of Physics, School of Science and Technology, University of Siegen, Germany

[3] Institute for Quantum and Quantum Inspired Computing, Hamburg University of Technology, Germany

We develop a method to synthesize a class of entangling multi-qubit gates for a quantum computing platform with fixed Ising-type interaction with all-to-all connectivity. The only requirement on the flexibility of the interaction is that it can be switched on and off for individual qubits. Our method yields a time-optimal implementation of the multi-qubit gates. We numerically demonstrate that the total multi-qubit gate time scales approximately linear in the number of qubits. Using this gate synthesis as a subroutine, we provide compilation strategies for important use cases: (i) we show that any Clifford circuit on n qubits can be implemented using at most $2n$ multi-qubit gates without requiring ancilla qubits, (ii) we decompose the quantum Fourier transform in a similar fashion, (iii) we compile a simulation of molecular dynamics, and (iv) we propose a method for the compilation of diagonal unitaries with time-optimal multi-qubit gates, as a step towards general unitaries. As motivation, we provide a detailed discussion on a microwave controlled ion trap architecture with magnetic gradient-induced coupling (MAGIC) for the generation of the Ising-type interactions.

1 Introduction

In order to run a program on any computing platform, it is necessary to decompose its higher-level logical operations into more elementary ones and eventually translate those into the platform's native instruction set. This process is called "compiling." Both for classical and quantum computers, this is a non-trivial task. The performance of the compiled program depends not only on the optimizations done by the compiler but also on the available instructions and their implementation.

Especially in the era of noisy and intermediate-scale quantum (NISQ) devices, quantum algorithms are limited by the coherence time of the noisy qubits and the number of noisy gates needed to run them [1]. Thus, it is imperative not only to improve the current quantum devices but also to design fast gates and optimized compilers that use the specific architecture's peculiarities to reduce the circuit depth. Moreover, these endeavors help to reduce the noise-levels of physical gates and are thus also important for reducing the overhead in quantum error correction [2].

Quantum compilation is further complicated by the fact that the type and performance of the native instructions depend severely on the available physical interactions and the extent to which they can be controlled. Most of the compiling literature has focused on native instructions given by single and two-qubit gates. Two-qubit gates are arguably the simplest entangling gates that can be experimentally realized and dominate in most quantum computing architectures, such as supercomputing qubits.

In contrast, ion trap quantum computers naturally involve all-to-all interactions and thus allow for the realization of multi-qubit gates which entangle multiple qubits simultaneously [3, 4].

P. Baßler: bassler@hhu.de
M. Kliesch: science@mkliesch.eu

Consequently, there has been a growing interest in studying compilation with multi-qubit gates, and advantages over the use of two-qubit gates have been demonstrated [5–10].

The experimental realization of multi-qubit gates in ion trap quantum computers remains an active field of research. Recent proof-of-principle experiments have demonstrated such gates acting on up to 10 qubits [11–13]. These rely on precomputing and controlling rather complicated laser pulse shapes to physically implement the desired interactions.

In this work, we propose a simple method that uses *some* all-to-all interaction to emulate arbitrary couplings. We use this idea to synthesize time-optimal multi-qubit gates under minimal experimental setup and control hardware assumptions.

Concretely, we consider a quantum computing platform that satisfies the following requirements:

(I) single-qubit rotations can be executed in parallel, and

(II) it offers Ising-type interactions with all-to-all connectivity.

We also develop compilation strategies with these gates, for which we additionally require that

(III) there is a way to exclude specific qubits from participating in the interaction.

This assumption is sufficient to guarantee that unitaries can be compiled in a circuit depth depending only on the size of their support.

The requirements (I)–(III) are satisfied, for example, in ion trap platforms [11–13]. The motivation for our research originates from working with an ion trap where all gate control is based on microwave pulses and where Ising-type interactions with all-to-all connectivity are mediated through magnetic gradient-induced coupling (MAGIC) [14–21], see Section 1.4.

For concreteness, we assume that all Ising interactions are of ZZ type, and we call the multi-qubit gates generated by arbitrary ZZ couplings 'GZZ *gates*'. Furthermore, by requirement (II), there is an Ising Hamiltonian H with fixed ZZ interactions We then present a synthesis method which realizes an arbitrary GZZ gate as a sequence of time evolutions under H, interleaved with suitable X layers. The purpose of these X layers is to temporarily flip the signs of some ZZ coupling terms in H to accumulate the desired coupling over the sequence. We show that such a sequence can always be found and use a linear program (LP) to find a time-optimal realization of the desired GZZ gate. The resulting gate time scales approximately linear with the number of participating qubits n and requires at most $n(n-1)/2$ X layers.

This method may produce very short evolution times that can lead to problematically crammed single-qubit rotations in practice. We address this issue with a variation of our approach that extends the LP to a mixed integer program (MIP).

We proceed by developing several compiling strategies with GZZ gates. We show that any Clifford circuit on n qubits can be implemented using at most $n+1$ GZZ gates, n two-qubit gates and few single-qubit gates. As an example for non-Clifford unitaries, we decompose the quantum Fourier transform (QFT) in a similar fashion into $n/2$ GZZ gates, $n/2$ two-qubit gates and single-qubit gates. An important application of quantum computers is the simulation of molecular dynamics. We present a method to tailor the approximate simulation in Ref. [22] to our setup by compiling layers of Givens rotations into time-optimal GZZ gates. This method significantly reduces the required number of single-qubit rotations with arbitrary small angles, which are challenging to implement in practice. Moreover, we propose a compilation method for diagonal unitaries as a step toward compilation strategies for general unitaries.

1.1 Comparison to previous works

Synthesis of multi-qubit gates. Previous works [11–13] have mainly focused on implementing multi-qubit gates on ion trap quantum computers using the laser-controlled Mølmer-Sørensen (MS) mechanism [23–25]. This setup requires segmented, amplitude-modulated laser pulses, the shape of which can be efficiently precomputed using the efficient, arbitrary, simultaneously entangling (EASE) gate implementation [13].

Here, the novelty of our work is that we only require the engineering of a single, fixed Ising Hamiltonian, which can be calibrated and fine-tuned to high accuracy. This situation can be found in MAGIC ion traps [14–21] but may also serve as a practical design principle for other

architectures. With our synthesis method, multi-qubit GZZ gates can be realized using only additional X gates, resulting in a sequential series of simple pulses. Arguably, this requires less fine-grained control of the pulse shapes than the EASE gate protocol [13] and may thus be more implementation-friendly. However, we leave a detailed comparison of the approaches for future experimental work.

Furthermore, we introduce *gate time*, instead of gate count, as the central metric for our synthesis of multi-qubit gates. This metric is meaningful, especially for NISQ devices, since the execution of circuits is limited by the coherence time of the qubits. As we show, a side effect of our method is that it also produces rather short circuits, but not necessarily the shortest ones. From our numerical studies, we expect that the gate time of our approach scales at most linear with the number of *participating* qubits n. Hence, we expect our method to produce faster multi-qubit gates than the EASE gate protocol, which additionally scales linearly in the *total* number of qubits N.

A conceptually related approach to our synthesis method was presented in Ref. [26] in the context of digital-analog quantum computing (DAQC). There, the gate synthesis is based on solving a system of linear equations and is inherently restricted to X layers acting on at most two qubits. In contrast, our work optimizes for the total gate time of the sequence and, to this end, allows for layers with arbitrary support. In this way, we avoid the problem of negative times encountered in Ref. [26] and observe a gate time scaling approximately linear in n, in contrast to the quadratic scaling in Ref. [26].

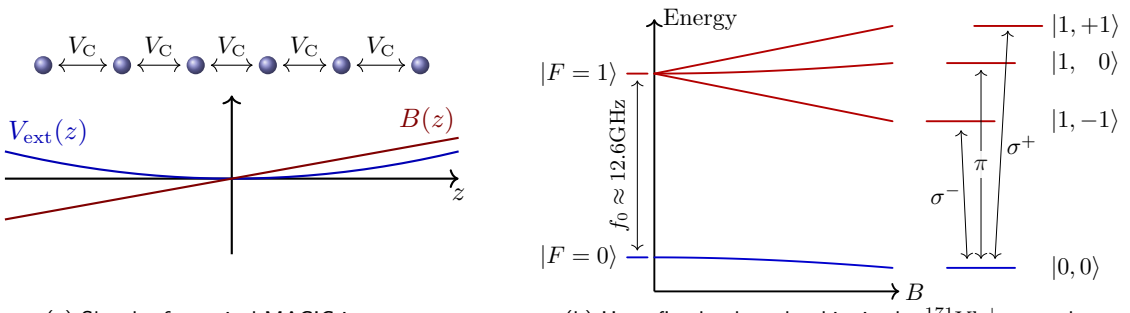
Compilation with GZZ gates. A strategy to decompose general unitaries with multi-qubit GZZ gates is presented in Ref. [6]. It is based on maximizing the fidelity while using as few multi-qubit gates as possible. This optimization is computationally costly, so the numerical results in Ref. [6] cover only up to 4 qubits.

Different compiling strategies with multi-qubit GZZ gates have recently been investigated for Clifford unitaries. In Ref. [7], an implementation with $12n - 18$ GZZ gates is reported, which has been improved to $6n-8$ GZZ gates in Ref. [8]. Subsequently, it was shown that $6\log(n)+O(1)$ GZZ gates are enough if $n/2$ ancillary qubits are used [9]. Here, our ancilla-free approach reduces the prefactor because it requires at most $n+1$ multi-qubit GZZ gates and n two-qubit gates. Shortly after publishing the preprint of this work, it was shown in Ref. [10] that any Clifford unitary on n qubits can, in fact, be implemented with at most 26 so-called GCZ gates which are equivalent to GZZ up to single-qubit Z rotations. In Ref. [10], the authors also pointed out that the results in Ref. [27] can be used to obtain an ancilla-free implementation with $2\log(n) + O(1)$ GZZ gates. The constant-depth scheme of Ref. [10] can be readily combined with the time-optimal synthesis of GZZ gates discussed in Section 2 to show that any Clifford unitary can be realized in linear time on a platform satisfying the requirements (I)–(III). For large n this would further reduce the number of required GZZ gates. However, for small $n \leq 13$ the compilation method presented in Section 3.2 may still be advantageous.

Refs. [19, 28] propose a hand-tailored implementation of the quantum Fourier transform on three qubits that uses simultaneous Ising-type interactions to achieve a speed-up. We use the same interactions, but our scheme can be applied to systems of arbitrary size and employs our time-optimal multi-qubit gates (cf. Section 3.3.)

1.2 Outline

The remainder of the paper is structured as follows: We close this introductory section with a brief introduction to the computational primitives obtained from the requirements (I)–(III) and how these are realized in microwave controlled ion traps with MAGIC. In Section 2, we introduce our GZZ gate and the time-optimal gate synthesis method. Also, we define the MIP to solve the problem of too short Ising-evolution times and support our claim of time-optimality with numerical results. In Section 3 we present compiling strategies with our multi-qubit gate for Clifford circuits, the QFT, molecular dynamics, and general diagonal unitaries. Moreover, we demonstrate the performance of these compilation schemes with numerical results for the Clifford circuits and the QFT.



(a) Sketch of a typical MAGIC ion trap

(b) Hyperfine levels and qubits in the $^{171}\mathrm{Yb}^+$ ground state

Figure 1: Schematics of a MAGIC ion trap. **Left:** six ions in linear configuration, with the plot below indicating the confining trap potential V_{ext} in axial direction and magnetic gradient field B. There is Coulomb repulsion between any two ions (only indicated for next neighbors). **Right:** the transition frequencies of the hyperfine sublevels of the $^{171}\mathrm{Yb}^+$ ground statedepend on the magnetic field strength (Breit-Rabi diagram [29], Zeeman effect exaggerated). The transitions on the right correspond to σ^-, π and σ^+ qubit, respectively.

1.3 Computational primitives

On the abstract quantum computing platform with N qubits specified by the requirements (I)–(III) above, interactions between the qubits are generated by the Ising-type Hamiltonian

$$H_0 \coloneqq -\frac{1}{2}\boldsymbol{Z}^T J \boldsymbol{Z} = -\sum_{i<j}^{N} J_{ij} Z_i Z_j, \tag{1.1}$$

where the vector $\boldsymbol{Z} = (Z_1, \ldots, Z_N)^T$ collects all local Pauli Z operators. The *coupling matrix* $J \in \mathbb{R}^{N\times N}$ is a symmetric matrix which encapsulates the physical properties of the platform. Since the diagonal entries of J merely generate a global and hence unobservable phase, we henceforth assume that J has vanishing diagonal. By requirement (III), we can assume that H_0 acts only on the $n \leq N$ relevant qubits, and we thus assume w.l.o.g. that J is a $n \times n$ matrix.

Letting the system evolve under the Hamiltonian (1.1) for some time t generates a unitary operation on the qubits. Our gate synthesis method is based on the observation that layers of local X gates can be used to emulate the evolution under a modified Hamiltonian: For any binary vector $\boldsymbol{s} \in \mathbb{F}_2^n$, set

$$\mathsf{X}^{\boldsymbol{s}} \coloneqq \bigotimes_{i=1}^{n} \mathsf{X}^{s_i}, \tag{1.2}$$

and define the *qubit encoding* $\boldsymbol{m} = (-1)^{\boldsymbol{s}} \in \{-1,+1\}^n$ (to be understood entry-wise). We then have the following modified time evolution:

$$\mathsf{X}^{\boldsymbol{s}} \exp(-\mathrm{i}tH_0)\, \mathsf{X}^{\boldsymbol{s}} = \exp\big(-\mathrm{i}tH(\boldsymbol{m})\big). \tag{1.3}$$

Here, $H(\boldsymbol{m}) \coloneqq -\frac{1}{2}\boldsymbol{Z}^T J(\boldsymbol{m}) \boldsymbol{Z}$ is the modified Hamiltonian with coupling matrix $J(\boldsymbol{m}) \coloneqq J \circ \boldsymbol{m}\boldsymbol{m}^T$, and $\circ$ denotes the Hadamard (entry-wise) product.

If we apply multiple time evolution operators with different encodings in succession, we can further simplify this scheme. For two encodings $\boldsymbol{m} = (-1)^{\boldsymbol{s}}$ and $\boldsymbol{m}' = (-1)^{\boldsymbol{s}'}$, we can combine the adjacent X layers in Eq. (1.3) and obtain

$$\mathsf{X}^{\boldsymbol{s}}\, \mathrm{e}^{-\mathrm{i}tH_0}\, \mathsf{X}^{\boldsymbol{s}}\, \mathsf{X}^{\boldsymbol{s}'}\, \mathrm{e}^{-\mathrm{i}t'H_0}\, \mathsf{X}^{\boldsymbol{s}'} = \mathsf{X}^{\boldsymbol{s}}\, \mathrm{e}^{-\mathrm{i}tH_0}\, \mathsf{X}^{\boldsymbol{s}\oplus\boldsymbol{s}'}\, \mathrm{e}^{-\mathrm{i}t'H_0}\, \mathsf{X}^{\boldsymbol{s}'}. \tag{1.4}$$

Hence, a change of encoding can be performed with a number of X gates equal to the number of sign flips needed to obtain $\boldsymbol{m}'$ from $\boldsymbol{m}$. The total number of X layers needed to traverse a sequence of encodings is only one more than the length of the sequence.

1.4 Experimental motivation: Ion trap quantum computing with microwaves

Let us give a brief overview of a physical platform on which our computational primitives can be realized. For details, we refer the reader to Appendix A and Ref. [15].

The energy difference between hyperfine sublevels of some atomic state typically falls into the microwave regime of the electromagnetic spectrum, which makes pairs of such hyperfine states natural candidates for microwave-controlled qubits. For example, the "ground state" of ions with nuclear spin $I = \frac{1}{2}$ and total electron angular momentum $J = \frac{1}{2}$ (e.g. Ytterbium-171 ions, $^{171}\mathrm{Yb}^+$) exhibits four hyperfine sublevels. They group into a singlet with $F = 0$ and a triplet with $F = 1$, where F is the quantum number specifying the magnitude of total angular momentum $\boldsymbol{F} = \boldsymbol{I} + \boldsymbol{J}$. The triplet states are energetically degenerate, but can be distinguished by their value of the magnetic quantum number $m_F \in \{-1, 0, +1\}$. The non-degenerate singlet state has $m_F = 0$. There is an energy gap between the multiplets, which for $^{171}\mathrm{Yb}^+$ corresponds to a microwave frequency of about 12.6GHz, see also Figure 1b.

We use one ion to implement a single qubit and choose the singlet state as the computational zero state $|0\rangle \coloneqq |F = 0, m_F = 0\rangle$. We then have the freedom to encode the computational one state $|1\rangle$ into any of the triplet states, and indicate this by the magnetic quantum number $m_F \in \{-1, 0, +1\}$ of the chosen $|1\rangle \coloneqq |F = 1, m_F\rangle$.

In ion traps, magnetic fields can be used to lift the degeneracy of the triplet through the Zeeman effect, see Figure 1b. This separates the different qubit encodings in frequency space and enables single-qubit operations through microwave-driven two-level Rabi oscillations. Certain sequences of pulses on different transitions in the multilevel system can also be used to change the qubit encoding coherently (see Appendix A). However, this possibility only plays a minor role in our analysis, as we will explain below.

We now extend our scope to multiple ions in the same trap. They are stored in a linear configuration and form a "Coulomb crystal" due to their mutual repulsion. In the MAGIC scheme, the eponymous magnetic gradient along the crystal axis leads to different field strengths for the different ion positions, see Figure 1a. The consequence are different Zeeman splittings, which make the qubits distinguishable in frequency space. Thus, addressability is achieved, although the microwaves cannot be focused onto single ions. Additionally, the ions experience a "dipole force" in the inhomogeneous field, which couples internal and external degrees of freedom (s. Appendix A). This effect can be interpreted as an Ising-like interaction between the qubits, which we use in this work to generate multi-qubit gates.

To sum it up, the abstract requirements (I)–(III) are realized in microwave-driven ion traps exposed to inhomogeneous magnetic fields as follows: (I) single qubit rotations are realized by microwave-driven Rabi oscillations which can be executed in parallel through digitally generated microwave signals [30]. (II) the Ising-type interaction is the natural interaction in this setup. (III) selected ions can be taken out of the interaction by encoding them into the magnetic insensitive state with $m_F = 0$.

2 Synthesizing multi-qubit gates with Ising-type interactions

In this section, we investigate the set of gates which is generated by all possible time evolution operators of the Hamiltonians $H(\boldsymbol{m})$ defined in Eq. (1.1). Given time steps $\lambda_{\boldsymbol{m}} \geq 0$ during which the encoding $\boldsymbol{m}$ is used, we thus consider unitaries of the form

$$\prod_{\boldsymbol{m}} \mathrm{e}^{-\mathrm{i}\lambda_{\boldsymbol{m}} H(\boldsymbol{m})} = \mathrm{e}^{-\mathrm{i}\sum_{\boldsymbol{m}} \lambda_{\boldsymbol{m}} H(\boldsymbol{m})}, \tag{2.1}$$

where we used that the diagonal Hamiltonians $H(\boldsymbol{m})$ mutually commute. For all possible encodings $\boldsymbol{m} \in \{-1, +1\}^n$ we collect the time steps $\lambda_{\boldsymbol{m}}$ in a vector $\boldsymbol{\lambda} \in \mathbb{R}^{2^n}$ and interpret $t = \sum_{\boldsymbol{m}} \lambda_{\boldsymbol{m}}$ as the total time of the unitary $\mathrm{e}^{-\mathrm{i}H}$.

We interpret the generated unitary as the time evolution operator under the *total Hamiltonian*

$$H \coloneqq -\frac{1}{2} \boldsymbol{Z}^T A \boldsymbol{Z}\,, \tag{2.2}$$

where we defined the *total coupling matrix*

$$A \coloneqq \sum_{\boldsymbol{m}} \lambda_{\boldsymbol{m}} J(\boldsymbol{m}) = J \circ \sum_{\boldsymbol{m}} \lambda_{\boldsymbol{m}} \boldsymbol{m}\boldsymbol{m}^T \tag{2.3}$$

and used the linearity of the Hadamard product.

Since the $\boldsymbol{m}\boldsymbol{m}^T$ are symmetric, A inherits the symmetry and the vanishing of the diagonal of J, see Section 1.3 and Appendix A. We wish to make our description of the coupling matrix independent of the platform dependent details given by J. Therefore, we define the Hadamard quotient M with entries

$$M_{ij} \coloneqq \begin{cases} A_{ij}/J_{ij}, & i \neq j\,, \\ 0, & i = j\,. \end{cases} \tag{2.4}$$

The implicit assumption that J has no vanishing non-diagonal entries is commonly met by experiments. Our objective is to minimize the total gate time and the amount of $\boldsymbol{m}$'s needed to express the matrix M. To this end we formulate the following linear program (LP):

$$\begin{aligned} \text{minimize} \quad & \mathbf{1}^T\boldsymbol{\lambda} \\ \text{subject to} \quad & M = \sum_{\boldsymbol{m}} \lambda_{\boldsymbol{m}} \boldsymbol{m}\boldsymbol{m}^T, \\ & \boldsymbol{\lambda} \in \mathbb{R}_{\geq 0}^{2^{n-1}}, \\ & \boldsymbol{m} \in \{-1,+1\}^n. \end{aligned} \tag{2.5}$$

As above, $\mathbf{1} = (1,1,\ldots,1)$ is the all-ones vector such that $\mathbf{1}^T\boldsymbol{\lambda} = \sum_{\boldsymbol{m}} \lambda_{\boldsymbol{m}}$. Moreover, we use the symmetry $(-\boldsymbol{m})(-\boldsymbol{m})^T = \boldsymbol{m}\boldsymbol{m}^T$ to reduce the degree of freedom in $\boldsymbol{\lambda}$ from 2^n to 2^{n-1}.

This LP has the form of a ℓ_1-norm minimization over the non-negative vector $\boldsymbol{\lambda}$. As such, it is a convex relaxation of minimizing the number of non-zero entries of $\boldsymbol{\lambda}$, sometimes called the ℓ_0-"norm". Heuristically, it is thus expected that the LP (2.5) favors sparse solutions. As we shall see shortly in Theorem 2.2, the LP (2.5) always has a *feasible* solution (i.e. there are variables $\boldsymbol{\lambda}$ such that all constraints are satisfied) for any symmetric matrix M with vanishing diagonal. The theory of linear programming then guarantees the existence of an optimal solution with at most $n(n-1)/2$ non-zero entries, see Proposition 2.3.

For any symmetric $n \times n$ matrix A with vanishing diagonal, we define an associated multi-qubit gate $\mathsf{GZZ}(A)$, where GZZ stands for "global ZZ interactions",

$$\mathsf{GZZ}(A) \coloneqq \mathrm{e}^{\mathrm{i}\frac{1}{2}\boldsymbol{Z}^T A \boldsymbol{Z}}\,. \tag{2.6}$$

Here, the decomposition of A is found using the LP (2.5) and involves at most $n(n-1)/2$ different encodings $\boldsymbol{m}$. Recall from Section 1.3, that these encodings can be emulated with suitable X layers and hence $\mathsf{GZZ}(A)$ can be implemented using at most $n(n-1)/2+1$ such layers. We call the exact number of X layers the *encoding cost* of $\mathsf{GZZ}(A)$, and $\mathbf{1}^T\boldsymbol{\lambda}$ the *total* GZZ *time*. For this derivation, we have intentionally been agnostic of the physical details of the ion trap but note that the values of $\lambda_{\boldsymbol{m}}$ and therefore t depend on the (physical) coupling matrix J.

Finally, given an optimal decomposition of A, it is also possible to minimize the total number of X gates needed for the implementation. Since every X gate introduces noise, such a minimization improves the fidelity of GZZ gates in practice. By Eq. (1.4), the number of X gates needed to change the encoding from $\boldsymbol{m}$ to $\boldsymbol{m}'$ is exactly the number of signs in $\boldsymbol{m}$ that have to be flipped to obtain $\boldsymbol{m}'$. Since the resulting gate $\mathrm{e}^{-\mathrm{i}H}$ does not depend on the order of encodings $\boldsymbol{m}$, one can minimize the total number of sign flips over all possible orderings. However, finding an optimal ordering generally corresponds to solve a traveling salesman problem on the support of $\boldsymbol{\lambda}$ and is thus NP-hard [31]. Nevertheless, there are good heuristic algorithms, e.g. Christofides's algorithm introduced in Ref. [32].

Before demonstrating how to use the flexibility and the all-to-all connectivity of GZZ gates for compiling, we discuss some theoretical aspects as well as limitations and extensions of the above approach. We conclude by presenting numerical results for the synthesis of $\mathsf{GZZ}(A)$ gates for randomly chosen matrices A.

2.1 Theoretical aspects

First, we show the existence of a solution for the LP (2.5) via frame theoretic arguments, then we investigate the sparsity of optimal solutions from a geometrical viewpoint. Let us define the

$n(n-1)/2$-dimensional subspace of symmetric matrices with vanishing diagonal by

$$\mathrm{Sym}_0(\mathbb{R}^n) \coloneqq \{M \in \mathrm{Sym}(\mathbb{R}^n) \mid M_{ii} = 0 \ \forall i \in [n]\}\,. \tag{2.7}$$

Moreover, we denote the set of outer products generated by all possible encodings by

$$\mathcal{V} \coloneqq \left\{\boldsymbol{m}\boldsymbol{m}^T \;\middle|\; \boldsymbol{m} \in \{-1,+1\}^n, m_n = +1\right\}. \tag{2.8}$$

Due to the symmetry $\boldsymbol{m}\boldsymbol{m}^T = (-\boldsymbol{m})(-\boldsymbol{m})^T$ we can uniformly fix the value of one of the entries of $\boldsymbol{m}$. We chose the convention $m_n = +1$.

Definition 2.1. *Let V be a (finite-dimensional) Hilbert space. A set of vectors $v_1, \ldots, v_N \in V$ is called a* frame *if their linear span is V. A frame is said to be* tight *if there exists $a > 0$ such that for all $v \in V$*

$$a\|v\|^2 = \sum_{i=1}^{N} |\langle v, v_i\rangle|^2\,. \tag{2.9}$$

Moreover, a frame is said to be balanced *if $\sum_{i=1}^{N} v_i = 0$.*

With this definition, we obtain the following:

Theorem 2.2. *The set $\mathcal{V}$ is a balanced tight frame for $\mathrm{Sym}_0(\mathbb{R}^n)$. In particular, the LP (2.5) has a feasible solution for any $M \in \mathrm{Sym}_0(\mathbb{R}^n)$.*

The proof that $\mathcal{V}$ is a balanced tight frame can be found in Appendix B, along with other properties of $\mathcal{V}$. Since $\mathcal{V}$ is a frame for $\mathrm{Sym}_0(\mathbb{R}^n)$, any matrix $M \in \mathrm{Sym}_0(\mathbb{R}^n)$ can be decomposed as $M = \sum_{\boldsymbol{m}} \lambda_{\boldsymbol{m}} \boldsymbol{m}\boldsymbol{m}^T$. In other words, the LP (2.5) has a feasible solution for any $M \in \mathrm{Sym}_0(\mathbb{R}^n)$. Then, a standard linear programming argument shows that there is always an optimal solution with sparsity at most $n(n-1)/2$. Such a solution can be numerically found by using variants of the simplex algorithm (see e.g. Ref. [33] for more details). We formulate this fact as the following proposition and defer the proof to Appendix C, where we also show some geometric properties of optimal solutions of a more general class of LP's.

Proposition 2.3 (Sparsity of optimal solutions)**.** *There exists an optimal solution to the LP (2.5) with sparsity $\leq n(n-1)/2$ for every $M \in \mathrm{Sym}_0(\mathbb{R}^n)$. The simplex algorithm is guaranteed to return such an optimal solution.*

2.2 Practical limitations

In the previous section we showed how to implement a GZZ gate through the Ising-evolution time under at most $n(n-1)/2$ different encodings. However, in an actual ion trap, practical limitations might occur for very short evolution times. For this work, we neglect the potential error introduced by a finite recoding time, i.e. the time needed to perform X gates. During recoding we have additional Hamiltonian terms corresponding to the X gates simultaneously with the "always-on" Ising Hamiltonian (1.1). These cause unwanted effects and the introduced errors become non-negligible when the Ising-evolution time approaches the recoding time, see also Appendix A.3. Below we observe in our numerical results for the LP that some $\lambda_{\boldsymbol{m}}$ are below that recoding time. In this section, we address these problems and propose extensions to our approach to mitigate them. First, we discuss the amount of error we would make in an appropriate norm by ignoring too small $\lambda_{\boldsymbol{m}}$, i.e. by defining a threshold ε and setting $\lambda_{\boldsymbol{m}} \equiv 0$ if $\lambda_{\boldsymbol{m}} < \varepsilon$. To avoid small evolution times in the first place, we then define a mixed integer program (MIP) which can solve the problem exactly with a lower (and upper) bound on the $\lambda_{\boldsymbol{m}}$.

2.2.1 Truncation error

Suppose the target Hamiltonian H is decomposed as in Eq. (2.2). Given a threshold $\varepsilon > 0$ for the $\lambda_{\boldsymbol{m}}$, we define $C \coloneqq \{\boldsymbol{m} \mid \lambda_{\boldsymbol{m}} \leq \varepsilon\}$ and approximate H by the Hamiltonian

$$H' = -\sum_{\boldsymbol{m} \notin C} \lambda_{\boldsymbol{m}} \boldsymbol{Z}^T (J \circ \boldsymbol{m}\boldsymbol{m}^T) \boldsymbol{Z}\,. \tag{2.10}$$

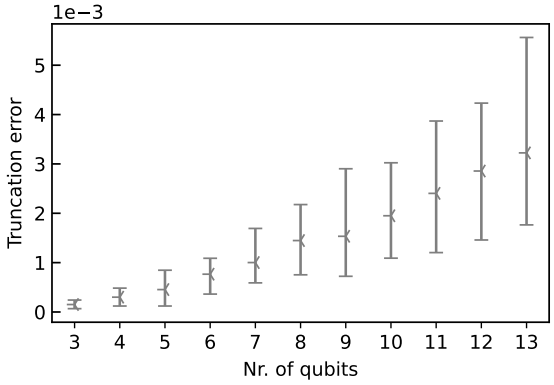


Figure 2: The error scaling, from Eq. (2.11), introduced by truncating all $\lambda_{\boldsymbol{m}} < \varepsilon_l = 27\mu$s from the solution of the LP is displayed. The error bars show the minimum/maximum deviation from the mean over 20 randomly sampled binary A matrices.

The diamond norm error made by replacing the time evolution operator of H with the one of H', is upper bounded by half their spectral norm distance (see e.g. Ref. [34]):

$$\frac{1}{2}\left\|\mathrm{e}^{-\mathrm{i}H} - \mathrm{e}^{-\mathrm{i}H'}\right\|_\infty = \max_{x\in\mathbb{F}_2^n}\left|\sin\left(\frac{1}{2}(H_{x,x} - H'_{x,x})\right)\right|. \tag{2.11}$$

Here, we used that H and H' are diagonal in the computational basis with diagonal entries $H_{x,x} \coloneqq \langle x|H|x\rangle$ and, moreover, that $|1 - \mathrm{e}^{\mathrm{i}\varphi}| = 2\,|\sin(\varphi/2)|$. The difference of the diagonal entries is

$$H_{x,x} - H'_{x,x} = -\sum_{i\neq j} J_{ij} \sum_{\boldsymbol{m}\in C} \lambda_{\boldsymbol{m}} m_i m_j (-1)^{x_i + x_j}. \tag{2.12}$$

Since $|\sin(\theta)| \leq |\theta|$ for all $\theta \in \mathbb{R}$, we find that

$$\frac{1}{2}\left\|\mathrm{e}^{-\mathrm{i}H} - \mathrm{e}^{-\mathrm{i}H'}\right\|_\infty \leq \frac{1}{4}\sum_{i\neq j} |J_{ij}| \sum_{\boldsymbol{m}\in C} \lambda_{\boldsymbol{m}}\,. \tag{2.13}$$

Hence, the truncation error scales with the total truncated time and the interaction strength. Both depend on the number of participating qubits n, however not in a straightforward way. The scaling of the truncation error under realistic assumptions is showed in Figure 2.

2.2.2 The mixed integer program approach

To avoid small $\lambda_{\boldsymbol{m}}$ without introducing an additional error as above, we propose to add a lower bound on their values to the LP (2.5). To this end, we introduce additional binary variables $\boldsymbol{b}$, as in Ref. [35], which renders the optimization into the following mixed integer program (MIP):

$$\begin{aligned}
\text{minimize} \quad & \alpha \mathbf{1}^T \boldsymbol{\lambda} + (1-\alpha)\mathbf{1}^T \boldsymbol{b} \\
\text{subject to} \quad & M = \sum_{\boldsymbol{m}} \lambda_{\boldsymbol{m}} \boldsymbol{m}\boldsymbol{m}^T, \\
& \varepsilon_l \boldsymbol{b} \leq \boldsymbol{\lambda} \leq \varepsilon_u \boldsymbol{b}, \\
& \boldsymbol{\lambda} \in \mathbb{R}_{\geq 0}^{2^{n-1}}, \\
& \boldsymbol{m} \in \{-1,+1\}^n, \\
& \boldsymbol{b} \in \mathbb{F}_2^{2^{n-1}}.
\end{aligned} \tag{2.14}$$

Here $0 \leq \varepsilon_l < \varepsilon_u$ are bounds on the entries of $\boldsymbol{\lambda}$: The lower bound ε_l can be set to the minimal Ising-evolution time which can be realized in practice. The upper bound ε_u can be chosen freely, but small values of ε_u are favorable since the runtime of the MIP is generally shorter for smaller intervals $[\varepsilon_l, \varepsilon_u]$ as explained in Ref. [35]. Since $\max(\boldsymbol{\lambda})$ depends on $\max_{i<j}(M_{ij})$ one can select $\varepsilon_u \propto \max_{i<j}(M_{ij})$. The parameter $\alpha \in [0,1]$ is used to assign weights to the optimization of $\mathbf{1}^T\boldsymbol{b}$ (sparsity) and $\mathbf{1}^T\boldsymbol{\lambda}$ (total GZZ time), respectively.

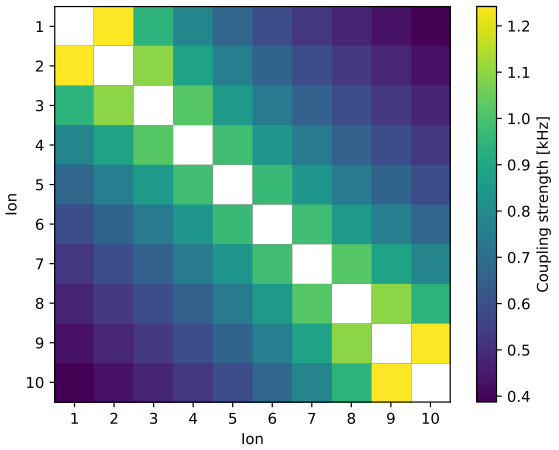


Figure 3: An example coupling matrix J for an ion trap with 10 ions and harmonic trap potential. J is determined by the physical parameters of the trap, see Section 2.3 for the concrete values, and can be computed by Eq. (A.12).

2.3 Numerical results

We investigate how well the proposed methods for the synthesis of GZZ(A) gates performs in practice. To this end, we solve the LP (2.5) and the MIP (2.14) numerically for randomly chosen matrices A and compare the solutions to a naive approach. As explicated above, our goal is to minimize the total GZZ time $\mathbf{1}^T\boldsymbol{\lambda}$ subject to $A/J = M = \sum_{\boldsymbol{m}} \lambda_{\boldsymbol{m}} \boldsymbol{m}\boldsymbol{m}^T$. The numerical results in this section show the performance of GZZ gates on all qubits, i.e. $n = N$.

To demonstrate the performance of our approach in a realistic setting we consider the ion trap architecture of Appendix A with a harmonic trap potential. Concretely, we take Ytterbium $^{171}\mathrm{Yb}^+$ ions with Rabi frequency $\Omega = 2\pi\,100\mathrm{kHz}$, magnetic field gradient $B_1 = 100\mathrm{T/m}$ and axial trap frequency $\omega_z = 2\pi\,100\mathrm{kHz}$. This determines the coupling matrix J via Eq. (A.12), see Ref. [36] for more details. An example coupling matrix J for 10 ions is shown in Figure 3. We made the Python code for its computation available on GitHub [37].

On a logical level, we consider a symmetric binary matrix $A \in \mathrm{Sym}_0(\mathbb{F}_2^n)$ defined in Eq. (2.3), which indicates where the ZZ gates are located: For all $i < j$, $A_{ij} = 1$ if there is a ZZ gate between qubits i and j. We simulate a random ZZ gate layer by sampling the entries of the lower/upper triangular part of A uniformly from $\{0, 1\}$.

We compare the results of the LP and the MIP with a "naive approach", which corresponds to a sequential execution of the ZZ gates. As before, we neglect the gate time for single-qubit gates. Moreover, "total gate time" refers to the total GZZ time, i.e. $\mathbf{1}^T\boldsymbol{\lambda}$, for the LP and the MIP. For the naive approach, the "total gate time" is the time needed to execute the sequence of ZZ gates, i.e. $\sum_{i<j} A_{ij}/J_{ij}$, and the encoding cost is the number of ZZ gates $\sum_{i<j} A_{ij}$.

Figure 4 shows the numerical results for solving the LP (2.5) and the MIP (2.14). The encoding cost for the naive approach scales roughly with $n^2/4$ since on average half of the randomly chosen $n(n-1)/2$ entries of the lower triangular part of A do not vanish. In Figure 4 we find that the total GZZ time scales linearly with the number of participating qubits. In contrast, by Proposition 2.3 the number of encodings needed to implement a GZZ gate increases quadratically with the number of participating qubits. Thus, the time between the encodings becomes shorter and shorter, which results in arbitrarily small evolution times $\lambda_{\boldsymbol{m}}$ for many qubits, see also Section 2.2. Hence, the more qubits we consider, the more solutions $\lambda_{\boldsymbol{m}}$ of the LP are smaller than the lower bound ε_l, see Figure 2. This explains the deviation of the encoding cost of the MIP from that of the LP.

Figure 4 also shows that the MIP can be solved in practice and, similar to the LP, yields nearly time-optimal GZZ gates. Thus, even when taking practical limitations into account, GZZ gates can be implemented in linear time and an encoding cost of approximately $n(n-1)/2$.

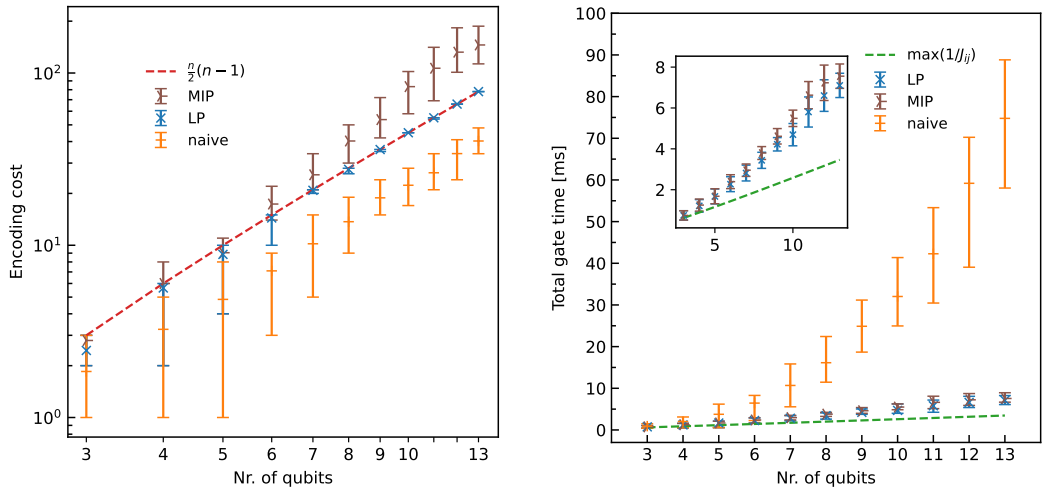


Figure 4: Comparing the performances of the LP and the MIP to the naive approach for the implementation of a random ZZ layer. The error bars show the minimum/maximum deviation from the mean over 20 randomly sampled binary A matrices. **Left:** The encoding cost to implement a random ZZ layer on n qubits. The naive approach only costs half as much as the LP, see Section 2.3. For many qubits, the LP approaches the upper bound (red dashed line) of Proposition 2.3, whereas the MIP exceeds this upper bound. **Right:** The total gate time of the naive approach scales quadratically with the number of qubits due to the quadratic scaling of the number of possible ZZ gates. The total gate time of both the LP and the MIP scales approximately linear. The inset shows a zoomed in version of the plot. One can see that the LP and the MIP roughly take twice as long as the slowest two-qubit ZZ gate (green dashed line).

Details of computer implementation. We use the Python package CVXPY [38, 39] with the GNU linear program kit simplex solver [40] to solve the LP, and the MOSEK solver [41] for the MIP. We change one parameter of MOSEK to improve the runtime at the expense of not finding the optimal solution. Concretely, we set the MOSEK parameter `MSK_DPAR_MIO_TOL_REL_GAP` to 0.6.

For the MIP we choose the lower bound $27\mu\text{s} = \varepsilon_l \leq \lambda_{\boldsymbol{m}}$ for all $\boldsymbol{m}$, which is motivated by the concrete ion trap setup: the duration of a robust X gate is about five times longer than the duration of a X gate (i.e. a π-pulse), which is roughly $\pi/\Omega = 5\mu\text{s}$ [42]. We further pick the upper bound $\varepsilon_u = 3/2 \max_{i<j} |M_{ij}|$. Note that if the interval $[\varepsilon_l, \varepsilon_u]$ is too narrow, we might not be able to find any feasible solution. If it is too wide, the runtime of the solver might increase [35]. We observed in our numerical studies (not shown) that both the total gate time and the encoding cost are essentially constant in α as long as α is not too close to the extremal values 0 and 1. We therefore, deliberately put equal weights on the two terms in the objective function and set $\alpha = 0.5$.

In practice, the simplex algorithm has a runtime which is polynomial in the problem size [43]. Since the LP (2.5) has 2^{n-1} variables, the runtime is exponential in n. For moderate n, this is however still manageable on modern hardware – a Laptop with Intel Core i7 Processor (8x 1.8 GHz) and 16 GB RAM needs on average only 20 seconds to solve the LP for $n = 13$. This is about the size of most ion trap quantum computers nowadays. The runtime of mixed integer programs is exponential in the worst case. Nevertheless, solving MIP (2.14) for $n = 13$ qubits using the MOSEK solver took on average about seven minutes on the same hardware.

An implementation of the LP and MIP used to generate Figure 4 is provided on GitHub [37].

3 Compilation with GZZ gates

The GZZ gates defined in Eq. (2.6) can be used in compilation schemes to improve multi-qubit gate counts over previous results [7, 8, 26]. Our main result is that we can implement any Clifford circuit on n qubits using only $n + 1$ GZZ gates, n two-qubit gates and single-qubit gates.

First, we introduce the notation used in later sections and derive some gate equivalences that will help us later. Then, we study the practically relevant case of compiling global Clifford unitaries. These ubiquitous unitaries play an important role as basic building blocks of quantum circuits, especially in fault-tolerant quantum computing, and are of major importance for cryptographic and tomographic protocols due to their statistical properties. For the compilation of Clifford unitaries, we make use of their Bruhat decomposition, also used in Ref. [44], and compile the entangling layers into GZZ gates. Interestingly, this compilation scheme for entangling Clifford layers can be generalized beyond pure Clifford circuits. As an example, we show how it can be used to compile an n-qubit QFT. Finally, we propose a compiling scheme for general diagonal unitaries which may be used as a step towards decomposing general unitaries.

3.1 Notation

We introduce some notation and link our GZZ gate from Eq. (2.6) to other known entangling gates. Subscripts on gates indicate on which qubits they act. For $\boldsymbol{x} \in \mathbb{F}_2^n$ and $\alpha \in [0, 2\pi)$ we thus have

$$\begin{aligned}
\text{Z rotation:} \quad & \mathsf{R}_\mathsf{Z}(\alpha)_j|\boldsymbol{x}\rangle = \mathrm{e}^{\mathrm{i}\alpha x_j}|\boldsymbol{x}\rangle\,, \\
\text{ZZ gate:} \quad & \mathsf{ZZ}(\alpha)_{i,j}|\boldsymbol{x}\rangle = \mathrm{e}^{\mathrm{i}\alpha(x_i \oplus x_j)}|\boldsymbol{x}\rangle\,, \\
\text{Controlled Z rotation:} \quad & \mathsf{CR}_\mathsf{Z}(\alpha)_{i,j}|\boldsymbol{x}\rangle = \mathrm{e}^{\mathrm{i}\alpha x_i x_j}|\boldsymbol{x}\rangle\,, \\
\text{Controlled X gate:} \quad & \mathsf{CX}_{i,j}|\boldsymbol{x}\rangle = |x_1, \ldots, x_{j-1}, x_j \oplus x_i, x_{j+1}, \ldots, x_n\rangle\,, \\
\text{Hadamard gate:} \quad & \mathsf{H}_j|\boldsymbol{x}\rangle = \frac{1}{\sqrt{2}}|x_1, \ldots, x_{j-1}\rangle\big(|0\rangle + (-1)^{x_j}|1\rangle\big)|x_{j+1}, \ldots, x_n\rangle\,,
\end{aligned} \tag{3.1}$$

where $\oplus$ denotes addition modulo 2. In particular, we denote the phase gate and the controlled-Z gate as

$$\mathsf{S}_j \coloneqq \mathsf{R}_\mathsf{Z}(\pi/2)_j \quad \text{and} \quad \mathsf{CZ}_{i,j} \coloneqq \mathsf{CR}_\mathsf{Z}(\pi)_{i,j}, \tag{3.2}$$

respectively. Note that in the definition of the ZZ gate α can be identified with entries of the matrix A in Eq. (2.3). In terms of integer arithmetic , we have $2xy = x + y - (x \oplus y)$ for $x, y \in \{0, 1\}$ which yields

$$\mathsf{CR}_\mathsf{Z}(\alpha)_{i,j} \equiv \mathsf{R}_\mathsf{Z}(\alpha/2)_i\, \mathsf{R}_\mathsf{Z}(\alpha/2)_j\, \mathsf{ZZ}(-\alpha/2)_{i,j}\,. \tag{3.3}$$

Since the GZZ gate from Eq. (2.6) consists only of ZZ gates, we can express it as

$$\mathsf{GZZ}(A) = \mathrm{e}^{\mathrm{i}a} \prod_{i<j} \mathsf{ZZ}(-2A_{ij})_{i,j}\,, \tag{3.4}$$

where $A \in \mathrm{Sym}_0(\mathbb{R}^n)$ is a symmetric matrix with vanishing diagonal and $a \coloneqq \sum_{i<j} A_{ij}$. Similarly, a layer of arbitrary controlled R_Z rotations is characterized by $A \in \mathrm{Sym}_0(\mathbb{R}^n)$ via

$$\mathsf{GCR}_\mathsf{Z}(A) \coloneqq \prod_{i<j} \mathsf{CR}_\mathsf{Z}(A_{ij})_{i,j} = \mathrm{e}^{-\frac{\mathrm{i}}{4}a}\, \mathsf{GZZ}(A/4) \prod_{i=1}^{n} \mathsf{R}_\mathsf{Z}(b_i/2)_i\,, \tag{3.5}$$

where we used Eq. (3.3) and abbreviated $b_i \coloneqq \sum_j A_{ij}$. A general CX layer is given by

$$\mathsf{GCX}(B)|\boldsymbol{x}\rangle \coloneqq |B\boldsymbol{x}\rangle\,, \tag{3.6}$$

with a matrix $B \in \mathrm{GL}_n(\mathbb{F}_2)$. We call $\mathsf{GCX}(B)$ a *directed* CX *layer* if B is lower or upper triangular.

3.2 Clifford circuits

The *Clifford group* Cl_n is a finite subgroup of the unitary group $\mathrm{U}(2^n)$ that is generated by the single-qubit Hadamard gate H_j and phase gate S_j, as well as the two-qubit $\mathsf{CX}_{i,j}$ gate. Conversely, it is a natural task to decompose an arbitrary Clifford unitary $U \in \mathrm{Cl}_n$ into these generators. This task is solved by a number of algorithms, see e.g. Refs. [45–47], with the same asymptotic gate count, but a differently structured output circuit .

Here, we make use of the so-called "Bruhat decomposition" [44, 46]. This decomposition has the advantage that the entangling gates are grouped either in CZ gate layers or directed CX layers, which both can be directly compiled into GZZ gates. More precisely, the algorithm in Ref. [44] writes any Clifford unitary in the form -X-Z-CX-CZ-S-H-CX-CZ-S-, where

- -X-Z- is a layer of Pauli gates,
- -CX- are layers of directed CX gates,
- -CZ- are layers of controlled Z gates,
- -S- are layers of phase gates S and
- -H- is a layer consisting of Hadamard gates H (and permutation operations).

In the following, we concentrate on the decomposition of the CZ and CX layers as the remaining layers consist of local gates with straightforward implementation. As we show, the compilation of a directed CX layer is a lot more expensive than the compilation of a CZ layer.

As a corollary of CZ layer compilation, we show how to efficiently prepare multi-qubit stabilizer states with GZZ gates, i.e. states of the form $U|0\rangle$ where $U \in \mathrm{Cl}_n$ is a Clifford unitary. Stabilizer states are important e.g. for the construction of mutually unbiased bases, or more generally, informationally complete positive operator valued measures (POVMs), and their preparation is thus of practical relevance for tomographic protocols, see e.g. Refs. [48, 49].

First we transform the directed CX layer by conjugating the CX gate targets with Hadamard gates. Then we use the structure of the resulting gate layer to reduce the encoding cost with the GZZ gate. Furthermore, we use the same method to reduce the encoding cost of the QFT. We underpin the advantages of our method with numerical simulations.

3.2.1 Implementing CZ layers

Since CZ gates commute, we can rewrite any CZ circuit, using Eqs. (3.2) and (3.5), as

$$\begin{aligned}\mathsf{GCZ}(A) &:= \prod_{i<j} \mathsf{CZ}_{i,j}^{A_{ij}} = \prod_{i<j} \mathsf{CR}_\mathsf{Z}(\pi A_{ij})_{i,j} \\ &= \mathsf{GCR}_\mathsf{Z}(\pi A) \\ &= \mathrm{e}^{-\frac{\mathrm{i}\pi}{4}a}\, \mathsf{GZZ}\left(\frac{\pi}{4}A\right) \prod_{i=1}^{n} \mathsf{R}_\mathsf{Z}\left(\frac{\pi}{2}b_i\right)_i \\ &= \mathrm{e}^{-\frac{\mathrm{i}\pi}{4}a}\, \mathsf{GZZ}\left(\frac{\pi}{4}A\right) \prod_{i=1}^{n} \mathsf{S}_i^{b_i},\end{aligned} \tag{3.7}$$

where $A \in \mathrm{Sym}_0(\mathbb{F}^n)$ is again a symmetric matrix with zero diagonal and a and b_i are defined as in Section 3.1. Note that the phase gate S has order 4, so effectively only b_i mod 4 plays a role and the single-qubit gates are from the set $\{\mathsf{I}, \mathsf{S}, \mathsf{Z}, \mathsf{S}^\dagger\}$.

3.2.2 Stabilizer state preparation

Although any stabilizer state can be written as $U|0\rangle$ for some Clifford unitary $U \in \mathrm{Cl}_n$, not the full Clifford group is needed to generate all stabilizer states. In fact, it is well known that any stabilizer state can be obtained by acting with local Clifford gates on *graph states* [50, 51]. Graph states are defined as

$$|A\rangle := \prod_{i<j} \mathsf{CZ}_{i,j}^{A_{ij}} |+^n\rangle, \tag{3.8}$$

where $A \in \mathrm{Sym}_0(\mathbb{F}_2^n)$ and $|+^n\rangle = H^{\otimes n}|0\rangle$. Hence, by the above, any stabilizer state can be prepared by an initial global Hadamard layer, a GZZ gate, and a final layer of single-qubit Clifford gates.

3.2.3 Decomposing directed CX layers

Let us now consider the two directed CX layers in the Bruhat decomposition [44]. With the notation introduced in Eq. (3.6), a *fan-out* gate with the following action on a state $\boldsymbol{x} \in \mathbb{F}_2^n$,

$$\mathsf{GCX}(B_{FO})|\boldsymbol{x}\rangle = |x_1, x_2 \oplus x_1, \ldots, x_n \oplus x_1\rangle\,, \tag{3.9}$$

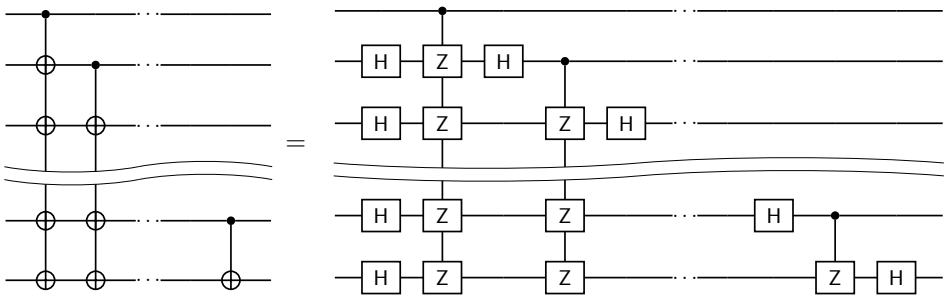
has the binary matrix

$$B_{FO} = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 1 & & & \\ \vdots & & \mathbb{1}_{n-1} & \\ 1 & & & \end{pmatrix}. \tag{3.10}$$

From the structure of the matrix B_{FO} it can be seen that any directed CX layer can be realized by at most $n-1$ fan-out gates. Each fan-out gate is equivalent to a $\mathsf{GCR_Z}$ gate, conjugated with Hadamard gates on the target qubits. Since by Eq. (3.5) each $\mathsf{GCR_Z}$ can be realized with one GZZ gate, we need at most $n-1$ GZZ gates to realize a directed CX layer, see Eq. (3.11) for an example. Concretely, the total encoding cost for a directed CX layer, implementing each of the $n-1$ fan-out gates with one GZZ gate scales as $\mathrm{O}(1/6n^3)$. Below Eq. (3.11) we show that $\lfloor \frac{n-1}{2} \rfloor$ GZZ gates are enough to implement a directed CX layer. Therefore, one requires $n-1$ (if n is odd) or $n-2$ (if n is even) GZZ gates, for the two directed CX layers appearing in the Bruhat decomposition of Ref. [44]. Since we can realize the CZ layer with exactly one GZZ gate, each Clifford circuit requires only $n+1$ or n GZZ gates and only $n-1$ or n two-qubit CZ gates for n odd or n even, respectively.

Fully directed CX layer. We call a directed CX layer *fully directed* if the corresponding gate $\mathsf{GCX}(B)$ is characterized by a $n \times n$ matrix B with zeros in the upper triangular matrix and ones everywhere else. Fully directed CX layers are related to the textbook QFT, see Section 3.3 below.

Remember that we can represent any directed CX layer as a concatenation of $n-1$ fan-out gates. Commuting Hadamard gates through each target of the fan-out gate transforms the controlled X to a controlled Z gate by HXH = Z. This converts a fan-out gate into a CZ-type fan-out gate, which we also call fan-out gate for short. We can thus transform a fully directed CX layer by applying $\mathbb{1} = H^2$ from the left and commuting one of the Hadamard gates to the right until it hits a control:



(3.11)

Note that such a transformation obviously works also for an arbitrary directed CX layer. The locations of the resulting H and CZ gates can be represented as tables

$$T_{\mathsf{H}} = \begin{bmatrix} 0 & 0 & 0 & \cdots & 0 \\ 1 & 1 & 0 & \cdots & 0 \\ & & & & \\ 1 & 0 & 1 & & \vdots \\ \vdots & \vdots & & \ddots & 0 \\ 1 & 0 & \cdots & 0 & 1 \end{bmatrix} \quad \text{and} \quad T_{\mathsf{CZ}} = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 1 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & \cdots & 1 \end{bmatrix}, \tag{3.12}$$

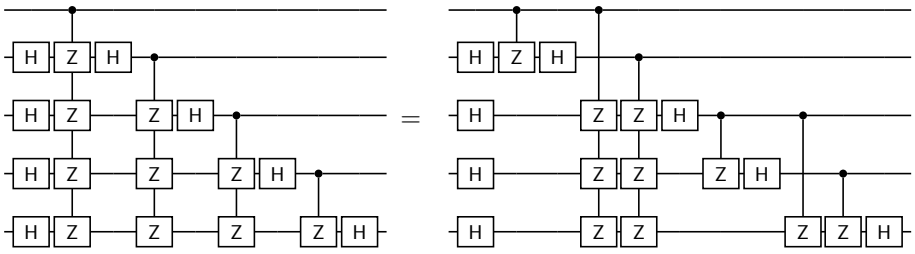
where T_{CZ} has the same form as the matrix B for the fully directed CX layer. The row index of these tables indicates the qubit on which the corresponding gate acts, while the column index

indicates the "time step", i.e. the horizontal position in the circuit diagram. Thus, $T_{\mathsf{CZ}}[i,j] = 1$ if qubit i is either a control or a target of a CZ at time step j, and $T_{\mathsf{CZ}}[i,j] = 0$ if qubit i is idle at time step j. Similarly, $T_{\mathsf{H}}[i,j] = 1$ if a Hadamard gate acts on qubit i at time step j, and $T_{\mathsf{H}}[i,j] = 0$ if qubit i remains unchanged. For a fully directed CX layer, T_{H} and T_{CZ} always have this form. To locate the components of the circuit on the right-hand side of Eq. (3.11) one starts with reading the first column of T_{H}, then the first column of T_{CZ} and so on.

We now aim at reducing the encoding cost of the fully directed CX layer on n qubits. To this end, we reduce the supports m of the GZZ gates implementing the fan-out gates in the CX layer, since the encoding cost of a GZZ gate scales as $m(m-1)/2$, see Section 2. Consider an odd column i in T_{CZ} which corresponds to a CZ-type fan-out gate with control on qubit i. We split column i into two columns as follows: The first one representing a two-qubit CZ gate on qubits i and $i+1$, and the second column is the same as the original one except that it does not target qubit $i+1$. This splitting increases the number of columns in T_{CZ} by one. For example, for $n = 5$ and $i = 1$ we split the column $[1,1,1,1,1]^T$ into $[1,1,0,0,0]^T$ and $[1,0,1,1,1]^T$, where we keep in mind that the first nonzero entry in a column denotes the control qubit and hence has to appear in both parts. Furthermore, we update the Hadamard table T_{H} by inserting a zero column after column $i+1$ to account for the new column in T_{CZ}. The fan-out gate resulting from the split of the odd column i of T_{CZ} together with the even column $i+1$ can be implemented with one GZZ on i qubits.

Note that we can not move parts of columns of T_{CZ} to the left since there is always a Hadamard gate on the left of the control qubit of that column that blocks it. Therefore, we only split odd columns i and move them to the right.

This splitting of columns of T_{CZ} corresponds to moving the Hadamard gate on the $i+1$ qubit to the left or, equivalently, moving all CZ gates except the one acting on the $i+1$ qubit to the right, as exemplified in the following for $n = 5$ qubits:



(3.13)

For general n, the scheme works exactly the same as in this example.

The circuit before the splitting on the left-hand side of Eq. (3.13) can be implemented with $n-1$ GZZ gates each acting on $n,\dots,2$ qubits respectively, where n is the number of qubits of the directed CX layer, and thus have an encoding cost of $\sum_{i=2}^{n} i(i-1)/2 = n^3/6 + \mathrm{O}(n^2)$. On the right-hand side we need $\lceil \frac{n-1}{2} \rceil$ CZ gates and $\lfloor \frac{n-1}{2} \rfloor$ GZZ gates, resulting in an encoding cost of

$$\left\lceil \frac{n-1}{2} \right\rceil + \sum_{i=0}^{\lfloor \frac{n-1}{2} \rfloor} \frac{n-2i}{2}(n-2i-1) = \frac{1}{12}n^3 + \mathrm{O}(n^2)\,. \tag{3.14}$$

The first term comes from the encoding cost of the CZ gates, which is one per CZ gate. The second term comes from combining a GZZ gate on n qubits with a GZZ gate on $n-1$ qubits resulting in a GZZ gate on n qubits. The encoding cost in Eq. (3.14) has the same cubic scaling with n as before combining GZZ gates, but we were able to improve the coefficient from 1/6 to 1/12. Recall from Section 2.3 that the encoding cost of the naive approach scales only quadratic, so we trade higher encoding cost for faster gates. Each CZ gate can be implemented as described in Eq. (3.3) for $\alpha = \pi$ by a single ZZ gate and two additional single-qubit $\mathsf{R_Z}(\pi/2) \equiv \sqrt{\mathsf{Z}} = \mathsf{S}$ gates on the control and target qubit, respectively. Since the S gates do commute with ZZ and GZZ gates but do not commute with Hadamard gates, we can combine most of the S gates to an S^k gate, where $k \in \{0,1,2,3\}$. There are two Hadamard gates on $n-1$ qubits and none on the first qubit, therefore we have $2(n-1)+1$ S^k gates. To summarize, in addition to $\lceil \frac{n-1}{2} \rceil$ CZ gates and $\lfloor \frac{n-1}{2} \rfloor$ GZZ gates we need $2n-1$ S^k gates to implement a fully directed CX layer on n qubits.

Our method optimizes both the encoding cost and the total gate time of the directed CX layer. We expect to lose time-optimality for a fully directed CX layer if we split any GZZ gate into

smaller pieces and therefore reduce the encoding cost. In the extreme case, we would split all GZZ gates into two-qubit ZZ gates and thereby would end up with the naive approach explained in Section 2.3. Our method combines parts of the i-th column of T_{CZ} with the $i+1$-th column to a GZZ gate on $n-i+1$ qubits. One could think that moving parts of the larger GZZ gate farther to the right might improve the encoding cost. But the support of the part left behind, and therefore the encoding cost increases the farther we push the other part to the right. In the extreme case of pushing all parts as far as possible to the right we end up with a "transposed" table where the lower triangular part is zero, and we did not achieve any reduction of the encoding cost.

Algorithm 1 Moving Hadamard gates.

Input: T_{CZ}

Initialize T_{H} as in Eq. (3.12)
$h_{\max} \leftarrow 0$ ▷ Position of the rightmost H that has already been moved left
for $i = 1, \dots, n-1$ **do**
  $T_{\mathsf{H}}[i,i] \leftarrow 0$ ▷ H on i-th qubit leaves its position
  $c \leftarrow \max\{j = 0, \dots, i-1 | T_{\mathsf{CZ}}[i,j] = 1\} + 1$ ▷ Find position directly after first CZ to the left
  if c = `NaN` **then** ▷ No CZ found (max was taken on an empty set)
    $T_{\mathsf{H}}[i,0] \leftarrow 0$ ▷ Cancel H in the first layer
  else if $c = i$ **then** ▷ Unable to move left, attempt to move right
    if $\{j = i, \dots, n-1 | T_{\mathsf{CZ}}[i,j] = 1\} = \emptyset$ **then** ▷ If no CZ is to the right...
      $T_{\mathsf{H}}[i,n-1] \leftarrow 1$ ▷ ...move H to the last layer, ...
    else
      $T_{\mathsf{H}}[i,i] \leftarrow 1$ ▷ ...otherwise remain in place.
  else
    $h_{\max} \leftarrow \max\{h_{\max}, c\}$ ▷ Find the more restrictive condition (either CZ on current qubit or H on previous)
    $T_{\mathsf{H}}[i,h_{\max}] \leftarrow 1$ ▷ Move H to the target layer

Output: T_{H}

Arbitrary directed CX layer. Until now, we considered only fully directed CX layers which in practice is a very special case. More common are arbitrary directed CX layers which corresponds to a $\mathsf{GCX}(B)$ with $B \in \mathbb{F}_2^{n \times n}$ still being lower/upper triangular but more sparse. This sparsity, which translates to the table T_{CZ}, can be used to further reduce the encoding cost. One might be able to move the Hadamard gates to the left/right which changes the support of the GZZ gates. We explain three different scenarios which can be easily verified by the simple structure of T_{CZ} and T_{H}. Obviously, the two Hadamard gates on any qubit $i > 1$ cancel if they are separated only by identities. Similarly, if there is no control or target of a CZ gate to the right of a Hadamard gate H, one can move H to the last position of the circuit (without cancellation). Otherwise, it might still be possible to push the Hadamard gate on qubit i to the left until it hits a target of some CZ gate. However, one needs to be careful to not disrupt any previously generated GZZ gates with qubit i in its support. This can be accomplished by disallowing the Hadamard gate to move across other Hadamard layers. Algorithm 1 implements these moves of the Hadamard gates by updating the table T_{H} accordingly.

Algorithm 2 pools multiple columns of T_{CZ} together into a single GZZ gate similar as for fully directed CX layers. This takes into account the sparsity of T_{CZ} and the positions of the Hadamard gates, i.e. T_{H} generated by Algorithm 1. As for fully directed CX layers, we can split a column of T_{CZ} into a two-qubit CZ gate and a GZZ gate if necessary. The algorithm starts from the left and tries to move columns of T_{CZ}, or parts of it, to the right. The following cases occur: If a column of T_{CZ} has a Hadamard gate left to the first nonzero entry, i.e. the control of the fan-out gate, then it can only be moved to the right. If a column of T_{CZ} has no Hadamard gate left to the first nonzero entry, then this column can be combined with the previous column. If a column has one Hadamard gate to the right, then the column can be split into a two-qubit CZ gate whose target is the qubit on which the Hadamard gate acts, and a GZZ gate which can be moved to the right. Due to the structure of T_{CZ} and T_{H} there is never a Hadamard gate to the right of the first nonzero entry, i.e. the control of the fan-out gate, of a column.

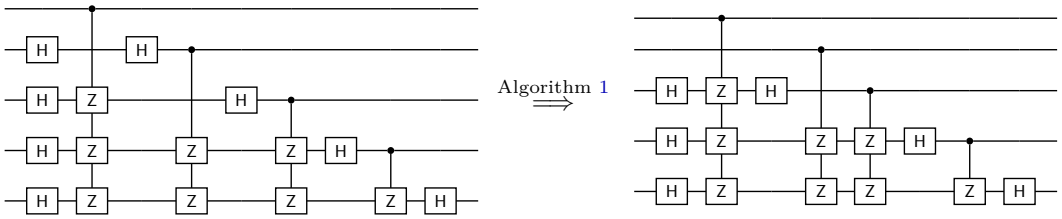
Algorithm 2 Moving CZ gates.

Input: T_{CZ}, T_{H}

for $i = 1, \ldots, n-1$ **do**
  if $T_{\mathsf{H}}[:, i] \wedge T_{\mathsf{CZ}}[:, i]$ has at least one 1 **then** ▷ $T_{\mathsf{CZ}}[:, i]$ has Hadamard gates to the left
    if $T_{\mathsf{H}}[:, i+1]$ has exactly one 1 **then** ▷ $T_{\mathsf{CZ}}[:, i]$ has one Hadamard gate to the right
      Split $T_{\mathsf{CZ}}[:, i]$ into:
      $T^1_{\mathsf{CZ}} := T_{\mathsf{H}}[:, i+1] \oplus e_i$ and ▷ $e_i = [0, \ldots, 0, 1, 0, \ldots, 0]^T$ with 1 at the ith position.
      $T^2_{\mathsf{CZ}} := T_{\mathsf{H}}[:, i+1] \oplus T_{\mathsf{CZ}}[:, i]$
      if T^2_{CZ} has at least two 1's **then** ▷ Check that T^2_{CZ} is not trivial
        Move T^2_{CZ} to the right.
  else ▷ $T_{\mathsf{CZ}}[:, i]$ has no Hadamard gates to the left
    Move $T_{\mathsf{CZ}}[:, i]$ to the left.

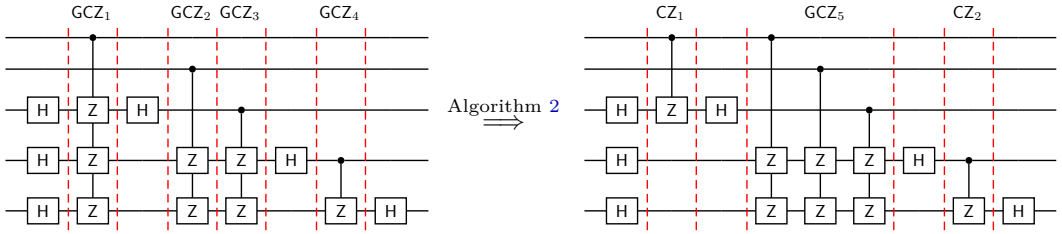
Output: T_{CZ} (modified)

Example. Consider the following circuit implementing an arbitrary directed CX layer on $n = 5$ qubits where the first two fan-out gates do not target qubits 2 and 3, respectively:



Algorithm 1 cancels two Hadamard gates on the second qubit which is not targeted by the first fan-out gate, i.e. $T_{\mathsf{CZ}}[2, 1] = 0$. On the third qubit, Algorithm 1 moves the rightmost Hadamard gate one layer to the left since $T_{\mathsf{CZ}}[3, 2] = 0$ but $T_{\mathsf{CZ}}[3, 1] = 1$.

Algorithm 2 takes the output of Algorithm 1 and splits GCZ_1 into CZ_1 and GCZ'_1 since it has Hadamard gates to the left and right. Then it pools GCZ'_1, GCZ_2 and GCZ_3 together into a single gate GCZ_5:



Note that any GCZ gate is equivalent to a GZZ up to single-qubit $\mathsf{R_Z}$ rotations. The total encoding cost of the compiled circuit is $2 + n/2(n-1) = 12$ whereas the original circuit has the encoding cost $6 + 3 + 3 + 1 = 13$.

In this section we discussed how to implement the entangling operations of a Clifford unitary. We only need one GZZ gate to implement one fan-out gate, as one can see in Eq. (3.5), (3.10) and Hadamard gate commutation, instead of two multi-qubit gates used in the compilation schemes in Refs. [7, 8]. We further showed that we only require $n + 1$ GZZ and n two-qubit CZ gates to implement a Clifford unitary on n qubits. Due to the flexibility of our GZZ gate, we further optimized the implementation of a directed CX layer to reduce the encoding cost.

The compilation of directed CX circuits via Algorithms 1 and 2 is available as a Python implementation on GitHub [37].

3.2.4 Numerical results for the directed CX layer

We demonstrate the performance of Algorithms 1 and 2 for compiling an arbitrary directed CX layer. Since the CX layer is the most costly gate layer, in the Bruhat decomposition we only present

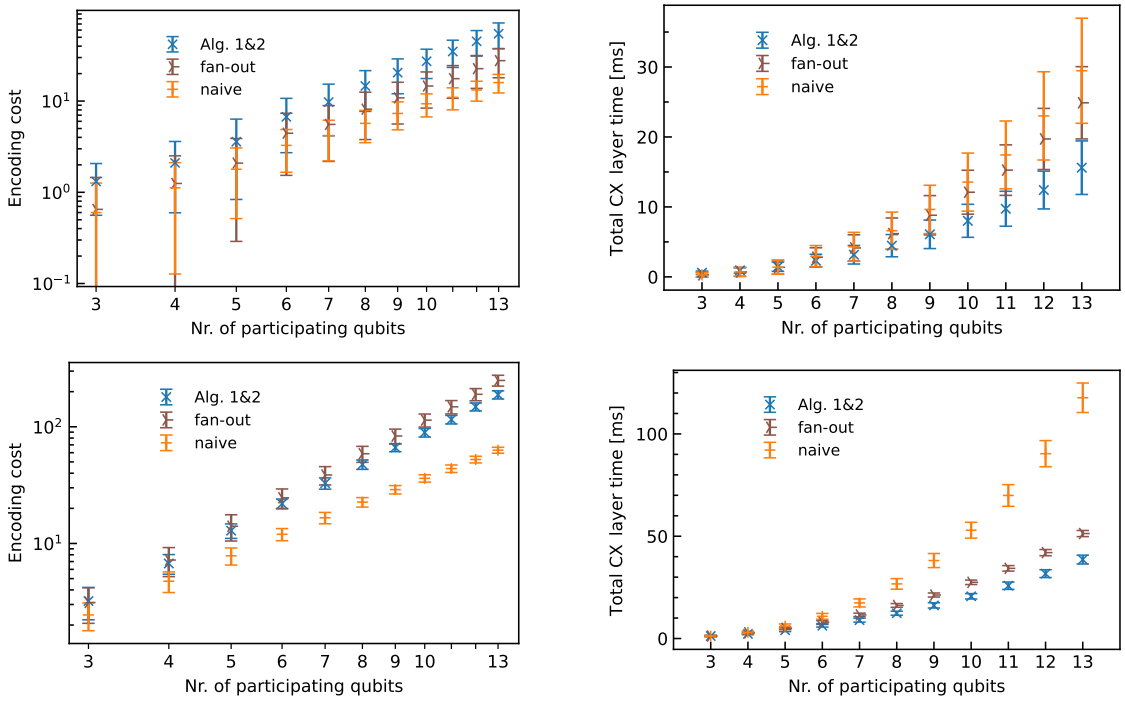


Figure 5: Comparing the performances of Algorithms 1 and 2, fan-out and the naive approach for the implementation of a random directed CX layer. The error bars show the variance over 100 samples. The numerical values are obtained using the hardware specific parameters from Section 2.3. **Top and bottom:** Sparse and dense T_{CZ} with the probability 0.2 and 0.8 for picking a "1", respectively. **Left:** The encoding cost of a random directed CX layer on n qubits. In the top left the large variance in the encoding cost is due to the sparsity of T_{CZ}. In contrast, for dense T_{CZ} (bottom left) Algorithms 1 and 2 lead to a reduced encoding cost compared to the fan-out approach. **Right:** Neglecting local gates, for both sparse and dense T_{CZ} the Algorithms 1 and 2 yields the lowest total CX layer time.

the compilation of this layer.

Consider a directed CX layer with randomly chosen entries of the lower triangular part of T_{CZ}, and T_{H} as in Eq. (3.12). We distinguish between the naive implementation, the implementation of the fan-out gates directly as a GZZ gate (and local gates) and the application of Algorithms 1 and 2 to T_{CZ} and T_{H}. Like in Section 2.3, the naive implementation corresponds to the sequential execution of two-qubit ZZ gates. Therefore, the encoding cost and the total gate time for the naive approach is $\sum_{i<j} T_{\mathsf{CZ}}[i,j]$ and $\sum_{i<j} T_{\mathsf{CZ}}[i,j]/J_{ij}$, respectively.

Implementing the fan-out gate directly associates one GZZ gate with each column of T_{CZ}. We further take advantage of the sparsity of T_{CZ}. If we consider the fan-out gate represented by the i-th column of T_{CZ}, then for all j with $T_{\mathsf{CZ}}[i,j] = 0$ we can exclude all the j-th qubits from the participation in that GZZ gate, reducing the encoding cost. Thus, for the fan-out gate implementation we have $n-1$ symmetric matrices A with ones in the first row/column and zeros everywhere else. The total encoding cost for the fan-out approach is therefore the sum of the encoding costs for the $n-1$ GZZ(A) gates. The same holds for the total CX layer time as the sum of the total GZZ times.

Algorithms 1 and 2 take a different advantage of the sparsity of T_{CZ} by commuting Hadamard gates and combining parts of multiple fan-out gates to one GZZ gate. As stated above, we need at most $\lfloor \frac{n-1}{2} \rfloor$ GZZ gates. If k fan-out gates are combined, the resulting GZZ gate is characterized by a symmetric matrix A where the first k rows (and also columns) can have non-zero values. The total encoding cost for the Algorithms 1 and 2 is the sum of the encoding costs of the $\lfloor \frac{n-1}{2} \rfloor$ GZZ gates plus the encoding cost of the $\lceil \frac{n-1}{2} \rceil$ ZZ gates. The same holds for the total CX layer time as the sum of the total GZZ times and the times for the $\lceil \frac{n-1}{2} \rceil$ ZZ gates. Note that we neglect local gates in our considerations.

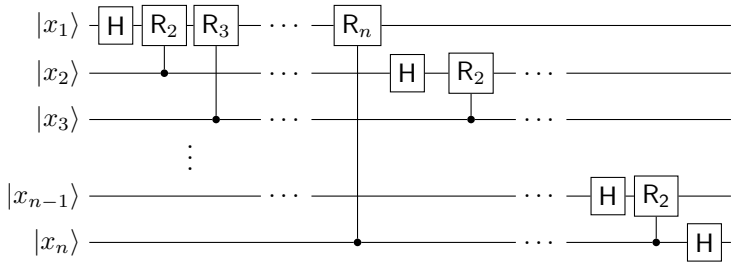


Figure 6: The quantum Fourier transform (with reversed order of the output qubits, i.e. without the swapping gates at the end).

Figure 5 shows that for a dense directed CX layer Algorithms 1 and 2 have a significant advantage in the total CX layer time over the naive and the fan-out implementation. Also, the encoding cost is reduced compared to the fan-out implementation. For sparse T_{CZ} the advantage is still visible in the total CX layer time, but the difference between the approaches is less substantial.

3.3 Quantum Fourier transform

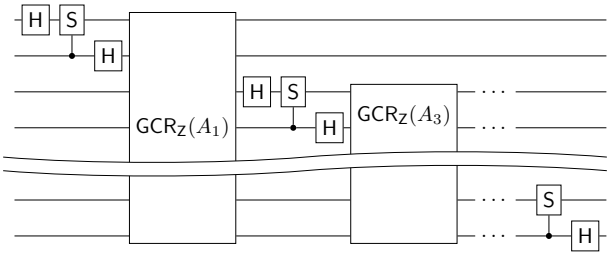
The quantum Fourier transform is an essential ingredient in many quantum algorithms. To define the corresponding unitary operator on an n-qubit register, we identify the elements of the computational basis $|x_1, \ldots, x_n\rangle$ for $x_j \in \mathbb{F}_2$ with integers in binary representation, i.e. $x = \sum_{j=1}^{n} x_j 2^{n-j}$. The QFT is then given as

$$\mathrm{QFT}|x\rangle = \frac{1}{2^{n/2}} \sum_{y=0}^{2^n-1} e^{2\pi \mathrm{i} x y 2^{-n}} |y\rangle = \frac{1}{2^{n/2}} \bigotimes_{j=1}^{n} \left(|0\rangle + \mathrm{e}^{2\pi \mathrm{i} x 2^{-j}} |1\rangle\right). \tag{3.15}$$

The latter form immediately leads to the efficient quantum circuit in Figure 6 which uses $n(n-1)/2$ controlled $\mathsf{R_Z}$-rotations with angles $2\pi/2^j$ for $j = 2, \ldots, n$ and n Hadamard gates. For convenience, we introduce the shorthand notation $\mathsf{R}_j := \mathsf{R_Z}(2\pi/2^j)$, in particular $\mathsf{Z} := \mathsf{R}_1$, $\mathsf{S} := \mathsf{R}_2 = \sqrt{\mathsf{Z}}$, and $\mathsf{T} := \mathsf{R}_3 = \sqrt[4]{\mathsf{Z}}$. In the QFT circuit in Figure 6 we can collect for each $j < n$ the subsequent controlled $\mathsf{R}_{n-j}, \ldots, \mathsf{R}_1$ to a (multi-qubit) $\mathsf{GCR_Z}(A)$ gate specified by the $(n-j+1) \times (n-j+1)$ symmetric matrix

$$A = 2\pi \begin{pmatrix} 0 & 2^{-1} & \ldots & 2^{-n+j} \\ 2^{-1} & 0 & \ldots & 0 \\ \vdots & \vdots & \ddots & \\ 2^{-n+j} & 0 & & 0 \end{pmatrix}. \tag{3.16}$$

Since our compilation scheme for the fully directed CX layer in Section 3.2.3 is agnostic of the rotation angles, up to local $\mathsf{R_Z}$ rotations, we can apply it directly to the QFT circuit. This yields the compiled QFT circuit:



Note that we are able to achieve this form without any Hadamard gate obstructing the required movement of $\mathsf{CR_Z}$ gates. The combination of diagonal multi-qubit gates across non-diagonal local gates has been carried out for small systems e.g. in Refs. [19, 28]. However, this approach requires numerics in unitary groups beyond SU(2), which we avoid to ensure scalability (w.r.t. the Hilbert

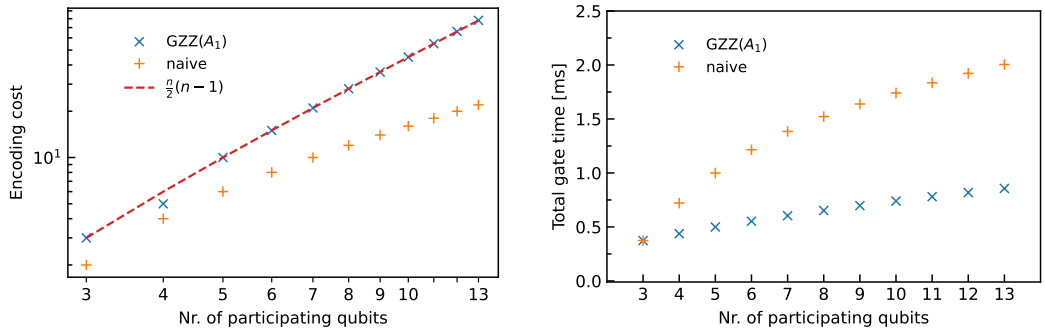


Figure 7: **Left:** The encoding cost, i.e. the different encodings, we need to implement $\mathsf{GZZ}(A_1)$ on n qubits. For the naive approach the encoding cost only increases with $2(n-2)$, since that is the number of non-zero entries in A_1. **Right:** Due to the exponentially small non-zero entries in A_1 even the naive approach seems to have an asymptotic linear scaling of the total gate time. As before, the numerical values are obtained using the hardware specific parameters from Section 2.3.

space dimension). We also do not address the compilation of local $\mathsf{R_Z}$ rotations here, see e.g. Ref. [52] which covers that topic.

After our compilation scheme we have $\lceil \frac{n-1}{2} \rceil$ CS gates (controlled-$\sqrt{\mathsf{Z}}$ gates) and $\lfloor \frac{n-1}{2} \rfloor$ $\mathsf{GCR_Z}$ gates characterized by the symmetric matrix

$$A_j \coloneqq 2\pi \begin{pmatrix} 0 & 0 & 2^{-2} & \cdots & 2^{-n+j} \\ 0 & 0 & 2^{-1} & \cdots & 2^{-n+j+1} \\ 2^{-2} & 2^{-1} & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \\ 2^{-n+j} & 2^{-n+j+1} & 0 & & 0 \end{pmatrix}. \tag{3.17}$$

A $\mathsf{GCR_Z}(A_j)$ gate can be implemented as described in Section 3.1 using a $\mathsf{GZZ}(A_j)$ and additional single-qubit $\mathsf{R_Z}$ gates on the qubits they act on. For the QFT, we can push the resulting $\mathsf{R_Z}$ gates on the target qubits to the end of circuit, and likewise we can push the ones on the control qubits to the beginning.

In conclusion, we can apply the fully directed CX layer compiling scheme to implement a QFT circuit with the same amount of GZZ gates and the same encoding cost. The most difficult part in the QFT are the local $\mathsf{R_Z}$ rotations with exponentially small angles. Due to the small values in the matrices A_j one can also expect a practical issue in the implementation with too small $\lambda_{\boldsymbol{m}}$, see Section 2.2.

Numerical results for the QFT

We provide the numerical results for the performance of the $\mathsf{GZZ}(A_j)$ gate with A_j from Eq. (3.17) in Figure 7. It is a priori unclear how the exponentially small entries in A_j effect the time spent in each encoding $\lambda_{\boldsymbol{m}}$. We set $j = 1$ since $\mathsf{GZZ}(A_1)$ acts on n qubits and is therefore the most costly GZZ gate in the QFT. As in Section 2.3 we compare the result of the GZZ gate implementation via the LP (2.5) against the naive approach, i.e. the sequential implementation of two-qubit controlled R_j gates. As before, we neglect the finite recoding time, i.e. the time for executing the X gate layers, and only consider the time needed to execute a $\mathsf{GZZ}(A_1)$ gate or the naive approach. Note that in Figure 7 we consider only a single $\mathsf{GZZ}(A_1)$ gate, while the total QFT circuit consists of $\lfloor \frac{n-1}{2} \rfloor$ $\mathsf{GZZ}(A_j)$ gates.

Due to the exponentially small entries in A_1 we might in practice run into the problem of too small $\lambda_{\boldsymbol{m}}$ for a QFT on a moderate amount of qubits.

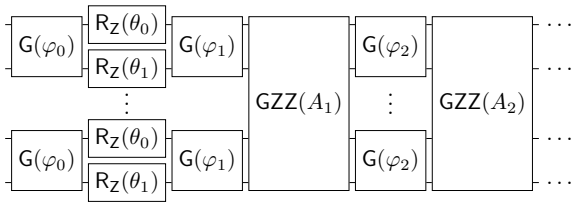
3.4 Circuit reduction for quantum chemistry applications

Simulating molecular dynamics is probably one of the main applications for quantum computations. Molecular dynamics are governed by the Coulomb Hamiltonian, consisting of the kinetic energy terms and of the Coulomb interactions between the electrons and the nuclei. It is common to neglect the kinetic terms of the nuclei which is called "Born-Oppenheimer approximation" [53].

The determination of approximate ground and excited eigenstates of the remaining electronic Hamiltonian is a hard task even on a quantum device [54]. Since simulating the time-dependent Schrödinger equation is more natural on a quantum computer there has been much effort to approximate eigenstates by solving the time-dependent Schrödinger equations [55–57]. The corresponding time evolution can be approximated by factorizing the time dependent interactions into $m-1$ layers [22]:

$$U_t \approx U_{Ext}\, \hat{\mathsf{G}}(\varphi_m) \left[\prod_{k=1}^{m-1} \mathsf{GZZ}(A_k)\, \hat{\mathsf{G}}(\varphi_k) \right] \hat{\mathsf{R}}_\mathsf{Z}(\theta_{0,1})\, \hat{\mathsf{G}}(\varphi_0)\,, \tag{3.18}$$

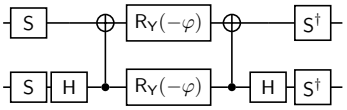
where the hat denotes layers of the gates and $\mathsf{G}(\varphi)$ is a Givens rotation on two qubits. U_{Ext} represents the constant nucleus-nucleus Coulomb interaction and corresponds to a global phase which we henceforth omit. $A_k \in \mathrm{Sym}_0(\mathbb{R}^n)$ denotes the total coupling matrix, characterizing the GZZ gate. The right-hand side can then be represented as the circuit



. (3.19)

The implementation of the GZZ gates is straightforward, see Section 2, and we again do not address the compilation of local $\mathsf{R}_\mathsf{Z}(\theta)$ rotations in this work. Therefore, we focus on the decomposition of the Givens rotation

$$\mathsf{G}(\varphi) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\varphi) & \sin(\varphi) & 0 \\ 0 & -\sin(\varphi) & \cos(\varphi) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \equiv$$



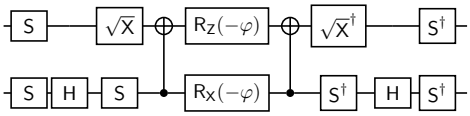
(3.20)

where $\mathsf{R}_\mathsf{Y}(\varphi) = e^{-iY\varphi/2}$ is a rotation around the y-axis. Using the Euler decomposition, up to global phases we can express $\mathsf{R}_\mathsf{Y}(\varphi)$ as

$$\mathsf{R}_\mathsf{Y}(\varphi) \equiv \sqrt{\mathsf{X}}^\dagger\, \mathsf{R}_\mathsf{Z}(\varphi) \sqrt{\mathsf{X}} \equiv \mathsf{S}^\dagger\, \mathsf{R}_\mathsf{X}(\varphi)\, \mathsf{S}\,. \tag{3.21}$$

Inserting this into Eq. (3.20) we obtain

$\mathsf{G}(\varphi) \equiv$



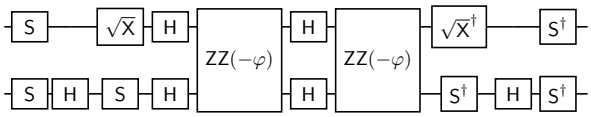
. (3.22)

Using the commutation rule

$$\mathsf{CX}_{2,1}\, \mathsf{R}_\mathsf{Z}(\varphi)_1 \equiv \mathsf{ZZ}_{1,2}(\varphi)\, \mathsf{CX}_{2,1} \tag{3.23}$$

twice, first we commute the R_Z gate with the right CX, then conjugate the R_X gate and the left CX with Hadamard gates and commute them. This gives

$\mathsf{G}(\varphi) \equiv$



. (3.24)

Since any layer of ZZ gates can be implemented with only one GZZ gate, see Eq. (3.4), we can collect the parallel $\mathsf{ZZ}(-\varphi)$ gates of the Givens rotation layer $\hat{\mathsf{G}}(\varphi)$ into a single $\mathsf{GZZ}(-\varphi A_{NN})$ gate. Overall, this results in two GZZ gates per Givens rotation layer. Here, $A_{NN} \in \mathrm{Sym}_0(\mathbb{F}_2^n)$ denotes the total coupling matrix that pairwise couples next-neighbor qubits, i.e. qubits 1 and 2, qubits 3 and 4, etc. Concretely, A_{NN} takes the form

$$A_{NN} = \begin{pmatrix} 0 & 1 & & & \\ 1 & 0 & & & \\ & & \ddots & & \\ & & & 0 & 1 \\ & & & 1 & 0 \end{pmatrix}. \tag{3.25}$$

Usually, the subdiagonal of the hardware-given coupling matrix J contains the largest values and thus corresponds to the fastest ZZ gates. This results in very fast $\mathsf{GZZ}(-\varphi A_{NN})$ gates.

A straightforward implementation of the Givens rotation layer with Eq. (3.20) results in n local $\mathsf{R_Y}$ rotations with arbitrary angle. On a quantum computer with a finite native gate set the rotations with arbitrary angle might be challenging to implement. Although there are methods to decompose rotations with arbitrary angle into a native gate set, see e.g. Refs. [58–60], they always come with a considerable encoding cost.

To sum up, to approximate molecular dynamics as described by the circuit (3.19) one needs $2(m+1)$ $\mathsf{GZZ}(-\varphi_k A_{NN})$ gates, $(m-1)$ $\mathsf{GZZ}(A_k)$ gates, n local $\mathsf{R_Z}(\theta_{0,1})$ gates and some local Clifford gate.

3.5 Decomposing general diagonal unitaries

Since the GZZ gate is diagonal, it is natural to look for a compilation procedure to implement general diagonal unitaries using the GZZ gate together with local $\mathsf{R_Z}$ and Hadamard gates. We assume that we are able to implement $\mathsf{R_Z}$ gates with arbitrarily small angles which in practice might be difficult. Yet, there are algorithms to approximate any $\mathsf{R_Z}$-angle using only Clifford+T gates [59] or Clifford+$\sqrt{T}$ [52]. This compilation scheme leads us beyond Clifford circuits and should be understood as a stepping stone towards more general compilation schemes.

First we introduce the representation of diagonal unitaries by phase polynomials [61, 62]. Then we show how certain terms of the phase polynomial can be implemented in parallel. We optimize the order of these parallelized layers such that the encoding cost and the total GZZ time is reduced.

Any diagonal n-qubit unitary acts as

$$U_f|\boldsymbol{x}\rangle \coloneqq \mathrm{e}^{2\pi \mathrm{i} f(\boldsymbol{x})}|\boldsymbol{x}\rangle, \tag{3.26}$$

with $\boldsymbol{x} \in \mathbb{F}_2^n$ for some pseudo-Boolean function $f : \mathbb{F}_2^n \to \mathbb{R}$. Such a function can be uniquely expressed as a multilinear *phase polynomial* [63, Theorem 1.1],

$$f(\boldsymbol{x}) = \sum_{\boldsymbol{y}\in\mathbb{F}_2^n} \alpha_{\boldsymbol{y}} \chi_{\boldsymbol{y}}(\boldsymbol{x}) \tag{3.27}$$

with coefficients or phases $\alpha_{\boldsymbol{y}} \in \mathbb{R}$ and *parity function* $x \mapsto \chi_{\boldsymbol{y}}(x) = y_1 x_1 \oplus \cdots \oplus y_n x_n$ for each *parity* $\boldsymbol{y} \in \mathbb{F}_2^n$. Additionally taking into account basis state transformations $|\boldsymbol{x}\rangle \mapsto |g(\boldsymbol{x})\rangle$ the above extends to the so-called *sum-over-paths representation*, which was used in Refs. [61, 62].

Terms in the expansion of f of the form αx_i and $\alpha(x_i \oplus x_j)$ can directly be implemented as $\mathsf{R_Z}(\alpha)_i$ and $\mathsf{ZZ}(\alpha)_{i,j}$ gate, respectively, and are therefore easy. We thus only consider terms with more than two variables x_i.

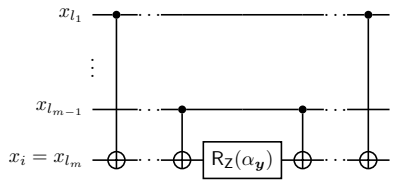
We define the unitary representing one term in $f(\boldsymbol{x})$ by

$$U_{i,\boldsymbol{y}}|\boldsymbol{x}\rangle \coloneqq \mathrm{e}^{2\pi \mathrm{i} \alpha_{\boldsymbol{y}} \chi_{\boldsymbol{y}}(\boldsymbol{x})}|\boldsymbol{x}\rangle\,, \tag{3.28}$$

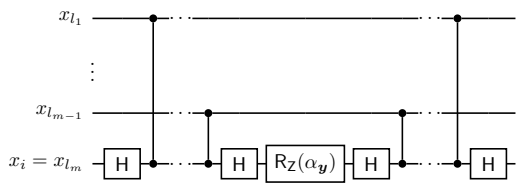
where i is a qubit index. The role of the subscript i gets apparent in the circuit representation of

the unitary $U_{i,\boldsymbol{y}}$

$$U_{i,\boldsymbol{y}} \equiv$$



$$=$$



(3.29)

where $l = (k \in [n] : y_k \neq 0) \in [n]^m$ is a sequence containing the m non-zero components of $\boldsymbol{y}$ and with the conjugating Hadamard gates, the $\mathsf{R_Z}$ gate and all the CX gate targets on the i-th qubit. Note that we omit $\alpha_{\boldsymbol{y}}$ in $U_{i,\boldsymbol{y}}$ since the following discussion is agnostic of the phase $\alpha_{\boldsymbol{y}}$. The domain of $U_{i,\boldsymbol{y}}$ is given by the support of the parity $\mathrm{supp}(\boldsymbol{y}) \coloneqq \{k \mid y_k \neq 0\}$. Since the unitaries of the two CX layers conjugating the $\mathsf{R_Z}$ gate in $U_{i,\boldsymbol{y}}$ are permutation matrices one has the freedom in choosing any $i \in \mathrm{supp}(\boldsymbol{y})$. That is, for any $i, j \in \mathrm{supp}(\boldsymbol{y})$, $U_{i,\boldsymbol{y}}$ and $U_{j,\boldsymbol{y}}$ implement the same unitary. Applying Eq. (3.5) to the right-hand side of Eq. (3.29), we find that each $U_{i,\boldsymbol{y}}$ can be implemented with two GZZ gates, one $\mathsf{R_Z}(\alpha_{\boldsymbol{y}})$ gate, four Hadamard gates and some $\mathsf{R_Z}(\pi/2)$ gates.

Clearly, two such diagonal unitaries $U_{i,\boldsymbol{y}}$ and $\mathrm{U}_{i,\boldsymbol{y}'}$ can be implemented in parallel if their supports are disjoint, i.e. if $\mathrm{supp}(\boldsymbol{y}) \cap \mathrm{supp}(\boldsymbol{y}') = \emptyset$. Since our GZZ gates are time-optimal, this results in a time-optimal implementation of those unitaries, assuming as above a time-optimal implementation of the single-qubit $\mathsf{R_Z}$ gates. Denote by $\mathcal{S} \coloneqq \{\mathrm{supp}(\boldsymbol{y}) \mid \boldsymbol{y} \in \mathbb{F}_2^n, |\boldsymbol{y}| > 2\}$ the set of all supports of the parities without the easy terms. Algorithm 3 is a heuristic algorithm, which parallelizes $U_{i,\boldsymbol{y}}$ with disjoint supports such that the resulting support is as large as possible.

Algorithm 3 Parallelizing supports.

Input: $\mathcal{S}$ ▷ Set of supports

```
L ← ∅
while S ≠ ∅ do
    Choose s_k ∈ S
    S ← S \ {s_k}                                   ▷ Remove s_k from S
    s ← {s_k}
    while There exists s_i ∈ S s.t. s ∩ s_i = ∅ and max_{s_i∈S} |s ∪ s_i| do
        s ← s ∪ {s_i}                               ▷ Append disjoint support with maximal union size
        S ← S \ {s_i}
    L ← L ∪ s                                       ▷ Append to set of parallelized layers
```

Output: $\mathcal{L}$ ▷ Set of parallelized layers

Between any two consecutive unitaries $U_{i,\boldsymbol{y}}$ and $\mathrm{U}_{i',\boldsymbol{y}'}$ with support overlap $o \coloneqq \mathrm{supp}(\boldsymbol{y}) \cap \mathrm{supp}(\boldsymbol{y}') = s \cap s' \neq \emptyset$ two types of cancellation can happen. First, the Hadamard gates cancel, provided that they act on the same qubit. This is achieved by choosing the qubit i on which the Hadamard gates act, concretely by demanding $i = i' \in o$. Thereafter, one can combine the two CZ layers of $U_{i,\boldsymbol{y}}$ and $\mathrm{U}_{i',\boldsymbol{y}'}$ and implement them with just one GZZ gate, see Eq. (3.5). Doing so, a second cancellation happens automatically: One can see from the right-hand side of Eq. (3.29) that between $U_{i,\boldsymbol{y}}$ and $\mathrm{U}_{i,\boldsymbol{y}'}$, $2|o|$ CZ gates cancel and therefore $|o|$ less qubits participate in the GZZ gate. This leads to a quadratic reduction $\mathrm{O}(|o|^2)$ of the encoding cost, which for a n qubit GZZ gate is $n(n-1)/2$.

Let $\mathcal{L}_r \coloneqq \{u \in \mathcal{L} \mid |u| = r\}$, where $\mathcal{L}$ is the set of all parallelized layers returned by Algorithm 3, be the subset of all parallelized layers that contain r disjoint supports of unitaries $U_{i,\boldsymbol{y}}$. The set $\mathcal{L}_r$ is important for the placement of the Hadamard gates. For example, if two parallelized layers $u, u' \in \mathcal{L}_r$ are executed consecutively and there exist i and j for all $s_1, \ldots, s_r \in u$ and $s'_1, \ldots, s'_r \in u'$

such that $s_i \cap s'_j \neq \emptyset$, then all interleaving Hadamard gates cancel if they are placed on the qubits contained in $s_i \cap s'_j$.

We now extend this argument to all parallelized layers $u^{(k)} \in \mathcal{L}_r$. If there is a qubit contained in the repeated overlap of the r supports of all consecutive parallelized layers, all Hadamard gates between the parallelized layers cancel. Concretely, for the cancellation of all Hadamard gates there must exist indices $i_k \in \{1, \dots, r\}$ for all $s_1^{(k)}, \dots, s_r^{(k)} \in u^{(k)}$ such that $\bigcap_k^{|\mathcal{L}_r|} s_{i_k}^{(k)} \neq \emptyset$. We further note that if we apply $u \in \mathcal{L}_r$ and $v \in \mathcal{L}_{r'}$ successively for $r \neq r'$, then it is impossible that all interleaving Hadamard gates cancel.

The next lemma gives the encoding cost for the set of parallelized layers $\mathcal{L}_r$ and deals with the placement of the Hadamard gates by introducing few ancilla qubits.

Lemma 3.1. *The set of parallelized layers $\mathcal{L}_r$ can be implemented using $2r$ Hadamard gates, $|\mathcal{L}_r|r$ non-Clifford RZ rotations and $|\mathcal{L}_r| + 1$ GZZ gates by introducing at most r ancilla qubits.*

Proof. Each $u^{(k)} \in \mathcal{L}_r$ contains $2r$ Hadamard gates. The position of the $2r$ Hadamard gates can be chosen freely on the r supports $s_1^{(k)}, \dots, s_r^{(k)} \in u^{(k)}$. So we want to find qubits q_i in the overlap of all supports, i.e. $q_i \in \bigcap_k^{|\mathcal{L}_r|} s_{i_k}^{(k)}$ with for each $i = 1, \dots, r$. If this is not possible for qubit q_i, i.e. $\bigcap_k^{|\mathcal{L}_r|} s_{i_k}^{(k)} = \emptyset$, we add an ancilla qubit to all supports containing q_i and set q_i to that qubit. One sees that in the worst case, setting all $q_1, \dots, q_r$ to the ancilla qubits and adding them to the supports $s_1, \dots, s_r$ for all $u \in \mathcal{L}_r$ results in the cancellation of all interleaving Hadamard gates, leaving only $2r$ Hadamard gates. The GZZ gate count results from combining the two GZZ gates of subsequent u's as discussed above. □

Note that the r ancilla qubits can be reused by different $\mathcal{L}_r$ since we do not encode any information on them. For an n-qubit diagonal unitary one needs $r \leq \lfloor \frac{n}{3} \rfloor$ ancilla qubits to ensure that all Hadamard gates cancel. The upper bound comes from the fact that we consider only supports with $|\boldsymbol{y}| > 2$ so at most $\lfloor \frac{n}{3} \rfloor$ many supports can be in parallel.

Assume now, that we have r ancilla qubits or that all interleaving Hadamard gates cancel. Then one can freely permute the order of the parallelized layers. Such reordering does not change the GZZ gate count, but it possibly reduces the amount of qubits participating in the GZZ gates and therefore the encoding cost. We thus want to find an order such that the cancellation of CZ gates is maximized, i.e. we want to maximize the overlap of the supports of consecutive parallelized layers. We define the *shared support size* between two parallelized layers $u, u' \in \mathcal{L}_r$ with r elements as $S_{u,u'} \coloneqq \sum_{i=0}^{r} |s_i \cap s'_i|$, where $s_i \in u$ and $s'_i \in u'$. Choose the ordering of elements in $\mathcal{L}_r$ that maximizes the shared support size $S_{u,u'}$ over all pairs of parallelized layers $u, u' \in \mathcal{L}_r$. This relates to the traveling salesman problem in the formulation of Ref. [64] with each parallelized layer as a vertex and weights $S_{u,u'}$ on the edge between u and u'. Hence the reordering of the parallelized layers cannot be solved efficiently. However, there are heuristic algorithms, e.g. the Christofides algorithm [32] which only takes $\mathrm{O}(|\mathcal{L}_r|^3)$ steps and guarantees that the solution is within a factor $3/2$ of the optimal solution.

In the worst case of the proposed compiling method for general diagonal unitaries, we consider the set of all supports $\mathcal{S}$ without easy gates, i.e., without single- and two-qubit gates. This set has $|\mathcal{S}| = 2^n - (n + n(n-1)/2) = 2^n - n(n+1)/2$ elements, and contains for every element $s \in \mathcal{S}$ also the complement $\bar{s}$ with $|s \cup \bar{s}| = n$. Therefore, we can always parallelize two gates, the one corresponding to s and that corresponding to $\bar{s}$. We thus only have $\mathcal{L}_2$ which has size $|\mathcal{L}_2| = |\mathcal{S}|/2 = 2^{n-1} - n(n+1)/4$. Using Lemma 3.1, this requires only two ancilla qubits and $|\mathcal{S}|/2$ GZZ gates.

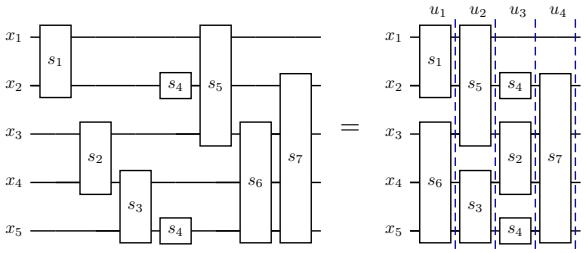
In this section we proposed a method to decompose arbitrary diagonal unitaries into local gates and GZZ gates. We showed that this leads to an optimization problem which in general is NP-hard. Using heuristic algorithms might still achieve a significant reduction in the encoding cost.

Example. We illustrate the compilation scheme explained above on a diagonal unitary on five qubits. Let $\boldsymbol{x} \in \mathbb{F}_2^5$ and define a diagonal unitary as in Eq. (3.26) with

$$\begin{aligned} f(\boldsymbol{x}) = \frac{1}{2}\big(&x_1 \oplus x_2 + x_3 \oplus x_4 + x_4 \oplus x_5 \\ &+ x_1 \oplus x_2 \oplus x_3 + x_2 \oplus x_4 \oplus x_5 \\ &+ x_2 \oplus x_3 \oplus x_4 \oplus x_5\big)\,. \end{aligned} \tag{3.30}$$

The factor 1/2 is chosen for convenience and yields a $R_Z(\pi) = \mathsf{Z}$ gate in the circuit representation of Eq. (3.29) for each term in f. Conjugating these Z gates with Hadamard gates as in Eq. (3.29) we get one $\mathsf{HZH} = \mathsf{X}$ gate for each term.

Note that in the above we considered only $s_i = \mathrm{supp}(\boldsymbol{y}_i) > 2$, since all terms with $s_i = 2$ can be implemented with a single GZZ gate. To keep this example short we here also allow $s_i = 2$ since this gives more non-trivial possibilities to parallelize layers. We illustrate the diagonal unitary with the supports s_i in the circuit diagram below. Applying Algorithm 3 yields

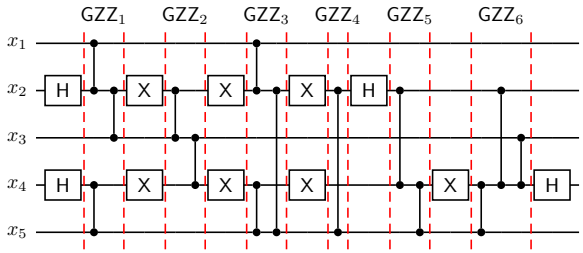


(3.31)

We now group the parallelized layers u_i together in the sets $\mathcal{L}_2 = \{u_1, u_2, u_3\}$ and $\mathcal{L}_1 = \{u_4\}$. In this example we do not make use of ancilla qubits. One can see that the placement of the Hadamard gates should be on $s_1 \cap s_5 \cap s_4 = \{x_2\}$ for the supports s_1, s_5, s_4 and on $s_6 \cap s_3 \cap s_2 = \{x_4\}$ for the supports s_6, s_3, s_2 such that all the interior Hadamard gates cancel. For s_7 the Hadamard gate can be placed on x_2 or x_4. Since $|s_4 \cap s_7| = |s_2 \cap s_7|$ both choices are equally good for the cancellation of CZ gates. We chose the Hadamard position for s_7 to be x_4. The shared support size S_{u_i,u_j} between two parallelized layers for the set $\mathcal{L}_2$ can be calculated and expressed as the matrix

$$S = \begin{pmatrix} * & 4 & 3 \\ 4 & * & 2 \\ 3 & 2 & * \end{pmatrix} \tag{3.32}$$

With $S_{u_2,u_1} = 4$ and $S_{u_1,u_3} = 3$ we get the optimal order for $\mathcal{L}_2$ as u_2, u_1, u_3, and $2(S_{u_2,u_1} + S_{u_1,u_3}) = 14$ CZ gates cancel. Using Eq. (3.29) and $\mathsf{HZH} = \mathsf{X}$, the resulting circuit is



(3.33)

One needs six GZZ gates to implement the diagonal unitary. The encoding cost for each GZZ_i in Eq. (3.33) with support s_i is $|s_i|(|s_i| - 1)/2$, such that the total implement cost is $10 + 3 + 6 + 1 + 3 + 6 = 29$. For the circuit on the left-hand side of Eq. (3.31) we require two GZZ gates for each support s_i. We have 7 supports s_i, i.e. 14 GZZ gates which leads to the total encoding cost is $2(2 + 2 + 2 + 2 + 3 + 3 + 6) = 40$. Thus, in this example our compilation scheme reduces the encoding cost by $\approx 25\%$.

4 Conclusion

In this work, we showed how to synthesize a time-optimal multi-qubit gate, the GZZ gate, on an abstract quantum computing platform. The only requirements on the platforms are that (I) single-qubit rotations can be executed in parallel, (II) it offers global Ising-type interactions with all-to-all connectivity, and (III) that there is a way to exclude certain qubits from the participation in the multi-qubit gate. We showed that and how to realize these requirements in a microwave controlled ion trap using magnetic gradient-induced coupling (MAGIC).

Arbitrary couplings between the qubits can be generated via X gate layers interleaved by the Ising-type evolution. The required X gate layers as well as the durations of the evolution times are determined by solving an LP. In numerical experiments we showed that the gate time of the resulting GZZ gates scales approximately linear with the number of participating qubits under reasonable assumptions on an implementing physical platform. Based on these time-optimal GZZ gates, we presented an improved compiling strategy for Clifford circuits, and applied this strategy to compile the QFT. Moreover, we applied the GZZ gates to the simulation of molecular dynamics, and presented a compiling strategy for general diagonal unitaries. This can be thought of as a step towards compilation strategies for arbitrary unitaries.

In the future, it will be interesting to investigate how the GZZ gates perform on a real-world ion trap, and how robust they can be made against errors similar to the error mitigation scheme for the DAQC setting [65]. Moreover, we hope that our time-optimal gate synthesis method will be applied to small ion trap registers which are embedded in a quantum processing unit (QPU) module.

Acknowledgements

We are grateful to the MIQRO collaboration, in particular to Robert Jördens for helpful discussions and valuable feedback on our manuscript and Christof Wunderlich for discussions on the platform and the QFT. Moreover, we would like to thank Christian Gogolin for making us aware of the connection to molecular dynamics and Lennart Bittel for fruitful discussions on convex optimization.

This work has been funded by the German Federal Ministry of Education and Research (BMBF) within the funding program "quantum technologies – from basic research to market" via the joint project MIQRO (grant numbers 13N15522 and 13N15521).

Appendices

A Ion trap quantum computing with microwave control

In this section we explain in more detail how qubits are encoded into cold ions stored in a trap, how MAGIC can be used to tailor interactions between these qubits, and how microwaves perform single-qubit gates. A full in-depth treatment can be found in Refs. [15, 36].

In short, pairs of hyperfine states of ions are interpreted as the computational basis states of qubits. These ions are cooled down to form a "Coulomb crystal", a stable configuration confined by the trap potential. An inhomogeneous magnetic field superimposed with the trap makes the crystal equilibrium depend on the internal state (the qubits). Changes in the internal state hence lead to excitations (phonons), which can be interpreted as interactions between the qubits. The position-dependent Zeeman effect makes individual qubit transitions addressable. Microwaves drive Rabi oscillations on these transitions to perform single-qubit gates.

A.1 Ytterbium ions and qubits

We consider ions with nuclear spin $I = \frac{1}{2}$ and total electron angular momentum $J = \frac{1}{2}$, for example single-ionized Ytterbium-171 ($^{171}\mathrm{Yb}^{+}$). These values imply that the "ground state" (the lowest main quantum number) is spanned by four "hyperfine" sublevels. Since nuclear spin and electron angular momentum couple to each other through an interaction term $\propto \boldsymbol{I} \cdot \boldsymbol{J}$, the full Hamiltonian is not diagonal in the naive product basis $|m_I, m_J\rangle$. One instead introduces the total angular momentum $\boldsymbol{F} \coloneqq \boldsymbol{I} + \boldsymbol{J}$. The corresponding quantum numbers F and $m_F = -F, \ldots, F$ can be used to label the hyperfine eigenstates, which group into a singlet $|F = 0, m_F = 0\rangle$ and a triplet $|F = 1, m_F = -1, 0, +1\rangle$. The singlet forms the overall ground state and is separated by an energy gap from the triplet states, which all have the same energy. The energy difference is called the "hyperfine structure constant" of the ion. This quantity is usually determined experimentally and typically lies in the microwave regime. For $^{171}\mathrm{Yb}^{+}$, it corresponds to a frequency of $f_0 = 12.642812118466(2)\mathrm{GHz}$ [66].

Placing the ion in an external magnetic field $\boldsymbol{B}$ yields new dipole terms $\propto \boldsymbol{B}\cdot\boldsymbol{I}$ and $\propto \boldsymbol{B}\cdot\boldsymbol{J}$ in the Hamiltonian. Since these new terms do not preserve F, the $|F, m_F\rangle$ basis is no longer an eigenbasis. However, they still preserve m_F, so the new energy eigenstates are superpositions of states with equal m_F. The "outer" triplet states $|F=1, m_F=\pm1\rangle$ remain basis states due to their unique m_F, but $|F=0, m_F=0\rangle$ and $|F=1, m_F=0\rangle$ form two orthogonal linear combinations with amplitudes depending on the field strength $B \coloneqq |\boldsymbol{B}|$. Also the eigenvalues of the Hamiltonian become B-dependent, and the previous triplet degeneracy is lifted. For the present case of four states, the problem can be solved analytically leading to the Breit-Rabi formula [29]. We are mainly interested in the weak field limit $B \to 0$, commonly known as the Zeeman effect. The energy shift of the states $|F=1, m_F=\pm1\rangle$ is exactly proportional to B ("linear Zeeman effect"), while the other two eigenvalues are field-independent to first order ("magnetic insensitive"). However, they do have a higher-order dependence starting from $\propto B^2$, which is known as the "quadratic Zeeman effect", see also Figure 1b. Not only are these eigenvalues close to the zero-field values, but also the linear combinations have a dominant component of either $|F=0, m_F=0\rangle$ or $|F=1, m_F=0\rangle$. So although this is technically wrong, one casually keeps the $|F, m_F\rangle$ nomenclature for the basis states even in the presence of a small magnetic field.

We now wish to "encode" a qubit into this four-dimensional state space, i.e. to pick two of the energy eigenstates as computational basis states $|0\rangle$ and $|1\rangle$. A priori, there are six possible choices. The Zeeman splitting between the $F=1$ states is typically on the order of MHz, so orders of magnitude smaller than the hyperfine splitting. Also, the transitions between $|F=1, m_F=0\rangle$ and the other two triplet states are degenerate to first order in B. This is why one usually chooses $|0\rangle \coloneqq |F=0, m_F=0\rangle$ and only considers the three remaining possibilities which we refer to as

$\boldsymbol{\sigma^+}$ **qubit**	with $\|1\rangle \coloneqq \|F=1, m_F=+1\rangle$,
$\boldsymbol{\pi}$ **qubit**	with $\|1\rangle \coloneqq \|F=1, m_F=0\rangle$,
$\boldsymbol{\sigma^-}$ **qubit**	with $\|1\rangle \coloneqq \|F=1, m_F=-1\rangle$.

One may also label the qubits by the m_F of their $|1\rangle$ state. The B-dependent transitions frequencies between $|0\rangle$ and $|1\rangle$ can then be written

$$\omega(m_F, B) = 2\pi f_0 + m_F \frac{\mu B}{\hbar} + \mathrm{O}(B^2), \tag{A.1}$$

where μ has units of a magnetic moment ($\mu \approx \mu_\mathrm{B}$, the Bohr magneton, for $^{171}\mathrm{Yb}^+$). This reflects both the quadratic ($m_F = 0$) and the linear ($m_F = \pm1$) Zeeman effect.

A.2 Trap Hamiltonian and MAGIC

So far, we only considered a single ion. In an ion trap quantum computer, N such ions are confined in an (effective) potential generated by DC and RF electrodes. We assume that the potential geometry and N have been selected such that the ions form a "linear Coulomb crystal", i.e. a one-dimensional chain stabilized by the external trap potential and the mutual Coulomb repulsion of the ions, see Figure 1a. If the crystal has been cooled sufficiently, we can disregard the radial dynamics and focus on the direction of the ion chain which we choose as the z axis. The system is described by a Hamiltonian with three contributions: the kinetic energy $T(\boldsymbol{p}) \propto \boldsymbol{p}^2$ of the ions with momenta $\boldsymbol{p} = (p_1, \ldots, p_N)^T$, and the external and Coulomb potentials

$$V_\mathrm{tot}(\boldsymbol{z}) \coloneqq \sum_{i=1}^{N} V_\mathrm{ext}(z_i) + K \sum_{i<j}^{N} \frac{1}{|z_i - z_j|}, \tag{A.2}$$

with ions positions $\boldsymbol{z} \coloneqq (z_1, \ldots, z_N)^T$ and Coulomb constant K. Again, after sufficient cooling it is reasonable to assume that the system is close to an equilibrium configuration $\bar{\boldsymbol{z}}$. The potential can then be expanded in terms of the elongation $\boldsymbol{q} \coloneqq \boldsymbol{z} - \bar{\boldsymbol{z}}$. The first order term vanishes due to equilibrium, such that

$$V_\mathrm{tot}(\bar{\boldsymbol{z}} + \boldsymbol{q}) = V_\mathrm{tot}(\bar{\boldsymbol{z}}) + \frac{1}{2}\boldsymbol{q}^T \mathrm{H}_V(\bar{\boldsymbol{z}})\boldsymbol{q} + \mathrm{O}(|\boldsymbol{q}|^3), \tag{A.3}$$

where H_V is the Hessian matrix of V_{tot}. We drop the constant term and neglect the higher-order terms in the following (harmonic approximation). This results in a quadratic many-body Hamiltonian with a well-known spectrum of phonon excitations.

We add two more ingredients to the system: An inhomogeneous magnetic field with constant gradient B_1 in z direction,

$$\boldsymbol{B}(z) = B(z)\hat{e}_z, \qquad B(z) = B_0 + B_1 z, \tag{A.4}$$

and the internal dynamics of ionic qubits with chosen bases specified by $\boldsymbol{m}_F \coloneqq (m_{F,1}, \ldots, m_{F,N})^T$. Plugging Eq. (A.4) into Eq. (A.1) yields a qubit-specific transition frequency that depends on both z_i and $m_{F,i}$. In the Hamiltonian, the splitting can be expressed using the Pauli Z operator of the qubit. Expanded to first order in $\boldsymbol{q}$, the corresponding term reads

$$\begin{aligned} -\frac{\hbar}{2}\sum_{i=1}^{N}\omega(m_{F,i}, B(z_i))Z_i &= -\frac{\hbar}{2}\sum_{i=1}^{N}\left(\omega_i^{(0)} + m_{F,i}\frac{\mu B_1}{\hbar}q_i\right)Z_i \\ &= -\frac{\hbar}{2}\sum_{i=1}^{N}\omega_i^{(0)}Z_i - \frac{\mu B_1}{2}\boldsymbol{q}^T(\boldsymbol{m}_F \circ \boldsymbol{Z})\,, \end{aligned} \tag{A.5}$$

with $\boldsymbol{Z} \coloneqq (Z_1, \ldots, Z_N)^T$ the local Z operators and frequencies $\omega_i^{(0)}$ that depend on $m_{F,i}$, $\bar{z}_i$ and various constants. The first term of the result is a typical Zeeman term involving only the qubits, while the second term couples external ($\boldsymbol{q}$) and internal ($\boldsymbol{Z}$) degrees of freedom. This last term is the central ingredient of magnetic gradient-induced coupling (MAGIC).

Collecting all contributions gives the Hamiltonian

$$H = T(\boldsymbol{p}) + \frac{1}{2}\boldsymbol{q}^T\,\mathrm{H}_V(\bar{\boldsymbol{z}})\boldsymbol{q} - \frac{\mu B_1}{2}\boldsymbol{q}^T(\boldsymbol{m}_F \circ \boldsymbol{Z}) - \frac{\hbar}{2}\sum_{i=1}^{N}\omega_i^{(0)}Z_i\,. \tag{A.6}$$

It is still second-order in the external degrees of freedom, but no longer purely quadratic. We fix this by completing the square in $\boldsymbol{q}$,

$$\begin{aligned} H = T(\boldsymbol{p}) &+ \frac{1}{2}\left(\boldsymbol{q} - \frac{\mu B_1}{2}\,\mathrm{H}_V^{-1}(\bar{\boldsymbol{z}})(\boldsymbol{m}_F \circ \boldsymbol{Z})\right)^T \mathrm{H}_V(\bar{\boldsymbol{z}})\left(\boldsymbol{q} - \frac{\mu B_1}{2}\,\mathrm{H}_V^{-1}(\bar{\boldsymbol{z}})(\boldsymbol{m}_F \circ \boldsymbol{Z})\right) \\ &- \frac{\hbar}{2}\sum_{i=1}^{N}\omega_i^{(0)}Z_i - \frac{1}{2}\left(\frac{\mu B_1}{2}\right)^2(\boldsymbol{m}_F \circ \boldsymbol{Z})^T\,\mathrm{H}_V^{-1}(\bar{\boldsymbol{z}})(\boldsymbol{m}_F \circ \boldsymbol{Z})\,. \end{aligned} \tag{A.7}$$

The existence of the inverse Hessian $\mathrm{H}_V^{-1}(\bar{\boldsymbol{z}})$ is guaranteed by the assumption of a stable equilibrium, which implies that H_V is positive definite. Note also that H_V and its inverse are symmetric matrices. In Eq. (A.7), it is evident that the first line is quadratic in the external degrees of freedom "up to a state dependent translation", while the second line only involves the qubits. Formally, this means that we can conjugate the Hamiltonian with the unitary

$$U \coloneqq \exp\left(\mathrm{i}\,\frac{\mu B_1}{2\hbar}\boldsymbol{p}^T\,\mathrm{H}_V^{-1}(\bar{\boldsymbol{z}})(\boldsymbol{m}_F \circ \boldsymbol{Z})\right) \tag{A.8}$$

to separate external and internal dynamics. Noting that U commutes with all terms of H except for the one involving $\boldsymbol{q}$, this results in

$$\tilde{H} \coloneqq UHU^\dagger = H_{\mathrm{phonons}}(\boldsymbol{p}, \boldsymbol{q}) + H_{\mathrm{qubits}}(\boldsymbol{Z}), \tag{A.9}$$

with $H_{\mathrm{phonons}}(\boldsymbol{p}, \boldsymbol{q})$ being a standard many-body phonon Hamiltonian that is easily solvable in normal coordinates. The internal part is simply the second line of Eq. (A.7), which we rewrite as

$$H_{\mathrm{qubits}}(\boldsymbol{Z}) = -\frac{\hbar}{2}\sum_{i=1}^{N}\omega_i^{(0)}Z_i - \frac{\hbar}{2}\boldsymbol{Z}^T J(\boldsymbol{m}_F)\boldsymbol{Z}. \tag{A.10}$$

Here, we used the Hadamard product identity (for vectors $\boldsymbol{a}, \boldsymbol{b}$ and a matrix C)

$$(\boldsymbol{a} \circ \boldsymbol{b})^T C(\boldsymbol{a} \circ \boldsymbol{b}) = \boldsymbol{a}^T (C \circ \boldsymbol{b}\boldsymbol{b}^T)\boldsymbol{a} \tag{A.11}$$

and defined

$$\begin{aligned} J(\boldsymbol{m}_F) &\coloneqq J \circ \boldsymbol{m}_F \boldsymbol{m}_F^T, \\ J &\coloneqq \frac{1}{\hbar}\left(\frac{\mu B_1}{2}\right)^2 \mathrm{H}_V^{-1}(\bar{\boldsymbol{z}}). \end{aligned} \tag{A.12}$$

The two terms of Eq. (A.10) commute, so we can treat them separately, or remove the first one entirely by another unitary transformation. In any case, the local operators Z_i do not entangle the qubits, so we focus our analysis on the second term. The diagonal entries of $J(\boldsymbol{m}_F)$ simply generate a constant term $-\frac{\hbar}{2}\operatorname{Tr}[J(\boldsymbol{m}_F)]\,\mathbb{1}$, which amounts to a global and hence unobservable phase. Without loss of generality, we can thus set the diagonal terms of J to zero. Finally, for the discussion on the logical layer, we choose units in which $\hbar = 1$. After these simplifying steps, Eq. (A.10) becomes the Hamiltonian (1.1) on which our analysis is based.

We provide Python code for the computation of coupling matrices based on the trap potential and physical parameters on GitHub [37].

A.3 Microwaves and single-qubit gates

A comprehensive discussion of the physics of microwave-controlled single-qubit gates, in particular within the MAGIC scheme, goes even beyond the scope of this appendix. It involves common concepts from quantum optics like the rotating wave approximation and Lamb-Dicke parameter, extended to the situation with a permanent inhomogeneous magnetic field. This extension is crucial, because it is the magnetic gradient that enables reasonable gate times even for microwave radiation which are otherwise much slower than optical gates. We recommend Refs. [15, 36] for more details.

Important for our analysis is that resonant microwave excitation of a hyperfine transition induces a Rabi Hamiltonian on the corresponding qubit,

$$H_{\mathrm{Rabi}}(\Omega, \phi) = \frac{\hbar\Omega}{2}\big(\cos(\phi)X + \sin(\phi)Y\big). \tag{A.13}$$

Here, X and Y are Pauli operators, Ω is the Rabi frequency which is proportional to the microwave amplitude, and ϕ is a parameter controlled by the relative phase between the microwave and essentially the qubit's Larmor precession. Resonant excitation means that the microwave frequency matches the transition frequency $\omega(m_F, B)$ of the targeted qubit. While the microwave is switched on, one (or several, if the experimental setup allows for multitone signals) Hamiltonian(s) of the form (A.13) act simultaneously with (A.10), potentially with different Rabi frequencies Ω_i. As these Hamiltonians do not commute, the time evolution of the system usually eludes exact analytical treatment. However, in applications one aims to make the Rabi frequencies orders of magnitude larger than the entries of J, which govern the entangling dynamics. This allows for the approximation that the gates induced by the Hamiltonian (A.13) are instantaneous, or equivalently that the Hamiltonian (A.10) is negligible while the microwave is on. The validity of this approximation and the errors introduced by it are discussed in Section 2.2 and Ref. [26]. Time evolution under the Rabi Hamiltonian (A.13) for a time θ/Ω implements a family of Bloch rotation gates,

$$\mathsf{R}(\theta, \phi) \coloneqq \exp\left(-\frac{\mathrm{i}\theta}{\hbar\Omega} H_{\mathrm{Rabi}}(\Omega, \phi)\right). \tag{A.14}$$

On the Bloch sphere, they rotate by an angle θ around an axis in the xy-plane specified by azimuth ϕ. The X and H gates necessary for our analysis are generated by

$$\mathsf{X} = \mathrm{i}\,\mathsf{R}(\pi, 0), \qquad \mathsf{H} = \mathrm{i}\,\mathsf{R}(\pi, 0)\,\mathsf{R}\left(\frac{\pi}{2}, \frac{\pi}{2}\right). \tag{A.15}$$

A.4 Physical and virtual recoding

As mentioned before, the π qubit with $m_F = 0$ is called "magnetic insensitive" because it depends on B only in second order. This can be used to exclude (up to quadratic corrections) qubits from the interaction and perform "subset operations". Corresponding rows and columns in the coupling matrix $J(\boldsymbol{m}_F)$ in Eq. (A.12) manifestly vanish. However, insensitivity also means that all π qubits have roughly the same transition frequency, see Eq. (A.1). Hence, they are not separately addressable, and we cannot perform independent single-qubit operations on them. The advantages and disadvantages of π qubits naturally pose the question, whether one can change the qubit basis during computation.

This can be answered in the affirmative, but one intermediately has to consider (at least) three-dimensional subspaces instead of qubits. Let $\mathsf{X}_\pm$ denote the X gate on the $\sigma^\pm$ qubit of the same ion. These gates "swap" the two states on which they act, and one can easily confirm that the local sequences $\mathsf{X}_+\,\mathsf{X}_-\,\mathsf{X}_+$ and $\mathsf{X}_-\,\mathsf{X}_+\,\mathsf{X}_-$ both effectively swap the states $|F=1, m_F=+1\rangle$ and $|F=1, m_F=-1\rangle$. Thus, they "recode" between the σ^+ and σ^- qubit.

Coding into and out of the π qubit is slightly more complicated due to the mentioned degenerate transitions. Yet we can use the global swap-like operation $\mathsf{X}_0^{\otimes N}$, which applies an X gate to all π qubits at once (assuming that shifts due to the quadratic Zeeman effect do not become prohibitive). Sequences of the form

$$\mathsf{X}_0^{\otimes N}\left(\ldots\otimes\mathsf{I}\otimes\ldots\otimes\mathsf{X}_+\otimes\ldots\otimes\mathsf{X}_-\otimes\ldots\right)\mathsf{X}_0^{\otimes N}, \tag{A.16}$$

i.e. with an arbitrary combination of local $\mathsf{X}_\pm$ and idle gates in between two global $\mathsf{X}_0^{\otimes N}$ gates, achieve a recoding between π and $\sigma^\pm$ qubits (for an $\mathsf{X}_\pm$ gate) or just swap back and forth, effectively doing nothing (for an idle gate).

The described sequences of three rounds of microwave gates are able to physically manipulate the way quantum information is encoded in the ions. However, our main concern is not how information is stored, but how it can be processed. To that end one usually does not need to recode between the physical σ^+ and σ^- qubits, but can instead use single layers of X gates (on the physical qubit in use) to emulate the opposite encoding. This can significantly reduce the number of single-qubit gates in a circuit, and we make heavy use of this possibility. Using the notation of Eq. (1.3), a detailed derivation of this technique reads

$$\begin{aligned}
\mathsf{X}^{\boldsymbol{s}}\exp\left(\frac{\mathrm{i}t}{2}\boldsymbol{Z}^T J(\boldsymbol{m}_F)\boldsymbol{Z}\right)\mathsf{X}^{\boldsymbol{s}} &= \exp\left(\frac{\mathrm{i}t}{2}X^{\boldsymbol{s}}\boldsymbol{Z}^T X^{\boldsymbol{s}} J(\boldsymbol{m}_F) X^{\boldsymbol{s}}\boldsymbol{Z}X^{\boldsymbol{s}}\right)\\
&= \exp\left(\frac{\mathrm{i}t}{2}\left((-1)^{\boldsymbol{s}}\circ\boldsymbol{Z}\right)^T J(\boldsymbol{m}_F)\left((-1)^{\boldsymbol{s}}\circ\boldsymbol{Z}\right)\right)\\
&= \exp\left(\frac{\mathrm{i}t}{2}\boldsymbol{Z}^T\left(J(\boldsymbol{m}_F)\circ\left((-1)^{\boldsymbol{s}}\left((-1)^{\boldsymbol{s}}\right)^T\right)\right)\boldsymbol{Z}\right)\\
&= \exp\left(\frac{\mathrm{i}t}{2}\boldsymbol{Z}^T J\big((-1)^{\boldsymbol{s}}\circ\boldsymbol{m}_F\big)\boldsymbol{Z}\right),
\end{aligned} \tag{A.17}$$

where we used that Pauli strings $X^{\boldsymbol{s}}$ are self-inverse and applied the Hadamard product identity Eq. (A.11). Negative signs in $(-1)^{\boldsymbol{s}}$ exchange, from the perspective of the J coupling, the σ qubits, but leave π qubits invariant. If this technique is used consecutively, subsequent X gate layers can be combined into a single one,

$$\mathsf{X}^{\boldsymbol{s}}\,\mathsf{X}^{\boldsymbol{s}'} = \mathsf{X}^{\boldsymbol{s}\oplus\boldsymbol{s}'}, \tag{A.18}$$

saving even more single-qubit gates, as detailed in the main text.

B Frame theory

In this section we connect our approach to frame theory, which is concerned with spanning sets for Hilbert spaces, i.e. with generalizations of the notion of a basis, see e.g. Ref. [67].

We consider the space of symmetric matrices with vanishing diagonal that is defined by

$$\mathrm{Sym}_0(\mathbb{R}^n) \coloneqq \{M\in\mathrm{Sym}(\mathbb{R}^n)\mid M_{ii}=0\ \forall i\in[n]\}\,. \tag{B.1}$$

Moreover, we denote the set of outer products of all possible encodings $\boldsymbol{m} = (-1)^{\boldsymbol{b}}$, where $\boldsymbol{b} \in \mathbb{F}_2^n$, by

$$\mathcal{V} \coloneqq \left\{ \boldsymbol{m}\boldsymbol{m}^T \,\middle|\, \boldsymbol{m} \in \{-1,+1\}^n, m_n = +1 \right\}. \tag{B.2}$$

We further define what it means for a frame to be harmonic:

Definition B.1. *[67, Definition 11.1] Let G be a finite abelian group, and let $\hat{G}$ be the set of irreducible characters of G. A tight frame $\{\phi_i\}_{i\in J}$ for $\mathbb{R}^k$ is called a* harmonic *if it is unitarily equivalent to*

$$(\xi|_J)_{\xi\in\hat{G}} \subset \mathbb{R}^{|J|} \equiv \mathbb{R}^k, \tag{B.3}$$

where $J \subset G$ and $|J| = k$.

The following theorem extends Theorem 2.2 and shows that $\mathcal{V}$ is a frame with certain properties.

Theorem B.2. *$\mathcal{V}$ is a balanced equal-norm harmonic tight frame for* $\mathrm{Sym}_0(\mathbb{R}^n)$.

Proof. Let $k = n(n-1)/2 = \dim(\mathrm{Sym}_0(\mathbb{R}^n))$ and $l = 2^{n-1} = |\mathcal{V}|$. We denote the synthesis operator of $\mathcal{V}$ by $V : \mathbb{R}^l_{\geq 0} \to \mathrm{Sym}_0(\mathbb{R}^n) : \boldsymbol{\lambda} \mapsto \sum_{\boldsymbol{m}} \lambda_{\boldsymbol{m}} \boldsymbol{m}\boldsymbol{m}^T$ which can be represented by a matrix $V \in \{-1,+1\}^{l\times k}$. We further denote the column $\boldsymbol{v}(\boldsymbol{m})$ of V, containing the lower triangular elements of $\boldsymbol{m}\boldsymbol{m}^T$. First we want to show that each row of V corresponds to a row of a Hadamard matrix. Since the elements of any Hadamard matrix $H_{\boldsymbol{y},\boldsymbol{x}}$ (with normalization factor) are given by Walsh functions $H_{\boldsymbol{y},\boldsymbol{x}} = 2^{-n/2} W_{\boldsymbol{y}}(\boldsymbol{x}) = 2^{-n/2}(-1)^{\boldsymbol{y}\cdot\boldsymbol{x}}$ for $\boldsymbol{y}, \boldsymbol{x} \in \mathbb{F}_2^n$, this amounts to showing that the rows of V correspond to Walsh functions $W_{\boldsymbol{y}}(\boldsymbol{x}) \coloneqq (-1)^{\boldsymbol{y}\cdot\boldsymbol{x}}$, where $\boldsymbol{y}\cdot\boldsymbol{x} = \sum_i^n y_i x_i$.

Writing $\boldsymbol{m} = (-1)^{\boldsymbol{b}}$, the components of the matrix $\boldsymbol{m}\boldsymbol{m}^T$ are $(\boldsymbol{m}\boldsymbol{m}^T)_{gh} = (-1)^{b_g \oplus b_h}$. Then the entries of the columns $\boldsymbol{v}(\boldsymbol{m})$ are $\boldsymbol{v}(\boldsymbol{m})_{gh} = (-1)^{b_g\oplus b_h} = (-1)^{\boldsymbol{b}\cdot(\boldsymbol{e}^g \oplus \boldsymbol{e}^h)}$, where the tuple g, h is the index of the row of V and $\boldsymbol{e}^i \in \mathbb{F}_2^n$ denotes the i-th standard unit vector.

This shows that $\boldsymbol{v}(\boldsymbol{m})_{gh} = W_{\boldsymbol{b}}(\boldsymbol{e}^g \oplus \boldsymbol{e}^h)$. So the $\boldsymbol{e}^g \oplus \boldsymbol{e}^h$ encodes the row indices and the $\boldsymbol{b}$ encodes the column indices of the Hadamard matrix. Since we consider all $\boldsymbol{b} \in \mathbb{F}_2^n$ we have all columns of the Hadamard matrix. This shows that each row of V corresponds to a row of a Hadamard matrix.

Since the rows/columns of a Hadamard matrix form an orthonormal basis one can use the row construction of tight frames from orthogonal projections [67, Theorem 2.3] to get an equal-norm tight frame. Furthermore, the Hadamard matrix is the character table of the cyclic group $C_2^n = C_2 \times \cdots \times C_2$ which is abelian. This can be seen by its recursive definition $H_n = H_1 \otimes H_{n-1}$, where H_1 is the character table of C_2. Thus, Ref. [67, Theorem 11.1] implies that this equal-norm tight frame is also a harmonic.

It is balanced if $\sum_{\boldsymbol{b}}(-1)^{b_g\oplus b_h} = 0$ for all $g, h = 1, \ldots, n$ with $g \neq h$. Since we only consider the lower/upper triangular matrix without the diagonal $g = h$, our frame is indeed balanced. □

Corollary B.3. *The normalized frame $\{\boldsymbol{m}\boldsymbol{m}^T/\sqrt{k}\}_{\boldsymbol{m}\in\{-1,+1\}^n}$ for* $\mathrm{Sym}_0(\mathbb{R}^n)$ *forms a spherical 2-design.*

Proof. The elements of the normalized frame $\{\boldsymbol{m}\boldsymbol{m}^T/\sqrt{k}\}_{\boldsymbol{m}\in\{-1,+1\}^n}$ constitute a set of unit vectors in $\mathbb{R}^k$ which form a normalized balanced tight frame by Theorem B.2. Then, by Refs. [68, Proposition 1.2] and [67, Proposition 6.1], $\{\boldsymbol{m}\boldsymbol{m}^T/\sqrt{k}\}_{\boldsymbol{m}\in\{-1,+1\}^n}$ is also a spherical 2-design. □

Theorem B.4. *The entries of the Gram matrix $P = V^T V$ are given by*

$$P_{\boldsymbol{m},\boldsymbol{m}'} = \langle \boldsymbol{v}(\boldsymbol{m}), \boldsymbol{v}(\boldsymbol{m}')\rangle = \frac{n}{2}(n-1) - 2\Delta_{\boldsymbol{b},\boldsymbol{b}'}(n - \Delta_{\boldsymbol{b},\boldsymbol{b}'}), \tag{B.4}$$

where $\boldsymbol{v}(\boldsymbol{m})$ are the columns of V, containing the lower triangular elements of $\boldsymbol{m}\boldsymbol{m}^T$, and $\Delta_{\boldsymbol{b},\boldsymbol{b}'} \coloneqq |\boldsymbol{b} \oplus \boldsymbol{b}'|$ is the Hamming distance between $\boldsymbol{b}$ and $\boldsymbol{b}'$.

Proof. As in the proof above, write the (g,h)-th entry of the column $\boldsymbol{v}(\boldsymbol{m})$ as

$$\boldsymbol{v}(\boldsymbol{m})_{gh} = (-1)^{b_g \oplus b_h}. \tag{B.5}$$

This yields

$$\begin{aligned}\langle \boldsymbol{v}(\boldsymbol{m}), \boldsymbol{v}(\boldsymbol{m}')\rangle &= \sum_{g<h} (-1)^{b_g \oplus b_h} (-1)^{b'_g \oplus b'_h} \\ &= \sum_{g<h} (-1)^{b_g \oplus b'_g \oplus b_h \oplus b'_h} \\ &= \sum_{g<h} (-1)^{c_g \oplus c_h} ,\end{aligned} \tag{B.6}$$

where we set $\boldsymbol{c} \coloneqq \boldsymbol{b} \oplus \boldsymbol{b}'$. The summand after the last equation is the outer product $\tilde{\boldsymbol{m}}\tilde{\boldsymbol{m}}^T$ for $\tilde{\boldsymbol{m}} \in \{-1, +1\}^n$ and $\tilde{\boldsymbol{m}} = (-1)^{\boldsymbol{c}}$. If the Hamming distance $\Delta_{\boldsymbol{b},\boldsymbol{b}'}$ vanishes, then $\tilde{\boldsymbol{m}} = (+1, \ldots, +1)$ and $\langle \boldsymbol{v}(\boldsymbol{m}), \boldsymbol{v}(\boldsymbol{m}')\rangle = n(n-1)/2$ which is the maximal entry of the Gram matrix located at the diagonal. If $\Delta_{\boldsymbol{b},\boldsymbol{b}'} \neq 0$, $\tilde{\boldsymbol{m}}$ contains $\Delta_{\boldsymbol{b},\boldsymbol{b}'}$-many summands -1, such that the lower triangular part of $\tilde{\boldsymbol{m}}\tilde{\boldsymbol{m}}^T$ contains $\Delta_{\boldsymbol{b},\boldsymbol{b}'}(n - \Delta_{\boldsymbol{b},\boldsymbol{b}'})$-many summands -1. Therefore,

$$\langle \boldsymbol{v}(\boldsymbol{m}), \boldsymbol{v}(\boldsymbol{m}')\rangle = \frac{n}{2}(n-1) - 2\Delta_{\boldsymbol{b},\boldsymbol{b}'}(n - \Delta_{\boldsymbol{b},\boldsymbol{b}'}) . \tag{B.7}$$

□

C Convex optimization arguments for the linear program (LP)

We investigate the sparsity and geometric properties of the optimal solutions of the LP (2.5). First, we prove Proposition 2.3, which is a standard result in linear programming:

Proposition C.1 (Sparse optimal solutions)**.** *There is an optimal solution to the LP (2.5) with sparsity $\leq n(n-1)/2$ for every $M \in \mathrm{Sym}_0(\mathbb{R}^n)$. The simplex algorithm is guaranteed to return such an optimal solution.*

Proof. The LP (2.5) has $m = n(n-1)/2$ equality constraints. The feasible polytope is the convex polytope obtained by intersecting the $(n-m)$-dimensional subspace defined by those with the positive cone $x \geq 0$. To define a vertex of the feasible polytope, a point x has to saturate at least $n - m$ many inequalities $x_i \geq 0$, hence, it has at most $n - (n-m) = m$ many non-zero entries. The optimal solutions are obtained by minimizing over the feasible polytope and thus correspond, in general, to a face of this polytope. Any vertex of this face defines an optimal solution which has to be at least $m = n(n-1)/2$-sparse. The last statement follows since the simplex algorithm only returns vertices of the feasible polytope. □

Alternative algorithms like interior point methods should be avoided. Since those are not guaranteed to return vertices of the feasible polytope if there is an entire face of optimal solutions, their solution will generally be a dense vector.

We can say a bit more about the solutions of the LP (2.5) by geometrical observations. These observations hold more generally for LPs of the following form.

Definition C.2. *Suppose that vectors $v_1, \ldots, v_N \in \mathbb{R}^d$ are the vertices of a full-dimensional polytope P and the origin is contained in the convex hull of P, i.e. $0 \in \mathrm{conv}(P)$. Given a vector $u \in \mathbb{R}^d$, we define the following linear program:*

$$\begin{aligned} \text{minimize} \quad & \langle \mathbf{1}, x\rangle = \sum_{i=1}^{N} x_i \\ \text{subject to} \quad & u = Vx, \\ & x \geq 0 \end{aligned} \tag{C.1}$$

where $V = \sum_{i=1}^{N} v_i e_i^\top \in \mathbb{R}^{d \times N}$.

Lemma C.3. *In the setting of Definition C.2, the following holds:*

(i) The point $u \in \mathbb{R}^d$ lies in a cone generated by a face F of P. In particular, there is a feasible solution x of the LP (C.1) with support only on the vertices of F.

(ii) *Every feasible solution with support on the vertices of F has the same objective value, namely $\langle f, u\rangle$ where f is a normal vector of F, i.e. $F \subset \{y \in \mathbb{R}^d \mid \langle f, y\rangle = 1\}$. This value does not depend on the choice of normal vector.*

(iii) *There is a feasible solution with sparsity $\leq \dim F + 1 \leq d$ and objective value $\langle f, u\rangle$.*

(iv) *The optimal solutions of the LP (C.1) correspond exactly to the possible conical combinations of u in $\operatorname{cone}(F^*) := \{\sum_i x_i v_i \mid v_i \in F^*, x_i \in \mathbb{R}_{\geq 0}\}$ where F^* is the lowest-dimensional face of P such that $u \in \operatorname{cone}(F^*)$. In particular, the minimum is given by $\langle f, u\rangle$ where f is some normal vector of F^*.*

Proof. Statement (i) follows from the observation that given a point $u \in \mathbb{R}^d \setminus 0$, there is a unique $\boldsymbol{\lambda} > 0$ such that $\boldsymbol{\lambda} u$ lies on a face F of P. Thus, $\operatorname{cone}(F)$ contains u. More precisely, the cones spanned by the facets of P form a partition of $\mathbb{R}^d$ where the intersections between any two cones is either $\{0\}$ or a cone spanned by a lower-dimensional face of P.

For statement (ii), let w.l.o.g. $v_1, \ldots, v_s$ be the vertices that lie in F. By assumption, $u \in \operatorname{cone}(v_1, \ldots, v_s)$ and hence $u = \sum_{i=1}^{s} x_i v_i$. We find

$$\langle f, u\rangle = \sum_{i=1}^{s} x_i \langle f, v_i\rangle = \sum_{i=1}^{s} x_i = \langle \mathbf{1}, x\rangle. \tag{C.2}$$

Note that the objective value $\langle f, u\rangle$ is necessarily the same, for any choice of normal vector.

Statement (iii) follows by triangulating the face F with simplices. These simplices have $\dim F+1$ vertices and u has to lie in the cone spanned by one of those. Thus, there is a feasible solution with sparsity at most $\dim F + 1 \leq d$ and objective value $\langle f, u\rangle$.

Finally, let x^* be an optimal solution. Suppose that there is a $i \in \operatorname{supp}(x^*)$ such that the vertex v_i is not in F. Then, we can find a supporting hyperplane of F with normal vector f such that $\langle f, v_i\rangle < 1$. We thus have

$$\langle f, u\rangle = \sum_{j \neq i} x_j^* \langle f, v_j\rangle + x_i^* \langle f, v_i\rangle < \sum_j x_j^* = \langle \mathbf{1}, x^*\rangle. \tag{C.3}$$

According to (ii), $\langle f, u\rangle$ is the objective value of a feasible solution with support on F. Hence, x^* could not have been optimal, and all optimal solutions have to have support on F. Clearly, all previous arguments still hold if we find a face $F' \subset F$ of P such that $u \in \operatorname{cone}(F')$ and hence the statement follows. □

D Acronyms

DAQC digital-analog quantum computing . . . 3
EASE efficient, arbitrary, simultaneously entangling . . . 2
LP linear program . . . 2
MAGIC magnetic gradient-induced coupling . . . 1
MIP mixed integer program . . . 2
MS Mølmer-Sørensen . . . 2
NISQ noisy and intermediate-scale quantum . . . 1
POVM positive operator valued measure . . . 12
QFT quantum Fourier transform . . . 2
QPU quantum processing unit . . . 25

References

[1] J. Preskill, *Quantum computing in the NISQ era and beyond*, Quantum **2**, 79 (2018), arXiv:1801.00862.

[2] D. A. Lidar and T. A. Brun, *Quantum error correction* (Cambridge University Press, 2013).
[3] X. Wang, A. Sørensen, and K. Mølmer, *Multibit gates for quantum computing*, Phys. Rev. Lett. **86**, 3907 (2001), arXiv:quant-ph/0012055.
[4] T. Monz, P. Schindler, J. T. Barreiro, M. Chwalla, D. Nigg, W. A. Coish, M. Harlander, W. Hänsel, M. Hennrich, and R. Blatt, *14-qubit entanglement: Creation and coherence*, Phys. Rev. Lett. **106**, 130506 (2011), arXiv:1009.6126.
[5] N. M. Linke, D. Maslov, M. Roetteler, S. Debnath, C. Figgatt, K. A. Landsman, K. Wright, and C. Monroe, *Experimental comparison of two quantum computing architectures*, PNAS **114**, 3305 (2017), arXiv:1702.01852.
[6] E. A. Martinez, T. Monz, D. Nigg, P. Schindler, and R. Blatt, *Compiling quantum algorithms for architectures with multi-qubit gates*, New J. Phys. **18**, 063029 (2016), arXiv:1601.06819.
[7] D. Maslov and Y. Nam, *Use of global interactions in efficient quantum circuit constructions*, New J. Phys. **20**, 033018 (2018), arXiv:1707.06356.
[8] J. van de Wetering, *Constructing quantum circuits with global gates*, New J. Phys. **23**, 043015 (2021), arXiv:2012.09061.
[9] N. Grzesiak, A. Maksymov, P. Niroula, and Y. Nam, *Efficient quantum programming using EASE gates on a trapped-ion quantum computer*, Quantum **6**, 634 (2022), arXiv:2107.07591.
[10] S. Bravyi, D. Maslov, and Y. Nam, *Constant-cost implementations of Clifford operations and multiply controlled gates using global interactions*, Phys. Rev. Lett. **129**, 230501 (2022), arXiv:2207.08691.
[11] C. Figgatt, A. Ostrander, N. M. Linke, K. A. Landsman, D. Zhu, D. Maslov, and C. Monroe, *Parallel entangling operations on a universal ion-trap quantum computer*, Nature **572**, 368 (2019), arXiv:1810.11948.
[12] Y. Lu, S. Zhang, K. Zhang, W. Chen, Y. Shen, J. Zhang, J.-N. Zhang, and K. Kim, *Global entangling gates on arbitrary ion qubits*, Nature **572**, 363 (2019), arXiv:1901.03508.
[13] N. Grzesiak, R. Blümel, K. Wright, K. M. Beck, N. C. Pisenti, M. Li, V. Chaplin, J. M. Amini, S. Debnath, J.-S. Chen, and Y. Nam, *Efficient arbitrary simultaneously entangling gates on a trapped-ion quantum computer*, Nat. Commun. **11**, 2963 (2020), arXiv:1905.09294.
[14] F. Mintert and C. Wunderlich, *Ion-trap quantum logic using long-wavelength radiation*, Phys. Rev. Lett. **87**, 257904 (2001), arXiv:quant-ph/0104041.
[15] C. Wunderlich, *Conditional spin resonance with trapped ions*, in *Laser Physics at the Limits*, edited by H. Figger, C. Zimmermann, and D. Meschede (Springer, 2002) pp. 261–273, arXiv:quant-ph/0111158.
[16] N. Timoney, I. Baumgart, M. Johanning, A. F. Varon, C. Wunderlich, M. B. Plenio, and A. Retzker, *Quantum Gates and Memory using Microwave Dressed States*, Nature **476**, 185 (2011), arXiv:1105.1146.
[17] C. Ospelkaus, U. Warring, Y. Colombe, K. R. Brown, J. M. Amini, D. Leibfried, and D. J. Wineland, *Microwave quantum logic gates for trapped ions*, Nature **476**, 181 (2011), arXiv:1104.3573.
[18] A. Khromova, C. Piltz, B. Scharfenberger, T. Gloger, M. Johanning, A. Varón, and C. Wunderlich, *Designer spin pseudomolecule implemented with trapped ions in a magnetic gradient*, Phys. Rev. Lett. **108**, 220502 (2012), arXiv:1112.5302.
[19] C. Piltz, T. Sriarunothai, S. S. Ivanov, S. Wölk, and C. Wunderlich, *Versatile microwave-driven trapped ion spin system for quantum information processing*, Sci. Adv. **2**, e1600093 (2016), arXiv:1509.01478.
[20] B. Lekitsch, S. Weidt, A. G. Fowler, K. Mølmer, S. J. Devitt, C. Wunderlich, and W. K. Hensinger, *Blueprint for a microwave trapped-ion quantum computer*, Sci. Adv. **3**, e1601540 (2017), arXiv:1508.00420.
[21] S. Wölk and C. Wunderlich, *Quantum dynamics of trapped ions in a dynamic field gradient using dressed states*, New J. Phys. **19**, 083021 (2017), arXiv:1606.04821.
[22] J. Cohn, M. Motta, and R. M. Parrish, *Quantum filter diagonalization with compressed double-factorized Hamiltonians*, PRX Quantum **2**, 040352 (2021), arXiv:2104.08957.
[23] A. Sørensen and K. Mølmer, *Spin-spin interaction and spin squeezing in an optical lattice*, Phys. Rev. Lett. **83**, 2274 (1999), arXiv:quant-ph/9903044.
[24] A. Sørensen and K. Mølmer, *Entanglement and quantum computation with ions in thermal motion*, Phys. Rev. A **62**, 022311 (2000), arXiv:quant-ph/0002024.

[25] T. Choi, S. Debnath, T. A. Manning, C. Figgatt, Z. X. Gong, L. M. Duan, and C. Monroe, *Optimal quantum control of multimode couplings between trapped ion qubits for scalable entanglement*, Phys. Rev. Lett. **112**, 190502 (2014), arXiv:1401.1575.
[26] A. Parra-Rodriguez, P. Lougovski, L. Lamata, E. Solano, and M. Sanz, *Digital-analog quantum computation*, Phys. Rev. A **101**, 022305 (2020), arXiv:1812.03637.
[27] D. Maslov and B. Zindorf, *Depth optimization of CZ, CNOT, and Clifford circuits*, IEEE Transactions on Quantum Engineering **3**, 1 (2022), arXiv:2201.05215.
[28] S. S. Ivanov, M. Johanning, and C. Wunderlich, *Simplified implementation of the quantum Fourier transform with Ising-type Hamiltonians: Example with ion traps*, arXiv:1503.08806 (2015).
[29] G. Breit and I. I. Rabi, *Measurement of nuclear spin*, Phys. Rev. **38**, 2082 (1931).
[30] S. Bourdeauducq, whitequark, R. Jördens, D. Nadlinger, Y. Sionneau, and F. Kermarrec, *ARTIQ* 10.5281/zenodo.6619071 (2021), Version 6.
[31] I. Dumer, D. Micciancio, and M. Sudan, *Hardness of approximating the minimum distance of a linear code*, IEEE Trans. Inf. Theory **49**, 22 (2003).
[32] N. Christofides, *Worst-case analysis of a new heuristic for the travelling salesman problem*, Tech. Rep. (Defense Technical Information Center, 1976).
[33] H. Karloff, *Linear Programming* (Birkhäuser Boston, 1991) pp. 23–47.
[34] M. Kliesch and I. Roth, *Theory of quantum system certification*, PRX Quantum **2**, 010201 (2021), tutorial, arXiv:2010.05925.
[35] N. B. Karahanoğlu, H. Erdoğan, and Ş. İ. Birbil, *A mixed integer linear programming formulation for the sparse recovery problem in compressed sensing*, in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing* (2013) pp. 5870–5874.
[36] M. Johanning, A. F. Varón, and C. Wunderlich, *Quantum simulations with cold trapped ions*, J. Phys. B **42**, 154009 (2009), arXiv:0905.0118.
[37] P. Baßler and M. Zipper, *Source code for "Synthesis of and compilation with time-optimal multi-qubit gates"*, `https://github.com/matt-zipp/arXiv-2206.06387` (2022).
[38] S. Diamond and S. Boyd, *CVXPY: A Python-embedded modeling language for convex optimization*, J. Mach. Learn. Res. **17**, 1 (2016).
[39] A. Agrawal, R. Verschueren, S. Diamond, and S. Boyd, *A rewriting system for convex optimization problems*, J. Control Decis. **5**, 42 (2018), arXiv:1709.04494.
[40] Free Software Foundation, *GLPK (GNU Linear Programming Kit)* (2012), version: 0.4.6.
[41] MOSEK ApS, *MOSEK Optimizer API for Python 9.3.14* (2022).
[42] S. Kukita, H. Kiya, and Y. Kondo, *Short composite quantum gate robust against two common systematic errors*, J. Phys. Soc. Japan **91**, 104001 (2022), arXiv:2112.12945.
[43] D. A. Spielman and S.-H. Teng, *Smoothed analysis of algorithms: Why the simplex algorithm usually takes polynomial time*, Journal of the ACM **51**, 385 (2004), arXiv:cs/0111050.
[44] S. Bravyi and D. Maslov, *Hadamard-free circuits expose the structure of the Clifford group*, IEEE Trans. Inf. Theory **67**, 4546 (2021), arXiv:2003.09412.
[45] S. Aaronson and D. Gottesman, *Improved simulation of stabilizer circuits*, Phys. Rev. A **70**, 052328 (2004), arXiv:quant-ph/0406196.
[46] D. Maslov and M. Roetteler, *Shorter stabilizer circuits via Bruhat decomposition and quantum circuit transformations*, IEEE Trans. Inf. Theory **64**, 4729 (2018), arXiv:1705.09176.
[47] R. Duncan, A. Kissinger, S. Perdrix, and J. van de Wetering, *Graph-theoretic simplification of quantum circuits with the ZX-calculus*, Quantum **4**, 279 (2020), arXiv:1902.03178.
[48] R. Kueng and D. Gross, *Qubit stabilizer states are complex projective 3-designs*, arXiv:1510.02767.
[49] H.-Y. Huang, R. Kueng, and J. Preskill, *Predicting many properties of a quantum system from very few measurements*, Nat. Phys. **16**, 1050 (2020), arXiv:2002.08953.
[50] D. Schlingemann, *Stabilizer codes can be realized as graph codes*, arXiv:quant-ph/0111080 (2001).
[51] D. Schlingemann and R. F. Werner, *Quantum error-correcting codes associated with graphs*, Phys. Rev. A **65**, 012308 (2001), arXiv:quant-ph/0012111.
[52] V. Kliuchnikov, K. Lauter, R. Minko, A. Paetznick, and C. Petit, *Shorter quantum circuits*, arXiv:2203.10064 (2022).
[53] M. Born and R. Oppenheimer, *Zur Quantentheorie der Molekeln*, Ann. Phys. **389**, 457 (1927).

[54] J. Kempe, A. Kitaev, and O. Regev, *The complexity of the local Hamiltonian problem*, SIAM J. Comput. **35**, 1070 (2006), arXiv:quant-ph/0406180.
[55] R. M. Parrish and P. L. McMahon, *Quantum filter diagonalization: Quantum eigendecomposition without full quantum phase estimation*, arXiv:1909.08925 (2019).
[56] K. Klymko, C. Mejuto-Zaera, S. J. Cotton, F. Wudarski, M. Urbanek, D. Hait, M. Head-Gordon, K. B. Whaley, J. Moussa, N. Wiebe, W. A. de Jong, and N. M. Tubman, *Real time evolution for ultracompact Hamiltonian eigenstates on quantum hardware*, PRX Quantum **3**, 020323 (2022), arXiv:2103.08563.
[57] N. H. Stair, R. Huang, and F. A. Evangelista, *A multireference quantum Krylov algorithm for strongly correlated electrons*, J. Chem. Theory Comput. **16**, 2236 (2020), pMID: 32091895.
[58] V. Kliuchnikov, D. Maslov, and M. Mosca, *Fast and efficient exact synthesis of single qubit unitaries generated by Clifford and T gates*, Quantum Inform. Comput. **13**, 607 (2013), arXiv:1206.5236.
[59] N. J. Ross and P. Selinger, *Optimal ancilla-free Clifford+T approximation of z-rotations*, Quantum Inform. Compu. **16**, 901 (2016), arXiv:1403.2975.
[60] A. Bouland and T. Giurgica-Tiron, *Efficient universal quantum compilation: An inverse-free Solovay-Kitaev algorithm*, arXiv:2112.02040.
[61] M. Amy, P. Azimzadeh, and M. Mosca, *On the CNOT-complexity of CNOT-PHASE circuits*, Quantum Sci. Technol. **4**, 015002 (2018), arXiv:1712.01859.
[62] M. Amy, D. Maslov, and M. Mosca, *Polynomial-time T-depth optimization of Clifford+T circuits via matroid partitioning*, IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst. **33**, 1476 (2014), arXiv:1303.2042.
[63] R. O'Donnell, *Analysis of Boolean Functions* (Cambridge University Press) arXiv:2105.10386 .
[64] G. Dantzig, R. Fulkerson, and S. Johnson, *Solution of a large-scale traveling-salesman problem*, Operations Research Society of America **2**, 393 (1954).
[65] P. García-Molina, A. Martin, and M. Sanz, *Noise in digital and digital-analog quantum computation*, arXiv:2107.12969.
[66] P. T. Fisk, M. J. Sellars, M. A. Lawn, and G. Coles, *Accurate measurement of the 12.6 GHz "clock" transition in trapped* $^{171}Yb^{+}$ *ions*, IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control **44**, 344 (1997).
[67] S. F. D. Waldron, *An Introduction to Finite Tight Frames*, Applied and Numerical Harmonic Analysis (Springer New York, New York, NY, 2018).
[68] R. B. Holmes and V. I. Paulsen, *Optimal frames for erasures*, Linear Algebra Its Appl. **377**, 31 (2004).