Bonsai Algorithm: Grow Your Own Fermion-to-Qubit Mappings

Aaron Miller,[1,2,*] Zoltán Zimborás,[1,3] Stefan Knecht,[1,4] Sabrina Maniscalco,[1] and Guillermo García-Pérez[1]

[1] *Algorithmiq Ltd, Kanavakatu 3C, Helsinki FI-00160, Finland*
[2] *Trinity Quantum Alliance, Unit 16, Trinity Technology and Enterprise Centre, Pearse Street, Dublin 2, Ireland*
[3] *Wigner Research Centre for Physics, P.O. Box 49, Budapest H-1525, Hungary*
[4] *ETH Zürich, Laboratory for Physical Chemistry, Vladimir-Prelog-Weg 2, Zürich 8093, Switzerland*

(Received 21 February 2023; accepted 27 June 2023; published 1 August 2023)

Fermion-to-qubit mappings are used to represent fermionic modes on quantum computers, an essential first step in many quantum algorithms for electronic structure calculations. In this work, we present a formalism to design flexible fermion-to-qubit mappings from ternary trees. We intuitively discuss the connection between the structure of the generating trees and certain properties of the resulting mapping, such as the Pauli weight and the delocalization of mode occupation. Moreover, we introduce a recipe that guarantees that Fock basis states are mapped to computational basis states in qubit space, a desirable property for many applications in quantum computing. Based on this formalism, we introduce the Bonsai algorithm, which takes as input the potentially limited topology of the qubit connectivity of a quantum device and returns a tailored fermion-to-qubit mapping that reduces the SWAP overhead compared to other paradigmatic mappings. We illustrate the algorithm by producing mappings for the heavy-hexagon topology widely used in IBM quantum computers. The resulting mappings have a favorable Pauli-weight scaling $\mathcal{O}(\sqrt{N})$ on this connectivity while ensuring that no SWAP gates are necessary for single-excitation operations.

DOI: 10.1103/PRXQuantum.4.030314

I. INTRODUCTION

The field of quantum computing has witnessed astounding developments in the past decade. While the technology is improving quickly, so-called fault-tolerant quantum computing still seems a distant milestone. Current devices are limited to relatively few qubits and cannot reliably execute the deep circuits required by many paradigmatic quantum computing algorithms [1]. Yet, near-term computers with a few hundred qubits and low levels of noise can prepare entangled states that cannot be efficiently simulated classically, which can be a computational resource in itself if combined with classical compute appropriately [2–5], e.g., to mitigate the detrimental effects of noise [6–9]. This is the reasoning behind most hybrid quantum-classical computing approaches [10]. Within this hybrid framework, it was recognized early on that one application stands out from the rest in terms of suitability: the simulation of many-body fermionic quantum systems.

The solution of electronic structure problems with near-term devices is of paramount importance in fields such as computational chemistry [11–13], which itself has a crucial impact on many industries ranging from material science [14] to drug discovery [15,16], amongst many others. In many cases, the limitations of classical methods for chemistry stem from the inability to account for the complexity of electronic wave functions, which can easily involve a superposition of a combinatorially large number of electronic configurations. Given that quantum processors can physically exhibit complex superpositions, this problem seems particularly appropriate for them to tackle. Indeed, in most near-term approaches, the physical state of the device is taken to represent the state of the many-body fermionic system of interest and the physical properties of the latter are inferred by appropriately measuring physical properties of the former [17,18]. However, since the fermionic and the many-qubit wave functions live in fundamentally different Hilbert spaces, equipped with different algebraic structures, realization of this prospect calls for a concrete way to establish connections between them, the so-called fermion-to-qubit mappings. The type of

[*] aaron.miller@algorithmiq.fi

mapping between fermions and qubits used has a direct impact on the quantum simulation. A fermionic wave function $|\Psi_f\rangle$ will be encoded into different many-qubit states $|\Psi_q\rangle$ by different mappings and these different states will generally not be equally easy to prepare on a given quantum processor. Ultimately, the latter point depends on the specifics of the hardware used, e.g., its connectivity, coherence time, etc. Moreover, one is not generally interested in merely simulating state $|\Psi_f\rangle$ but in determining physical properties, $\langle\Psi_f|\mathcal{O}_f|\Psi_f\rangle$, where $\mathcal{O}_f$ is some fermionic quantity of interest. The fermion-to-qubit mapping of choice will map said operator into its qubit counterpart, $\mathcal{O}_q$, and the evaluation of $\langle\Psi_q|\mathcal{O}_q|\Psi_q\rangle$ will be carried out via physical measurements on the device. Therefore, in general, the measurement cost incurred will also depend on the choice of mapping. The importance of the encoding has thus motivated considerable research toward designing convenient fermion-to-qubit mappings beyond the paradigmatic ones, such as Jordan-Wigner (JW) [19], Bravyi-Kitaev (BK) [20], and parity [21]. Much work in relation to the design of fermion-to-qubit mappings is directed toward the reduction of the qubit and Pauli weight (the number of qubits that mapped fermionic operations involve) requirements on lattice models [22–27]. Some works avoid fermionic encoding of the wave function altogether in an effort to mitigate the associated costs [28,29] Other works have addressed the measurement cost in certain fermionic simulation tasks, which directly depends on the Pauli weight of the mapping when using informationally complete measurements and thus introduces a mapping with provable optimal Pauli weight [30]. The latter mapping is generated using regular ternary trees and has largely inspired the current work. However, the connection between ternary trees and mappings has been established earlier. In Ref. [31], it has been used to find representations of the Clifford algebras and spin groups and there has been discussion about how JW and BK can be generated from ternary trees as linear and binary subgraphs of them with the appropriate pairing of the resulting strings. More recently, a framework to design mappings based on BK has been put forward in Ref. [32] but it cannot achieve optimal Pauli-weight scaling.

In this paper, we consider the family of mappings generated by identifying Majorana operators with linearly and algebraically independent Pauli strings obeying equivalent anticommutation relations. In Sec. II, we give an overview of fermion-to-qubit mappings and present a classification in terms of the number of qubits that the nontrivial overlap (NTO) between strings involves. We then focus on one-qubit-wise anticommuting (1-NTO) encodings and introduce a ternary-tree-based framework to design mappings within this subclass. More precisely, we prove that any n-node connected ternary tree yields a valid mapping.

Importantly, this framework presents a number of desirable properties. On the one hand, it is clear and intuitive. It enables understanding and consequentially the control of many important properties of the mappings, such as their mode locality (i.e., on how many qubits fermionic mode occupation is stored, which has been suggested to impact the resilience of quantum simulations to noise [33]) and their Pauli weight, in terms of simple properties of the underlying ternary trees generating them. Part of the discussion in Sec. II is aimed at explaining these aspects in a pedagogical manner. On the other hand, it contains the aforementioned paradigmatic mappings, as well as the recently discovered optimal-weight fermion-to-qubit mapping [30], as specific instances. Hence, we can regard our framework as being able to *interpolate between* and *combine* well-known mappings, as well as generate completely novel ones that bear little resemblance with these. In addition, these two properties together provide, as a byproduct, a clean and transparent perspective on these widely used encodings and their properties.

We also show that with the right assignment between Majorana operators and Pauli strings, for which we provide a recipe, the tree-based-mapping design framework introduced here guarantees a crucial property of the sampled encodings: uncorrelated fermionic states, i.e., Fock basis states (including the vacuum state) are mapped to computational basis states in qubit space. This property, which may be easily overlooked, is of utmost importance, as it ensures that no additional quantum resources (more precisely, entanglement) are needed to prepare reference wave functions, such as the Hartree-Fock Slater determinant often used as a starting point in many quantum simulation algorithms.

In Sec. IV, we put our framework into use. We exploit its versatility to devise an algorithm, which we name the *Bonsai algorithm*, that takes a hardware connectivity layout as an input and returns a tailored mapping. The resulting encoding is designed to minimize the SWAP overhead required in the implementation of one- and two-electron excitation operations, which are the building blocks of many adaptive ansatz construction algorithms of the ADAPT-VQE family [34–36]. At the same time, the algorithm aims at minimizing the spread of fermionic occupancy over qubits. When applied to heavy-hexagon qubit-connectivity graphs, the standard layout in current IBM devices, the Bonsai algorithm returns a mapping with no SWAP overhead for single-excitation operations and a quadratically lower Pauli weight than JW. Moreover, the mapping also drastically reduces the circuit complexity of the worst-case scenario implementation of single- and double-excitation generated unitaries with respect to the latter mapping.

II. FERMION-TO-QUBIT MAPPINGS

This section is devoted to the discussion of features of general fermion-to-qubit mappings. After setting up the notation for fermionic systems (for detailed reviews, see Refs. [37–39]), the JW transformation, general fermion-to-qubit maps, and finally Majorana-string mappings are described.

A. Fermionic systems

Consider an N-mode fermionic system in second quantization described in terms of N creation and annihilation operators, $\{a_i^\dagger\}_{i=0,\ldots,N-1}$ and $\{a_i\}_{i=0,\ldots,N-1}$, which fulfill the usual canonical fermionic anticommutation relations

$$\{a_i, a_j\} = \{a_i^\dagger, a_j^\dagger\} = 0 \quad \text{and} \quad \{a_i^\dagger, a_j\} = \delta_{ij}\,\mathbb{1}. \tag{1}$$

The creation and annihilation operators act on $\mathcal{F}(\mathbb{C}^N)$, the Fock space belonging to an N-dimensional one-particle space. This is a 2^N-dimensional Hilbert space spanned by the fermionic vacuum $|\text{vac}_f\rangle$ and the vectors obtained by applying subsets of fermionic creation operators; this orthonormal basis, also called the Fock basis, can be denoted as

$$|n_0, n_1, \ldots\ n_{N-1}\rangle := (a_0^\dagger)^{n_0}(a_1^\dagger)^{n_1}\cdots\ (a_{N-1}^\dagger)^{n_{N-1}}\,|\text{vac}_f\rangle\,, \tag{2}$$

where $n_j \in \{0,1\}$ are the so-called occupation numbers of mode j and the notation $(a_j^\dagger)^0 = \mathbb{1}$ is used. The fermion-number operator for mode j is given as $\hat{n}_j = a_j^\dagger a_j$. It is easy to show that the Fock basis states are eigenstates of these local fermion-number operators with eigenvalue given by the occupation numbers, e.g., $\hat{n}_0\,|n_0, n_1, \ldots n_{N-1}\rangle = n_0\,|n_0, n_1, \ldots n_{N-1}\rangle$.

Besides the fermionic creation and annihilation operators, another useful set of generators for the fermion observables are the $2N$ Majorana operators $\{m_k\}_{k=0,\ldots,2N-1}$, defined as

$$m_{2j} = a_j^\dagger + a_j \quad \text{and} \quad m_{2j+1} = i\left(a_j^\dagger - a_j\right), \tag{3}$$

which are unitary, self-adjoint, and obey the Majorana anticommutation relations

$$\{m_i, m_j\} = 2\delta_{ij}\,\mathbb{1}. \tag{4}$$

Any fermionic observable can be uniquely expressed as a linear combination of Majorana monomials $m_{x_0}\cdots\ m_{x_j}$. In particular, the number operator for mode j is $\hat{n}_j = \frac{1}{2}(\mathbb{1} + i m_{2j} m_{2j+1})$.

B. Jordan-Wigner transformation and general fermion-to-qubit mappings

The fermionic Fock space $\mathcal{F}(\mathbb{C}^N)$ and the Hilbert space of N qubits $(\mathbb{C}^2)^{\otimes N}$ are both 2^N-dimensional Hilbert spaces; thus one can map one into the other unitarily. A very natural unitary mapping is to map the Fock basis states of $\mathcal{F}(\mathbb{C}^N)$ to the computational basis states of the qubits such that the occupation number of the j th fermionic mode matches with the state of the j th qubit [19]:

$$\mathcal{F}(\mathbb{C}^N) \ni |n_0, n_1, \ldots, n_{N-1}\rangle \mapsto \bigotimes_{i=0}^{N-1} |n_i\rangle \in (\mathbb{C}^2)^{\otimes N}. \tag{5}$$

On the operator level, this correspondence induces a linear mapping between the corresponding observable algebras given by

$$m_{2j} \mapsto X_j \prod_{k=0}^{j-1} Z_k, \tag{6}$$

$$m_{2j+1} \mapsto Y_j \prod_{k=0}^{j-1} Z_k, \tag{7}$$

for $j = 0, 1, \ldots N-1$. Here and in the rest of the paper, we use the notation P_j with $P \in \{X, Y, Z\}$ for an operator that acts as the Pauli operator P on the j th qubit and as identity on the other qubits.

In general, a unitary mapping between the fermionic and qubit Hilbert spaces induces a linear mapping on the corresponding observable algebras such that

$$\begin{aligned} & m_k \mapsto R_k, \quad k = 0, \ldots 2N-1, \\ & \text{(i) the } R_k \text{ are algebraically independent,} \\ & \text{(ii) } \{R_k, R_\ell\} = 2\delta_{k\ell}\,\mathbb{1}. \end{aligned} \tag{8}$$

Conversely, any linear mapping between the fermionic and qubit observable algebras satisfying the properties (i) and (ii) in Eq. (8) defines uniquely (up to a global phase factor) a unitary mapping between $\mathcal{F}(\mathbb{C}^N)$ and the Hilbert space of N qubits $(\mathbb{C}^2)^{\otimes N}$. This unitary mapping between the two Hilbert spaces can be constructed as follows. Since the fermionic vacuum state $|\text{vac}_f\rangle$ is the unique vector (up to a scalar factor) satisfying the relations $a_j\,|\text{vac}_f\rangle = \frac{1}{2}(m_{2j} + i m_{2j+1})\,|\text{vac}_f\rangle = 0$ for all $j = 0, \ldots, N-1$, the vacuum state is mapped to the state $|\psi\rangle$ which satisfies that $\frac{1}{2}(R_{2j} + i R_{2j+1})\,|\psi\rangle = 0$ for all $j = 0, \ldots, N-1$ (note that such a $|\psi\rangle$ is unique up to a phase factor). Any other Fock basis vector $a_{j_0}^\dagger a_{j_1}^\dagger \ldots a_{j_\ell}^\dagger\,|\text{vac}_f\rangle$ is mapped to $\frac{1}{2}(R_{2j_0} - i R_{2j_0+1})\frac{1}{2}(R_{2j_1} - i R_{2j_1+1})\cdots\frac{1}{2}(R_{2j_\ell} - i R_{2j_\ell+1})\,|\psi\rangle$.

C. Majorana-string mappings

When it comes to mapping fermionic systems to qubit systems, the Pauli basis suggests a path: finding a suitable set $\mathcal{S}$ of $2N$ Pauli strings (i.e., products of Pauli operators) S_k $(k = 0 \ldots 2N-1)$ fulfilling the anticommutation $\{S_i, S_j\} = 2\delta_{ij}\,\mathbb{1}$. For this approach to result in a proper fermion-to-qubit mapping, the Pauli strings in $\mathcal{S}$ must also be linearly and algebraically independent. Linear independence is trivially satisfied if all the strings differ but algebraic independence is more subtle. This means that it must not be possible to find two different subsets $A \subseteq \mathcal{S}$ and $B \subseteq \mathcal{S}$, $A \neq B$, such that $\prod_{S_i \in A} S_i \propto \prod_{S_j \in B} S_j$, given that the corresponding products of Majorana operators in fermion space result in distinct operators. Throughout this work, we use the term *Majorana strings* to refer to the Pauli strings within a set $\mathcal{S}$ satisfying these conditions.

Summarizing the above, in this paper we consider so-called *Majorana-string fermion-to-qubit mappings*, which are linear mappings between the fermionic and qubit system observable algebras that satisfy the following criteria:

(a) *Criterion (A)*. Each Majorana operator is mapped to a Pauli string, $m_j \to S_j \in \mathcal{S}$ for $j = 0, \ldots, 2N-1$.
(b) *Criterion (B)*. The above Pauli strings satisfy $\{S_i, S_j\} = 2\delta_{ij}\mathbb{1}$.
(c) *Criterion (C)*. For any unequal subsets $A \subseteq \mathcal{S}$ and $B \subseteq \mathcal{S}, A \neq B$, $\prod_{S_i \in A} S_i \propto \prod_{S_j \in B} S_j$ is not fulfilled.

Furthermore, one often considers Majorana-string fermion-to-qubit mappings that satisfy an additional criterion:

(a) *Criterion (D)*. Vacuum preservation: the fermionic vacuum is mapped to the all-zero computational basis state, i.e., $|\text{vac}_f\rangle \mapsto |0\rangle^{\otimes N}$.

Mappings that satisfy criterion (D) in addition to criteria (A)–(C) are called *vacuum preserving* Majorana-string fermion-to-qubit mappings.

Consider a Majorana string S_j that is (up to a phase factor) a product of (nonidentity) Pauli operators over a subset of sites $A_j \subset \{0, \ldots, N-1\}$. We call A_j the *support of* S_j. Let A_j and A_k be the supports of S_j and S_k, respectively. We call $A_j \cap A_k$ the *overlapping sites* of S_j and S_k. The subset of $N_{j,k} \subseteq A_j \cap A_k$, where the local Paulis corresponding to the Majorana strings S_j and S_k are different, is called the *nontrivial overlapping sites of* S_j *and* S_k. As any two Majorana strings anticommute, the number of nontrivial overlapping sites of any pair of Majorana strings must be odd. Given a Majorana-string fermion-to-qubit mapping, let k be the maximum of this odd number considering all the pairs of Majorana strings. We call such a mapping a k-non-trivial-overlap (k-NTO) Majorana-string fermion-to-qubit mapping. Most of the known fermion-to-qubit mappings (e.g., JW, BK, and parity) are 1-NTO, but non-1-NTO mappings also exist (see Appendix A).

III. MAPPINGS ORIGINATING FROM GENERAL TERNARY TREES

In this subsection, we explore the correspondence between a certain class of graphs, called ternary trees (TT), and vacuum-preserving fermion-to-qubit mappings. As mentioned previously, the connection between the two has been established before [30,31]. In Ref. [30], the minimum-depth TT is used to find a fermion-to-qubit mapping with optimal Pauli weight. Inspired by that work, we now extend the TT formalism and show that *any* TT can result in a valid mapping, the properties of which can be directly connected to the graph-theoretical properties of the tree. This connection is explored more carefully in Sec. III A. Moreover, the mapping-generating method introduced here guarantees that, for any sampled mapping, the fermionic vacuum is mapped to the all-zeros qubit state and Fock basis states are mapped to computational basis states; we refer to this property as *product preservation*.

A. Ternary trees

It is useful for this section and the next one to start by reviewing some graph-theoretic concepts. A graph is a pair $\mathcal{G} = (V, E)$, where V is a set of vertices or nodes and $E \subseteq \{(x,y) \mid (x,y) \in V^2, x \neq y\}$ are the edges, or links, which are sets of paired vertices. We consider undirected graphs, meaning that $(x,y) \in E \Rightarrow (y,x) \in E$. A length-$\ell$ path $\mathbf{p}$ in a graph is a length-ℓ ordered sequence of vertices $\mathbf{p} = \{p_0, p_1, \ldots, p_{\ell-1}\}$ such that any pair of consecutive vertices in the sequence are connected in the graph, i.e., for every $l < \ell$, $(p_{l-1}, p_l) \in E$. We call a graph $\mathcal{G}$ connected if, for any pair of vertices u, v, there exists at least one path $\mathbf{p}$ with u and v as endpoints. In any such graph $\mathcal{G}$, the path structure induces a well-defined metric distance $d(u,v)$ between all pairs of vertices, defined as the length ℓ of the shortest path (or paths, as they may not be unique) with the two vertices as endpoints. It can be easily shown that the set of distances $d(u,v)$ define a proper metric space, given that they are positive, symmetric, zero if and only if $u = v$, and they fulfill the triangle inequality, $d(u,v) \leq d(u,w) + d(w,v)$, $\forall u, v, w \in V$.

The path structure also allows us to define a special kind of graph, the tree. A tree $\mathcal{T}$ is a graph that contains no loops, i.e., for which there are no paths $\mathbf{p}$ containing any node more than once. It is a fact that any connected N-node tree contains exactly $N-1$ edges and that any connected undirected graph with $N-1$ edges is a tree. It is also useful to define the degree of a node u as the number of edges reaching u,

$$\Delta(u) = |\{v \in V \mid (u,v) \in E\}|. \tag{9}$$

With these definitions at hand, we can introduce the TT. Essentially, a TT is a tree in which the branching rate is at most three, i.e., each node has at most three *descendants*. To explain this concept more precisely, let us describe a process in which a TT is built by adding nodes sequentially. First, we start with a single node, the *root* r. Next, we add $k_r \leq 3$ nodes to the graph and connect each of them to the root r. The root now has degree $\Delta(r) = k_r$. Next,

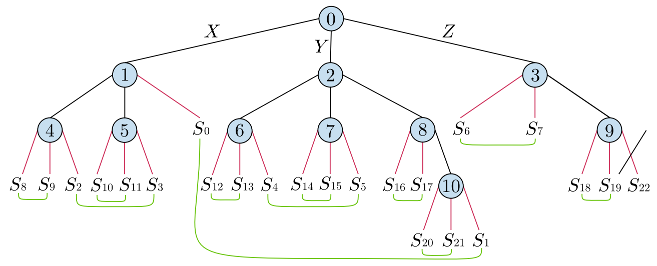

X
Y
Z
0
1
2
3
4
5
S_0
6
7
8
S_6
S_7
9
S_8 S_9 S_2 S_{10} S_{11} S_3
S_{12} S_{13} S_4 S_{14} S_{15} S_5
S_{16} S_{17}
10
S_{18} S_{19} S_{22}
S_{20} S_{21} S_1

FIG. 1. An example of a mapping derived from a ternary tree. The black lines represent the edges, which always connect two qubits, while the legs are depicted in red. Throughout this paper, we always represent the Pauli labeling on the links (both edges and legs) by their position: the leftmost link below a node is labeled with an X, while the rightmost link is labeled with a Z. Every leg in the tree can be associated with a Pauli string by following the path from the root node (0, in this case) to the leg. Every time a link with label P stemming downward from a qubit u is crossed, the operator P acting on qubit u is added to the string. The resulting string acts trivially on all qubits not visited along the path. For example, with the tree in the figure, the leg labeled with S_0 generates the string $S_0 = X_0Z_1$, while $S_1 = Y_0Z_2Z_8Z_{10}$. The green lines depict the pairing between Majorana strings that guarantees product preservation. Note that following the upward paths starting from the two legs of any such pair, the two paths meet at a qubit u once they both cross their first link not labeled with a Z. In the previous example with strings S_0 and S_1, this corresponds to the root node 0.

for each of these new nodes (*descendants* of r) u, we add $0 \leq k_u \leq 3$ nodes, which connect to u. The process can be iterated until the graph contains N nodes. Note that since each time we add a node, we add one link along with it, the total number of edges will be $N-1$ (this observation also shows why the resulting graph is a tree: it is not possible to close any loops in the graph by connecting one new node to a single existing one only). Any node without descendants is denoted *leaf* throughout this work.

B. From ternary trees to fermion-to-qubit mappings

The starting point of this work is the observation that, following a simple procedure introduced in Refs. [30,31] for specific TTs, any N-node TT can be used to generate valid fermion-to-qubit mappings. We now explain this procedure and show its generality. The scheme is illustrated in Fig. 1, where we present an 11-qubit example.

Suppose that we have an N-qubit system that we want to use to simulate an N-mode fermionic system. Without loss of generality, in what follows we label each qubit with an integer number $u = 0, \ldots, N-1$. To generate a fermion-to-qubit mapping, we first generate an N-node TT by, e.g., following the iterative procedure introduced previously. Next, we assign a qubit label u to each node. Different assignments will lead to different mappings (a degree of freedom that can be exploited) but any such labeling is admissible. In the example (Fig. 1), this corresponds to the tree with blue nodes and black edges.

The next step in the process is to add $3 - k_u$ *legs*, i.e., edges without a node at the other end of the link, to each node u, where k_u is the number of descendants of u. Note that $k_r = \Delta(r)$ for the root r, while $k_u = \Delta(u) - 1$ for all other nodes as $\Delta(r) \leq 3$ and $\Delta(u) \leq 4$. These are the red links in Fig. 1. Importantly, doing this will result in $2N+1$ legs for any tree. To see this, let us denote the number of legs by L. Every node except the root is now reached by four *links* (a term that we use in this work to refer to both edges in the original tree and legs), while the root is reached by three. Thus, if we sum all the new degrees (including legs) for all the nodes, we obtain $F = 4N - 1$. In this sum, we count each of the edges in the original tree twice (once per each node at its endpoints), while the legs are counted only once. Hence, we have $F = |E| + L$, with $|E| = 2(N-1)$ (since each edge is contained twice in E), so $L = 2N + 1$.

Once the legs have been added and every node has exactly three descending links (either edges or legs), we distribute the labels X, Y, and Z among said three links of each node. In order to ease the graphical depictions of the trees in this work, the leftmost link implicitly carries the label X, the central Y, and the rightmost one, Z, as in Ref. [30].

With this labeling of nodes, edges, and legs, the tree can be used to generate Pauli strings in the following manner. For every leg, there exists a unique path leading from it to the root r. The path only includes one leg, the starting one, and it may cross some edges as well. With every link in the path, we can associate a unique Pauli matrix, P_u, where u is the parent node reaching the link, and P is the label X, Y, or Z, corresponding to the link. The Pauli string is therefore formed by taking the tensor product of these Pauli operators, along with identity on nodes not along the path. Since every path results in a different Pauli string, this procedure generates $2N+1$ strings for N qubits.

Importantly, all these Pauli strings anticommute with one another. This can be seen by considering two Pauli strings S_i and S_j stemming from two different legs in the tree. The two paths corresponding to the legs must meet, i.e., when traversing them upward toward the root, they must have a first node in common (which may be the root itself). If the first common ancestor to both legs is not the root, both paths from that node upward are equal and hence both S_i and S_j contain the same Pauli operators for those qubits. The first common ancestor, on the other hand, must be reached following two different descendants of said node, so the Pauli matrices for that qubit in the strings are distinct and both different from identity (what we refer to as *nontrivial overlap*). If the legs are not directly connected to their common ancestor, their paths include other nodes lying below the latter in the tree. However, note that, by definition of a first common ancestor, those nodes cannot be present in both paths and hence one of the strings must act trivially on each such qubit. In short, S_i and S_j have a single-qubit nontrivial overlap, i.e., for every qubit different from the aforementioned first common ancestor, the corresponding Pauli matrices are either both equal or at least one of them is equal to the identity.

Since all the $2N+1$ resulting Pauli strings are different, they are obviously linearly independent but they are not algebraically independent. In particular, any two disjoint subsets of strings A and B, $A \cap B = \emptyset$ such that $A \cup B$ is the whole set of strings fulfill $\prod_{S_i \in A} S_i \propto \prod_{S_j \in B} S_j$. However, as we prove in Appendix B, any subset missing at least one Pauli string is algebraically independent. Therefore, by dropping any of the strings, the remaining $2N$ ones can be readily identified with the $2N$ Majorana operators associated with the N-mode fermionic system, thus defining a valid fermion-to-qubit mapping.

C. Majorana-string pairing for product preservation

The previous discussion illustrates how any N-node TT can be used to obtain N-mode mappings. While any association between the generated Pauli strings and Majorana operators results in a legitimate mapping, not all are equally useful in practice. Many applications of fermion-to-qubit mappings require that at least some reference state (e.g., the vacuum) be known in qubit space. In near-term quantum computing, for instance, it is also desirable that Fock basis states be mapped to computational basis states. We now provide a simple recipe to enforce this product-preservation feature of the map, including guaranteeing that the fermionic vacuum is mapped to the state $|0\rangle^{\otimes N}$.

First, let us introduce the concept of *pairing*. According to Eq. (3), there are two Majorana operators, m_{2j} and m_{2j+1}, associated with every fermionic mode j. Therefore, after identifying Majorana strings with Majorana operators, every creation and annihilation operator $a_j^{(\dagger)}$ will be associated with two Pauli strings. The key to product preservation lies in how the Pauli strings are paired into fermionic modes.

Let the set of Majorana strings be the set obtained by removing the Pauli string corresponding to the path that only involves links with label Z and consider the following pairing algorithm. For every node u in the TT, follow its downward link labeled with X. If the link is not a leg, keep traveling downward, always taking the Z links, until a leg is reached. Denote the leg by $s_x^{(u)}$. The same procedure, starting from the link with label Y, will lead to a different final leg $s_y^{(u)}$. The two Pauli strings $S_{s_x^{(u)}}$ and $S_{s_y^{(u)}}$ should then be paired together into some fermionic mode j, i.e., one of them should be identified with m_{2j} and the other one with m_{2j+1}. These pairings are illustrated with green lines in Fig. 1. The simplest identification corresponds to the mapping

$$m_{2j} \leftrightarrow S_{s_x^{(u)}} \quad \text{and} \quad m_{2j+1} \leftrightarrow S_{s_y^{(u)}} \tag{10}$$

but it is worth mentioning that it is also possible to ensure that the mapped creation and annihilation operators are real in qubit space by associating with m_{2j} the Pauli string that contains an even number of Y operators and with m_{2j+1} the one containing an odd number of them. However, for the sake of simplicity, we only consider the first type of identification explicitly throughout this work. Importantly, note that the identification between modes j and qubits u in Eq. (10) implicitly establishes a bijection between the two sets.

By inverting Eq. (3), we see that the fermionic creation and annihilation operators are mapped into qubit space according to

$$\begin{aligned} a_j^\dagger &= \frac{1}{2}(m_{2j} - i m_{2j+1}) \leftrightarrow \frac{1}{2}\left(S_{s_x^{(u)}} - i S_{s_y^{(u)}}\right), \\ a_j &= \frac{1}{2}(m_{2j} + i m_{2j+1}) \leftrightarrow \frac{1}{2}\left(S_{s_x^{(u)}} + i S_{s_y^{(u)}}\right). \end{aligned} \tag{11}$$

From these expressions, it is possible to show that the fermionic vacuum is represented by the qubit state $|0\rangle^{\otimes N}$. To do so, consider the two Pauli strings $S'_{s_x^{(u)}}$ and $S'_{s_y^{(u)}}$ obtained by substituting the Z operators on the qubits that lie below u on the tree (if any) with identity. Clearly,

```
1  Choose a bijection $f$ between modes $j$ and qubits $u$,
     $j = f(u)$.
2  Let $V$ be the set of qubits in the tree.
3  **for** $u \in V$ **do**
4      Define $s$ as the $X$-labelled downward link
         stemming from $u$.
5      **while** $s$ *not a leg* **do**
6          Define $v$ as the qubit reached following $s$
             downwards.
7          Define $s$ as the $Z$-labelled downward link
             stemming from $v$.
8      Set $s \to s_x^{(u)}$.
9      Define $s$ as the $Y$-labelled downward link
         stemming from $u$.
10     **while** $s$ *not a leg* **do**
11         Define $v$ as the qubit reached following $s$
             downwards.
12         Define $s$ as the $Z$-labelled downward link
             stemming from $v$.
13     Set $s \to s_y^{(u)}$.
14 Remove the unpaired right-most $Z$-leg from the tree.
15 Create mode operators $a_j$ and $a_j^\dagger$ using Eq. (11) with
     $j = f(u)$.
16 Return mapped mode operators.
```

Algorithm 1. Pairing scheme.

$S'_{s_x^{(u)}}$ and $S'_{s_y^{(u)}}$ are equal except for the Pauli operator acting on qubit u, which implies that $S'_{s_x^{(u)}} + iS'_{s_y^{(u)}}$ contains Pauli operators on all qubits except for u, where it contains a $2P^- = X + iY$ operator. Moreover, since $S'_{s_x^{(u)}} |0\rangle^{\otimes N} = S_{s_x^{(u)}} |0\rangle^{\otimes N}$, and similarly for $S'_{s_y^{(u)}}$,

$$\left(S_{s_x^{(u)}} + iS_{s_y^{(u)}}\right) |0\rangle^{\otimes N} = 0. \tag{12}$$

A similar argument for the creation operator can be used to show that $(S_{s_x^{(u)}} - iS_{s_y^{(u)}}) |0\rangle^{\otimes N}$ is proportional to a computational basis state: since $(S_{s_x^{(u)}} - iS_{s_y^{(u)}}) |0\rangle^{\otimes N} = (S'_{s_x^{(u)}} - iS'_{s_y^{(u)}}) |0\rangle^{\otimes N}$, its action is to flip the state of qubit u from $|0\rangle$ to $|1\rangle$, as well as possibly to flip any other qubits above u in the tree for which the traversed links are labeled with an X or a Y. In Appendix C, we extend this argument to prove that any Fock basis state is mapped into a computational basis state in qubit space.

After application of this pairing scheme, the jth mode operators take the form

$$a_j \mapsto \frac{1}{2}\left(X_u \prod_{k\in\mathcal{Z}_u^x} Z_k + iY_u \prod_{k\in\mathcal{Z}_u^y} Z_k\right) G_u,$$
$$a_j^\dagger \mapsto \frac{1}{2}\left(X_u \prod_{k\in\mathcal{Z}_u^x} Z_k - iY_u \prod_{k\in\mathcal{Z}_u^y} Z_k\right) G_u, \tag{13}$$

where $\mathcal{Z}_u^x$ and $\mathcal{Z}_u^y$ are sets of qubits that $S_{s_x^{(u)}}$ and $S_{s_y^{(u)}}$ act nontrivially on below qubit u in the tree and G_u is a common Pauli string that we can factor out. The sets $\mathcal{Z}_{x/y}^u$ may be empty, in which case the operator reduces to a similar form to the JW mapping, $a_i^{(\dagger)} \to P_u^{\pm} G_u$, where G_u enforces fermionic antisymmetry with other qubits analogous to the Z chain. This equation is graphically understood as G_u being the common path of $S_{x/y}^{(u)}$ from the root to qubit u and sets $\mathcal{Z}_u^{x/y}$ are qubits along the Z paths bifurcating from the X/Y legs of qubit u.

D. Properties of the mappings and the effect of labeling

In the construction of a TT mapping, there are two main degrees of freedom: the tree and the labeling. In this subsection, we briefly discuss how the properties of these elements impact the resulting mappings.

An important feature of a fermion-to-qubit mapping is its *Pauli weight*. The Pauli weight of a Pauli string is defined as the number of qubits on which the string acts nontrivially (i.e., with a Pauli operator different from identity). In the case of a mapping, this refers to the Pauli weight of the strings of the mapped fermionic operators. In the case of TT mappings, this can be easily analyzed by considering the Majorana strings. More concretely, note that the Pauli weight of a Majorana string generated from a TT is precisely the length of the corresponding path from root to leg. Thus, there is a straightforward connection between the Pauli weight of the mapping and the average shortest path length to the root in the tree. More generally, the topology of the tree impacts the sets of qubits involved when applying creation or annihilation operators but not how (i.e., by which Pauli operators).

The labeling, instead, has a more subtle impact on the resulting map. While it cannot affect the average Pauli weight of the resulting Majorana strings, it impacts how the occupation of the fermionic modes is *delocalized* over qubits. More precisely, consider the number operator $n_j = a_j^\dagger a_j$ for a fermionic mode j in qubit space. Using Eq. (11), we see that

$$n_j = \frac{1}{2}\left(\mathbb{1} + im_{2j}m_{2j+1}\right) \leftrightarrow \frac{1}{2}\left(\mathbb{1} + iS_{s_x^{(u)}}S_{s_y^{(u)}}\right). \tag{14}$$

Since the two Pauli strings $S_{s_x^{(u)}}$ and $S_{s_y^{(u)}}$ are equal on all qubits above u in the tree, the product on the right-hand side results in the identity operator for those. On u, on the other hand, the product yields $XY = iZ$. For the qubits below u in the strings, however, one string acts with a Z operator while the other one acts with identity. Therefore, if we define the set $\mathcal{Z}_u$ of qubits below u on which $S_{s_x^{(u)}}$ and $S_{s_y^{(u)}}$ act nontrivially and include u itself too, we have

$$n_j = a_j^\dagger a_j \leftrightarrow \frac{1}{2}\left(\mathbb{1} - \prod_{q\in\mathcal{Z}_u} Z_q\right), \tag{15}$$

i.e., the occupation of a mode j is encoded in the parity of the state of the qubits in $\mathcal{Z}_u$. Now, given a qubit u with edges directly below itself, the choice of label for each of these edges will generally affect the structure of sets $\{\mathcal{Z}_u\}_u$ in the resulting mapping and, with it, its delocalization structure.

To analyze this in an illustrative manner, let us introduce a convenient definition of mode-specific delocalization D_u in terms of the qubit u the mode is associated with in a mapping,

$$D_u = |\mathcal{Z}_u| - 1. \tag{16}$$

Now, consider a generic example in which a node i, which is not the root, has two descendants j and k (see Fig. 2). Nodes j and k themselves may have descendants. Suppose that we add legs to all qubits and we label all links in the resulting tree except for the three links stemming downward from i. The question is then how those three links should be labeled. Of course, there are three possibilities

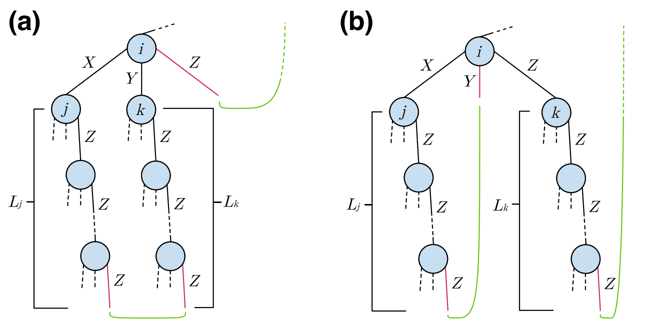


FIG. 2. The effect of edge labeling below a node i. Its two descendants, j and k, both have descendants and the all-Z paths underneath them have length L_j and L_k, respectively. We assume that all links in the tree except for the ones right below i have the same label in both (a) and (b). In (a), both paths contribute to occupation delocalization of the mode associated with i, while in (b), only the path below j does. In (b), the path of length L_k may contribute to the delocalization of some other mode unless the path from i to the root only crosses Z-labeled edges.

(three labels to be distributed among three links) but since swapping $X \leftrightarrow Y$ labels between two links stemming from the same node has a simple impact on the mapping (switching the roles of $S_{s_x^{(u)}}$ and $S_{s_y^{(u)}}$), the only two situations to be discussed are whether the Z label should be assigned to an edge or the leg.

In the left figure, we depict the case in which the leg is assigned the Z label and the two edges X and Y. In this case, $\mathcal{Z}_i$ contains i and all the nodes in the Z strings lying below j and k, which in the example have length L_j and L_k, respectively. Therefore, the occupation of the mode associated with qubit i is delocalized among $D_i = L_j + L_k$ qubits. In the opposite case, in which the Z label is assigned to one of the edges, on the other hand, one of the two Z strings no longer contributes to the delocalization of node i. In the illustration, we have $D_i = L_j$, i.e., the occupation of the mode is less spread in the second case.

However, note one important fact: assume that we follow the path from i upward toward the root r until we reach an edge labeled with an X or a Y and let us call u the node reached by traversing that edge. In the latter case (right-hand side of the figure), the Z string along qubit k is now part of one of the Z strings directly below qubit u. In other words, while the delocalization of the mode associated with i decreases by an amount L_k, the delocalization of the mode associated with u increases by the same amount. Therefore, under this assumption, the labeling cannot affect the average delocalization of the mapping but only its distribution among the qubits. Crucially, if the path from i to r only crosses Z-labeled edges, this is no longer true and the second labeling does not increase the delocalization of any other mode.

From the above discussion, we can draw a very useful overall conclusion regarding the delocalization structure of a mapping: the average delocalization among nodes is given by

$$\langle D_u \rangle = 1 - \frac{h_Z}{N}, \tag{17}$$

where h_Z is simply the number of nodes that can reach the root node r by traversing only Z-labeled edges, including the root itself. This can be seen as follows. If a node i cannot be traced back to the root following Z-labeled edges, the path toward r must cross an X- or Y-labeled edge attached to some node u and thus i contributes one unit to the delocalization of node u. Therefore, the sum of all delocalizations must be equal to the number of nodes not in the Z-labeled path, i.e., $\sum_u D_u = N - h_Z$. This observation implies that in order to minimize the delocalization of the modes, which may be a desirable property of a fermion-to-qubit mapping [33], we must maximize the number of nodes along the Z-only path. Interestingly, since $h_Z \in \{1, \dots, N\}$, the average delocalization is bounded $\langle D_u \rangle \in [0, 1 - 1/N]$, i.e., on average, the occupation of the fermionic modes is stored in less than two qubits, $\langle |\mathcal{Z}_u| \rangle \in [1, 2 - 1/N]$.

E. The ternary trees of paradigmatic mappings

It is illustrative to analyze paradigmatic fermion-to-qubit mappings in this context. In particular, the JW, BK, and parity mappings and obviously the optimal mapping from Ref. [30] (JKMN) are all 1-NTO and can be generated from TT. In Fig. 3, we depict their corresponding trees. By analyzing their graph topologies,and following the insights from the previous discussion, we can easily understand their main properties.

Both JW and parity are given by linear graphs. Since these are depth-N trees, the Pauli weight of the resulting Majorana strings is $O(N)$. However, their occupation delocalization is different. JW is an extreme case, given that all nodes are in the Z-labeled path and thus has average delocalization $\langle D_u \rangle = 0$; this is the only possible TT mapping with no delocalization. In the case of parity, the occupation is maximally delocalized, with each occupation encoded between two consecutive nodes in the chain, except for the last qubit, in which it is fully localized.

BK and JKMN, on the other hand, are generated by trees with constant branching rates 2 and 3, respectively. Therefore, their depths, and hence the resulting Pauli weights, scale as $O(\log N)$, with JKMN having a smaller depth owing to its higher branching rate (in fact, the authors prove the optimality of the Pauli weight of their mapping in Ref. [30]). In both cases, however, the price to pay is the delocalization of the occupation. More precisely, note that the higher up the tree the common ancestor of a given pairing is, the more Z links are involved in the resulting Majorana strings. Thus, only the lowest-lying nodes lead to completely localized modes.

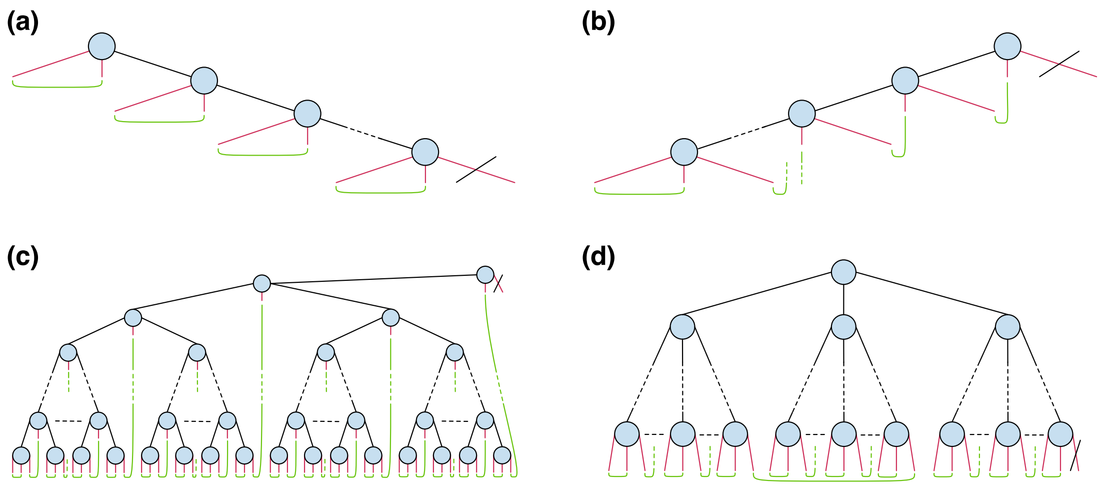


FIG. 3. Ternary trees generating four paradigmatic mappings. (a) JW, (b) parity, (c) BK, and (d) JKMN mapping. From the analysis presented throughout Sec. III D, we can see that both JW and parity have linear-scaling depth, and hence Pauli weight, while BK and JKMN have $O(\log_2 N)$ and $O(\log_3 N)$ scaling, respectively. Regarding the delocalization, it can be seen that JW minimizes it, while BK and parity have no nodes other than the root in the all-Z branch. JKMN has a nonminimal all-Z branch including $O(\log_3 N)$ nodes and hence is nearly maximally delocalized for large N.

IV. GROWING HARDWARE-EFFICIENT MAPPINGS WITH THE BONSAI ALGORITHM

Ternary tree mappings can be used as a tool for the design of custom fermion-to-qubit mappings. The framework introduced in Sec. III is general and can help find mappings with specific desired properties by tailoring the trees according to different cost functions. In what follows, we introduce an algorithm to produce mappings aimed at reducing the complexity of fermionic simulations on quantum computers by minimizing the impact of limited qubit connectivity in the quantum processor. We start this section by briefly introducing the problem and we then present the Bonsai algorithm along with an illustrative and important use case: heavy-hexagon qubit lattices, the topology of choice for current IBM quantum computers.

A. Fermionic simulation under limited-connectivity constraints

The simulation of fermionic many-body systems is one of the most promising applications of quantum computing, both in the near term and in the fault-tolerant era. Many of the existing algorithms work in second quantization and thus typically require mapping the fermionic operators into qubit space. Fermionic operations are then mapped to unitary gates among the qubits in the device. However, many platforms (such as superconducting qubits) have limited connectivity, meaning that many pairs of qubits in the processor cannot physically interact directly. Thus, when a quantum gate involves qubits that are not physically connected, SWAP gates are iteratively applied so that the states of distant qubits are transported to neighboring ones and the gate is then applied. While this is always possible in theory, in practice, the additional SWAP gates increase the circuit complexity, which results in longer run times and, consequently, an increased detrimental effect of noise.

In order to illustrate how limited connectivity impacts the circuit complexity, let us consider a minimal example with four fermionic modes simulated with four qubits on two different platforms, one with linear connectivity (the physical-connectivity graph being a one-dimensional chain) and a second one with starlike connectivity (three of the four qubits connected to the fourth and no other connections). Both are depicted in Fig. 4. We now map four fermionic modes to these qubits using JW and TT mapping. For the latter, we consider the specific situation in which the ternary tree is congruent with the connectivity graph of the qubits: qubit 0 is the root and the other three qubits are its descendants.

We now examine the simulation of the even-even Majorana terms, $m_{2i}m_{2j}$ for $i \neq j$, arising from single-excitation terms, $a_i^\dagger a_j$. The resulting six Pauli strings are tabulated in the figure. In many applications, these terms must be exponentiated and implemented as rotations, i.e., $\exp -i\theta m_{2i}m_{2j}$. This requires entangling gates between the qubits not acted upon by an identity in the corresponding Pauli string. As explained above, if two such qubits are not neighbors in the physical-connectivity graph of the device, SWAPs must be applied. Figure 4 highlights the qubits involved in implementing the rotations. The SWAP overhead is indicated by thin lines skipping over the qubits.

In Fig. 4, we observe that with TT and linear connectivity, three operators, m_0m_6, m_2m_6, and m_4m_6, involve all four qubits even though the actual Pauli strings only act on three qubits each. The JW mapping, on the other hand, is more congruent with the underlying connectivity.

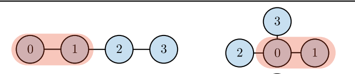
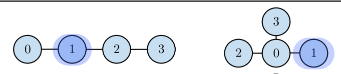
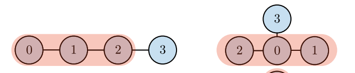
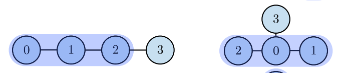
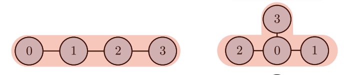
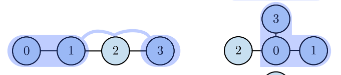
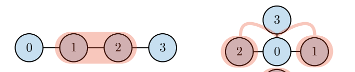
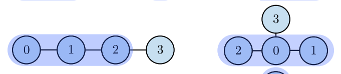
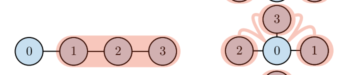
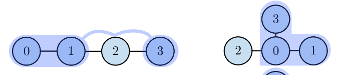
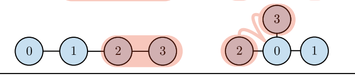
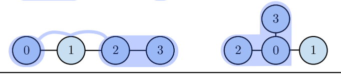
$m_{2i}m_{2j}$	$S_{2i}S_{2j}$	JW	$S_{2i}S_{2j}$	Ternary tree
m_0m_2	*YXII*		*IYII*	
m_0m_4	*YZXI*		*ZZXI*	
m_0m_6	*YZZX*		*YZIX*	
m_2m_4	*IYXI*		*ZXXI*	
m_2m_6	*IYZX*		*YXIX*	
m_4m_6	*IIYX*		*XIXX*	

FIG. 4. Qubit operators resulting from the product of two even Majorana operators, $m_{2i}m_{2j}$, implemented on a linear and a clustered topology. The operators are mapped to qubit space using JW and a four-qubit TT mapping in which the root has degree 3. Each product of Majorana operators yields a Pauli string, which is written explicitly in the leftmost column of each mapping. The red (blue) highlight corresponds to qubits on which the Pauli strings act nontrivially. A highlighted line skipping over qubits denotes ones that are not present in the strings but are involved in the cascade of SWAP gates.

In the large-N limit, the regular TT mapping presents an advantage in terms of Pauli weight with respect to JW (the former scales as $O(\log_3 N)$, while the latter as $O(N)$) so, in principle, each such rotation would involve many fewer qubits. However, the SWAP overhead with limited connectivity reduces the Pauli-weight advantage for the regular TT (and similarly for BK) and a number of controlled-NOT (CNOT) gates equivalent to JW may typically be required. In the case of star-graph connectivity, on the other hand, the TT mapping never requires SWAPs, as opposed to JW, and moreover, no operation involves more than three qubits.

This simple example illustrates why limited connectivity can be an issue for the implementation of fermionic operations and also that the right choice of mapping—in particular, one that is congruent with the underlying connectivity—can help mitigate the overhead.

B. The Bonsai algorithm

In this section, we introduce an algorithm to generate custom fermion-to-qubit mappings tailored to device-specific connectivity graphs. More precisely, the problem is, given a quantum processor, to find a mapping such that: (1) it is product preserving, (2) the resulting Pauli weight is low, and (3) the mode occupancy is local in qubit space. The first condition is satisfied by appropriate pairing as described in Sec. III C. The second and third points are suitably satisfied by finding a ternary tree that is a subgraph of the physical-connectivity graph (or close to one) and then exploiting the labeling freedom to define how the mode occupancy is distributed over qubits in a rational manner. In the following, we present this heuristic strategy in detail and illustrate it with an important application: designing mappings for heavy-hexagon quantum computers. The steps of the algorithm are summarized in Algorithm 2, while the specific subroutines are described in detail in Appendix D.

1. Finding the ternary tree

The input of the Bonsai algorithm is a physical-connectivity graph $\mathcal{P} = (V_\mathcal{P}, E_\mathcal{P})$, in which the nodes are the qubits in the processor and the edges represent the pairs of qubits onto which it is possible to physically apply entangling gates. In Fig. 5(a), we depict the physical-connectivity graph $\mathcal{P}$ of a 37-qubit heavy-hexagon computer. Now, the strategy to minimize the SWAP overhead is to find a TT, $\mathcal{T} = (V_\mathcal{T}, E_\mathcal{T})$, that is congruent with the topology of $\mathcal{P}$. More precisely, suppose that $\mathcal{T}$ is a subgraph of $\mathcal{P}$ (i.e., $V_\mathcal{T} = V_\mathcal{P}$ and $E_\mathcal{T} \subseteq E_\mathcal{P}$). Then, any path from the root to leaf in $\mathcal{T}$ is a path in $\mathcal{P}$ and, consequently, no SWAPs are required to apply a gate generated by a Majorana string. A similar argument can be used for gates generated by single-excitation operators.

A tree subgraph $\mathcal{T}$ that spans all the nodes in a graph $\mathcal{P}$ is called a *spanning tree* (ST). If $\mathcal{P}$ is a tree itself, then the choice of ST is unique. A general graph, however, may have several STs. A degree-Δ-constrained ST is one such tree that has no vertices with a degree greater than Δ. In our case, since we need the subgraph $\mathcal{T}$ to be a ternary tree in order to define a mapping, all nodes but one must be at most degree 4 and the root degree 3. This implies that it is not always possible to find such a tree (e.g., if $\mathcal{P}$ is a tree but not degree-Δ constrained with $\Delta \leq 4$). Moreover, even if a degree-4-constrained tree subgraph exists, finding it is generally hard (in fact, simply determining

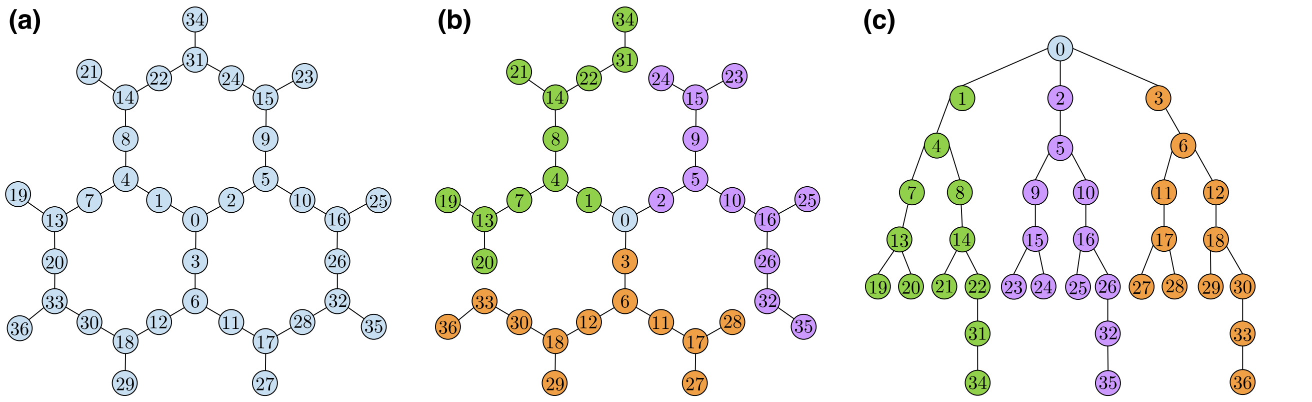


FIG. 5. (a) The connectivity graph of a 37-qubit heavy-hexagon processor. (b) The unfolded tree, illuminating how the paths of the tree follow the hardware topology in order to create a hardware-efficient mapping. The coloring of vertices indicates where the three initial branches from the root qubit unfold to. (c) The same tree as in (b), folded in the layout used in the rest of this work.

whether there is one is an NP-complete problem [40]). For our purposes, if $\mathcal{T}$ is not an ST of $\mathcal{P}$, it can nevertheless define a proper fermion-to-qubit mapping, although in such case some SWAPs may be needed to implement Majorana-generated unitary gates. Therefore, we propose using a greedy heuristic to find a TT that is close to a spanning tree and is in fact guaranteed to find an ST for some specific topologies. The routine is explained precisely in Appendix D (Algorithm 3).

The idea is to start by defining $\mathcal{T} = (V_\mathcal{T}, E_\mathcal{T})$, with empty $V_\mathcal{T}$ and $E_\mathcal{T}$, and grow the tree iteratively. First, choose a node to be the root r of the TT and define $\mathcal{L}_0 = r$. The choice of the root has an impact on the resulting Pauli weight and average delocalization of the mapping, as is discussed later on; we now choose it to be central in $\mathcal{P}$ (i.e., such that it minimizes the distance to its furthest node, $r = \text{argmin}_u \max_v d(u, v)$, where $d(u, v)$ is the topological distance between nodes u and v in $\mathcal{P}$). In Fig. 5(b), this is the pale blue central node. Next, define an empty set $\mathcal{L}_1$, and add to it $\min(\Delta(r), 3)$ neighbors of r in $\mathcal{P}$. For every node u that is added, add the link between r and u to $E_\mathcal{T}$. Note that r may have degree $\Delta(r) > 3$. In that case, the choice is not unique. For simplicity, we suggest choosing three of them randomly. Then, the process is repeated for each node in $\mathcal{L}_1$: define $\mathcal{L}_2 = \emptyset$ and add to it up to three neighbors of each node in $\mathcal{L}_1$ that have not yet been added, i.e., not in $\mathcal{L}_0 \cup \mathcal{L}_1 \cup \mathcal{L}_2$, and the corresponding links to $E_\mathcal{T}$. By iterating this process, at some point, all neighbors of all nodes in $\mathcal{L}_L$ for some L are added to some $\mathcal{L}_i$, so the procedure must stop. Now, let $V_\mathcal{T} = \bigcup_{i=1,\ldots,L} \mathcal{L}_i$. If $V_\mathcal{T} = V_\mathcal{P}$, we find a degree-4-constrained ST of $\mathcal{P}$. Note that this procedure succeeds with heavy-hexagon lattices, as shown in Figs. 5(b) and 5(c).

If the above procedure does not span all the qubits in $\mathcal{P}$, we need to add the remaining nodes in $V_\mathcal{P} \setminus V_\mathcal{T}$ to $\mathcal{T}$ according to some criterion. Note that it is always possible to include these nodes in $\mathcal{T}$ through "virtual edges" that connect physically detached nodes at the expense of SWAPs in the compilation. In order to minimize the resulting SWAP overhead, a good strategy is to try to minimize the physical topological distance between qubits connected in $\mathcal{T}$. This can be achieved following a greedy criterion: for every node u in $V_\mathcal{P} \setminus V_\mathcal{T}$, find amongst the nodes v in $V_\mathcal{T}$ with a downward degree less than 3 in $\mathcal{T}$ the ones that minimize the distance $d(u, v)$ in $\mathcal{P}$, and connect u to one of them.

It is worth noting a few aspects of this method. On the one hand, since at each step in the first part of the algorithm we add as many neighbors of each node as the topology allows, we are implicitly minimizing the depth of the resulting tree. Indeed, note that if the physical device is all-to-all connected, then the resulting graph is the TT from Ref. [30] with optimal depth $\mathcal{O}(\log_3 N)$. In the case of the heavy-hexagon topology, the greedy algorithm succeeds in finding degree-constrained spanning trees with depth scaling as $O(\sqrt{N})$, i.e., with quadratically lower Pauli weight than JW. This latter point can be seen through geometric arguments: the number of qubits at a given topological distance smaller than R from a chosen root node r scales as R^2. On the other hand, if the connectivity graph is a chain, the algorithm, as presented above, would choose as root r a node in the center of the chain. While this would lead to a mapping with Pauli weight lower than JW, the occupation would be more delocalized than in the latter case. Instead, if one is interested in minimizing delocalization, a better choice of the root is a node that lies on an extreme of a *diameter* of $\mathcal{P}$ (i.e., one of its longest shortest paths), so that the edges along the longest shortest path can be later labeled with Z, hence maximizing h_Z. In such a case, JW would be obtained for a chain. In general, this tradeoff between Pauli weight and delocalization can be easily controlled with the choice of the root node.

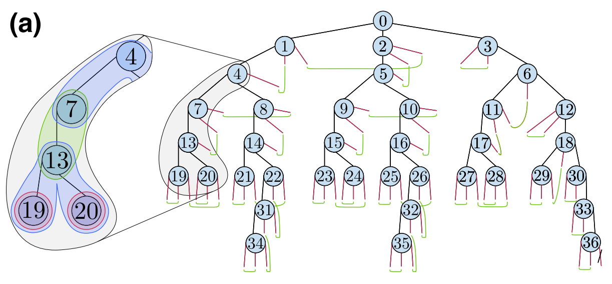
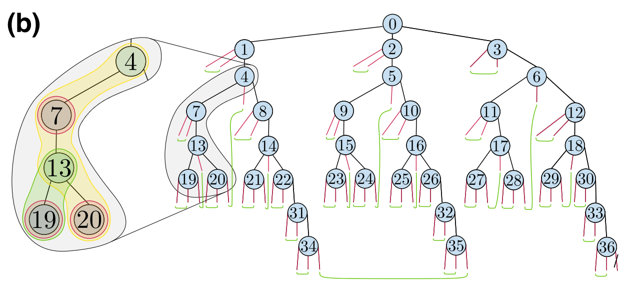


FIG. 6. Mappings resulting from two different labeling strategies applied to the tree in Fig. 5. (a) By applying the homogeneous localization labeling, modes associated with qubits with descendants are delocalized in a rather even fashion: nodes with one and two descendants have delocalization $D_u = 1$ and $D_u = 2$, respectively. The enlarged area involving qubits 4, 7, 13, 19, and 20 further illustrates the occupancy distribution. The modes associated with 4 and 13 involve three qubits, the mode in qubit 7 involves two, and the ones in 19 and 20 only one. (b) The application of the heterogeneous-localization strategy yields a very different delocalization structure. Nodes with one descendant are completely localized in this case ($D_u = 0$) but nodes with two descendants can be fairly delocalized. For instance, looking again at the four nodes in the shaded area, we see that the modes in qubits 7 and 13 are more localized than in the previous case, while the one in qubit 4 is delocalized among more qubits, with $D_4 = 3$. The particular case of the root node is a clear example, as it now exhibits delocalization $D_0 = 14$.



2. Labeling the tree

Once the TT has been identified, the next step is to introduce terminating legs and Pauli labels to the links to create a qubit tree. As discussed in the previous sections, in order to minimize the average delocalization, we must label with Z all edges from the root to the most distal node from it. In the case of the heavy hexagon in Fig. 5, we may do so with all the edges between nodes 0 (the root) and 36. Using the different labeling techniques, we can decide to a certain degree how mode-occupancy localization is spread. While these cannot affect the average delocalization, $\langle D_u \rangle$, they can determine how heterogeneously distributed among the qubits the occupancy can be. To that end, we introduce two different labeling strategies, which we coin *homogeneous* and *heterogeneous* localization, based on the discussion in Sec. III D.

Homogeneous localization proceeds by maximizing the number of XY branches amongst edges stemming from the same nodes, in a similar fashion as in Fig. 2 (left). Thus, pairs of edges below a node are assigned X and Y labels, while single edges have an X label. Heterogeneous localization, instead, maximizes the number of Z labels amongst edges: if a node has two descendants, one of the edges is assigned a Z and the other one an X. If the node has only one edge, it is assigned a Z. In this way, the heterogeneous-localization assignment tends to localize the occupation of single-edge nodes at the expense of the delocalization of nodes above them, hence resulting in typically more heterogeneous distributions of localization.

In Fig. 6, we depict the two labeling outcomes for the heavy-hexagon topology. Homogeneous localization produces many operators with occupancy depending on at worst case three qubits, giving a typical delocalization $D_u = 2$ for those. This is also reflected in the fact that the green lines representing the pairings do not span distant qubits. The number of completely local operators, with specific delocalization $D_u = 0$, is 16 in this case, whereas in the case of heterogeneous localization, 27 operators are fully localized. This has the side effect of creating fewer operators with higher delocalization, such as the number operator for the mode associated with the root (qubit 0), with delocalization $D_0 = 14$. Table I in Appendix E exhibits mode operators generated for both localization schemes in this heavy-hexagon example.

The choice of one delocalization scheme or the other is application specific and will alter the structure of the resulting circuit. This is evident when considering a highly delocalized mode operator. Operations derived from this mode will generally act on more qubits than its localized counterpart, consequently resulting in more expensive

```
Find a ternary tree $\mathcal{T}$ congruent with the physical
 connectivity graph $\mathcal{P}$ using Algorithm 3, which
 consists of the routines:
    a. Find a degree-constrained tree subgraph $\mathcal{T}$ using
       greedy search throughout $\mathcal{P}$.
    b. If the resulting tree does not span all nodes, add
       the remaining ones connecting them as to
       minimise the physical distance to nodes already
       in $\mathcal{T}$.
Add legs and introduce labels to $\mathcal{T}$ using Algorithm 4,
choosing among
    a. Homogeneous localisation: occupancy is spread
       evenly over qubits in the tree.
    b. Heterogeneous localisation: a subset of mode
       operators will act on many qubits while
       reducing the amount that others act on.
Pair the generated strings using Algorithm 1.
```

Algorithm 2. Bonsai algorithm.

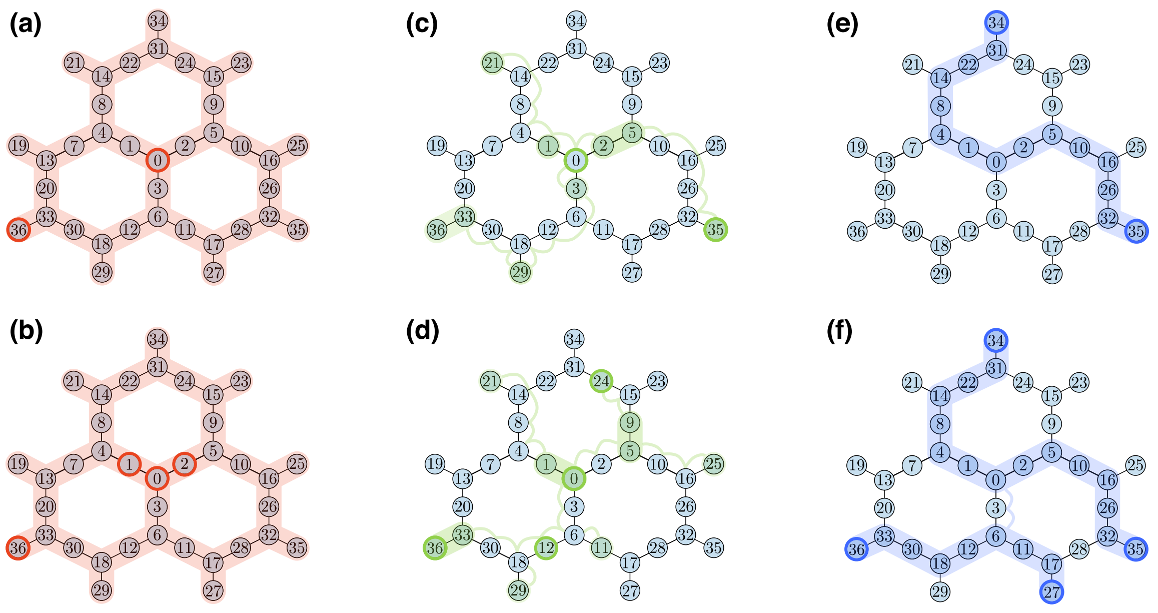


FIG. 7. The highlight indicates qubits involved in application of worst-case excitations, both for one- and two-particle terms. The top (a),(c),(e), and bottom (b),(d),(f) rows correspond to single- and double-excitation terms, respectively. For simplicity, we assume that the identification between modes j and qubits u in Eq. (10) is such that $j = u$. Qubits to which modes are associated are circled in red, blue, and green for the JW, BK, and the custom map. For example, (a) and (b) correspond to qubit operators of modes-$(0, 36)$ and modes-$(0, 1, 2, 36)$ accordingly. In the case of JW [(a) and (b)], both excitations result in gates acting on all qubits. For BK [(c) and (d)], the resulting gates will act on many disconnected qubits, resulting in a high SWAP cost completely mitigating its logarithmic scaling benefit. The custom mapping [(e) and (f)] generated by the Bonsai algorithm, on the other hand, presents much simpler worst-case scenarios. The single- and double-excitation terms involve many fewer qubits than JW, and only two SWAPs are required to connect the separate highlighted regions in the latter case.

circuits. However, since every tree mapping has a certain level of average delocalization, careful handling is required. In certain cases, a few modes may hold little importance and can be delocalized without significant detrimental consequences. In such scenarios, employing the heterogeneous-localization strategy to localize the relevant modes can prove beneficial. This will lead to an overall improvement as frequently used modes become less delocalized. Conversely, in cases where assumptions about the structure cannot be made, applying homogeneous localization may prove a safe option. This strategy spreads the delocalization uniformly, ensuring a balanced distribution across the modes.

With these mappings, circuit cost is reduced in two ways. Given that Majorana products follow paths along the paths in the tree structure, the number of SWAP gates needed for single excitations is zero and the number is diminished for double excitations due to Majorana products following paths along the tree structure. The number of entangling gates is further mitigated by lessening the Pauli weight. This is illustrated in Fig. 7, where we highlight the interaction maps of worst-case single and double excitations for the mapping obtained through homogeneous localization, compared to the JW and BK mappings. In both cases, JW acts extensively on the whole system. For BK, the qubits involved are disconnected and will need many SWAPs to compile, mitigating the circuit benefits of the logarithmic Pauli-weight scaling of the encoding. This is not the case for our custom encoding, where the reduction is approximately two thirds of the system for single and one third for double excitations. For the latter case, two SWAP gates are required to bridge across the disconnected interaction regions.

Another interesting aspect of these custom mappings is that it simplifies the transpilation of the circuits. Since the mapping is designed to be congruent with the hardware, it is not necessary to search for the optimal qubit assignment but only to solve the Steiner-graph problem to determine where SWAP gates are required. In addition, for two-dimensional devices other than the heavy-hexagon based connectivity studied here (e.g., Google's Sycamore grid topology [41]), we expect a similar square-root scaling.

It is noteworthy that the formalisms presented in this study can be expanded to tackle the difficulties posed by fully connected devices like ion traps. This could involve exploring modified mappings to minimize the Hamiltonian Pauli weight linked to reduced measurement cost [42] or devising mappings based on the structure of the circuit rather than the hardware to mitigate costs. However, such investigations are deferred to future research.

V. CONCLUSIONS

In this work, we consider fermion-to-qubit mappings relying on the identification of sets of Pauli strings obeying the anticommutation relations of Majorana operators. Within this context, we focus on a specific class, arguably the simplest one to work with, in which the Pauli strings have a nontrivial overlap involving just one qubit. We then present a framework that enables sampling such mappings while designing many of their resulting properties. An important element of the methodology is the pairing algorithm that ensures the preservation of separability, i.e., that uncorrelated fermionic states are mapped to uncorrelated qubit states. Interestingly, the framework contains paradigmatic mappings as particular instances, which allows us to interpolate between them at will.

With this framework at hand, we devise an algorithm to design hardware-specific mappings with lower SWAP overhead than other paradigmatic mappings while retaining a fair localization of the fermionic occupation in qubit space. When applied to the heavy-hexagon architecture, we obtain a mapping with a quadratically lower Pauli weight than JW. Importantly, the mapping enables the application of single-excitation operations with no SWAP overhead and of double excitations with a minor one. This can result in a significant improvement in circuit complexity with respect to hardware-agnostic mappings.

Currently, JW is the mapping of choice in most simulations on limited-connectivity hardware [43–46], partly due to the fact that its linear generating tree structure (see Fig. 3) makes it easy to find a set of qubits with that connectivity within the device. For other mappings such as BK or TT, suitable subgraphs that have treelike structure are unattainable on limited connectivity, resulting in a SWAP overhead negating the logarithmic advantage. Bonsai encodings, on the other hand, enable the leveraging of higher dimensions of limited-connectivity graphs to reduce simulation cost, as they naturally extend the hardware suitability of JW while reducing the nonlocality of the mapping.

The versatility of the approach here presented enables many other possibilities. In terms of designing mappings, the choice of the cost function to be optimized for is not unique, so the Bonsai algorithm can be naturally extended to produce encodings with different desirable properties. In particular, an important application is to extract relevant physical quantities of the system using local informationally complete positive operator-valued measures (POVMs) [30,42,47]. In this case, the Pauli weight of the observable is the dominant figure of merit, which is why the authors have proposed the logarithmic depth regular ternary tree in Ref. [30]. While that is the optimal mapping in terms of measurement cost for arbitrary fermionic reduced density matrix elements, in practice, one is typically interested in specific observables such as the energy. In that case, the mapping may be further optimized to reduce the measurement cost of, e.g., the Hamiltonian of the system. Moreover, it would be interesting to do so while limiting the incurred SWAP overhead on specific hardware.

In broader more theoretical terms, we emphasize that the bulk of the work here presented is devoted to a specific subset of all the possible mappings, the 1-NTO class, which includes all the widely used encodings. As we prove, with the pairing that we introduce, any root-containing connected ternary tree yields a valid product-preserving fermion-to-qubit mapping. However, we also show with a counter-example that not all 1-NTO maps can be generated in this fashion, so the question of how to characterize and represent the space of 1-NTO encodings remains open. In addition, as noted in Sec. II, k-NTO maps with $k > 1$ do exist. This opens up the interesting prospect of studying these somewhat exotic mappings.

The Bonsai algorithm is part of *Aurora*'s suite of algorithms for chemistry simulation.

APPENDIX A: EXAMPLES OF EXOTIC FERMION-TO-QUBIT MAPPINGS

The most-used fermion-to-qubit mappings, such as the JW, BK, and parity mappings, are all 1-NTO mappings, and can even be generated from ternary trees. In this appendix, we provide a toy example of a fermion-to-qubit mapping that is not 1-NTO and another one that is 1-NTO but cannot be generated from ternary trees.

Consider the following mapping of a four-mode fermion system to a four-qubit system:

$$\begin{aligned} m_0 &\mapsto X_1X_2X_3, & m_1 &\mapsto Y_1Y_2Y_3,\\ m_2 &\mapsto X_0Z_1Y_2Y_3, & m_3 &\mapsto Y_0Z_1X_2X_3,\\ m_4 &\mapsto Y_0Y_1X_3, & m_5 &\mapsto X_0X_1Y_3,\\ m_6 &\mapsto X_0X_1X_2Z_3, & m_7 &\mapsto Y_0Y_1Y_2Z_3. \end{aligned}$$

One can easily check that the mapping satisfies criteria (A)–(C) of Sec. II C; thus it is a valid Majorana-string mapping. The nontrivial overlap between the Majorana strings $X_1X_2X_3$ and $Y_1Y_2Y_3$ is 3; thus this cannot be a 1-NTO mapping but is instead 3-NTO.

An example of a 1-NTO Majorana-string mapping (for three fermionic modes) which cannot be generated from a ternary tree is the following:

$$m_0 \mapsto X_0Z_1, \quad m_1 \mapsto Y_0Z_1,$$
$$m_2 \mapsto X_1Z_2, \quad m_3 \mapsto Y_1Z_2, \quad m_4 \mapsto Z_0X_2, \quad m_5 \mapsto Z_0Y_2.$$

APPENDIX B: ALGEBRAIC INDEPENDENCE OF SUBSETS OF TT-GENERATED PAULI STRINGS

The aim of this appendix is to prove that any subset $\mathcal{S}' \subset \mathcal{S}$ ($|\mathcal{S}'| < |\mathcal{S}|$) of the set $\mathcal{S}$ of $2N + 1$ Pauli strings generated by an N-node TT is algebraically independent, i.e., that there are no two different subsets $A \subseteq \mathcal{S}'$ and $B \subseteq \mathcal{S}'$, $A \neq B$, such that $\prod_{S_i \in A} S_i \propto \prod_{S_j \in B} S_j$.

First, note that it is enough to prove that no two *disjoint* subsets A and B leading to equal products exist, given that

$$\prod_{S_i \in A} S_i \propto \prod_{S_j \in B} S_j \Rightarrow \prod_{S_i \in A \setminus A \cap B} S_i \propto \prod_{S_j \in B \setminus A \cap B} S_j. \tag{B1}$$

The above implication stems from the fact that both products on the left-hand side can be multiplied by the Pauli strings in $A \cap B$. Since these Pauli strings appear twice in each resulting product and they anticommute with any Pauli string different from themselves, they cancel out to identity incurring at most a change of sign.

Following a similar reasoning as above, if there are two distinct and disjoint subsets $A \subset \mathcal{S}'$ and $B \subset \mathcal{S}'$ such that $\prod_{S_i \in A} S_i \propto \prod_{S_j \in B} S_j$,

$$\prod_{S_i \in A \cup B} S_i \propto \mathbb{1}_N, \tag{B2}$$

where $\mathbb{1}_N$ is the identity operator in the Hilbert space of N qubits. In short, it is enough to prove that there is no subset $\mathcal{I} \subseteq \mathcal{S}' \subset \mathcal{S}$ fulfilling $\prod_{S_i \in \mathcal{I}} S_i \propto \mathbb{1}_N$. In what follows, we prove this by showing that

$$\prod_{S_i \in \mathcal{I}} S_i \propto \mathbb{1}_N \Rightarrow \mathcal{I} = \mathcal{S}, \tag{B3}$$

so that no such $\mathcal{I} \subseteq \mathcal{S}'$ exists for any incomplete subset $\mathcal{S}'$ of $\mathcal{S}$.

Given a TT and a subset of its legs $\mathcal{I} \subseteq \mathcal{S}$, we can define a set of *link multiplicities* $\{\varphi_l\}$, where φ_l is an integer defined for every link l in the tree (be it an edge or a leg) and counts the number of paths from the root node to each of the legs in $\mathcal{I}$ that traverse link l. Now, if we assume that $\prod_{S_i \in \mathcal{I}} S_i \propto \mathbb{1}_N$, we can make the following observations:

(1) For any node u in the tree, the link multiplicities $\varphi_{l_x^{(u)}}$, $\varphi_{l_y^{(u)}}$, and $\varphi_{l_z^{(u)}}$ of the three links stemming downward from u must either be all even or all odd. This is a consequence of the fact that the product of Pauli strings in $\mathcal{I}$ results in a product of Pauli operators X_u, Y_u, and Z_u on qubit u. Since these operators anticommute with one another and their product must be proportional to identity according to our assumption above,

$$(X_u)^{\varphi_{l_x^{(u)}}} (Y_u)^{\varphi_{l_y^{(u)}}} (Z_u)^{\varphi_{l_z^{(u)}}} \propto \mathbb{1}, \tag{B4}$$

which can only be fulfilled if all three link multiplicities have equal parity.

(2) Consider a node u different from the root and let us refer to its upward-edge multiplicity by $\varphi_{l_{\mathrm{up}}^{(u)}}$. The downward-link multiplicities are $\varphi_{l_x^{(u)}}$, $\varphi_{l_y^{(u)}}$, and $\varphi_{l_z^{(u)}}$, as above. If the assumption $\prod_{S_i \in \mathcal{I}} S_i \propto \mathbb{1}_N$ holds, then $\varphi_{l_{\mathrm{up}}^{(u)}}$ must have the same parity as the three downward links. This is a direct consequence of observation 1 and of the fact that edge multiplicity is conserved,

$$\varphi_{l_{\mathrm{up}}^{(u)}} = \varphi_{l_x^{(u)}} + \varphi_{l_y^{(u)}} + \varphi_{l_z^{(u)}}, \tag{B5}$$

since every path that traverses $l_{\mathrm{up}}^{(u)}$ must traverse one of the three downward links. Indeed, the sum of an odd number of odd numbers is odd and no odd number can be obtained by adding even numbers.

These two observations imply that the parity of the link multiplicities is conserved at each node, i.e., all links reaching a node must have equal multiplicity parity if the assumption $\prod_{S_i \in \mathcal{I}} S_i \propto \mathbb{1}_N$ is true. Since the tree is connected, it follows that the multiplicity of all links in the graph must have the same parity. Given that the legs in $\mathcal{I}$ have multiplicity one, all links in the graph must have odd multiplicity. Thus, all legs in $\mathcal{S}$ must have multiplicity one and hence be in $\mathcal{I}$, which proves Eq. (B3).

APPENDIX C: FROM FOCK BASIS STATES TO COMPUTATIONAL BASIS STATES

In the main text, we show that, with the pairing introduced in Sec. III C, the fermionic vacuum is mapped to $|0\rangle^{\otimes N}$ and that states of the form $a_i^\dagger |\mathrm{vac}_f\rangle$ lead to computational basis states in qubit space. We now show that this is also true for any Fock basis state.

Consider an arbitrary Fock basis state $|\psi\rangle$ in which the fermionic modes in the subset $\mathcal{F} \subseteq \{0, 1, \ldots, N-1\}$ are occupied, i.e., $|\psi\rangle = \prod_{k \in \mathcal{F}} a_k^\dagger |\mathrm{vac}_f\rangle$. Since all the creation operators in the expression are different, they anticommute, so $|\psi\rangle$ can be written, up to a sign, by applying them in an arbitrary order.

Recall that, given a TT mapping, every fermionic mode j can be associated with a qubit u_j according to the pairing strategy [see Eq. (10)]. This identification allows us to associate an integer h_j to every mode in $\mathcal{F}$ indicating how deep u_j lies down the tree. More precisely, h_j is the topological distance between the associated qubit u_j of mode j and the root node. Now, consider a sequence $(R_0, \ldots, R_{|\mathcal{F}|-1})$ of the elements in $\mathcal{F}$ (i.e., $R_i \in \mathcal{F}$ for all $i \in \{0, \ldots, |\mathcal{F}|-1\}$ and $R_i = R_j \Leftrightarrow i = j$) following a top-down order, $h_{R_i} \le h_{R_{i+1}}\ \forall i \in \{0, \ldots, |\mathcal{F}|-1\}$. We can then construct $|\psi\rangle$ by applying the sequence of creation operators starting from the highest modes up in the tree

```
1  Define physical connectivity graph $\mathcal{P} = (V_\mathcal{P}, E_\mathcal{P})$.
2  Determine root node $r = \mathrm{argmin}_u \max_v d(u, v; \mathcal{P})$, where $d(u, v; \mathcal{P})$ is the topological distance between $u$ and $v$ in $\mathcal{P}$.
3  Define initial layer, $\mathcal{L}_0 = r$, height $h = 0$, and tree $\mathcal{T} = (V_\mathcal{T}, E_\mathcal{T})$ with $V_\mathcal{T} = E_\mathcal{T} = \emptyset$.
4  while $\mathcal{L}_h \neq \emptyset$ do
5      Define $\mathcal{L}_{h+1} = \emptyset$.
6      for $v \in \mathcal{L}_h$ do
7          Define the set of unassigned neighbours of $v$, $\mathcal{N}_v = \{w \in V_\mathcal{P} : (v, w) \in E_\mathcal{P} \wedge w \notin V_\mathcal{T}\}$.
8          if $|\mathcal{N}_v| > 3$ then
9              Define $\mathcal{N}'_v \subset \mathcal{N}_v$ containing three nodes randomly chosen from $\mathcal{N}_v$.
10             Set $\mathcal{N}'_v \to \mathcal{N}_v$.
11         Set $\mathcal{L}_{h+1} \cup \mathcal{N}_v \to \mathcal{L}_{h+1}$.
12         Set $V_\mathcal{T} \cup \mathcal{N}_v \to V_\mathcal{T}$.
13         Set $E_\mathcal{T} \cup \bigcup_{w \in \mathcal{N}_v} \{(v, w), (w, v)\} \to E_\mathcal{T}$.
14     Set $h + 1 \to h$.
15 for $u \in V_\mathcal{P} \setminus V_\mathcal{T}$ do
16     Determine set $\mathcal{A} \subseteq V_\mathcal{T}$ of nodes in $\mathcal{T}$ available to connect, $\mathcal{A} = \{u \in V_\mathcal{T} : |\{v \in V_\mathcal{T} : (u, v) \in E_\mathcal{T}\}| + \delta_{u,r} < 4\}$.
17     Find set $\mathcal{C} \subseteq \mathcal{A}$ of closest nodes to $u$, $\mathcal{C} = \{v \in \mathcal{A} : d(u, v; \mathcal{P}) = \min_w(\{d(u, w; \mathcal{P}) : w \in \mathcal{A}\})\}$.
18     if $|\mathcal{C}| > 1$ then
19         Define $\mathcal{C}' \subset \mathcal{C}$ containing one node randomly chosen from $\mathcal{C}$.
20         Set $\mathcal{C}' \to \mathcal{C}$.
21     Set $V_\mathcal{T} \cup \mathcal{C} \to V_\mathcal{T}$.
22     Set $E_\mathcal{T} \cup \{(u, v), (v, u)\} \to E_\mathcal{T}$, where $v \in \mathcal{C}$.
23 return $\mathcal{T}$
```

Algorithm 3. Qubit spanning-tree subroutine.

and following downward:

$$|\psi\rangle \propto \prod_{i=0,\ldots,|\mathcal{F}|-1} a^\dagger_{R_i} |\mathrm{vac}_f\rangle . \tag{C1}$$

Each fermionic operator $a^\dagger_j$ involves the two Majorana strings $S_{s_x^{(u_j)}}$ and $S_{s_y^{(u_j)}}$, which only differ on qubit u_j (on which they act with X_{u_j} and Y_{u_j}, respectively) and on all qubits in the X and Y branches lying below u_j in the tree; each of the two Majorana strings acts with a Z operator on the qubits on one of the branches but trivially on the qubits in the other branch. Therefore, if $|\phi\rangle$ is a computational basis state in which u_j and all the qubits below it are in the $|0\rangle$ state, $(S_{s_x^{(u_j)}} - iS_{s_y^{(u_j)}})\,|\phi\rangle = (S'_{s_x^{(u_j)}} - iS'_{s_y^{(u_j)}})\,|\phi\rangle$, where the Z Pauli operators on the qubits below u_j are substituted with identities in the primed Pauli strings, as in Sec. III C. Importantly, in the output vector, the state of u_j and possibly of other qubits above u_j in the tree are flipped but not the state of qubits below u_j. In addition, the vector remains a computational basis one.

With this setup, we can proceed in an inductive way. First, it is clear from the above discussion (and the one in the main text) that the state $a^\dagger_{R_0}|\mathrm{vac}_f\rangle$ in qubit space—let us denote it by $|\phi_0\rangle$—is a computational basis state. Second, it can be seen that if the mapped state $\prod_{i=0,\ldots,n} a^\dagger_{R_i}|\mathrm{vac}_f\rangle$, $|\phi_n\rangle$, is a computational basis state, then so is $|\phi_{n+1}\rangle$. This follows from

$$|\phi_{n+1}\rangle = \frac{1}{2}\left(S_{s_x^{(u_{N+1})}} - iS_{s_y^{(u_{n+1})}}\right)|\phi_n\rangle , \tag{C2}$$

```
1 Procedure Minimise delocalisation by maximising all-Z branch length:
2     Find the longest path $\ell$ in $\mathcal{T}$.
3     Associate a $Z$ label to every edge along $\ell$.
4 Procedure Homogeneous localisation:
5     For every node in the tree, add labels to each of its unlabelled descending edges with priority 1) $X$, 2) $Y$, and 3) $Z$
        (that is, single edges are labelled with $X$ and double edges with $XY$).
6     Add labels to all legs.
7 Procedure Heterogeneous localisation:
8     For every node in the tree, add labels to each of its unlabelled descending edges with priority 1) $Z$ (if available), 2)
        $X$, and 3) $Y$ (that is, single edges are labelled with $Z$ and double edges with $ZX$).
9     Add labels to all legs.
```

Algorithm 4. Labeling subroutine.

TABLE I. The qubit mode operators generated from the trees in Fig. 6. Localized operators are qubit mode operators with raising, P_j^+, and lowering, P_j^- operators acting on the jth qubit. Specific delocalization is clear from the number of Pauli Z operators in the brackets.

	$a_j^{(\dagger)}$	
j	Homogeneous localization	Heterogeneous localization
0	$\frac{1}{2}(X_0Z_1 \mp iY_0Z_2)$	$\frac{1}{2}(X_0Z_1Z_4Z_8Z_{14}Z_{22}Z_{31}Z_{34} \mp iY_0Z_2Z_5Z_{10}Z_{16}Z_{26}Z_{32}Z35)$
1	$\frac{1}{2}X_0(X_1Z_4 \mp iY_0)$	$X_0P_1^{\pm}$
2	$\frac{1}{2}Y_0(X_2Z_5 \mp iY_2)$	$Y_0P_2^{\pm}$
3	$Z_0P_3^{\pm}$	$Z_0P_3^{\pm}$
4	$\frac{1}{2}X_0X_1(X_4Z_7 \mp iY_4Z_8)$	$\frac{1}{2}X_0Z_1(X_4Z_7Z_{13}Z_{20} \mp Y_4)$
5	$\frac{1}{2}Y_0X_2(X_5Z_9 \mp iY_5Z_{10})$	$\frac{1}{2}Y_0Z_2(X_5Z_9Z_{15}Z_{24} \mp Y_5)$
6	$\frac{1}{2}Z_0Z_3(X_6Z_{11} \mp iY_6)$	$\frac{1}{2}Z_0Z_3(X_6Z_{11}Z_{17}Z_{28} \mp Y_6)$
7	$\frac{1}{2}X_0X_1X_4(X_7Z_{13} \mp iY_7)$	$X_0Z_1X_4P_7^{\pm}$
8	$\frac{1}{2}X_0X_1Y_4(X_8Z_{14} \mp iY_8)$	$X_0Z_1Z_4P_8^{\pm}$
9	$\frac{1}{2}Y_0X_2X_5(X_9Z_{15} \mp iY_9)$	$Y_0Z_2X_5P_9^{\pm}$
10	$\frac{1}{2}Y_0X_2Y_5(X_{10}Z_{16} \mp iY_{10})$	$Y_0Z_2Z_5P_{10}^{\pm}$
11	$\frac{1}{2}Z_0Z_3Z_6(X_{11}Z_{17} \mp iY_{11})$	$Z_0Z_3X_6P_{11}^{\pm}$
12	$Z_0Z_3Z_6P_{12}^{\pm}$	$Z_0Z_3Z_6P_{12}^{\pm}$
13	$\frac{1}{2}X_0X_1X_4X_7(X_{13}Z_{19} \mp iY_{13}Z_{20})$	$\frac{1}{2}X_0Z_1X_4Z_7(X_{13}Z_{19} \mp iY_{13})$
14	$\frac{1}{2}X_0X_1Y_4X_8(X_{14}Z_{21} \mp iY_{14}Z_{22})$	$\frac{1}{2}X_0Z_1Z_4Z_8(X_{14}Z_{21} \mp iY_{14})$
15	$\frac{1}{2}Y_0X_2X_5X_9(X_{15}Z_{23} \mp iY_{15}Z_{24})$	$\frac{1}{2}Y_0Z_2X_5Z_9(X_{15}Z_{23} \mp iY_{15})$
16	$\frac{1}{2}Y_0X_2Y_5X_{10}(X_{16}Z_{25} \mp iY_{16}Z_{26})$	$\frac{1}{2}Y_0Z_2Z_5Z_{10}(X_{16}Z_{25} \mp iY_{16})$
17	$\frac{1}{2}Y_0X_2X_5X_9(X_{15}Z_{23} \mp iY_{15}Z_{24})$	$\frac{1}{2}Z_0Z_3X_6Z_{11}(X_{17}Z_{27} \mp iY_{17})$
18	$\frac{1}{2}Y_0X_2Y_5X_{10}(X_{16}Z_{25} \mp iY_{16}Z_{26})$	$\frac{1}{2}Z_0Z_3Z_6Z_{12}(X_{18}Z_{29} \mp iY_{18})$
19	$X_0X_1X_4X_7X_{13}P_{19}^{\pm}$	$X_0Z_1X_4Z_7X_{13}P_{19}^{\pm}$
20	$X_0X_1X_4X_7Y_{13}P_{20}^{\pm}$	$X_0Z_1X_4Z_7Z_{13}P_{20}^{\pm}$
21	$X_0X_1Y_4Y_8X_{14}P_{21}^{\pm}$	$X_0Z_1Z_4Z_8X_{14}P_{21}^{\pm}$
22	$\frac{1}{2}X_0X_1Y_4X_8Y_{14}(X_{22}Z_{31} \mp iY_{22})$	$X_0Z_1Z_4Z_8Z_{14}P_{22}^{\pm}$
23	$Y_0X_2X_5X_9X_{15}P_{23}^{\pm}$	$Y_0Z_2X_5Z_9X_{15}P_{23}^{\pm}$
24	$Y_0X_2X_5X_9Y_{15}P_{24}^{\pm}$	$Y_0Z_2X_5Z_9Z_{15}P_{24}^{\pm}$
25	$Y_0X_2Y_5X_{10}X_{16}P_{25}^{\pm}$	$Y_0Z_2Z_5Z_{10}X_{16}P_{25}^{\pm}$
26	$\frac{1}{2}Y_0X_2Y_5X_{10}Y_{16}(X_{26}Z_{32} \mp iY_{26})$	$Y_0Z_2Z_5Z_{10}Z_{16}P_{26}^{\pm}$
27	$Z_0Z_6X_6X_{11}X_{17}P_{27}^{\pm}$	$Z_0Z_3X_6Z_{11}X_{17}P_{27}^{\pm}$
28	$Z_0Z_6X_6X_{11}Y_{17}P_{28}^{\pm}$	$Z_0Z_3X_6Z_{11}Z_{17}P_{28}^{\pm}$
29	$Z_0Z_6X_6Z_{12}X_{18}P_{29}^{\pm}$	$Z_0Z_3Z_6Z_{12}X_{18}P_{29}^{\pm}$
30	$Z_0Z_6X_6Z_{12}Z_{18}P_{30}^{\pm}$	$Z_0Z_3Z_6Z_{12}Z_{18}P_{30}^{\pm}$
31	$\frac{1}{2}X_0X_1Y_4Y_8Y_{14}X_{22}(X_{31}Z_{34} \mp iY_{31})$	$X_0Z_1Z_4Z_8Z_{14}Z_{22}P_{31}^{\pm}$
32	$\frac{1}{2}Y_0X_2Y_5X_{10}Y_{16}X_{26}(X_{32}Z_{35} \mp iY_{32})$	$Y_0Z_2Z_5Z_{10}Z_{16}Z_{26}P_{32}^{\pm}$
33	$Z_0Z_6X_6Z_{12}Z_{18}Z_{30}P_{33}^{\pm}$	$Z_0Z_3Z_6Z_{12}Z_{18}Z_{30}P_{33}^{\pm}$
34	$X_0X_1Y_4X_8Y_{14}X_{22}X_{31}P_{34}^{\pm}$	$X_0Z_1Z_4Z_8Z_{14}Z_{22}Z_{31}P_{34}^{\pm}$
35	$Y_0X_2Y_5X_{10}Y_{16}X_{26}X_{32}P_{35}^{\pm}$	$Y_0Z_2Z_5Z_{10}Z_{16}Z_{26}Z_{32}P_{35}^{\pm}$
36	$Z_0Z_6X_6Z_{12}Z_{18}Z_{30}Z_{33}P_{36}^{\pm}$	$Z_0Z_3Z_6Z_{12}Z_{18}Z_{30}Z_{33}P_{36}^{\pm}$

where u_{n+1} is the qubit associated with the fermionic mode R_{n+1}. Since all the Majorana strings that must be applied to prepare $|\phi_n\rangle$ from $|0\rangle^{\otimes N}$ act on u_{n+1} and all qubits below it with either identity or with Z (again, given that $h_{R_i} \leq h_{R_{n+1}}\ \forall i \in \{0,\ldots,n\}$), the state of each of those qubits must be $|0\rangle$. As we show above, along with the condition that $|\phi_n\rangle$ be a computational basis state (which here is true by assumption) this guarantees that $|\phi_{n+1}\rangle$ is a computational basis state as well.

APPENDIX D: ALGORITHMS AND ROUTINES IN MORE DETAIL

APPENDIX E: THE HEAVY-HEXAGON MAPPINGS EXPLICITLY

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

[2] F. Arute, K. Arya, R. Babbush, D. Bacon, J. C. Bardin, R. Barends, R. Biswas, S. Boixo, F. G. Brandao, D. A. Buell, and B. Burkett, Quantum supremacy using a programmable superconducting processor, Nature **574**, 505 (2019).

[3] H.-S. Zhong, H. Wang, Y. H. Deng, M. C. Chen, L. C. Peng, Y. H. Luo, J. Qin, D. Wu, X. Ding, Y. Hu, and P. Hu,

Quantum computational advantage using photons, Science **370**, 1460 (2020).

[4] Y. Wu, W. S. Bao, S. Cao, F. Chen, M. C. Chen, X. Chen, T. H. Chung, H. Deng, Y. Du, D. Fan, and M. Gong, Strong Quantum Computational Advantage Using a Superconducting Quantum Processor, Phys. Rev. Lett. **127**, 180501 (2021).

[5] L. S. Madsen, F. Laudenbach, M. F. Askarani, F. Rortais, T. Vincent, J. F. Bulmer, F. M. Miatto, L. Neuhaus, L. G. Helt, M. J. Collins, and A. E. Lita, Quantum computational advantage with a programmable photonic processor, Nature **606**, 75 (2022).

[6] S. Endo, S. C. Benjamin, and Y. Li, Practical Quantum Error Mitigation for Near-Future Applications, Phys. Rev. X **8**, 031027 (2018).

[7] S. Endo, Z. Cai, S. C. Benjamin, and X. Yuan, Hybrid quantum-classical algorithms and quantum error mitigation, J. Phys. Soc. Jpn. **90**, 032001 (2021).

[8] Z. Cai, R. Babbush, S. C. Benjamin, S. Endo, W. J. Huggins, Y. Li, J. R. McClean, and T. E. O'Brien, Quantum error mitigation (2022), arXiv preprint ArXiv:2210.00921.

[9] E. v. d. Berg, Z. K. Minev, A. Kandala, and K. Temme, Probabilistic error cancellation with sparse Pauli-Lindblad models on noisy quantum processors (2022), arXiv preprint ArXiv:2201.09866.

[10] K. Bharti, A. Cervera-Lierta, T. H. Kyaw, T. Haug, S. Alperin-Lea, A. Anand, M. Degroote, H. Heimonen, J. S. Kottmann, T. Menke, W.-K. Mok, S. Sim, L.-C. Kwek, and A. Aspuru-Guzik, Noisy intermediate-scale quantum algorithms, Rev. Mod. Phys. **94**, 015004 (2022).

[11] A. Kandala, A. Mezzacapo, K. Temme, M. Takita, M. Brink, J. M. Chow, and J. M. Gambetta, Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets, Nature **549**, 242 (2017).

[12] P. K. Barkoutsos, J. F. Gonthier, I. Sokolov, N. Moll, G. Salis, A. Fuhrer, M. Ganzhorn, D. J. Egger, M. Troyer, A. Mezzacapo, S. Filipp, and I. Tavernelli, Quantum algorithms for electronic structure calculations: Particle-hole Hamiltonian and optimized wave-function expansions, Phys. Rev. A **98**, 022322 (2018).

[13] P. J. Ollitrault, A. Kandala, C.-F. Chen, P. K. Barkoutsos, A. Mezzacapo, M. Pistoia, S. Sheldon, S. Woerner, J. M. Gambetta, and I. Tavernelli, Quantum equation of motion for computing molecular excitation energies on a noisy quantum processor, Phys. Rev. Res. **2**, 043140 (2020).

[14] V. Lordi and J. M. Nichol, Advances and opportunities in materials science for scalable quantum computing, MRS Bull. **46**, 589 (2021).

[15] Y. Cao, J. Romero, and A. Aspuru-Guzik, Potential of quantum computing for drug discovery, IBM J. Res. Dev. **62**, 6 (2018).

[16] N. S. Blunt, J. Camps, O. Crawford, R. Izsák, S. Leontica, A. Mirani, A. E. Moylett, S. A. Scivier, C. Sünderhauf, and P. Schopf, *et al.*, A perspective on the current state-of-the-art of quantum computing for drug discovery applications (2022), arXiv preprint ArXiv:2206.00551.

[17] S. McArdle, S. Endo, A. Aspuru-Guzik, S. C. Benjamin, and X. Yuan, Quantum computational chemistry, Rev. Mod. Phys. **92**, 015003 (2020).

[18] J. Tilly, H. Chen, S. Cao, D. Picozzi, K. Setia, Y. Li, E. Grant, L. Wossnig, I. Rungger, and G. H. Booth, *et al.*, The variational quantum eigensolver: A review of methods and best practices, Phys. Rep. **986**, 1 (2022).

[19] P. Jordan and E. Wigner, Über das Paulische Äquivalenzverbot, Z. für Phys. **47**, 631 (1928).

[20] S. B. Bravyi and A. Y. Kitaev, Fermionic quantum computation, Ann. Phys. (NY) **298**, 210 (2002).

[21] S. Bravyi, J. M. Gambetta, A. Mezzacapo, and K. Temme, Tapering off qubits to simulate fermionic Hamiltonians (2017), arXiv preprint ArXiv:1701.08213.

[22] M. Chiew and S. Strelchuk, Optimal fermion-qubit mappings (2021), arXiv preprint ArXiv:2110.12792.

[23] K. Setia and J. D. Whitfield, Bravyi-Kitaev superfast simulation of electronic structure on a quantum computer, J. Chem. Phys. **148**, 164104 (2018).

[24] M. Steudtner and S. Wehner, Fermion-to-qubit mappings with varying resource requirements for quantum simulation, New J. Phys. **20**, 063010 (2018).

[25] V. Havlíček, M. Troyer, and J. D. Whitfield, Operator locality in the quantum simulation of fermionic models, Phys. Rev. A **95**, 032332 (2017).

[26] R. W. Chien and J. D. Whitfield, Custom fermionic codes for quantum simulation (2020), arXiv preprint ArXiv:2009.11860.

[27] M. Steudtner and S. Wehner, Quantum codes for quantum simulation of fermions on a square lattice of qubits, Phys. Rev. A **99**, 022308 (2019).

[28] D. A. Mazziotti, S. E. Smart, and A. R. Mazziotti, Quantum simulation of molecules without fermionic encoding of the wave function, New J. Phys. **23**, 113037 (2021).

[29] S. E. Smart and D. A. Mazziotti, Many-fermion simulation from the contracted quantum eigensolver without fermionic encoding of the wave function, Phys. Rev. A **105**, 062424 (2022).

[30] Z. Jiang, A. Kalev, W. Mruczkiewicz, and H. Neven, Optimal fermion-to-qubit mapping via ternary trees with applications to reduced quantum states learning, Quantum **4**, 276 (2020).

[31] A. Vlasov, Clifford algebras, spin groups and qubit trees, Quanta **11**, 97 (2022).

[32] Q.-S. Li, H.-Y. Liu, Q. Wang, Y.-C. Wu, and G.-P. Guo, A unified framework of transformations based on the Jordan-Wigner transformation, J. Chem. Phys. **157**, 134104 (2022).

[33] N. P. Sawaya, M. Smelyanskiy, J. R. McClean, and A. Aspuru-Guzik, Error sensitivity to environmental noise in quantum circuits for chemical state preparation, J. Chem. Theory Comput. **12**, 3097 (2016).

[34] H. L. Tang, V. O. Shkolnikov, G. S. Barron, H. R. Grimsley, N. J. Mayhall, E. Barnes, and S. E. Economou, Qubit-ADAPT-VQE: An Adaptive Algorithm for Constructing Hardware-Efficient Ansätze on a Quantum Processor, PRX Quantum **2**, 020310 (2021).

[35] H. R. Grimsley, S. E. Economou, E. Barnes, and N. J. Mayhall, An adaptive variational algorithm for exact molecular simulations on a quantum computer, Nat. Commun. **10**, 1 (2019).

[36] P. G. Anastasiou, Y. Chen, N. J. Mayhall, E. Barnes, and S. E. Economou, Tetris-ADAPT-VQE: An adaptive algorithm

that yields shallower, denser circuit ansätze (2022), arXiv preprint ArXiv:2209.10562.

[37] R. Alicki and M. Fannes, *Quantum Dynamical Systems* (Oxford University Press, 2001).

[38] Z. Zimborás, R. Zeier, M. Keyl, and T. Schulte-Herbrüggen, A dynamic systems approach to fermions and their relation to spins, EPJ Quantum Technol. **1**, 11 (2014).

[39] S. Szalay, Z. Zimborás, M. Máté, G. Barcza, C. Schilling, and Ö. Legeza, Fermionic systems for quantum information people, J. Phys. A: Math. Theor. **54**, 393001 (2021).

[40] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-completeness* (Freeman, 1979), https://books.google.ie/books?id=fjxGAQAAIAAJ.

[41] F. Arute, *et al.*, Quantum supremacy using a programmable superconducting processor, Nature **574**, 505 (2019).

[42] G. García-Pérez, M. A. C. Rossi, B. Sokolov, F. Tacchino, P. K. Barkoutsos, G. Mazzola, I. Tavernelli, and S. Maniscalco, Learning to Measure: Adaptive Informationally Complete Generalized Measurements for Quantum Algorithms, PRX Quantum **2**, 040342 (2021).

[43] A. Bentellis, A. Matic-Flierl, C. B. Mendl, and J. M. Lorenz, Benchmarking the variational quantum eigensolver using different quantum hardware (2023), ArXiv:2305.07092.

[44] H. Ma, M. Govoni, and G. Galli, Quantum simulations of materials on near-term quantum computers, npj Computational Mater. **6**, 85 (2020).

[45] M. Motta, E. Ye, J. R. McClean, Z. Li, A. J. Minnich, R. Babbush, and G. K.-L. Chan, Low rank representations for quantum simulation of electronic structure, npj Quantum Inf. **7**, 83 (2021).

[46] G. A. Quantum, *et al.*, Hartree-Fock on a superconducting qubit quantum computer, Science **369**, 1084 (2020).

[47] A. Glos, A. Nykänen, E.-M. Borrelli, S. Maniscalco, M. A. Rossi, Z. Zimborás, and G. García-Pérez, Adaptive POVM implementations and measurement error mitigation strategies for near-term quantum devices (2022), arXiv preprint ArXiv:2208.07817.