Hardware-efficient entangled measurements for variational quantum algorithms

Francisco Escudero,[1, 2] David Fernández-Fernández,[1, 3] Gabriel Jaumà,[1] Guillermo F. Peñas,[1] and Luciano Pereira[1, *]

[1]*Instituto de Física Fundamental, IFF-CSIC, Calle Serrano 113b, 28006 Madrid, Spain*
[2]*CWI & QuSoft, Science Park 123, 1098 XG Amsterdam, The Netherlands*
[3]*Instituto de Ciencia de Materiales de Madrid, ICMM-CSIC, 28049 Madrid, Spain*

Variational algorithms have received significant attention in recent years due to their potential to solve practical problems using noisy intermediate-scale quantum (NISQ) devices. A fundamental step of these algorithms is the evaluation of the expected value of Hamiltonians, and hence efficient schemes to perform this task are required. The standard approach employs local measurements of Pauli operators and requires a large number of circuits. An alternative is to use entangled measurements, which might introduce additional gates between physically disconnected qubits that harm the performance. To solve this problem, we propose hardware-efficient entangled measurements (HEEM), that is, measurements that permit only entanglement between physically connected qubits. We show that this strategy enhances the evaluation of molecular Hamiltonians in NISQ devices by reducing the required circuits without increasing their depth. We provide quantitative metrics of how this approach offers better results than local measurements and arbitrarily entangled measurements. We estimate the ground-state energy of the H_2O molecule with classical simulators and quantum hardware using the variational quantum eigensolver with HEEM.

I. INTRODUCTION.

We are currently in the era of noisy intermediate-scale quantum computers (NISQ). The main limitations of these devices are short coherence times and noisy entanglement gates; therefore, NISQ circuits must inevitably have a low depth [1]. Given that, a lot of effort has been devoted to the design and implementation of quantum algorithms that only use low-depth circuits [2–5]. Among such algorithms, one family that has gained attention is variational quantum algorithms (VQAs) [6–8]: hybrid quantum-classical methods where a classical computer guides a quantum computer to produce variational quantum states and to measure their expected value, with the goal of minimizing an objective function encoded in a Hamiltonian. The most famous VQAs are the variational quantum eigensolver (VQE) [9] and the quantum approximate optimization algorithm (QAOA) [10]. Multiple VQAs have been applied in a wide range of areas, such as chemistry [11–14], finance [15, 16], traffic prediction [17], machine learning [18–21], entanglement detection [22–24], and differential equations [25–27].

Despite the great advances made in recent years, implementing VQA remains a challenge. One of the main drawbacks is the large number of measurements required to evaluate the objective function. Such evaluation can be done by decomposing the Hamiltonian on the basis of tensor products of Pauli operators, also called Pauli strings, and then measuring each of these terms independently. Using this approach, the number of measurements needed to evaluate the objective function is equal to the Pauli terms of the Hamiltonian, which scales as N^4 for typical instances such as second-quantized chemical Hamiltonians on N qubits [28, 29]. The preparation and measurement of each circuit require a non-negligible amount of time and resources; thus, decreasing the number of circuits is essential to speed up VQAs and to reach realistic applications of quantum computing.

Several methods have been proposed to reduce the total number of measurements required to efficiently measure observables, such as classical shadows [30, 31], quantum tomography [32–35], machine learning [36, 37], sampling methods [38, 39], hamiltonian moments [40], and adaptive protocols [41]. Other proposals are grouping methods, which exploit the commutative relation between the Pauli strings to make groups that can be measured simultaneously, reducing the total number of required experiments. The most widely known approach is the grouping with tensor product basis (TPB) [12–14, 42, 43], which uses qubit-wise commutativity and does not require entanglement. Another alternative is to use entangled measurements, which further reduces the number of measurements compared to TPB [44–47]. However, this last approach assumes unlimited entanglement resources and is unsuitable for NISQ devices. There is a midpoint between the two extreme alternatives of assuming limitless entanglement or none at all: using measurements whose entanglement requirements are within the limits of NISQ devices. There are several works in this line [48–51], nevertheless, none of these techniques takes into account the connectivity of the particular quantum processor where the algorithm is run.

In this article, we address the problem of grouping Pauli strings with entangled measurements (EMs), but only between physically connected qubits, that is, hardware-efficient entangled measurements (HEEMs). Given a set of Pauli strings and the processor connectivity, there is a vast number of possible HEEMs, some of which will be more effective than others. Finding the optimal HEEM requires, among other things, finding the optimal mapping between the theoretical qubits of the algorithm and the physical qubits of the device.

Typeset by REVTeX

We have named this issue the *processor embedding* problem. We propose heuristic algorithms to solve the *processor embedding* problem and to perform the grouping with HEEMs. We run VQE with HEEM to estimate the ground-state energy of the H_2O molecule using both classical simulators and quantum hardware. The proposed method reduces the number of measurements due to the entangled measurements and avoids long-range qubit interactions involving noisy, deep circuits, thanks to the hardware-efficient approach.

II. HARDWARE-EFFICIENT GROUPING

A general N-qubit Hamiltonian can be expanded in terms of Pauli strings as

$$H = \sum_\alpha h_\alpha P_\alpha, \tag{1}$$

where P_α are the tensor products of the identity and Pauli operators (I, X, Y, Z), and $h_\alpha \in \mathbb{R}$ are the coefficients of each Pauli string. The standard routine to compute the expected value of H consists of measuring each Pauli string P_i with a TPB measurement, which are the tensor products of the bases of eigenstates of the Pauli operators $\{\mathcal{X}, \mathcal{Y}, \mathcal{Z}\}$, with $\mathcal{X} = \{|0\rangle \pm |1\rangle\}$, $\mathcal{Y} = \{|0\rangle \pm i|1\rangle\}$ and $\mathcal{Z} = \{|0\rangle, |1\rangle\}$.

Multiple Pauli strings can be evaluated simultaneously with a single TPB. In this case, we say that they are *compatible* with that TPB. For example, the 3-qubit Pauli strings XIZ and XYZ are compatible with TPB $\mathcal{X} \otimes \mathcal{Y} \otimes \mathcal{Z}$. This property allows us to define a grouping method [42], which consists of searching for the smallest set of TPB that can be used to evaluate the expected value of H. Finding the best TPB grouping for a given Hamiltonian is equivalent to finding the best coloring for its Pauli graph, which is NP-Complete [43, 52].

One can go beyond the TPB grouping and measure the expected value of H with fewer groups thanks to EM, that is, measuring after an entangling operation. In [48] the authors proposed a heuristic algorithm to construct groups using EM between pairs of qubits (see Appendix A). The EM grouping proved to be more efficient than the TPB grouping, significantly reducing the number of measurements and the uncertainty in evaluating the observables. One drawback of this approach is that it neglects the error of entangling gates in NISQ devices, and hence, it might worsen the results in some scenarios. This is particularly pronounced when performing entangling operations between nonphysically connected qubits, since these require the usage of mediating qubit—and hence additional entangling gates—that increase the depth and thus the error of the circuits.

As a solution to this problem, we propose three heuristic algorithms to group Pauli strings with HEEM (see Appendix B). These algorithms introduce two important improvements to the algorithm proposed by Hamamura *et al.* [48]. First, the algorithms check whether two Pauli strings are simultaneously measurable by an EM given the connectivity of a device, isolating from all possible EM only those that are hardware efficient. Second, since the groups obtained by our algorithms or by Hamamura's algorithms depend on the orders followed by several loops, we introduce three algorithms to choose these orders. This is explained in the next section.

A. The processor embedding problem

Multiple factors determine the performance of our grouping algorithms, for instance, the orders that its loops follow to run through the qubits, the assignable measurements, and the Pauli strings (here we refer to the orders of Subroutine 2 in Appendix B 2 a). These orders can be optimized considering the *processor embedding* problem: the problem of finding the optimal way to map the theoretical qubits of the Hamiltonian into the physical qubits of the device. In this section, we introduce a heuristic approach to solve this problem, allowing us to optimize the order of the loops in the qubits and in the assignable measurements.

Given the Hamiltonian shown in Eq. (1), we define its *compatibility matrix* C as an $N \times N$ symmetric matrix whose non-diagonal entries C_{ij} are equal to the number of compatible entangled measurements (see Table II in the Appendix A) involving qubits i and j. Diagonal entries of C are irrelevant for the *processor embedding* problem.

The number of compatible entangled measurements that involve qubits (i, j) is defined in the following way. The factors i and j of each Pauli string P_α can be regarded as a set of Pauli sub-strings of 2 qubits, $\{P_\alpha^{ij}\}$, and the number of compatible entangled measurements that involve qubits (i, j) is the number of pairs of Pauli sub-strings $\{P_\alpha^{ij}\}$ that are compatible through an entangled measurement. Thus, the compatibility matrix encapsulates how many entangled measurements can be established between two qubits. As an example, consider the following set of Pauli strings

$$P_\alpha \begin{cases} XXZ \\ YYZ \\ YZZ \end{cases} \Rightarrow P_\alpha^{01} \begin{cases} XX \\ YY \\ YZ \end{cases} \Rightarrow C_{01} = C_{10} = 2.$$

In this example $C_{10} = C_{01} = 2$ because XX and YY can be measured simultaneously with a Bell measurement and XX and YZ with an Ω^X measurement (see Appendix A 2). However, YY and YZ cannot be measured simultaneously, i.e., $[YY, YZ] \neq 0$.

With the compatibility matrix, we can tackle the *processor embedding* problem by maximizing the number of potential entangled measurements between neighboring qubits. We formulate this problem in terms of graphs, as shown in Fig. 1. Let $G' := (V', E', C)$ be the weighted graph defined by the vertices $V' := \{0, \dots, N-1\}$, where N is the number of theoretical qubits, the edges $E' := V' \times V'$ and the weights C_{ij}. In this way, G' represents the compatibility between the theoretical qubits
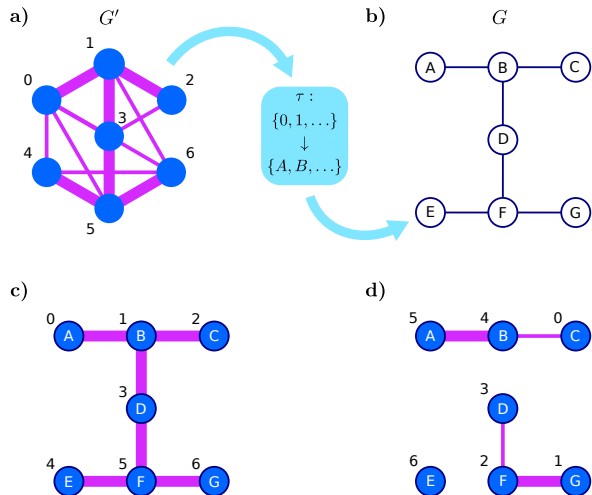
Figure 1. Schematic representation of the *processor embedding* problem. **a)** Compatibility graph of theoretical qubits, G'. Given a set of Pauli strings, the width of the edges $\{i,j\}$ of this graph represents the number of entangled measurements that can be established between qubits i and j, C_{ij}. **b)** Connectivity graph of the physical qubits of the *ibmq_jakarta* device, G. **c)** and **d)** are good and bad *processor embedding* maps τ, respectively.

(Fig. 1 a)). Let $G := (V, E)$ be the graph defined by the vertices $V := \{0, \ldots, M-1\}$, with $M \geq N$ the number of physical qubits and the edges $E := \{(i,j) \in V \times V :$ qubits i and j that are physically connected$\}$. In this way, G represents the topology of the chip (Fig. 1 b)). The objective is to find the map $\tau : \{0, \ldots, N-1\} \rightarrow \{0, \ldots, M-1\}$ that maximizes

$$\omega(\tau) = \sum_{(\tau(i),\tau(j)) \in E} C_{ij}. \tag{2}$$

Note that $\omega(\tau)$ is the total number of compatibilities between theoretical qubits once they are mapped to physical qubits through τ. For the example shown in Fig. 1 a), where the thin edges correspond to $C_{ij} = 1$ and the wide ones to $C_{ij} = 2$, the optimal *processor embedding* map (Fig. 1 c)) results in $\omega(\tau) = 12$, while a bad mapping (Fig. 1 d)) gives a lower value $\omega(\tau) = 6$.

The problem of finding the best τ is what we refer to by the *processor embedding* problem, which we know is NP-Hard, and hence there is no efficient algorithm for the general case. There are several ways to prove that the *processor embedding* problem is NP-Hard. For example, the max-cliqué problem, which is itself NP-Hard [53], can be regarded as a simplification of the *processor embedding* problem so that all nondiagonal entries of C are positive and equal. Additionally, the k-densest sub-graph problem is NP-Complete and can also be seen as a particular instance of the *processor embedding* problem [54, 55]. Efficient algorithms could be found for the

physically relevant instances of the *processor embedding* problem if one finds some kind of structure within them. For now, we propose two heuristic algorithms (see Appendix B 2 b) to construct the map τ. The first is the *order-disconnected* map (Algorithm 2 and Subroutine 3). Here, each pair of theoretical qubits (i,j) with the highest entries C_{ij} is assigned to physically connected qubits. The second alternative is the *order-connected* map (Algorithm 3 and Subroutine 4). This algorithm does the same as the previous but also ensures that the graph $\tau(G')$ is a connected sub-graph of G. The third alternative is the *naive* map (Algorithm 1). This is simply the trivial map $\tau(i) = i$ for $i \in \{0, \ldots, N-1\}$, and it allows us to benchmark our *order-connected* and *order-disconnected* maps. An schematic flow chart of the different algorithms is shown in Fig. 2.

Once the map τ has been chosen, one can propose an order for the loops of our grouping algorithm (Subroutine 2), both in qubits and measurements. For this purpose, we introduce the τ-compatibility matrix C^τ

$$C_{ij}^\tau := \begin{cases} C_{ij} & \text{if } (\tau(i),\tau(j)) \in E \\ 0 & \text{if } (\tau(i),\tau(j)) \notin E. \end{cases} \tag{3}$$

Let CQ^τ be the N-vector whose i entry is given by

$$CQ_i^\tau := C\mathcal{X}_i + C\mathcal{Y}_i + C\mathcal{Z}_i + \sum_{j \neq i} C_{ij}^\tau, \tag{4}$$

where $C\mathcal{X}_i$, $C\mathcal{Y}_i$, and $C\mathcal{Z}_i$ are the numbers of compatibilities involving the qubit i through measurements $\mathcal{X}$, $\mathcal{Y}$, and $\mathcal{Z}$, respectively. In this way, CQ_i^τ is the number of compatibilities involving the qubit i, since τ has been chosen as the embedding map of the processor. Now, we make the natural choice of running through the qubits in descending order of CQ^τ.
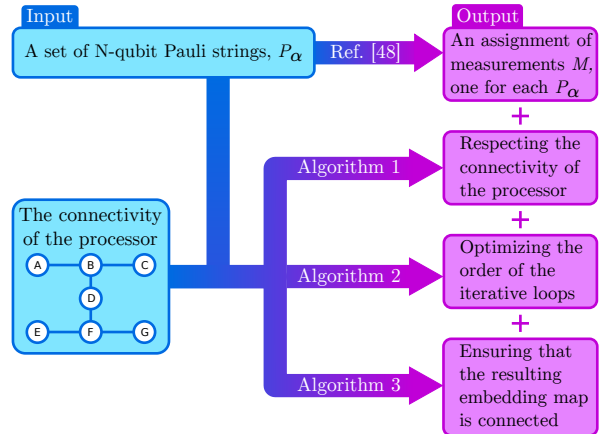

Figure 2. Flow chart of the different algorithms used in this work. The algorithm shown in [48] is referred in the main text as EM. The Algorithms 1, 2 and 3 are named *naive*, *order-disconnected*, and *order-connected*, respectively. These three algorithms are gathered under HEEM grouping.
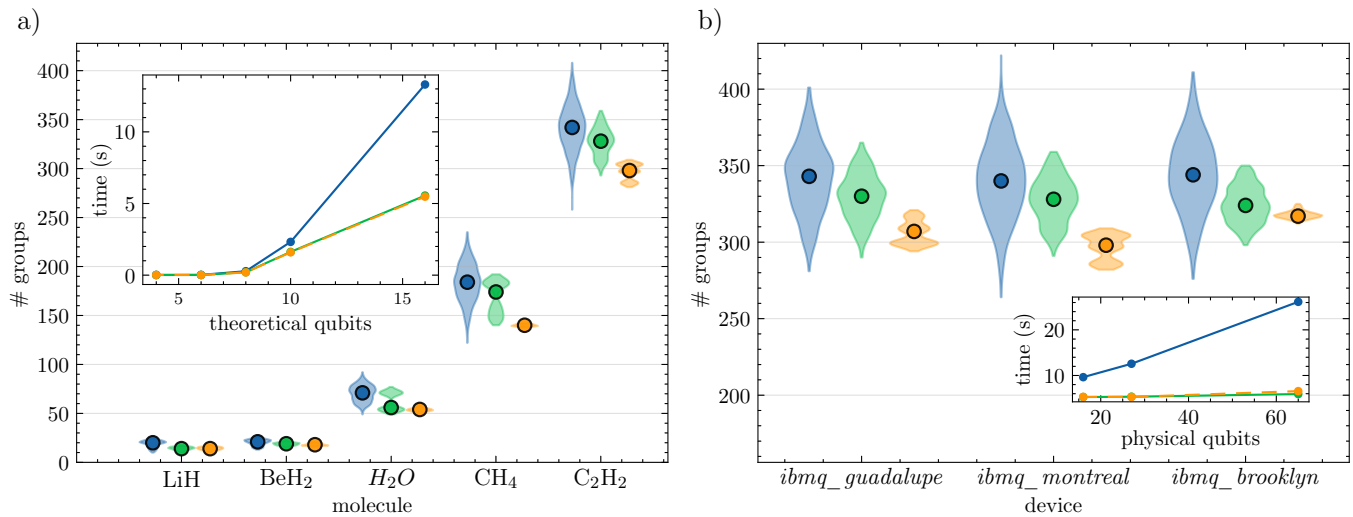
Figure 3. Results for grouping algorithms. The violins correspond to the distributions on the number of groups after running a Monte Carlo method over the order of the Pauli strings and the order of the theoretical qubits for different **a)** molecules, and **b)** devices. Blue violins correspond to the *naive* algorithm (see Appendix B Algorithm 1), green violins correspond to the *order-disconnected* algorithm (Algorithm 2), and orange violins to the *order-connected* algorithm (Algorithm 3). The colored dots represent the mean value in the number of groups in each case. The insets show the average execution time of the grouping algorithm. In **a)** the algorithms are grouped according to HEEM with the connectivity of *ibmq_montreal*. In **b)** the number of groups corresponds to the C_2H_2 molecule.

Let CM^τ be the 9-vector given by

$$CM^\tau := (CM_{\mathcal{X}}, CM_{\mathcal{Y}}, CM_{\mathcal{Z}}, CM^\tau_{\text{Bell}},$$
$$CM^\tau_{\Omega^X}, CM^\tau_{\Omega^Y}, CM^\tau_{\Omega^Z}, CM^\tau_{\chi}, CM^\tau_{\tilde{\chi}}), \quad (5)$$

where CM^τ_{Bell} is the number of compatibilities due to Bell measurement once τ has been chosen as the *processor embedding* map. $CM^\tau_{\Omega^X}$, $CM^\tau_{\Omega^Y}$, $C^\tau_{\Omega^Z}$, CM^τ_{χ} and $CM^\tau_{\tilde{\chi}}$ are defined analogously for the other entangled measurements (see Appendix A). $CM_{\mathcal{X}}$, $CM_{\mathcal{Y}}$ and $CM_{\mathcal{Z}}$ are the numbers of compatibilities by the measurements $\mathcal{X}$, $\mathcal{Y}$ and $\mathcal{Z}$ (these numbers do not depend on τ because these measurements are separable). Now, similarly to what we did for the order of the qubits, we choose to run through the measurements in descending order of CM^τ.

An additional optimization that remains is the iterative order in which the algorithms visit each of the Pauli strings. As the *processor embedding* map does not provide means to optimize this order, we adopt the traditional approach. This consists of building the Pauli graph and visiting the Pauli strings in descending order with respect to their degree in this graph, similar to what is done in the largest degree-first coloring algorithm (LDFC) [56].

Finally, we must emphasize that the *processor embedding* problem is not only relevant in choosing the best orders for the loops of our grouping algorithm (Subroutine 2), but also affects the number of groups obtained. For example, suppose that we have $N = M = 3$, two Pauli strings XXZ and ZXX and $E = \{(0,1),(1,2)\}$. If we choose $\tau: 0 \to 0, 1 \to 1, 2 \to 2$ we need to measure both strings separately, but choosing $\tilde{\tau}: 0 \to 1, 1 \to 0, 2 \to 2$ a single measurement suffices. In addition, the

processor embedding problem is crucial to reduce the error caused by the gates on NISQ computers, as a good choice of τ would also reduce the number of CNOTs used in the algorithm that precedes the measurement, which is a relevant task in the physical layout problem [57]. In fact, if we had to apply a theoretical CNOT in our algorithm between qubits i and j that are mapped to physical qubits whose distance on the chip is D, then in practice we would need to apply $\mathcal{O}(D)$ CNOTs between physically connected qubits, so reducing D using an appropriate τ is essential. Also, in a real device, the accuracy of the CNOT gates between connected qubits will vary depending on the chosen qubits. If one takes into account these other tasks and not only the grouping, the weights of G' should vary: it should depend not only on the compatibility matrix, but also on those other features. This extension that takes into account both the quantum algorithm and the measurement scheme will be addressed in future works.

III. RESULTS

We begin by comparing the three different HEEM grouping algorithms that we have proposed: the *naive*, the *order-disconnected*, and the *order-connected* methods (Algorithms 1, 2 and 3). The reasons that led us to develop the *order-connected* and *order-disconnected* methods were improving the *naive* method and reducing its dependence on the order followed in its loops. To verify the enhancements, we studied the performance of the three algorithms in several molecular Hamiltonians.

	Qubits	No grouping	Grouping			CNOTs		Relative error (%)		
			TPB	EM	HEEM	EM	HEEM	TPB	EM	HEEM
H_2	2	5	2	2	2	1	1	2.9 ± 0.3	2.5 ± 0.2	2.4 ± 0.3
LiH	4	100	25	11	10	8	8	0 ± 1	0 ± 1	0.2 ± 0.9
BeH_2	6	95	24	15	13	74	18	-	-	-
H_2O	8	444	93	51	47	563	80	7 ± 1	9 ± 1	3.2 ± 0.8
CH_4	10	1181	246	113	117	2677	224	15 ± 2	11 ± 2	6 ± 3
C_2H_2	16	1884	457	189	258	8969	433	16 ± 2	19 ± 2	12 ± 3
CH_3OH	22	9257	2225	682	1503	9830	2770	25 ± 5	31 ± 3	20 ± 4
C_2H_6	26	8919	2069	758	1529	55809	2873	34 ± 7	40 ± 3	22 ± 4

Table I. Number of groups, number of CNOTs, and relative error of the energy evaluation for some molecules using different grouping strategies. TPB groups have been obtained using the LDFC algorithm. EM groups are those proposed by [48]. HEEM groups have been obtained using the best performing method among Algorithms 1, 2 and 3 in each case, and assuming that the physical qubits have the connectivity of the device *ibmq_montreal*. The relative error refers to the error in estimating energy for an initial state $|0\rangle^{\otimes N}$. Simulations are carried out considering the noise model of the device *ibmq_montreal*, and the uncertainty in relative error is given by the standard deviation for a total of 25 simulations. Each simulation has a total of 2^{14} shots evenly distributed across all measurements in each grouping. We omit the relative error of BeH_2 because the expected energy is equal to zero.

They were constructed from their fermionic counterpart on STO3G basis [58] with the parity map [59]. In addition, we freeze the core and remove the unoccupied orbital of the molecules to reduce their number of qubits (for more details, see [60]).

Despite the optimizations implemented in the algorithms, their effectiveness still depends on the initial order of the theoretical qubits and the Pauli strings in the Hamiltonian. To address this dependency, we employed a Monte Carlo method to evaluate the average performance of the three algorithms. For a given Hamiltonian, we consider the set of all permutations of the order of the theoretical qubits and of the Pauli strings of that specific Hamiltonian. Each element of this set represents a potential input for the grouping algorithms. Monte Carlo is executed by randomly selecting elements on this set. Figure 3 a) shows that the *order-connected* and *order-disconnected* algorithms are better on average and dispersion than the *naive* algorithm, reducing their dependence on the iterative orders. The *order-connected* algorithm is the one that performs best by having the lowest average value and the smallest dispersion. Notice that the absolute minimum number of groups is obtained with the *naive* algorithm. This feature can be used to find the best grouping in small-scale Hamiltonians by running the Monte Carlo method, as we explained before. However, this procedure is impractical for large-scale Hamiltonians, where only a small fraction of all potential inputs can be explored. Thus, in this case, the *order-connected* scheme is the best alternative because its performance is concentrated around the smallest average number of groupings.

In terms of execution time, the *order-connected* and *order-disconnected* algorithms are much faster than the *naive* one. The time complexity of the *order-connected* and *disconnected* algorithms is $\mathcal{O}(N^5)$ on the number

of qubits in the Hamiltonian, versus the time complexity $\mathcal{O}(N^{5.5})$ needed for a *naive* grouping. Figure 3 b) compares the performance of these three algorithms with three different processor architectures. It shows that the execution time of the non-naive algorithms does not depend on the architecture in contrast to the *naive* one. This result is important for the practical application of the proposed algorithms on quantum devices with a large number of qubits. The opposite situation occurs for the grouping configurations: the *naive* algorithm does not differentiate between processors, while the other two do. Thus, only the non-naive algorithms have the desired behavior with respect to different architectures: they are always quickly executed and can take advantage of the topology of the chip.

Let us now confront our grouping methods with the previous ones. Table I shows the number of groups obtained from Pauli strings of molecules of sizes between 2 and 26 qubits using different methods: TPB grouping [42], EM grouping [48], and HEEM grouping with the connectivity of the quantum device *ibmq_montreal*. Analogously to the results of Fig. 3, the Hamiltonians were constructed with the parity map on the basis of STO3G, freezing the core and removing unoccupied orbitals [60]. The results of the HEEM grouping correspond to the minimum number of groups obtained with our Monte Carlo study. Table I also includes the number of CNOTs required to measure the groups. On the one hand, the HEEM grouping significantly outperforms TPB in terms of the number of groups. However, the EM grouping has a smaller number of groups than the HEEM but uses a number of CNOTs that grows much more rapidly with the size of the molecules. The number of groups with HEEM scales as $\mathcal{O}(N^{2.45})$ with the number of qubits. Furthermore, HEEM circuits involve only one layer of entangling gates, while the depth of EM

circuits depends on the distance between the qubits to be connected. All the scalings shown here were obtained by fitting the numerical results obtained with each grouping. More details on this topic, together with the asymptotic scaling values for other magnitudes, can be found in the Appendix C.

Both the number of CNOTs, which translates into experimental error, and number of groups, which translates into statistical error for a fixed total number of shots, are relevant for the final accuracy of the method. This means that the results discussed so far do not completely conclude which is the best grouping method. In Table I we show the relative error for the energy evaluation of the state $|\psi\rangle = |0\rangle^{\otimes N}$ for different molecule Hamiltonians and grouping methods. We consider this state because, having no entanglement, it is prepared with high accuracy. This implies that in the simulations only the error due to the energy evaluation appears. We set the total number of shots at 2^{14} for all molecules and methods, and these shots are divided equally between all circuits of each scenario. The relative error is computed as

$$\text{Relative error} = |(E_{\text{exact}} - E_{\text{simulation}})/E_{\text{exact}}|, \quad (6)$$

where the exact energy is obtained as $E_{\text{exact}} = \langle\psi| H |\psi\rangle$, while $E_{\text{simulation}}$ is calculated considering the noise model of the device $ibmq\_montreal$. This simulation is repeated 25 times and from there we compute the average relative error shown in Table I. The uncertainty in this relative error is defined by the standard deviation between the different runs.

With TPB grouping, no entangled measurements are performed, and hence there are no errors due to two-qubit gates. However, the total number of circuits to compute the energy is the largest, and consequently, TPB has the lowest number of shots per circuit and the highest statistical error. With EM, the scenario is the opposite; this method has the lowest statistical error, but the highest experimental error due to the large number of CNOT gates used to connect distant qubits. HEEM takes the best of both methods, as it uses fewer groups than TPB, reducing the statistical error, and it does not connect distant qubits, reducing the experimental error. Because of this, as can be seen in Table I, HEEM provides the lowest relative error for all the studied molecules.

Finally, to study the performance of the HEEM grouping in a practical scenario, we have run noisy simulations and a NISQ experiment of the VQE for the H_2O molecule at its bond distance. The Hamiltonian was constructed by the parity map from the fermionic Hamiltonian on STO3G basis. In addition, we freeze the core and remove unoccupied orbitals of the H_2O molecule to reduce the number of theoretical qubits to 8, so that the minimum energy is given by -13.9 Ha. Figure 4 a) shows the mean energy per iteration of 240 instances of noisy VQE simulations using TPB, EM and HEEM. They were carried out using the noise model and the connectivity of $ibmq\_montreal$, provided by IBM-Q [61]. The variational ansatz is composed of 2 layers of local gates and 1 layer of CNOT between the physically connected qubits (see Appendix D). The minimum energy attainable with this ansatz is -12.3 Ha. The groups used for HEEM correspond to the best groups obtained among the three proposed algorithms. We can see that the better performance grouping scheme is HEEM, achieving energies below -12.0 Ha after 200 iterations, while neither TPB nor EM reaches this value in 300 iterations. The inset of Fig. 4 a) shows the mean energy in terms of the number of circuits. We can see that the HEEM grouping achieves energy below -12.0 Ha with about 1200 circuits, while TPB and EM only attain -11.6 Ha and -11.2 Ha, respectively. This means that HEEM allows us to obtain the same results as the previous approaches, but with fewer circuits.

We also performed an experimental implementation of the VQE for the H_2O molecule with TPB, EM, and HEEM in an IBM quantum device. The experiment was carried out on the first 8 qubits of the device $ibmq\_guadalupe$. This is a quantum device of 16 qubits, 32 of quantum volume, and 1.245×10^{-2} of average error in the CNOT gates. The VQE was implemented with the same configuration as the simulations, that is, the same Hamiltonian, ansatz, and classical optimizer. Figure 4 b) shows the results of the experiments. We can see that TPB and HEEM outperform EM in precision at the same number of iterations, while HEEM provides a slight advantage over TPB. The realization of TPB was very favorable, being approximately one standard deviation below the mean shown in the simulations. Although HEEM performs similarly to TPB in the number of iterations, the scenario differs in the number of circuits. The inset of Fig. 4 b), which are results of experimental VQEs in terms of the number of circuits, shows that HEEM overcomes both TPB and EM in the number of circuits, achieving better energies at the same number of circuits. Grouping with HEEM also provides an advantage over TPB and EM at runtime. The time for each experiment was 3.8 hours with TPB, 4.4 hours with EM, and 2.7 hours with HEEM. Thus, HEEM provides a speed-up with respect to the other grouping approaches in real hardware.

IV. CONCLUSIONS AND OUTLOOK

In the NISQ era, in order to achieve a quantum advantage with variational algorithms, we need efficient techniques to evaluate the expected values of Hamiltonians. In this article, we introduce the Hardware Efficient Entangled Measurements (HEEMs). They allow one to evaluate the expected value of a Hamiltonian by simultaneously measuring groups of Pauli strings and only employing entangled gates between qubits that are physically connected on the device. This makes HEEM an efficient and noise-robust alternative for evaluating expected values and speeding up variational algorithms in
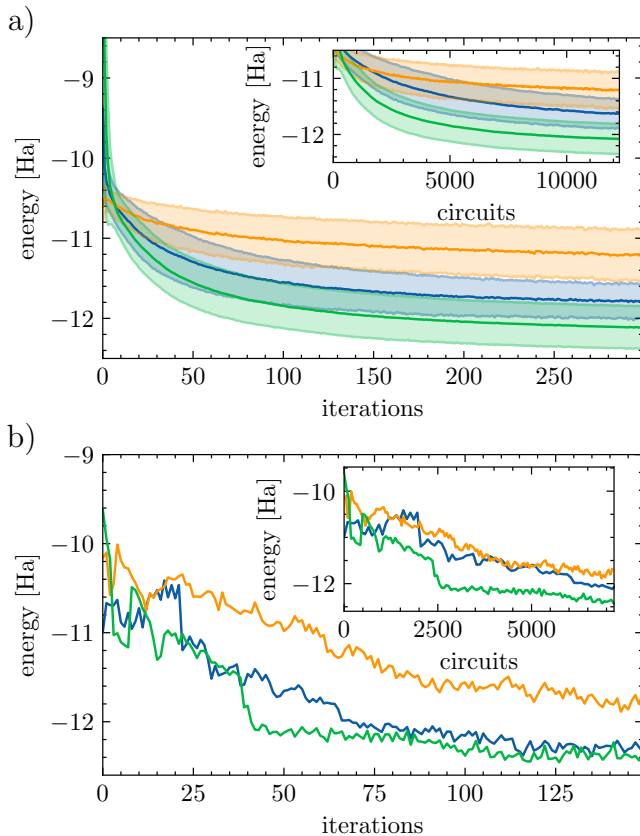
a)

b)

Figure 4. Simulation **a)**, and experimental implementation **b)**, of the VQE for the 8-qubits H_2O Hamiltonian, with a distance of $d = 0.96$ Å between hydrogen and oxygen atoms, for TPB in blue, EM in orange, and HEEM in green lines. The simulations were performed considering the basis gate, noise model, and connectivity of *ibmq_montreal*, while the real experiment is performed in *ibmq_guadalupe*. The classical optimizer used was SPSA with 300 iterations for the simulations and 150 iterations for the experiment. There are 2^{14} shots per circuit in both cases. **a)** Each point represents 240 independent instances of VQE where the solid lines correspond to the mean energy and the shaded region to the standard deviation. In both panels, the insets show the mean energy versus the number of circuits needed to measure the Hamiltonian.

NISQ devices. Since there are multiple ways to group a set of Pauli strings with HEEM, we have introduced three algorithms to carry out the grouping. The first is the *naive* HEEM, which does not optimize the order of the grouping loops. The other two are *order-connected* HEEM and the *order-disconnected* HEEM. They optimize the grouping loops by using the *processor embedding* problem, the problem of finding the optimal map of the theoretical qubits into the physical qubits.

We compare our methods and conclude that the *order-connected* and *order-disconnected* algorithms outperform the *naive* algorithm on the average number of groups and the time execution. Our methods improve previous works, using fewer groups than TPB [42], with a slight

increase in the number of required CNOTs. Furthermore, HEEM requires fewer CNOTs than EM [48], using only a few more groups. Note that the number of groups quantifies the statistical error, while the number of CNOTs quantifies the experimental error. Then, taking both sources of errors into account, HEEM outperforms both TPB and EM. We have shown, with noisy simulations and experiments, that HEEM achieves better results with the same number of circuits. In addition, HEEM is faster than TPB and EM to be implemented both in an experiment and in a simulation for practical scenarios. HEEM grouping can be useful not only in variational algorithms, but also in any task that requires the evaluation of several Pauli strings, such as full quantum tomography [62, 63], compressed sensing [64, 65], reduced density matrix tomography [33, 34], classical shadows [30, 31], or direct fidelity estimation [66, 67].

Several extensions can be incorporated into our proposal. The grouping of Pauli strings by HEEM can be improved by considering entanglement between a larger number of qubits. This would allow for further reduction of the number of groups at the expense of more entanglement resources. The method can also be refined by including the error of the CNOT gates of the chip as weights in the graph that represents the connectivity of the chip. This would produce better results by ensuring that the majority of entangling operations are performed over pairs of qubits with the lowest CNOT gate errors. Given that the number of groups and CNOTs are proxies for the error of the measurement, we can explore the behavior of HEEM using more elaborate metrics, such as those proposed in [49]. In this article, it is suggested to sort the Pauli strings according to their weights in the Hamiltonian, which could improve HEEM in practical scenarios. The combination of HEEM with other estimation protocols could provide an even more accurate estimate of observables on current devices, such as error mitigation techniques [68–70] or adaptive schemes [41].

The code for reproducing the algorithms and figures in this article can be found in [60].

Appendix A: Entangled measurements

We have identified the two-qubit measurements that allow us to measure all maximal sets of compatible Pauli strings of length two. They are displayed in Fig. 5. In the last two entangled measurements, the gate U_2 refers to the following operation in matrix form

$$U_2(\phi, \lambda) = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & -e^{i\lambda} \\ e^{i\phi} & e^{i(\phi+\lambda)} \end{pmatrix}. \tag{A1}$$
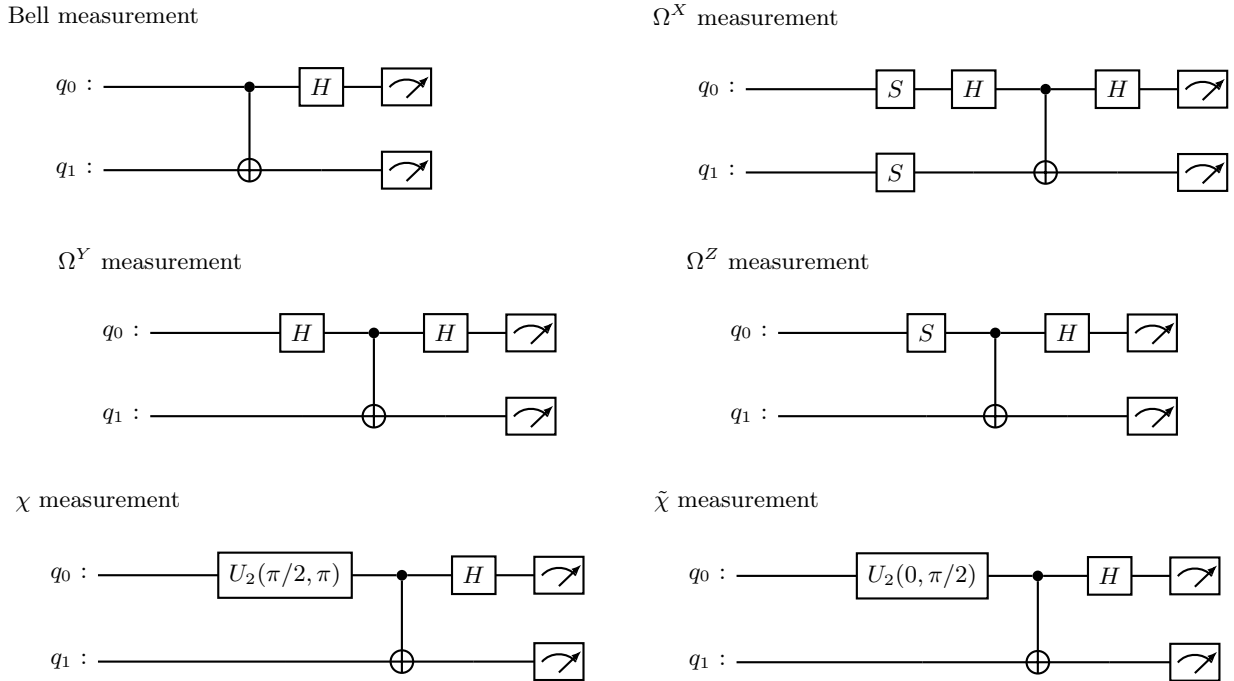
Figure 5. Circuits to implement the entanglement measurements. The gates H and S are Hadamard and $\pi/2$-phase gate, respectively, and the gate U_2 is given by Eq. (A1)

1. Computing the expected values of Hamiltonians with a certain grouping

In this section, we explain how to obtain the expected value of a Hamiltonian once we have grouped its Pauli strings. Let $H = \sum_\alpha h_\alpha P_\alpha$ be a multi-qubit Hamiltonian and suppose that the m first Pauli strings are compatible with a single HEEM measurement into the basis $\mathcal{B} = \mathcal{B}_1 \otimes \cdots \otimes \mathcal{B}_t$, with B_i single- or two-qubit bases with $i = 1, \ldots, t$. This means that the Pauli strings P_α with $\alpha = 1, \ldots, m$ are diagonal on the basis of $\mathcal{B}$. Let be $\vec{W}_P = \text{Diag}_{\mathcal{B}}\{P\}$ the vector with the diagonal entries of the Pauli string P on the basis of $\mathcal{B}$. The elements of $\vec{W}_P$ correspond to the eigenvalues of P, each of them repeated by its multiplicity. Therefore, if $\vec{\mathcal{P}}$ is the probability distribution of a measurement on the basis $\mathcal{B}$, the expected value of the compatible Pauli strings is given by

$$\left\langle \sum_{\alpha=1}^{m} h_\alpha P_\alpha \right\rangle = \left(\sum_{\alpha=1}^{m} h_\alpha \vec{W}_{P_\alpha}, \vec{\mathcal{P}} \right), \tag{A2}$$

where $(\cdot, \cdot)$ is the usual inner product. To illustrate the procedure, we evaluate the energy of a simple Hamiltonian consisting only of two Pauli strings,

$$H = 2IZY + 4ZXZ. \tag{A3}$$

We have to check in Table II on which basis, if possible, these two strings can be measured together. After a careful look, we realize that the operators I and Z of the first qubits are compatible with $\mathcal{Z}$, and that the operators ZY and XZ of the second and third qubits are compatible with $\tilde{\chi}$. The basis $\mathcal{Z}$ allows us to diagonalize the operators $I = + |0\rangle \langle 0| + |1\rangle \langle 1|$, and $Z = + |0\rangle \langle 0| - |1\rangle \langle 1|$, while the basis $\tilde{\chi}$ the 2-qubit operators

$$YX = - |\tilde{\chi}_0\rangle \langle \tilde{\chi}_0| + |\tilde{\chi}_1\rangle \langle \tilde{\chi}_1| + |\tilde{\chi}_2\rangle \langle \tilde{\chi}_2| - |\tilde{\chi}_3\rangle \langle \tilde{\chi}_3|, \tag{A4}$$
$$ZY = + |\tilde{\chi}_0\rangle \langle \tilde{\chi}_0| + |\tilde{\chi}_1\rangle \langle \tilde{\chi}_1| - |\tilde{\chi}_2\rangle \langle \tilde{\chi}_2| - |\tilde{\chi}_3\rangle \langle \tilde{\chi}_3|, \tag{A5}$$
$$XZ = + |\tilde{\chi}_0\rangle \langle \tilde{\chi}_0| - |\tilde{\chi}_1\rangle \langle \tilde{\chi}_1| + |\tilde{\chi}_2\rangle \langle \tilde{\chi}_2| - |\tilde{\chi}_3\rangle \langle \tilde{\chi}_3|. \tag{A6}$$

One last important thing that we need is the explicit expression of the vectors of the $\tilde{\chi}$ basis in the computational basis, to correctly relate the vector of weights with the vector of outcomes

$$|\tilde{\chi}_0\rangle = +i|00\rangle - |01\rangle + i|10\rangle + |11\rangle, \tag{A7}$$
$$|\tilde{\chi}_1\rangle = +|00\rangle + i|01\rangle - |10\rangle + i|11\rangle, \tag{A8}$$
$$|\tilde{\chi}_2\rangle = +i|00\rangle + |01\rangle + i|10\rangle - |11\rangle, \tag{A9}$$
$$|\tilde{\chi}_3\rangle = -|00\rangle + i|01\rangle + |10\rangle + i|11\rangle. \tag{A10}$$

See subsection A 2 for explicit expressions of how to construct every entangled measurement and the diagonal representation of pairs of observables in all of these bases. Now we can tackle the problem of evaluating the energy of Eq. (A3) with a measurement on the basis $\mathcal{B} = \mathcal{Z}_1 \otimes \tilde{\chi}_{2,3}$, where the subscripts refer to the subspace spanned by such qubits. For the Pauli string IZY, we have $\text{Diag}_{\mathcal{Z}}\{I\} = [+1, +1]$ and $\text{Diag}_{\tilde{\chi}}\{ZY\} = [+1, +1, -1, -1]$. The vector $\vec{W}_{IZY}$ is computed as the Kronecker product of the vectors $\text{Diag}_{\mathcal{Z}}$ and $\text{Diag}_{\tilde{\chi}}$. Thus,

$$h_{IZY}\vec{W}_{IZY} = 2 \times [+1, +1] \otimes [+1, +1, -1, -1] = [+2, +2, -2, -2, +2, +2, -2, -2]. \tag{A11}$$

Analogously, for the string ZXZ, we have $\text{Diag}_{\mathcal{Z}}\{Z\} = [+1, -1]$ and $\text{Diag}_{\tilde{\chi}}\{XZ\} = [+1, -1, +1, -1]$. Then

$$h_{ZXZ}\vec{W}_{ZXZ} = 4 \times [+1, -1] \otimes [+1, -1, +1, -1] = [+4, -4, +4, -4, -4, +4, -4, +4]. \tag{A12}$$

Therefore, we have that

$$\sum_{\alpha=1}^{m} h_\alpha \vec{W}_{P_\alpha} = [+6, -2, +2, -6, -2, +6, -6, +2]. \tag{A13}$$

In summary, what our algorithm does once it has identified which Pauli strings can be measured together is to check with which measurement basis it can do so and in which order these basis vectors have to be taken into account. After sorting out all the plus and minus signs, all that is left to do is multiply by the weight in the Hamiltonian h_α, and finally plug in the actual result of the experiment with Eq. (A2).

	XX	YZ	ZY	YY	XZ	ZX	ZZ	XY	YX
XX	—	Ω^X	Ω^X	Bell	✖	✖	Bell	✖	✖
YZ		—	Ω^X	✖	✖	χ	✖	χ	✖
ZY			—	✖	$\tilde{\chi}$	✖	✖	✖	$\tilde{\chi}$
YY				—	Ω^Y	Ω^Y	Bell	✖	✖
XZ					—	Ω^Y	✖	✖	$\tilde{\chi}$
ZX						—	✖	χ	✖
ZZ							—	Ω^Z	Ω^Z
XY								—	Ω^Z
YX									—

Table II. Compatibility relation between two-qubit Pauli strings. The symbol ✖ indicates that the corresponding Pauli strings are not jointly measurable. The other boxes contain the entangled measurements that are compatible with the corresponding Pauli strings. See Appendix A 2 for the definitions of these measurements.

2. Jointly diagonalizable pairs in all entangled bases.

Bell :

$|\Phi_0\rangle = |00\rangle + |11\rangle$

$|\Phi_1\rangle = |00\rangle - |11\rangle$

$|\Phi_2\rangle = |01\rangle + |10\rangle$

$|\Phi_3\rangle = |01\rangle - |10\rangle$

Commuting pairs

$XX = +|\Phi_0\rangle\langle\Phi_0| - |\Phi_1\rangle\langle\Phi_1| + |\Phi_2\rangle\langle\Phi_2| - |\Phi_3\rangle\langle\Phi_3|$

$YY = -|\Phi_0\rangle\langle\Phi_0| + |\Phi_1\rangle\langle\Phi_1| + |\Phi_2\rangle\langle\Phi_2| - |\Phi_3\rangle\langle\Phi_3|$

$ZZ = +|\phi_0\rangle\langle\Phi_0| + |\Phi_1\rangle\langle\Phi_1| - |\Phi_2\rangle\langle\Phi_2| - |\Phi_3\rangle\langle\Phi_3|$

Ω^X :

$|\Omega_0^X\rangle = +|00\rangle - i|01\rangle - i|10\rangle + |11\rangle$

$|\Omega_1^X\rangle = +|00\rangle - i|01\rangle + i|10\rangle - |11\rangle$

$|\Omega_2^X\rangle = +|00\rangle + i|01\rangle - i|10\rangle - |11\rangle$

$|\Omega_3^X\rangle = -|00\rangle - i|01\rangle - i|10\rangle - |11\rangle$

Commuting pairs

$YZ = -|\Omega_0^X\rangle\langle\Omega_0^X| + |\Omega_1^Y\rangle\langle\Omega_1^X| - |\Omega_2^X\rangle\langle\Omega_2^X| + |\Omega_3^X\rangle\langle\Omega_3^X|$

$XX = +|\Omega_0^X\rangle\langle\Omega_0^X| - |\Omega_1^X\rangle\langle\Omega_1^X| - |\Omega_2^X\rangle\langle\Omega_2^X| + |\Omega_3^X\rangle\langle\Omega_3^X|$

$ZY = -|\Omega_0^X\rangle\langle\Omega_0^X| - |\Omega_1^X\rangle\langle\Omega_1^X| + |\Omega_2^X\rangle\langle\Omega_2^X| + |\Omega_3^X\rangle\langle\Omega_3^X|$

Ω^Y :

$|\Omega_0^Y\rangle = +|00\rangle + |01\rangle + |10\rangle - |11\rangle$

$|\Omega_1^Y\rangle = +|00\rangle + |01\rangle - |10\rangle + |11\rangle$

$|\Omega_2^Y\rangle = +|00\rangle - |01\rangle + |10\rangle + |11\rangle$

$|\Omega_3^Y\rangle = -|00\rangle + |01\rangle + |10\rangle + |11\rangle$

Commuting pairs

$XZ = +|\Omega_0^Y\rangle\langle\Omega_0^Y| - |\Omega_1^Y\rangle\langle\Omega_1^Y| + |\Omega_2^Y\rangle\langle\Omega_2^Y| - |\Omega_3^Y\rangle\langle\Omega_3^Y|$

$YY = +|\Omega_0^Y\rangle\langle\Omega_0^Y| - |\Omega_1^Y\rangle\langle\Omega_1^Y| - |\Omega_2^Y\rangle\langle\Omega_2^Y| + |\Omega_3^Y\rangle\langle\Omega_3^Y|$

$ZX = +|\Omega_0^Y\rangle\langle\Omega_0^Y| + |\Omega_1^Y\rangle\langle\Omega_1^Y| - |\Omega_2^Y\rangle\langle\Omega_2^Y| - |\Omega_3^Y\rangle\langle\Omega_3^Y|$

Ω^Z :

$|\Omega_0^Z\rangle = +|00\rangle - i|11\rangle$

$|\Omega_1^Z\rangle = +|00\rangle + i|11\rangle$

$|\Omega_2^Z\rangle = +|01\rangle + i|10\rangle$

$|\Omega_3^Z\rangle = +|00\rangle - i|10\rangle$

Commuting pairs

$XY = -|\Omega_0^Z\rangle\langle\Omega_0^Z| + |\Omega_1^Z\rangle\langle\Omega_1^Z| - |\Omega_2^Z\rangle\langle\Omega_2^Z| + |\Omega_3^Z\rangle\langle\Omega_3^Z|$

$YX = -|\Omega_0^Z\rangle\langle\Omega_0^Z| + |\Omega_1^Z\rangle\langle\Omega_1^Z| + |\Omega_2^Z\rangle\langle\Omega_2^Z| - |\Omega_3^Z\rangle\langle\Omega_3^Z|$

$ZZ = +|\Omega_0^Z\rangle\langle\Omega_0^Z| + |\Omega_1^Z\rangle\langle\Omega_1^Z| - |\Omega_2^Z\rangle\langle\Omega_2^Z| - |\Omega_3^Z\rangle\langle\Omega_3^Z|$

χ :

$|\chi_0\rangle = -|00\rangle + |01\rangle - i|10\rangle + i|11\rangle$

$|\chi_1\rangle = +|00\rangle + |01\rangle + i|10\rangle - i|11\rangle$

$|\chi_2\rangle = +|00\rangle - |01\rangle + i|10\rangle + i|11\rangle$

$|\chi_3\rangle = -|00\rangle + |01\rangle + i|10\rangle + i|11\rangle$

Commuting pairs

$XY = +|\chi_0\rangle\langle\chi_0| - |\chi_1\rangle\langle\chi_1| + |\chi_2\rangle\langle\chi_2| - |\chi_3\rangle\langle\chi_3|$

$YZ = -|\chi_0\rangle\langle\chi_0| + |\chi_1\rangle\langle\chi_1| + |\chi_2\rangle\langle\chi_2| - |\chi_3\rangle\langle\chi_3|$

$ZX = +|\chi_0\rangle\langle\chi_0| + |\chi_1\rangle\langle\chi_1| - |\chi_2\rangle\langle\chi_2| - |\chi_3\rangle\langle\chi_3|$

$\tilde{\chi}$:

$|\tilde{\chi}_0\rangle = +i|00\rangle - |01\rangle + i|10\rangle + |11\rangle$

$|\tilde{\chi}_1\rangle = +|00\rangle + i|01\rangle - |10\rangle + i|11\rangle$

$|\tilde{\chi}_2\rangle = +i|00\rangle + |01\rangle + i|10\rangle - |11\rangle$

$|\tilde{\chi}_3\rangle = -|00\rangle + i|01\rangle + |10\rangle + i|11\rangle$

Commuting pairs

$YX = -|\tilde{\chi}_0\rangle\langle\tilde{\chi}_0| + |\tilde{\chi}_1\rangle\langle\tilde{\chi}_1| + |\tilde{\chi}_2\rangle\langle\tilde{\chi}_2| - |\tilde{\chi}_3\rangle\langle\tilde{\chi}_3|$

$ZY = +|\tilde{\chi}_0\rangle\langle\tilde{\chi}_0| + |\tilde{\chi}_1\rangle\langle\tilde{\chi}_1| - |\tilde{\chi}_2\rangle\langle\tilde{\chi}_2| - |\tilde{\chi}_3\rangle\langle\tilde{\chi}_3|$

$XZ = +|\tilde{\chi}_0\rangle\langle\tilde{\chi}_0| - |\tilde{\chi}_1\rangle\langle\tilde{\chi}_1| + |\tilde{\chi}_2\rangle\langle\tilde{\chi}_2| - |\tilde{\chi}_3\rangle\langle\tilde{\chi}_3|$

Appendix B: Algorithms

In this appendix, we present the heuristic algorithms announced in the main text. All of them can be found in [60]. For the pseudo-code shown in this appendix, we only use variables representing integers, booleans, strings, and lists. Other data structures, such as hash maps or queues, can be used to improve performance. However, we stick to the most common data structures to increase code readability. Items within the lists can be integers or other lists to represent matrices. To index the lists, we use square brackets as $List[i]$, with the indices starting at 0. Lists are created as $List = [item1, item2, \ldots]$. When calling other functions, we use parentheses as FunctionName(parameter1, parameter2, ...). Finally, comments inside the codes start with //. Note that the notation used in this appendix may be different from the one shown in the main text, this is so as to have a notation closer to common programming languages in the pseudo codes, thus facilitating their understanding.

1. Grouping algorithms

All our grouping algorithms (*naive*, *order-connected*, and *order-disconnected*) take into account the connectivity of the chip, but differ in the way theoretical qubits are mapped onto physical qubits. The *naive* algorithm (Algorithm 1) maps the i-th theoretical qubit to the i-th physical qubit (i.e., it chooses the trivial *processor embedding* map $\tau[i] = i$). The *order-disconnected* (Algorithm 2) uses a subroutine that generates a *processor embedding* map τ that maps the theoretical qubits to the physical ones seeking a high number of measurement compatibilities. The *order-connected* (Algorithm 3) does the same as the *order-disconnected*, but with the additional requirement that the graph G' of theoretical qubits is connected once it is mapped to the graph G of physical qubits through τ, i.e., $\tau[G']$ is a connected subgraph of G (which can always be done if G is connected and has at least the same number of vertices as G'). This requirement aims to ensure that all connections of the chip are useful to measure, as explained in Figures 8 and 9.

The core of our algorithms is Subroutine 1, which takes the following inputs: a set of Pauli strings, a *processor embedding* map τ, and the order in which the subroutine runs over the qubits and measurements. All algorithms use Subroutine 7 to decide the orders in which Subroutine 1 runs over the qubits and measurements. The *naive* algorithm does that assuming the trivial *processor embedding* map $\tau[i] = i$, while the *order-disconnected* algorithm uses Subroutine 3 to design τ and the *order-connected* uses Subroutine 4 for the same purpose.

Finally, we would like to remark that despite that Algorithm 3 may look like an arbitrary modification of Algorithm 2, the results show that performance improves when imposing that $\tau[G']$ is connected (see Figure 3 of the main text or Figures 8 and 9).

Algorithm 1: Naive grouping

Input: n Pauli strings PS of N qubits and chip's connectivity G
1 Define the trivial map $\tau[i] = i$
2 CM^τ, CQ^τ = Orders(PS, τ, G)   // (Subroutine 7) CM^τ encodes the measurements and CQ^τ the qubits
3 M, Gr = Grouping(PS, τ, G, CM^τ, CQ^τ)   // (Subroutine 1) M encodes the measurements and Gr the groups
4 **return** M, Gr

Algorithm 2: Order-disconnected grouping

Input: n Pauli strings PS of N qubits and chip's connectivity G
1 τ = DisonnectedMap(PS, G)   // Subroutine 3
2 CM^τ, CQ^τ = Order(PS, τ, G)   // (Subroutine 7) CM^τ encodes the measurements and CQ^τ the qubits
3 M, Gr = Grouping(PS, τ, G, CM^τ, CQ^τ)   // (Subroutine 1) M encodes the measurements and Gr the groups
4 **return** M, Gr

Algorithm 3: Order-connected grouping

Input: n Pauli strings PS of N qubits and chip's connectivity G
1 τ = ConnectedMap(PS, G)   // Subroutine 4
2 CM^τ, CQ^τ = Orders(PS, τ, G)   // (Subroutine 7) CM^τ encodes the measurements and CQ^τ the qubits
3 M, Gr = Grouping(PS, τ, G, CM^τ, CQ^τ)   // (Subroutine 1) M encodes the measurements and Gr the groups
4 **return** M, Gr

---

2. Subroutines

In this subsection, we present the subroutines used in Algorithms 1, 2 and 3, and provide examples that illustrate how they work.

a. Grouping taking chip's connectivity into account

Herewith we describe the core subroutine of our grouping algorithms, which is inspired by the algorithm of [48]. Our subroutine allows for a non-trivial *processor embedding* map and optimizes the orders in which the algorithm runs through qubits and measurements, while the algorithm in [48] does not.

Subroutine 1: Greedy grouping taking chip's connectivity into account

---

Input: n Pauli strings PS, chip's connectivity G, *processor embedding* map τ, and vectors CM^τ and CQ^τ encoding the orders of measurements and qubits, respectively

1 Build the Pauli graph PG of PS // The elements $PG[i]$ are the Pauli strings
2 Sort PG in descending order according to the nodes degree
3 $M = [\,]$ // Measurements
4 $Gr = [\,]$ // Groups
5 **for** i in $[0, \ldots, n-1]$ **do**
6 $\quad$ **if** i not in Gr **then**
7 $\quad\quad$ $m = [\,]$
8 $\quad\quad$ $gr = [i]$
9 $\quad\quad$ **for** j in $[i+1, \ldots, n-1]$ **do**
10 $\quad\quad\quad$ **if** j not in Gr **then**
11 $\quad\quad\quad\quad$ success, $\tilde{m} = $ CheckCompatibility$(PG[i], PG[j], G, m, \tau, CM^\tau, CQ^\tau)$ // Subroutine 2
12 $\quad\quad\quad\quad$ **if** success **then**
13 $\quad\quad\quad\quad\quad$ $m = \tilde{m}$
14 $\quad\quad\quad\quad\quad$ Append j to gr

15 $\quad\quad$ Let U be the list of qubits not measured in m
16 $\quad\quad$ **for** q in U **do**
17 $\quad\quad\quad$ Let b be the basis compatible with the Pauli operator at position q in $PG[i]$
18 $\quad\quad\quad$ Append $[b, q]$ to m

19 $\quad\quad$ Append m to M
20 $\quad\quad$ Append gr to Gr
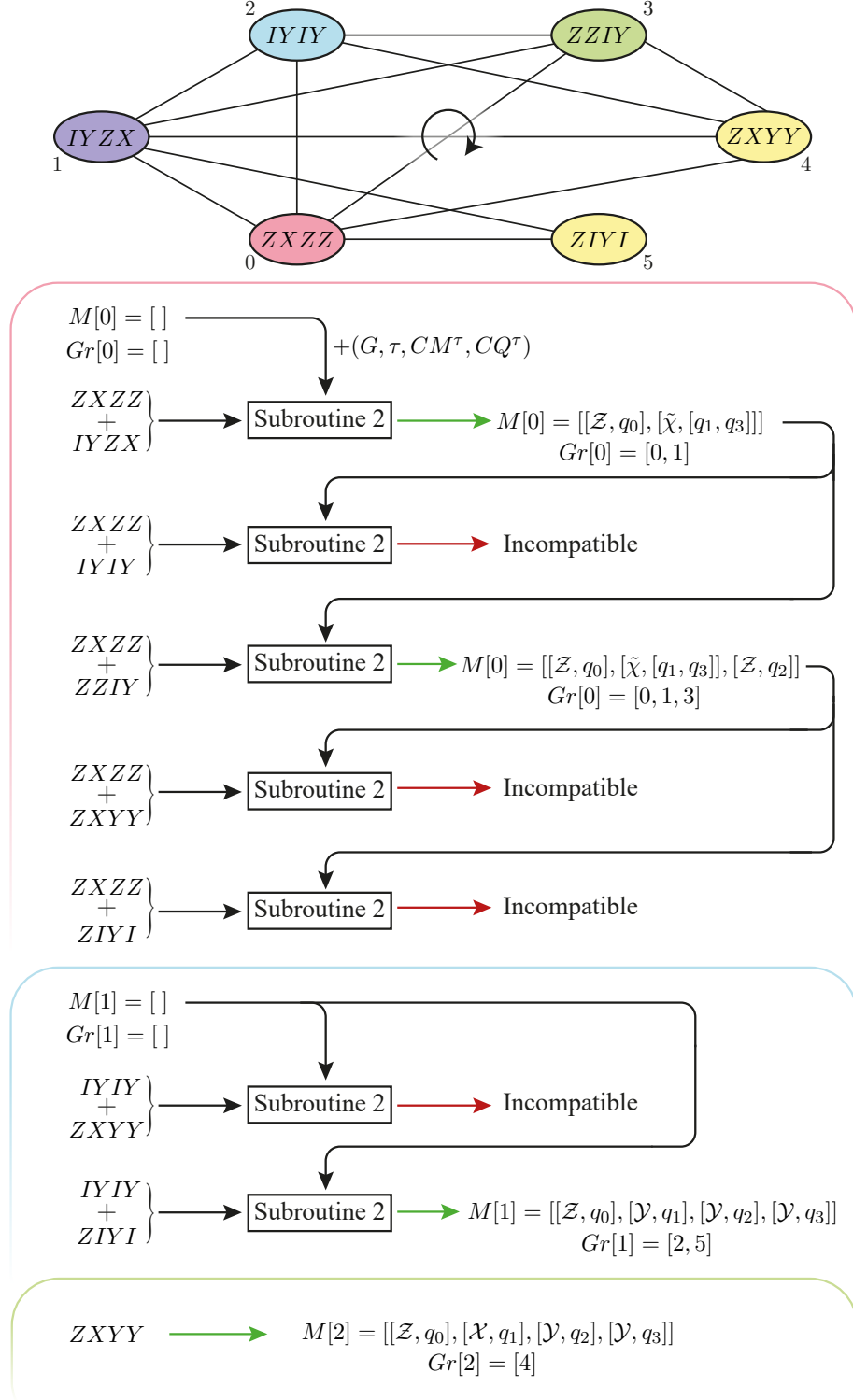
21 **return** M, Gr

Figure 6. **Example of Subroutine 1 (greedy Grouping taking chip's connectivity into account).** In the example, there are six Pauli strings to be grouped. On the top, the Pauli graph is depicted and on the bottom, a diagram details the algorithm used by Subroutine 1. On the one hand, if we used the LDFC algorithm to group the Pauli strings in the TPB, we would obtain five groups, represented by the five different colors in the Pauli graph. On the other hand, if we use Subroutine 1 we obtain only three groups. The algorithm begins by sorting the Pauli strings in descending order of degree with respect to the Pauli graph. Then, it picks the highest degree string, $ZXZZ$ in our example, and tries to group it with the other strings using the Subroutine 2. Note that this subroutine also needs the chip's connectivity G, the *processor embedding* map τ, the order for the measurements CM^τ, and the order for the qubits CQ^τ. However, these extra inputs are not depicted for simplicity. If the grouping is successful (green arrow), the Subroutine 2 outputs the measurements and the qubits to which it must be performed, encoded in $M[i]$, and updates the list that encodes the Pauli strings grouped $Gr[i]$. For the next iteration, the current measurements are provided to the Subroutine 2 so that all the grouped strings are compatible with each other. If grouping is not possible (red arrow), the algorithm does not update the measurements and passes to the next string. Once all Pauli strings are tested, we pick the next string with the highest degree which has not been grouped yet, and repeat the process until all strings are measured.
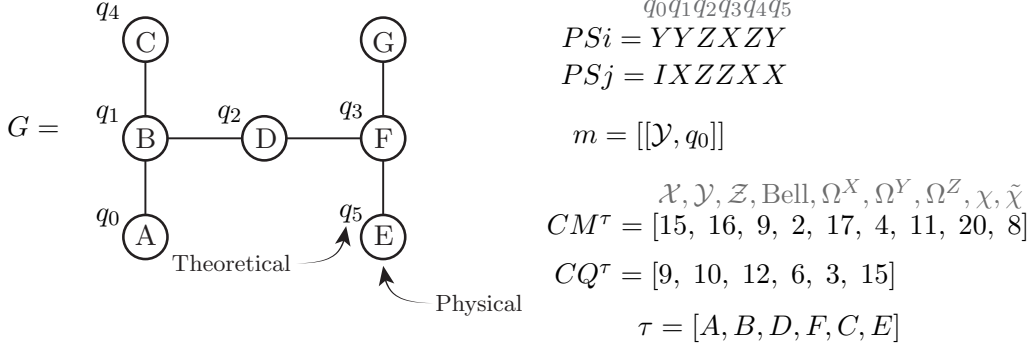

Subroutine 2: Greedy measurement assignment taking chip's connectivity into account

---

Input: Two Pauli strings PSi and PSj, chip's connectivity G, current assignment of measurements m, *processor embedding* map τ, and vectors CM^τ and CQ^τ encoding the orders of measurements and qubits, respectively

1 **if** PSj is not compatible with m **then**
2 $\quad$ **return False**, m

3 Copy m in $\tilde{m}$ $\quad$ // Partial measurements
4 Initialize U as the list of qubits not measured in m
5 Sort U in descending order according to CQ^τ
6 Remove from U the indices where PSi and PSj coincide
7 $B = [\mathcal{X}, \mathcal{Y}, \mathcal{Z}, \text{Bell}, \Omega^X, \Omega^Y, \Omega^Z, \chi, \tilde{\chi}]$
8 Sort B in descending order according to CM^τ
9 **while** Length$(U) \neq 0$ **do**
10 $\quad$ **for** ε in B **do**
11 $\quad\quad$ **for** p in permutations of U of size equal to the number of qubits where ε acts **do**
12 $\quad\quad\quad$ **if** ε acts on one qubit **then**
13 $\quad\quad\quad\quad$ **if** $PSi[p]$ and $PSj[p]$ are compatible with ε **then**
14 $\quad\quad\quad\quad\quad$ Append $[\varepsilon, p]$ to $\tilde{m}$
15 $\quad\quad\quad\quad\quad$ Remove qubit in p from U
16 $\quad\quad\quad\quad\quad$ Go to line 9
17 $\quad\quad\quad$ **else if** $\tau[p]$ connected in G **then**
18 $\quad\quad\quad\quad$ **if** $PSi[p]$ and $PSj[p]$ are compatible with ε **then**
19 $\quad\quad\quad\quad\quad$ Append $[\varepsilon, p]$ to $\tilde{m}$
20 $\quad\quad\quad\quad\quad$ Remove qubits in p from U
21 $\quad\quad\quad\quad\quad$ Go to line 9

22 $\quad$ **return False**, m
23 **return True**, $\tilde{m}$

$$G = $$

$$q_0 q_1 q_2 q_3 q_4 q_5$$
$$PSi = YYZXZY$$
$$PSj = IXZZXX$$

$$m = [[\mathcal{Y}, q_0]]$$

$$\mathcal{X}, \mathcal{Y}, \mathcal{Z}, \text{Bell}, \Omega^X, \Omega^Y, \Omega^Z, \chi, \tilde{\chi}$$
$$CM^\tau = [15, \ 16, \ 9, \ 2, \ 17, \ 4, \ 11, \ 20, \ 8]$$

$$CQ^\tau = [9, \ 10, \ 12, \ 6, \ 3, \ 15]$$

$$\tau = [A, B, D, F, C, E]$$

$$U = [q_5, q_1, q_3, q_4] \qquad B = [\chi, \Omega^X, \mathcal{Y}, \mathcal{X}, \Omega^Z, \mathcal{Z}, \tilde{\chi}, \Omega^Y, \text{Bell}] \qquad \tilde{m} = [[\mathcal{Y}, q_0]]$$

$\boxed{\varepsilon = \chi}$

$p = (q_5, q_1)$ disconnected

(q_5, q_3) connected

(Y, X) and (X, Z) are not compatible with χ

(q_3, q_5) connected

(X, Y) and (Z, X) are compatible with χ

$$\tilde{m} = [[\mathcal{Y}, q_0], [\chi, [q_3, q_5]]]$$
$$U = [q_1, q_4]$$

(q_1, q_4) connected

(Y, Z) and (X, X) are not compatible with χ

(q_1, q_4) connected

(Z, Y) and (X, X) are not compatible with χ

$\boxed{\varepsilon = \Omega^X}$

(q_1, q_4) connected

(Y, Z) and (X, X) are compatible with Ω^X

$$\tilde{m} = [[\mathcal{Y}, q_0], [\chi, [q_3, q_5]],$$
$$[\Omega^X, [q_1, q_4]]]$$
$$U = [\ ]$$

Figure 7. **Example of Subroutine 2 (Greedy measurement assignment taking chip's connectivity into account).** In this example, the algorithm has to either find HEEMs that can simultaneously measure the Pauli strings $YYZXZY$ and $IXZZXX$ (assuming that the theoretical qubit q_0 has been assigned the measurement $\mathcal{Y}$) or otherwise state the impossibility of the task. First of all, the algorithm creates U, which is the list of theoretical qubits where there is no assigned measurement nor the Pauli strings coincide. Note that U is sorted in descending order of CQ^τ. B is the vector of possible measurements, sorted in descending order of CM^τ. The first measurement to check is χ. The qubits q_1 and q_5 are not connected, so we try to measure the qubits q_5 and q_3, which in fact are compatible with χ. Then the algorithm tries to measure the qubits q_1 and q_4, but they are not compatible with χ. Thus, the algorithm checks if they are compatible with the *second preferred* measurement according to CM^τ, which is Ω^X. It succeeds in doing that, so both Pauli strings are compatible with HEEMs. Thus, the subroutine succeeds in finding HEEMs compatible with PSj, PSi, and the previous measurement defined in m.

In this section, we provide heuristic subroutines to build the *processor embedding* map (Subroutines 3 and 4) and the compatibility matrix (Subroutine 6).

Subroutine 3: Processor embedding map construction, without ensuring that $\tau(G')$ is connected

---

Input: n Pauli strings PS of N qubits the chip's connectivity G

1 C = Subroutine 6(PS)   // Compatibility matrix
2 $AQ = [\,]$ // Assigned qubits
3 Initialize τ as a list of size N // Embedding map
4 **while** Length(AQ) $\neq N$ **do**
5   Choose i, j such that $C[i,j] = \text{Max}(C)$
6   **if** (i in AQ) xor (j in AQ) **then**
7   **if** j in AQ **then**
8   Swap $i \leftrightarrow j$ // Ensure that $i \in AQ$ and $j \notin AQ$
9   Let J be an unassigned neighbor of $\tau[i]$ in G
10         $\tau[j] = J$
11   Append j to AQ
12   **else if** (i not in AQ) and (j not in AQ) **then**
13   Choose physical qubits I and J that are connected in G and unassigned
14         $\tau[i] = I$
15         $\tau[j] = J$
16   Append i and j to AQ
17   RemoveAssignedQubits(C, $\tau[i]$, AQ, G, τ)   // Subroutine 5
18   RemoveAssignedQubits(C, $\tau[j]$, AQ, G, τ)   // Subroutine 5
19     $C[i,j] = C[j,i] = \text{NaN}$
20 **return** τ

Subroutine 4: Processor embedding map construction, ensuring that $\tau(G')$ is connected

---

Input: n Pauli strings PS of N qubits and chip's connectivity G

1 $C = $ Subroutine 6(PS)   // Compatibility matrix
2 $AQ = [\,]$ // Assigned qubits
3 Initialize τ as a list of size N // Embedding map
4 Choose i, j such that $C[i,j] = \text{Max}(C)$
5 Choose I, J a pair of connected qubits in G
6 $\tau[i] = I$
7 $\tau[j] = J$
8 Append i and j to AQ
9 $C[i,j] = C[j,i] = \text{NaN}$
10 RemoveAssignedQubits(C, $\tau[i]$, AQ, G, τ)   // Subroutine 5
11 **while** Length(AQ) $\neq N$ **do**
12     $C' = C[AQ, :]$
13   Choose i, j such that $C'[i,j] = \text{Max}(C')$
14   **if** j not in AQ **then**
15   Let J be an unassigned neighbor of $\tau[i]$ in G
16         $\tau[j] = J$
17   Append j to AQ
18   RemoveAssignedQubits(C, $\tau[j]$, AQ, G, τ)   // Subroutine 5
19     $C[i,j] = C[j,i] = \text{NaN}$
20 **return** τ

Figure 8. **Example of Subroutine 3 (Processor embedding map construction, without ensuring that $\tau(G')$ is connected).** The subroutine begins by mapping the pair of theoretical qubits (q_1, q_2), corresponding to the highest entry of the compatibility matrix C, to a pair of physically connected qubits. Then, it updates the compatibility matrix, setting to NaN all the entries corresponding to q_1, as it has been mapped to a physical qubit with no more available connections. After that, it repeats this step two more times and determines a τ with a total number of $20 + 0 + 9 + 0 = 29$ compatibilities, which is 11 less than what the Subroutine 4 would get, as shown in Figure 9. Note that the physical connections B-C and D-E are not going to be useful to measure entangled pairs, which is something Subroutine 4 precludes.

A — B — C — D — E — F

$$C = \begin{pmatrix} \text{NaN} & 0 & 0 & 9 & 5 \\ 0 & \text{NaN} & \boxed{20} & 8 & 7 \\ 0 & 20 & \text{NaN} & 0 & 6 \\ 9 & 8 & 0 & \text{NaN} & 0 \\ 5 & 7 & 6 & 0 & \text{NaN} \end{pmatrix} \begin{matrix} q_0 \\ q_1 \\ q_2 \\ q_3 \\ q_4 \end{matrix}$$

Max

q_1 q_2
A — B — C — D — E — F
20

$$C = \begin{pmatrix} \text{NaN} & \text{NaN} & 0 & 9 & 5 \\ \text{NaN} & \text{NaN} & \text{NaN} & \text{NaN} & \text{NaN} \\ 0 & \text{NaN} & \text{NaN} & 0 & \boxed{6} \\ 9 & \text{NaN} & 0 & \text{NaN} & 0 \\ 5 & \text{NaN} & 6 & 0 & \text{NaN} \end{pmatrix} \begin{matrix} q_0 & q_1 & q_2 & q_3 & q_4 \end{matrix}$$

Assigned qubits

q_1 q_2 q_4
A — B — C — D — E — F
20 6

$$C = \begin{pmatrix} \text{NaN} & \text{NaN} & \text{NaN} & 9 & 5 \\ \text{NaN} & \text{NaN} & \text{NaN} & \text{NaN} & \text{NaN} \\ \text{NaN} & \text{NaN} & \text{NaN} & \text{NaN} & \text{NaN} \\ 9 & \text{NaN} & \text{NaN} & \text{NaN} & 0 \\ \boxed{5} & \text{NaN} & \text{NaN} & 0 & \text{NaN} \end{pmatrix} \begin{matrix} q_0 & q_1 & q_2 & q_3 & q_4 \end{matrix}$$

q_1 q_2 q_4 q_0
A — B — C — D — E — F
20 6 5

$$C = \begin{pmatrix} \text{NaN} & \text{NaN} & \text{NaN} & \boxed{9} & \text{NaN} \\ \text{NaN} & \text{NaN} & \text{NaN} & \text{NaN} & \text{NaN} \\ \text{NaN} & \text{NaN} & \text{NaN} & \text{NaN} & \text{NaN} \\ 9 & \text{NaN} & \text{NaN} & \text{NaN} & \text{NaN} \\ \text{NaN} & \text{NaN} & \text{NaN} & \text{NaN} & \text{NaN} \end{pmatrix} \begin{matrix} q_0 & q_1 & q_2 & q_3 & q_4 \end{matrix}$$

q_1 q_2 q_4 q_0 q_3
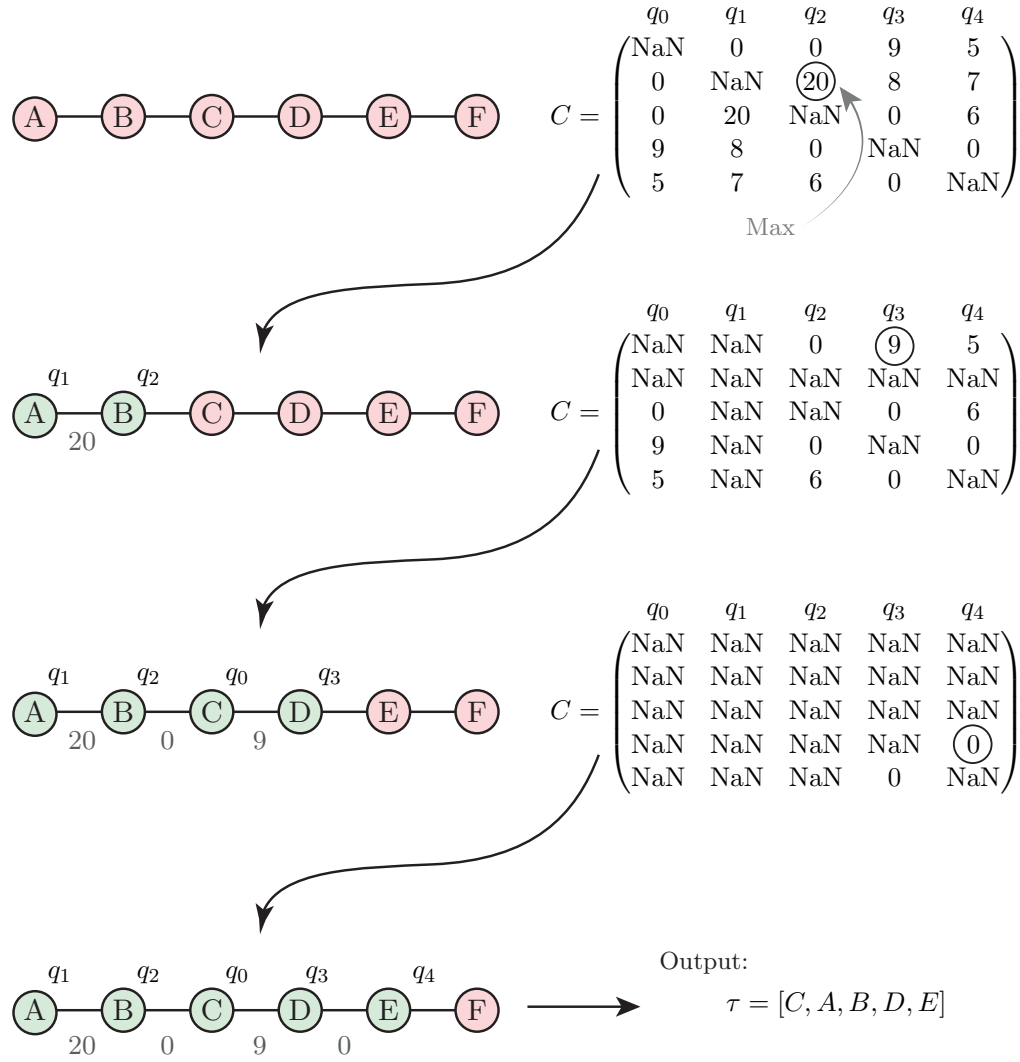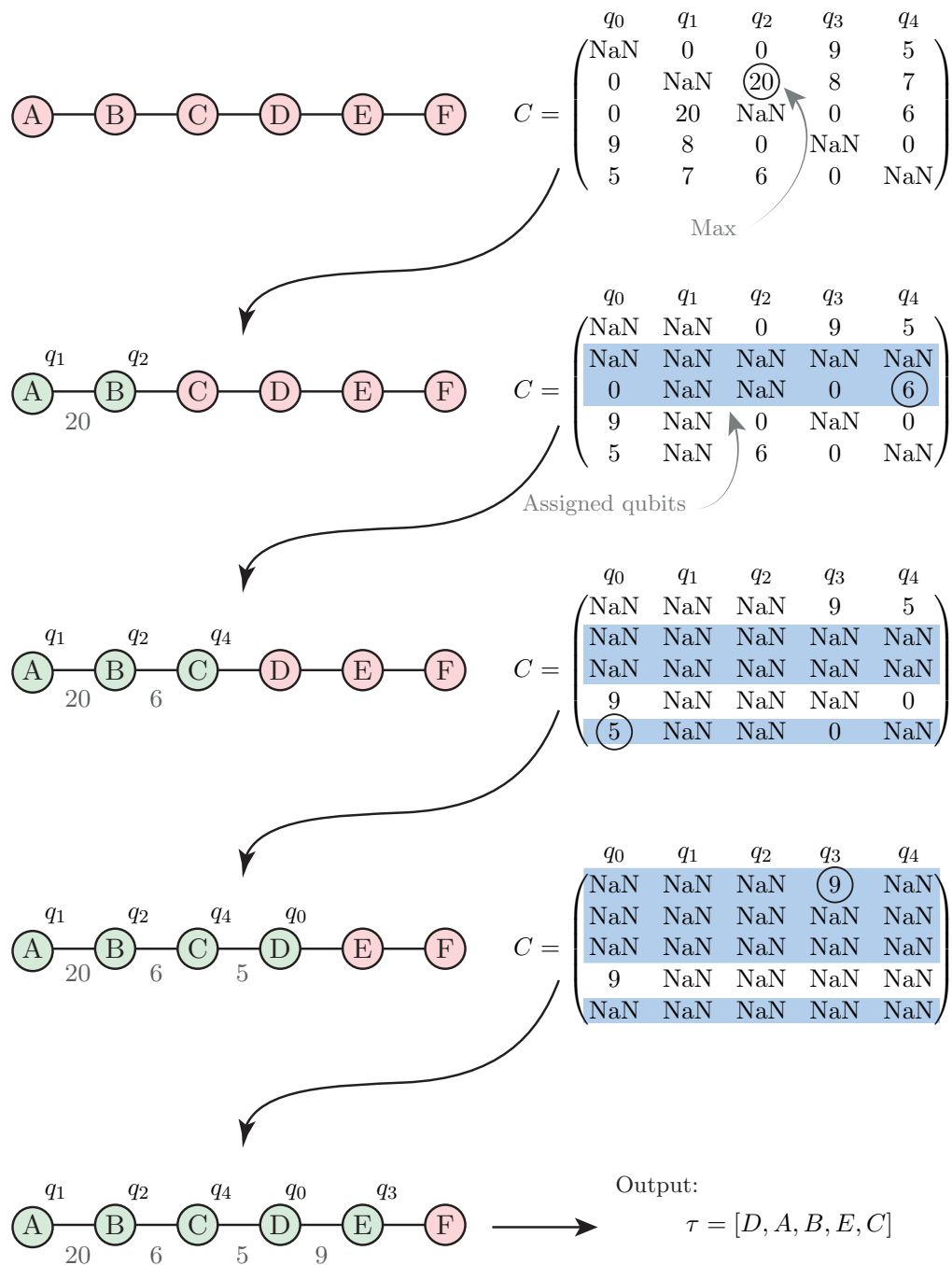A — B — C — D — E — F
20 6 5 9

Output:

$$\tau = [D, A, B, E, C]$$

Figure 9. Example of Subroutine 4 (Processor embedding map construction, ensuring that $\tau(G')$ is connected). The subroutine begins by mapping the pair of theoretical qubits (q_1, q_2), corresponding to the highest entry of the compatibility matrix C, to a pair of physically connected qubits. Then it updates the compatibility matrix, setting NaN for all entries corresponding to q_1, since it has been assigned to a physical qubit with no more available connections. Then, we search for the entry with maximum compatibility among the ones shared with the assigned qubits (blue shadow). This step ensures that no physical connection in the chip is wasted. The same procedure is repeated until all theoretical qubits have been assigned and ends up with a τ that has $20 + 6 + 5 + 9 = 40$ compatibilities, 11 more than Subroutine 3, as shown in Figure 8.

Subroutine 5: Remove qubits with no free neighbors

Input: Compatibility matrix C, physical qubit I, assigned qubits AQ, chip's connectivity G and *processor embedding* map τ

1 Let PQs be the list of assigned neighbors of I in G

2 Append I to PQs

3 **for** S in PQs **do**

4 $\quad$ Let s be the theoretical qubit assigned to S in τ

5 $\quad$ **if** all neighbors of S in G are assigned **then**

6 $\quad\quad$ $C[s,:] = C[:,s] = $ NaN

Subroutine 6: Construction of compatibility matrix

Input: n Pauli strings PS of N qubits

1 Initialize C as an $N \times N$ matrix of zeros

2 Set diagonal entries of C equal to NaN

3 **for** i in $[0, \ldots, N-2]$ **do**

4 $\quad$ **for** j in $[i+1, \ldots, N-1]$ **do**

5 $\quad\quad$ $nII = 0$

6 $\quad\quad$ **for** ps in PS **do**

7 $\quad\quad\quad$ $nII \mathrel{+}= (ps[i,j] == II)$ // 1 if True, 0 if False

8 $\quad\quad$ // Same for nXX, nYY, $\ldots$

9 $\quad\quad$ $n\text{Bell} = \text{Binomial}(nII + nXX + nYY + nZZ, 2)$

10 $\quad\quad$ $n\Omega^X = \text{Binomial}(nII + nXX + nYZ + nZY, 2)$

11 $\quad\quad$ // Repeat for all other measurements

12 $\quad\quad$.

13 $\quad\quad$.

14 $\quad\quad$.

15 $\quad\quad$ $C[i,j] = C[j,i] = n\text{Bell} + n\Omega^X + n\Omega^Y + n\Omega^Z + n\chi + n\tilde{\chi}$
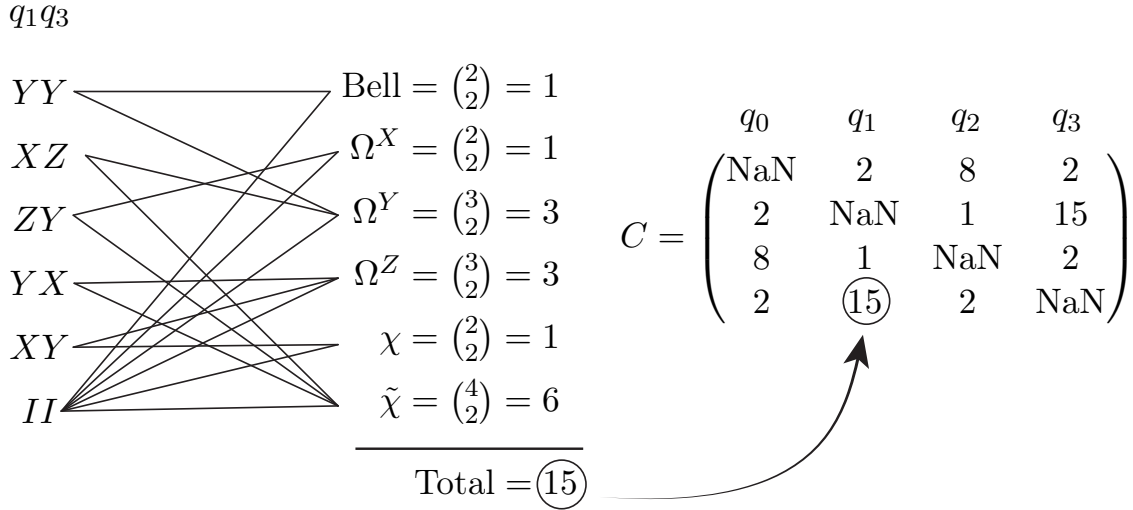
16 **return** C

Figure 10. **Example of Subroutine 6 (Construction of compatibility matrix).** The example shows how to compute the entry of the compatibility matrix corresponding to the compatibilities between q_1 and q_3 (these sub Pauli strings are those that come from the example of Figure 6). The graph on the left has an edge between a size 2 Pauli string and a HEEM if that Pauli string can be measured with that HEEM. Hence, the number of compatibilities due to the $\tilde{\chi}$ measurement is $\binom{4}{2}$ because $\tilde{\chi}$ is compatible with 4 Pauli strings and one can make $\binom{4}{2}$ different sets out of 4 different elements. The analog is true for the rest of the HEEMs. Summing all these numbers gives the total number of compatibilities between the theoretical qubits q_1 and q_3.

c. Choosing the iterative orders

Finally, Subroutine 7 outputs the τ-compatibility matrix C^τ and the vectors CM^τ and CQ^τ, which determines the order in which Subroutine 2 runs through the iterative elements.

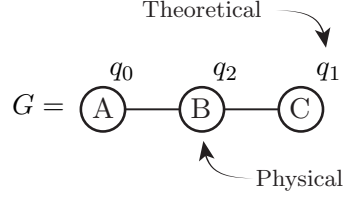**Subroutine 7:** Choosing the iterative orders

Input: n Pauli strings PS of N qubits, *processor embedding* map τ and chip's connectivity G

1 Initialize C^τ as an $N \times N$ matrix of zeros // Compatibility matrix
2 Set diagonal entries of C^τ equal to NaN
3 Initialize CM^τ as a list of size 9 with zeros // Measurement compabilities
4 Initialize CQ^τ as a list of size N with zeros // Qubit compabilities
5 Identify $\mathcal{X} = 0$, $\mathcal{Y} = 1$, $\mathcal{Z} = 2$, Bell $= 3$, $\Omega^X = 4$, ... // Analogue for all measurements
6 **for** i in $[0, \ldots, N-1]$ **do**
7 $\quad nI = 0$
8 $\quad$ **for** ps in PS **do**
9 $\quad\quad nI \mathrel{+}= (ps[i] == I)$ // 1 if True, 0 if False
10 $\quad$ // Same for nX, nY and nZ
11 $\quad CQ^\tau[i] \mathrel{+}= \text{Binomial}(nI + nX, 2) + \text{Binomial}(nI + nY, 2) + \text{Binomial}(nI + nZ, 2)$
12 $\quad CM^\tau[\mathcal{X}] \mathrel{+}= \text{Binomial}(nI + nX, 2)$
13 $\quad CM^\tau[\mathcal{Y}] \mathrel{+}= \text{Binomial}(nI + nY, 2)$
14 $\quad CM^\tau[\mathcal{Z}] \mathrel{+}= \text{Binomial}(nI + nZ, 2)$
15 $\quad$ **for** j in $[i+1, \ldots, N-1]$ **do**
16 $\quad\quad$ **if** $\tau[i]$ and $\tau[j]$ connected in G **then**
17 $\quad\quad\quad$ Define nXX, nXY, nXZ, ..., similar to nI for $ps[i,j]$
18 $\quad\quad\quad$ $n\text{Bell} = \text{Binomial}(nII + nXX + nYY + nZZ, 2)$
19 $\quad\quad\quad$ $n\Omega^X = \text{Binomial}(nII + nXX + nYZ + nZY, 2)$
20 $\quad\quad\quad$ // Repeat for all other measurements
21 $\quad\quad\quad$.
22 $\quad\quad\quad$.
23 $\quad\quad\quad$.
24 $\quad\quad\quad$ $C^\tau[i,j] = C^\tau[j,i] = n\text{Bell} + n\Omega^X + n\Omega^Y + n\Omega^Z + n\chi + n\tilde{\chi}$
25 $\quad\quad\quad$ $CM^\tau[\text{Bell}] \mathrel{+}= n\text{Bell}$
26 $\quad\quad\quad$ $CM^\tau[\Omega^X] \mathrel{+}= n\Omega^X$
27 $\quad\quad\quad$.
28 $\quad\quad\quad$.
29 $\quad\quad\quad$.
30 $\quad\quad\quad$ $CQ^\tau[i] \mathrel{+}= C^\tau[i,j]$
31 $\quad\quad\quad$ $CQ^\tau[j] \mathrel{+}= C^\tau[i,j]$

32 **return** C^τ, CM^τ, CQ^τ

Inputs

Theoretical

$$G = \begin{array}{ccc} q_0 & q_2 & q_1 \\ \text{(A)}\!-\!\!\text{(B)}\!-\!\!\text{(C)} \end{array}$$

Physical

$\text{PS} = [\text{IYI, YZX, ZXI, ZII, XIZ, XXZ}]$

$\tau = [A, C, B]$

$i = 0 \to q_0$

$CM^\tau[\mathcal{X}]+ = \binom{3}{2} = 3 \quad CM^\tau[\mathcal{Y}]+ = \binom{2}{2} = 1 \quad CM^\tau[\mathcal{Z}]+ = \binom{3}{2} = 3 \quad CQ^\tau[0]+ = 7$

$j = 1 \to [q_0, q_1]$
Disconnected

$j = 2 \to [q_0, q_2]$

$\quad CM^\tau[\Omega^Y]+ = \binom{3}{2} = 3 \quad CM^\tau[\Omega_Z]+ = \binom{2}{2} = 1 \quad CM^\tau[\tilde{\chi}]+ = \binom{4}{2} = 6$

$\quad CQ^\tau[0]+ = 10 \quad\quad CQ^\tau[2]+ = 10 \quad\quad C^\tau[0,2] = C^\tau[2,0] = 10$

$i = 1 \to q_1$

$CM^\tau[\mathcal{X}]+ = \binom{4}{2} = 6 \quad CM^\tau[\mathcal{Y}]+ = \binom{2}{2} = 1 \quad CM^\tau[\mathcal{Z}]+ = \binom{4}{2} = 6 \quad CQ^\tau[1]+ = 13$

$j = 2 \to [q_1, q_2]$

$\quad CM^\tau[\Omega^Y]+ = \binom{3}{2} = 3 \quad CM^\tau[\chi]+ = \binom{2}{2} = 1 \quad CM^\tau[\tilde{\chi}]+ = \binom{2}{2} = 1$

$\quad CQ^\tau[1]+ = 5 \quad\quad CQ^\tau[2]+ = 5 \quad\quad C^\tau[1,2] = C^\tau[2,1] = 5$

$i = 2 \to q_2$

$CM^\tau[\mathcal{X}]+ = \binom{4}{2} = 6 \quad CM^\tau[\mathcal{Y}]+ = \binom{3}{2} = 3 \quad CM^\tau[\mathcal{Z}]+ = \binom{5}{2} = 10 \quad CQ^\tau[2]+ = 19$

Outputs

$$
\begin{array}{c}
\begin{array}{ccc} q_0 & q_1 & q_2 \end{array} \\
C^\tau = \begin{pmatrix} 0 & 0 & 10 \\ 0 & 0 & 5 \\ 10 & 5 & 0 \end{pmatrix}
\end{array}
$$

$CQ^\tau = [17, \ 18, \ 34]$

$\mathcal{X}, \mathcal{Y}, \mathcal{Z}, \text{Bell}, \Omega^X, \Omega^Y, \Omega^Z, \chi, \tilde{\chi}$
$CM^\tau = [15, \ 7, \ 16, \ 0, \ 0, \ 6, \ 1, \ 1, \ 7]$

Figure 11. **Example of Subroutine 6 (Choosing the iterative orders).** The subroutine inputs a list of Pauli strings PS and a *processor embedding* map τ, and chooses an adequate way of running through the qubits' and measurements' loops of Subroutine 2. To do that, it computes the vectors CM^τ and CQ^τ, respectively. It begins by choosing the first theoretical qubit, q_0. Then it computes how many one-qubit $\mathcal{X}$-compatibilities arise in that qubit and adds them to $CM^\tau[\mathcal{X}]$ and to $CQ^\tau[0]$. The same is done for $\mathcal{Y}$ and $\mathcal{Z}$. Then, it runs through all pairs of theoretical qubits including q_0. First, it checks if τ maps the pair of qubits to connected physical qubits. If so, it counts the compatibilities due to two-qubit measurements involving that pair and adds them to the corresponding entries of C^τ, CM^τ, and CQ^τ. After that, the same is done with the remaining qubits.

Appendix C: Asymptotic scaling

In Fig. 12 a), we show the dependence of the number of groups on the total number of Pauli strings in the Hamiltonian of different molecules. We found that TPB obtains the largest number of groups for all molecules, while EM results in a significant reduction in the number of groups. HEEM obtains better results than TPB, but due to the constraint in the non-connected measurements, it does not reach the results of EM. However, in Fig. 12 b), we find a considerable improvement in the total number of CNOT gates. After transpiling, taking into account a real quantum device architecture, EM needs more CNOT gates to perform SWAP gates between non-connected qubits. This problem is solved by using HEEM, which does not need any SWAP gate, resulting in fewer CNOT gates.
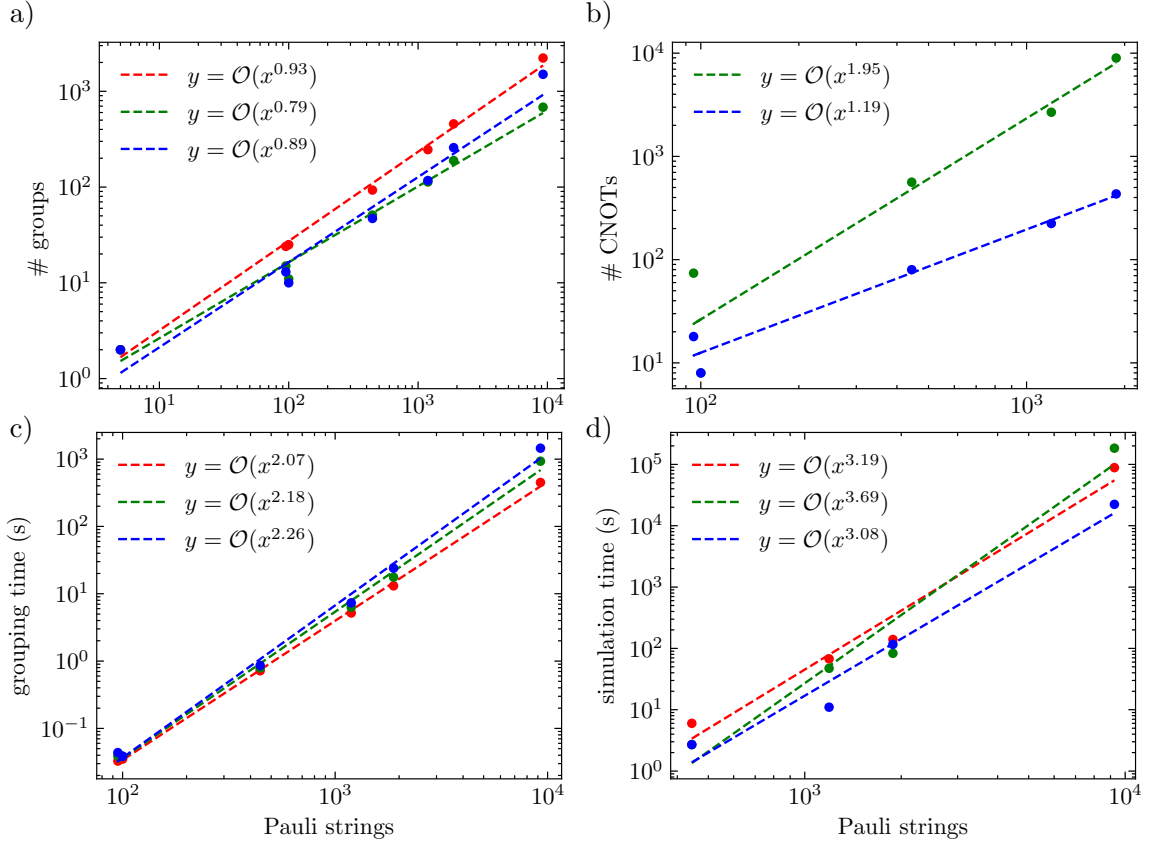

Figure 12. Dependence of **a)** the total number of groups, **b)** the number of CNOTs gates, **c)** CPU time for the grouping, and **d)** CPU time for the simulation of different molecules on the number of Pauli strings. The grouping algorithms used are TPB (red), EM (green), and HEEM (blue). The dashed lines represent a fit to the function $y = \beta x^{\alpha}$. Both EM and HEEM use *ibmq_montreal* connectivity. In **d)**, the simulation mimics the architecture, base gate, and noise of the quantum device *ibmq_montreal*. Each simulation has a total of 2^{14} shots evenly distributed across all measurements in each grouping. The simulation is performed using the qasm HPC simulator provided by IBMQ. The transpiling and queue times are not included in these times.

It is also important to consider the time needed to perform the grouping on a classical CPU, see Fig. 12 c). All three algorithms begin with the construction of the Pauli graph, which contains information about the commutative Pauli strings. Once the graph is obtained, TPB uses LDFC for graph coloring, whose time complexity is $\mathcal{O}(n^2)$, where n is the total number of Pauli strings. EM uses a more sophisticated algorithm which needs to run through the graph and check if it can group a pair of terms with any of the existing bases. This extra check results in a slower algorithm. Furthermore, HEEM also checks if the grouping is compatible with the chip's connectivity. However, despite the fact that the HEEM grouping is slower than the other methods, looking at the simulation time, Fig. 12 d), it is clear that the reduction in the simulation time is notorious. Even if TPB circuits do not require CNOT gates, the number of circuits to simulate is much larger than that of HEEM. On the other hand, EM grouping results in lower circuits, but the number of SWAP gates needed to perform the simulation grows much faster than that of HEEM, making it the slower algorithm to simulate.

Appendix D: Variational circuits

To compute the VQE of the H_2O molecule given in the main text, we use a circuit similar to the one shown below. First, the circuit is initialized with a Hartree-Fock state. The variational ansatz used is known as EfficientSU2, which is composed of a first layer of one-qubit rotations around the y-axis with the angles θ_i, and a second layer of one-qubit rotations around the z-axis. Then the entangling layer is composed of CNOT gates for all well-connected qubits on the chip. After that, two more one-qubit rotations around the y- and z-axes are applied. There is a total of $4N$ degrees of freedom, where N is the total number of qubits. Finally, we apply different one- and two-qubit gates to group the Hamiltonian terms, as explained in the main text, and measure the qubits.
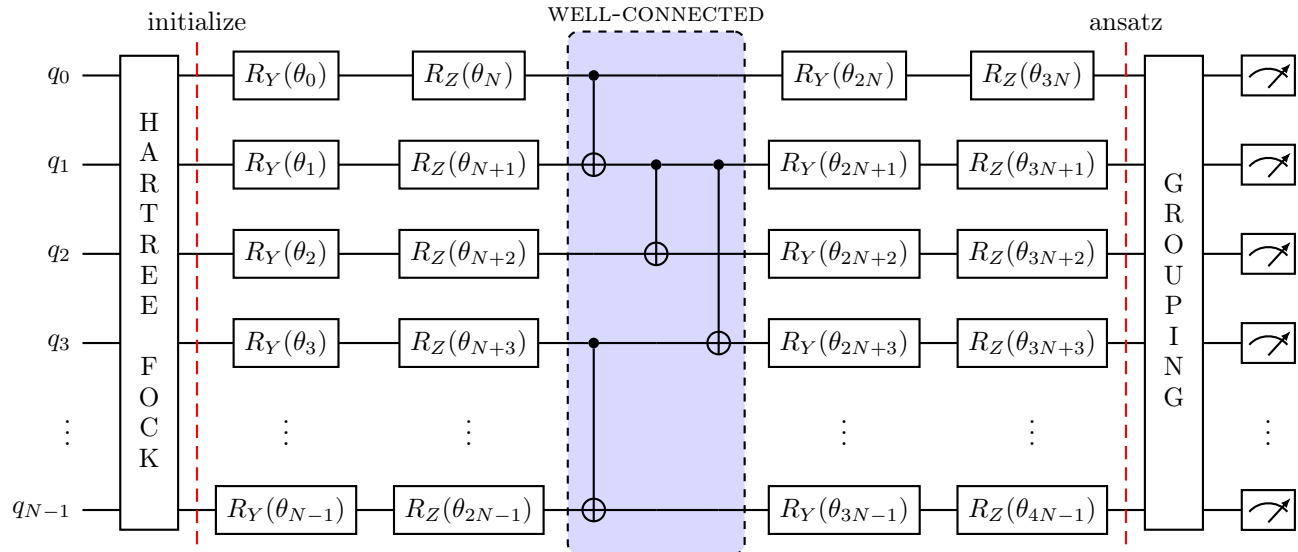

Figure 13. Variational circuit for variational quantum eigensolver with a total of N qubits, and $4N$ rotation angles θ_i. The entangling gates are only performed between qubits that are connected in the quantum chip.

[1] J. Preskill, Quantum computing in the NISQ era and beyond, Quantum **2**, 79 (2018).
[2] I. G. Ryabinkin, S. N. Genin, and A. F. Izmaylov, Constrained variational quantum eigensolver: Quantum computer search engine in the fock space, Journal of Chemical Theory and Computation **15**, 249 (2018).
[3] J. Romero, R. Babbush, J. R. McClean, C. Hempel, P. J. Love, and A. Aspuru-Guzik, Strategies for quantum computing molecular energies using the unitary coupled cluster ansatz, Quantum Science and Technology **4**, 014008 (2018).
[4] I. G. Ryabinkin, T.-C. Yen, S. N. Genin, and A. F. Izmaylov, Qubit coupled cluster method: A systematic approach to quantum chemistry on a quantum computer, Journal of Chemical Theory and Computation **14**, 6317 (2018).
[5] W. Kirby, B. Fuller, C. Hadfield, and A. Mezzacapo, Second-quantized fermionic operators with polylogarithmic qubit and gate complexity, PRX Quantum **3**, 020351 (2022).
[6] 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).
[7] J. Tilly, H. Chen, S. Cao, D. Picozzi, K. Setia, Y. Li, E. Grant, L. Wossnig, I. Rungger, G. H. Booth, and J. Tennyson, The variational quantum eigensolver: A review of methods and best practices, Physics Reports **986**, 1 (2022).
[8] A. F. Izmaylov, T.-C. Yen, and I. G. Ryabinkin, Revising the measurement process in the variational quantum eigensolver: is it possible to reduce the number of separately measured operators?, Chemical Science **10**, 3746 (2019).
[9] A. Peruzzo, J. McClean, P. Shadbolt, M.-H. Yung, X.-Q. Zhou, P. J. Love, A. Aspuru-Guzik, and J. L. O'Brien, A variational eigenvalue solver on a photonic quantum processor, Nat. Commun. **5** (2014).
[10] E. Farhi, J. Goldstone, and S. Gutmann, A quantum approximate optimization algorithm (2014), arXiv:1411.4028 [quant-ph].
[11] N. Moll, P. Barkoutsos, L. S. Bishop, J. M. Chow, A. Cross, D. J. Egger, S. Filipp, A. Fuhrer, J. M. Gambetta, M. Ganzhorn, A. Kandala, A. Mezzacapo, P. Müller, W. Riess, G. Salis, J. Smolin, I. Tavernelli, and K. Temme, Quantum optimization using variational algorithms on near-term quantum devices, Quantum Science and Technology **3**, 030503 (2018).

[12] 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).

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

[14] Y. Nam, J. S. Chen, N. C. Pisenti, K. Wright, C. Delaney, D. Maslov, K. R. Brown, S. Allen, J. M. Amini, J. Apisdorf, K. M. Beck, A. Blinov, V. Chaplin, M. Chmielewski, C. Collins, S. Debnath, K. M. Hudek, A. M. Ducore, M. Keesan, S. M. Kreikemeier, J. Mizrahi, P. Solomon, M. Williams, J. D. Wong-Campos, D. Moehring, C. Monroe, and J. Kim, Ground-state energy estimation of the water molecule on a trapped-ion quantum computer, npj Quantum Information **6**, 33 (2020).

[15] D. J. Egger, C. Gambella, J. Marecek, S. McFaddin, M. Mevissen, R. Raymond, A. Simonetto, S. Woerner, and E. Yndurain, Quantum computing for finance: State-of-the-art and future prospects, IEEE Transactions on Quantum Engineering **1**, 1 (2020).

[16] P. Vikstål, M. Grönkvist, M. Svensson, M. Andersson, G. Johansson, and G. Ferrini, Applying the quantum approximate optimization algorithm to the tail-assignment problem, Physical Review Applied **14**, 1 (2020).

[17] F. Neukart, G. Compostella, C. Seidel, D. von Dollen, S. Yarkoni, and B. Parney, Traffic flow optimization using a quantum annealer, Frontiers in ICT **4**, 29 (2017).

[18] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, and S. Lloyd, Quantum machine learning, Nature **549**, 195 (2017).

[19] M. Benedetti, E. Grant, L. Wossnig, and S. Severini, Adversarial quantum circuit learning for pure state approximation, New Journal of Physics **21**, 043023 (2019).

[20] A. Patterson, H. Chen, L. Wossnig, S. Severini, D. Browne, and I. Rungger, Quantum state discrimination using noisy quantum neural networks, Physical Review Research **3**, 013063 (2021).

[21] H. Chen, L. Wossnig, S. Severini, H. Neven, and M. Mohseni, Universal discriminative quantum neural networks, Quantum Machine Intelligence **3**, 1 (2021).

[22] X. Wang, Z. Song, and Y. Wang, Variational quantum singular value decomposition, Quantum **5**, 483 (2021).

[23] K. Wang, Z. Song, X. Zhao, Z. Wang, and X. Wang, Detecting and quantifying entanglement on near-term quantum devices (2020), arXiv:2012.14311 [quant-ph].

[24] A. D. Muñoz-Moller, L. Pereira, L. Zambrano, J. Cortés-Vega, and A. Delgado, Variational determination of multi-qubit geometrical entanglement in nisq computers (2021), arXiv:2110.03709 [quant-ph].

[25] X. Yuan, S. Endo, Q. Zhao, Y. Li, and S. C. Benjamin, Theory of variational quantum simulation, Quantum **3**, 191 (2019).

[26] T. Jones, S. Endo, S. McArdle, X. Yuan, and S. C. Benjamin, Variational quantum algorithms for discovering hamiltonian spectra, Phys. Rev. A **99**, 062304 (2019).

[27] P. García-Molina, J. Rodríguez-Mediavilla, and J. J. García-Ripoll, Quantum fourier analysis for multivariate functions and applications to a class of schrödinger-type partial differential equations, Physical Review A **105**, 012433 (2022).

[28] P. Jordan, E. Wigner, P. Jordan, and E. Wigner, Über das paulische Äquivalenzverbot, Eur. Phys. J. A **47**, 631 (1928).

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

[30] H. Y. Huang, R. Kueng, and J. Preskill, Predicting many properties of a quantum system from very few measurements, Nature Physics **16**, 1050 (2020).

[31] S. Chen, W. Yu, P. Zeng, and S. T. Flammia, Robust shadow estimation, PRX Quantum **2**, 030348 (2021).

[32] S. Aaronson, Shadow tomography of quantum states, SIAM Journal on Computing **49**, STOC18 (2020).

[33] J. Cotler and F. Wilczek, Quantum overlapping tomography, Physical Review Letters **124**, 100401 (2020).

[34] X. Bonet-Monroig, R. Babbush, and T. E. O'Brien, Nearly optimal measurement scheduling for partial tomography of quantum states, Phys. Rev. X **10**, 031064 (2020).

[35] N. C. Rubin, R. Babbush, and J. McClean, Application of fermionic marginal constraints to hybrid quantum algorithms, New Journal of Physics **20**, 053020 (2018).

[36] R. G. Melko, G. Carleo, J. Carrasquilla, and J. I. Cirac, Restricted boltzmann machines in quantum physics, Nature Physics **15**, 887 (2019).

[37] G. Torlai, G. Mazzola, G. Carleo, and A. Mezzacapo, Precise measurement of quantum observables with neural-network estimators, Physical Review Research **2**, 022060 (2020).

[38] J. R. McClean, J. Romero, R. Babbush, and A. Aspuru-Guzik, The theory of variational hybrid quantum-classical algorithms, New Journal of Physics **18**, 023023 (2016).

[39] A. Arrasmith, L. Cincio, R. D. Somma, and P. J. Coles, Operator sampling for shot-frugal optimization in variational algorithms (2020), arXiv:2004.06252.

[40] H. J. Vallury, M. A. Jones, C. D. Hill, and L. C. L. Hollenberg, Quantum computed moments correction to variational estimates, Quantum **4**, 373 (2020).

[41] A. Shlosberg, A. J. Jena, P. Mukhopadhyay, J. F. Haase, F. Leditzky, and L. Dellantonio, Adaptive estimation of quantum observables (2021), arXiv:2110.15339 [quant-ph].

[42] S. Bravyi, J. M. Gambetta, A. Mezzacapo, and K. Temme, Tapering off qubits to simulate fermionic hamiltonians (2017), arXiv:1701.08213 [quant-ph].

[43] T.-C. Yen, V. Verteletskyi, and A. F. Izmaylov, Measuring all compatible operators in one series of single-qubit measurements using unitary transformations, Journal of Chemical Theory and Computation **16**, 2400 (2020).

[44] V. Verteletskyi, T.-C. Yen, and A. F. Izmaylov, Measurement optimization in the variational quantum eigensolver using a minimum clique cover, The Journal of Chemical Physics **152**, 124114 (2020).

[45] P. Gokhale, O. Angiuli, Y. Ding, K. Gui, T. Tomesh, M. Suchara, M. Martonosi, and F. T. Chong, Minimizing state preparations in variational quantum eigensolver by partitioning into commuting families (2019), arXiv:1907.13623 [quant-ph].

[46] R. Kondo, Y. Sato, S. Koide, S. Kajita, and H. Takamatsu, Computationally Efficient Quantum Expectation with Extended Bell Measurements, Quantum **6**, 688 (2022).

[47] A. F. Izmaylov, T.-C. Yen, R. A. Lang, and V. Verteletskyi, Unitary partitioning approach to the measurement problem in the variational quantum eigensolver method, Journal of Chemical Theory and Computation **16**, 190 (2020).

[48] I. Hamamura and T. Imamichi, Efficient evaluation of quantum observables using entangled measurements, npj Quantum Information **6**, 56 (2020).

[49] O. Crawford, B. van Straaten, D. Wang, T. Parks, E. Campbell, and S. Brierley, Efficient quantum measurement of pauli operators in the presence of finite sampling error, Quantum **5**, 385 (2021).

[50] A. Zhao, A. Tranter, W. M. Kirby, S. F. Ung, A. Miyake, and P. J. Love, Measurement reduction in variational quantum algorithms, Phys. Rev. A **101**, 062322 (2020).

[51] A. Jena, S. Genin, and M. Mosca, Pauli partitioning with respect to gate sets (2019), arXiv:1907.07859 [quant-ph].

[52] P. Formanowicz and K. Tanaś, A survey of graph coloring - its types, methods and applications, Foundations of Computing and Decision Sciences **37**, 223 (2012).

[53] R. M. Karp, Reducibility among combinatorial problems, in *Complexity of Computer Computations* (Springer US, 1972) pp. 85–103.

[54] P. Manurangsi, Almost-polynomial ratio eth-hardness of approximating densest k-subgraph, in *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2017 (Association for Computing Machinery, New York, NY, USA, 2017) p. 954–961.

[55] R. Sotirov, On solving the densest k-subgraph problem on large graphs, Optimization Methods and Software **35**, 1160 (2020).

[56] C. Avanthay, A. Hertz, and N. Zufferey, A variable neighborhood search for graph coloring, European Journal of Operational Research **151**, 379 (2003).

[57] B. Tan and J. Cong, Optimal layout synthesis for quantum computing, in *Proceedings of the 39th International Conference on Computer-Aided Design* (ACM, 2020).

[58] D. C. Young, *Computational Chemistry* (John Wiley & Sons, Inc., 2001).

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

[60] F. Escudero, D. Fernández-Fernández, G. Jaumà, G. F. Peñas, and L. Pereira, Hardware efficient variational quantum eigensolver with entangled measurements: Code repository (2023).

[61] IBM Quantum Computing (2022).

[62] D. F. V. James, P. G. Kwiat, W. J. Munro, and A. G. White, Measurement of qubits, Phys. Rev. A **64**, 052312 (2001).

[63] H. Häffner, W. Hänsel, C. F. Roos, J. Benhelm, D. C. al kar, M. Chwalla, T. Körber, U. D. Rapol, M. Riebe, P. O. Schmidt, C. Becher, O. Gühne, W. Dür, and R. Blatt, Scalable multiparticle entanglement of trapped ions, Nature **438**, 643 (2005).

[64] D. Gross, Y.-K. Liu, S. T. Flammia, S. Becker, and J. Eisert, Quantum state tomography via compressed sensing, Phys. Rev. Lett. **105**, 150401 (2010).

[65] C. A. Riofrío, D. Gross, S. T. Flammia, T. Monz, D. Nigg, R. Blatt, and J. Eisert, Experimental quantum compressed sensing for a seven-qubit system, Nature Communications **8**, 15305 (2017).

[66] S. T. Flammia and Y.-K. Liu, Direct fidelity estimation from few pauli measurements, Phys. Rev. Lett. **106**, 230501 (2011).

[67] M. P. da Silva, O. Landon-Cardinal, and D. Poulin, Practical characterization of quantum devices without tomography, Phys. Rev. Lett. **107**, 210404 (2011).

[68] S. Endo, Z. Cai, S. C. Benjamin, and X. Yuan, Hybrid quantum-classical algorithms and quantum error mitigation, Journal of the Physical Society of Japan **90**, 032001 (2021).

[69] K. Temme, S. Bravyi, and J. M. Gambetta, Error mitigation for short-depth quantum circuits, Phys. Rev. Lett. **119**, 180509 (2017).

[70] Y. Li and S. C. Benjamin, Efficient variational quantum simulator incorporating active error minimization, Phys. Rev. X **7**, 021050 (2017).