Handbook for Quantifying Robustness of Magic

Hiroki Hamaguchi[1], Kou Hamada[1], and Nobuyuki Yoshioka[2,3,4]

[1]Graduate School of Information Science and Technology, University of Tokyo, Tokyo, 7-3-1 Hongo, Bunkyo-ku, Tokyo 113-8656, Japan

[2]Department of Applied Physics, University of Tokyo, 7-3-1 Hongo, Bunkyo-ku, Tokyo 113-8656, Japan

[3]Theoretical Quantum Physics Laboratory, RIKEN Cluster for Pioneering Research (CPR), Wako-shi, Saitama 351-0198, Japan

[4]JST, PRESTO, 4-1-8 Honcho, Kawaguchi, Saitama, 332-0012, Japan

Nonstabilizerness, or magic, is an essential quantum resource for performing universal quantum computation. Specifically, Robustness of Magic (RoM) characterizes the usefulness of a given quantum state for non-Clifford operations. While the mathematical formalism of RoM is concise, determining RoM in practice is highly challenging due to the necessity of dealing with a super-exponential number of stabilizer states. In this work, we present novel algorithms to compute RoM. The key technique is a subroutine for calculating overlaps between all stabilizer states and a target state, with the following remarkable features: (i) the time complexity per state is reduced *exponentially*, and (ii) the total space complexity is reduced *super-exponentially*. Based on this subroutine, we present algorithms to compute RoM for arbitrary states up to $n = 8$ qubits, whereas the naive method requires a memory size of at least $86\,\mathrm{PiB}$, which is infeasible for any current classical computer. Additionally, the proposed subroutine enables the computation of stabilizer fidelity for a mixed state up to $n = 8$ qubits. We further propose novel algorithms that exploit prior knowledge of the structure of the target quantum state, such as permutation symmetry or disentanglement, and numerically demonstrate our results for copies of magic states and partially disentangled quantum states. This series of algorithms constitutes a comprehensive "handbook" for scaling up the computation of RoM. We envision that the proposed technique will also apply to the computation of other quantum resource measures.

1 Introduction

Universal fault-tolerant quantum computation is often formulated such that the elementary gates consist of both classically simulatable gates and costful gates, such as in the most well-known Clifford+T formalism of the magic state model [1, 2, 3, 4, 5, 6, 7]. Since the consumption of the non-Clifford gates is indispensable for any quantum advantage [8, 9, 10, 11], there is a surging need to evaluate the complexity of quantum circuits using the framework of resource theory, in order to explore the boundary of quantum and classical computers [12, 13, 14, 15, 16, 17, 18]. One such attempt is the proposal of a quantity called

Hiroki Hamaguchi: hamaguchi-hiroki0510@g.ecc.u-tokyo.ac.jp

Kou Hamada: zkouaaa@g.ecc.u-tokyo.ac.jp

Nobuyuki Yoshioka: nyoshioka@ap.t.u-tokyo.ac.jp

arXiv:2311.01362v4 [quant-ph] 25 Aug 2024

Robustness of Magic (RoM) by Howard and Campbell [19, 20, 21]; RoM characterizes the classical simulation overhead or complexity of a given quantum state based on its effective amount of magic, and it geometrically quantifies the distance from the convex set of stabilizer states. More advanced resource measures have been proposed for quantifying the classical simulation cost, such as the stabilizer rank/extent [12, 22] and dyadic negativity [16], while RoM remains one of the most insightful measures in the context of gate synthesis. Together with the fact that the mathematical formulation of RoM is among the simplest and, hence, insightful for building large-scale computation techniques for other monotones as well, we focus on the computation of RoM in this work.

Although we can reduce the computation of RoM to a simple L^1 norm minimization problem, since the set of stabilizer states grows super-exponentially with the number of qubits n, the time and space complexity become prohibitively large, making computations for $n > 5$ qubits extremely challenging. For instance, memory consumption explodes to $86 \, \mathrm{PiB}$ already for an $n = 8$ qubit system. Existing works have attempted to mitigate this burden; for example, Heinrich et al. proposed leveraging the symmetry of the target state [21]. By exploiting the permutation symmetry between copies of identical states and internal (or local) symmetry, they demonstrated that up to $n = 26$ qubits RoM can be computed for symmetric magic states such as $|H\rangle^{\otimes n}$ used for T-gates. However, when investigating the magic resource of noisy states, there has been no valid method to scale up the resource characterization.

In this work, we propose a systematic procedure, presented in Table 1 and Fig. 1, to compute RoM that overcomes the limitations of existing methods. Central to our approach is a subroutine that computes the overlaps of a given quantum state with all stabilizer states, featuring (i) exponentially faster time complexity per state and (ii) super-exponentially smaller space complexity in total. Utilizing this subroutine, we present algorithms that surpass the current state-of-the-art RoM calculation results for arbitrary states up to $n = 8$ qubits. Furthermore, we extend the capability of these methods by incorporating preknowledge of the target quantum state's structure, such as permutation symmetry and decoupled structure, demonstrating that we can approximate RoM for multiple copies of arbitrary single-qubit states up to $n = 17$ qubits.

The remainder of this work is organized as follows. In Sec. 2, we present the preliminaries regarding the formalism of RoM. In Sec. 3, we first give the main subroutine on the overlap calculation in Theorem 1 and then present the algorithms that compute RoM with the reduced computational resource by utilizing the information of overlaps. In Sec. 4, we present algorithms for practical target states that are decoupled from each other, such as the multiple copies of single-qubit states or tensor products over subsystems. Finally, in Sec. 5, we provide the discussion and future perspective of our work.

Method	Target	Qubit count	Exact/Approximate
Naive LP [19]	Arbitrary	$n \leq 5$	Exact
Column Generation (CG)	Arbitrary	$n \leq 8$	Exact
Minimal Feasible Solution	Arbitrary	$n \leq 14$	2^n-approximation
Symmetry Reduction	$\rho^{\otimes n}$	$n \leq 17$	Exact up to $n \leq 7$
Partition Optimization	$\bigotimes_i \rho_i$	$n \leq 15$	Approximation
Symmetry Reduction [21]	$\rho_{H,F}^{\otimes n}$	$n \leq 26$	Exact up to $n \leq 9, 10$

Table 1: Methods for calculating RoM. We can apply the top three methods to arbitrary n-qubit states, whereas the subsequent three assume specific structures such as permutation symmetry, decoupled structure, and local symmetry, respectively.
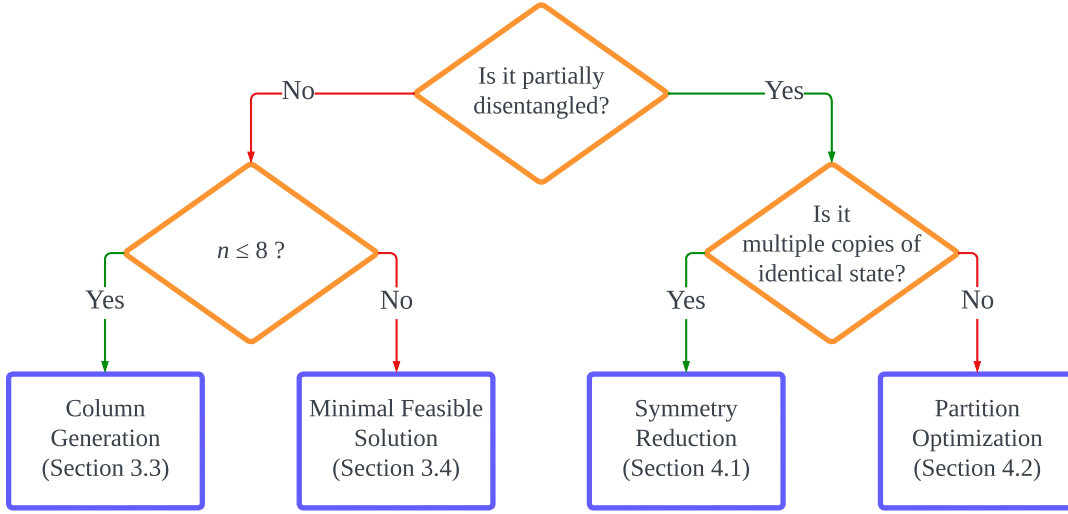
Figure 1: Flow chart of RoM computation.

2 Preliminaries

2.1 Stabilizer State

Let $\mathcal{P}_n = \{\pm 1, \pm i\} \times \{I, X, Y, Z\}^{\otimes n}$ denote the n-qubit Pauli group. For any n-qubit stabilizer state $|\phi\rangle$, we define the stabilizer group of $|\phi\rangle$ as $\mathrm{Stab}(|\phi\rangle) = \langle P_1, \ldots, P_n \rangle$, where $P \in \mathrm{Stab}(|\phi\rangle)$ satisfies $P|\phi\rangle = |\phi\rangle$ and each $P_i \in \mathcal{P}_n$ is an independent stabilizer generator. We denote the entire set of n-qubit stabilizer states by $\mathcal{S}_n$, whose size scales super-exponentially as $|\mathcal{S}_n| = 2^n \prod_{k=0}^{n-1}(2^{n-k}+1) = 2^{\mathcal{O}(n^2)}$ [23, Proposition 2][24, Proposition 2]. Additionally, we denote their convex hull as $\mathrm{STAB}_n = \{\sum_j x_j \sigma_j \mid \sigma_j \in \mathcal{S}_n, x_j \geq 0, \sum_j x_j = 1\}$. There are concise representations of a stabilizer state, such as the stabilizer tableau [23] or the check matrix representation [2, Section 10.5.1]. In this paper, we use the check matrix representation. For a stabilizer group $\langle P_1, \ldots, P_n \rangle$, let each generator P_i be expressed as

$$P_i = (-1)^{\delta_i} P_{i,1} \otimes P_{i,2} \otimes \cdots \otimes P_{i,n}, \tag{1}$$

where $P_{i,j} \in \{I, X, Y, Z\}$ and $\delta_i \in \{0, 1\}$. Note that the complex term is unnecessary. For any element $P \in \{\pm 1\} \times \{I, X, Y, Z\}^{\otimes n}$, if iP is in the stabilizer group, then $(iP)(iP) = -I^{\otimes n}$ should also be an element, which is a contradiction. Let $\mathbb{F}_2$ be the finite field with two elements. The check matrix of the stabilizer group is an $\mathbb{F}_2^{n \times 2n}$ matrix defined as

$$C := [X_n \ Z_n]$$

where $(X_n)_{i,j} \in \mathbb{F}_2$ is 1 iff $P_{i,j}$ is X or Y, and $(Z_n)_{i,j} \in \mathbb{F}_2$ is 1 iff $P_{i,j}$ is Z or Y. The i-th generator P_i corresponds to the i-th row vector of the matrix C, which is denoted by $r(P_i)$. Using this check matrix representation, we can confirm the independence and the commutativity of stabilizer generators as follows:

Lemma 1 ([2, Proposition 10.3]). *The generators $\{P_1, \ldots, P_n\}(\not\ni -I^{\otimes n})$ are mutually independent iff the rows of the corresponding check matrix C are linearly independent.*

Lemma 2 ([2, Exercise 10.33]). *The generators $\{P_1, \ldots, P_n\}$ are commute ($[P_i, P_j] = 0$) iff the corresponding check matrix C satisfies $C \begin{pmatrix} 0 & I_n \\ I_n & 0 \end{pmatrix} C^\top = 0$, where I_n is the identity matrix of size n.*

2.2 Robustness of Magic

Robustness of Magic (RoM) of a given n-qubit quantum state ρ can be interpreted as distance from the polytope of free states STAB_n and is defined as [19][25, Section 2]

$$\mathcal{R}(\rho) := \min_{\sigma_+, \sigma_- \in \mathrm{STAB}_n} \left\{ 2p + 1 \mid \rho = (p+1)\sigma_+ - p\sigma_-, \ p \geq 0 \right\}. \tag{2}$$

Given the entire set of stabilizer states $\mathcal{S}_n = \{\sigma_j\}_{j=1}^{|\mathcal{S}_n|}$, it is straightforward to demonstrate that this yields an equivalent expression as

$$\mathcal{R}(\rho) = \min_{\boldsymbol{x} \in \mathbb{R}^{|\mathcal{S}_n|}} \left\{ \|\boldsymbol{x}\|_1 \ \middle| \ \rho = \sum_{j=1}^{|\mathcal{S}_n|} x_j \sigma_j \right\},$$

which can be further simplified as

$$\mathcal{R}(\rho) = \min_{\boldsymbol{x} \in \mathbb{R}^{|\mathcal{S}_n|}} \left\{ \|\boldsymbol{x}\|_1 \mid \boldsymbol{A}_n \boldsymbol{x} = \boldsymbol{b} \right\}. \tag{3}$$

Here, we have utilized the unique decomposition of the quantum state into n-qubit Pauli operators defined by $b_i = \mathrm{Tr}[P_i \rho]$ and $(\boldsymbol{A}_n)_{i,j} = \mathrm{Tr}[P_i \sigma_j]$ where P_i ($1 \leq i \leq 4^n$) is the i-th Pauli operator in $\{I, X, Y, Z\}^{\otimes n}$. We call $\boldsymbol{b}$ as the Pauli vector of ρ, which can be computed with time complexity of $\mathcal{O}(n4^n)$ as described in Appendix B.1. Let $\sigma_j \in \mathcal{S}_n$ be expressed as $|\phi_j\rangle\langle\phi_j|$, and $\boldsymbol{a}_j$ denote the j-th column of $\boldsymbol{A}_n$. Note that $(\boldsymbol{a}_j)_i = (\boldsymbol{A}_n)_{i,j}$ is $1(= \mathrm{Tr}[P_i |\phi_j\rangle\langle\phi_j|])$ if P_i is a stabilizer of $|\phi_j\rangle$, -1 if $-P_i$ is a stabilizer of $|\phi_j\rangle$, and 0 otherwise. In practice, one may solve Eq. (3) as

$$\begin{aligned} \underset{\boldsymbol{u}}{\text{minimize}} \quad & \sum_i u_i \\ \text{subject to} \quad & \begin{pmatrix} \boldsymbol{A}_n & -\boldsymbol{A}_n \end{pmatrix} \boldsymbol{u} = \boldsymbol{b}, \\ & \boldsymbol{u} \geq \boldsymbol{0}, \end{aligned} \tag{4}$$

where the inequality for $\boldsymbol{u}$ is element-wise. This is a standard form of Linear Programming (LP), and we can solve this by LP solvers such as Gurobi [26] or CVXPY [27, 28].

2.3 Dualized Robustness of Magic

Since RoM can be formalized via the standard form of the linear program, the strong duality holds, which implies that the dual problem gives an equivalent definition. Concretely, RoM can be computed via the following:

$$\mathcal{R}(\rho) = \max_{\boldsymbol{y} \in \mathbb{R}^{4^n}} \left\{ \boldsymbol{b}^\top \boldsymbol{y} \ \middle| \ -\boldsymbol{1} \leq \boldsymbol{A}_n^\top \boldsymbol{y} \leq \boldsymbol{1} \right\}, \tag{5}$$

where $\boldsymbol{1}$ is a length-$|\mathcal{S}_n|$ vector with all the elements given by unity. By nature of the dual problem, any feasible solution yields a lower bound for RoM. For instance, RoM can be lower-bounded by the st-norm $\|\rho\|_{\mathrm{st}} = \frac{1}{2^n}\|\boldsymbol{b}\|_1$ by taking $\boldsymbol{y}$ as $y_i = \mathrm{sgn}(b_i)/2^n$ [21, Appendix B].

For the subsequent discussion, we introduce some notations. Let $\boldsymbol{C}$ be a submatrix of $\boldsymbol{A}_n$ with 4^n rows. We define the set of all columns in $\boldsymbol{A}_n$ as $\mathcal{A}_n = \{\boldsymbol{a}_j\}_{j=1}^{|\mathcal{S}_n|}$, and that in $\boldsymbol{C}$ as $\mathcal{C} \subseteq \mathcal{A}_n$. We define the minimization problem $\texttt{Prob}(\boldsymbol{C}, \boldsymbol{b})$ as Eq. (4) using the matrix $\boldsymbol{C}$ instead of $\boldsymbol{A}_n$. We also define a function $\texttt{SolveLP}(\mathcal{C}, \boldsymbol{b})$ which solves $\texttt{Prob}(\boldsymbol{C}, \boldsymbol{b})$. It returns the primal solution $\boldsymbol{x}$ for Eq. (3) and the dual solution $\boldsymbol{y}$ for Eq. (5).

3 Scaling Up RoM Calculation for Arbitrary States

It is well known that linear programming problems are solvable in polynomial time to the matrix size. However, the matrix size of $\boldsymbol{A}_n$ itself is $4^n \times |\mathcal{S}_n|$ where $|\mathcal{S}_n| = 2^{\mathcal{O}(n^2)}$, and hence it is impractical to use the entire $\boldsymbol{A}_n$ to tackle $n > 5$ qubit systems [15].

Motivated by this challenging scenario, we propose numerical algorithms to compute RoM for arbitrary quantum states, surpassing the state-of-the-art system size. The key technique involves overlap calculation with two notable features: (i) the time complexity is reduced from $\mathcal{O}(2^n|\mathcal{S}_n|)$ to $\mathcal{O}(n|\mathcal{S}_n|)$, representing an exponential reduction from $\mathcal{O}(2^n)$ to $\mathcal{O}(n)$ per stabilizer state, and (ii) the space complexity is super-exponentially reduced from $\mathcal{O}(2^n|\mathcal{S}_n|)$ to $\mathcal{O}(2^n)$, as we avoid explicitly constructing the entire $\boldsymbol{A}_n$.

Based on this subroutine, Algorithm 1, referred to as the top-overlap method, solves the primal problem (3) using a limited set of stabilizers. These stabilizers are selected based on the largest or smallest overlaps with the target quantum states. Algorithm 2, using the Column Generation method, improves upon Algorithm 1. It iteratively adds stabilizer states to the decomposition until all the inequality constraints in the dual problem (5) are satisfied. This ensures that the algorithm provides exact RoM.

In the following, we first present the fast overlap computation algorithm in Sec. 3.1, and then proceed to introduce two algorithms in Sec. 3.2 and 3.3, respectively. We demonstrate that these algorithms allow us to compute exact RoM for systems up to $n = 8$ qubits. In this context, the memory consumption for the subroutine is reduced by a factor of 10^8; compared to the entire $\boldsymbol{A}_n$ size of $86\,\mathrm{PiB}$, we can execute the subroutine only with $512\,\mathrm{MiB}$.

3.1 Core Subroutine: Fast Computation of Stabilizer Overlaps

First, we introduce the core subroutine in our work that computes the overlaps between all the stabilizer states and the target state, i.e., $\boldsymbol{A}_n^\top \boldsymbol{b}$, or the stabilizer overlaps in short. When we resort to a naive calculation, it requires the time complexity of $\mathcal{O}(2^n|\mathcal{S}_n|)$ to compute all the stabilizer overlaps via the matrix-vector product of $\boldsymbol{A}_n^\top \boldsymbol{b}$, even if we utilize the sparsity of $\boldsymbol{A}_n$. We can speed up this subroutine to $\mathcal{O}(n|\mathcal{S}_n|)$.

Theorem 1 (Computing stabilizer overlaps). *$\boldsymbol{A}_n^\top \boldsymbol{b}$ can be computed in $\mathcal{O}(n|\mathcal{S}_n|)$ time and in $\mathcal{O}(2^n)$ extra space excluding inputs and outputs.*

As we detail later in Sec. 3.2, this technique is crucial for scaling up RoM calculation to larger systems. To prove Theorem 1, it is beneficial to utilize the Fast Walsh–Hadamard Transform (FWHT) algorithm [29], which performs matrix-vector product operations when the matrix has a certain tensor product structure. Here we use the unnormalized Walsh–Hadamard matrix $H_n := \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}^{\otimes n}$, and refer to the matrix-vector multiplication of H_n as the FWHT algorithm. As evident from the pseudocode in Appendix A, its computational cost is as follows.

Lemma 3 (Complexity of FWHT algorithm). *In-place matrix-vector multiplication of H_n can be done with time complexity of $\mathcal{O}(n2^n)$ and space complexity of $\mathcal{O}(2^n)$.*

Indeed, $\boldsymbol{A}_n$ is essentially constructed from unnormalized Walsh–Hadamard matrices (see Fig. 2). We define $\mathcal{W}_n$ as the set of all sparsified Walsh–Hadamard matrices, which can be represented as $\begin{bmatrix} H_n \\ 0 \end{bmatrix} \in \mathbb{R}^{4^n \times 2^n}$ by appropriately reordering and flipping the signs of the rows. $\boldsymbol{A}_n$ is formed by concatenating these $W \in \mathcal{W}_n$ as follows:
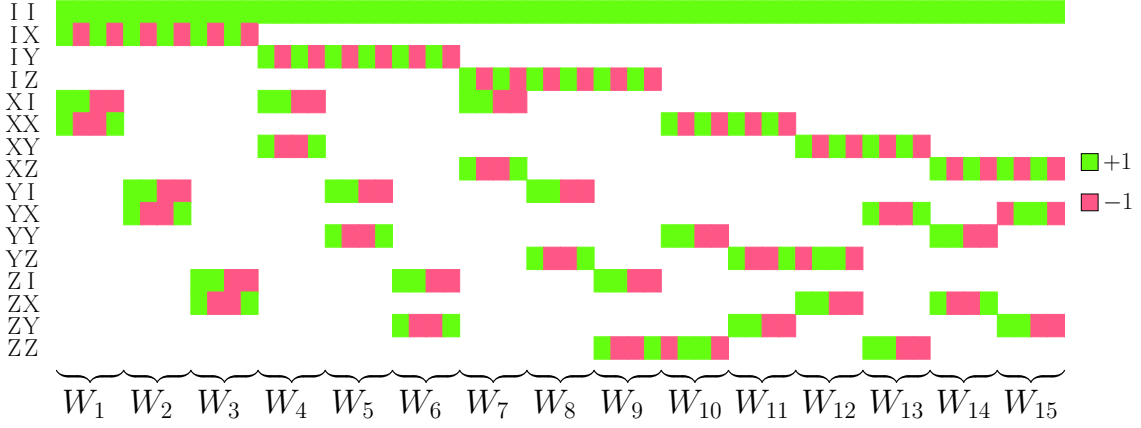
Figure 2: Visualization of $\boldsymbol{A}_n$ for $n = 2$. As stated in Lemma 4, $\boldsymbol{A}_n$ is composed of $W_i \in \mathcal{W}_n$.

Lemma 4 (Construction of $\boldsymbol{A}_n$). *For all $n \in \mathbb{N}$, using $W_j \in \mathcal{W}_n$, $\boldsymbol{A}_n$ can be written as*

$$\boldsymbol{A}_n = \begin{bmatrix} W_1 & \cdots & W_{|\mathcal{S}_n|/2^n} \end{bmatrix},$$

and there exists a constructive method to enumerate $\{W_j\}_{j=1}^{|\mathcal{S}_n|/2^n}$.

Proof. Firstly, for a check matrix C that satisfies both conditions in Lemma 1 and Lemma 2, we consider the stabilizer group $\langle P_1, \ldots, P_n \rangle$ with $\delta_i = 0$ ($1 \leq i \leq n$) in Eq. (1). We define δ as $\sum_{i=1}^n 2^{i-1}\delta_i$ and γ as $\sum_{i=1}^n 2^{i-1}\gamma_i$ ($\gamma_i \in \{0, 1\}$). Each element of the stabilizer group is expressed as $P_\gamma = P_1^{\gamma_1} \cdots P_n^{\gamma_n}$. Note that all generators commute, and all elements are distinct due to the conditions of the lemmas. For a stabilizer group with δ, i.e., $\langle (-1)^{\delta_1}P_1, \ldots, (-1)^{\delta_n}P_n \rangle$, the γ-th element is given by $((-1)^{\delta_1}P_1)^{\gamma_1} \cdots ((-1)^{\delta_n}P_n)^{\gamma_n} = (-1)^{\sum_{i=1}^n \gamma_i\delta_i}P_\gamma$, whose coefficient matches $(H_n)_{\gamma,\delta} = (-1)^{\sum_{i=1}^n \gamma_i\delta_i}$. Thus, for a check matrix C, the Walsh–Hadamard matrix H_n arises. By reordering and flipping the sign of each row in H_n according to the Pauli operator and phase of each P_γ, $W \in \mathcal{W}_n$ for the check matrix C can be obtained.

Next, we present a constructive method to enumerate all the check matrices. Note that this allows us to enumerate $\{W_j\}_{j=1}^{|\mathcal{S}_n|/2^n}$ simply by computing all P_γ for a given check matrix C and multiplying by H_n. To enumerate check matrices, we use the following standard form of the check matrix representation:

$$\begin{pmatrix} I_k & \hat{X} & \hat{Z} & O \\ O & O & \hat{X}^\top & I_{n-k} \end{pmatrix}, \tag{6}$$

where $\hat{X} \in \mathbb{F}_2^{k \times (n-k)}$ is given by reduced row echelon form with rank k and $\hat{Z} = \hat{Z}^\top \in \mathbb{F}_2^{k \times k}$ is a symmetric matrix. Since all the choices of $\hat{X}$ and $\hat{Z}$ produce mutually distinct check matrices that satisfy both Lemmas 1 and 2, we can ensure that Eq. (6) yields a unique construction of the set of check matrices. We can verify that this method indeed constructs all the check matrices. The number of choices for $\hat{Z}$ is $2^{k(k+1)/2}$, and for $\hat{X}$ it is given by the q-binomial coefficient $\begin{bmatrix} n \\ k \end{bmatrix}_2$ [30, Theorem 7.1], defined for a general $q(\neq 1)$ as

$$\begin{bmatrix} n \\ k \end{bmatrix}_q = \frac{(1-q^n)(1-q^{n-1})\ldots(1-q^{n-k+1})}{(1-q)(1-q^2)\ldots(1-q^k)}.$$

Now, the q-binomial theorem [31] shows $\sum_{k=0}^n \begin{bmatrix} n \\ k \end{bmatrix}_2 2^{k(k+1)/2} = \prod_{k=0}^{n-1}(2^{n-k}+1) = |\mathcal{S}_n|/2^n$. This verifies that the constructive method enumerates all the stabilizer states, which completes the proof. $\square$

We utilized the Gray code to enumerate the matrix representations in the actual numerical implementation. Refer to the codes available on GitHub [32] for details. Now, by applying Lemma 3 for each W_j in Lemma 4, we complete the proof of Theorem 1.

We can compute the maximum fidelity as well. The fidelity between quantum states $\sigma_j = |\phi_j\rangle\langle\phi_j|$ and ρ is defined as

$$F(\sigma_j, \rho) := \mathrm{Tr}\left[\sqrt{\sigma_j^{1/2}\rho\sigma_j^{1/2}}\right] = \mathrm{Tr}\left[\sqrt{|\phi_j\rangle\langle\phi_j|\rho|\phi_j\rangle\langle\phi_j|}\right] = \sqrt{\langle\phi_j|\rho|\phi_j\rangle}.$$

Since $\boldsymbol{b}$ is a Pauli vector of ρ, we have $\rho = \frac{1}{2^n}\sum_{i=1}^{4^n} b_i P_i$ and

$$F^2(\sigma_j, \rho) = \langle\phi_j|\rho|\phi_j\rangle = \mathrm{Tr}[\sigma_j\rho] = \mathrm{Tr}\left[\left(\frac{1}{2^n}\sum_{i=1}^{4^n}(\boldsymbol{a}_j)_i P_i\right)\left(\frac{1}{2^n}\sum_{i=1}^{4^n} b_i P_i\right)\right] = \frac{1}{2^n}\boldsymbol{a}_j^\top\boldsymbol{b}.$$

In the last equality, we used the orthogonality of Pauli matrices. Based on this relationship, we can derive the following corollary of Theorem 1.

Corollary 1 (Computing maximum fidelity). *Maximum fidelity for ρ can be computed with the time complexity of $\mathcal{O}(n|\mathcal{S}_n|)$ and the space complexity of $\mathcal{O}(2^n)$ excluding inputs.*

For the case when $\rho = |\psi\rangle\langle\psi|$ is a pure state, we can compute the stabilizer fidelity [22] as

$$F_{\mathrm{STAB}}(\psi) := \max_{|\phi_j\rangle\in\mathcal{S}_n}|\langle\phi_j|\psi\rangle|^2 = \frac{1}{2^n}\left(\max_{1\leq j\leq|\mathcal{S}_n|}\boldsymbol{a}_j^\top\boldsymbol{b}\right).$$

It has been recognized that the stabilizer fidelity cannot be computed with a moderate computational cost for $n > 5$ [15]. Meanwhile, our algorithm allowed us to compute up to $n = 8$ in 4 hours. This numerical experiment was conducted using C++17 compiled by GCC 9.4.0 and a *cluster computer* powered by Intel(R) Xeon(R) CPU E5-2640 v4 with 270 GB of RAM using 40 threads. Even if we use a *laptop* powered by Intel(R) Core(TM) i7-10510U CPU with 16 GB RAM, we can compute $n = 7$ in 2 minutes using 8 threads.

3.2 Top-Overlap Method for RoM

Using the overlap calculation subroutine presented in Sec. 3.1, we propose a novel algorithm that computes the approximate or exact RoM by utilizing the following properties: (i) by nature of L^1 norm minimization problem, the solution of the optimal stabilizer decomposition is sparse [33], and (ii) the stabilizer overlap is closely related to the optimal stabilizer decomposition (see Fig. 3(a)). Concretely, as we provide the detail in Algorithm 1, we restrict the number of columns in $\boldsymbol{A}_n$ and consider only a fraction K of stabilizer states with the largest or smallest overlaps; the fraction of $1 - K$ is neglected.

Algorithm 1: Top-Overlap Method for RoM

Input: Pauli vector $\boldsymbol{b}$ for the quantum state ρ, fraction K $(0 < K \leq 1)$
Output: An approximate or exact RoM
1 Compute the stabilizer overlaps $\boldsymbol{A}_n^\top\boldsymbol{b}$ using the FWHT algorithm.
2 $\mathcal{C}_0 \leftarrow$ Partial set of $\mathcal{A}_n$ with size of $K|\mathcal{S}_n|$ using the largest and smallest overlaps.
3 $\boldsymbol{x}_0, \boldsymbol{y}_0 \leftarrow \texttt{SolveLP}(\mathcal{C}_0, \boldsymbol{b})$
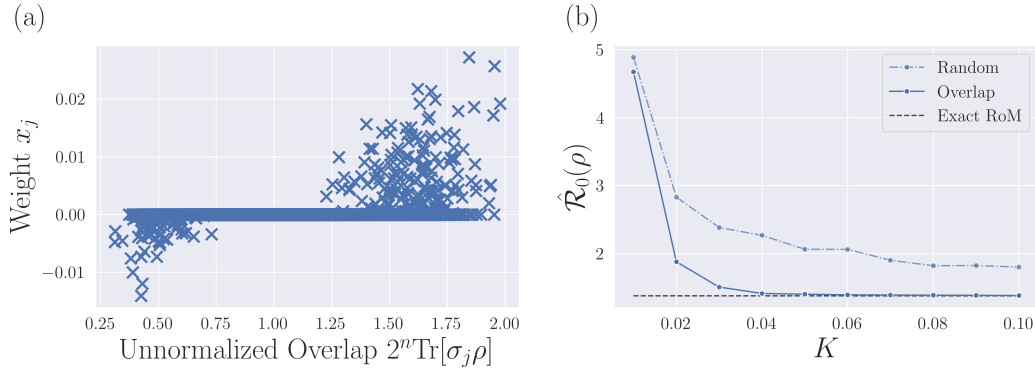4 **return** $\hat{\mathcal{R}}_0(\rho) = \|\boldsymbol{x}_0\|_1$

Figure 3: (a) Stabilizer overlaps $2^n \operatorname{Tr}[\sigma_j \rho]$ and the weights in the primal solution x_j for a Haar random mixed 4-qubit state ρ. (b) The computed value $\hat{\mathcal{R}}_0(\rho)$ in Algorithm 1 with a restricted set of stabilizers $\mathcal{C}_0$ of fraction $0 < K \leq 1$ for the same state ρ.

We can confirm the observation (ii) in Fig. 3(a). We show the distribution of stabilizer overlaps $\boldsymbol{a}_j^\top \boldsymbol{b} = 2^n \operatorname{Tr}[\sigma_j \rho]$ and their weights in the primal solution x_j for Haar random 4-qubit mixed state ρ. Indeed, we find strong correspondence between the stabilizer overlaps and weights in various instances. An intuitive explanation for this property can be provided by recalling the definition of RoM in Eq. (2), in which two states in STAB_n are used: ρ_+ and ρ_-. As illustrated in the visualization of RoM [19, Appendix A], ρ_+ resembles ρ as much as possible, while ρ_- diverges from ρ as much as possible. States with large overlaps support ρ_+, whereas states with small overlaps support ρ_-, indicating this tendency.

As we highlight in Fig. 3(b), we only need a small fraction $\mathcal{C}_0$ from the entire $\mathcal{A}_n$ to obtain a nearly exact solution $\hat{\mathcal{R}}_0(\rho) \approx \mathcal{R}(\rho)$ in Algorithm 1. We find that, for a Haar random mixed state of $n = 4$ qubit system, it is sufficient to use a fraction of $K := |\mathcal{C}_0|/|\mathcal{A}_n| = 0.05$ to achieve an absolute error of 0.023. The value of K required to achieve similar accuracy for larger systems are $K = 10^{-2}, 10^{-3}, 10^{-5}$ for $n = 5, 6, 7$, respectively. Meanwhile, we must take a significantly larger column set to obtain the exact RoM; the fraction should be $K \sim 0.32$ for $n = 4$ qubit case. We provide further numerical details in Appendix E.1.

3.3 Column Generation Method for Dualized RoM

Despite the significant improvement over the naive method, Algorithm 1 contains some issues: (i) we cannot tell whether the column set $\mathcal{C}_0 \subset \mathcal{A}_n$ is sufficient to yield the exact RoM, (ii) there is no quantitative measure to judge the quality of the solution, and (iii) there is a large gap between "highly approximate" and "exact" RoMs in the computational resource. These issues are well addressed when we consider dualized formalism instead. Recall that the dualized formulation of RoM in Eq. (5) is given by

$$\mathcal{R}(\rho) = \max_{\boldsymbol{y} \in \mathbb{R}^{4^n}} \left\{ \boldsymbol{b}^\top \boldsymbol{y} \ \middle| \ -\boldsymbol{1} \leq \boldsymbol{A}_n^\top \boldsymbol{y} \leq \boldsymbol{1} \right\}.$$

If there exists $\boldsymbol{a} \in \mathcal{A}_n$ such that violates the constraint in Eq. (5), $|\boldsymbol{a}^\top \boldsymbol{y}| > 1$, then we must use those violated $\boldsymbol{a}$'s to attain the exact RoM (Also refer to Fig. 4). Conversely, without any violation, the solution is exact owing to the problem's strong duality. This motivates us to derive Algorithm 2 using the Column Generation (CG) technique [34].
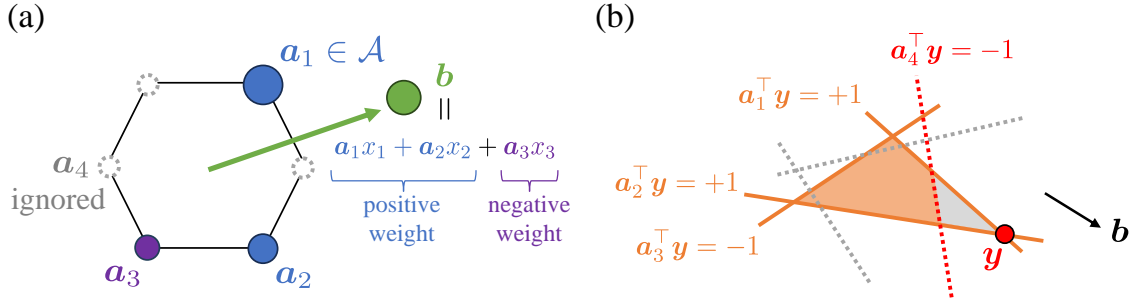
Figure 4: (a) Graphical description of the primal formalism. The Pauli vector $\boldsymbol{b}$ of the target state is decomposed into a sum over those of pure stabilizer states $\boldsymbol{a}$ denoted by vertices on the stabilizer polytope to minimize $\|\boldsymbol{x}\|_1$. The restriction on columns of $\boldsymbol{A}_n$ is expressed by gray vertices, which are eliminated from the decomposition. (b) Graphical description of the dual formalism. The equality constraints in the primal problem are now given as inequalities $-\boldsymbol{1} \leq \boldsymbol{A}_n^\top \boldsymbol{y} \leq \boldsymbol{1}$ for the dual variable $\boldsymbol{y}$, while the objective function is the inner product with $\boldsymbol{b}$. A solution $\boldsymbol{y}$ obtained from reduced column set $\mathcal{C}$ (orange) may violate some constraints (red). So, one should add those columns to improve the solution, while some columns denoted by gray dotted lines do not affect the result.

Algorithm 2: Exact RoM Calculation by Column Generation

Input: Pauli vector $\boldsymbol{b}$ of the target state ρ
Output: Exact RoM $\mathcal{R}(\rho)$

1   $\mathcal{C}_0 \leftarrow$ Partial set of $\mathcal{A}_n$ with size of $K|\mathcal{S}_n|$ using the largest and smallest overlaps.
2   **for** $k = 0, 1, 2, \ldots$ **do**
3      $\boldsymbol{x}_k, \boldsymbol{y}_k \leftarrow \texttt{SolveLP}(\mathcal{C}_k, \boldsymbol{b})$
4      $\hat{\mathcal{R}}_k(\rho) \leftarrow \|\boldsymbol{x}_k\|_1$
5      $\mathcal{C}' \leftarrow \left\{ \boldsymbol{a}_j \in \mathcal{A}_n \mid \left| \boldsymbol{a}_j^\top \boldsymbol{y}_k \right| > 1 \right\}$ /* Use of FWHT */
6   **if** $\mathcal{C}' = \emptyset$ **then**
7   **return** $\mathcal{R}(\rho) = \hat{\mathcal{R}}_k(\rho)$
8      $\mathcal{C}_{k+1} \leftarrow \mathcal{C}_k \cup \mathcal{C}'$ /* Two modifications in the main text */

The CG technique means to compute the overlap $\left| \boldsymbol{a}_j^\top \boldsymbol{y}_k \right|$ for all $\boldsymbol{a}_j \in \mathcal{A}_n$ at each iteration to find and generate the columns violating the constraint. By iteratively updating $\mathcal{C}_k$ until there is no violation, we obtain the exact RoM. As in Table 2, We initialized the partial column set $\mathcal{C}_0$ with $K = 10^{-5}, 10^{-8}$ for $n = 7, 8$, respectively. In Fig. 5, we present the numerical experiment results of Algorithm 2. The number of violated columns decreased rapidly, and the exact RoM was obtained after a small number of iterations. The run time was 2 hours using the laptop for $n = 7$ and 2 days using the cluster computer for $n = 8$. Although the algorithm is not guaranteed to yield exact solutions within a realistic run time, we expect the exact RoM to be obtained within a similar run time for almost all cases. See Appendix E.1 for more detailed results.

We remark two practical modifications in line 8 of Algorithm 2. First, we introduce a threshold d ($0 < d < 1$), such as 0.8, to discard columns $\boldsymbol{a}_j \in \mathcal{C}_k$ which satisfy both $\left| \boldsymbol{a}_j^\top \boldsymbol{y}_k \right| < d$ and $(\boldsymbol{x}_k)_j = 0$. Such columns do not affect the solution with high probability. Second, instead of adding the entire violating columns $\mathcal{C}'$ to $\mathcal{C}_{k+1}$, we set an upper bound on the number of the columns to add in order to suppress the memory consumption. For the calculation in Fig. 5, we have added only $K|\mathcal{S}_n|$ columns with either the largest or smallest overlaps at each iteration, significantly suppressing the memory consumption.

Qubit count n	Full-size $\boldsymbol{A}_n$	Our work	K
4	3 MiB	301 KiB	10^{-1}
5	379 MiB	4 MiB	10^{-2}
6	95 GiB	97 MiB	10^{-3}
7	86 TiB	499 MiB	10^{-5}
8	86 PiB	512 MiB	10^{-8}

Table 2: Memory size necessary for storing the entire $\mathcal{A}_n$, i.e., the matrix $\boldsymbol{A}_n$, and the reduced matrix for $\mathcal{C}_0$. $\mathcal{C}_0$ extracts only $K|\mathcal{A}_n|$ columns from $\mathcal{A}_n$ and is used in Algorithm 1 or the initialization step in Algorithm 2. This data size information relies on the sparse matrix format in SciPy [35].
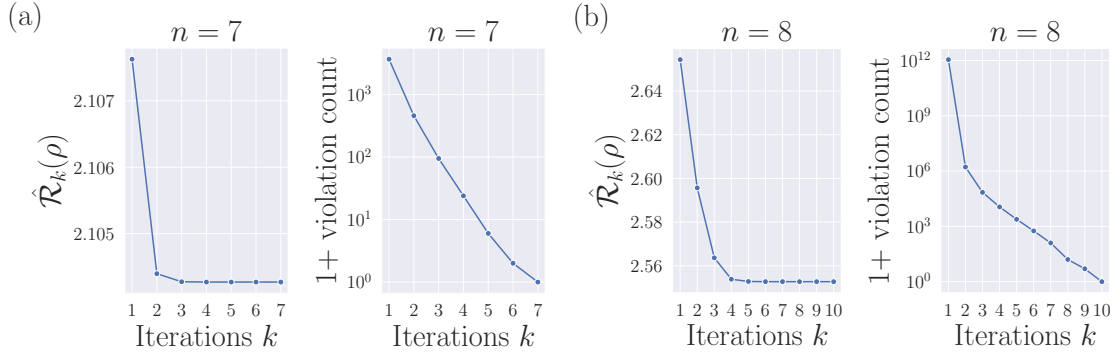

Figure 5: Demonstration of Algorithm 2 for Haar random mixed states of (a) $n = 7$ qubits and (b) $n = 8$ qubits. We display the value of $\hat{\mathcal{R}}_k(\rho)$ in Algorithm 2 and the number of violated inequality conditions ($|\mathcal{C}'|$). $|\mathcal{C}'|$ quickly decreases to zero, which assures that the exact RoM is obtained.

3.4 Minimal Feasible Solution With Accuracy Guarantee

Exact solutions for $n \geq 9$ qubit systems may require prohibitively enormous computational resources, while we may still wish to compute a feasible solution. In this section, we propose a method with minimal computational resources that always guarantees to yield a feasible solution. Consequently, we can obtain an approximate RoM with its corresponding feasible solution $\boldsymbol{x}$ for any state of $n = 14$ qubits within a minute.

The method's main idea is to construct a set $\mathcal{V}_n = \{W_1, \ldots, W_{2^n+1}\} \subset \mathcal{W}_n$ that always yields a feasible solution. We define a matrix $\boldsymbol{M}_n$ as

$$\boldsymbol{M}_n := \begin{bmatrix} W_1 & \cdots & W_{2^n+1} \end{bmatrix},$$

and for $W_j \in \mathcal{V}_n$, we denote the set of all the nonzero rows of the matrix W_j by $R(W_j)$. Then, $\mathtt{Prob}(\boldsymbol{M}_n, \boldsymbol{b})$ guarantees a feasible solution if and only if $\{R(W_j) \mid W_j \in \mathcal{V}_n\}$ is a cover of all the rows, i.e., $\bigcup_{W_j \in \mathcal{V}_n} R(W_j)$ equals the set of all the rows. The necessity is trivial, and the sufficiency is later discussed in Theorem 2. We refer to $\mathcal{V}_n$ as a cover set and $\boldsymbol{M}_n$ as a cover matrix. The following proposition assures the existence of the cover set; we prove it in Appendix C by taking a set of mutually unbiased bases [36, 37] as $\mathcal{V}_n$.

Proposition 1. *A cover set $\mathcal{V}_n$ with size $2^n + 1$ can be constructed.*

As a remark, the size $2^n + 1$ is minimum to ensure the feasibility. The nonzero-row set $R(W)$ covers 2^n rows, and the number of rows to be covered is 4^n. Since all the nonzero-row sets $R(W)$ shares the first row corresponding to $I^{\otimes n}$, the size of the cover set $\mathcal{V}_n$ should be equal or larger than $(4^n - 1)/(2^n - 1) = 2^n + 1$.
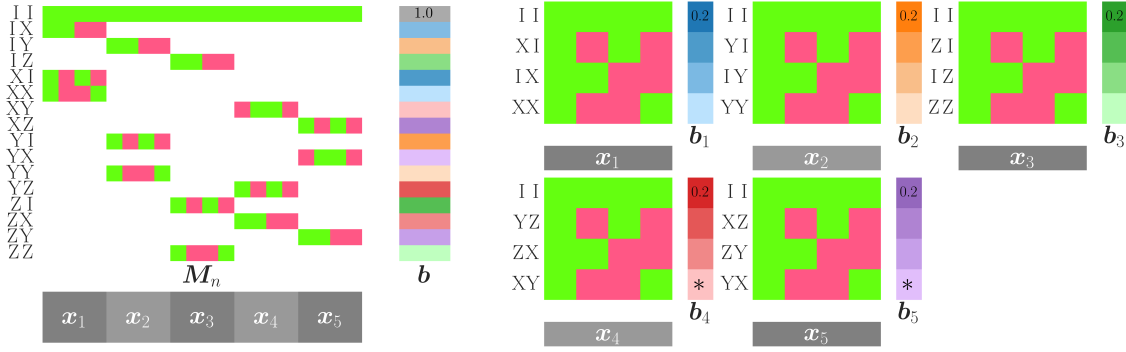
Figure 6: Graphical description of the cover matrix M_n and b_j. The first element of each b_j is $1/(2^n + 1)$, and the star mark (*) means the sign flip. We can execute the FWHT algorithm to solve $H_n x_j = b_j$.

Fig. 6 shows an example of the cover matrix with $n = 2$. Since the cover matrix uses the minimum possible number of blocks, only one block covers each row except for the first one. The following theorem exploits the structure of the cover matrix to obtain a feasible solution.

Theorem 2 (Minimal Feasible Solution). *A feasible solution for $\mathtt{Prob}(M_n, b)$ can be obtained with time complexity of $\mathcal{O}(n4^n)$.*

Proof. Our goal is to obtain a feasible solution for $\mathtt{Prob}(M_n, b)$, in other words, x such that $M_n x = b$. This can be obtained by solving the simultaneous equations $W_j x_j = b'_j$ $(1 \le j \le 2^n + 1)$ where $x_j \in \mathbb{R}^{2^n}$ and $b'_j \in \mathbb{R}^{4^n}$ which is defined as

$$\text{The } i\text{-th element of } b'_j := \begin{cases} 1/(2^n + 1) & \text{if } i = 0, \\ \text{the } i\text{-th element of } b & \text{if } i\text{-th row in } R(W_j), \\ 0 & \text{otherwise.} \end{cases}$$

Thus, by defining $b_j \in \mathbb{R}^{2^n}$ as the nonzero part of b'_j with reordering and sign flip so that W_n coincides with H_n, we have $H_n x_j = b_j \iff x_j = \frac{1}{2^n} H_n b_j$. By combining each x_j we can construct x. The time complexity is $\mathcal{O}((2^n + 1)n2^n)$ by the FWHT, i.e., $\mathcal{O}(n4^n)$. $\square$

The solution obtained in Theorem 2 is evidently feasible not only for $\mathtt{Prob}(M_n, b)$ but also for $\mathtt{Prob}(A_n, b)$, which provide an approximate value for the RoM, $R_{\text{FWHT}} := \|x\|_1$. By utilizing the st-norm $\|\rho\|_{\text{st}} = \frac{1}{2^n}\|b\|_1$ mentioned in Sec. 2.3, we can derive

$$R_{\text{FWHT}} = \sum_{j=1}^{2^n+1} \|x_j\|_1 = \sum_{j=1}^{2^n+1} \frac{1}{2^n}\|H_n b_j\|_1 \le \sum_{j=1}^{2^n+1} \|b_j\|_1 = 2^n\left(\frac{1}{2^n}\|b\|_1\right) = 2^n\|\rho\|_{\text{st}} \le 2^n\mathcal{R}(\rho).$$

Now, let us assume that each element $b_{j,i}$ of b_j is normally distributed with a mean of 0, which roughly approximates the distribution of a Haar random state. Then, since $\sum_{i=1}^{2^n} b_{ji}$ follows the distribution with a $2^{n/2}$ times standard deviation of b_{ji}, we have

$$\mathbb{E}\left[\frac{1}{2^n}\|H_n b_j\|_1\right] = \mathbb{E}\left[\left|\sum_{i=1}^{2^n} b_{j,i}\right|\right] = 2^{n/2}\mathbb{E}[|b_{j,i}|] = \frac{1}{2^{n/2}}\mathbb{E}[\|b_j\|_1],$$

which allows us to anticipate that $R_{\text{FWHT}} \approx 2^{n/2}\|\rho\|_{\text{st}}$.
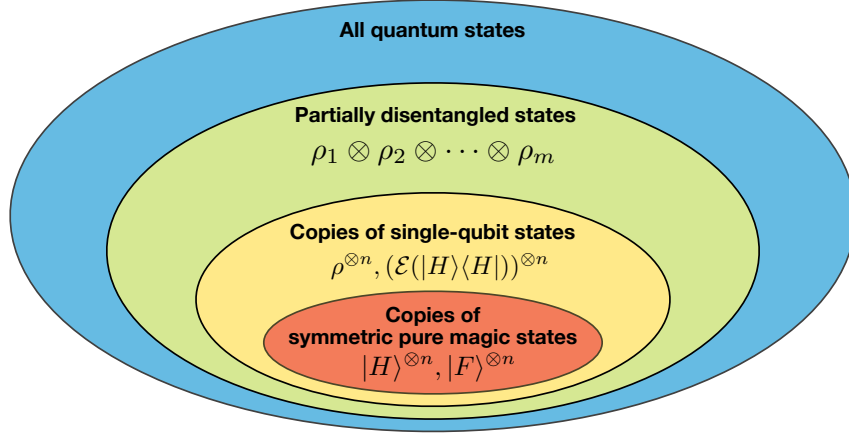
Figure 7: Hierarchy of partially disentangled quantum states.

Empirically, we confirmed that 100 Haar random 10-qubit mixed states satisfy

$$0.994 \leq \frac{R_{\mathrm{FWHT}}}{2^{n/2}\|\rho\|_{\mathrm{st}}} \leq 1.002.$$

This indicates $R_{\mathrm{FWHT}} \leq 2^{n/2}\mathcal{R}(\rho) + \mathcal{O}(1)$ with high probability. While R_{FWHT} deviates from the exact RoM, it is useful in certain scenarios, such as providing an initial solution for exact RoM calculation or assuring the feasibility of the algorithms (see Appendix E.2).

4 Quantum Resource of Multiple Magic States

One of the important applications of RoM is measuring the total nonstabilizerness of multiple magic states decoupled from each other. For instance, one may wish to evaluate the amount of magic resources for copies of states $\rho^{\otimes n}$ to estimate the upper bound on the number of generatable clean magic states. This could even include situations where quantum states are nonequivalent, such as partially decoupled states $\bigotimes_i \rho_i$. There exists a previous work by Heinrich and Gross [21] that has utilized the symmetry of some pure magic states such as $|H\rangle\langle H| = \frac{1}{2}\left(I + \frac{1}{\sqrt{2}}(X+Y)\right)$ and $|F\rangle\langle F| = \frac{1}{2}\left(I + \frac{1}{\sqrt{3}}(X+Y+Z)\right)$ to scale up the simulation up to $n = 26$ qubits, while we still lack a method to investigate general quantum states with partial disentangled structure (see Fig. 7).

In this section, we apply the algorithms proposed in Sec. 3 to practical problems: copies of identical quantum states $\rho^{\otimes n}$ and partially disentangled quantum states $\bigotimes_i \rho_i$. In particular, we first discuss the case of permutation symmetric state $\rho^{\otimes n}$ in Sec. 4.1, and then also consider general partially disentangled states in Sec. 4.2.

4.1 Copies of Single-Qubit States

When the target quantum state is given as identical copies of a quantum state $\rho^{\otimes n}$, we may compress the size of $\boldsymbol{A}_n$ by utilizing the permutation symmetry to combine multiple rows and columns of $\mathcal{A}_n$. In this work, we employed the compression method for $\mathcal{A}_n$ proposed in Ref. [21] to define a set of permutation symmetric columns $\mathcal{Q}_n$ (see Fig. 8). As in Ref. [21], we also used the data by Danielsen [38], which enables us to obtain the exact solutions for $n \leq 7$ qubits. Beyond $n = 7$ or 8 qubits, although a matrix $\boldsymbol{Q}_n$ of $\mathcal{Q}_n$ and $\boldsymbol{b}^{\otimes n}$ are reduced by permutation symmetry, it is not realistic to obtain the exact solution.
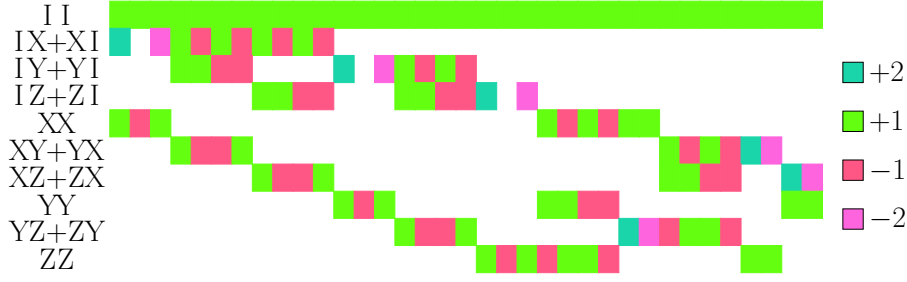
Figure 8: Visualization of $\boldsymbol{Q}_n$ for $n = 2$. Compare with Fig. 2 to confirm how the matrix is compressed.

Therefore, we propose an approximate method with the divide-and-conquer method. In the following, we use `Prob` and `SolveLP` for the compressed objects with the same meaning. Also, the tensor product of the compressed vectors is defined as the compressed vector of the tensor product of the uncompressed operands. In Algorithm 3, we consider all possible decomposition of m qubits into two groups with j and k qubits ($j + k = m$). Since we store the solutions for `Prob`$(\boldsymbol{Q}_i, \boldsymbol{b}^{\otimes i})$ for $i < m$, we can load the results. We take the union of tensor product of columns with nonzero weights as $\bigcup_{j+k=m}\{\boldsymbol{q}_j \otimes \boldsymbol{q}_k \mid \boldsymbol{q}_j \in \mathcal{C}_j, \boldsymbol{q}_k \in \mathcal{C}_k\}$ to construct $\mathcal{C}_m$, then we compute the approximate RoM with $\mathcal{C}_m$, which should be less than the product of RoM computed for each subsystem.

Algorithm 3: Approximate RoM for Permutation Symmetric States

Data: Compressed column set $\mathcal{Q}_n$ of matrix $\boldsymbol{Q}_n$
Input: Positive integer n, k ($n \geq k$), Pauli vector $\boldsymbol{b}$ for the target state ρ
Output: Approximate RoM R_i of $\rho^{\otimes i}$ ($i = 1, \ldots, n$)

1 **for** $i \leftarrow 1$ **to** k **do**
2      $\boldsymbol{x}_i, \boldsymbol{y}_i \leftarrow$ `SolveLP`$(\mathcal{Q}_i, \boldsymbol{b}^{\otimes i})$
3      $R_i \leftarrow \|\boldsymbol{x}_i\|_1$
4      $\mathcal{C}_i \leftarrow$ columns $\{\boldsymbol{q}_j\} \subseteq \mathcal{Q}_i$ where $(\boldsymbol{x}_i)_j$ is nonzero.

5 **for** $i \leftarrow k + 1$ **to** n **do**
6      $\mathcal{C}' \leftarrow \emptyset$
7   **for** $l \leftarrow 1$ **to** $\lfloor i/2 \rfloor$ **do**
8          $m \leftarrow i - l$
9          $\mathcal{C}' \leftarrow \mathcal{C}' \cup \{\boldsymbol{q}_l \otimes \boldsymbol{q}_m \mid \boldsymbol{q}_l \in \mathcal{C}_l, \boldsymbol{q}_m \in \mathcal{C}_m\}$
10      $\boldsymbol{x}_i, \boldsymbol{y}_i \leftarrow$ `SolveLP`$(\mathcal{C}', \boldsymbol{b}^{\otimes i})$
11      $R_i \leftarrow \|\boldsymbol{x}_i\|_1$
12      $\mathcal{C}_i \leftarrow$ columns $\{\boldsymbol{q}_j\} \subseteq \mathcal{C}'$ where $(\boldsymbol{x}_i)_j$ is nonzero.

13 **return** $(R_1, \ldots, R_n)$

Figure 9 shows the results of a numerical demonstration of the proposed algorithm applied to Haar random pure state, mixed state, and copies of pure magic state $|H\rangle$. Using the exact stabilizer decomposition up to $k = 7$ qubits, we have successfully computed the approximate RoM up to $n = 17$ for the pure and mixed random state. For $|H\rangle$, the compressed column set size $|\mathcal{C}_n|$ is significantly smaller so that we have reached $n = 21$. While this is not as large as $n = 26$ reported in Ref. [21], we emphasize that the present work is based on an algorithm that is agnostic to the internal symmetry of the single-qubit state. As a remark, the approximate RoM for copies of pure magic states $|H\rangle$ are almost identical to those presented in Ref. [21]; the value was at most 1.007 times larger.
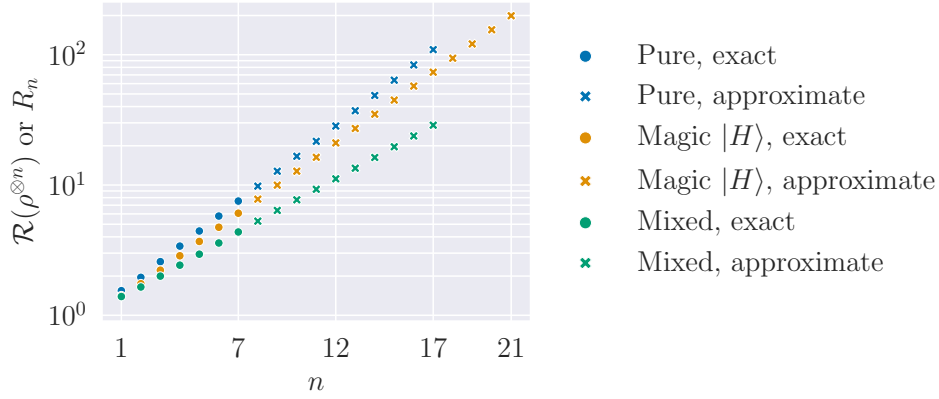
Figure 9: Exact $\mathcal{R}(\rho^{\otimes n})$ or approximate R_n computed in Algorithm 3. The blue, orange, and green data correspond to the Haar random pure state, the magic state $|H\rangle$, and the Haar random mixed state, respectively. The circle or crossed points indicate whether the solution is exact or approximate. The approximate values are computed from the exact solutions for $n \leq 7$ qubits.

4.2 Partially Disentangled States

Let us assume that we are interested in a general partially decoupled state of m subsystems as $\rho = \bigotimes_{i=1}^{m} \rho_i$, where $\sum_{i=1}^{m} n_i = n$ with n_i being the qubit count of i-th subsystem. Similar to the previous section, we may first compute the optimal solutions for each subsystem and then take tensor products over nonzero-weight stabilizers to construct the reduced basis for the total system. Although the computed value is not necessarily the exact RoM, this is an optimal solution in the following meaning. Refer to Appendix D for the proof.

Proposition 2. *Let ρ_i be given for the i-th subsystem, $\mathcal{R}(\rho_i)$ and $\boldsymbol{x}_i$ be the optimal solution and primal variable for $\mathtt{Prob}(\boldsymbol{A}_{n_i}, \boldsymbol{b}_i)$. Then,*

$$R = \prod_{i=1}^{m} \mathcal{R}(\rho_i), \quad \boldsymbol{x} = \bigotimes_{i=1}^{m} \boldsymbol{x}_i,$$

is one of the optimal solutions for $\mathtt{Prob}(\bigotimes_{i=1}^{m} \boldsymbol{A}_{n_i}, \boldsymbol{b})$.

By noting the submultiplicativity of RoM [19], i.e., $\mathcal{R}(\bigotimes_i \rho_i) \leq \prod_{i=1}^{m} \mathcal{R}(\rho_i) = R$, it is natural to expect that one can further improve the approximate RoM by combining some of the systems and solving it directly. For instance, one may group several subsystems so that each group consists of 6 or 7 qubits, compute the RoM for these systems using the CG method, and multiply them to obtain a better approximation of the RoM.

We find that it is more effective to consider multiple variations to divide subsystems. As shown in Algorithm 4, we divide subsystems into various kinds of groups. By comparing the product of those values, we can take the minimal value as the approximate RoM. For instance, we consider the case of a 15-qubit system that is decoupled into 5 subsystems as $\rho = \rho_1 \otimes \rho_2 \otimes \rho_3 \otimes \rho_4 \otimes \rho_5$, where each ρ_i is a 3-qubit state. One may compute the approximate value as $\mathcal{R}(\rho_1 \otimes \rho_2) \times \mathcal{R}(\rho_3 \otimes \rho_4) \times \mathcal{R}(\rho_5)$ or $\mathcal{R}(\rho_1) \times \mathcal{R}(\rho_2 \otimes \rho_3) \times \mathcal{R}(\rho_4 \otimes \rho_5)$ and take the minimal value as the approximate output. One could also speed up the computation by brute-force parallelization.

Algorithm 4: Optimization via Subsystem Division

Input: Target state $\rho = \bigotimes_{i=1}^m \rho_i$
Output: Approximate RoM

1 $R \leftarrow \prod_{i=1}^m \mathcal{R}(\rho_i)$
2 foreach Partition of a set $\{i\}_{i=1}^n$ to $\bigcup_j I_j$ **do**
3 $\quad\left\lfloor\ R \leftarrow \min\left(R, \prod_j \mathcal{R}(\bigotimes_{i\in I_j}\rho_i)\right)\right.$

4 return R

---

5 Discussion

In this work, we have introduced a systematic procedure for computing RoM. This procedure has proven highly effective, surpassing the state-of-the-art results for various quantum states, including random arbitrary states, multiple copies of single-qubit magic states, and partially disentangled quantum states. We have presented the core subroutine capable of computing the overlaps, algorithms to compute exact RoM for arbitrary quantum states, and algorithms to incorporate the nature of the target quantum state, such as the permutation symmetry between multiple copies of a state and the partially decoupled structure for inhomogeneous magic resources.

Numerous future directions can be envisioned. First, it is intriguing to seek generalization to other quantum resource measures. While the optimization strategy, namely the column generation technique, can be widely applied to L^p optimization problems, the overlap calculation in this work has heavily exploited the property of qubit stabilizer states. This means that, our technique may be applied to the case of robustness-like measures for multiqubit systems (such as channel robustness [25] and dyadic negativity [16]), whereas it is nontrivial whether there exists a fast overlap computation method for multiqudit case. Considering that generalized Pauli operators for qudits constitute a complete basis for operator space, we expect that algorithm similar to the Fast Walsh-Hadamard Transform (FWHT) is applicable, thus speeding up the evaluation of negativity in multiqudit systems [20]. We note that a recent work on stabilizer extent has shown a technique that does not rely on FWHT [39], which could be a promising direction for rank-based monotones. Furthermore, it is nontrivial if we can extend the framework when the pure free states constitute a continuous set, such as in the case of fermionic non-Gaussianity [40, 41, 42]. Second, it is interesting to investigate whether it is possible to scale further computations for weakly decoupled states such as tensor network states. While exact computation may require as costly calculation as in the generic case, we may perform approximate computation with an accuracy bound.

Acknowledgments

Note added.— Hantzko et al. [43], and we independently proposed Pauli decomposition algorithms that share the essence of recursively slicing and transforming a density matrix. Their first preprint was uploaded about two weeks before our first preprint was uploaded.

After T. Jones informed us of their Pauli decomposition algorithm along with a Python benchmark [44], we worked on the C++ benchmark shown in Appendix B.1.

References

[1] Daniel Gottesman. "The Heisenberg Representation of Quantum Computers" (1998). arXiv:quant-ph/9807006.

[2] Michael A. Nielsen and Isaac L. Chuang. "Quantum Computation and Quantum Information: 10th Anniversary Edition". Cambridge University Press. (2010).

[3] Sergei Bravyi and Alexei Kitaev. "Universal Quantum Computation with ideal Clifford gates and noisy ancillas". Physical Review A **71**, 022316 (2005).

[4] Daniel Litinski. "A Game of Surface Codes: Large-Scale Quantum Computing with Lattice Surgery". Quantum **3**, 128 (2019).

[5] Dominic Horsman, Austin G Fowler, Simon Devitt, and Rodney Van Meter. "Surface code quantum computing by lattice surgery". New Journal of Physics **14**, 123011 (2012).

[6] Austin G. Fowler and Craig Gidney. "Low overhead quantum computation using lattice surgery" (2019). arXiv:1808.06709.

[7] Victor Veitch, Christopher Ferrie, David Gross, and Joseph Emerson. "Negative quasi-probability as a resource for quantum computation". New Journal of Physics **14**, 113011 (2012).

[8] Craig Gidney and Martin Ekerå. "How to factor 2048 bit RSA integers in 8 hours using 20 million noisy qubits". Quantum **5**, 433 (2021).

[9] Joonho Lee, Dominic W. Berry, Craig Gidney, William J. Huggins, Jarrod R. McClean, Nathan Wiebe, and Ryan Babbush. "Even More Efficient Quantum Computations of Chemistry Through Tensor Hypercontraction". PRX Quantum **2**, 030305 (2021).

[10] Vera von Burg, Guang Hao Low, Thomas Häner, Damian S. Steiger, Markus Reiher, Martin Roetteler, and Matthias Troyer. "Quantum computing enhanced computational catalysis". Physical Review Research **3**, 033055 (2021).

[11] Nobuyuki Yoshioka, Tsuyoshi Okubo, Yasunari Suzuki, Yuki Koizumi, and Wataru Mizukami. "Hunting for quantum-classical crossover in condensed matter problems". npj Quantum Information **10**, 45 (2024).

[12] Sergey Bravyi, Graeme Smith, and John Smolin. "Trading classical and quantum computational resources". Physical Review X **6**, 021043 (2016).

[13] Emanuele Tirrito, Poetri Sonya Tarabunga, Gugliemo Lami, Titas Chanda, Lorenzo Leone, Salvatore F. E. Oliviero, Marcello Dalmonte, Mario Collura, and Alioscia Hamma. "Quantifying nonstabilizerness through entanglement spectrum flatness". Physical Review A: Atomic, Molecular, and Optical Physics **109**, L040401 (2024).

[14] Oliver Hahn, Alessandro Ferraro, Lina Hultquist, Giulia Ferrini, and Laura García-Álvarez. "Quantifying Qubit Magic Resource with Gottesman-Kitaev-Preskill Encoding". Physical Review Letters **128**, 210502 (2022).

[15] Tobias Haug, Soovin Lee, and M. S. Kim. "Efficient quantum algorithms for stabilizer entropies". Phys. Rev. Lett. **132**, 240602 (2024).

[16] James R. Seddon, Bartosz Regula, Hakop Pashayan, Yingkai Ouyang, and Earl T. Campbell. "Quantifying Quantum Speedups: Improved Classical Simulation From Tighter Magic Monotones". PRX Quantum **2**, 010345 (2021).

[17] Zi-Wen Liu and Andreas Winter. "Many-body quantum magic". PRX Quantum **3**, 020333 (2022).

[18] Lorenzo Leone, Salvatore F. E. Oliviero, and Alioscia Hamma. "Stabilizer Rényi Entropy". Physical Review Letters **128**, 050402 (2022).

[19] Mark Howard and Earl T. Campbell. "Application of a resource theory for magic states to fault-tolerant quantum computing". Physical Review Letters **118**, 090501 (2017).

[20] Hakop Pashayan, Joel J. Wallman, and Stephen D. Bartlett. "Estimating outcome probabilities of quantum circuits using quasiprobabilities". Physical Review Letters **115**, 070501 (2015).

[21] Markus Heinrich and David Gross. "Robustness of Magic and Symmetries of the Stabiliser Polytope". Quantum **3**, 132 (2019).

[22] Sergey Bravyi, Dan Browne, Padraic Calpin, Earl Campbell, David Gosset, and Mark Howard. "Simulation of quantum circuits by low-rank stabilizer decompositions". Quantum **3**, 181 (2019).

[23] Scott Aaronson and Daniel Gottesman. "Improved Simulation of Stabilizer Circuits". Physical Review A **70**, 052328 (2004).

[24] Héctor J. García, Igor L. Markov, and Andrew W. Cross. "On the geometry of stabilizer states". Quantum Info. Comput. **14**, 683–720 (2014).

[25] James Seddon and Earl Campbell. "Quantifying magic for multi-qubit operations". Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences **475**, 20190251 (2019).

[26] Gurobi Optimization, LLC. "Gurobi Optimizer Reference Manual" (2023).

[27] Steven Diamond and Stephen Boyd. "CVXPY: A python-embedded modeling language for convex optimization". Journal of Machine Learning Research **17**, 2909–2913 (2016).

[28] Akshay Agrawal, Robin Verschueren, Steven Diamond, and Stephen Boyd. "A rewriting system for convex optimization problems". Journal of Control and Decision **5**, 42–60 (2018).

[29] Fino and Algazi. "Unified Matrix Treatment of the Fast Walsh-Hadamard Transform". IEEE Transactions on Computers **C-25**, 1142–1146 (1976).

[30] Victor Kac and Pokman Cheung. "Quantum Calculus". Universitext. Springer. (2002).

[31] G.I. Struchalin, Ya. A. Zagorovskii, E.V. Kovlakov, S.S. Straupe, and S.P. Kulik. "Experimental Estimation of Quantum State Properties from Classical Shadows". PRX Quantum **2**, 010307 (2021).

[32] Hiroki Hamaguchi, Kou Hamada, and Nobuyuki Yoshioka. "RoM-handbook". GitHub Repository (2024).

[33] Gilbert Strang. "Linear algebra and learning from data". Wellesley-Cambridge Press. (2019).

[34] Guy Desaulniers, Jacques Desrosiers, and Marius M. Solomon Solomon, editors. "Column Generation". Springer US. (2005).

[35] SciPy. "scipy.sparse.csc_matrix". SciPy.

[36] William K Wootters and Brian D Fields. "Optimal state-determination by mutually unbiased measurements". Annals of Physics **191**, 363–381 (1989).

[37] Kathleen S. Gibbons, Matthew J. Hoffman, and William K. Wootters. "Discrete phase space based on finite fields". Physical Review A: Atomic, Molecular, and Optical Physics **70**, 062101 (2004).

[38] Danielsen Lars Eirik. "Database of self-dual quantum codes". link.

[39] Hiroki Hamaguchi, Kou Hamada, Naoki Marumo, and Nobuyuki Yoshioka. "Faster computation of stabilizer extent" (2024). arXiv:2406.16673.

[40] Beatriz Dias and Robert Koenig. "Classical simulation of non-Gaussian fermionic circuits". Quantum **8**, 1350 (2024).

[41] Oliver Reardon-Smith, Michał Oszmaniec, and Kamil Korzekwa. "Improved simulation of quantum circuits dominated by free fermionic operations" (2024). arXiv:2307.12702.

[42] Joshua Cudby and Sergii Strelchuk. "Gaussian decomposition of magic states for matchgate computations" (2023). arXiv:2307.12654.

[43] Lukas Hantzko, Lennart Binkowski, and Sabhyata Gupta. "Tensorized pauli decomposition algorithm". Physica Scripta **99**, 085128 (2024).

[44] Tyson Jones. "Densepaulidecomposer". GitHub Repository (2024).

[45] Tyson Jones. "Decomposing dense matrices into dense Pauli tensors" (2024). arXiv:2401.16378.

[46] Sebastián Vidal Romero and Juan Santos-Suárez. "PauliComposer: Compute tensor products of Pauli matrices efficiently". Quantum Information Processing **22**, 449 (2023).

[47] Hiroki Hamaguchi, Kou Hamada, and Nobuyuki Yoshioka. "paulidecomp". GitHub Repository (2024).

[48] Arno Jaeger and Bertram Mond. "On direct sums and tensor products of linear programs". Zeitschrift für Wahrscheinlichkeitstheorie und Verwandte Gebiete **3**, 19–31 (1964).

A   Pseudocode of Fast Walsh–Hadamard Transform Algorithm

Here, we provide the pseudocode of the Fast Walsh–Hadamard Transform (FWHT) algorithm, which computes $H_n v = H_n^\top v = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}^{\otimes n} v$ for $v \in \mathbb{R}^{2^n}$. We omitted the normalization factor in the pseudocode since the Walsh–Hadamard matrix H_n itself is unnormalized. The time complexity is $\mathcal{O}(n2^n)$ and the space complexity is $\mathcal{O}(2^n)$, since this is a in-place algorithm.

Algorithm 5: Fast Walsh–Hadamard Transform (FWHT) Algorithm

Input: $v \in \mathbb{R}^{2^n}$
Output: In-place computation result of $H_n v$

1 **Function** FWHT(v)
2 $\quad$ $h \leftarrow 1$
3 $\quad$ **while** $h < 2^n$ **do**
4 $\quad\quad$ **for** $i \leftarrow 0$ **to** $2^n - 2h$ **by** $2h$ **do**
5 $\quad\quad\quad$ **for** $j \leftarrow i$ **to** $i + h - 1$ **do**
6 $\quad\quad\quad\quad$ $x \leftarrow v_j$
7 $\quad\quad\quad\quad$ $y \leftarrow v_{j+h}$
8 $\quad\quad\quad\quad$ $v_j \leftarrow x + y$
9 $\quad\quad\quad\quad$ $v_{j+h} \leftarrow x - y$
10 $\quad\quad\quad$ **end**
11 $\quad\quad$ **end**
$\quad\quad$ // $v \leftarrow v/\sqrt{2}$, if normalize.
12 $\quad\quad$ $h \leftarrow 2h$
13 $\quad$ **end**

B Basic Properties of Pauli Vector Representation

This section reviews the basic properties of Pauli vector representation.

B.1 Complexity of Computing Pauli Vector Representation

Let ρ be an n-qubit quantum state whose Pauli vector representation is given by $b_j = \text{Tr}[P_j\rho]$ where P_j is the j-th Pauli operator. To compute all the elements, naively, the computational complexity scales as $\mathcal{O}(8^n)$ even if we use the sparse structure of each Pauli matrix. In the following, we show that we can perform an in-place computation that exponentially reduces the time complexity:

Lemma 5 (Complexity of computing Pauli vector). *Given the full density matrix representation of n-qubit quantum state ρ, its Pauli vector representation can be computed with time complexity of $\mathcal{O}(n4^n)$.*

Proof. First, let us introduce a map from an n-qubit density matrix to a $2n$-qubit state vector as follows:

$$\rho = \sum_{i,j} \rho_{i_1\cdots i_n, j_1\cdots j_n} |i_1\cdots i_n\rangle\langle j_1\cdots j_n| \mapsto \sum_{i,j} \rho_{i_1\cdots i_n, j_1\cdots j_n} |i_1, j_1, \ldots, i_n, j_n\rangle.$$

Note that this is different from the well-known Choi map $\rho \mapsto \sum_{i,j} \rho_{i,j} |i\rangle |j\rangle$, and thus we refer to it as modified Choi vectorization. We introduce a modified Choi vector $\boldsymbol{c}$. Let c_k denote the k-th element of $\boldsymbol{c}$, practically obtained via extracting the matrix elements of ρ in the Z-order curve. Then, we find that $\boldsymbol{c}$ is related with the Pauli vector representation $\boldsymbol{b}$ as

$$\boldsymbol{b} = T_n \boldsymbol{c} \quad \text{where} \quad T_n := \begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & i & -i & 0 \\ 1 & 0 & 0 & -1 \end{pmatrix}^{\otimes n}.$$

For example, for $n = 1$, we have

$$\rho = \begin{pmatrix} \rho_{1,1} & \rho_{1,2} \\ \rho_{2,1} & \rho_{2,2} \end{pmatrix}, \quad \boldsymbol{c} = \begin{pmatrix} \rho_{1,1} \\ \rho_{1,2} \\ \rho_{2,1} \\ \rho_{2,2} \end{pmatrix}, \quad \boldsymbol{b} = \begin{pmatrix} \rho_{1,1} + \rho_{2,2} \\ \rho_{1,2} + \rho_{2,1} \\ i\rho_{1,2} - i\rho_{2,1} \\ \rho_{1,1} - \rho_{2,2} \end{pmatrix} = \begin{pmatrix} \text{Tr}[\text{I}\rho] \\ \text{Tr}[\text{X}\rho] \\ \text{Tr}[\text{Y}\rho] \\ \text{Tr}[\text{Z}\rho] \end{pmatrix}.$$

Similar to the FWHT algorithm as provided in Algorithm 5, in-place computation for such a tensor-product structure can be done with time complexity of $\mathcal{O}(n4^n)$ and space complexity of $\mathcal{O}(4^n)$, which completes the proof. $\qquad\square$

We describe the numerical comparison of our Pauli decomposition algorithm with other related studies [43, 45, 46] which are available via GitHub repository [47]. The algorithms are implemented with C++ and run on the laptop. Our C++ implementation is mainly based on the Python implementation by Jones [44], except for the iterative algorithm by Hantzko et al. since it is not included in the Python implementation. The algorithms run on 50 random density matrices; each entry's real and imaginary parts are independently and uniformly sampled from $[0, 1)$.

Fig. 10 shows the benchmark of Pauli decomposition. It shows that our Pauli decomposition algorithm is the fastest. Although our algorithm and the algorithms by Hantzko et al. both have the best time complexity $\mathcal{O}(n4^n)$ among the algorithms, our algorithm is faster by a constant factor. We consider it is because our algorithm uses in-place computation, which is cache efficient.
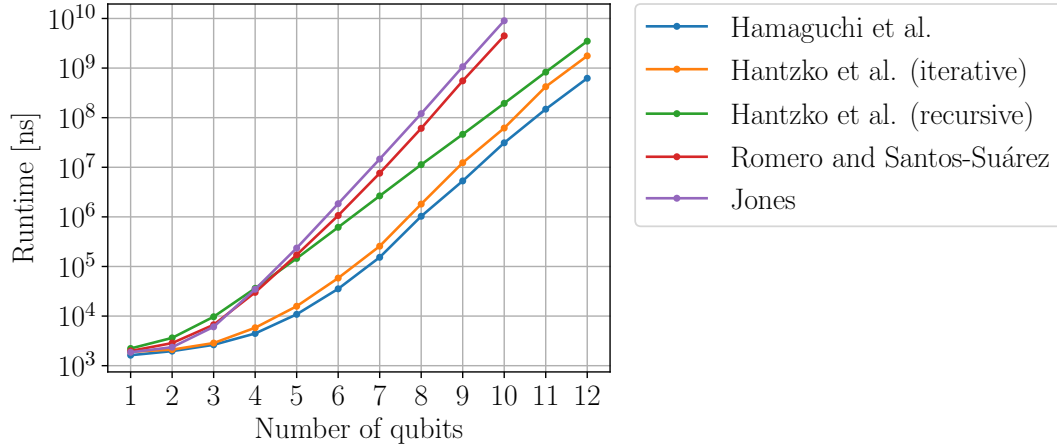
Figure 10: Comparison of Pauli decomposition algorithms. The algorithms are implemented with C++ and run on the laptop. The first three methods in the legend have a time complexity of $\mathcal{O}(n4^n)$, while the next two methods have a time complexity of $\mathcal{O}(8^n)$.

B.2 Bound on Overlap Counts

We provide a fact regarding the distribution of overlaps in addition to $0 \leq \boldsymbol{a}_j^\top \boldsymbol{b} \leq 2^n$.

Lemma 6 (Bound on overlap counts)**.** *Let ρ be an arbitrary n-qubit quantum state. Then, for all $n \in \mathbb{N}$, the count on the pure stabilizer states satisfies the following:*

$$\left|\{\boldsymbol{a}_j \in \mathcal{A}_n \mid \boldsymbol{a}_j^\top \boldsymbol{b} \in [0,1]\}\right| \geq |\mathcal{S}_n|/2^n, \ \left|\{\boldsymbol{a}_j \in \mathcal{A}_n \mid \boldsymbol{a}_j^\top \boldsymbol{b} \in [1,2^n]\}\right| \geq |\mathcal{S}_n|/2^n.$$

Proof. Consider an arbitrary sparsified Walsh–Hadamard matrix W_j in Lemma 4, and let $\mathcal{W} \subset \mathcal{A}_n$ denote its columns. It follows that

$$\sum_{\boldsymbol{a} \in \mathcal{W}} \boldsymbol{a}^\top \boldsymbol{b} = 2^n. \tag{7}$$

Therefore, assuming that the overlaps are either entirely $\boldsymbol{a}^\top \boldsymbol{b} < 1$ or entirely $\boldsymbol{a}^\top \boldsymbol{b} > 1$ contradicts Eq. (7). Applying the same reasoning for every W_j completes the proof. $\qquad\square$

C Proof of Proposition 1

In this section, we prove Proposition 1.

Proposition 1. *A cover set $\mathcal{V}_n$ with size $2^n + 1$ can be constructed.*

Let us describe the structure of the subsequent sections. Section C.1 defines MUBs, discusses the connection between the cover set and MUBs, and provides a short proof by using the previous result by Gibbons et al. [37]. For convenience, the subsequent sections provide a full proof of Proposition 1. The full proof shares the essence with Ref. [37], but dispenses with concepts and discussions by Ref. [37] that are not useful for our goal.

C.1 Connection between Cover Set and MUBs

MUBs are defined as bases of the state vector space that satisfy the following conditions: for any two different bases of MUBs, any state of one basis has the same amplitudes with respect to the other basis. Gibbons et al. [37, Section 4] constructs MUBs of maximal size $2^n + 1$ by using the following fact.

Lemma 7 (Partition of Pauli Operators [37, Section 4])**.** *The set of all non-identity n-qubit Pauli operators $\{I, X, Y, Z\}^{\otimes n} \setminus \{I^{\otimes n}\}$, which has $4^n - 1$ elements, can be partitioned into $2^n + 1$ subsets of $2^n - 1$ elements that commute with one another.*

Let $\mathcal{T}_1, \ldots, \mathcal{T}_{2^n+1}$ be the subsets of commuting Pauli operators in Lemma 7. Ref. [37] takes a basis B_i consisting of simultaneous eigenvectors of $\mathcal{T}_i$ and proves that $\{B_1, \ldots, B_{2^n+1}\}$ is MUBs of maximal size. Also, we can construct the cover set in Proposition 1 by using Lemma 7.

Proof. Regarding B_i as a column set of the matrix $\boldsymbol{A}_n$, the basis B_i corresponds to a sparsified Walsh–Hadamard matrix W_i. Then, the nonzero-row set $R(W_i)$ consists of $\mathcal{T}_i$ and the first row $I^{\otimes n}$ by the definition of B_i. Thus, Lemma 7 implies that $\mathcal{V}_n = \{W_1, \ldots, W_{2^n+1}\}$ is the desired cover set. $\square$

C.2 Full Proof

C.2.1 Proof on the Existence of Minimum Cover Stabilizer Set

Next, we directly prove Proposition 1. We utilize the bijection between Pauli operators $P \in \{I, X, Y, Z\}^{\otimes n}$ and $2n$-dimensional row vectors $r(P) \in \mathbb{F}_2^{2n}$ of check matrices. First, the proof of Lemma 4 shows that each sparsified Walsh–Hadamard matrix W corresponds to a single check matrix $C = [X_n \ Z_n]$. Also, the elements of the nonzero-row set $R(W)$ can be enumerated as follows: for each $\boldsymbol{v} \in \mathbb{F}_2^n$, take the Pauli operator $P_{\boldsymbol{v}}$ satisfying $r(P_{\boldsymbol{v}}) = \boldsymbol{v}^\top C = [\boldsymbol{v}^\top X_n \ \boldsymbol{v}^\top Z_n]$. Using these observations, it suffices to find a set of check matrices $\{C_1, \ldots, C_{2^n+1}\}$ that satisfies the following condition: for every $2n$-dimensional row vector $\boldsymbol{r} \in \mathbb{F}_2^{2n}$, there exists a check matrix C_k in the set and a vector $\boldsymbol{v} \in \mathbb{F}_2^n$ that satisfies $\boldsymbol{r} = \boldsymbol{v}^\top C_k$. In the following, we explicitly construct the check matrices and confirm that they satisfy the condition.

Let us define the $2^n + 1$-st check matrix as $C_{2^n+1} = [I \ O]$. Also, for each k from 1 to 2^n, we take the k-th check matrix $C_k = [X_k \ Z_k]$ as follows:

1. Z_k is an $n \times n$ identity matrix.

2. X_k is given in the following lemma.

Lemma 8. *There exists an explicit construction for a set of symmetric matrices $\{X_k \mid X_k \in \mathbb{F}_2^{n \times n}\}_{k=1}^{2^n}$ such that the following is satisfied.*

$$\forall \boldsymbol{v} \in \mathbb{F}_2^n \setminus \{\boldsymbol{0}\}, \ \{X_k \boldsymbol{v}\}_{k=1}^{2^n} = \mathbb{F}_2^n. \tag{8}$$

From the fact that X_k and Z_k are all symmetric matrices, it follows that the condition in Lemma 2 is satisfied, and therefore the check matrices $C_1, \ldots, C_{2^n+1}$ are valid.

We show that the constructed check matrices indeed satisfy the above condition. Let us take any $2n$-dimensional row vector $\boldsymbol{r}$. Let $\boldsymbol{x}$ and $\boldsymbol{z}$ be the former and latter half of the row vector $\boldsymbol{r}$, respectively. If $\boldsymbol{z} = \boldsymbol{0}$, then $C_{2^n+1} = [I \ O]$ and $\boldsymbol{v} = \boldsymbol{x}$ satisfy the condition. If $\boldsymbol{z} \neq 0$, Lemma 8 allows us to take k such that $\boldsymbol{z} X_k = \boldsymbol{x}$. Using $\boldsymbol{v} = \boldsymbol{z}$ implies $\boldsymbol{v} C_k = [\boldsymbol{z} X_k \ \boldsymbol{z}] = \boldsymbol{r}$.

C.3 Proof of Technical Lemma 8

Now the remaining work is to prove Lemma 8. We first provide the explicit construction of $\{X_k\}_{k=1}^{2^n}$, and then prove that it indeed satisfies Eq. (8).

C.3.1 Construction of X_k

We first introduce some algebraic concepts necessary for the discussion. We denote the polynomial ring over $\mathbb{F}_2$ by $\mathbb{F}_2[x]$. Let f be an arbitrary irreducible polynomial of degree n. We consider a quotient ring $\mathbb{F}_2[x]/(f)$, where (f) denotes the ideal generated by f. Then, $\mathbb{F}_2[x]/(f)$ is a field because f is irreducible. It is also noteworthy that $\mathbb{F}_2[x]/(f)$ is a vector space over $\mathbb{F}_2$ and $\{x^0, \ldots, x^{n-1}\}$ can be taken as a basis.

In what follows, we discuss the construction of $\{X_k\}_{k=1}^{2^n}$. We define a symmetric matrix $C(x) \in (\mathbb{F}_2[x]/(f))^{n \times n}$ as a matrix whose (i,j) entry equals x^{i+j-2}. Namely, $C(x)$ can be represented as follows:

$$
C(x) = \begin{pmatrix}
x^0 & x^1 & x^2 & \cdots & x^{n-1} \\
x^1 & x^2 & & & \\
x^2 & & \ddots & & \vdots \\
\vdots & & & & \\
x^{n-1} & & \cdots & & x^{2n-2}
\end{pmatrix}.
$$

Every entry of $C(x)$ can be represented as a linear combination of a basis $\{x^0, \ldots, x^{n-1}\}$, and we define C_i be a matrix consisting of such x^i coefficients. In other words, $C_i \in \mathbb{F}_2^{n \times n}$ is defined so that $C(x) = C_0 x^0 + \cdots + C_{n-1} x^{n-1}$ holds. Note that C_i is also symmetric.

We give a concrete example below. We take $n = 3$ and $f = 1 + x + x^3$, which is irreducible. Then, C_0, C_1, C_2 can be derived in the following way:

$$
\begin{aligned}
C(x) &= \begin{pmatrix} x^0 & x^1 & x^2 \\ x^1 & x^2 & x^3 \\ x^2 & x^3 & x^4 \end{pmatrix} \\
&= \begin{pmatrix} 1 & x & x^2 \\ x & x^2 & 1+x \\ x^2 & 1+x & x+x^2 \end{pmatrix} \\
&= \underbrace{\begin{pmatrix} 1 & & \\ & & 1 \\ & 1 & \end{pmatrix}}_{C_0} x^0 + \underbrace{\begin{pmatrix} & & 1 \\ 1 & & 1 \\ & 1 & 1 \end{pmatrix}}_{C_1} x^1 + \underbrace{\begin{pmatrix} & & 1 \\ & 1 & \\ 1 & & 1 \end{pmatrix}}_{C_2} x^2.
\end{aligned}
$$

Next, we consider the following set of symmetric matrices:

$$
\left\{ \sum_{i=0}^{n-1} a_i C_i \,\middle|\, a_i \in \mathbb{F}_2 \right\}.
$$

Since the elements of the set are distinct, the set has 2^n elements. We take this set as the set $\{X_k\}_{k=1}^{2^n}$.

C.3.2 Proof that X_k Satisfies Lemma 8

Next, we prove that $\{X_k\}_{k=1}^{2^n}$ given in the previous subsection satisfies Eq. (8).

By the definition of X_k, one can show that Eq. (8) holds if and only if $\{C_i \boldsymbol{v}\}_{i=0}^{n-1}$ is linearly independent for any $\boldsymbol{v} \in \mathbb{F}_2^n \setminus \{\boldsymbol{0}\}$. From here, we show the linear independence of $\{C_i \boldsymbol{v}\}_{i=0}^{n-1}$ by proving several lemmas.

Lemma 9. *For any vectors $\boldsymbol{u}, \boldsymbol{v} \in \mathbb{F}_2^n \setminus \{\boldsymbol{0}\}$, $\boldsymbol{u}^\top C(x)\boldsymbol{v} \neq 0$.*

Proof. Using a vector $\boldsymbol{x} = (x^0, \dots, x^{n-1})^\top$, we have $C(x) = \boldsymbol{x}\boldsymbol{x}^\top$. Thus, by defining two polynomials $u(x) = \boldsymbol{u}^\top \boldsymbol{x} = \sum_{i=0}^{n-1} u_i x^i$ and $v(x) = \boldsymbol{v}^\top \boldsymbol{x} = \sum_{i=0}^{n-1} v_i x^i$, $\boldsymbol{u}^\top C(x)\boldsymbol{v}$ can be represented as $u(x)v(x)$. Hence, it suffices to show that $u(x)v(x) \neq 0$. Because $\boldsymbol{u}$ and $\boldsymbol{v}$ are nonzero, $u(x)$ and $v(x)$ are nonzero as well. Noting that $\mathbb{F}_2/(f)$ is a field, the product $u(x)v(x)$ is also nonzero. $\qquad\square$

Lemma 10. *For any vectors $\boldsymbol{u}, \boldsymbol{v} \in \mathbb{F}_2^n \setminus \{\boldsymbol{0}\}$, there exists i such that $\boldsymbol{u}^\top C_i \boldsymbol{v} \neq 0$.*

Proof. By multiplying $C(x) = C_0 x^0 + \cdots + C_{n-1} x^{n-1}$ by $\boldsymbol{u}$ from the left and $\boldsymbol{v}$ from the right, we obtain

$$\boldsymbol{u}^\top C(x)\boldsymbol{v} = (\boldsymbol{u}^\top C_0 \boldsymbol{v})x^0 + \cdots + (\boldsymbol{u}^\top C_{n-1}\boldsymbol{v})x^{n-1}. \tag{9}$$

Eq. (9) expresses $\boldsymbol{u}^\top C(x)\boldsymbol{v}$ as a polynomial with coefficients $\boldsymbol{u}^\top C_i \boldsymbol{v} \in \mathbb{F}_2$. Since $\boldsymbol{u}^\top C(x)\boldsymbol{v}$ is nonzero from Lemma 9, there exists a nonzero coefficient, i.e, $\boldsymbol{u}^\top C_i \boldsymbol{v} \neq 0$ for some i. $\qquad\square$

Lemma 11. *For any vector $\boldsymbol{v} \in \mathbb{F}_2^n \setminus \{\boldsymbol{0}\}$, $\{C_i \boldsymbol{v}\}_{i=0}^{n-1}$ is linearly independent.*

Proof. We consider a matrix $[C_0 \boldsymbol{v}, \dots, C_{n-1}\boldsymbol{v}]$ with $C_i \boldsymbol{v}$ as the column vectors. By multiplying an arbitrary nonzero vector $\boldsymbol{u} \in \mathbb{F}_2^n \setminus \{\boldsymbol{0}\}$ from the left, we obtain a vector $(\boldsymbol{u}^\top C_0 \boldsymbol{v}, \dots, \boldsymbol{u}^\top C_{n-1}\boldsymbol{v})$, which is nonzero by Lemma 10. Therefore, we can confirm that the matrix $[C_0 \boldsymbol{v}, \dots, C_{n-1}\boldsymbol{v}]$ is non-singular, which implies the linear independence of $\{C_i \boldsymbol{v}\}_{i=0}^{n-1}$. $\qquad\square$

Having shown Lemma 11, it is also proved that $\{X_k\}_{k=1}^{2^n}$ given in the previous subsection satisfies Eq. (8), i.e., Lemma 8 is proved.

D Proof of Proposition 2

In this section, we prove Proposition 2.

Proposition 2. *Let ρ_i be given for the i-th subsystem, $\mathcal{R}(\rho_i)$ and $\boldsymbol{x}_i$ be the optimal solution and primal variable for $\mathtt{Prob}(\boldsymbol{A}_{n_i}, \boldsymbol{b}_i)$. Then,*

$$R = \prod_{i=1}^{m} \mathcal{R}(\rho_i), \quad \boldsymbol{x} = \bigotimes_{i=1}^{m} \boldsymbol{x}_i,$$

is one of the optimal solutions for $\mathtt{Prob}(\bigotimes_{i=1}^{m} \boldsymbol{A}_{n_i}, \boldsymbol{b})$.

This assures that when the stabilizer set is restricted only to the tensor product states $\bigotimes_i \boldsymbol{A}_{n_i} \subset \boldsymbol{A}_n$, the optimal solution is simply a tensor product of individual optimal solutions. The proof can be similarly done with [48]. We are considering the following primal problem:

$$\begin{aligned}
\text{(P) } \underset{\boldsymbol{x}}{\text{minimize}} \quad & \|\boldsymbol{x}\|_1 \\
\text{subject to} \quad & \left(\bigotimes_{i=1}^{m} \boldsymbol{A}_{n_i}\right)\boldsymbol{x} = \bigotimes_{i=1}^{m} \boldsymbol{b}_i,
\end{aligned}$$

and the i-th primal problem is formulated as

$$(\text{P}_i) \quad \underset{\boldsymbol{x}_i}{\text{minimize}} \quad \|\boldsymbol{x}_i\|_1$$
$$\text{subject to} \quad \boldsymbol{A}_{n_i}\boldsymbol{x}_i = \boldsymbol{b}_i.$$

The proposition asserts that if there exist optimal solutions $\boldsymbol{x}_i^*$ for every (P_i), then $\boldsymbol{x}^* = \bigotimes_{i=1}^m \boldsymbol{x}_i^*$ is the optimal solution for (P). Let OPT be the optimal value for the original problem (P). It is clear that $\boldsymbol{x}^*$ is a feasible solution for (P), which means $\text{OPT} \le \|\boldsymbol{x}^*\|_1$. The dual problem of (P) is given as

$$(\text{D}) \quad \underset{\boldsymbol{y}}{\text{maximize}} \quad \left(\bigotimes_{i=1}^m \boldsymbol{b}_i^\top\right)\boldsymbol{y}$$
$$\text{subject to} \quad \left\|\left(\bigotimes_{i=1}^m \boldsymbol{A}_{n_i}^\top\right)\boldsymbol{y}\right\|_\infty \le 1,$$

and the i-th dual problem can also be formulated as

$$(\text{D}_i) \quad \underset{\boldsymbol{y}_i}{\text{maximize}} \quad \boldsymbol{b}_i^\top \boldsymbol{y}_i$$
$$\text{subject to} \quad \left\|\boldsymbol{A}_{n_i}^\top \boldsymbol{y}_i\right\|_\infty \le 1.$$

Now the strong duality of the problems assures that optimal solution $\boldsymbol{y}_i^*$ exists for all (D_i) satisfying $\boldsymbol{b}_i^\top \boldsymbol{y}_i^* = \|\boldsymbol{x}_i\|_1$. By taking $\boldsymbol{y}^* := \bigotimes_{i=1}^m \boldsymbol{y}_i^*$, it follows from the property of L^∞ norm that

$$\left\|\left(\bigotimes_{i=1}^m \boldsymbol{A}_{n_i}^\top\right)\boldsymbol{y}^*\right\|_\infty = \left\|\bigotimes_{i=1}^m \left(\boldsymbol{A}_{n_i}^\top \boldsymbol{y}_i^*\right)\right\|_\infty = \prod_{i=1}^m \left\|\boldsymbol{A}_{n_i}^\top \boldsymbol{y}_i^*\right\|_\infty \le 1,$$

which guarantees that $\boldsymbol{y}^*$ is a feasible solution of (D). Thus, the optimal value of the dual problem (D), which is OPT by the strong duality, is no less than

$$\left(\bigotimes_{i=1}^m \boldsymbol{b}_i^\top\right)\boldsymbol{y}^* = \prod_{i=1}^m \left(\boldsymbol{b}_i^\top \boldsymbol{y}_i^*\right) = \prod_{i=1}^m \|\boldsymbol{x}_i^*\|_1 = \left\|\bigotimes_{i=1}^m \boldsymbol{x}_i^*\right\|_1 = \|\boldsymbol{x}^*\|_1,$$

which means $\|\boldsymbol{x}^*\|_1 \le \text{OPT}$, i.e., $\|\boldsymbol{x}^*\|_1 = \text{OPT}$. This completes the proof of Proposition 2.

E   Numerical Details on RoM Calculation

In this section, we provide details on the numerical results of RoM calculation.

E.1   More Results on Top-Overlap Method

Here, we provide the results of a numerical experiment results regarding the top-overlap method introduced in Sec. 3.2. We present results for Haar random mixed, pure, and tensor product states of $n = 4, 5, 6, 7$ qubit system.

As shown in Fig. 11, we can see that the top-overlap method significantly outperforms a naive random selection method in all cases. The improvement becomes more evident when we compute larger systems; in particular, for $n = 7$ qubit case, it suffices to take only $K = 10^{-5}$ to reach near-optimal value for all three targets.

Two remarks are in order. First, we added the column set of the cover matrix in Sec. 3.4 to ensure feasibility. In other words, the restriction of the column set solely using

the information of overlaps may lead to a rank-deficient matrix. Second, the run time of our algorithm for a $n = 6$ qubit system was 5 seconds for computing all the overlaps and 3 minutes for solving the LP. For a $n = 7$ qubit system, the overlap computation consumes 15 minutes at most and 15 minutes for solving the LP.

E.2 Assuring feasibility

As mentioned in the previous subsection, we cannot obtain any feasible solution if the stabilizer states are not appropriately restricted. One of the most robust ways to ensure the feasibility for arbitrary quantum states is to utilize the cover matrix as mentioned before. The second method applicable for tensor product states is to utilize the solution obtained from small-scale systems. As shown in Appendix D, we can always obtain a feasible solution by taking tensor products, which can be used as an initial solution. Then, one may extend the set of stabilizers to improve the quality of the solution.
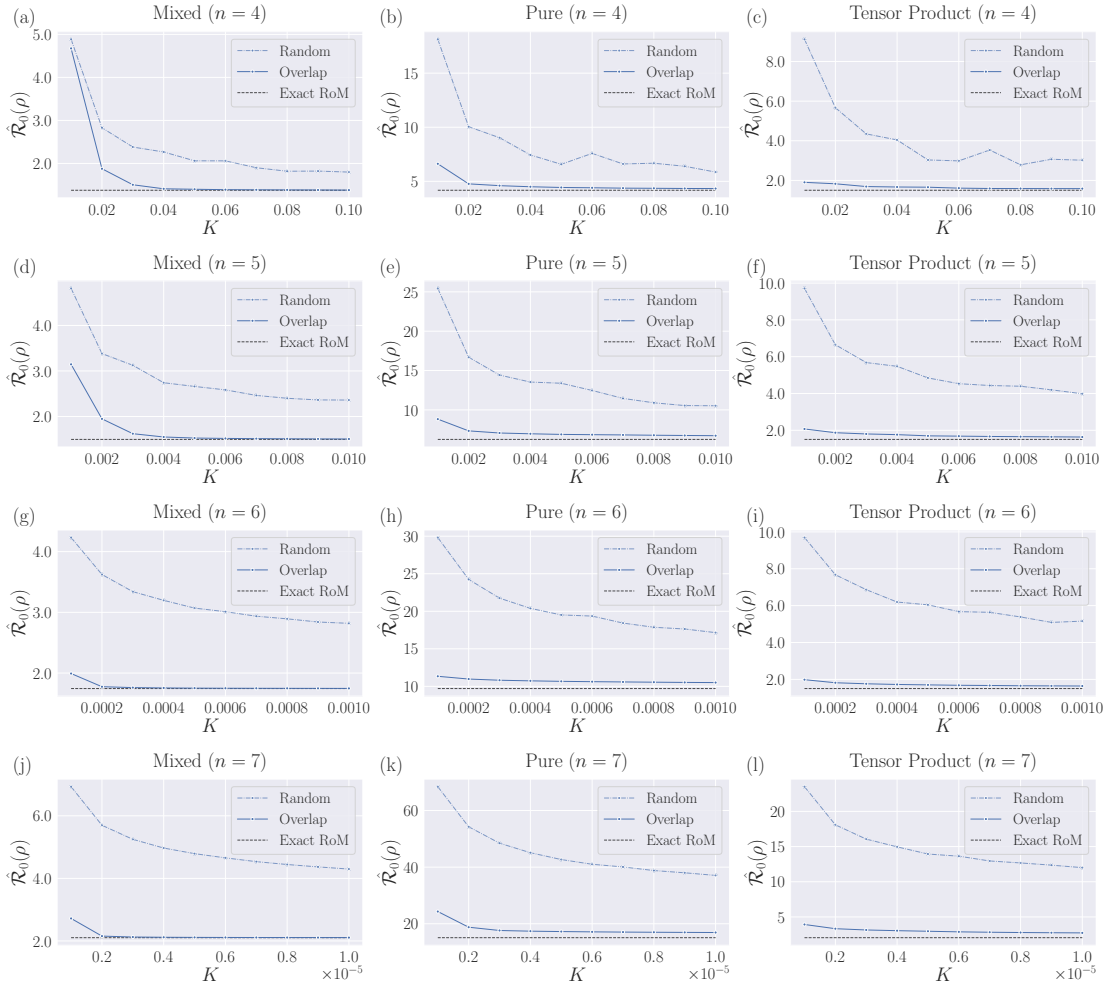

Figure 11: Numerical demonstration of top-overlap method introduced in Sec. 3.2. The column set is restricted to the size of $K|\mathcal{S}_n|$ ($0 < K \leq 1$). The cyan and blue lines denote the approximate RoM $\hat{\mathcal{R}}_k(\rho)$ computed with a randomly restricted column set or the columns with the largest or smallest overlaps, respectively. The black dotted lines indicate the exact RoM computed with the CG method.