Classical and Quantum Algorithms for Characters of the Symmetric Group

Sergey Bravyi,[1] David Gosset,[2,3,*] Vojtech Havlicek,[1] and Louis Schatzki[4]

[1] *IBM Quantum, IBM T.J. Watson Research Center, Yorktown Heights, New York 10598, USA*

[2] *Department of Combinatorics and Optimization and Institute for Quantum Computing, University of Waterloo*

[3] *Perimeter Institute for Theoretical Physics*

[4] *University of Illinois Urbana Champaign, Urbana, Illinois 61801, USA*

(Received 20 March 2025; accepted 25 June 2025; published 7 August 2025)

Characters of irreducible representations are ubiquitous in group theory. However, computing characters of some groups such as the symmetric group S_n is a challenging problem known to be #P-hard in the worst case. Here we describe a matrix product state (MPS) algorithm for characters of S_n. The algorithm computes an MPS encoding all irreducible characters of a given permutation. It relies on a mapping from characters of S_n to quantum spin chains proposed by Crichigno and Prakash. We also provide a simpler derivation of this mapping. We complement this result by presenting a poly(n) size quantum circuit that prepares the corresponding MPS obtaining an efficient quantum algorithm for certain sampling problems based on characters of S_n. To assess classical hardness of these problems, we present a general reduction from strong simulation (computing a given probability) to weak simulation (sampling with a small error). This reduction applies to any sampling problem with a certain granularity structure and may be of independent interest.

DOI: 10.1103/bq28-r2r7

I. INTRODUCTION

Representation theory provides a powerful set of tools for analyzing systems invariant under the action of some group such as a molecular symmetry group in chemistry or a gauge group in quantum field theory. For example, the eigenvalue spectrum of a quantum system is usually computed by decomposing the Hilbert space into a direct sum of irreducible representations of the symmetry group and diagonalizing each irreducible block of the Hamiltonian separately.

The character of an irreducible representation (irrep) can be viewed as a fingerprint that uniquely identifies the irrep and facilitates many computational tasks. In the case of Abelian groups such as the symmetry group of a periodic lattice, characters are basis functions of the Fourier transform associated with a given frequency or momentum. In general, the character of a group element g is defined as the trace of a matrix describing the action of g for a given irrep.

Here we address the problem of computing characters of the symmetric group S_n, that is, the group of all possible permutations of n objects. Symmetric groups S_n with small values of n often emerge as molecular symmetries in chemistry. For example, S_4 and S_5 describe molecules with tetrahedral and icosahedral symmetries such as methane and fullerene. Group S_6 is isomorphic to the symplectic group describing the action of two-qubit Clifford gates commonly used in quantum computation [1]. Moreover, Cayley's theorem asserts that any finite group is isomorphic to a subgroup of S_n [2].

Unfortunately, the irreducible characters of S_n lack a simple analytic formula. In fact, it was shown [3,4] that the problem of computing the character of a given permutation $g \in S_n$ for a given irrep of S_n is #P-hard, that is, at least as hard as counting the number of solutions of an NP-hard problem. While this leaves little hope for an efficient algorithm with runtime scaling polynomially with n, one may ask how well can we do in practice and whether it is possible to compute the characters of S_n for moderately large values of n, say $n \leq 100$.

To motivate this problem, we note that the characters of S_n with large values of n describe amplitudes of Laughlin states in the fractional quantum Hall effect [5–8]. Having access to the characters of S_n enabled computation of the electron density and correlation functions of Laughlin states [5]. Another application is computing integrals of high-degree polynomials over the unitary group [9]. Such

*Contact author: dgosset@uwaterloo.ca

Published by the American Physical Society under the terms of the Creative Commons Attribution 4.0 International license. Further distribution of this work must maintain attribution to the author(s) and the published article's title, journal citation, and DOI.

integrals can be computed using Weingarten functions that depend on the S_n characters [9]. Algorithms for S_n characters may also facilitate evaluation of kernel functions for solving ranking problems in machine learning [10,11].

Our first contribution is an algorithm for computing the characters of S_n based on matrix product states (MPSs). These are tensor networks commonly used for simulating ground-state properties of quantum spin chains [12]. A deep connection between the characters of S_n and quantum spin chains was recently established by Crichigno and Prakash [13]. These authors showed that the characters of a given permutation $g \in S_n$ associated with all possible irreps of S_n can be read off as amplitudes of a certain quantum state $|\psi_g\rangle$ describing $2n$ fermionic modes. State $|\psi_g\rangle$ is defined in terms of current operators

$$J_\ell = \sum_{i=1}^{2n-\ell} a_{i+\ell}^\dagger a_i,$$

where ℓ is an integer and the $a_i^\dagger$ (a_i) are fermionic creation (annihilation) operators. Specifically,

$$|\psi_g\rangle = \prod_{j=1}^{c(g)} J_{\nu_j} |1^n 0^n\rangle,$$

where $c(g)$ is the number of cycles in g, ν_j is the length of the jth cycle, and $|1^n 0^n\rangle$ is the Fock basis state with the first n modes occupied and remaining modes empty. The order in the product does not matter since the current operators pairwise commute [13]. For each irrep λ of S_n, the Crichigno and Prakash mapping gives a Fock basis vector $x_\lambda \in \{0,1\}^{2n}$ such that the amplitude $\langle x_\lambda|\psi_g\rangle$ coincides with the character of g associated with λ. Section III provides a simplified derivation of the Crichigno and Prakash mapping. Our derivation may be slightly more accessible to physicists as it avoids the heavy machinery of noncommutative symmetric polynomials used in the original work [13,14]. Instead, we apply an elegant formula derived by Frobenius in 1900 [15] to express the characters in terms of fermionic Slater determinants. Rewriting this formula using the fermionic second quantization formalism reproduces the result of Ref. [13].

The standard Jordan Wigner transformation maps $2n$ fermionic modes to $2n$ qubits. We show that the image of the current operator J_ℓ under this mapping is a matrix product operator (MPO) with bond dimension $\ell + 2$. Applying a sequence of MPOs to an initial product state is a common subroutine in quantum spin chain simulations and a variety of highly optimized software libraries for this task is available such as quimb [16] or mpnum [17]. Our problem is particularly well suited for MPO-MPS simulation because the characters of S_n are integer valued. Accordingly, a small error introduced by truncating singular values in the intermediate MPS can be tolerated. A detailed description of our algorithm and numerical experiments can be found in Sec. IV.

As mentioned above, computing S_n characters is a #P-hard problem. Thus, neither classical nor quantum algorithms for this problem are expected to run in time poly(n). The main obstacle for the classical MPS algorithm is a rapid growth of entanglement that results in an exponentially large MPS bond dimension. A natural question to ask is whether a quantum computer can efficiently prepare state $|\psi_g\rangle$. Surprisingly, we show that the answer is yes. To this end, we construct a quantum circuit of size poly(n) that prepares the uniform superposition of all permutations in the conjugacy class of g. We show that applying the quantum Fourier transform (QFT) over S_n to this superposition gives the desired state $|\psi_g\rangle$, with a suitable normalization coefficient. Importantly, the QFT over S_n can be realized by a quantum circuit of size poly(n), as was shown by Beals [18].

Using a similar technique, we describe a poly(n)-sized quantum circuit that, for a given irrep λ of S_n, prepares a quantum state $|\omega_\lambda\rangle$ that is a superposition of all permutations $g \in S_n$ with amplitudes proportional to the character of g for irrep λ. By preparing and then measuring either $|\psi_g\rangle$ or $|\omega_\lambda\rangle$, a quantum computer can efficiently *sample* either irreps or permutations with probability proportional to the character squared.

Finally, we ask whether or not these sampling problems admit efficient classical algorithms. To this end, we give a reduction from weak simulation (sampling with a small statistical error) to strong simulation (exact computation of probabilities) for a broad class of sampling problems that posses a certain granularity structure. Namely, if P is a probability distribution on some set Ω, we say that P is *granular* if $|\Omega| \cdot P(x)$ is integer for all $x \in \Omega$. Our weak-to-strong reduction applies to any granular (or approximately granular) distribution and does not require the anticoncentration property that is commonly used in reductions from weak-to-strong simulation [19–21]. Sampling problems based on characters of S_n are shown to be (approximately) granular due to the fact that S_n characters are integer valued. Using our reduction, we show that classical hardness of sampling from the measured distribution of $|\psi_g\rangle$ for permutations g with $O(\sqrt{n}/\log(n))$ cycles follows from a conjecture concerning average-case hardness of computing the group characters associated with g. Establishing or refuting this conjecture is left for future work.

Similarly to characters of the symmetric group, we give an MPS algorithm for Kostka numbers. The first step of mapping Kostka numbers onto quantum spin chains was shown in Ref. [13]. We give a corresponding MPO construction and code for running this algorithm in the final section of this paper.

Most of the existing classical algorithms for computing S_n characters use variants of a recursive formula known as

the Murnaghan-Nakayama rule [22,23]. Algorithmic versions of this formula can be found in, e.g., Refs. [3,24]. An efficient classical algorithm that computes a small additive error approximation of S_n characters normalized by the irrep dimension is described by Jordan [25]. Classical and quantum algorithms for computing multiplicity coefficients closely related to S_n characters were studied in Refs. [26,27].

The paper is organized as follows. Section II provides a necessary background on group representation theory. Section III gives a simplified derivation of the Crichigno and Prakash mapping from characters of S_n to quantum spin chains. Sections IV and V contain our main results—an MPS algorithm for characters of S_n and a quantum circuit that prepares the corresponding MPS. Sampling problems for granular probability distributions and reductions from weak-to-strong simulation are discussed in Sec. VI. An extension of our MPS algorithm to computing Kostka numbers is given in Sec. VII.

II. GROUP THEORY REFRESHER

Suppose that G is a finite group. A d-dimensional representation of G is a group homomorphism from G to the unitary group $\mathbb{U}(d)$. A representation is *irreducible* if the ambient Hilbert space $\mathbb{C}^d$ contains no proper subspace invariant under the action of all group elements. It is standard to identify irreps that differ only by a basis change in the ambient Hilbert space. Then any finite group has a finite set of irreps. Given a group element $g \in G$ and an irrep λ, let $\rho_\lambda(g)$ be the unitary matrix corresponding to g. The *character* of g associated with an irrep λ is defined as a complex number $\chi_\lambda(g) = \mathrm{Tr}[\rho_\lambda(g)]$. The fact that ρ_λ is a group homomorphism implies that $\chi_\lambda(g)$ depends only on the conjugacy class of g, that is, $\chi_\lambda(g) = \chi_\lambda(hgh^{-1})$ for all $g, h \in G$. Organizing all characters into a matrix with rows labeled by irreps and columns labeled by conjugacy classes of G gives the character table of G.

The symmetric group S_n is the group of permutations of n objects, say integers $[n] = \{1, 2, \ldots, n\}$. Irreps and conjugacy classes of S_n are labeled by partitions of n, that is, sorted sequences of positive integers $\lambda = (\lambda_1, \lambda_2, \ldots, \lambda_m)$ such that $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_m$ and $\sum_{j=1}^m \lambda_j = n$. Here m is the number of parts in λ. We write $\lambda \vdash n$ to indicate that λ is a partition of n. A permutation $g \in S_n$ is said to be a cycle of length ℓ if there exists a subset $J \subseteq [n]$ such that $|J| = \ell$ and g cyclically shifts elements of J while acting trivially on $[n] \setminus J$. Any permutation $g \in S_n$ can be uniquely written as a composition of cycles with pairwise disjoint supports. Denoting the number of cycles by $c(g)$ and the length of the jth longest cycle by ν_j one obtains a partition of n with $c(g)$ parts, namely, $\nu = (\nu_1, \nu_2, \ldots, \nu_{c(g)})$. The partition ν specifies the conjugacy class of S_n that contains g. Irreps of S_n are also labeled by partitions of n. For example, a partition $\lambda = (n)$ with a single part labels the trivial representation, while a partition $\lambda = (1, 1, \ldots, 1)$ with n parts labels the sign representation. The general recipe for constructing the unitary action ρ_λ for a given partition λ can be found in Ref. [28]. The explicit forms of matrices $\rho_\lambda(g)$ are irrelevant for our purposes since we are only interested in the characters $\chi_\lambda(g) = \mathrm{Tr}(\rho_\lambda(g))$.

III. BUILDING THE CHARACTER

We begin by reproducing the result of Crichigno and Prakash [13] that shows how to build a quantum spin chain encoding all characters $\chi_\lambda(g)$ of a given permutation $g \in S_n$.

Let $\lambda = (\lambda_1, \ldots, \lambda_m)$ and $\nu = (\nu_1, \ldots, \nu_{c(g)})$ be the partitions of n describing the irrep λ and the conjugacy class of g. Consider a complex vector $z \in \mathbb{C}^m$ and a polynomial

$$P_\nu(z) = \prod_{j=1}^{c(g)} \left(\sum_{k=1}^{m} z_k^{\nu_j} \right). \tag{1}$$

The Frobenius character formula [15] (see, e.g., Theorem 4.47 of Ref. [28]) states that $\chi_\lambda(g)$ coincides with the coefficient of a monomial

$$z(\lambda) = \prod_{j=1}^{m} z_j^{\lambda_j + m - j} \tag{2}$$

in the polynomial

$$Q(z) = P_\nu(z) \prod_{1 \leq i < j \leq m} (z_i - z_j). \tag{3}$$

Let $\mathbb{S}^1 = \{x \in \mathbb{C} : |x| = 1\}$ be the unit circle in the complex plane, and let $\mathcal{F}_1$ be a Hilbert space of functions $f : \mathbb{S}^1 \to \mathbb{C}$ with inner product

$$\langle f | g \rangle = \frac{1}{2\pi} \int_0^{2\pi} d\theta f(e^{i\theta})^* g(e^{i\theta})$$

and an orthonormal basis of functions $\{f(x) = x^k\}_{k \in \mathbb{Z}}$. Informally, $\mathcal{F}_1$ describes a particle moving on the circle and $f(x) = x^k$ is a state with angular momentum k. Consider now a many-body system composed of m indistinguishable particles moving on the circle that obey fermionic statistics. The fermionic first quantization principle stipulates that the m-particle Hilbert space $\mathcal{F}_m$ is the antisymmetric subspace of $\mathcal{F}_1^{\otimes m}$ and its orthonormal basis is formed by Slater determinants composed of m distinct single-particle basis functions. Such Slater determinants

have the form $|\kappa\rangle = D_\kappa(z)/\sqrt{m!}$, where

$$D_\kappa(z) = \det \begin{bmatrix} z_1^{\kappa_1} & z_1^{\kappa_2} & \cdots & z_1^{\kappa_m} \\ z_2^{\kappa_1} & z_2^{\kappa_2} & \cdots & z_2^{\kappa_m} \\ \vdots & \vdots & & \vdots \\ z_m^{\kappa_1} & z_m^{\kappa_2} & \cdots & z_m^{\kappa_m} \end{bmatrix},$$

and κ runs over strictly decreasing integer sequences of length m that represent momenta of the occupied single-particle states. The Vandermonde determinant formula gives

$$\prod_{1\le i<j\le m} (z_i - z_j) = \sqrt{m!}\,|\tau\rangle, \tag{4}$$

where

$$\tau = (m-1, m-2, \ldots, 1, 0).$$

Let $P_\nu : \mathcal{F}_m \to \mathcal{F}_m$ be a linear operator that maps a function $\Psi(z) \in \mathcal{F}_m$ to $P_\nu(z)\Psi(z)$. Note that P_ν preserves $\mathcal{F}_m$ since $P_\nu(z)$ is a symmetric polynomial. We conclude that polynomial $Q(z)$ defined in Eq. (3) describes a fermionic m-particle (unnormalized) state

$$|Q\rangle = \sqrt{m!}\,P_\nu|\tau\rangle. \tag{5}$$

The coefficient of the monomial $z(\lambda)$ in $Q(z)$ coincides with $\langle D_{\lambda+\tau}|Q\rangle/m!$, where

$$\lambda + \tau = (\lambda_1 + \tau_1, \ldots, \lambda_m + \tau_m).$$

Indeed, $Q(z)$ is antisymmetric under the exchange of any pair of variables, while $D_{\lambda+\tau}(z)/m!$ is the antisymmetrized version of the monomial $z(\lambda)$ defined in Eq. (2). Thus, the Frobenius character formula can be expressed as

$$\chi_\lambda(g) = \langle D_{\lambda+\tau}|Q\rangle/m! = \langle \lambda+\tau|P_\nu|\tau\rangle. \tag{6}$$

The next step is to rewrite Eq. (6) using the fermionic second quantization formalism. To this end, consider an infinite chain of qubits labeled by integers $k \in \mathbb{Z}$. Let X_k, Y_k, Z_k be the Pauli operators acting on the kth qubit. Define a fermionic annihilation operator

$$a_k = \frac{1}{2}(X_k + iY_k)\prod_{j<k} Z_j, \tag{7}$$

and let $a_k^\dagger$ be the corresponding creation operator. Here we used the Jordan-Wigner fermion-to-qubit mapping. The operators $a_k, a_k^\dagger$ obey the Fermi-Dirac commutation rules, that is, $a_i a_j = -a_j a_i$ and $a_i a_j^\dagger + a_j^\dagger a_i = \delta_{i,j} I$ for all i, j. Let $\mathcal{H}$ be the Hilbert space of the chain. It has basis vectors $|x\rangle$, where $x \in \{0,1\}^{\mathbb{Z}}$ is a bit string with a finite support.

Given a finite strictly decreasing integer sequence κ, let $x_\kappa \in \{0,1\}^{\mathbb{Z}}$ be a bit string with support κ. The Slater determinant basis state $|\kappa\rangle = D_\kappa(z)/\sqrt{m!}$ in the first quantization is represented by state $|x_\kappa\rangle \in \mathcal{H}$ in the second quantization.

Multiplication of a wave function by the polynomial $\sum_{j=1}^m z_j^\ell$ in the first quantization picture shifts the momentum of each particle by ℓ. Such a shift is described by an operator

$$J_\ell = \sum_{k\in\mathbb{Z}} a_{k+\ell}^\dagger a_k \tag{8}$$

in the second quantization picture. Following Ref. [13], we refer to J_ℓ as the current operator. Now the Frobenius character formula can be expressed as

$$\chi_\lambda(g) = \langle x_{\lambda+\tau}| \prod_{j=1}^{c(g)} J_{\nu_j} |x_\tau\rangle. \tag{9}$$

The order in the product does not matter since the current operators pairwise commute. At this point we have mapped S_n characters to an infinite spin chain. We can truncate the chain to a block of $2n$ consecutive qubits using the following lemma.

Lemma 1. Suppose that $x, y \in \{0,1\}^{\mathbb{Z}}$ are bit strings and that i^* is an integer such that $x_i = y_i$ for $i \le i^*$. Let x', y' be bit strings obtained from x, y, respectively, by setting to zero all bits $i \le i^*$. Then

$$\langle y| \prod_{j=1}^{c(g)} J_{\nu_j} |x\rangle = \langle y'| \prod_{j=1}^{c(g)} J_{\nu_j} |x'\rangle. \tag{10}$$

Proof. Let $\tilde{J}_\ell = \sum_{k>i^*} a_{k+\ell}^\dagger a_k$. One can easily check that

$$\prod_{j=1}^{c(g)} J_{\nu_j} |x\rangle = \prod_{j=1}^{c(g)} \tilde{J}_{\nu_j} |x\rangle + |\phi\rangle,$$

where $|\phi\rangle \in \mathcal{H}$ is a superposition of basis vectors $|z\rangle$ such that $z_i = 0$ and $x_i = 1$ for at least one qubit $i \le i^*$ (in the case that no sites $i \le i^*$ are occupied, $|\phi\rangle$ is 0). By assumption, $x_i = y_i$ for all $i \le i^*$. Thus, $|\phi\rangle$ is orthogonal to $|y\rangle$, that is,

$$\langle y| \prod_{j=1}^{c(g)} J_{\nu_j} |x\rangle = \langle y| \prod_{j=1}^{c(g)} \tilde{J}_{\nu_j} |x\rangle. \tag{11}$$

Since $\tilde{J}_{\nu_j}$ acts trivially on qubits $i \le i^*$, we also have

$$\langle y| \prod_{j=1}^{c(g)} \tilde{J}_{\nu_j} |x\rangle = \langle y'| \prod_{j=1}^{c(g)} \tilde{J}_{\nu_j} |x'\rangle. \tag{12}$$

Applying Eq. (11) again with x', y' instead of x, y proves Eq. (10). ■

As a consequence of Lemma 1, the matrix element in Eq. (9) does not change if we set $m = n$ and pad λ with $n - m$ zeros on the right. Accordingly, we can set

$$\tau = (n-1, n-2, \ldots, 1, 0).$$

It is immediate that any qubit $i < 0$ or $i \geq 2n$ is set to $|0\rangle$ in the initial state $|x_\tau\rangle$. Note that $\prod_{j=1}^{c(g)} J_{\nu_j}$ can only move a fermion at most n sites to the right. Thus, qubits $i < 0$ and $i \geq 2n$ are unchanged in the simulation and can be removed. This leaves only $2n$ active qubits $i = 0, 1, \ldots, 2n-1$. After this truncation the initial basis state $|x_\tau\rangle$ is mapped to $|1^n 0^n\rangle$, while the final basis state $x_{\lambda+\tau}$ is mapped to a weight-n bit string with support $\{\lambda_1 + n - 1, \lambda_2 + n - 2, \ldots, 0\}$ [29]. Now the Frobenius character formula becomes

$$\chi_\lambda(g) = \langle x_{\lambda+\tau}| \prod_{j=1}^{c(g)} J_{\nu_j} |1^n 0^n\rangle, \tag{13}$$

where

$$J_\ell = \sum_{k=0}^{2n-1-\ell} a_{k+\ell}^\dagger a_k. \tag{14}$$

This concludes the derivation of the Crichigno and Prakash mapping from characters of S_n to a quantum spin chain.

As pointed out in Ref. [13], state $|\psi_g\rangle = \prod_{j=1}^{c(g)} J_{\nu_j} |1^n 0^n\rangle$ has support only on bit strings encoding partitions of n. This can be seen as follows. Let $x = (x_0, x_1, \ldots, x_{2n-1})$ be a bit string of length $2n$. Let $|x| = \sum_{j=0}^{2n-1} x_j$ be the weight of x and $m(x) = \sum_{j=0}^{2n-1} j x_j$ be the momentum of x. For example, $m(1^n 0^n) = \sum_{j=0}^{n-1} j = n(n-1)/2$. We claim that x encodes a partition of n if and only if x has weight n and momentum $n(n+1)/2$. Indeed, suppose that $x = x_{\lambda+\tau}$ for some partition $\lambda \vdash n$. Then x has weight n since we agreed that λ has n parts (including the padded zero parts). By definition, x has support $\{\lambda_j + n - j\}_{j=1,\ldots,n}$. From $\sum_{j=1}^n \lambda_j = n$ one gets

$$m(x) = \sum_{j=1}^n (\lambda_j + n - j) = n(n+1)/2.$$

Conversely, suppose that x has weight n and momentum $n(n+1)/2$. Let $2n - 1 \geq d_1 > d_2 > \cdots > d_n \geq 0$ be the indices of nonzero bits of x. Set $\lambda_j = j + d_j - n$ for $j = 1, \ldots, n$. From $d_j > d_{j-1}$ one gets $n \geq \lambda_1 \geq \cdots \geq \lambda_n \geq 0$. Also, we have $\sum_{j=1}^n d_j = m(x)$, that is,

$$\sum_{j=1}^n \lambda_j = m(x) - n(n-1)/2 = n.$$

Thus, λ is a partition of n.

Clearly, the current operators J_{ν_j} preserve the weight of bit strings. Each application of J_{ν_j} increases the momentum by ν_j. Since the $\sum_{j=1}^{c(g)} \nu_j = n$ and the initial bit string $1^n 0^n$ has momentum $n(n-1)/2$, the final state is a superposition of bit strings with weight n and momentum $n(n-1)/2 + n = n(n+1)/2$. As shown above, these are exactly the bit strings encoding partitions of n.

IV. CLASSICAL MPS ALGORITHMS

Consider a permutation $g \in S_n$. Let $c(g)$ be the number of cycles in g and ν_j be the length of the jth cycle. Our strategy is to represent the $2n$-qubit state

$$|\psi_g\rangle = \prod_{j=1}^{c(g)} J_{\nu_j} |1^n 0^n\rangle \tag{15}$$

by an MPS. To this end, we need the following lemma.

Lemma 2. The current operator J_ℓ is an MPO with bond dimension $\ell + 2$.

Proof. Figure 1 defines a tensor $T \in \mathbb{C}^2 \otimes \mathbb{C}^{\ell+2} \otimes \mathbb{C}^{\ell+2} \otimes \mathbb{C}^2$, where $\mathbb{C}^2$ represents physical indices and $\mathbb{C}^{\ell+2}$ represents virtual indices of the MPO. Taking $2n$ copies of T and contracting all virtual indices (horizontal lines) as shown in Fig. 1 gives an operator

$$\sum_{j=0}^{2n-1-\ell} \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}_j Z_{j+1} \cdots Z_{j+\ell-1} \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix}_{j+\ell}.$$

This is the current operator defined in Eqs. (7) and (8). ■

The initial state $|1^n 0^n\rangle$ is clearly an MPS with bond dimension 1. Define a sequence of MPSs $|\phi_0\rangle = |1^n 0^n\rangle$ and

$$|\phi_j\rangle = \text{Compress}(J_{\nu_j} |\phi_{j-1}\rangle, \epsilon) \tag{16}$$

for $j = 1, \ldots, c(g)$. Here $\text{Compress}(|\phi\rangle, \epsilon)$ is a function that computes the canonical form of an MPS $|\phi\rangle$ and truncates the singular values (Schmidt coefficients) at each site, as discussed in Sec. 4.5.1 of Ref. [30]. The error tolerance $\epsilon \geq 0$ defines the maximum relative error incurred while truncating singular values. More precisely, suppose that $s_1 \geq s_2 \geq \cdots \geq s_N$ are singular values of some MPS

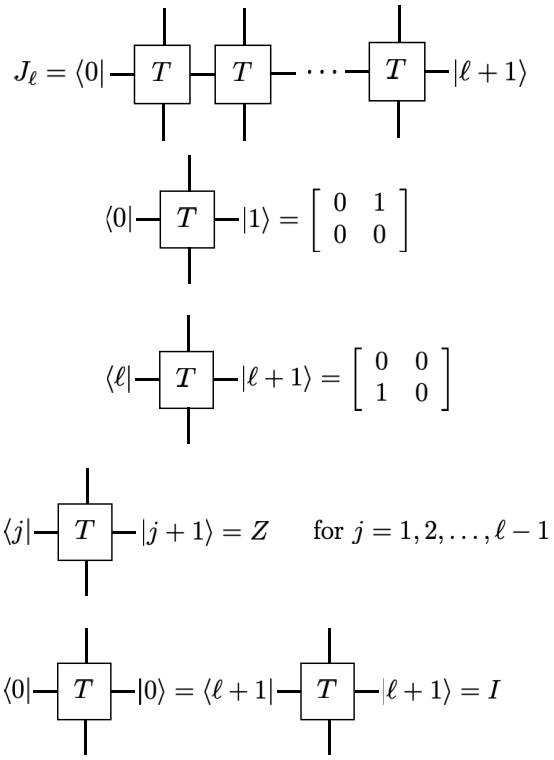


FIG. 1. MPO representation of the current operator J_ℓ. Horizontal lines represent virtual indices of dimension $\ell+2$. We label basis vectors of $\mathbb{C}^{\ell+2}$ by integers $\{0, 1, \ldots, \ell+1\}$. The MPO consists of $2n$ copies of T. We list all nonzero components of tensor T.

$|\phi_j\rangle$ for some bipartite cut of the chain. We discard singular values s_i with $i \geq d$, where d is the smallest index such that

$$\sum_{i=d+1}^{N} s_i \leq \epsilon \sum_{i=1}^{N} s_i.$$

We chose to impose a bound on the relative truncation error since the considered states are unnormalized and some singular values s_i can be exponentially large. Let $D = D(\epsilon)$ be the maximum bond dimension of all states $|\phi_j\rangle$ with $j \leq c(g)$. Lemma 2 ensures that

$$D \leq \prod_{j=1}^{c(g)} (\nu_j + 2). \tag{17}$$

We do not expect this bound to be tight, even for lossless compression, $\epsilon = 0$. For example, one can easily check that a state $J_{\nu_1}|1^n 0^n\rangle$ has a bond dimension of only ν_1 instead of $\nu_1 + 2$ predicted by Eq. (17). We empirically observed that the MPS bond dimension D at the compression error $\epsilon = 10^{-10}$ can be smaller than the upper bound (17) by several orders of magnitude; see Fig. 3 below.

Since the cost of MPO-MPS multiplication and MPS compression scales as $O(nD^3)$, an MPS representation of $|\psi_g\rangle$ can be computed in time $O(nc(g)D^3) = O(n^2D^3)$. Let

$$|\mathsf{mps}_g\rangle = |\phi_{c(g)}\rangle$$

be the final MPS approximating $|\psi_g\rangle$. The character of any irrep $\lambda \vdash n$ can then be extracted by computing a single amplitude of the final MPS,

$$\chi_\lambda(g) = \langle x_{\lambda+\tau}|\psi_g\rangle \approx \langle x_{\lambda+\tau}|\mathsf{mps}_g\rangle. \tag{18}$$

This takes time $O(nD^3)$.

Suppose that one needs to compute MPS amplitudes $\langle x_{\lambda+\tau}|\psi_g\rangle$ for all irreps λ. To speedup this computation, we divided the spin chain into four blocks of length $n/2$ and cached products of the MPS tensors associated with each block. As one iterates over irreps λ, a product of the MPS tensors is retrieved from the cache, if available. We observed that the runtime for computing MPS amplitudes with the caching strategy described above is comparable or smaller than the runtime for computing the MPS tensors.

Recall that S_n characters are integer valued. Thus, the exact value of $\chi_\lambda(g)$ can be recovered by rounding the final MPS amplitude to the nearest integer as long as

$$|\chi_\lambda(g) - \langle x_{\lambda+\tau}|\mathsf{mps}_g\rangle| < \tfrac{1}{2}. \tag{19}$$

An error introduced by the compression can be estimated by comparing the norm of the exact state $|\psi_g\rangle$ and the approximating MPS. The orthogonality theorem for characters [28] gives

$$\|\psi_g\|^2 = \sum_{\lambda \vdash n} \chi_\lambda(g)^2 = |E_g|, \tag{20}$$

where $E_g = \{h \in S_n \colon hg = gh\}$ is the centralizer of g. Denoting the number of length-ℓ cycles in g by a_ℓ, one gets

$$|E_g| = \prod_{\ell=1}^{n} (a_\ell!)\ell^{a_\ell}. \tag{21}$$

We used the mpnum software [17] for numerical experiments and set the compression error to $\epsilon = 10^{-10}$. The runtime of the MPS algorithm is compared with the runtime of two widely used computer algebra systems: GAP [31] and the symmetrica library included in SAGE [32]; see Fig. 2.

For numerical experiments, we selected permutations with a large number of short cycles that tend to be most expensive for all considered algorithms. Namely, for each n, we consider a single permutation $g \in S_n$ composed of $n/2$ length-2 cycles. Figure 2 shows the measured runtime for $10 \leq n \leq 30$. The reported runtimes include computation of the characters $\chi_\lambda(g)$ for all irreps $\lambda \vdash n$. In the

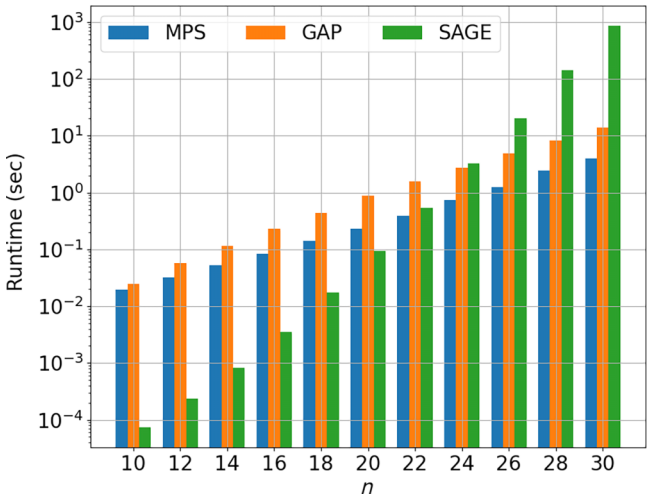


FIG. 2. Runtime for computing the characters $\chi_\lambda(g)$ for permutations $g \in S_n$ composed of $n/2$ length-2 cycles and all irreps $\lambda \vdash n$. The MPS compression error is set to $\epsilon = 10^{-10}$ for all the data.

case of the MPS the runtime additionally includes computing state $|\mathsf{mps}_g\rangle$. As one can see from the plot, the MPS algorithm shows a competitive performance for large values of n. We note that our implementation is in PYTHON, while SAGE and GAP have a kernel written in C, which tends to be much faster than PYTHON.

Recall that our MPS algorithm computes character $\chi_\lambda(g)$ with the zero error as long as the MPS amplitude $\langle x_{\lambda+\tau}|\mathsf{mps}_g\rangle$ obeys Eq. (19). We found that

$$|\chi_\lambda(g) - \langle x_{\lambda+\tau}|\mathsf{mps}_g\rangle| \leq 10^{-6}$$

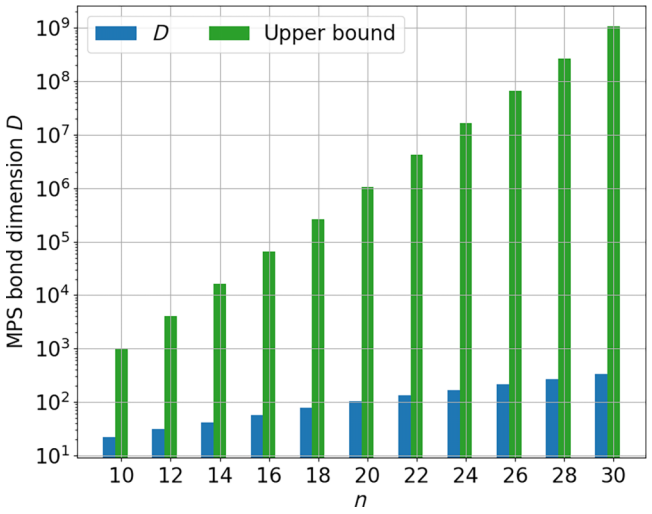


FIG. 3. Comparison between the observed maximum MPS bond dimension for the problem instances reported in Fig. 2 (blue) and the upper bound (17) (green).

for all problem instances reported at Fig. 2 [the exact value of $\chi_\lambda(g)$ was computed using the SAGE library]. This is many orders of magnitude smaller than the required precision $1/2$, suggesting that the MPS runtime may be improved further by truncating more singular values. Our PYTHON implementation of the MPS algorithm is available from GitHub [33]. We provide two versions of the algorithm that use the mpnum or quimb libraries for MPO-MPS manipulations.

Finally, for each problem instance presented in Fig. 2, we computed the maximum MPS bond dimension and compared it with the upper bound (17). This comparison is reported in Fig. 3.

V. QUANTUM ALGORITHM

Here we describe a quantum circuit of size poly(n) that prepares a state $|\psi_g\rangle/\||\psi_g\|$; see Eq. (15). Our circuit applies the quantum Fourier transform over group S_n to a certain subset state constructed in the following lemma.

Lemma 3. Let $g \in S_n$ be a permutation and $C_g \subseteq S_n$ be the conjugacy class of g. There exists a quantum circuit of size poly(n) that prepares a subset state

$$|C_g\rangle = \frac{1}{\sqrt{|C_g|}} \sum_{h \in C_g} |h\rangle. \tag{22}$$

We defer the proof till the end of this section. Recall that a QFT over the symmetric group S_n is defined as a unitary operator QFT_n with input Hilbert space $\mathbb{C}[S_n] = \mathrm{span}(|g\rangle : g \in S_n)$ and output Hilbert space $\bigoplus_{\lambda \vdash n} \mathbb{C}^{d_\lambda} \otimes \mathbb{C}^{d_\lambda}$ such that

$$\mathrm{QFT}_n|g\rangle = \sum_{\lambda \vdash n} \sqrt{\frac{d_\lambda}{n!}} \sum_{i,j=1}^{d_\lambda} \langle i|\rho_\lambda(g)|j\rangle |\lambda, i, j\rangle \tag{23}$$

for all permutations $g \in S_n$.

Here d_λ is the dimension of λ. The inverse QFT acts as

$$\mathrm{QFT}_n^\dagger|\lambda, i, j\rangle = \sqrt{\frac{d_\lambda}{n!}} \sum_{g \in S_n} \langle i|\rho_\lambda(g)|j\rangle^* |g\rangle. \tag{24}$$

It is known [18] that QFT_n can be implemented by a quantum circuit of size poly(n). Applying QFT_n to the subset state $|C_g\rangle$ gives a state

$$\begin{aligned} |\theta_g\rangle &= \mathrm{QFT}_n|C_g\rangle \\ &= \sum_{\lambda \vdash n} \sqrt{\frac{d_\lambda}{n!}} \sum_{i,j=1}^{d_\lambda} \frac{1}{\sqrt{|C_g|}} \sum_{h \in C_g} \langle i|\rho_\lambda(h)|j\rangle |\lambda, i, j\rangle. \end{aligned} \tag{25}$$

Since $\sum_{h\in C_g} h$ is a central element of the group algebra $\mathbb{C}[S_n]$, we infer that

$$\sum_{h\in C_g} \rho_\lambda(h) = bI_{d_\lambda} \tag{26}$$

for some complex number $b \in \mathbb{C}$. Here I_{d_λ} is the identity matrix of size d_λ. Taking the trace of Eq. (26) gives

$$b = \frac{|C_g|\chi_\lambda(g)}{d_\lambda}. \tag{27}$$

Substituting Eqs. (26) and (27) into Eq. (25) and using the identity $|\lambda, i, j\rangle = |\lambda\rangle \otimes |i, j\rangle$ shows that

$$|\theta_g\rangle = \sqrt{\frac{|C_g|}{n!}} \sum_{\lambda\vdash n} \chi_\lambda(g)|\lambda\rangle \otimes |\phi_\lambda\rangle,$$

where

$$|\phi_\lambda\rangle = \frac{1}{\sqrt{d_\lambda}} \sum_{i=1}^{d_\lambda} |i, i\rangle.$$

Let $\mathcal{U}$ be a unitary operator that maps $|\lambda\rangle \otimes |0\rangle$ to $|\lambda\rangle \otimes |\phi_\lambda\rangle$. One can realize $\mathcal{U}$ by a quantum circuit of size poly(n), for example using the techniques of Ref. [34], since the dimension d_λ can be computed in time poly(n) using the well-known hook length formula [35].

Applying $\mathcal{U}^{-1}$ to state $|\theta_g\rangle$ and tracing out the second register gives a state

$$|\omega_g\rangle = \sqrt{\frac{|C_g|}{n!}} \sum_{\lambda\vdash n} \chi_\lambda(g)|\lambda\rangle = \frac{1}{\sqrt{|E_g|}} \sum_{\lambda\vdash n} \chi_\lambda(g)|\lambda\rangle,$$

where E_g is the centralizer of g; see Eq. (21). This a normalized version of the desired state $|\psi_g\rangle$ if one encodes a partition $\lambda \vdash n$ by a bit string $x_{\lambda+\tau}$, as discussed in Sec. III. By measuring state $|\omega_g\rangle$ in the standard basis one can sample an irrep $\lambda \vdash n$ from a probability distribution

$$P_g(\lambda) = \frac{(\chi_\lambda(g))^2}{|E_g|} \qquad \text{(row sampling)}. \tag{28}$$

This is equivalent to sampling a row of the character table of S_n with the probability proportional to the character squared.

A similar method can prepare a superposition of all permutations $g \in S_n$ with amplitudes proportional to $\chi_\lambda(g)$ for a given irrep $\lambda \vdash n$. Indeed, apply $\mathrm{QFT}_n^\dagger$ to an initial state

$$|\theta_\lambda\rangle = \frac{1}{\sqrt{d_\lambda}} \sum_{i=1}^{d_\lambda} |\lambda, i, i\rangle.$$

This gives a state

$$|\omega_\lambda\rangle = \mathrm{QFT}_n^\dagger|\theta_\lambda\rangle = \frac{1}{\sqrt{n!}} \sum_{g\in S_n} \chi_\lambda(g)^*|g\rangle$$

(the complex conjugate can be omitted since S_n characters are real). By measuring state $|\omega_\lambda\rangle$ in the standard basis one can sample a permutation $g \in S_n$ from a probability distribution

$$P_\lambda(g) = \frac{(\chi_\lambda(g))^2}{n!} \qquad \text{(column sampling)}. \tag{29}$$

This is equivalent to sampling a column of the character table of S_n with the probability proportional to the character squared (here we assume that columns are labeled by permutations rather than conjugacy classes). We discuss the classical hardness of these sampling problems in the next section.

It is tempting to use our access to row and column sampling to estimate the characters themselves via repeated sampling. However, column sampling fails to yield efficient algorithms. To see this, note that $\chi_\lambda(g)^2 \le d_\lambda^2 \le e^{-C\sqrt{n}}n!$ ($C > 0$ is a constant), where the second inequality follows from known asymptotics [36]. Thus, the probabilities in Eq. (29) are always superpolynomially small in n and difficult to estimate. Could row sampling yield efficient estimates of characters? All characters could be estimated when $|E_g|$ is polynomial in n. This holds, for example, when $c(g) = O(1)$. But, in this case, the MPS algorithm of Sec. IV computes all characters in time polynomial in n. One could look for cases where both $|E_g|$ and $\chi_\lambda(g)^2$ are large; a task we leave to future work. We note that similar approaches were considered for computing multiplicities in tensor product representations [27,37].

Proof. Let $k = |C_g|$ and

$$C_g = \{h_1, h_2, \ldots, h_k\}.$$

Suppose that one can construct permutations $f_1, f_2, \ldots, f_k \in S_n$ such that

$$h_j = f_j g f_j^{-1} \tag{30}$$

for all $j = 1, \ldots, k$ and that there exists a quantum circuit $\mathcal{Q}$ of size poly(n) such that

$$\mathcal{Q}|h_j\rangle \otimes |0\rangle = |h_j\rangle \otimes |f_j\rangle \tag{31}$$

for all $j = 1, \ldots, k$. Then the desired state $|C_g\rangle$ can be prepared as follows. First prepare a state

$$|\psi_1\rangle = \frac{1}{\sqrt{n!}} \sum_{e \in S_n} |e\rangle \otimes |g\rangle.$$

Apply a classical reversible circuit that maps $|e\rangle \otimes |g\rangle$ to $|e\rangle \otimes |ege^{-1}\rangle$. This gives a state

$$|\psi_2\rangle = \frac{1}{\sqrt{n!}} \sum_{e \in S_n} |e\rangle \otimes |ege^{-1}\rangle.$$

Let E_g be the centralizer of g,

$$E_g = \{b \in S_n : gb = bg\}.$$

Since left cosets of E_g in S_n can be identified with elements of C_g, any permutation $e \in S_n$ can be uniquely decomposed as $e = fb$ for some $f \in \{f_1, \ldots, f_k\}$ and some $b \in E_g$. Thus,

$$|\psi_2\rangle = \frac{1}{\sqrt{k|E_g|}} \sum_{j=1}^{k} \sum_{b \in E_g} |f_j b\rangle \otimes |f_j g f_j^{-1}\rangle.$$

Here we noted that $n! = k|E_g|$. Tensoring in the third register prepared in the $|0\rangle$ state and applying circuit (31) to the second and third registers gives a state

$$|\psi_3\rangle = \frac{1}{\sqrt{k|E_g|}} \sum_{j=1}^{k} \sum_{b \in E_g} |f_j b\rangle \otimes |f_j g f_j^{-1}\rangle \otimes |f_j\rangle.$$

Next apply a classical reversible circuit that maps $|f_j b\rangle \otimes |f_j\rangle$ to $|b\rangle \otimes |f_j\rangle$ to the first and third registers of $|\psi_3\rangle$. This gives a state

$$|\psi_4\rangle = \frac{1}{\sqrt{k|E_g|}} \sum_{j=1}^{k} \sum_{b \in E_g} |b\rangle \otimes |f_j g f_j^{-1}\rangle \otimes |f_j\rangle.$$

Now the first register is left in a pure state $(1/\sqrt{|E_g|}) \sum_{b \in E_g} |b\rangle$. Tracing out the first register gives a state

$$|\psi_5\rangle = \frac{1}{\sqrt{k}} \sum_{j=1}^{k} |f_j g f_j^{-1}\rangle \otimes |f_j\rangle.$$

Applying the inverse of circuit (31) gives a state

$$|\psi_6\rangle = \frac{1}{\sqrt{k}} \sum_{j=1}^{k} |f_j g f_j^{-1}\rangle \otimes |0\rangle.$$

Finally, tracing out the second register we obtain a state

$$|\psi_7\rangle = \frac{1}{\sqrt{k}} \sum_{j=1}^{k} |f_j g f_j^{-1}\rangle = \frac{1}{\sqrt{|C_g|}} \sum_{h \in C_g} |h\rangle = |C_g\rangle.$$

Here the second equality makes use of Eq. (30) and the equality $C_g = \{h_1, h_2, \ldots, h_k\}$.

It remains to construct permutations $f_1, \ldots, f_k$ and a quantum circuit $\mathcal{Q}$ satisfying Eqs. (30) and (31). Suppose that g consists of a_ℓ cycles of length ℓ, where a_ℓ is an integer in the range $[0, n]$ and

$$n = \sum_{\ell=1}^{n} \ell a_\ell. \tag{32}$$

Clearly, the conjugacy class C_g depends only on $a_1, \ldots, a_n$. Let T be a Young tableau with n boxes and a_ℓ rows of length ℓ for $\ell = 1, \ldots, n$. We fill T with integers $1, 2, \ldots, n$ in the row major order starting from the leftmost box in the first (top) row. Now integers $1, 2, \ldots, n$ can be identified with boxes of T. We can assume without loss of generality that g cyclically shifts boxes in each row of T such that the ith box in a row j goes to the $(i+1)$th box in the same row and the last box in a row j goes to the first box in the same row.

Consider a permutation $h \in C_g$. Clearly, h has the same cyclic type as g although boxes spanning each cycle of h may be scattered over many rows of T. This determines a partition of T into a disjoint union of subsets

$$T = \bigcup_{\ell=1}^{n} \bigcup_{j=1}^{a_\ell} T_{\ell,j}$$

such that $|T_{\ell,j}| = \ell$ and the boxes in each subset $T_{\ell,j}$ span a length-ℓ cycle in h. Let $b_{\ell,j} \in T_{\ell,j}$ be a box contained in $T_{\ell,j}$ with the smallest integer label (recall that each box of T is labeled by an integer between 1 and n). Order the subsets $T_{\ell,j}$ with a fixed ℓ such that

$$b_{\ell,1} < b_{\ell,2} < \cdots < b_{\ell,a_\ell}.$$

Let $B_{\ell,j}$ be the first (leftmost) box of T in the jth length-ℓ row (rows are counted starting from the top). Let $f \in S_n$ be a permutation such that

$$f[g^p(B_{\ell,j})] = h^p(b_{\ell,j}) \quad \text{for all } p = 0, 1, \ldots, \ell - 1 \tag{33}$$

and all ℓ, j. Here $g^p \in S_n$ denotes the pth power of g. Since any box of T can be obtained from some box $B_{\ell,j}$ by applying powers of g, permutation f is uniquely determined by h. Also, it is clear that the map $h \to f$ is computable in time poly(n). Next we claim that $h = fgf^{-1}$. Since h cyclically shifts boxes in each subset $T_{\ell,j}$, it suffices to check that

$$h^p(b_{\ell,j}) = (fgf^{-1})^p(b_{\ell,j}) \tag{34}$$

for all ℓ, j, p. The above is equivalent to

$$h^p(b_{\ell,j}) = f[g^p f^{-1}(b_{\ell,j})] = f[g^p(B_{\ell,j})], \tag{35}$$

which follows from Eq. (33). A classical circuit of size poly(n) computing the map $h \to f$ can be converted to a polynomial-sized quantum circuit $\mathcal{Q}_1$ computing a map

$$\mathcal{Q}_1|h\rangle \otimes |0\rangle \otimes |0\rangle = |h\rangle \otimes |\text{garbage}(h)\rangle \otimes |f\,\rangle.$$

Here garbage(h) represents auxiliary bits used to make all gates reversible. Making a copy of the third register in the standard basis and applying the inverse of $\mathcal{Q}_1$ gives a polynomial-sized circuit $\mathcal{Q}$ computing a map

$$\mathcal{Q}|h\rangle \otimes |0\rangle = |h\rangle \otimes |f\,\rangle,$$

where $h = fgf^{-1}$. This is the circuit defined in Eq. (31). ■

We note that the problem of preparing conjugacy class states $|C_g\rangle$ for more general groups is considered in Ref. [38]. Our proof of Lemma 3 closely follows the strategy outlined in Appendix D of Ref. [38].

VI. CLASSICAL COMPLEXITY OF SAMPLING

In Sec. V we showed that a quantum computer can efficiently sample from probability distributions (28) and (29) associated with a given column or row of the character table of the symmetric group S_n. Are these sampling problems hard for classical computers?

A. Granularity and quantum advantage

Let us first consider a general setting in which a sampling problem is defined by a probability distribution P over some finite set Ω called the sample space and an error tolerance ϵ. We consider infinite families of problem instances labeled by an integer n called the problem size. It is required that each point $x \in \Omega$ can be specified by poly(n) bits. Likewise, the probability distribution P should admit a succinct classical description of size poly(n). We also assume that the error tolerance obeys $\epsilon \geq \text{poly}(1/n)$. A classical or quantum algorithm solves the sampling problem if it outputs a sample $x \in \Omega$ from some probability distribution P' that is ϵ close to P in the total variation distance, such that $\|P - P'\|_1 \equiv \sum_{x\in\Omega} |P(x) - P'(x)| \leq \epsilon$. Such an algorithm is efficient if its runtime is at most poly(n).

We are specifically interested in sampling problems that can be solved efficiently by a quantum computer. To provide evidence for classical hardness of such a sampling problem, one can follow the complexity-theoretic argument pioneered in Refs. [19,39]. Roughly speaking, this argument gives a reduction from *average-case strong simulation*—computing output probabilities of a quantum circuit in the average case over a suitable ensemble—to the sampling problem of interest. Here the reduction uses polynomial time classical resources as well as access to an NP oracle (i.e., it is a BPP^{NP} algorithm). If the average-case strong simulation problem is so hard that it cannot be solved using resources within the polynomial hierarchy then this implies that no efficient classical algorithm can solve the sampling problem of interest.

Here we describe a variant of this argument for sampling problems with a certain granularity structure, described below.

Definition 1 (Granularity). A probability distribution P over a finite set Ω is called granular if there exists a real number $\gamma \geq 1$ such that $P(x)$ is an integer multiple of $\gamma/|\Omega|$ for all points $x \in \Omega$. We refer to the number γ as the granularity of P.

For example, distribution (29) defined by columns of the character table has granularity $\gamma = 1$ since the sample space has size $|\Omega| = |S_n| = n!$ and characters of the symmetric group are integer valued. Distribution (28) defined by rows of the character table may or may not be granular depending on the choice of the permutation $g \in S_n$, as detailed below. If the input of a sampling problem is a granular distribution P, the granularity γ and the size of the sample space $|\Omega|$ have to be included in the classical description of P.

Lemma 4. Suppose that P_0 and P_1 are probability distributions over a finite set Ω such that P_0 is granular, P_1 can be sampled by a BPP algorithm and $\|P_0 - P_1\|_1 \leq \epsilon$ for a given ϵ that is at least inverse polynomial in n. Let $\gamma \geq 1$ be the granularity of P_0. There exists a BPP^{NP} algorithm that takes as input a point $x \in \Omega$ and outputs an integer $k(x)$ such that

$$P_0(x) = \frac{\gamma k(x)}{|\Omega|} \tag{36}$$

for at least a max $\{0, 1 - 20\epsilon\}$ fraction of points $x \in \Omega$.

Proof. Let $N = |\Omega|$. Define the sets

$$S = \left\{x \in \Omega\colon P_0(x) \leq \frac{1}{10N\epsilon}\right\}$$

and

$$T = \left\{x \in \Omega\colon |P_0(x) - P_1(x)| \leq \frac{1}{10N}\right\}.$$

Markov's inequality gives $|S| \geq (1 - 10\epsilon)N$ and $|T| \geq (1 - 10\epsilon)N$. By the union bound,

$$|S \cap T| \geq (1 - 20\epsilon)N. \tag{37}$$

The assumption that P_1 can be sampled by a BPP algorithm and Stockmeyer's approximate counting theorem [40] imply that there exists a BPP^{NP} algorithm that takes as

input a point $x \in \Omega$ and outputs a real number $P_2(x)$ such that

$$|P_2(x) - P_1(x)| \leq \epsilon P_1(x).$$

For any $x \in S \cap T$, one has $P_1(x) \leq P_0(x) + 1/(10N) \leq 1/(10N\epsilon) + 1/(10N) \leq 1/(5N\epsilon)$, which implies that $|P_2(x) - P_1(x)| \leq 1/(5N)$. Thus, for any $x \in S \cap T$, one has

$$\begin{aligned} |P_2(x) - P_0(x)| &\leq |P_2(x) - P_1(x)| + |P_0(x) - P_1(x)| \\ &\leq \frac{1}{5N} + \frac{1}{10N} \leq \frac{1}{3N}. \end{aligned} \tag{38}$$

By assumption, $P_0(x)$ is an integer multiple of γ/N for a given $\gamma \geq 1$. Let $P_3(x)$ be an integer multiple of γ/N closest to $P_2(x)$. Using Eq. (38), one gets $P_3(x) = P_0(x)$ whenever $x \in S \cap T$. The above shows that $P_3(x)$ can be computed by a $\mathrm{BPP}^{\mathrm{NP}}$ algorithm and Eq. (37) implies that $P_3(x) = P_0(x)$ for at least a $1 - 20\epsilon$ fraction of points $x \in \Omega$. Finally, set $k(x) = (N/\gamma)P_3(x)$. ■

A central challenge in quantum complexity theory is to prove the quantum advantage conjecture [19] in the following form.

Conjecture 1. Assume that the polynomial hierarchy does not collapse. Then there exists a sampling problem that can be solved efficiently on a quantum computer and that is provably hard for classical randomized algorithms.

Lemma 4 implies that the conjecture is true if there exists a sampling problem satisfying the following properties.

(1) *Granularity*: the sampling problem is based on a family of granular probability distributions P.
(2) *Quantum algorithm*: the sampling problem can be solved efficiently on a quantum computer.
(3) *Average-case hardness*: the problem of computing $P(x)$ exactly on a point $x \in \Omega$ picked uniformly at random with a sufficiently high success probability is not contained in the polynomial hierarchy. Here the success probability must be at least $1 - \mathrm{poly}(1/n)$ for problem instances of size n.

We see that, for granular distributions, the hardness of classical sampling follows from the average-case hardness property (3) and the assumption that the polynomial hierarchy does not collapse. This differs from the usual argument in that we do not need to establish the so-called anticoncentration property for the family of distributions P; moreover, the average-case hardness property (3) concerns *exact* computation of output probabilities rather than approximate computation.

Note that a dummy algorithm for computing $P(x)$ that outputs zero for all $x \in \Omega$ has success probability at least $1 - 1/\gamma$. Indeed, the dummy algorithm succeeds with probability $|S|/|\Omega|$, where $S = \{x \in \Omega : P(x) = 0\}$. Granularity gives $P(x) \geq \gamma/|\Omega|$ for $x \notin S$. Thus, $1 = \sum_{x\in\Omega} P(x) \geq \gamma(|\Omega| - |S|)/|\Omega|$, that is, $|S|/|\Omega| \geq 1 - 1/\gamma$, as claimed. This shows that the average-case hardness property (3) can only be achieved if the granularity γ is at most polynomial in n.

B. Row and column sampling

We have already seen that the column sampling problem obeys properties (1) and (2). Furthermore, computing the probability $P_\lambda(g) = |\chi_\lambda(g)|^2/n!$ is $\#P$-hard in the worst case [4]. Moreover, this problem is not contained in $\#P$ unless the polynomial hierarchy collapses to the second level [4]. However, it seems unlikely that average-case hardness property (3) holds for this problem. In particular, the following lemma gives a classical quasipolynomial time algorithm for computing $P_\lambda(g)$ that succeeds with high probability if $g \in S_n$ is a random permutation.

Lemma 5. There exists a classical algorithm that takes as input an irrep λ of the symmetric group S_n, a permutation $g \in S_n$, and outputs an integer $k(g)$ such that $k(g) = \chi_\lambda(g)$ for at least a $1 - 1/n^{100}$ fraction of permutations $g \in S_n$. The algorithm has runtime $n^{O(\log n)}$.

Proof. The proof combines the MPS algorithm of Sec. IV for computing characters of S_n with tail bounds for the number of cycles in a random permutation. We use the algorithm from Sec. IV with zero truncation error ($\epsilon = 0$). With input $\lambda \vdash n$ and $g \in S_n$, this algorithm computes an MPS $|\psi_g\rangle$ of bond dimension (17) that can be upped bounded as $D \leq (n+2)^{c(g)}$, where $c(g)$ is the number of cycles in g. Recall that computing this MPS and extracting the amplitude $\chi_\lambda(g)$ uses a runtime upper bounded as

$$O(n^2 D^3) = n^{O[c(g)]}.$$

This runtime is exponential in n in the worst case since the number of cycles $c(g)$ can be linear in n. However, we are only interested in the average case, that is, random permutations. Furthermore, random permutations are known to have only a few cycles with high probability. The following fact is a special case of Theorem 1.7 of Ref. [41].

Fact 1. Suppose that $g \in S_n$ is picked uniformly at random. Let $c(g)$ be the number of cycles in g. Then, for all $\ell \geq 1$, one has

$$\Pr[c(g) \geq 1 + \ell H_n] \leq 2e^{1-[\ell\log(\ell)-\ell+1]H_n}, \tag{39}$$

where $H_n = \sum_{j=1}^{n} 1/j = \log(n) + O(1)$ is the nth harmonic number and $e = 2.718\,28\ldots$.

Corollary 1. There exists a constant $B > 0$ such that the fraction of permutations $g \in S_n$ with $c(g) > B \log(n)$ cycles is at most $1/n^{100}$ for all large enough n.

If $c(g) \leq B \log n$, use the MPS algorithm to compute $\chi_\lambda(g)$, which takes time $n^{O(\log n)}$. Otherwise, if $c(g) > B \log n$, output some fixed integer (say zero). This gives the desired algorithm. ■

Next let us consider the row sampling problem. Lemma 5 suggests that classically hard instances of the problem can be constructed by choosing the input permutation $g \in S_n$ with a large number of cycles, for example, $c(g) \sim \sqrt{n}$. In this case best-known classical algorithms for computing the characters $\chi_\lambda(g)$ appear to require an exponential time. However, we need to check whether permutations g with a large number of cycles give rise to instances of the row sampling problem with a granular distribution. This is addressed in the following lemma.

Lemma 6 (Granularity for row sampling). There exist a universal constant $A > 0$ such that the probability distribution $P_g(\lambda)$ over partitions $\lambda \vdash n$ from Eq. (28) is granular whenever $g \in S_n$ is a permutation with at most $A\sqrt{n}/\log n$ cycles.

Proof. Let Ω_n be the set of partitions $\lambda \vdash n$ (our sample space). We can write

$$P_g(\lambda) = \gamma \frac{|\chi_\lambda(g)|^2}{|\Omega_n|}, \qquad \gamma = \frac{|\Omega_n|}{|E_g|}, \tag{40}$$

where E_g is the centralizer of g. Since $\chi_\lambda(g)$ is integer for all $\lambda \in \Omega_n$, distribution $P_g(\lambda)$ is granular whenever $\gamma \geq 1$.

Suppose that g consists of a_ℓ cycles of length ℓ. Then $|E_g|$ is given by Eq. (21). Also, it is known [42] that $|\Omega_n| \geq e^{A\sqrt{n}}$ for some universal constant $A > 0$ (one can choose $A = \pi\sqrt{2/3} - \epsilon$ for any constant $\epsilon > 0$) and all sufficiently large n. Thus,

$$\gamma > \frac{e^{A\sqrt{n}}}{\prod_{\ell=1}^n \ell^{a_\ell} a_\ell!}. \tag{41}$$

Since $a_\ell! \leq (a_\ell)^{a_\ell}$, one gets

$$\gamma \geq \exp\left[A\sqrt{n} - \sum_{\ell=1}^n a_\ell \log(\ell a_\ell)\right]. \tag{42}$$

From Eq. (32) one gets $\ell a_\ell \leq n$. Using this along with $c(g) = \sum_{\ell=1}^n a_\ell$ gives

$$\gamma \geq \exp\left[A\sqrt{n} - c(g)\log n\right]. \tag{43}$$

Thus, $\gamma \geq 1$ whenever $A\sqrt{n} - c(g)\log n \geq 0$. ■

We conclude that the row sampling problem for permutations g with $A\sqrt{n}/\log n$ or less cycles obeys properties (1) and (2). Proving or disproving property (3)—the average-case hardness of computing $P_g(\lambda)$ for random partitions $\lambda \vdash n$—is left as a challenge for future work.

The proposed reduction from strong-to-weak simulation may apply to sampling tasks unrelated to the symmetric group. As a concrete example, consider a sampling version of the commuting local Hamiltonian problem [43]. Suppose that $m \geq n$ are integers and that $H_1, \ldots, H_m$ are pairwise commuting n-qubit projectors such that each projector acts nontrivially only on a few qubits. Hamiltonians composed of few-qubit commuting projectors are often used as toy models of the topological quantum order [44]. Given a bit string $x \in \{0,1\}^m$, let $\mathcal{L}_x$ be the subspace spanned by n-qubit states ψ such that $H_a|\psi\rangle = x_a|\psi\rangle$ for all $a = 1, \ldots, m$. Let d_x be the dimension of $\mathcal{L}_x$. Consider a normalized probability distribution $P(x) = d_x/2^n$ describing the joint eigenvalue measurement for the projectors $H_1, \ldots, H_m$ on the maximally mixed state of n qubits. Since the sample space has size 2^m, distribution $P(x)$ has granularity $\gamma = 2^{m-n} \geq 1$. A quantum computer can efficiently sample $P(x)$ by measuring the projectors one by one in some fixed order. Assuming that the task of computing dimension d_x for a random uniform $x \in \{0,1\}^m$ is not contained in the class $\mathrm{BPP}^{\mathrm{NP}}$, our strong-to-weak reduction implies that no BPP algorithm can sample distribution $P(x)$ with a small error. We note that a closely related task of deciding whether $d_x > 0$ for a given bit string x is a variant of the commuting local Hamiltonian problem that has been studied by many authors [43,45–50].

VII. CLASSICAL MPS ALGORITHM FOR KOSTKA NUMBERS

In Sec. IV we showed that Crichigno and Prakash's connection between quantum spin chains and characters of the symmetric group can be leveraged in an MPS algorithm for characters. In Ref. [13] they also proved mappings between spin chains and Kostka numbers and Littlewood-Richardson coefficients. Using similar ideas to characters, we show that this correspondence can be used to give an MPS algorithm for Kostka numbers.

Kostka numbers appear frequently in combinatorics and representation theory. Let $\lambda \vdash n$. A semistandard Young tableau of shape λ is a filling of the boxes in the Young diagram corresponding to λ with positive integers between 1 and n such that the labels are nondecreasing across rows and strictly increasing down columns. The weight of such a tableau is an integer-valued vector of length n such that the ith entry is the number of occurrences of i. The Kostka number $K_{\lambda,\mu}$ is the number of semistandard Young tableaus of shape λ and weight μ. Note that $K_{\lambda,\mu}$ is independent of the order of the entries in μ and thus we assume that μ is a partition of n. A notable special case is $K_{\lambda,(1^n)}$, which

is the dimension of the irrep labeled by λ. More generally, $K_{\lambda,\mu}$ appears in the representation theory of S_n as the multiplicity of the irrep λ in the permutation module indexed by μ. In the language of symmetric polynomials, the Kostka numbers encode the change of basis from Schur polynomials to complete symmetric polynomials.

Crichigno and Prakash used this latter characterization to demonstrate that quantum spin chains encode Kostka numbers as well [13]. We briefly review their construction here. Again, let the initial state $|x_\tau\rangle$ correspond to the integer sequences $(n-1, n-2, \ldots, 0)$ in the second quantization picture. For commuting variables, the complete symmetric polynomial h_k is given by

$$h_k = \sum_{1 \le i_1 \le \cdots \le i_k} x_{i_1} x_{i_2} \cdots x_{i_k}. \tag{44}$$

For example, in three variables $h_2 = x_1^2 + x_2^2 + x_3^2 + x_1x_2 + x_1x_3 + x_2x_3$. In Crichigno and Prakaksh's construction the commutative variables are replaced with operators $\hat{x}_i := a_{i+1}^\dagger a_i$, where a ($a^\dagger$) is again an annihilation (creation) operator. Then, the noncommutative complete symmetric polynomials take the form

$$\hat{h}_k = \sum_{i_1 > \cdots > i_k} \hat{x}_{i_1} \hat{x}_{i_2} \cdots \hat{x}_k. \tag{45}$$

Note that the indices are required to be strictly decreasing, which follows from $\hat{x}_i^2 = 0$ for any index i. Say that $\mu \vdash n$ is a partition of length m. We define $\hat{h}_\mu = \prod_{j=1}^m \hat{h}_{\mu_j}$ (the operators commute and thus the order of the product does not matter [13]). The authors showed that

$$K_{\lambda,\mu} = \langle x_{\lambda+\tau} | \hat{h}_\mu | x_\tau \rangle. \tag{46}$$

Note that $\hat{h}_\mu$ moves fermions at most n sites to the right and we can again truncate the chain to $2n$ sites with the initial state $|x_\tau\rangle = |1^n 0^n\rangle$.

Similarly to the MPS algorithm for characters, we represent the $2n$-qubit state

$$|\psi_\mu\rangle = \prod_{i=1}^m \hat{h}_{\mu_i} |1^n 0^n\rangle \tag{47}$$

by an MPS. We do so via the following lemma.

Lemma 7. The noncommuting complete symmetric polynomial $\hat{h}_k$ is an MPO with bond dimension $2k+1$.

Proof. Figure 4 defines a tensor $T \in \mathbb{C}^2 \otimes \mathbb{C}^{2k+1} \otimes \mathbb{C}^{2k+1} \otimes \mathbb{C}^2$, where $\mathbb{C}^2$ represents physical indices and $\mathbb{C}^{2k+1}$ represents virtual indices of the MPO. Taking $2n$ copies of T and contracting the virtual indices gives an operator

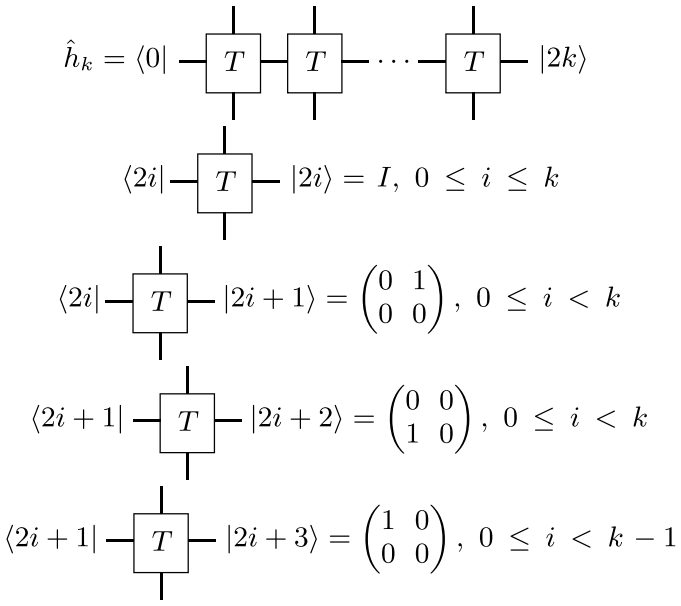


FIG. 4. MPO representation of the noncommuting complete symmetric polynomial $\hat{h}_k$. Horizontal lines represent virtual indices of dimension $2k+1$. We label basis vectors of $\mathbb{C}^{2k+1}$ by integers $\{0, 1, \ldots, 2k\}$. The MPO consists of $2n$ copies of T. We list all nonzero components of tensor T.

$$\sum_{i_1 > i_2 > \cdots > i_k} \hat{x}_{i_1} \hat{x}_{i_2} \cdots \hat{x}_{i_k}. \tag{48}$$

This is exactly $\hat{h}_k$ as defined in Eq. (45). ■

From the bond dimension of this MPO, we can upper bound the maximum bond dimension of all states during compression as

$$D \le \prod_{j=1}^m (2\mu_j + 1). \tag{49}$$

In running numerics, however, bond dimensions appeared to be significantly lower.

Denote the compressed MPS state as $|\mathsf{mps}_\mu\rangle$. Similarly to characters, Kostka numbers take integer values and thus the correct value is recovered as long as

$$|K_{\lambda,\mu} - \langle \lambda | \mathsf{mps}_\mu \rangle| < \tfrac{1}{2}. \tag{50}$$

We again used the mpnum software for numerical simulations with an error rate $\varepsilon = 10^{-11}$. We compared the MPS algorithm to the symmetrica library of SAGE; see Fig. 5.

For numerical experiments, we selected random partitions μ with a smaller number of parts, fewer than

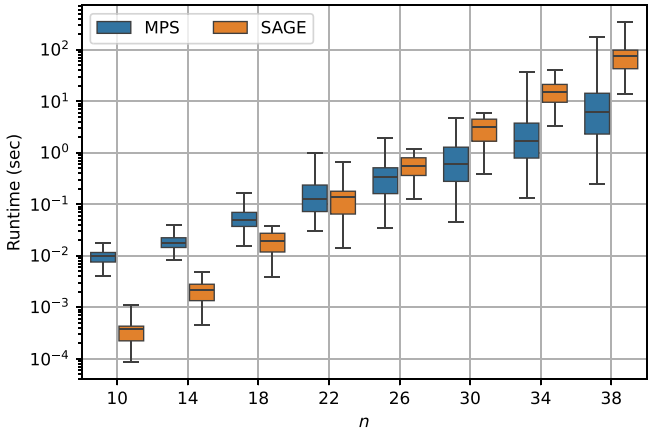


FIG. 5. Runtime for computing Kostka numbers $K_{\lambda,\mu}$. Each runtime is, fixing a random μ, for computing the set $\{K_{\lambda,\mu}\}_\lambda$. One hundred random μ's of length less than $n/3$ were sampled for each n. The MPS compression error is set to $\varepsilon = 10^{-12}$ for all the data. Boxes indicate the first and third quartiles and the whiskers denote 1.5 times the interquartile range.

$n/3$. On longer partitions the MPS algorithm appears to require smaller ε and longer run times. For example, $\mu = (1, \ldots, 1)$ proved difficult to compute via the MPS algorithm yet can be quickly computed via the hook length formula. However, in the regime of shorter partitions, our numerics offer evidence that the MPS is competitive and often faster than Sage for large n. It is further worth noting that most partitions are relatively short, the average being $\Theta[\sqrt{n}\log(n)]$ [51]. Our implementation, as well as that for skew Kostka numbers (a generalization), is available from GitHub [33].

An interesting question, which we leave for future work, is if Littlewood-Richardson coefficients can be recovered by a nontrivial MPS algorithm. Doing so via a mapping to spin chains would require MPO constructions for noncommutative Schur polynomials [13]. Schur polynomials do not admit an obvious modular construction unlike $\prod_{j=1}^{c(g)} J_{\eta_j}$ and $\prod_{j=1}^{m} \hat{h}_{\mu_j}$ (from the commuting products). This would seem to complicate MPO constructions.

We also leave open the question of sampling from distributions with probabilities defined by normalized Kostka numbers, in analogy to the results in Sec. V. A related task was studied in Refs. [27,37], but there the distributions contained just a single probability that was a normalized Kostka number. It would be interesting to find quantum circuits that prepare a quantum state that corresponds to the MPS we derive here.

ACKNOWLEDGMENTS

This research was supported in part by the Perimeter Institute for Theoretical Physics. Research at the Perimeter Institute is supported by the Government of Canada through the Department of Innovation, Science, and Economic Development, and by the Province of Ontario through the Ministry of Colleges and Universities. D.G. acknowledges support as a Canadian Institute for Advanced Research fellow, in the quantum information science program. L.S. was supported by IBM through the Illinois-IBM Discovery Accelerator Institute.

DATA AVAILABILITY

The data supporting the findings of this article and a Python package to implement the MPS algorithms for characters and Kostka numbers are openly available [33].

[1] Daniel Grier and Luke Schaeffer, in *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing* (Association for Computing Machinery, Chicago, USA, 2020), p. 875.
[2] David Lawrence Johnson, Minimal permutation representations of finite groups, Am. J. Math. **93**, 857 (1971).
[3] Charles Thomas Hepler, On the complexity of computing characters of finite groups. Master's thesis. https://prism.ucalgary.ca/items/0d83d63a-ec3b-47bd-b5ca-18bf532eb49a, (1996).
[4] Christian Ikenmeyer, Igor Pak, and Greta Panova, Positivity of the symmetric group characters is as hard as the polynomial time hierarchy, Int. Math. Res. Not. **2024**, 8442 (2023).
[5] Gerald V. Dunne, Slater decomposition of Laughlin states, Int. J. Mod. Phys. B **7**, 4783 (1993).
[6] Andrea Di Gioacchino, Luca Guido Molinari, Vittorio Erba, and Pietro Rotondo, Unified Fock space representation of fractional quantum Hall states, Phys. Rev. B **95**, 245123 (2017).
[7] Di Gioacchino, Fock-space operator construction of Laughlin fractional quantum Hall effect states: http://pcteserver.mi.infn.it/caraccio/Lauree/DiGioacchino_m.pdf, (2015).
[8] P. Di Francesco, M. Gaudin, C. Itzykson, and F. Lesage, Laughlin's wave functions, Coulomb gases and expansions of the discriminant, Int. J. Mod. Phys. A **9**, 4257 (1994).
[9] Benoît Collins and Piotr Śniady, Integration with respect to the Haar measure on unitary, orthogonal and symplectic group, Commun. Math. Phys. **264**, 773 (2006).
[10] Imre Risi Kondor, *Group Theoretical Methods in Machine Learning* (Columbia University, New York City, USA, 2008).
[11] Risi Kondor and Marconi S. Barbosa, in *COLT* (Association for Computing Machinery, Haifa, Israel, 2010), p. 451.
[12] Guifré Vidal, Efficient classical simulation of slightly entangled quantum computations, Phys. Rev. Lett. **91**, 147902 (2003).
[13] Marcos Crichigno and Anupam Prakash, Quantum spin chains and symmetric functions, ArXiv:2404.04322.
[14] Sergey Fomin and Curtis Greene, Noncommutative Schur functions and their applications, Discrete Math. **193**, 179 (1998).
[15] F. G. Frobenius, in *Sitz. Ber. Preuss. Akad.* (Prussian Academy of Sciences, Berlin, Germany, 1900), p. 516.

[16] Johnnie Gray, quimb: A python package for quantum information and many-body calculations, J. Open Source Softw. **3**, 819 (2018).

[17] Daniel Suess and Milan Holzäpfel, mpnum: A matrix product representation library for python, J. Open Source Softw. **2**, 465 (2017).

[18] Robert Beals, in *Proceedings of the twenty-ninth annual ACM symposium on Theory of computing* (Association for Computing Machinery, El Paso, USA, 1997), p. 48.

[19] Scott Aaronson and Alex Arkhipov, in *Proceedings of the forty-third annual ACM symposium on Theory of computing*, (Association for Computing Machinery, San Jose, USA, 2011), p. 333.

[20] Michael J. Bremner, Ashley Montanaro, and Dan J. Shepherd, Average-case complexity versus approximate simulation of commuting quantum computations, Phys. Rev. Lett. **117**, 080501 (2016).

[21] Dominik Hangleiter and Jens Eisert, Computational advantage of quantum random sampling, Rev. Mod. Phys. **95**, 035001 (2023).

[22] Francis D. Murnaghan, The characters of the symmetric group, Am. J. Math. **59**, 739 (1937).

[23] Tadasi Nakayama, in *Japanese journal of mathematics: transactions and abstracts* (The Mathematical Society of Japan, Tokyo, 1940), Vol. 17, p. 411.

[24] Ömer Eğecioğlu, in *European Conference on Computer Algebra*, (Springer, Linz, Austria, 1985), p. 206.

[25] Stephen P. Jordan, Fast quantum algorithms for approximating some irreducible representations of groups, (2009).

[26] Sergey Bravyi, Anirban Chowdhury, David Gosset, Vojtěch Havlíček, and Guanyu Zhu, Quantum complexity of the Kronecker coefficients, PRX Quantum **5**, 010329 (2024).

[27] Martin Larocca and Vojtech Havlicek, Quantum algorithms for representation-theoretic multiplicities, Phys. Rev. Lett. **135**, 010602 (2025).

[28] Pavel I. Etingof, Oleg Golberg, Sebastian Hensel, Tiankai Liu, Alex Schwendner, Dmitry Vaintrob, and Elena Yudovina, *Introduction to Representation Theory* (American Mathematical Soc., Providence, USA, 2011), Vol. 59.

[29] It may be helpful to understand this binary encoding of a partition as follows. If we start with the bit string $x_{\tau+\lambda}$ and then interpret each 1 as "up by one" and each 0 as "to the right by one" and then trace out a picture according to these instructions, we get a picture of the partition λ within a bounding box of size $n \times n$.

[30] Ulrich Schollwöck, The density-matrix renormalization group in the age of matrix product states, Ann. Phys. **326**, 96 (2011).

[31] The GAP Group, GAP – Groups, Algorithms, and Programming, version 4.12.2, 2022.

[32] W. A. Stein *et al.*, Sage Mathematics Software, version 9.8, The Sage Development Team, 2023, http://www.sagemath.org

[33] Sergey Bravyi, David Gosset, Vojtech Havlicek, and Louis Schatzki, https://github.com/sbravyi/symmetric_group_characters.

[34] Alok Shukla and Prakash Vedula, An efficient quantum algorithm for preparation of uniform quantum superposition states, Quantum Inf. Process. **23**, 38 (2024).

[35] Richard P. Stanley, *Enumerative Combinatorics: Volume 2* (Cambridge University Press, 1999), Vol. 62.

[36] A Vershik and S Kerov, Asymptotics of the maximal and typical dimension of irreducible representations of symmetric group, Func. Anal. Appl. **19**, 25 (1985).

[37] Greta Panova, Polynomial time classical versus quantum algorithms for representation theoretic multiplicities, ArXiv:2502.20253.

[38] Juan Bermejo-Vega and Kevin C. Zatloukal, Abelian hypergroups and quantum computation, ArXiv:1509.05806.

[39] Michael J. Bremner, Richard Jozsa, and Dan J. Shepherd, Classical simulation of commuting quantum computations implies collapse of the polynomial hierarchy, Proc. R. Soc. A: Math. Phys. Eng. Sci. **467**, 459 (2011).

[40] Larry Stockmeyer, in *Proceedings of the fifteenth annual ACM symposium on Theory of computing* (Association for Computing Machinery, New York City, USA, 1983), p. 118.

[41] Kevin Ford, Cycle type of random permutations: A toolkit, Discrete Analysis (2022).

[42] G. H. Hardy and S. Ramanujan, Asymptotic formulaæ in combinatory analysis, Proc. London Math. Soc. **s2-17**, 75 (1918).

[43] Sergey Bravyi and Mikhail Vyalyi, Commutative version of the k-local Hamiltonian problem and common eigenspace problem, Quantum Inf. Comput. **5**, 187 (2005).

[44] A. Yu Kitaev, Fault-tolerant quantum computation by anyons, Ann. Phys. **303**, 2 (2003).

[45] Norbert Schuch, Complexity of commuting Hamiltonians on a square lattice of qubits, Quantum Inf. Comput. **11**, 901 (2011).

[46] Dorit Aharonov and Lior Eldar, in *2011 IEEE 52nd Annual Symposium on Foundations of Computer Science* (IEEE, Palm Springs, USA, 2011), p. 334.

[47] Matthew B. Hastings, Matrix product operators and central elements: Classical description of a quantum state, Geom. Topol. Monogr. **18**, 276 (2012).

[48] Dorit Aharonov, Oded Kenneth, and Itamar Vigdorovich, in *Proceedings of the thirteenth conference on the Theory of Quantum Computation, Communication and Cryptography* (Schloss Dagstuhl – Leibniz-Zentrum fur Informatik, Sydney, Australia, 2018), p. 2.

[49] Dorit Aharonov and Lior Eldar, The commuting local Hamiltonian problem on locally expanding graphs is approximable in NP, Quantum Inf. Process. **14**, 1 (2015).

[50] Sandy Irani and Jiaqing Jiang, Commuting local Hamiltonian problem on 2D beyond qubits, ArXiv:2309.04910.

[51] Paul Erdös and Joseph Lehner, The distribution of the number of summands in the partitions of a positive integer, Duke Math. J. **8**, 335 (1941).