



Simulating lossy Gaussian boson sampling with matrix-product operatorsMinzhao Liu ^{1,2} Changhun Oh,³ Junyu Liu,^{3,4,5,6,7,8} Liang Jiang ^{3,5} and Yuri Alexeev^{2,4,5}¹*Department of Physics, The University of Chicago, Chicago, Illinois 60637, USA*²*Computational Science Division, Argonne National Laboratory, Lemont, Illinois 60439, USA*³*Pritzker School of Molecular Engineering, The University of Chicago, Chicago, Illinois 60637, USA*⁴*Department of Computer Science, The University of Chicago, Chicago, Illinois 60637, USA*⁵*Chicago Quantum Exchange, Chicago, Illinois 60637, USA*⁶*Kadanoff Center for Theoretical Physics, The University of Chicago, Chicago, Illinois 60637, USA*⁷*qBraid Co., Chicago, Illinois 60615, USA*⁸*SeQure, Chicago, Illinois 60615, USA*

(Received 22 May 2023; accepted 18 October 2023; published 13 November 2023)

Gaussian boson sampling, a computational model that is widely believed to admit quantum supremacy, has already been experimentally demonstrated and is claimed to surpass the classical simulation capabilities of even the most powerful supercomputers today. However, whether the current approach limited by photon loss and noise in such experiments prescribes a scalable path to quantum advantage is an open question. To understand the effect of photon loss on the scalability of Gaussian boson sampling, we analytically derive the asymptotic operator entanglement entropy scaling, which relates to the simulation complexity. As a result, we observe that efficient tensor network simulations are likely possible under the $N_{\text{out}} \propto \sqrt{N}$ scaling of the number of surviving photons N_{out} in the number of input photons N . We numerically verify this result using a tensor network algorithm with U(1) symmetry, and we overcome previous challenges due to the large local Hilbert-space dimensions in Gaussian boson sampling with hardware acceleration. Additionally, we observe that increasing the photon number through larger squeezing does not increase the entanglement entropy significantly. Finally, we numerically find the bond dimension necessary for fixed accuracy simulations, providing more direct evidence for the complexity of tensor networks.

DOI: [10.1103/PhysRevA.108.052604](https://doi.org/10.1103/PhysRevA.108.052604)**I. INTRODUCTION**

Exact classical simulations of quantum systems are intractable due to the exponential size of the Hilbert space. As a result, computations using quantum systems have been proposed to achieve improvement in algorithmic complexities in tasks such as integer factoring [1], unstructured search [2], linear algebra [3], Hamiltonian simulations [4–7], and more [8]. Present-day quantum computational devices, however, are susceptible to noise and cannot be perfectly controlled. As a result, numerous approaches based on sampling outputs of randomly configured devices have been proposed to demonstrate quantum supremacy, which are especially appealing considering near-term constraints. For example, boson sampling [9], a process of sampling the photon output patterns from interferometers, has resulted in numerous experimental demonstrations of quantum supremacy [10–28].

However, experimental imperfections such as photon loss can have implications on the computational complexity. The effects of noise are already examined in various contexts of quantum computing experiments. For qubits cases, it has long been known that without error correction, a quantum state after a large depth with a constant level of depolarizing noise becomes very close to the maximally mixed state [29], which enables an efficient approximate simulation. One proxy of classical simulation complexity is the entanglement entropy (EE). For pure state simulations, the computational cost using

tensor networks is exponential in the EE of the quantum system, which implies that systems with logarithmic growth in the EE can be efficiently simulated. Similarly, it has been argued that the density operator EE of mixed states implies a similar computational cost, albeit with some nuances [30–32]. In the context of noisy random circuit sampling (RCS) [33], it was numerically shown that the density operator EE decreases if the circuit depth is too high for one-dimensional (1D) [30] and two-dimensional [31] systems. Further, the maximum achievable EE follows area law scaling, suggesting the possibility of efficient tensor network simulation. More recently, polynomial time simulation of RCS with constant depolarizing noise per gate is proven to be possible in an asymptotic regime for larger than logarithmic depths, denying the scalability of RCS [34].

Meanwhile, in the context of boson sampling, a similar study has very recently shown that for a particular noise, which may not be experimentally relevant, there is an efficient classical algorithm for noisy boson sampling in an asymptotic regime [35]. Thus, the experimental noise might prohibit scalable quantum advantage in boson sampling, much like in RCS. However, it still remains possible for noisy boson sampling to be scalable under realistic noises, such as photon loss [36–39] and partial distinguishability [38,40–43]. Notably, the effects of photon loss are investigated in several studies. For single-photon boson sampling (SPBS) [36,37] and Gaussian boson sampling (GBS) [39,44], when the number

of photons N_{out} surviving before measurement scales as the square root of the number of input photons N ($N_{\text{out}} \propto \sqrt{N}$), classical state approximation of the output state provides an efficient method of simulation. However, the approximation error of these methods is fixed for given parameters and cannot be controlled with more resources. As a result, quantum supremacy intermediate-size experiments, where transmission is not as low as these approximate algorithms require, elude these methods.

Tensor network methods, on the other hand, allow us to control the simulation error by tuning time and memory resources, and have been used to numerically show the logarithmic scaling of the operator EE when $N_{\text{out}} \propto \sqrt{N}$ in SPBS [32]. However, the probabilistic nature of single-photon generation renders SPBS unscalable, and the community has long moved onto other photon sources [45–47]. The most promising approach is GBS, where no postselection is necessary and classical simulation is hard unless some plausible complexity-theoretic conjectures are false [48,49]. This allowed recent experimental demonstrations of GBS to claim quantum supremacy [27,28,50].

In this paper, we investigate the operator EE scaling of GBS, which is more experimentally relevant. We show analytically that in the asymptotic limit of large N , the logarithmic operator EE scaling holds for $N_{\text{out}} \propto \sqrt{N}$. For numerical verification, simulation of GBS is especially difficult with tensor networks due to the infinite-dimensional local Hilbert space for each squeezed mode, which remains high even under suitable truncation and leads to dramatically increased computational cost. As a result, we develop a hardware-accelerated, supercomputing tensor network algorithm that exploits U(1) symmetry, allowing us to simulate previously intractable systems such as GBS [51]. We numerically verify the operator EE scaling of GBS under various loss conditions against the asymptotic estimates, and further observe that increasing the photon number through higher squeezing has little impact on EE. Finally, we explicitly calculate the bond dimension and the computational cost as the most direct evidence on the complexity. Overall, our paper suggests that boson sampling with loss higher than the aforementioned scaling may be efficiently simulated with tensor networks as the system size grows.

Related work

The first class of methods for simulating Gaussian boson sampling is the class of exact methods, which directly computes the loop Hafnians to determine the probability amplitudes of detection events [52–54]. These methods have exponential time or space complexity, and do not deal with lossy states.

Besides exact simulation algorithms, approximate methods such as tensor network methods have been developed to reduce the simulation costs. Another approximate algorithm uses polynomial approximation of the marginals of the outputs, and the complexity is exponential in the order of approximation k [55]. There is no explicit use of the lossy nature of GBS. The relation between the required order k and photon loss is not well understood, and therefore the time complexity scaling with loss is similarly unknown.

The most relevant class of approximate algorithms that explicitly exploits photon loss is the aforementioned classical state approximation approach [39,44]. These methods find a classical state that approximates the squeeze states as closely as possible, and the interference outcomes of these classical states are efficient to simulate. However, this means that once the classical description is fixed, there is no control to further decrease the error by any means. It is shown that these methods can give asymptotically small error when $N_{\text{out}} \propto \sqrt{N}$ or less. The complexity is always polynomial regardless of the loss scaling, but the error becomes unacceptable for higher loss. Although an earlier experiment [26] has been found to be potentially well described by the classical states, more recent experiments cannot be simulated with classical states. The fact that finite-size quantum supremacy experiments have low loss makes this approach unsuitable.

Overall, algorithms applicable to finite-size system simulations do not directly use the lossy nature of GBS, and cannot control the simulation error with full freedom. Tensor network methods give a direct measure of the simulation complexity in terms of the bond dimension. This is the precise reason why we vary experimental parameters such as loss, squeezing, and system sizes while keeping the approximation error $1 - \text{Tr}(\rho)$ fixed. This unique ability of fixing the error by varying the bond dimension allows us to measure the simulation cost as a function of the experimental parameters. Further, while other sampling methods only seek to spoof the sampling benchmarks such as the cross entropy benchmark and low-order marginals, tensor networks directly approximate the quantum state, and offer a much richer set of information that can be potentially investigated for theoretical interest [56].

II. METHOD

Lossless and lossy quantum states in boson sampling can be represented by matrix-product states (MPSs) and matrix-product operators (MPOs) [32,57]. More explicitly, a general M -body pure state with local Hilbert-space dimension d can be written as

$$|\Psi\rangle = \sum_{i_1, \dots, i_M=0}^{d-1} c_{i_1, \dots, i_M} |i_1, \dots, i_M\rangle, \quad (1)$$

where the tensor c_{i_1, \dots, i_M} fully characterizes the state $|\Psi\rangle$. However, tensor c is M dimensional, leading to d^M entries in storage. To reduce the storage cost, the standard MPS ansatz represents the state in a compressed manner as

$$c_{i_1, \dots, i_M} = \sum_{\alpha_0, \dots, \alpha_M=0}^{\chi-1} \Gamma_{\alpha_0 \alpha_1}^{[1]i_1} \lambda_{\alpha_1}^{[1]} \Gamma_{\alpha_1 \alpha_2}^{[2]i_2} \lambda_{\alpha_2}^{[2]} \dots \lambda_{\alpha_{M-1}}^{[M-1]} \Gamma_{\alpha_{M-1} \alpha_M}^{[M]i_M}, \quad (2)$$

where χ is the so-called bond dimension, and larger χ corresponds to a lower approximation error. In the MPS representation, each $\Gamma^{[j]}$ tensor contains information about the j th body, and $\Gamma_{\alpha_{j-1} \alpha_j}^{[j]i_j}$ captures its amplitudes in state i_j , conditioned on the states of the left and right neighbors specified by the α_{j-1} and α_j indices. The λ tensors can be understood as singular values of Schmidt decomposition, which we show in more detail in Eq. (6). The MPS represents this large

tensor as a contraction (sum over the dummy or virtual α indices which capture correlations between particles) of a chain of tensors. One can observe that the i indices representing the physical degrees of freedom remain open (unsummed). The memory complexity of the MPS is $O(\chi^2 dM)$, and χ can be adjusted to represent c with the desired accuracy. Further, one can efficiently perform local unitary operations on the MPS and calculate expectation values of local observables with complexity $O(d^4 \chi^3)$.

Lossy boson sampling can be simulated using an MPO, which is essentially an MPS with additional dual indices. Time evolving the MPO can be accomplished by contracting it with Kraus operators. Further, in the case where loss is uniform (equal photon loss at every beam splitter), all losses can be moved to the initial state since loss commutes with linear optical transformations. As a result, full Kraus operator-based simulation of noisy channels is not necessary. We can represent the initial lossy state using an MPO, vectorize the MPO, build a tensor network analogous to Eq. (2), and update the MPO by contracting it with unitaries [57–59]. Specifically, vectorization of a density operator is defined as

$$\begin{aligned} \hat{\rho} &= \sum_{k,k'} |k\rangle \rho_{k,k'} \langle k'| \\ \rightarrow |\hat{\rho}\rangle &= \sum_{k,k'} \rho_{k,k'} |k, k'\rangle = \sum_K \rho_K |K\rangle, \end{aligned} \quad (3)$$

where K is the combined index for k, k' . Application of the unitary is also slightly modified since vectorization of $U \hat{\rho} U^\dagger$ becomes

$$\sum_{k,k'} U_{j,k} \rho_{k,k'} U_{k',j'}^\dagger \rightarrow \sum_K \mathcal{U}_{j,K} \rho_K, \quad (4)$$

where $\mathcal{U}_{j,K} = U_{j,k} U_{k',j'}^*$.

For lossless, pure state systems with large local dimensions d , the $O(d^4 \chi^3)$ computational complexity becomes significant. For lossy, mixed state simulations with MPOs, the complexity becomes $O(d^8 \chi^3)$, which is especially problematic for systems with large d such as GBS. Fortunately, for systems with global symmetry, such as particle number or total spin conservation, another level of reduction is possible. Symmetry-preserving operators can be expressed as a direct sum $\hat{T} = \bigoplus_n \hat{T}_n$ where \hat{T}_n preserves the subspace \mathbb{V}_n corresponding to some conserved charge n , and tensors can be therefore written in a block-diagonal form and stored efficiently [58]. Computations are performed on different blocks independently, and the time cost also reduces due to the non-linear polynomial complexity.

An important class of quantum systems has global U(1) symmetry, which arises when the system has some kind of conserved charge [58]. Examples of such systems include the hardcore Bose-Hubbard model [60], the spin-1/2 XXZ quantum spin chain [61], boson sampling [9], quantum walk [6,62–65], and monitored quantum circuits [66]. A model is said to be U(1) symmetric if the Hamiltonian commutes with the total charge operator [58]:

$$[\hat{H}, \hat{N}] = 0. \quad (5)$$

As a result, evolution under such Hamiltonians must preserve the charge number operator. More generally, systems can

preserve a global U(1) symmetry if the applied unitaries preserve the global charge.

In the case of boson sampling, the unitaries preserve the global photon number. As we discussed earlier, loss can be commuted to initialization, and time evolution is fully U(1) symmetric. To exploit the U(1) symmetry, one can define a so-called charge which is the number of photons to the left of the bipartition. The overall effect is that the Γ tensors lose their i indices corresponding to the physical degree of freedom (local photon number), reducing the memory complexity by a factor of d (see Appendix D for more details). This is instead captured by the size χ 1D charge tensors c . Second, the size of the matrices that we decompose with singular value decomposition (SVD) is also reduced to at most $\chi \times \chi$ instead of $\chi d \times \chi d$. Therefore, the use of U(1) symmetry significantly reduces the memory and time complexity of the algorithm.

The MPS formulation is especially convenient for quantifying entanglement. If we perform the Schmidt decomposition on the quantum state, which is to express the wave function as the sum of tensor products of states of two subsystems A and B ,

$$|\Psi\rangle = \sum_{\alpha} \lambda_{\alpha} |\alpha_A\rangle |\alpha_B\rangle, \quad (6)$$

where $\{|\alpha\rangle\}$ forms a basis set for each subsystem, we reveal the entanglement between the two subsystems, and the EE given by

$$- \sum_{\alpha} \lambda_{\alpha}^2 \log_2 \lambda_{\alpha}^2 \quad (7)$$

quantifies how much entanglement there is. Conveniently, if the subsystems are bipartitions of the MPS at site ℓ , the Schmidt decomposition singular values λ_{α} would be the MPS singular values $\lambda_{\alpha}^{[\ell]}$, allowing us to compute the MPS EE. For a mixed state represented by a vectorized MPO, we can formally perform Schmidt decomposition, identify the singular values λ_{α} with $\lambda_{\alpha}^{[\ell]}$ of the MPO, and similarly compute the MPO EE. In both cases, higher EE means more uniformly distributed singular values, and truncation leads to a higher approximation error. Larger bond dimensions are necessary to simulate systems with larger EE to fixed accuracy.

III. RESULTS

A. Supercomputing U(1)-symmetric tensor network algorithm

Details of the numerical protocols of time evolving the U(1)-symmetric tensor network are available in Appendix D. CPU based implementations have already been adopted in single-photon boson sampling simulations, but the computational cost for Gaussian boson sampling is still too high. Increasing the parallelism through the use of GPU and multi-node parallel computing is necessary. A naive implementation where each GPU thread computes one tensor entry performs worse than the CPU implementation due to technical reasons discussed in Appendix D. However, with the implementation in this paper, we achieve significant run time reduction. Table I shows the simulation time in seconds of different implementations for a lossy boson sampling experiment with 12 modes, 10 input squeezed modes, bond dimensions 1024 and 8192, photon loss rate 0.55, and local Hilbert-space dimension

TABLE I. Simulation time in seconds.

	CPU	Single GPU	One node	Six nodes
$\chi = 1024$	7966	126	60	42
$\chi = 8192$	>259000	2066	1045	322

15. Overall, the fully parallel implementation on six nodes (four NVIDIA A100 GPUs each) is on the order of 1000 times faster than the 32-core CPU implementation.

B. Analytic asymptotic entanglement entropy scaling

We provide the asymptotic MPO EE scaling under various loss conditions. Specifically, we consider cases where the number of photons surviving before measurement N_{out} scales with the number of input photons N , with different scaling exponents $0 < \gamma \leq 1$ ($N_{\text{out}} = \beta N^\gamma$ and the transmission rate is $\mu = \beta N^\gamma / N$, where β is a constant). Specifically, $\gamma = 1$ corresponds to a constant photon loss rate as the system size grows, whereas $0 < \gamma < 1$ corresponds to increasing loss rates as the system size grows. This is a reasonable scaling because loss should increase with the system size due to various experimental limitations such as an increase in the depth of beam splitters required to obtain a sufficiently Haar random state.

After derivations shown in Appendix B, we obtain the following scaling for the MPO EE:

$$\begin{aligned} S_1(|\hat{\rho}\rangle) &= O\left[N\left(\frac{\beta N^\gamma}{N}\right)^2 \log_2\left(\frac{\beta N^\gamma}{N}\right)\right] \\ &= O(N^{2\gamma-1} \log_2 N). \end{aligned} \quad (8)$$

Similarly, for the Rényi entropy with $\alpha < 1$, we have

$$S_\alpha = O(N^{1-2(1-\gamma)\alpha}). \quad (9)$$

For $\alpha > 1$, we have

$$S_\alpha = O\left(\frac{\alpha}{1-\alpha} N^{2\gamma-1}\right). \quad (10)$$

The MPO EE scaling becomes logarithmic when $\gamma = 1/2$. For an MPS algorithm, a logarithmic scaling of the MPO EE already rigorously implies a polynomial time complexity for the tensor network algorithm at fixed two-norm distance between the ideal and approximate state. This implies efficient fixed fidelity simulation. The situation for the MPO algorithm is trickier. The logarithmic MPO EE now implies efficient simulation for fixed two-norm distance between the vectorized states, which is also the two-norm distance between the density operators. However, for fixed fidelity, one needs to bound the one-norm distance, and the relationship $K\|A\|_2 \geq \|A\|_1$, where K is the dimension of the Hilbert space, means that the one norm cannot be efficiently bounded. In some cases, the MPO EE decreases as the system size increases, reducing the required bond dimension to bound the two-norm distance, but the required bond dimension to bound the one-norm distance may still increase. This is the case for a sufficiently low γ such as $\gamma = \frac{1}{4}$.

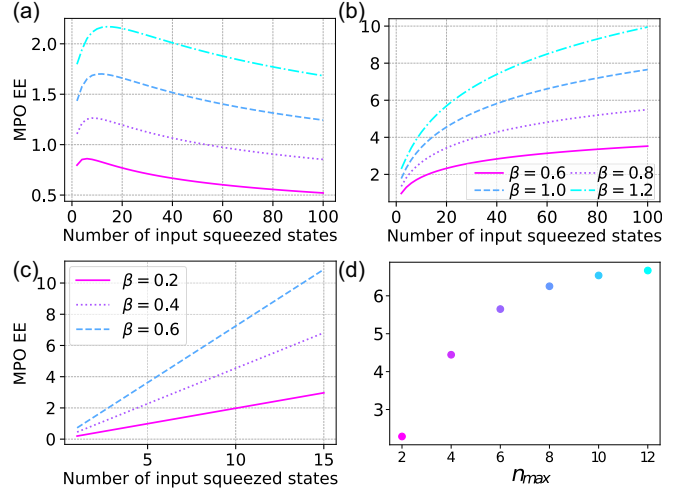


FIG. 1. Operator entanglement entropy vs the number of input squeezed modes for different photon survival scaling $N_{\text{out}} = \beta N^\gamma$ at $r = 0.88$. (a) $\gamma = \frac{1}{4}$. (b) $\gamma = \frac{1}{2}$. (c) $\gamma = 1$. (d) Convergence of MPO EE with increasing n_{max} for $N = 50, \beta = 1, \gamma = \frac{1}{2}, r = 0.88$.

C. Numerical asymptotic estimates of the entanglement entropy

We estimate the asymptotic MPO EE under photon survival scaling $N_{\text{out}} \propto N^\gamma$ with $\gamma = \frac{1}{4}, \frac{1}{2}, 1$. To make a fair comparison against SPBS, the squeezing parameter is fixed at $r = 0.88$, which averages to approximately one photon per squeezed mode. Appendix C discusses how we obtain the estimates for very large system sizes where direct MPO simulations are impractical.

Figure 1 shows the asymptotic estimates with $n_{\text{max}} = 8$ (maximum number of photons per density operator ρ_j that is simulated, see Appendix C) for large system sizes. Similar to what is observed in SPBS simulations, GBS shows MPO EE reduction when the loss is sufficiently high for $\gamma = \frac{1}{4}$, logarithmic scaling for $\gamma = \frac{1}{2}$, and linear scaling for $\gamma = 1$. A similar linear increase in MPO EE with β is also observed in all three cases. Further, we also show the numerical convergence of our asymptotic MPO EE estimates by increasing the cutoff of the initial maximum photon number n_{max} for the squeezed states.

D. Finite-size entanglement entropy from simulations

We further conduct full MPO simulations of GBS and numerically calculate the MPO EE. The MPO EE obtained from the full simulations and asymptotic estimates agree quantitatively, as shown in Fig. 2. However, we observe that the quality of agreement is poor when MPO EE is small such as in many $\gamma = \frac{1}{4}$ data points when the number of input squeezed states N is small. In the regime of small MPO EE but large N , we attribute the disagreement to the formal differences between regular MPOs and MPOs in a $U(1)$ symmetric form. This is easy to see as even a product state can have nonzero $U(1)$ symmetric MPO EE simply due to the existence of different charges. For small N , we expect the quality of the approximation to be poor because we are no longer in the asymptotic limit. Further disagreement can also be attributed to the fact that the $U(1)$ symmetric full simulations are limited

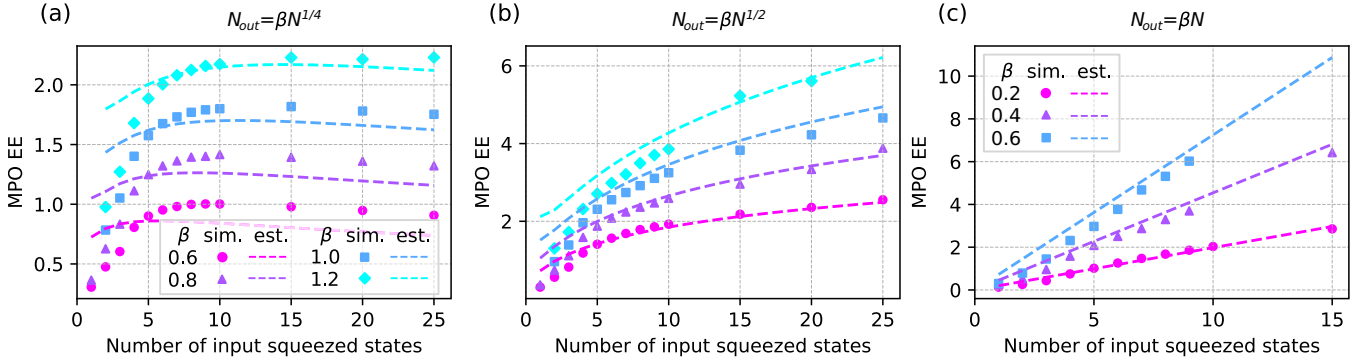


FIG. 2. Operator entanglement entropy vs the number of input squeezed modes for different photon survival scaling $N_{out} = \beta N^\gamma$ at $r = 0.88$. Details of experiment configurations can be found in Methods. (a) $\gamma = \frac{1}{4}$. (b) $\gamma = \frac{1}{2}$. (c) $\gamma = 1$. Dots are results obtained from full simulations using U(1) symmetry. Dashed lines are estimates using asymptotic assumptions.

by the bond dimension. We ensure that all plotted data points are simulated to $1 - \text{Tr}(\hat{\rho}) < 0.1$, which previous work established as a good proxy to the fidelity and the total variation error that is computationally lightweight [31,32].

Lastly, we investigate the effect of squeezing on MPO EE with our full U(1) symmetric simulations. We choose to investigate $\gamma = \frac{1}{4}$ for easier simulation. Figure 3 shows an increase in MPO EE with increasing squeezing parameter r . It is important to note that the average number of output photons scales with the average number of input photons N , not the number of squeezed states. This means that for the same number of input squeezed states and β , a higher squeezing parameter has a higher loss. Increasing the average number of photons per squeezed mode from 1 to 2 and 4 only moderately increases the MPO EE compared to increasing N . This observation is similar to the previous finding for Fock state boson sampling: if the number of input modes stays the same and the number of photons per mode increases, the MPO EE grows slowly and can be efficiently simulated [32].

Our numerical findings on the MPO EE growth for different loss scalings have complexity implications, but there is a lack of rigorous correspondence between MPO EE and simulation time. To make the statement on simulation complexity more direct, we validate the bond dimension growth explicitly. This is helpful in particular because the computational complexity is qubic in the bond dimension, both due

to SVD and matrix multiplication. We show in Fig. 4 the growth of bond dimension in the system size for fixed accuracy of $1 - \text{Tr}(\hat{\rho}) = 0.02$. Previous work has established that $1 - \text{Tr}(\rho)$ is a good proxy for the fidelity [31] and the total variational distance [32], which is the gold standard benchmark for boson sampling sample quality. It is clear that constant loss leads to exponential growth in the bond dimension. In higher loss cases, growth is much more moderate and appears subexponential. We also validate that increasing the bond dimension efficiently reduces the simulation error. We choose three experiments and simulate them with different bond dimensions.

IV. DISCUSSION

We show analytically that the matrix-product operator entanglement entropy of boson sampling scales logarithmically under high loss, which we numerically verify using U(1) symmetric tensor networks. We also numerically observe that increasing the photon number by squeezing has little impact compared to increasing the number of input squeezed modes. The computational complexity is also directly studied by calculating the bond dimension. This extends the previous entanglement entropy results for single-photon boson sampling to the more experimentally relevant Gaussian boson sampling, and extends the efficient simulation results

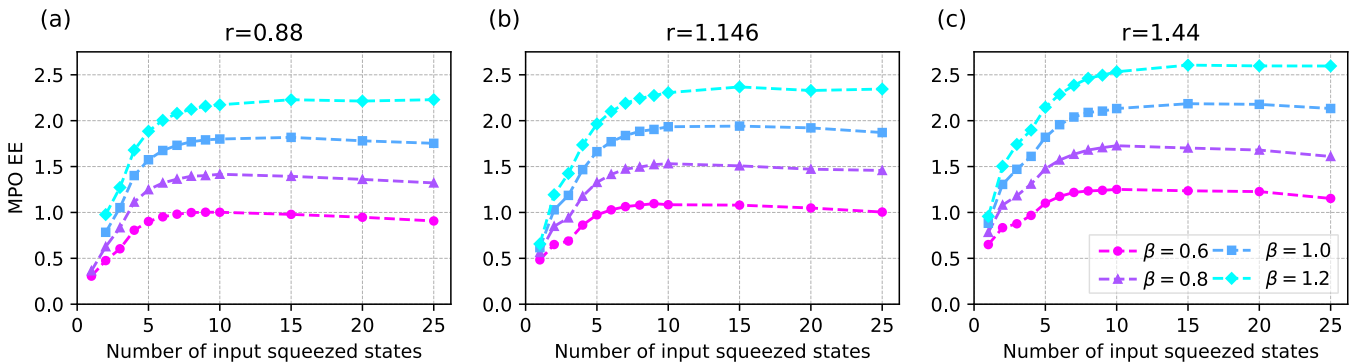


FIG. 3. Operator entanglement entropy vs the number of input squeezed modes for different squeezing parameters r . Dashed lines are guides to the eye. (a) $r = 0.88$, averaging approximately one photon per mode. (b) $r = 1.146$, averaging approximately two photons per mode. (c) $r = 1.44$, averaging approximately four photons per mode.

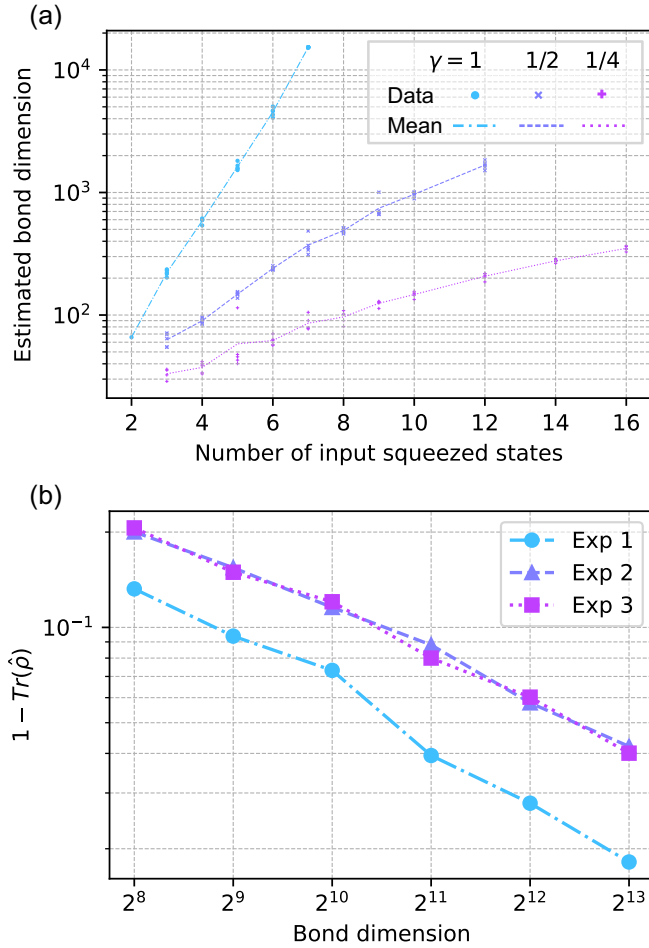


FIG. 4. Analysis of bond dimension, system size, and error. Details can be found in Methods. (a) Bond dimension needed to reach accuracy $1 - \text{Tr}(\hat{\rho}) = 0.02$ vs the number of input squeezed modes' photon survival scaling $N_{\text{out}} = 0.4N^\gamma$ at $r = 0.88$. Dots are individual estimates of the bond dimension obtained from full simulations using U(1) symmetry. Dashed lines are the means. Traces have higher estimated bond dimensions in ascending order in γ . (b) Reduction in $1 - \text{Tr}(\hat{\rho})$ error as bond dimension increases for three different experimental configurations.

using classical state approximation algorithms to an algorithm with controllable error, a necessary condition for simulating intermediate-size experiments with practical transmission rates. As a result, our analysis is more relevant to current quantum supremacy boson sampling experiments.

Although the MPO formalism intrinsically assumes a 1D architecture of the interferometer, the fact that we are simulating Haar random unitaries means that our findings are architecture independent. However, if we want to simulate high-dimensional low depth systems that are not Haar random [28,67], one potential direction to move forward is to adopt more exotic tensor networks such as projected entanglement pair states, and U(1) symmetric forms of generic tensor networks can be constructed in principle [58].

Data used to generate the figures are available upon request from the authors. The code used to generate the data and figures is available in the GitHub repository [68].

ACKNOWLEDGMENTS

This research used the resources of the Argonne Leadership Computing Facility, which is a U.S. Department of Energy (DOE) Office of Science User Facility supported under Grant No. DE-AC02-06CH11357. Y.A. acknowledges support from the Office of Science, U.S. Department of Energy, under Grant No. DE-AC02-06CH11357 at Argonne National Laboratory and Defense Advanced Research Projects Agency under Grant No. HR001120C0068. L.J. acknowledges support from the U.S. Army Research Office (ARO) (Grant No. W911NF-23-1-0077), ARO Multidisciplinary University Initiative (MURI) (Grant No. W911NF-21-1-0325), AFOSR MURI (Grants No. FA9550-19-1-0399 and No. FA9550-21-1-0209), U.S. Air Force Research Laboratory (Grant No. FA8649-21-P-0781), DOE Q-NEXT, NSF (Grants No. OMA-1936118, No. ERC-1941583, and No. OMA-2137642), NTT Research, and the Packard Foundation (Grant No. 2020-71479). J.L. is supported in part by IBM Quantum through the Chicago Quantum Exchange, and the Pritzker School of Molecular Engineering at the University of Chicago through AFOSR MURI (Grant No. FA9550-21-1-0209). M.L. acknowledges support from DOE Q-NEXT. C.O. acknowledges support from the ARO (Grants No. W911NF-18-1-0020 and No. W911NF-18-1-0212), ARO MURI (Grant No. W911NF-16-1-0349), AFOSR MURI (Grants No. FA9550-19-1-0399 and No. FA9550-21-1-0209), DOE Q-NEXT, NSF (Grants No. EFMA-1640959, No. OMA-1936118, and No. EEC-1941583), NTT Research, and the Packard Foundation (Grant No. 2013-39273).

M.L. developed the majority of the software, performed all numerical simulations and theoretical derivations, and wrote the majority of the paper. C.O. provided the original CPU implementation, developed the MPO initialization scheme, performed preliminary theoretical work, and contributed significantly to editing of the paper. All other authors contributed ideas, provided numerous scientific and writing improvements about the paper, and participated in discussions that shaped the project in a substantial manner and the understanding of its broader impact.

APPENDIX A: DETAILS OF NUMERICAL EXPERIMENTS

We carry out our GPU numerical simulations using the Polaris system at the Argonne Leadership Computing Facility (ALCF). Each node has a single 2.8-GHz AMD EPYC Milan 7543P 32-core CPU with 512 GB of DDR4 RAM and four Nvidia A100 GPUs connected via NVLink. We perform our CPU numerical simulations using the Bebop system at ALCF. Each node has a single 2.10-GHz Intel Xeon E5-2695v4 32-core CPU with 128 GB of DDR4 RAM.

All full U(1) symmetric MPO simulations have $M = \max(20, 4N)$ modes, and the local Hilbert-space dimension d is chosen such that $< 1\%$ of the probability is truncated. Because the leftmost charge is the sum of all photons, the required d is higher for higher squeezing parameters r and higher numbers of input squeezed states. Global Haar random interferometers are used and constructed using an M -layer array [69].

For Figs. 2 and 3, we simulate the system until there is no MPO EE increase for at least ten layers. This is reasonable since we only care about the maximum MPO EE throughout simulation which captures the computational cost, and the MPO EE generically increases as more depths are simulated until saturation, after which the MPO EE decreases slowly. All data points of MPO EE are obtained by only a single experiment, as we observe that no significant noise is present in our results because we only extract the maximum. We ensure that all plotted data points are simulated to $1 - \text{Tr}(\hat{\rho}) < 0.1$, which previous work established as a good proxy to the total variation error that is computationally lightweight. The largest simulation is for linear scaling simulations with $d = 18$, $\beta = 0.4$, $N = 15$, $M = 60$, $\chi = 16\,384$ ran on ten Polaris nodes with 40 GPUs in total.

For Fig. 4, the full depth of the interferometer is simulated. This is because simulating each layer produces additional error, and therefore affects the required bond dimension. For each configuration in Fig. 4(a), the bond dimension starts with a small value and doubles if the error exceeds 0.02. Once the bond dimension is large enough to exceed the desired accuracy, the bond dimension is refined a few more times to obtain a more precise estimate. For linear scaling and seven input squeezed states, the simulation is expensive and we verified that setting $\chi = 15\,296$ produced $1 - \text{Tr}(\hat{\rho}) = 0.020, 0.022, 0.021, 0.021, 0.019$ across five different experiments and used 15 296 as the estimated bond dimension. The three experiments for Fig. 4(b) are $M = 10, 15, 20$ and $\mu = 0.2, 0.15, 0.1$, respectively, and each data point is obtained from one simulation.

APPENDIX B: PROOF OF ASYMPTOTIC ENTANGLEMENT ENTROPY SCALING

In this section, we discuss the derivation of the operator EE scaling. We consider boson sampling where N independent input optical modes are sent into a linear optical interferometer. The interferometer has M modes, which can be larger than N , making $M - N$ modes at the input vacuum states. As photons interact throughout the interferometer, the quantum state gets transformed according to a unitary matrix describing the interferometer, and photons eventually exit the M optical modes with nontrivial correlation. For boson sampling, the claim is that this process is hard to simulate for a sufficiently random unitary describing the interferometer.

Formally, the quantum state of N independent and identical modes can be written as

$$|\psi_{\text{in}}\rangle = \otimes_{j=1}^N |\psi\rangle_j = \otimes_{j=1}^N \left(\sum_{n=0}^{\infty} c_n \frac{\hat{a}_j^{\dagger n}}{\sqrt{n!}} \right) |0\rangle, \quad (\text{B1})$$

where \hat{a}_j^{\dagger} is the input creation operator for mode j . The corresponding density operator is

$$\hat{\rho}_{\text{in}} = \otimes_{j=1}^N |\psi\rangle_j \langle\psi|_j = \otimes_{j=1}^N \hat{\rho}_{j,\text{in}}, \quad (\text{B2})$$

where the single input mode density operator is

$$\hat{\rho}_{j,\text{in}} = \sum_{n,m=0}^{\infty} c_n c_m^* |n\rangle_j \langle m|_j. \quad (\text{B3})$$

The action of an M -mode beam splitter array is to transform input creation operators:

$$\hat{a}_j^{\dagger} \rightarrow \hat{b}_j^{\dagger} = \sum_{k=1}^M U_{jk} \hat{a}_k^{\dagger}, \quad (\text{B4})$$

where \hat{b}_j^{\dagger} is the output creation operator for mode j . To study the entanglement entropy between bipartitions separated at the l th mode, we can define the normalized up and down bipartition creation operators $\hat{B}_{u,j}^{\dagger}, \hat{B}_{d,j}^{\dagger}$ as

$$\cos \theta_j \hat{B}_{u,j}^{\dagger} = \sum_{k=1}^l U_{jk} \hat{a}_k^{\dagger}, \quad \sin \theta_j \hat{B}_{d,j}^{\dagger} = \sum_{k=l+1}^M U_{jk} \hat{a}_k^{\dagger}, \quad (\text{B5})$$

with normalizations

$$\cos^2 \theta_j = \sum_{k=1}^l |U_{jk}|^2, \quad \sin^2 \theta_j = \sum_{k=l+1}^M |U_{jk}|^2. \quad (\text{B6})$$

In the collision-free cases where $M \geq N^2$, the bipartition creation operators satisfy the canonical commutation relations

$$\begin{aligned} [\hat{B}_{u,j}, \hat{B}_{u,k}^{\dagger}] &= \delta_{jk}, & [\hat{B}_{d,j}, \hat{B}_{d,k}^{\dagger}] &= \delta_{jk}, \\ [\hat{B}_{u,j}, \hat{B}_{d,k}^{\dagger}] &= 0, & [\hat{B}_{u,j}, \hat{B}_{d,k}^{\dagger}] &= 0. \end{aligned} \quad (\text{B7})$$

As a result, one can define the mutually orthogonal bipartition number states:

$$\hat{B}_{\text{side},j}^{\dagger k} |0\rangle = \frac{|k\rangle_{\text{side},j}}{\sqrt{k!}}, \quad \text{side} \in \{u,d\}. \quad (\text{B8})$$

The above formalism, described in [32], allows us to calculate the MPO EE without explicitly constructing the output state given the unitary representing the interferometer. Specifically, the details of the unitary matrix are hidden in the $|k\rangle_{\text{side},j}$ states constructed to satisfy orthogonality.

In this picture, the action of the unitary is to transform the input basis in the following way:

$$|n\rangle_j \rightarrow \sum_{k_j=0}^n \sqrt{\binom{n}{k_j}} \cos^{k_j} \theta_j \sin^{n-k_j} \theta_j |k_j\rangle_{u,j} |n-k_j\rangle_{d,j}, \quad (\text{B9})$$

and, therefore (omitting the j index),

$$\langle k_u, k_d | U | n \rangle = \sqrt{\binom{n}{k_u}} \cos^{k_u} \theta \sin^{k_d} \theta \delta(k_u + k_d - n). \quad (\text{B10})$$

If we apply this basis transform due to the unitary to the input lossless density operator $\hat{\rho}_{\text{in}}$, each single mode input density operator $\hat{\rho}_{j,\text{in}}$ in Eq. (B2) would transform independently. The full density operator remains a product of input modes in the new basis, and we have

$$\hat{\rho}_{\text{out}} = \otimes_j^N \hat{\rho}_{j,\text{out}}. \quad (\text{B11})$$

Although each $\hat{\rho}_{j,\text{out}}$ can be identified with an input mode j , $\hat{\rho}_{j,\text{out}}$ is no longer a single mode state, and is instead supported over all modes. Each $\hat{\rho}_{j,\text{out}}$ has some EE because it describes a state over both partitions, and the full system EE is additive in j due to the tensor product structure of $\hat{\rho}_{\text{out}}$. Therefore, the system EE scales linearly with the number of input modes N ,

and classical simulation of lossless boson sampling is always inefficient in N .

We can extend this analysis to lossy cases. Assuming that loss is uniform throughout the interferometer, loss commutes with all linear optical transforms and can be applied to the initial pure state. The basis transform (B9) due to the unitary is still independent on j , and the total output density operator is still in a product form with

$$\hat{\rho}_{j,\text{out}} = U \mathcal{E}_{\text{loss}}(\hat{\rho}_{j,\text{in}}) U^\dagger. \quad (\text{B12})$$

Therefore, the linear scaling of EE in N remains, and MPO simulation of lossy boson sampling is also inefficient in N .

However, as the number of input modes N increases, the complexity of the interferometer must grow as well in order to maintain reasonable randomness in the interferometer unitary and hardness of classical simulation. As a result, the depth of the interferometer should scale with the N , which leads to scaling of the transmission rate μ in N . The entanglement entropy for each $\hat{\rho}_{j,\text{out}}$ decreases as N increases, leading to an overall entanglement entropy that grows sublinearly, potentially allowing efficient simulation. To understand the scaling in loss and transmission, consider the Kraus operators corresponding to the single input state photon loss channel in the limit of small μ (from now on we ignore the mode index j):

$$\hat{\rho}_{\text{lossy}} = \mathcal{E}_{\text{loss}}(\hat{\rho}_{\text{in}}) = \sum_{n_{\text{loss}}=0}^{n_{\text{max}}} K^{(n_{\text{loss}})} \hat{\rho}_{\text{in}} K^{(n_{\text{loss}})\dagger}, \quad (\text{B13})$$

$$K^{(n_{\text{loss}})} = \sum_{n_{\text{out}}, n_{\text{in}}=0}^{n_{\text{max}}} K_{n_{\text{out}}, n_{\text{in}}}^{(n_{\text{loss}})} |n_{\text{out}}\rangle \langle n_{\text{in}}|, \quad (\text{B14})$$

$$K_{n_{\text{out}}, n_{\text{in}}}^{(n_{\text{loss}})} = \begin{cases} \sqrt{\binom{n_{\text{in}}}{n_{\text{out}}}} \mu^{n_{\text{out}}} (1-\mu)^{n_{\text{loss}}} & \text{if } n_{\text{in}} - n_{\text{out}} = n_{\text{loss}} \\ 0 & \text{otherwise,} \end{cases} \quad (\text{B15})$$

where $K^{(n_{\text{loss}})} \in \mathbb{C}^{n_{\text{max}}+1, n_{\text{max}}+1}$ captures processes that lose n_{loss} photons, and we limit the maximum photon number to n_{max} . The lossy density operator can be given in the input $|n\rangle$ basis in index notation:

$$\begin{aligned} \rho_{\text{lossy}, m, n} &= \sum_{n_{\text{loss}}=0}^{n_{\text{max}}} \sum_{k, l} K_{m, k}^{(n_{\text{loss}})} \rho_{\text{in}, k, l} K_{l, n}^{(n_{\text{loss}})\dagger} \\ &= \sum_{n_{\text{loss}}=0}^{n_{\text{max}}} \mathcal{O}(\mu^{\frac{m}{2}}) \rho_{\text{in}, m+n_{\text{loss}}, n+n_{\text{loss}}} \mathcal{O}(\mu^{\frac{n}{2}}) \\ &= \mathcal{O}(\mu^{\frac{m+n}{2}}), \end{aligned} \quad (\text{B16})$$

where the second line is due to the requirement that $k - m = n_{\text{loss}}$ and $l - n = n_{\text{loss}}$ from nonzero Kraus operator elements.

We can now apply the basis transform due to the unitary as described in Eq. (B9):

$$\hat{\rho}_{\text{out}} = U \hat{\rho}_{\text{lossy}} U^\dagger = U \left(\sum_{m, n=0}^{n_{\text{max}}} |m\rangle \rho_{\text{lossy}, m, n} \langle n| \right) U^\dagger. \quad (\text{B17})$$

In index notation in the bipartition number state basis,

$$\begin{aligned} \rho_{\text{out}, k_u, k_d; k'_u, k'_d} &= \langle k_u, k_d | \hat{\rho}_{\text{out}} | k'_u, k'_d \rangle \\ &= \sum_{m, n=0}^{n_{\text{max}}} \langle k_u, k_d | U | m \rangle \rho_{\text{lossy}, m, n} \langle n | U^\dagger | k'_u, k'_d \rangle. \end{aligned} \quad (\text{B18})$$

Substituting Eqs. (B10) and (B16) into the above expression yields

$$\rho_{\text{out}, k_u, k_d; k'_u, k'_d} = \sum_{n_{\text{loss}}=0}^{n_{\text{max}}} \rho_{\text{in}, k_u+k_d+n_{\text{loss}}, k'_u+k'_d+n_{\text{loss}}} \mathcal{O}(\mu^{\frac{k_u+k_d+k'_u+k'_d}{2}}). \quad (\text{B19})$$

To compute the contribution to the full system MPO entanglement entropy from $\hat{\rho}_{j,\text{out}}$, we need to vectorize the density operator to obtain $|\hat{\rho}_{j,\text{out}}\rangle\rangle$ so that we can pretend it is a pure state and compute its entanglement entropy. The standard procedure of computing the entanglement entropy of a pure state is to obtain the density operator by taking the outer product, obtain the reduced density operator by taking the partial trace over one subsystem, find the reduced density operator's eigenvalues, and take the log average of the eigenvalues. The only difference for the MPO entanglement entropy is that our "pure" state is actually a vectorized density operator, and the eigenvalues may not be normalized since the vectorized state is not L^2 normalized.

Vectorization of the density operator, which corresponds to flattening of the matrix and changing bras into kets, is defined as

$$\begin{aligned} \hat{\rho}_{\text{out}} &= \sum_{k_u, k_d, k'_u, k'_d} |k_u, k_d\rangle \rho_{\text{out}, k_u, k_d; k'_u, k'_d} \langle k'_u, k'_d| \\ \rightarrow |\hat{\rho}_{\text{out}}\rangle\rangle &= \sum_{k_u, k_d, k'_u, k'_d} \rho_{\text{out}, k_u, k_d; k'_u, k'_d} |k_u, k_d; k'_u, k'_d\rangle \\ &= \sum_{K_u, K_d} \rho_{\text{out}, K_u, K_d} |K_u, K_d\rangle, \end{aligned} \quad (\text{B20})$$

where K_{side} is defined as the combined index of k_{side} and k'_{side} . Next, we take the partial trace of its outer product over one bipartition to obtain $\hat{\rho}' = \text{tr}_u(|\hat{\rho}_{j,\text{out}}\rangle\rangle \langle\langle \hat{\rho}_{j,\text{out}}|)$, yielding

$$\begin{aligned} \rho'_{K_d, \bar{K}_d} &= \sum_{K_u} \rho_{\text{out}, K_u, K_d} \rho_{\text{out}, K_u, \bar{K}_d}^* \\ &= \sum_{k_u, k'_u} \mathcal{O}(\mu^{\frac{k_u+k_d+k'_u+k'_d}{2}}) \mathcal{O}(\mu^{\frac{k_u+\bar{k}_d+k'_u+\bar{k}'_d}{2}}) \\ &= \mathcal{O}(\mu^{\frac{k_d+k'_d+\bar{k}_d+\bar{k}'_d}{2}}), \end{aligned} \quad (\text{B21})$$

where \bar{K} is the dual of K , and $k_u, k'_u = 0$ terms are dominant.

The above analysis is general and independent of the input states. From now on, we will use the fact that the input state is a squeezed state. In GBS, $\rho_{\text{in}, m, n} = 0$ if either m or n is odd. Therefore, looking at Eq. (B19), $k_u + k_d + n_{\text{loss}}$ and $k'_u + k'_d + n_{\text{loss}}$ (or, more concisely, $k_u + k_d$ and $k'_u + k'_d$) must have the same parity for $\rho_{\text{in}, k_u+k_d+n_{\text{loss}}, k'_u+k'_d+n_{\text{loss}}}$ to be nonzero. This means that we do not have to consider terms like $\rho_{\text{out}, 0, 0; 0, 1}$, $\rho_{\text{out}, 1, 0; 0, 0}$, etc. As a result, no half-integer powers of

μ occur in any terms of the output density operator $\hat{\rho}_{\text{out}}$ or the reduced density operator $\hat{\rho}'$ of the vectorized state.

In this case, it turns out that $\hat{\rho}'$ has exactly one constant-order eigenvalue, it has no first-order eigenvalues, and all other eigenvalues are at least second order. To show this, it is sufficient to find all eigenvalues to the first order. Let us write down the form of $\hat{\rho}'$ to the first order, with the first row corresponding to $K_d = k_d = k'_d = 0$ and the first column corresponding to $\bar{K}_d = \bar{k}_d = \bar{k}'_d = 0$:

$$\hat{\rho}' = \begin{bmatrix} \hat{\rho}'_{1,1} & \hat{\rho}'_{1,2} & \hat{\rho}'_{1,3} & \cdots \\ \hat{\rho}'_{1,2}^* & 0 & 0 & \vdots \\ \hat{\rho}'_{1,3}^* & 0 & 0 & \vdots \\ \vdots & \cdots & \cdots & \ddots \end{bmatrix}, \quad (\text{B22})$$

which has eigenvalues

$$\lambda^2 = \frac{1}{2} \left(\hat{\rho}'_{1,1} \pm \sqrt{\hat{\rho}'_{1,1}{}^2 + 4 \sum_{n=2}^{(n_{\max}+1)^2} |\hat{\rho}'_{1,n}|^2} \right), \quad (\text{B23})$$

and all other eigenvalues are zero. Note that we denote the singular values of the Schmidt decompositions λ and the eigenvalues of the reduced density matrices λ^2 . However, $|\hat{\rho}'_{1,n}|^2$ is at least $O(\mu^2)$, and the Taylor expansion of the square root will be dominated by the constant- and first-order contributions from $\hat{\rho}'_{1,1}$. Therefore, the only nonzero first-order eigenvalue is $\lambda_1^2 = \hat{\rho}'_{1,1}$, which is $O(1)$.

The above analysis shows that to the second order, the eigenvalues are

$$\{a + b\mu + c\mu^2, O(\mu^2), O(\mu^2), O(\mu^2), \dots\}. \quad (\text{B24})$$

After normalization of the eigenvalues, the entropy contribution due to λ_1^2 is

$$\begin{aligned} & -\frac{a + b\mu + c_1\mu^2}{C} \log \frac{a + b\mu + c_1\mu^2}{C} \\ & = \frac{(d - c_1)\mu^2}{a \ln 2} + O(\mu^3) = O(\mu^2), \end{aligned} \quad (\text{B25})$$

where c_i is the second-order coefficient of λ_i^2 , $d = \sum_i c_i$, and $C = a + b\mu + d\mu^2$ is the normalization that must be treated explicitly and not as a constant. Contribution of other eigenvalues is

$$-\frac{c_i\mu^2}{C} \log_2 \frac{c_i\mu^2}{C} = O(-\mu^2 \log_2 \mu). \quad (\text{B26})$$

Overall, the entanglement entropy scales as $O(\mu^2 \log \mu)$. We would like to understand the scaling of the MPO EE under various loss scalings with the number of input optical modes. Generically, one can consider the situation where the number of surviving photons scales as $N_{\text{out}} \propto N^\gamma$, making the transmission rate $\mu = \beta N^\gamma / N$. Since the total entanglement entropy is the sum of all N input modes, we obtain

$$\begin{aligned} S_1(|\hat{\rho}\rangle) & = O\left[N \left(\frac{\beta N^\gamma}{N} \right)^2 \log_2 \left(\frac{\beta N^\gamma}{N} \right) \right] \\ & = O(N^{2\gamma-1} \log_2 N). \end{aligned} \quad (\text{B27})$$

Similarly, for the Rényi entropy, contribution from a single $\hat{\rho}_{j,\text{out}}$ is

$$\begin{aligned} & \frac{1}{1-\alpha} \log_2 \left[\left(\frac{a + b\mu + c_1\mu^2}{C} \right)^\alpha + \sum_{i \neq 1} \left(\frac{c_i\mu^2}{C} \right)^\alpha \right] \\ & \approx \frac{1}{(1-\alpha) \ln 2} \left(-\frac{d - c_1}{a} \alpha \mu^2 + \frac{1}{a} \sum_{i \neq 1} c_i^\alpha \mu^{2\alpha} \right). \end{aligned} \quad (\text{B28})$$

Therefore, for $\alpha < 1$, the second term dominates, and we have the familiar

$$S_\alpha = O(N^{1-2(1-\gamma)\alpha}). \quad (\text{B29})$$

Similarly, for $\alpha > 1$, the first term dominates, and we have

$$S_\alpha = O\left(\frac{\alpha}{1-\alpha} N^{2\gamma-1} \right). \quad (\text{B30})$$

APPENDIX C: METHOD OF ESTIMATING ASYMPTOTIC MPO EE

For our numerical estimates of GBS operator EE for very large system sizes where direct MPO simulations are impractical, we use the input wave function for a single squeezed mode,

$$|\psi_{\text{in}}\rangle = \frac{1}{\sqrt{\cosh r}} \sum_{n=0}^{n_{\max}/2} \frac{\tanh^n r}{2^n n!} \hat{a}^{\dagger 2n} |0\rangle, \quad (\text{C1})$$

follow the steps discussed above to compute the EE contribution from a single $\hat{\rho}_{j,\text{out}}$, and multiply by the number of input modes. Specifically, numerically we apply the photon loss Kraus operators defined earlier to this state (both the Kraus operators and the state are truncated to local Hilbert-space dimension n_{\max}). We then apply the basis transform due to Eq. (B9), and the operator is numerically stored in the bipartition $\{|k\rangle_u, |k\rangle_d\}$ basis. This transforms an $n_{\max} \times n_{\max}$ matrix into a $2n_{\max} \times 2n_{\max}$ matrix. This is then followed by vectorization, taking the partial trace of the outer product, finding the normalized eigenvalues, and computing the entropy.

The coefficients $\cos \theta_j$, $\sin \theta_j$ related to the unitary can also be simply approximated as $1/\sqrt{2}$ in the asymptotic limit of large M if l is chosen to be $M/2$ as can be seen from Eq. (B6). This choice of bipartition is justifiable because we are only interested in the largest EE bipartition since it implies a high simulation cost.

APPENDIX D: U(1) SYMMETRIC TENSOR NETWORK

In an MPS, the probability amplitude tensor c_{i_1, \dots, i_M} can be represented as

$$c_{i_1, \dots, i_M} = \sum_{\alpha_0, \dots, \alpha_M=0}^{\chi-1} \Gamma_{\alpha_0 \alpha_1}^{[1]i_1} \lambda_{\alpha_1}^{[1]} \Gamma_{\alpha_1 \alpha_2}^{[2]i_2} \lambda_{\alpha_2}^{[2]} \dots \lambda_{\alpha_{M-1}}^{[M-1]} \Gamma_{\alpha_{M-1} \alpha_M}^{[M]i_M}, \quad (\text{D1})$$

where χ is a free parameter called the bond dimension, which determines the accuracy of the approximate representation. For systems with higher entanglement, the required bond dimension to achieve a certain fidelity is higher. Overall, the

memory complexity of the MPS is determined by the M Γ matrix which has $O(\chi^2)$ complexity. This results in an overall memory complexity that is linear in M , contrary to the exponential cost of the exact state vector.

Intuitively speaking, the physical index i_k of $\Gamma^{[k]}$ captures the physical degree of freedom of particle k . Further, neighboring $\Gamma^{[k]}$ and $\Gamma^{[k+1]}$ tensors share a dummy index α_k which is contracted, capturing the entanglement between the two neighboring particles. The λ tensors correspond to the Schmidt coefficients if a Schmidt decomposition is performed on the wave function, and therefore capture entanglement. To apply a local unitary update on particle k and $k+1$, the unitary matrix needs to be contracted with the wave function at the physical indices, leading to the resulting tensor

$$\Theta_{\alpha_{k-1}\alpha_{k+1}}^{j_k, j_{k+1}} = \sum_{i_k, i_{k+1}=0}^{d-1} \sum_{\alpha_k=0}^{\chi-1} U_{i_k, i_{k+1}}^{j_k, j_{k+1}} \lambda_{\alpha_{k-1}}^{[k-1]} \Gamma_{\alpha_{k-1}\alpha_k}^{[k] i_k} \lambda_{\alpha_k}^{[k]} \Gamma_{\alpha_k \alpha_{k+1}}^{[k+1] i_{k+1}} \lambda_{\alpha_{k+1}}^{[k+1]}, \quad (\text{D2})$$

where the lower and upper indices of U represent input and output degrees of freedom, respectively.

The result of this computation is a single tensor of size $d^2 \chi^2$, which should be used in the new representation of the wave function to replace $\lambda^{[k-1]}$, $\Gamma^{[k] i_k}$, $\lambda^{[k]}$, $\Gamma^{[k+1] i_{k+1}}$, $\lambda^{[k+1]}$. However, this is no longer in the form of an MPS. To restore the MPS form, SVD is performed on Θ to produce

$$\Theta_{\alpha_{k-1}\alpha_{k+1}}^{j_k, j_{k+1}} = \sum_{\beta_k=0}^{d\chi-1} V_{(j_k, \alpha_{k-1}), \beta_k} \tilde{\lambda}_{\beta_k}^{[k]} W_{\beta_k, (j_{k+1}, \alpha_{k+1})}. \quad (\text{D3})$$

By retaining only the χ largest singular values, we can identify new Γ tensors as

$$\tilde{\Gamma}_{\alpha_{k-1}\alpha_k}^{[k] i_k} = V_{(j_k, \alpha_{k-1}), \beta_k} / \lambda_{\alpha_{k-1}}^{[k-1]}, \quad (\text{D4})$$

$$\tilde{\Gamma}_{\alpha_k \alpha_{k+1}}^{[k+1] i_{k+1}} = W_{\beta_k, (j_{k+1}, \alpha_{k+1})} / \lambda_{\alpha_{k+1}}^{[k+1]}, \quad (\text{D5})$$

which restores the MPS form.

Although MPS can efficiently represent many-body systems with controlled entanglement, it does not utilize any symmetry to further reduce the computational cost. To efficiently simulate U(1) symmetric systems, we need to modify the MPS formalism [32, 57, 59].

We denote the total number of particles to the right of position k corresponding to bond α_k as $c_{\alpha_k}^{[k]}$, then the probability amplitude tensor can be expressed as

$$c_{i_1, \dots, i_M} = \sum_{\alpha_0, \dots, \alpha_M=0}^{\chi-1} \Gamma_{\alpha_0 \alpha_1}^{[1]} \lambda_{\alpha_1}^{[1]} \Gamma_{\alpha_1 \alpha_2}^{[2]} \dots \lambda_{\alpha_{M-1}}^{[M-1]} \Gamma_{\alpha_{M-1} \alpha_M}^{[M]} \\ \times \prod_{k=1}^M \delta(c_{\alpha_{k-1}}^{[k-1]} - c_{\alpha_k}^{[k]} - i_k). \quad (\text{D6})$$

The δ function essentially determines the correct local particle number based on the charge value difference. Updating the wave function according to the unitary can be done with the following procedure. We first realize that a local two-site update does not change the charges at $k-1$ or $k+1$, and we can therefore compute the results for different resulting values of $c^{[k]}$. For each chosen value of $c^{[k]}$, $c^{[k-1]} \geq c^{[k]}$, and $c^{[k+1]} \leq c^{[k]}$, we can select a subset of bonds $\alpha_{k-1} \in$

\mathcal{A}_{k-1} , $\alpha_k \in \mathcal{A}_k$, $\alpha_{k+1} \in \mathcal{A}_{k+1}$ that satisfy the conditions on the three charges. We can then obtain the Θ tensor similar to the normal MPS algorithm:

$$\Theta_{\alpha_{k-1}\alpha_{k+1}}(c^{[k]}) \\ = \sum_{\substack{i_k, i_{k+1}=0 \\ j_k, j_{k+1}=0}}^{d-1} \sum_{\alpha_k \in \mathcal{A}_k} U_{j_k, j_{k+1}}^{i_k, i_{k+1}} \lambda_{\alpha_{k-1}}^{[k-1]} \Gamma_{\alpha_{k-1}\alpha_k}^{[k]} \lambda_{\alpha_k}^{[k]} \Gamma_{\alpha_k \alpha_{k+1}}^{[k+1]} \lambda_{\alpha_{k+1}}^{[k+1]} \\ \times \delta(c_{\alpha_{k-1}}^{[k-1]} - c_{\alpha_k}^{[k]} - j_k) \delta(c_{\alpha_k}^{[k]} - c_{\alpha_{k+1}}^{[k+1]} - j_{k+1}) \\ \times \delta(c_{\alpha_{k-1}}^{[k-1]} - c_{\alpha_{k+1}}^{[k+1]} - i_k) \delta(c_{\alpha_{k+1}}^{[k+1]} - c_{\alpha_{k+1}}^{[k+1]} - i_{k+1}), \quad (\text{D7})$$

where $0 \leq c^{[k]} \leq N$ and the δ function determines which entry of the unitary matrix to look up. Additionally, examining the U(1) symmetric MPS tells us that the Γ tensors lost their i indices corresponding to the physical degree of freedom (local particle number), reducing the memory complexity by a factor of d . This is instead captured by the size χ 1D charge tensors c . Second, the size of the Θ matrices that we decompose with SVD is also reduced to at most $\chi \times \chi$ instead of $\chi d \times \chi d$.

We similarly need to compress the new two-site tensor and restore the MPS representation as in the regular algorithm. The full Θ matrix capturing the result of the contraction should be a block-diagonal matrix with $\Theta(c^{[k]})$ as composing blocks. Therefore, performing SVD on the full matrix can be achieved by decomposing individual $\Theta(c^{[k]})$, which is the source of the computational complexity reduction of the U(1) symmetric algorithm. Our algorithm performs SVD on all $\Theta(c^{[k]})$ and keeps the χ largest singular values.

State-of-the-art simulations using tensor networks typically employ hardware acceleration, including the use of novel hardware platforms such as GPUs [70–73]. However, symmetry-preserving tensor network algorithms [57–59] are highly specialized and require data-dependent array entry lookup for the unitary matrix. This is an unusual requirement that is not commonly needed in normal tensor operations such as contraction, reshaping, index permutation, etc. As a result, no highly optimized hardware acceleration is readily available for our algorithm. In this paper, we aim to bridge this gap in hardware acceleration for the algorithm by optimizing a subroutine on GPU, and also target our implementation to supercomputing resources.

It is hard to improve SVD as it is a well-researched and optimized routine. The naive implementation of computing Θ also requires looping over all possible values of $c^{[k-1]}$ and $c^{[k+1]}$, which introduces an additional $O(d^2)$ complexity compared to SVD. Therefore, we focus our discussion on the Θ computation subroutine and how we optimize it.

1. CPU implementation

For a given center charge $c^{[k]}$, the CPU-based implementation loops through all possible left and right charge values $c^{[k-1]}$ and $c^{[k+1]}$ and selects a subset of left and right bonds α_{k-1} and α_{k+1} that satisfy the charge requirement. Since $c^{[k]}$ is fixed for each $\Theta(c^{[k]})$ submatrix, the only term that the delta function affects given the charges is U through $j_k, j_{k+1}, i_k, i_{k+1}$. With the correct unitary matrix value identified, the remaining computation is simply tensor contraction. Each iteration partially fills the $\Theta(c^{[k]})$ matrix at bonds

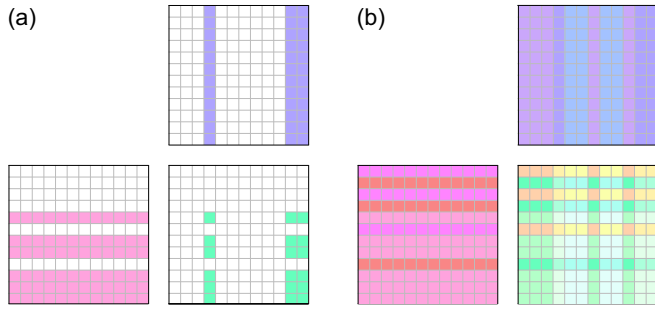


FIG. 5. Algorithms for computing Θ matrices. (a) CPU-based implementation. A subset of bonds is selected from $\Gamma^{[k]}, \Gamma^{[k+1]}$ that have the correct selected charge values $c^{[k-1]}, c^{[k+1]}$. A subset of Θ is computed. (b) GPU-based implementation. All bonds are used and the entire Θ matrix is computed at once.

$\alpha_{k-1}, \alpha_{k+1}$. Iterating over all possible left and right charges fills the entire matrix.

For large total particle number d , the $O(d^2)$ complexity due to the nested loop can significantly increase the computational time. Tensor contraction calculations that would otherwise be parallel have to be broken down into pieces. Therefore, the ability to parallelize across different left and right charges and unitary matrix entries is highly desirable, which is exactly what our GPU algorithm accomplishes. The differences between the CPU and GPU implementations are illustrated in Fig. 5.

2. Hierarchical GPU implementation

A naive parallel implementation of Θ matrix computation would assign the computation of a single array entry to a single thread. For example, the $\Theta_{i,j}$ can be calculated by a single thread that computes the inner product between the i th row of the first matrix and the j th column of the second matrix. However, this approach has several limitations, and a nontrivial hierarchical algorithm is used in reality for matrix multiplication. For a pedagogical introduction to the hierarchical approach in the context of matrix multiplication, see the work by Kerr *et al.* [74].

Consider multiplication of $A \in \mathbb{C}^{M \times K}$ and $B \in \mathbb{C}^{K \times N}$. The two matrices are stored in the *global memory* of the GPU, which every thread can access at any time. During inner product calculation of a single thread, the thread needs to read the global memory $2K$ times to complete the row and column vectors. Computing the whole matrix requires $2MNK$ reads of global memory, which turns out to be a limiting factor. Global memory is physically located far away from the compute cores of the GPU, and only a limited amount of memory can be fetched per second. The naive implementation would actually starve the compute cores due to a lack of data, leaving them idling most of the time.

Alternatively, we can replace elementwise inner products with the accumulation of outer products to reduce the memory read requirement, and Fig. 6 illustrates the differences between the two approaches. If a whole row or column of A or B is saved in some memory that is closer to the compute cores but has less capacity, all the threads can accumulate $A_{i,k}B_{k,j}$ once with an outer product. This can be repeated K times to

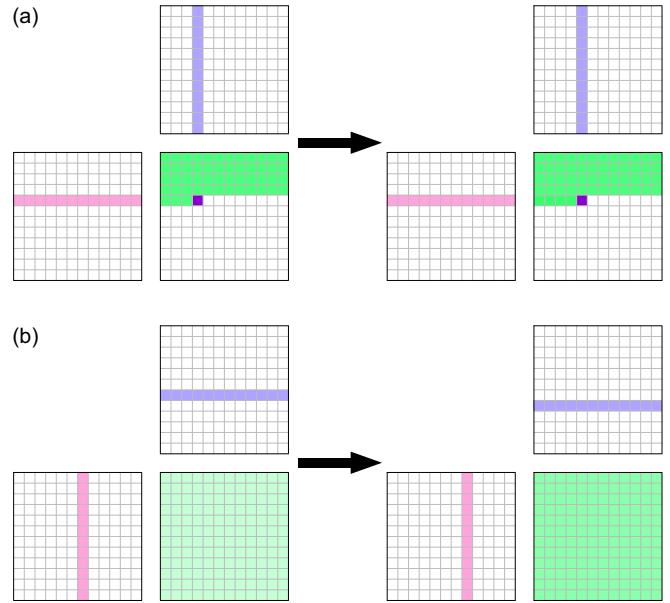


FIG. 6. Matrix multiplication with (a) inner products and (b) outer products.

complete matrix multiplication. On a GPU, this closer memory is called *shared memory*, which is shared by threads in its thread block of at most 2048 threads. Each thread block has its own shared memory. Each outer product requires transfer of data from global to shared memory with $M + N$ global reads, and the entire algorithm only needs $K(M + N)$ global reads and $2MNK$ shared memory reads. In reality, since a thread block has a limited number of threads and shared memory, we cannot fit everything in a single block and must compute the entire output matrix by sub-blocks.

The strategy of shifting the need for high memory access from large capacity broad access slow memory to small capacity local access fast memory can be repeated on lower levels. At the lowest level, a single thread actually computes multiple entries of the matrix, where data are stored in *registers* which are the fastest memory available and are private to each thread. Our GPU algorithm for the Θ computation subroutine only differs from matrix multiplication by U and λ value lookup. Therefore, our implementation adopts all the techniques mentioned above to maximize performance.

3. Memory alignment in GPU implementation

The charge data-dependent access of U poses difficulties in efficient GPU parallelization. In optimized numerical routines, threads access memory in an *aligned* manner, where consecutive threads access consecutive memory addresses, which allows data to be sent in chunks. Sending data chunks allows multiple units of data to be sent in a single clock cycle, otherwise only one unit of data is sent in a given cycle. In a GPU, this can lead to a 32-fold memory bandwidth reduction. If the charge values are completely unpredictable, the memory address of U that needs to be accessed will not be aligned.

This issue can be easily addressed by sorting the bonds according to the charge values. This leads to aligned memory access as illustrated in Fig. 7 and significantly improves

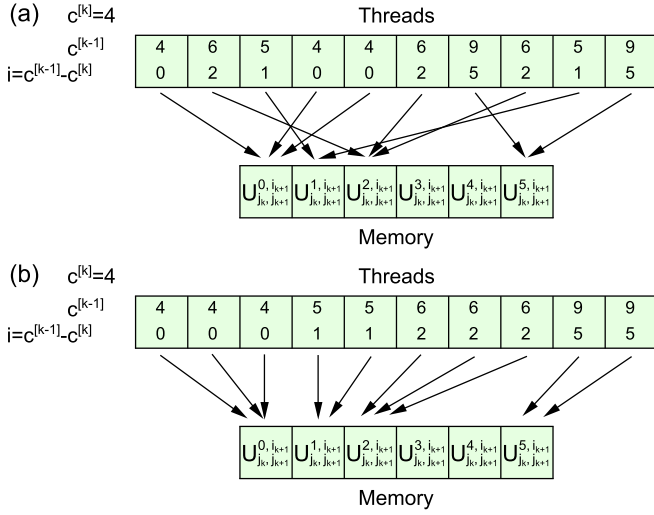


FIG. 7. Memory access pattern (a) without sorting and (b) with sorting.

performance. Additionally, since each thread calculates multiple entries, it might need to access multiple unitary values even after sorting. Due to the limited number of registers available to each thread, we cannot afford to store redundant unitary values. Therefore, we insert empty bonds to ensure that only one value of the unitary matrix corresponding to a single charge c and physical state i value is stored per thread. This scheme is illustrated in Fig. 8.

Additionally, bond indices are sorted such that $c_{\alpha_k}^{[k]}$ only increases as the bond index increases. For small d , this means that $c_{\alpha_k}^{[k]}$ is the same for many consecutive indices. This eliminates the need for threads to look up new U elements, except at boundaries where $c_{\alpha_k}^{[k]}$ changes. This further reduces the need for memory access and reduces latency.

4. High-level parallelization

Besides the numerical parallelization of individual SVD and Θ matrix computations through the use of GPUs,

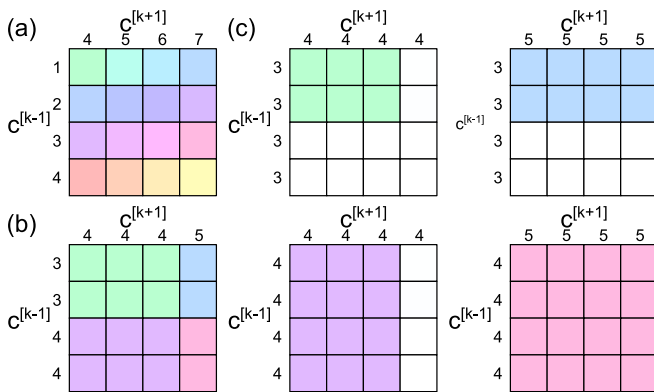


FIG. 8. Illustration of insertion of empty bonds. (a) Worst case scenario of charge value changes within a single fragment without empty bond insertion. The thread has to store 16 values of the unitary. (b) Generic case of a fragment at a charge change boundary. Less than 16 values need to be stored, but this is not known *a priori* and 16 values of the unitary still need to be stored. (c) With empty bond insertion, each thread only needs one unitary value.

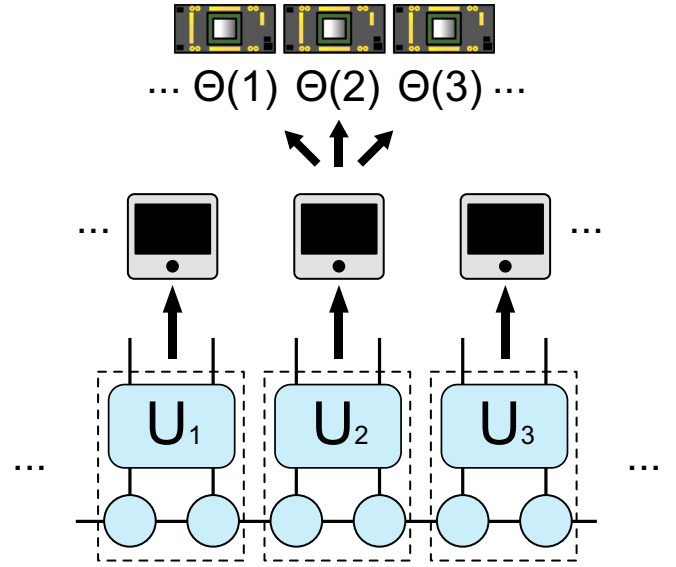


FIG. 9. High-level parallelization. Independent unitary gate updates are distributed to different nodes. Within each unitary update, $\Theta(c^{[k]})$'s are computed and decomposed with SVD independently on different GPUs.

additional parallelization is explicitly implemented on the algorithmic level. Further, for systems with large bond dimensions, storing the entire tensor network on a single GPU or even a single node may become prohibitive. We distribute the storage of individual Γ tensors to different nodes.

First, we parallelize independent two-site unitary updates. A host node identifies all parallel local unitary updates and keeps track of a list of available and busy nodes. Local unitary updates are allocated as soon as a node is available. During allocation, the compute process of the computational node requests the needed Γ , λ , c tensors from the storage processes of the corresponding storage nodes. Similar communication takes place after the computation to update the stored tensors. Second, for a single beam splitter MPO update, the overall Θ matrix is broken up into $\Theta(c^{[k]})$, which we compute and decompose in parallel. After the computation node receives the data needed, the data needed for each $\Theta(c^{[k]})$ are distributed to individual GPUs.

With the high-level parallelization discussed above and illustrated in Fig. 9, the algorithm can be easily scaled to supercomputers with multiple nodes and GPUs, especially when the system size is large. However, there are smaller systems that do not require multinode parallelization, and we provide implementations with intermediary parallelism as well to avoid the communication overhead of the fully parallel algorithm. On the lowest level, only one GPU is considered, and no distributed memory or computation is used. On the second level, all memory is managed by a single node, and unitary updates are distributed to individual GPUs instead of nodes.

5. Run time reduction

We evaluate the performance of our GPU supercomputing algorithm against the CPU-only implementation at the ALCF. All CPU simulations are performed with a single node of

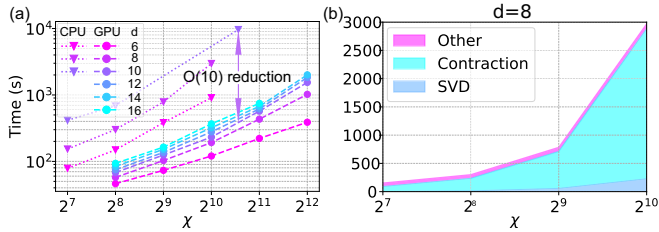


FIG. 10. (a) CPU and GPU simulation time. Traces have higher simulation time in ascending order in d . (b) Contribution to the CPU simulation time from subroutines.

the Bebop system with a 2.10-GHz Intel Xeon E5-2695v4 32-core CPU, and GPU simulations are performed on the Polaris system. A single node of the Polaris system has four Nvidia A100 GPUs. Table I shows the simulation time in seconds of different implementations for a lossy boson sampling experiment with 12 modes, two input squeezed modes, bond dimensions 1024 and 8192, photon loss rate 0.55, and local Hilbert-space dimension 15. Increasing the bond dimension χ increases the simulation accuracy and time. Moreover, lossy boson sampling requires the density matrix instead of the state vector, and the generalized algorithm is described in detail in by Oh *et al.* [32] The consequence of the density matrix generalization is that each charge can take on $15^2 = 225$ values instead of only 15, which means that the CPU-based algorithm needs to perform $225^2 = 50\,625$ iterations to fill the Θ matrix. On the other hand, our GPU algorithm computes all entries of Θ in parallel.

For the small χ experiment, we are able to simulate using only CPUs in a reasonable amount of time for comparison. Encouragingly, the single-GPU algorithm achieves a dramatic 63-time speedup even with a single-GPU. We further test the unitary-level parallel algorithm on one node and observe a further twofold speedup. Lastly, we use the fully parallel algorithm on six nodes, observing an additional 43% increase in the computational speed. We observe that the gain in computational speed by switching from less parallelized to highly parallelized implementation is less than the increase in computational resources. Higher-level parallelism incurs significant overhead, which indicates that there is still significant room for optimization.

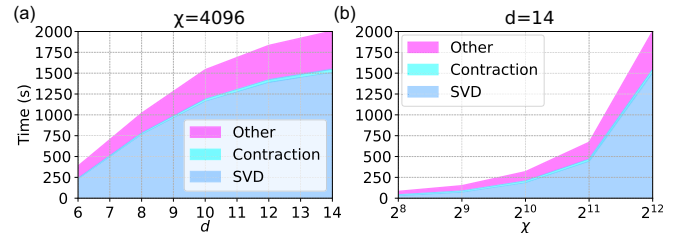


FIG. 11. Contribution to simulation time of the GPU algorithm from subroutines. (a) Bond dimension $\chi = 4096$. (b) Local Hilbert-space size $d = 14$.

Fortunately, this payoff in higher parallelism is more pronounced in the setting of larger system sizes. The CPU implementation failed to complete the simulation within the maximum allowed wall time of 72 h. This means that our single-GPU implementation achieves at least a 125-fold speedup. The computational time is further reduced twofold when going from the single-GPU implementation to the unitary-level parallel algorithm on one node, similar to the small bond dimension case. However, changing to the fully parallelized algorithm with six nodes further reduces the time more than threefold compared to a fractional reduction in the small bond dimension case. Overall, the fully parallel implementation on six nodes is on the order of 1000 times faster than the 32-core CPU implementation.

The exact speed-up depends on the system size, so we show more experiments with different configurations. All the following experiments are performed with $N = 5$, $M = 32$, $\mu = 0.5$, $r = 0.88$ on a single 32-core CPU or a single A100 GPU. We show the CPU and GPU simulation time for various systems in Fig. 10(a). For system sizes that the CPU can reasonably complete, we observe over 10 times speed up on a single GPU. Further, Fig. 10(b) shows that tensor contraction time dominates the overall run time due to the highly inefficient double nested loop. Figure 11 shows contributions to the GPU simulation time from subroutines. We see that the contribution from the tensor contraction step is minimal compared to the total simulation time thanks to the efficient custom kernel. Further, SVD takes up the majority of the simulation time, meaning that the overhead of tensor sorting, alignment, and storage is acceptable.

- [1] P. W. Shor, Algorithms for quantum computation: discrete logarithms and factoring, in *Proceedings of the 35th Annual Symposium on Foundations of Computer Science* (IEEE, New York, 1994), pp. 124–134.
- [2] L. K. Grover, A fast quantum mechanical algorithm for database search, in *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing, STOC '96* (Association for Computing Machinery, New York, 1996), pp. 212–219.
- [3] A. W. Harrow, A. Hassidim, and S. Lloyd, Quantum algorithm for linear systems of equations, *Phys. Rev. Lett.* **103**, 150502 (2009).
- [4] D. W. Berry, G. Ahokas, R. Cleve, and B. C. Sanders, Efficient quantum algorithms for simulating sparse Hamiltonians, *Commun. Math. Phys.* **270**, 359 (2007).

- [5] D. W. Berry, A. M. Childs, R. Cleve, R. Kothari, and R. D. Somma, Exponential improvement in precision for simulating sparse Hamiltonians, in *Proceedings of the Forty-Sixth Annual ACM Symposium on Theory of Computing, STOC '14* (Association for Computing Machinery, New York, 2014), pp. 283–292.
- [6] A. Childs, On the relationship between continuous- and discrete-time quantum walk, *Commun. Math. Phys.* **294**, 581 (2010).
- [7] G. H. Low and I. L. Chuang, Optimal Hamiltonian simulation by quantum signal processing, *Phys. Rev. Lett.* **118**, 010501 (2017).
- [8] Y. Alexeev, D. Bacon, K. R. Brown, R. Calderbank, L. D. Carr, F. T. Chong, B. DeMarco, D. Englund, E. Farhi, B. Fefferman, A. V. Gorshkov, A. Houck, J. Kim, S. Kimmel, M. Lange,

- S. Lloyd, M. D. Lukin, D. Maslov, P. Maunz, C. Monroe, J. Preskill, M. Roetteler, M. J. Savage, and J. Thompson, Quantum computer systems for scientific discovery, *PRX Quantum* **2**, 017001 (2021).
- [9] S. Aaronson and A. Arkhipov, The computational complexity of linear optics, in *Proceedings of the Forty-Third Annual ACM Symposium on Theory of Computing* (Association for Computing Machinery, New York, 2011), pp. 333–342.
- [10] M. A. Broome, A. Fedrizzi, S. Rahimi-Keshari, J. Dove, S. Aaronson, T. C. Ralph, and A. G. White, Photonic boson sampling in a tunable circuit, *Science* **339**, 794 (2013).
- [11] J. B. Spring, B. J. Metcalf, P. C. Humphreys, W. S. Kolthammer, X.-M. Jin, M. Barbieri, A. Datta, N. Thomas-Peter, N. K. Langford, D. Kundys *et al.*, Boson sampling on a photonic chip, *Science* **339**, 798 (2013).
- [12] M. Tillmann, B. Dakić, R. Heilmann, S. Nolte, A. Szameit, and P. Walther, Experimental boson sampling, *Nat. Photonics* **7**, 540 (2013).
- [13] A. Crespi, R. Osellame, R. Ramponi, D. J. Brod, E. F. Galvao, N. Spagnolo, C. Vitelli, E. Maiorino, P. Mataloni, and F. Sciarrino, Integrated multimode interferometers with arbitrary designs for photonic boson sampling, *Nat. Photonics* **7**, 545 (2013).
- [14] N. Spagnolo, C. Vitelli, M. Bentivegna, D. J. Brod, A. Crespi, F. Flamini, S. Giacomini, G. Milani, R. Ramponi, P. Mataloni *et al.*, Experimental validation of photonic boson sampling, *Nat. Photonics* **8**, 615 (2014).
- [15] J. Carolan, J. D. A. Meinecke, P. J. Shadbolt, N. J. Russell, N. Ismail, K. Wörhoff, T. Rudolph, M. G. Thompson, J. L. O’Brien, J. C. F. Matthews *et al.*, On the experimental verification of quantum complexity in linear optics, *Nat. Photonics* **8**, 621 (2014).
- [16] J. Carolan, C. Harrold, C. Sparrow, E. Martín-López, N. J. Russell, J. W. Silverstone, P. J. Shadbolt, N. Matsuda, M. Oguma, M. Itoh *et al.*, Universal linear optics, *Science* **349**, 711 (2015).
- [17] M. Bentivegna, N. Spagnolo, C. Vitelli, F. Flamini, N. Viggianiello, L. Latmiral, P. Mataloni, D. J. Brod, E. F. Galvão, A. Crespi *et al.*, Experimental scattershot boson sampling, *Sci. Adv.* **1**, e1400255 (2015).
- [18] H. S. Zhong, Y. Li, W. Li, L. C. Peng, Z. E. Su, Y. Hu, Y. M. He, X. Ding, W. Zhang, H. Li, L. Zhang, Z. Wang, L. You, X. L. Wang, X. Jiang, L. Li, Y. A. Chen, N. L. Liu, C. Y. Lu, and J. W. Pan, 12-photon entanglement and scalable scattershot boson sampling with optimal entangled-photon pairs from parametric down-conversion, *Phys. Rev. Lett.* **121**, 250505 (2018).
- [19] H.-S. Zhong, L.-C. Peng, Y. Li, Y. Hu, W. Li, J. Qin, D. Wu, W.-J. Zhang, H. Li, L. Zhang, Z. Wang *et al.*, Experimental Gaussian boson sampling, *Sci. Bull.* **64**, 511 (2019).
- [20] S. Paesani, Y. Ding, R. Santagati, L. Chakhmakhchyan, C. Vigiari, K. Rottwitt, L. K. Oxenløwe, J. Wang, M. G. Thompson, and A. Laing, Generation and sampling of quantum states of light in a silicon chip, *Nat. Phys.* **15**, 925 (2019).
- [21] Y. He, X. Ding, Z.-E. Su, H.-L. Huang, J. Qin, C. Wang, S. Unsleber, C. Chen, H. Wang, Y.-M. He *et al.*, Time-bin-encoded boson sampling with a single-photon device, *Phys. Rev. Lett.* **118**, 190501 (2017).
- [22] J. C. Loredó, M. A. Broome, P. Hilaire, O. Gazzano, I. Sagnes, A. Lemaitre, M. P. Almeida, P. Senellart, and A. G. White, Boson sampling with single-photon fock states from a bright solid-state source, *Phys. Rev. Lett.* **118**, 130503 (2017).
- [23] H. Wang, Y. He, Y.-H. Li, Z.-E. Su, B. Li, H.-L. Huang, X. Ding, M.-C. Chen, C. Liu, J. Qin *et al.*, High-efficiency multiphoton boson sampling, *Nat. Photonics* **11**, 361 (2017).
- [24] H. Wang, W. Li, X. Jiang, Y.-M. He, Y.-H. Li, X. Ding, M.-C. Chen, J. Qin, C.-Z. Peng, C. Schneider *et al.*, Toward scalable boson sampling with photon loss, *Phys. Rev. Lett.* **120**, 230502 (2018).
- [25] H. Wang, J. Qin, X. Ding, M.-C. Chen, S. Chen, X. You, Y.-M. He, X. Jiang, L. You, Z. Wang *et al.*, Boson sampling with 20 input photons and a 60-mode interferometer in a 10^{14} -dimensional hilbert space, *Phys. Rev. Lett.* **123**, 250503 (2019).
- [26] H.-S. Zhong, H. Wang, Y.-H. Deng, M.-C. Chen, L.-C. Peng, Y.-H. Luo, J. Qin, D. Wu, X. Ding, Y. Hu *et al.*, Quantum computational advantage using photons, *Science* **370**, 1460 (2020).
- [27] H. S. Zhong, Y. H. Deng, J. Qin, H. Wang, M. C. Chen, L. C. Peng, Y. H. Luo, D. Wu, S. Q. Gong, H. Su, Y. Hu, P. Hu, X. Y. Yang, W. J. Zhang, H. Li, Y. Li, X. Jiang, L. Gan, G. Yang, L. You, Z. Wang, L. Li, N. L. Liu, J. J. Renema, C. Y. Lu, and J. W. Pan, Phase-programmable gaussian boson sampling using stimulated squeezed light, *Phys. Rev. Lett.* **127**, 180502 (2021).
- [28] L. S. Madsen, F. Laudenbach, M. F. Askarani, F. Rortais, T. Vincent, J. F. F. Bulmer, F. M. Miatto, L. Neuhaus, L. G. Helt, M. J. Collins *et al.*, Quantum computational advantage with a programmable photonic processor, *Nature (London)* **606**, 75 (2022).
- [29] D. Aharonov, M. Ben-Or, R. Impagliazzo, and N. Nisan, Limitations of noisy reversible computation, [arXiv:quant-ph/9611028](https://arxiv.org/abs/quant-ph/9611028).
- [30] K. Noh, L. Jiang, and B. Fefferman, Efficient classical simulation of noisy random quantum circuits in one dimension, *Quantum* **4**, 318 (2020).
- [31] M. Zhang, C. Wang, S. Dong, H. Zhang, Y. Han, and L. He, Entanglement entropy scaling of noisy random quantum circuits in two dimensions, *Phys. Rev. A* **106**, 052430 (2022).
- [32] C. Oh, K. Noh, B. Fefferman, and L. Jiang, Classical simulation of lossy boson sampling using matrix product operators, *Phys. Rev. A* **104**, 022407 (2021).
- [33] F. Arute, K. Arya, R. Babbush, D. Bacon, J. C. Bardin, R. Barends, R. Biswas, S. Boixo, F. G. S. L. Brandao, D. A. Buell *et al.*, Quantum supremacy using a programmable superconducting processor, *Nature (London)* **574**, 505 (2019).
- [34] D. Aharonov, X. Gao, Z. Landau, Y. Liu, and U. Vazirani, A polynomial-time classical algorithm for noisy random circuit sampling, in *Proceedings of the 55th Annual ACM Symposium on Theory of Computing* (ACM, New York, 2023), pp. 945–957.
- [35] C. Oh, L. Jiang, and B. Fefferman, On classical simulation algorithms for noisy boson sampling, [arXiv:2301.11532](https://arxiv.org/abs/2301.11532).
- [36] M. Oszmaniec and D. J. Brod, Classical simulation of photonic linear optics with lost particles, *New J. Phys.* **20**, 092002 (2018).
- [37] R. García-Patrón, J. J. Renema, and V. Shchesnovich, Simulating boson sampling in lossy architectures, *Quantum* **3**, 169 (2019).
- [38] J. Renema, V. Shchesnovich, and R. Garcia-Patron, Classical simulability of noisy boson sampling, [arXiv:1809.01953](https://arxiv.org/abs/1809.01953).

- [39] H. Qi, D. J. Brod, N. Quesada, and R. García-Patrón, Regimes of classical simulability for noisy Gaussian boson sampling, *Phys. Rev. Lett.* **124**, 100502 (2020).
- [40] M. C. Tichy, Sampling of partially distinguishable bosons and the relation to the multidimensional permanent, *Phys. Rev. A* **91**, 022316 (2015).
- [41] J. J. Renema, A. Menssen, W. R. Clements, G. Triginer, W. S. Kolthammer, and I. A. Walmsley, Efficient classical algorithm for boson sampling with partially distinguishable photons, *Phys. Rev. Lett.* **120**, 220502 (2018).
- [42] V. S. Shchesnovich, Noise in boson sampling and the threshold of efficient classical simulatability, *Phys. Rev. A* **100**, 012340 (2019).
- [43] A. E. Moylett, R. García-Patrón, J. J. Renema, and P. S. Turner, Classically simulating near-term partially-distinguishable and lossy boson sampling, *Quantum Sci. Technol.* **5**, 015001 (2019).
- [44] J. Martínez-Cifuentes, K. M. Fonseca-Romero, and N. Quesada, Classical models may be a better explanation of the Jiuzhang 1.0 Gaussian Boson Sampler than its targeted squeezed light model, *Quantum* **7**, 1076 (2023).
- [45] A. P. Lund, A. Laing, S. Rahimi-Keshari, T. Rudolph, J. L. O'Brien, and T. C. Ralph, Boson sampling from a Gaussian state, *Phys. Rev. Lett.* **113**, 100502 (2014).
- [46] S. Barkhofen, T. J. Bartley, L. Sansoni, R. Kruse, C. S. Hamilton, I. Jex, and C. Silberhorn, Driven boson sampling, *Phys. Rev. Lett.* **118**, 020502 (2017).
- [47] L. Chakhmakhchyan and N. J. Cerf, Boson sampling with gaussian measurements, *Phys. Rev. A* **96**, 032326 (2017).
- [48] C. S. Hamilton, R. Kruse, L. Sansoni, S. Barkhofen, C. Silberhorn, and I. Jex, Gaussian boson sampling, *Phys. Rev. Lett.* **119**, 170501 (2017).
- [49] R. Kruse, C. S. Hamilton, L. Sansoni, S. Barkhofen, C. Silberhorn, and I. Jex, Detailed study of Gaussian boson sampling, *Phys. Rev. A* **100**, 032326 (2019).
- [50] Y.-H. Deng, Y.-C. Gu, H.-L. Liu, S.-Q. Gong, H. Su, Z.-J. Zhang, H.-Y. Tang, M.-H. Jia, J.-M. Xu, M.-C. Chen *et al.*, Gaussian Boson sampling with pseudo-photon-number-resolving detectors and quantum computational advantage, *Phys. Rev. Lett.* **131**, 150601 (2023).
- [51] M. Liu, C. Oh, J. Liu, L. Jiang, and Y. Alexeev, Supercomputing tensor networks for U(1) symmetric quantum many-body systems, [arXiv:2303.11409](https://arxiv.org/abs/2303.11409).
- [52] N. Quesada and J. M. Arrazola, Exact simulation of gaussian boson sampling in polynomial space and exponential time, *Phys. Rev. Res.* **2**, 023005 (2020).
- [53] J. F. F. Bulmer, B. A. Bell, R. S. Chadwick, A. E. Jones, D. Moise, A. Rigazzi, J. Thorbecke, U.-U. Haus, T. V. Vaerenbergh, R. B. Patel, I. A. Walmsley, and A. Laing, The boundary for quantum advantage in gaussian boson sampling, *Sci. Adv.* **8**, eab19236 (2022).
- [54] N. Quesada, R. S. Chadwick, B. A. Bell, J. M. Arrazola, T. Vincent, H. Qi, and R. García-Patrón, Quadratic speed-up for simulating Gaussian Boson sampling, *PRX Quantum* **3**, 010306 (2022).
- [55] B. Villalonga, M. Y. Niu, L. Li, H. Neven, J. C. Platt, V. N. Smelyanskiy, and S. Boixo, Efficient approximation of experimental Gaussian boson sampling, [arXiv:2109.11525](https://arxiv.org/abs/2109.11525).
- [56] Motivated by the evidence of efficient tensor network simulation of GBS under high loss, some authors of our paper also released new results on Gaussian boson sampling simulations using tensor networks during the review process of this paper. The new results allow them to significantly reduce the cost and simulate all supremacy experiments while obtaining better benchmarking results that can be verified with reasonable resources [75]. Efficient simulation using this modified tensor network representation under the $N_{\text{out}} \propto \sqrt{N}$ scaling is rigorously proven.
- [57] H.-L. Huang, W.-S. Bao, and C. Guo, Simulating the dynamics of single photons in boson sampling devices with matrix product states, *Phys. Rev. A* **100**, 032305 (2019).
- [58] S. Singh, R. N. C. Pfeifer, and G. Vidal, Tensor network states and algorithms in the presence of a global U(1) symmetry, *Phys. Rev. B* **83**, 115125 (2011).
- [59] C. Guo and D. Poletti, Matrix product states with adaptive global symmetries, *Phys. Rev. B* **100**, 134304 (2019).
- [60] M. Aizenman, E. H. Lieb, R. Seiringer, J. P. Solovej, and J. Yngvason, Bose-Einstein quantum phase transition in an optical lattice model, *Phys. Rev. A* **70**, 023612 (2004).
- [61] F. Alcaraz, U. Grimm, and V. Rittenberg, The XXZ Heisenberg chain, conformal invariance and the operator content of $c < 1$ systems, *Nucl. Phys. B* **316**, 735 (1989).
- [62] T. Kitagawa, M. S. Rudner, E. Berg, and E. Demler, Exploring topological phases with quantum walks, *Phys. Rev. A* **82**, 033429 (2010).
- [63] A. M. Childs, D. Gosset, and Z. Webb, Universal computation by multiparticle quantum walk, *Science* **339**, 791 (2013).
- [64] X. Cai, H. Yang, H.-L. Shi, C. Lee, N. Andrei, and X.-W. Guan, Multiparticle quantum walks and fisher information in one-dimensional lattices, *Phys. Rev. Lett.* **127**, 100406 (2021).
- [65] A. Schreiber, A. Gábris, P. P. Rohde, K. Laiho, M. Štefaňák, V. Potoček, C. Hamilton, I. Jex, and C. Silberhorn, A 2D quantum walk simulation of two-particle dynamics, *Science* **336**, 55 (2012).
- [66] U. Agrawal, A. Zabalo, K. Chen, J. H. Wilson, A. C. Potter, J. H. Pixley, S. Gopalakrishnan, and R. Vasseur, Entanglement and charge-sharpening transitions in U(1) symmetric monitored quantum circuits, *Phys. Rev. X* **12**, 041002 (2022).
- [67] A. Deshpande, A. Mehta, T. Vincent, N. Quesada, M. Hinsche, M. Ioannou, L. Madsen, J. Lavoie, H.-Y. Qi, J. Eisert *et al.*, Quantum computational advantage via high-dimensional Gaussian boson sampling, *Sci. Adv.* **8**, eabi7894 (2022).
- [68] <https://github.com/sss441803/BosonSampling>.
- [69] N. J. Russell, L. Chakhmakhchyan, J. L. O'Brien, and A. Laing, Direct dialling of haar random unitary matrices, *New J. Phys.* **19**, 033007 (2017).
- [70] T. Nguyen, D. Lyakh, E. Dumitrescu, D. Clark, J. Larkin, and A. McCaskey, Tensor network quantum virtual machine for simulating quantum circuits at exascale, *ACM Transactions on Quantum Computing* **4**, 1 (2022).
- [71] D. I. Lyakh, T. Nguyen, D. Claudino, E. Dumitrescu, and A. J. McCaskey, ExaTN: Scalable GPU-accelerated high-performance processing of general tensor networks at exascale, *Front. Appl. Math. Stat.* **8**, 8838601 (2022).
- [72] D. Lykov, A. Chen, H. Chen, K. Keipert, Z. Zhang, T. Gibbs, and Y. Alexeev, Performance evaluation and acceleration of the qtensor quantum circuit simulator on GPUs, in *Proceedings of the 2021 IEEE/ACM Second International Workshop on Quantum Computing Software* (IEEE, New York, 2021), pp. 27–34.

- [73] D. Lykov, R. Schutski, A. Galda, V. Vinokur, and Y. Alexeev, Tensor network quantum simulator with step-dependent parallelization, in *Proceedings of the 2022 IEEE International Conference on Quantum Computing and Engineering*, (IEEE, New York, 2022), pp. 582–593.
- [74] A. Kerr, D. Merrill, J. Demouth, and J. Tran, CUTLASS: Fast linear algebra in CUDA C++ (2017), <https://developer.nvidia.com/blog/cutlass-linear-algebra-cuda/>.
- [75] C. Oh, M. Liu, Y. Alexeev, B. Fefferman, and L. Jiang, Tensor network algorithm for simulating experimental Gaussian boson sampling, [arXiv:2306.03709](https://arxiv.org/abs/2306.03709).