Efficient tensor network contraction algorithms

Linjian Ma, Edgar Solomonik

Abstract

Tensors are multidimensional arrays that generalize the vector and matrix concepts. Formally-speaking, an N-way or Nth-order tensor is an element of the tensor product of N vector spaces. A scalar, vector, and matrix correspond to tensors of order zero, one, and two, respectively. One of the key challenges in working with high-order tensors is called the "curse of dimensionality", where tensors with large dimensionality can have an extremely large number of components, making it difficult to analyze and extract meaningful information from them. *Tensor networks* are powerful techniques for addressing this challenge. A tensor network [14] employs a collection of small tensors, where some or all of their dimensions are contracted according to some pattern, to implicitly represent a high-dimensional tensor. Tensor networks have been originally used in computational quantum physics [23, 22, 24, 21, 20, 19], where low-rank tensor networks can be used efficiently and accurately to represent quantum states and operators based on the area law. Recently, tensor networks are also widely used in simulating quantum computers [11, 25, 18, 17] and neural networks [13].

Tensor network contraction explicitly evaluates the single tensor represented by a given tensor network. When each tensor in the network is dense, tensor network contraction is typically achieved through a sequence of pairwise tensor contractions. This sequence, known as the *contraction path*, is determined by a topological sort of the underlying *contraction tree*. The contraction tree is a rooted binary tree that depicts the complete contraction of the tensor network. In this tree, the leaves correspond to the tensors in the network, and each internal vertex represents the tensor contraction of its two children.

Tensor network contraction has found diverse applications in different fields of research. For instance, in quantum computing, each quantum algorithm can be viewed as a tensor network contraction, making this method a useful tool for simulating quantum computers [11, 25, 18, 17]. In statistical physics, tensor network contraction has been used to evaluate the classical partition function of physical models defined on specific graphs [8]. Tensor network contraction has also been used for counting satisfying assignments of constraint satisfaction problems (#CSPs) [7]. In this approach, an arbitrary #CSP formula is transformed into a tensor network, where its full contraction yields the number of satisfying assignments of that formula.

Contracting tensor networks with arbitrary structure is #P-hard in the general case [3, 16, 1], even when the network represents a scalar. The reason for this is that during the contraction of general tensor networks, intermediate tensors with high orders or large dimension sizes can emerge, leading to a substantial computational cost for precise contraction. Nonetheless, in some applications such as many-body physics, it has been observed that tensor networks built on top of specific models can often be approximately contracted with satisfactory accuracy, without incurring exponential costs [15]. A common approach is to represent or approximate large intermediate tensors as (low-rank) tensor networks, which reduces the memory usage and computational overhead for downstream contractions. Common tensor networks used for approximation include the matrix product states (MPS) and the tree tensor networks (TTN) [20].

Efficient approximate contraction algorithms based on MPSs have been proposed for tensor network contractions defined on regular structures such as the Projected Entangled Pair States (PEPS)

[21, 22, 10, 9], which has a 2D lattice structure. However, these methods are not easily extendable to other general tensor network structures.

Recent works have proposed approximation algorithms for contracting tensor networks with more general graph structures. For example, [6] approximates each intermediate tensor produced during the contraction path as a binary tree tensor network, while [17] approximates each intermediate tensor as an MPS. In [2], each intermediate tensor is also approximated as an MPS, but the system is designed for the specific unbalanced contraction paths and only targets the approximate contraction of tensor networks defined on planar graphs. Another approach proposed in [5] is to perform low-rank approximation on the remaining tensor network after contractions, rather than on the intermediate tensors. The experimental results demonstrate that this framework is more efficient and accurate than [17].

We introduce two approximate tensor network contraction algorithms. First of all, we present a swap-based algorithm named Contracting Arbitrary Tensor Network with Global Ordering (CATN-GO) that can efficiently approximate the contraction of arbitrary tensor networks. Our algorithm builds on the approach outlined in [17], which approximates each intermediate tensor generated during the contraction as an MPS with a bounded rank. When contracting two tensors, the algorithm merges two MPSs, with swaps of adjacent dimensions in the MPS being the bottleneck for complexity.

For a tensor network defined on $G = (V, E)$, we prove that the minimum number of swaps required during contraction is lower bounded by the least number of edge crossings in any vertex linear ordering of the tensor network graph, denoted by $\min_\sigma \mathrm{cr}(G, \sigma)$. A vertex linear ordering $\sigma : V \to \{1, \ldots, |V|\}$ assigns each vertex a unique number, and two edges with adjacent vertex orders $(i, j), (k, l)$ cross if $i < k < j < l$. Hence, we reduce the problem of finding the minimum number of swaps to the problem of finding a vertex linear ordering that minimizes the number of edge crossings. In addition, for a fixed vertex ordering σ^V, the number of swaps used in CATN-GO equals the lower bound, $\mathrm{cr}(G, \sigma^V)$, implying optimality for this metric. Furthermore, CATN-GO includes a dynamic programming algorithm to select the contraction tree under a given vertex ordering. This algorithm aims to minimize the overall computational cost, under the assumption that all MPSs have a uniform rank. The uniform rank assumption makes the problem equivalent to minimizing the total length of the MPSs generated during the contractions and has a time complexity of $O(|V|^3|E|)$. Experimental results demonstrate that when contracting tensor networks defined on 3D lattices using the Ising model, our algorithm is more efficient than the algorithm proposed in [17] in terms of speed, and achieves a 5.9X speed-up while maintaining the same accuracy.

We propose another approximate tensor network contraction method named Partitioned Contract. Like similar methods proposed in [6, 17, 2], our algorithm approximates each intermediate tensor as a binary tree tensor network. Compared to previous works, the proposed algorithm has the flexibility to incorporate a larger portion of the environment when performing low-rank approximations. Here, the environment refers to the remaining set of tensors in the network, and low-rank approximations with larger environments can generally provide higher accuracy. In addition, our proposed algorithm includes a cost-efficient density matrix algorithm [12, 4] for approximating a tensor network with a general graph structure into a tree structure. The computational cost of the density matrix algorithm is asymptotically upper-bounded by that of the standard algorithm that uses canonicalization (the process of orthogonalizing all tensors except one in the tenosr network). Experimental results indicate that the proposed algorithm outperforms both algorithms proposed in [17] and [2] when considering tensor networks defined on lattices using the Ising model. Specifically, our approach achieves a 9.2X speed-up while maintaining the same level of accuracy.

References

[1] J. D. Biamonte, J. Morton, and J. Turner. Tensor network contractions for # SAT. *Journal of Statistical Physics*, 160(5):1389–1404, 2015.

[2] C. T. Chubb. General tensor network decoding of 2D Pauli codes. *arXiv preprint arXiv:2101.04125*, 2021.

[3] C. Damm, M. Holzer, and P. McKenzie. The complexity of tensor calculus. *computational complexity*, 11(1-2):54–89, 2002.

[4] M. Fishman, S. White, and E. Stoudenmire. The ITensor software library for tensor network calculations. *SciPost Physics Codebases*, page 004, 2022.

[5] J. Gray and G. K. Chan. Hyper-optimized compressed contraction of tensor networks with arbitrary geometry. *arXiv preprint arXiv:2206.07044*, 2022.

[6] A. Jermyn. Automatic contraction of unstructured tensor networks. *SciPost Physics*, 8(1):005, 2020.

[7] S. Kourtis, C. Chamon, E. Mucciolo, and A. Ruckenstein. Fast counting with tensor networks. *SciPost Physics*, 7(5):060, 2019.

[8] M. Levin and C. P. Nave. Tensor renormalization group approach to two-dimensional classical lattice models. *Physical review letters*, 99(12):120601, 2007.

[9] M. Lubasch, J. I. Cirac, and M.-C. Banuls. Algorithms for finite projected entangled pair states. *Physical Review B*, 90(6):064425, 2014.

[10] M. Lubasch, J. I. Cirac, and M.-C. Banuls. Unifying projected entangled pair state contractions. *New Journal of Physics*, 16(3):033014, 2014.

[11] L. Ma and C. Yang. Low rank approximation in simulations of quantum algorithms. *Journal of Computational Science*, page 101561, 2022.

[12] Tensornetwork.org contributors. Density matrix algorithm - tensornetwork.org. 2021.

[13] A. Novikov, D. Podoprikhin, A. Osokin, and D. P. Vetrov. Tensorizing neural networks. In *Advances in neural information processing systems*, pages 442–450, 2015.

[14] R. Orús. A practical introduction to tensor networks: Matrix product states and projected entangled pair states. *Annals of Physics*, 349:117–158, 2014.

[15] R. Orús. Tensor networks for complex quantum systems. *Nature Reviews Physics*, 1(9):538–550, 2019.

[16] B. O'Gorman. Parameterization of tensor network contraction. In *14th Conference on the Theory of Quantum Computation, Communication and Cryptography*, 2019.

[17] F. Pan, P. Zhou, S. Li, and P. Zhang. Contracting arbitrary tensor networks: general approximate algorithm and applications in graphical models and quantum circuit simulations. *Physical Review Letters*, 125(6):060503, 2020.

[18] Y. Pang, T. Hao, A. Dugad, Y. Zhou, and E. Solomonik. Efficient 2D tensor network simulation of quantum systems. In *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–14. IEEE, 2020.

[19] U. Schollwöck. The density-matrix renormalization group. *Reviews of modern physics*, 77(1):259, 2005.

[20] Y.-Y. Shi, L.-M. Duan, and G. Vidal. Classical simulation of quantum many-body systems with a tree tensor network. *Physical review A*, 74(2):022320, 2006.

[21] F. Verstraete and J. I. Cirac. Renormalization algorithms for quantum-many body systems in two and higher dimensions. *arXiv preprint cond-mat/0407066*, 2004.

[22] F. Verstraete, V. Murg, and J. I. Cirac. Matrix product states, projected entangled pair states, and variational renormalization group methods for quantum spin systems. *Advances in physics*, 57(2):143–224, 2008.

[23] G. Vidal. Efficient classical simulation of slightly entangled quantum computations. *Physical review letters*, 91(14):147902, 2003.

[24] S. R. White. Density matrix formulation for quantum renormalization groups. *Physical review letters*, 69(19):2863, 1992.

[25] Y. Zhou, E. M. Stoudenmire, and X. Waintal. What limits the simulation of quantum computers? *Physical Review X*, 10(4):041038, 2020.