Challenging Claims of Quantum Supremacy Using Reinforcement Learning

Samuel Garnett [1] Darwin Vargas [1] Yanglet Xiao-Yang Liu [1]

[1]Rensselaer Polytechnic Instituite

Background

Introduction: This research use machine learning to optimize the classical simulation of quantum circuits using a tensor network representation, challenging Google's claim of quantum supremacy with their Sycamore Circuit. We utilize reinforcement learning methods and transformer architecture to train a model capable of optimizing tensor network contraction order (TNCO) problems.

Qubits: Qubits can be in a superposition of states, represented by $|\psi\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle$ where $\alpha_0, \alpha_1 \in \mathbb{C}$ and $|\alpha_0|^2 + |\alpha_1|^2 = 1$. For n qubits in a system, 2^n states are represented..

Quantum Gates: Google's Sycamore Circuit uses the following single-qubit gates (1) and the following double-qubit gate (2):

$$\sqrt{\mathbf{X}} = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & -i \\ -i & 1 \end{bmatrix}, \sqrt{\mathbf{Y}} = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix}, \sqrt{\mathbf{W}} = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & -\sqrt{-i} \\ \sqrt{-i} & 1 \end{bmatrix} \quad (1)$$

$$fSim(\theta,\phi) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\theta) & -i\sin(\theta) & 0 \\ 0 & -i\sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 0 & e^{-i\phi} \end{bmatrix} \quad (2)$$

Sycamore Circuit: A quantum circuit is a series of gates quantum gates. Within Google's Sycamore Circuit, $\mathbf{R} \in \{\sqrt{\mathbf{X}}, \sqrt{\mathbf{Y}}, \sqrt{\mathbf{W}}\}$ is chosen at random. $\mathbf{U}$ represents the double-qubit gate.

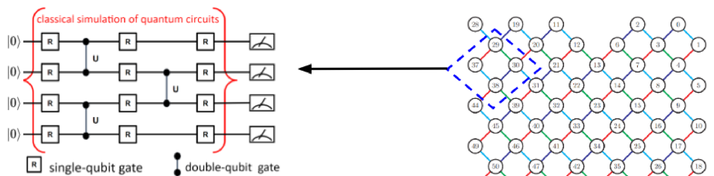

Figure 1. Example Sycamore Circuit

A contracting tensor network can represent the circuit, with 2D tensors representing single-qubit gates and 4D tensors representing double-qubit gates. Nodes denote tensors, and lines denote contractions.

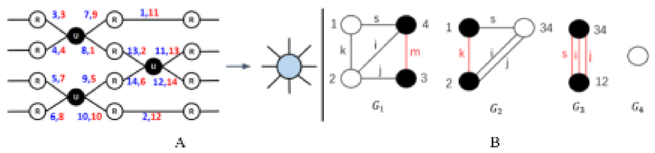

Figure 2. Tensor Network representation of the Sycamore Circuit (a) and contraction visualization (b)

Fig. 2A contains two contraction orders (red and blue) which take 976 and 5056 multiplications respectively.

Methodology

Problem Space: We represent the contraction problem as a two dimensional k-spin ising model

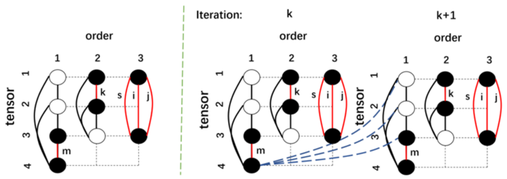

Figure 3. Conventional Ising Model vs K-spin ising model

Methodology (Continued)

Equation five is the cost function used to evaluate the computational cost of each Ising-model configuration:

$$C(x) = \sum_{i=1}^{N-1}\left\{(2 - \sum_{u=1}^{N-i} x_{u,i})^2 + \sum_{u=1}^{N}\sum_{v=1}^{N} J_{u,v}^{i} x_{u,i} x_{v,i}\right\} \quad (5)$$

where $J_{u,v}^{i}$ represents the process of contraction between a pair of tensors.

Policy Network: The policy network is built using a transformer neural network. We represent the tensor network as an undirected graph represented by a symmetric matrix $\mathbf{M}$. E_t represents the set of remaining edges after each contraction. The policy network inputs $\mathbf{M}$ and generates a contraction ordering $P = (e_1 ... e_n - 1)$ where $e_t \in E_t$ is an edge connecting two tensors. This network is then evaluated using the cost function; the model receives positive feedback for a low-cost contraction and negative feedback for a high-cost contraction.

Implementation Methods:

- **Massively Parallel Gym Environment**: We maintain multiple gym environments for algorithm training.
- **Learn To Optimize (L2O)**: We utilize LSTM networks [3] expressed as $f(\nabla L, L) = \nabla'$.
- **Dual Replay Buffers**: We maintain two replay buffers: One for all TNCOs, and one for high-quality TNCOs.
- **Swarm optimization**: We employ swarm optimization to converge on a global optimum [4].
- **Curriculum Learning**: The algorithm sequentially solves more complex TNCO problems [5].

Results

Synthetic Random Tensor Networks: Random tensor networks were generated using the same method as used in "Optimizing tensor network contraction using reinforcement learning" (Eli Meirom et. al). Again the RL-Ising model outperforms all baselines with a speed up of 3.98x.

Table 1: Results on synthetic tensor-train networks.

Tensors	400	600	800	1000	1500	2000
Scale	$\times 10^{120}$	$\times 10^{180}$	$\times 10^{241}$	$\times 10^{301}$	$\times 10^{451}$	$\times 10^{602}$
OE-Greedy [13]	17.22	27.67	4.44	3.83	-	-
CTG-Greedy [18]	10.33	16.60	2.67	4.28	-	-
CTG-Kahypar [18]	10.23	16.60	4.67	4.26	14.12	4.57
RL-Ising	**5.16**	**8.28**	**2.14**	**2.14**	**7.01**	**2.29**

Synthetic Random Tensor Networks: Random tensor networks were generated using the same method as used in "Optimizing tensor network contraction using reinforcement learning" (Eli Meirom et. al). Again the RL-Ising model outperforms all baselines with a speed up of 3.98x. (Table 2)

Table 2: Number of multiplications for synthetic random tensor networks.

Qbits	25	50	75	100
Scale	$\times 10^4$	$\times 10^7$	$\times 10^{10}$	$\times 10^{12}$
OE-Greedy [13]	53.7/27.7	75.4/11.3	104/4.5	5296/26.4
CTG-Greedy [18]	40.3/20.3	12.8/ 4.2	8.3/0.9	27.9/ 2.2
CTG-Kahypar [18]	46.4/24.8	13.4/ 4.3	4.1/0.4	54.2/ 1.2
RL-TNCO [34]	13.1/12.5	3.2/ 1.8	1.2/0.2	5.5/ 1.8
RL-Ising	**12.5/11.4**	**2.5/ 1.5**	**0.7/0.1**	**4.9/ 1.1**

Results (Continued)

Benchmark Performance on Google's Sycamore Circuits: Google's Sycamore circuit has 53 qubits and m cycles. Each cycle has one layer of random single-qubit gates and one layer of two-qubit gates. In different cycles, the two-qubit gates are applied to different pairs of quantum bits. Notably the RL-Ising method bests the RL-TNCO method. (Table 3)

Table 3: Number of multiplications for Google's Sycamore circuits.

Cycles	$m = 12$	$m = 14$	$m = 16$	$m = 18$	$m = 20$
Scale	$\times 10^{10}$	$\times 10^{12}$	$\times 10^{13}$	$\times 10^{16}$	$\times 10^{18}$
OE-Greedy [13]	6.23×10^7	4.77×10^7	7.74×10^{12}	6.21×10^{10}	9.59×10^8
CTG-Greedy [18]	1.16×10^7	1.91×10^7	1.42×10^{10}	3.71×10^7	4.19×10^7
CTG-Kahypar [18]	2.55×10^3	1.41×10^2	1.03×10^4	48.0	6.69
ACQDP [21]	1.09×10^3	71	1.15×10^4	25.8	6.65
RL-TNCO [34]	5.44	7.39	-	-	3.49
RL-Ising	**1.31**	**1.07**	**9.27**	**12.98**	**1.23**

Next Steps

After finding cheaper contraction orders for the simulation of the Sycamore Circuit, we aim to formulate the optimization of the QFT and FFT algorithms as TNCO problems (Figure Four).

The QFT uses single and double qubit gates similar to the Sycamore Circuit and has a simulation complexity of $O(N \log N)$, where $N = 2^n$ for n qubits. The QFT is used to create a frequency domain for a qubit system.

The FFT is an optimized way of creating a Fourier domain of a discrete data set and can be represented as a branching tensor tree network with a current complexity of $O(N \log N)$.

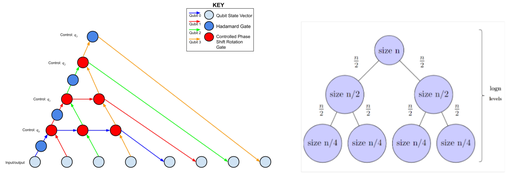

Figure 4. Tree Diagram for Tensor Models

Currently, the RL model is being modified to search through these networks to find optimal TNCOs.

References

1. Liu, Xiao-Yang, and Zeliang Zhang. Classical Simulation of Quantum Circuits Using Reinforcement Learning: Parallel Environments and Benchmark. Proceedings of the 37th Conference on Neural Information Processing Systems (NeurIPS 2023), Track on Datasets and Benchmarks, 2023.
2. Eli Meirom, Haggai Maron, Shie Mannor, and Gal Chechik. Optimizing tensor network contraction using reinforcement learning. In International Conference on Machine Learning, pages 15278–15292. PMLR, 2022.
3. Tianlong Chen, Xiaohan Chen, Wuyang Chen, Zhangyang Wang, Howard Heaton, Jialin Liu, and Wotao Yin. Learning to optimize: A primer and a benchmark. The Journal of Machine Learning Research, 23(1):8562–8620, 2022.
4. Christian Blum and Xiaodong Li. Swarm intelligence in optimization. In Swarm intelligence: introduction and applications, pages 43–85. Springer, 2008.
5. Yoshua Bengio, J´ erˆ ome Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In International Conference on Machine Learning (ICML), pages 41–48, 2009.