Compressing Quantum Circuit Simulation Tensor Data

MILAN SHAH, North Carolina State University, USA
XIAODONG YU (ADVISOR), Argonne National Laboratory, USA
SHENG DI (ADVISOR), Argonne National Laboratory, USA
FRANCK CAPPELLO (ADVISOR), Argonne National Laboratory, USA
MICHELA BECCHI (ADVISOR), North Carolina State University, USA

1 INTRODUCTION

Quantum circuit simulation can be carried out as a contraction over many quantum tensors. QTensor [2], a library built for quantum circuit simulation using a bucket elimination algorithm, contracts tensors to return a final energy value. As bucket elimination advances, tensors can grow large, and memory becomes a bottleneck. To address memory limitations of circuit simulation while enabling more complex circuits to be simulated, we focus on implementing a lossy compressor that can compress the floating-point data stored in quantum circuit tensors while simultaneously preserving a final energy value within an error bound after decompression. We study the effects of various lossy compression/decompression strategies on data compressibility, throughput, and result error to ensure compression/decompression can be effective, fast, and does not heavily distort data. We target GPU as a compression platform due to its massively parallel architecture and optimize for GPU efficiency using CUDA 11 and CUDA-based libraries like NVCOMP and Thrust.

2 BACKGROUND

Tensors represent quantum circuit gates that operate on quantum states, thus they are composed of floating-point complex numbers. A tensor with dimension d has 2^d data points. Tensors, when laid out in a 1-D fashion, can exhibit periodic behavior and sparsity. These attributes of tensors present a challenge to lossy compressors that rely on relatedness between neighboring data points and can make tensors more difficult to compress.

Because of their ability to achieve high compression ratios and high throughput as well as having existing ports to GPU, SZ [1] and SZx [4] are used as two existing lossy compressors that are the basis for extension in this work. Since SZ and SZx are lossy compressors that rely on relatedness of data points, we introduce optimizations and pre/postprocessing steps for these compressors that can further boost compression ratios and throughput. Pre- and postprocessing steps transform incoming tensor data to increase data "smoothness" and thus boost compressibility. Optimizations target the underlying SZ framework to increase throughput.

Authors' addresses: Milan Shah, North Carolina State University, Raleigh, North Carolina, USA, mkshah5@ncsu.edu; Xiaodong Yu (Advisor), Argonne National Laboratory, Lemont, Illinois, USA; Sheng Di (Advisor), Argonne National Laboratory, Lemont, Illinois, USA; Franck Cappello (Advisor), Argonne National Laboratory, Lemont, Illinois, USA; Michela Becchi (Advisor), North Carolina State University, Raleigh, North Carolina, USA.

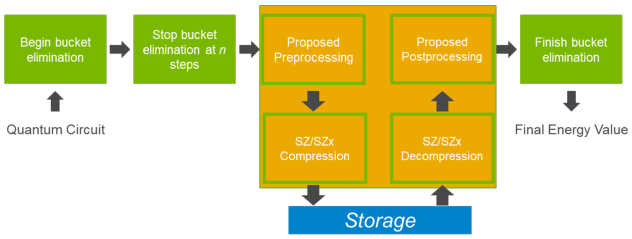

Fig. 1. QTensor data pipeline with compression stages

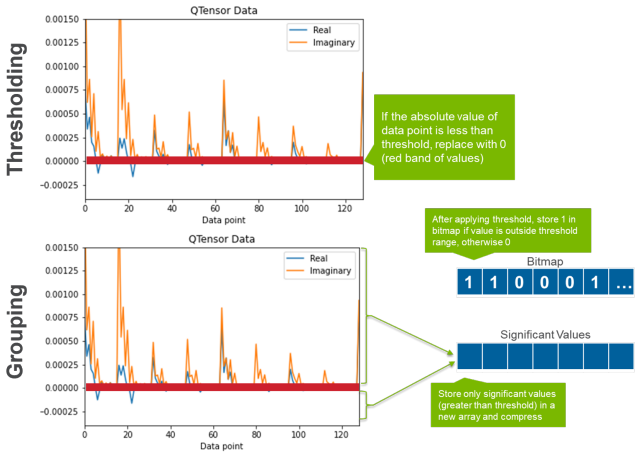

Fig. 2. Thresholding and grouping data transformation strategies

3 DESIGN

Fig. 1 provides an overview of the quantum circuit simulation pipeline. Stages highlighted in orange operate on tensor data and storage can be either in memory or on disk.

3.1 Pre/Postprocessing Steps

We propose two data transformation methods that leverage tensor sparsity: 1. Thresholding and 2. Grouping. The thresholding method entails flushing all data points with absolute value less than some threshold to zero. The grouping method first applies thresholding, then only nonzero values are moved to a new array to be compressed. A bitmap stores the location of zero and nonzero values and is used during postprocessing after decompression to place nonzero and zero values in the correct location of the tensor. The grouping method can directly reduce data size since zero values can be discarded, and both methods increase similarity among data

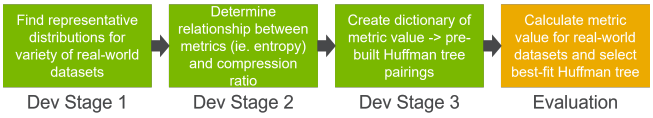

Fig. 3. Design and evaluation stages for pre-built huffman trees

Percent Error of True Energy Value and R2R Threshold/Absolute Error Applied Energy Value									
		Threshold (R2R)							
		0.25	0.1	0.05	0.01	0.005	0.001	0.0005	0.0001
R2R Error	0.25	0.743	0.605	0.268	0.232	0.211	1.206	0.338	0.666
	0.1	0.754	0.511	0.185	0.013	0.219	0.104	0.157	0.168
	0.05	0.751	0.440	0.279	0.073	0.158	0.069	0.094	0.066
	0.01	0.757	0.457	0.256	0.038	0.013	0.023	0.002	0.012
	0.005	0.757	0.457	0.242	0.031	0.015	0.010	0.002	0.011
	0.001	0.756	0.455	0.242	0.033	0.008	0.005	0.002	0.000
	0.0005	0.756	0.455	0.243	0.035	0.009	0.002	0.001	0.001
	0.0001	0.756	0.455	0.243	0.035	0.008	0.001	0.000	0.000

Fig. 4. Effect of applying threshold to tensor data on final energy value from contraction

points. Additionally, there is a low performance overhead with parallelization opportunities that are further explored in this work. Grouping uses a parallel stream compaction algorithm to move data points to the new array and a prefix scan on the bitmap during decompression to assign nonzero values to their original location in the tensor. The bitmap is also compressed during compression using LZ4, a lossless compressor in the CUDA-based NVCOMP library.

3.2 Huffman Optimization

SZ and cuSZ [3], a GPU implementation of SZ, have several stages of compression that advance as follows: 1. Data point prediction from previous data points 2. Quantizing prediction distance from true value 3. Lossless compression on quantization codes from step 2. In stage 3, Huffman encoding is used to carry out lossless compression of quantization codes, a process that can be costly in a parallel environment since Huffman tree building is an irregular operation. We propose a process of selecting a pre-built Huffman tree as opposed to building a custom Huffman tree for each data file being processed. Creating the custom Huffman tree can be a performance bottleneck and thus pre-built trees can increase the throughput of SZ and cuSZ since Huffman encoding is a difficult process to parallelize on GPU. Selection of a Huffman tree depends on the entropy of both the Huffman tree and the distribution of quantization codes for a dataset. The relationship between tree and distribution entropy is additionally studied. In future work, we will explore a range of distributions, but for preliminary experiments we vary entropy through standard deviation changes of the normal distribution.

Fig. 3 outlines the process of creating and testing pre-built Huffman trees.

4 LOSSY COMPRESSION EFFECT ON ENERGY RESULT

A relative-to-range (R2R) value is multiplied with the maximum of a dataset minus the minimum of the dataset to yield either a threshold or absolute error bound. Preliminary results for understanding the effect of threshold on final energy value are in Fig .4. These results show that there exists a range of thresholds for zeroing values that can keep final energy results within a 1% to 5% error range. Lowering the threshold can decrease final energy error more, but a reasonably

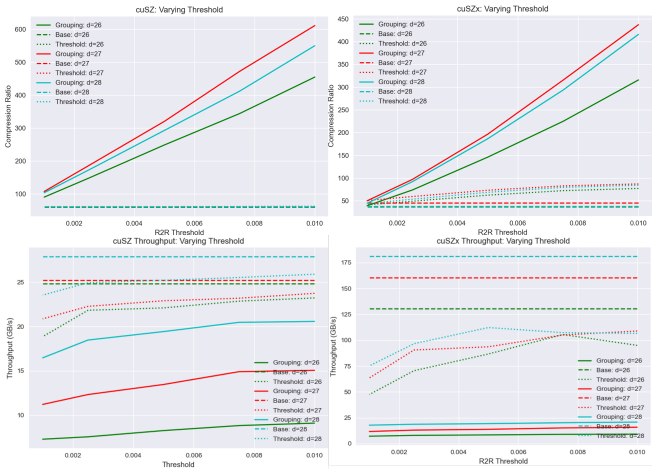

Fig. 5. Results from threshold and grouping methods using cuSZ and cuSZx on a Pascal Titan Xp GPU. d refers to dimension of tensor.

high threshold that targets tensor sparsity can be used to increase compressibility with little effect on energy result.

5 PRELIMINARY RESULTS

5.1 Pre/Postprocessing

Fig. 5 displays compression/decompression times and compression ratios on sample tensor data using cuSZ and cuSZx. For cuSZ and cuSZx, thresholding can increase compression ratio up to two times. Grouping coupled with thresholding can vastly increase compression ratio, as high as 600, compared to an absolute error bound only approach using cuSZ and cuSZx.

5.2 Huffman

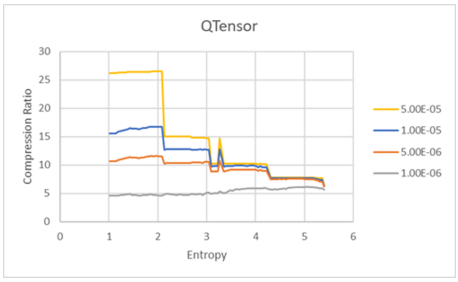

Fig. 6. Plot of varying pre-built Huffman tree with associated entropy on x-axis and compression ratio for QTensor data on y-axis. Each series corresponds to differing absolute error bound.

Fig. 6 indicates the compression ratios of QTensor data using cuSZ and normal distribution Huffman trees with entropy varying from 1 to 5.5. Preliminary results studying the normal distribution as a basis for Huffman trees indicate that the lowest entropy normal distributions create Huffman trees that provide the closest-to-optimal compression ratios for datasets. Initially, we hypothesized that the closest entropy tree would yield the best compression ratio for datasets, but results indicate that for quantization code distributions, lower entropy performs better.

ACKNOWLEDGMENTS

This material was supported by the U.S. Department of Energy, Office of Science, Advanced Scientific Computing Research (ASCR), under contract DE-AC02-06CH11357, and supported by the National Science Foundation under Grant OAC-2003709, OAC-2104023. We acknowledge the computing resources provided on Bebop (operated by Laboratory Computing Resource Center at Argonne).

REFERENCES

[1] Sheng Di and Franck Cappello. 2016. Fast Error-Bounded Lossy HPC Data Compression with SZ. In *2016 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*. 730–739. https://doi.org/10.1109/IPDPS.2016.11

[2] Roman Schutski, Danil Lykov, and Ivan Oseledets. 2020. Adaptive algorithm for quantum circuit simulation. *Phys. Rev. A* 101 (Apr 2020), 042335. Issue 4. https://doi.org/10.1103/PhysRevA.101.042335

[3] Jiannan Tian, Sheng Di, Kai Zhao, Cody Rivera, Megan Hickman Fulp, Robert Underwood, Sian Jin, Xin Liang, Jon Calhoun, Dingwen Tao, and Franck Cappello. 2020. cuSZ: An Efficient GPU-Based Error-Bounded Lossy Compression Framework for Scientific Data. In *Proceedings of the ACM International Conference on Parallel Architectures and Compilation Techniques* (Virtual Event, GA, USA) *(PACT '20)*. Association for Computing Machinery, New York, NY, USA, 3–15. https://doi.org/10.1145/3410463.3414624

[4] Xiaodong Yu, Sheng Di, Kai Zhao, Jiannan Tian, Dingwen Tao, Xin Liang, and Franck Cappello. 2022. Ultrafast Error-Bounded Lossy Compression for Scientific Datasets. In *Proceedings of the 31st International Symposium on High-Performance Parallel and Distributed Computing* (Minneapolis, MN, USA) *(HPDC '22)*. Association for Computing Machinery, New York, NY, USA, 159–171. https://doi.org/10.1145/3502181.3531473