QXTools: A Julia framework for distributed quantum circuit simulation

John Brennan[1], Lee O'Riordan[1], Kenneth Hanley[1], Myles Doyle[1], Momme Allalen[2], David Brayford[2], Luigi Iapichino[2], and Niall Moran*[1]

1 Irish Centre for High-End Computing, Ireland **2** Leibniz Supercomputing Centre

Summary

QXTools is a framework for simulating quantum circuits using tensor network methods. Weak simulation is the primary use case where given a quantum circuit and input state QXTools will efficiently calculate the probability amplitude of a given output configuration or set of configurations. Given this ability one can sample from the output distribution using random sampling approaches. QXTools is intended to be used by researchers interested in simulating circuits larger than those possible with full wave-function simulators or those interested in research and development of tensor network circuit simulation methods. See (Brennan et al., 2021) for more complete background and scaling results and (Brayford et al., 2021) for details about deploying in containerised environments.

QXTools is written in Julia (Bezanson et al., 2017) and is designed to run on large distributed compute clusters and to support GPU accelerators. The simulation workflow is broken down into a number of stages, each of which is managed by a special purpose package which can be used independently or as part of the QXTools framework. To find efficient contraction orders for tensor networks, an algorithm called FlowCutter (Hamann & Strasser, 2018) is used to construct tree decompositions with optimal treewidth of the network's line graph by iteratively partitioning it using maximal flows on the graph. A domain specific language (DSL) is used to express the simulation as a set of tensor network operations. This separates the high level index accounting and contraction planning from the low level implementation of the tensor network operations and makes it easier to support new hardware and network architectures.

Statement of need

As quantum processing devices continue to scale and the algorithms and experiments being run on them grow in complexity, simulations of these systems become much more computationally demanding. To reduce the turnaround time and allow larger systems to be simulated it is necessary to move beyond single workstations and use distributed compute clusters. QXTools provides a flexible, extensible open source framework for performing these simulations. The use of Julia (Bezanson et al., 2017) makes it easy for the code to be understood, modified and extended while not sacrificing performance compared to compiled languages.

Background

Classical simulation of quantum circuits is essential for debugging and validating the accuracy of quantum computing devices and algorithms. This is a very computationally demanding

---

*corresponding author

problem owing to the exponential growth in the state space as the number of qubits is increased. Up until recently it has been possible to simulate the largest prototype universal quantum computers using direct evolution of the quantum state (where the full wave-function is stored in memory (or on disk)) using modest personal computing resources. With the recent emergence of Noisy Intermediate Scale Quantum (NISQ) devices, it has become intractable to use this approach to simulate devices of this size on even the largest supercomputers. These advances have necessitated the development of new circuit simulation methods which can simulate large systems without requiring the memory to store the full wave-function. Tensor network methods have been demonstrated to achieve state of the art performance in simulating Random Quantum Circuits (RQC) as part of the quantum supremacy experiments (Villalonga et al., 2019). Despite the impressive results achieved with these methods to date they are not suitable for all types of circuits and in many cases full wave-function methods are preferable. For example for highly entangled circuits, the tensor network representation will consume the same memory as full wave-function methods but will incur additional overhead. However for circuits with moderate entanglement and cases where one is not interested in exact results, but in results up to a particulary fidelity, tensor network approaches can offer significant advantages.

Tensor networks refer to networks of interconnected tensors, the use of which originated in the condensed matter physics and quantum information communities as a means of simulating strongly correlated many body quantum systems. Expressing a quantum circuit as a tensor network is very straightforward and involves replacing each gate with a tensor and registers with sets of interconnected tensors. Single qubit Hadamard gates remain unchanged from their matrix representation, while two qubit gates can be expressed as a single rank 4 tensor or two connected rank 3 tensors. Once the quantum circuit is expressed as a network of interconnected tensors operations can be performed by contracting tensors together. For further details there are many excellent resources on tensor networks and their use in quantum information, see (Biamonte & Bergholm, 2017), (Bridgeman & Chubb, 2017), (Wood et al., 2011).

This is a very active area of research with many packages available offering quantum circuit simulation capabilities using tensor network methods, each with different capabilities. These include the quimb package (Gray, 2018), ExaTN (McCaskey et al., 2019), Koala (Pang et al., 2020), PastaQ (Torlai & Fishman, 2020) and Jet (Vincent et al., 2021). Some of the more distinctive features of QXTools is the use of the Flowcutter algorithm for contraction path finding, the focus on distributed simulation, the use of Julia and the modular design.

Functionality and design

QXTools consists of a number of Julia packages available under the ("JuliaQX," 2021) organization and registered in the Julia package registry. The QXTools.jl package ties these together to enable circuit simulation workflows. The individual packages and their roles are as follows:

- QXTns.jl: Provides data structures representing tensor networks and tensor network circuits along with functionality for contracting these and keeping track of tensor indices and hyper-indices.
- QXGraphDecompositions.jl: Provides specialised graph algorithms for optimizing tensor network calculations and finding edges to slice to decompose computations. Here, FlowCutter (Hamann & Strasser, 2018) is used to find good contraction orderings for a network.
- QXContexts.jl: Provides computation tree data structures to represent computations and the ability to execute these compute graphs on different hardware platforms.
- QXZoo.jl: Quantum circuit representations and manipulation functionality.
- YaoQX.jl: Enables QXTools to be used as a backend for Yao.jl (Luo et al., 2020).

Acknowledgements

References

Bezanson, J., Edelman, A., Karpinski, S., & Shah, V. (2017). Julia: A Fresh Approach to Numerical Computing. *SIAM Review*, *59*(1), 65–98. https://doi.org/10.1137/141000671

Biamonte, J., & Bergholm, V. (2017). *Tensor networks in a nutshell*. http://arxiv.org/abs/1708.00006

Brayford, D., Brennan, J., Allalen, M., Hanley, K., Iapichino, L., ORiordan, L., & Moran, N. (2021). Deploying Containerized QuantEx Quantum Simulation Software on HPC Systems. *arXiv:2110.05162 [cs]*. http://arxiv.org/abs/2110.05162

Brennan, J., Allalen, M., Brayford, D., Hanley, K., Iapichino, L., O'Riordan, L. J., Doyle, M., & Moran, N. (2021). Tensor Network Circuit Simulation at Exascale. *arXiv:2110.09894 [quant-Ph]*. http://arxiv.org/abs/2110.09894

Bridgeman, J. C., & Chubb, C. T. (2017). Hand-waving and interpretive dance: An introductory course on tensor networks. *Journal of Physics A: Mathematical and Theoretical*, *50*(22), 223001. https://doi.org/10.1088/1751-8121/aa6dc3

Gray, J. (2018). Quimb: A python package for quantum information and many-body calculations. *Journal of Open Source Software*, *3*(29), 819. https://doi.org/10.21105/joss.00819

Hamann, M., & Strasser, B. (2018). *Graph bisection with pareto-optimization*. 1–34. https://doi.org/10.1145/3173045

JuliaQX. (2021). In *GitHub organization*. GitHub. https://github.com/JuliaQX

Luo, X.-Z., Liu, J.-G., Zhang, P., & Wang, L. (2020). Yao.jl: Extensible, Efficient Framework for Quantum Algorithm Design. *Quantum*, *4*, 341. https://doi.org/10.22331/q-2020-10-11-341

McCaskey, A., Dumitrescu, E., Liakh, D., Alvarez, G., & Mintz, T. (2019). *ExaTN - A Scalable Exascale Math Library for Hierarchical Tensor Network Representations and Simulations*. *2019*, B22.012. https://ui.adsabs.harvard.edu/abs/2019APS..MARB22012M

Pang, Y., Hao, T., Dugad, A., Zhou, Y., & Solomonik, E. (2020). Efficient 2D Tensor Network Simulation of Quantum Systems. *arXiv:2006.15234 [physics, Physics:quant-Ph]*. http://arxiv.org/abs/2006.15234

Torlai, G., & Fishman, M. (2020). *PastaQ: A package for simulation, tomography and analysis of quantum computers*. https://github.com/GTorlai/PastaQ.jl/

Villalonga, B., Lyakh, D., Boixo, S., Neven, H., Humble, T. S., Biswas, R., Rieffel, E. G., Ho, A., & Mandrà, S. (2019). *Establishing the Quantum Supremacy Frontier with a 281 Pflop/s Simulation*. 1–11. http://arxiv.org/abs/1905.00444

Vincent, T., O'Riordan, L. J., Andrenkov, M., Brown, J., Killoran, N., Qi, H., & Dhand, I. (2021). Jet: Fast quantum circuit simulations with parallel task-based tensor-network contraction. *arXiv:2107.09793 [quant-Ph]*. http://arxiv.org/abs/2107.09793

Wood, C. J., Biamonte, J. D., & Cory, D. G. (2011). *Tensor networks and graphical calculus for open quantum systems*. http://arxiv.org/abs/1111.6950