The Journal of Open Source Software

HDH:The Distributed Quantum Computing library

Maria Gragera Garces [1*]

1 University of Edinburgh * These authors contributed equally.

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor:

Submitted: 17 December 2025
Published: unpublished

Summary

Quantum computing aims to solve computational problems that are classically hard. To achieve this in utility settings, quantum computers will require thousands if not millions of qubits. Current devices hold hundreds of qubits at most. It is believed that the path towards these scales will come from distribution, meaning the collaboration of various devices to complete tasks larger than their individual capacities. The main goal behind Distributed Quantum Computing (DQC) is to allocate sub-partitions of large quantum computations across multiple devices smaller than the computation itself. Existing approaches abstract computations to hypergraphs which are then partitioned, but they lack a unified framework for comparing and developing partitioning strategies within and across computational models.

Statement of need

`HDH` is a Python package designed for researchers to test and develop partitioning strategies for quantum workloads.

HDHs (Hybrid Dependency Hypergraphs) are an abstraction which transforms quantum computation, originating from any quantum computational model (including circuits, measurement-based quantum computing, quantum cellular automata, and quantum walks), to a directed hypergraph that expresses all possible partitions available within the computation. They were originally proposed in (Gragera Garces et al., 2025) as a unifying approach to quantum distribution, extending the hypergraph abstraction method for partitioning across devices originally proposed in (Andrés-Martínez & Heunen, 2019). Since then, various partitioning strategies have been proposed (Clark et al., 2023; Escofet et al., 2023; Sundaram & Gupta, 2023), but many are tested on inconsistent hypergraph abstractions, hindering cross-partitioner comparison and improvement.

Having an easy to implement, open-source, and model-agnostic abstraction will enable the fair and consistent cross-comparison of partitioning strategies in future work. Furthermore, HDHs extend this capability beyond the circuit model, addressing a current blind spot in DQC research.

`HDH` is designed to be used by both distributed quantum architecture researchers and compiler developers. No other libraries are dedicated to the specific advancement of partitioning heuristics based on directed hypergraph abstraction. While quantum compilation frameworks like Qiskit (Javadi-Abhari, A., Treinish, M., Krsulich, K., Wood, C. J., Lishman, J., Gacon, J., Martiel, S., Nation, P., Bishop, L. S., Cross, A. W., Johnson, B. R., & Gambetta, J. M., 2024), Cirq (Developers, 2025), and PennyLane (Bergholm et al., 2018) provide circuit optimization and device mapping, they do not offer model-agnostic abstractions for distributed quantum computing. The `HDH` library is compatible with these SDKs, making it a seamless addition to state of the art quantum software stacks.

Model conversions

Any quantum computing model comprises a series of commands which establish qubit state rotations, measurements and entanglements. For instance, quantum circuits are comprised of a sequence of quantum gates applied to qubits. Single-qubit gates perform rotations on the Bloch sphere, while multi-qubit gates (such as CNOT) create entanglement dependencies between qubits.

HDHs use the following notation to describe quantum workload dependencies, including predicted elements that represent potential future state transformations based on classical measurement outcomes:

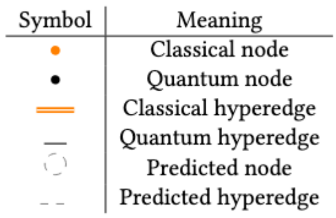

Figure 1: HDH symbol legend.

Mapping a quantum workload such as a circuit to an HDH involves applying specific correspondences between model elements and hypergraph motifs. This library provides model-specific classes such as the `Circuit` class that enable straightforward conversions to HDHs using mapping tables:

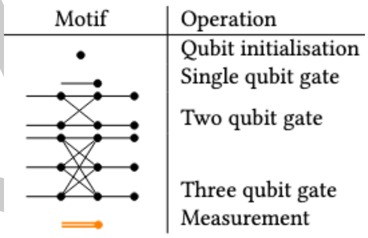

Figure 2: Circuit to HDH mapping table.

In the context of DQC, entangling operations in a model can be made non-local (namely non-local gates) and thus partitioned through a quantum network via quantum communication primitives (Anbang, W. and Hezi, Z. and Gushu, L and Alireza, S and Yuan, X and Yufei, D, 2022). Alternatively, qubit states can be individually forwarded through teleportation protocols (Zomorodi-Moghadam et al., 2017). HDHs aim to showcase all possible partitionings, thus enabling heuristic partitioners to exploit recurring patterns when mapping workloads to quantum or hybrid networks, thereby minimizing communication and other costs.

The table below shows how HDHs supersede previous abstractions in their expressivity of these partitioning options. Unlike prior approaches that represent only non-local gates or only teleportation, HDHs capture both strategies simultaneously, enabling partitioners to optimize across all available distribution methods:

The Journal of Open Source Software

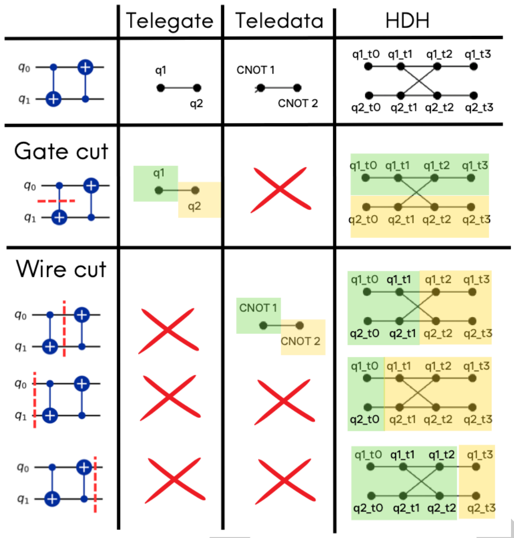

Figure 3: Table showing HDH expressivity.

The library provides model-specific classes such as the `Circuit` class to enable workload to HDH translation:

```
import hdh
from hdh.models.circuit import Circuit
from hdh.visualize import plot_hdh

circuit = Circuit()

# Set of instructions
circuit.add_instruction("ccx", [0, 1, 2])
circuit.add_instruction("h", [3])
circuit.add_instruction("h", [5])
circuit.add_instruction("cx", [3, 4])
circuit.add_instruction("cx", [2, 1])

circuit.add_conditional_gate(5, 4, "z")

circuit.add_instruction("cx", [0, 3])
circuit.add_instruction("measure", [2])
circuit.add_instruction("measure", [4])

hdh = circuit.build_hdh()  # Generate HDH
fig = plot_hdh(hdh)  # Visualize HDH
```

The resulting HDH is shown below as a graph representation of a hypergraph, since visualizing large, multi-colored hypergraphs directly becomes impractical at scale. Gates have hyperedges corresponding to the qubit state transformations they generate, as well as preceding and following hyperedges that capture pre- and post-teleportation of the involved states. HDHs differ from previous abstractions in two key ways:

(1) nodes represent possible state transformations rather than individual qubits or operations, and

(2) classical data flows are explicitly included (shown in orange):

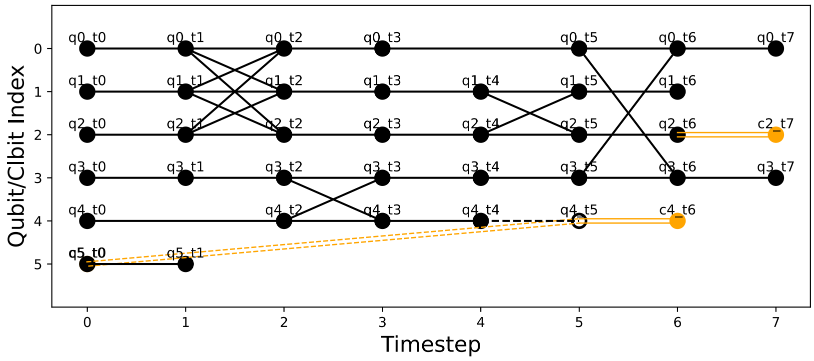

Figure 4: Example circuit and its HDH representation.

Acknowledgements

We acknowledge contributions from Joseph Tedds, Manuel Alejandro, and Alessandro Cosentino.

We thank Unitary Fund for supporting this project through their quantum microgrant program.

The work of the author is supported by the EPSRC UK Quantum Technologies Programme under grant EP/T001062/1 and VeriQloud.

References

Anbang, W. and Hezi, Z. and Gushu, L and Alireza, S and Yuan, X and Yufei, D. (2022). AutoComm: A Framework for Enabling Efficient Communication in Distributed Quantum Programs. *55th IEEE/ACM International Symposium on Microarchitecture*. https://ieeexplore.ieee.org/abstract/document/9923799?casa_token=1tvAGu_y3NIAAAAA:26TLU9SEhr9YWo1oTLUK9dQIVcH4OUQQ1DPdk1ZQNGU76kjqIQmZ6xvycLJI9ltNIdPbjCBafw

Andrés-Martínez, P., & Heunen, C. (2019). Automated distribution of quantum circuits via hypergraph partitioning. *Physical Review A*. https://journals.aps.org/pra/abstract/10.1103/PhysRevA.100.032308

Bergholm, V., Izaac, J., Schuld, M., Gogolin, C., Ahmed, S., Ajith, V., Alam, M. S., Alonso-Linaje, G., AkashNarayanan, B., Asadi, A., & others. (2018). PennyLane: Automatic differentiation of hybrid quantum-classical computations. *arXiv Preprint arXiv:1811.04968*. https://arxiv.org/abs/1811.04968

Clark, J., Humble, T., & Thapliyal, H. (2023). TDAG: Tree-based Directed Acyclic Graph Partitioning for Quantum Circuits. *Proceedings of the Great Lakes Symposium on VLSI*. https://dl.acm.org/doi/10.1145/3583781.3590234

Developers, C. (2025). *Cirq*. Zenodo. https://doi.org/10.5281/ZENODO.4062499

Escofet, P., Ovide, A., Almudever, C. G., Alarcón, E., & Abadal, S. (2023). Hungarian Qubit Assignment for Optimized Mapping of Quantum Circuits on Multi-Core Architectures. *IEEE COMPUTER ARCHITECTURE LETTERS*. https://ieeexplore.ieee.org/abstract/document/10262036?casa_token=h_0XHyNKYYgAAAAA:CK2iz7IfvOP8pQAYzCkf0muy8e_sYFz2QI1AC517ngFIPgB0_4OEsxnMY5JObDYgT-I3KQFIPA

Gragera Garces, M., Heunen, C., & Marina, M. K. (2025). Distributed Quantum Computing Across Heterogeneous Hardware with Hybrid Dependency Hypergraphs. *Proceedings of the ACM SIGCOMM 2025 Posters and Demos*. http://adsabs.harvard.edu/abs/2017arXiv170304627P

Javadi-Abhari, A., Treinish, M., Krsulich, K., Wood, C. J., Lishman, J., Gacon, J., Martiel, S., Nation, P., Bishop, L. S., Cross, A. W., Johnson, B. R., & Gambetta, J. M. (2024). Quantum computing with Qiskit. *ArXiv e-Prints*. arxiv.org/abs/2405.08810

Sundaram, R. G., & Gupta, H. (2023). Distributing Quantum Circuits Using Teleportations. *IEEE International Conference on Quantum Software (QSW)*. https://www.computer.org/csdl/proceedings-article/qsw/2023/047900a186/1Q5oLz6DIZ2

Zomorodi-Moghadam, M., Houshmand, M., & Houshmand, M. (2017). Optimizing Teleportation Cost in Distributed Quantum Circuits. *International Journal of Theoretical Physics*. https://link.springer.com/article/10.1007/s10773-017-3618-x