POSTER: Distributed Quantum Computing Across Heterogeneous Hardware with Hybrid Dependency Hypergraphs

Maria Gragera Garces
The University of Edinburgh
United Kingdom
m.gragera.garces@ed.ac.uk

Chris Heunen
The University of Edinburgh
United Kingdom
chris.heunen@ed.ac.uk

Mahesh Marina
The University of Edinburgh
United Kingdom
mahesh@ed.ac.uk

Abstract

Distributing quantum computations across heterogeneous devices introduces communication and coordination costs that depend on both quantum and classical operations. We present an architecture-aware approach using Hybrid Dependency Hypergraphs (HDHs) to model space, time, and type dependencies in distributed execution. This poster explores how HDHs support network-level reasoning about communication patterns, and how they can expose cost trade-offs across circuit-based, measurement-based, and quantum walk models. We highlight patterns in HDH structure that inform scheduling, device assignment, and cut placement in heterogeneous quantum clusters.

CCS Concepts

• **Computing methodologies → Distributed computing methodologies**; • **Theory of computation → Quantum computation theory**; • **Software and its engineering** → *Compilers.*

Keywords

Distributed quantum computing, Quantum compilation, Hybrid Dependency Hypergraphs, Model-agnostic representation

ACM Reference Format:
Maria Gragera Garces, Chris Heunen, and Mahesh Marina . 2025. POSTER: Distributed Quantum Computing Across Heterogeneous Hardware with Hybrid Dependency Hypergraphs. In *ACM SIGCOMM 2025 Conference (SIGCOMM Posters and Demos '25), September 8–11, 2025, Coimbra, Portugal.* ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/3744969.3748417

1 Introduction

Scaling quantum computations to practical problems will require distributing workloads across multiple processors. Today's quantum hardware landscape is already heterogeneous, spanning technologies like superconducting, ion trap, and neutral atom devices. To fully leverage this diversity, we must enable quantum workloads to run across such systems in coordinated, distributed architectures.

Current quantum distribution approaches abstract quantum circuits to hypergraphs for partitioning across devices [1]. Several strategies exist to effectively partition these hypergraphs [4, 5, 11]. However, these methods are restricted to the quantum circuit model and exclude classical control or alternative quantum computing paradigms such as measurement-based quantum computing (MBQC) [10], quantum cellular automata (QCA) [6] or quantum walks (QW) [9]. This narrow focus limits deployment, as different platforms support different computational models and hybrid workflows are central to many protocols. Notably, photonic quantum computers, which natively interface with quantum networks, do not operate within the circuit paradigm [2].

We recently introduced Hybrid Dependency Hypergraphs (HDHs) [7]: a model-agnostic formalism that represents any quantum computation, including classical control, as a typed hypergraph. HDHs capture space, time, and type dependencies, enabling minimum cost, meaning communication, distributions across heterogeneous architectures.

This poster presents HDHs as a foundation for unified, architecture-aware distribution. We present how quantum workloads induce HDHs, and how emerging structural patterns in HDHs could inform efficient distributed implementations.

2 Hybrid Dependency Hypergraphs

HDHs offer a unifying abstraction for distributing quantum computations across different computational models. As a typed hypergraph representation of quantum workloads, they capture both classical and quantum operations within a single structure that mirrors the computational flow:

- **Nodes**: represent intermediate computational states (quantum or classical).
- **Hyperedges**: represent operations (quantum or classical) acting on these states.
- **Time structure**: is encoded as a partial order over operations, reflecting execution dependencies.

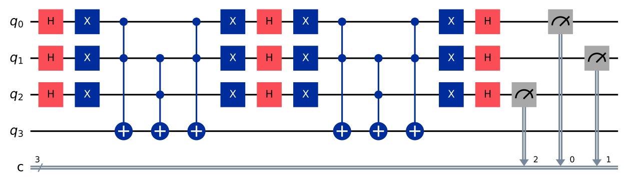
The following quantum computation models have defined mappings to the HDH representation:

Model	Node Types	Edge Types	Time
Circuit	States (q), Meas. (c)	Gates, Meas.	Gate order
MBQC	States (q), Meas. (c)	NEMC Ops.	gflow
QW	Positions (q), Meas. (c)	Shift/Coin	Stepwise
QCA	Zones (q), Updates (c)	Evolution deps.	Stepwise

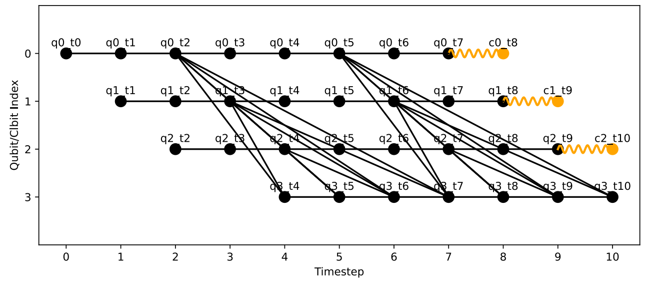
Figure 1, illustrates an HDH translation through a simplified version of a Grover's algorithm implementation. The visualizations were generated using our hdh Python package, developed for this work [1]. The package provides tools to construct HDHs from model specifications, QASM files [3], and Qiskit circuits [8]. It also performs evaluations and partitioning based on the properties of HDHs described below.

[1]Available at: https://github.com/grageragarces/HDH; install via `pip install hdh`.

ACM ISBN 979-8-4007-2026-0/2025/09
https://doi.org/10.1145/3744969.3748417



(a) Qiskit circuit



(b) HDH (4-qubit)

Figure 1: HDH construction from Grover's algorithm. (a) 4-qubit circuit generated for Grover. (b) Corresponding HDH with classical (orange) and quantum (black) node types.

3 Cost Functions for Distribution

A HDH partition corresponds to a distribution of the quantum computation over multiple devices, each represented by a group in the partition. To evaluate such partitions, we define a suite of cost functions that quantify communication overhead, concurrency, and physical feasibility. These cost functions generalize existing circuit-distribution metrics to a model-agnostic setting and enable principled selection of distribution strategies.

3.1 Communication Cost

Every cut in an HDH implies communication across device boundaries. The total communication cost depends on both the number and type of cut hyperedges. In an HDH, partitioning separates nodes into groups assigned to distinct devices. When a quantum edge connects nodes in different partitions, this indicates a requirement for a quantum channel between the devices, or the simulation of one, to support entanglement propagation. Similarly, a classical edge crossing a partition requires a classical communication channel for transmitting measurement outcomes.

We define communication cost between a left L and a right R partition as:

$$\text{CommCost}(L, R) = |\{\tau(e) \mid a \in L, b \in R, \{a, b\} \in C\}| \qquad (1)$$

where $\tau(e) \in \{q, c\}$ tracks the type of the connection and C is the channel connecting the node pair a, b.

A telegate 2-way partition of the HDH generated from the Grover implementation in Figure 1 that asigns the top two qubits to a device and the bottom two to another, induces a cut cost of six. Lower cost partitionings permited by the native support of both telegate and teledata cuts can be available if one assumes availble higher QPU capacity.

3.2 Parallelism and Load Balancing

HDHs make explicit the partial ordering of operations, exposing opportunities for concurrent execution across devices. We define *parallelism* as the number of operations executable per timestep across partitions. Let O be the operations, T the timesteps, and $P(p)$ the nodes in partition $p \in 1, \ldots, k$. Each $o \in O$ has support $\iota(o) \subseteq V$ and timestep $\rho(o) \in T$. For each timestep, we count operations that run in parallel across partitions, subtracting one per partition to offset serial execution.

$$\text{Parallelism} = \sum_{t \in T} \left(\sum_{p=1}^{k} |\{o \in O \mid \iota(o) \subseteq P(p), \rho(o) = t\}| - 1 \right) \quad (2)$$

This expression captures the excess of concurrent operations beyond sequential execution, aggregated over all timesteps.

In the 2-way partitioned Grover implementation example total parallelism of the proposed cut is of 11.

Timestep	t0	t1	t2	t3	t4	t5	t6	t7	t8	t9	t10
Parallelism	0	0	1	1	2	2	2	2	1	0	0

Table 1: Parallelism contributions per time step of the HDH telegate 2-way cut described, generated from the Grover implementation in Figure 1.

3.3 Type-Conscious Architecture Mapping

Cost functions are not sufficient; partition feasibility must also respect physical hardware constraints. We define a *Device Architecture Graph* (DAG) that encodes the connectivity and capacity of each device. An HDH cut is feasible in a DAG if every inter-partition hyperedge maps to available channels in the architecture, and node assignments do not exceed device capacities. This ensures HDHs are not only theoretically distributable but also physically realizable.

Futhermore, quantum and classical connections are not always equally available. Typed HDHs enable *type-aware contractions*, allowing for resource substitution. For example, a quantum edge can be replaced by classical edges with additional processing, reflecting tradeoffs already common in NISQ-era hardware. This enriches the space of viable distributions and highlights HDHs as a tool for resource-aware compilation.

3.4 Pattern-Aware Partitioning

Since our initial design of HDHs, we have observed that *recurring structural motifs* emerge consistently across HDHs. For instance, as seen in Figure 1, Grover's algorithm gives rise to a characteristic M-shaped dependency pattern that becomes more pronounced as the problem size grows. These patterns reflect reusable computational structures and could guide partitioning strategies that go beyond generic hypergraph cuts. By identifying and leveraging regions of regularity, we could enable pattern-aware, cost-driven distribution informed by the semantics of the underlying computation.

Acknowledgments

This work was supported by the EPSRC UK Quantum Technologies Programme under grant EP/T001062/1 and VeriQloud.

References

[1] Pablo Andrés-Martínez and Chris Heunen. 2019. Automated Distribution of Quantum Circuits via Hypergraph Partitioning. *Physical Review A* 100, 3 (Sept. 2019), 032308. doi:10.1103/PhysRevA.100.032308 arXiv:1811.10972 [quant-ph]

[2] Sara Bartolucci, Patrick Birchall, Hector Bombin, Hugo Cable, Chris Dawson, Mercedes Gimeno-Segovia, Eric Johnston, Konrad Kieling, Naomi Nickerson, Mihir Pant, et al. 2023. Fusion-based quantum computation. *Nature Communications* 14, 1 (2023), 912.

[3] Lev S Bishop. 2017. Qasm 2.0: A quantum circuit intermediate representation. 2017 (2017), P46–008.

[4] Joseph Clark, Travis Humble, and Himanshu Thapliyal. 2023. TDAG: Tree-based Directed Acyclic Graph Partitioning for Quantum Circuits. In *Proceedings of the Great Lakes Symposium on VLSI 2023 (GLSVLSI '23)*. Association for Computing Machinery, New York, NY, USA, 587–592. doi:10.1145/3583781.3590234

[5] Pau Escofet, Anabel Ovide, Carmen G Almudever, Eduard Alarcón, and Sergi Abadal. 2023. Hungarian qubit assignment for optimized mapping of quantum circuits on multi-core architectures. *IEEE Computer Architecture Letters* 22, 2 (2023), 161–164.

[6] Terry Farrelly. 2020. A review of quantum cellular automata. *Quantum* 4 (2020), 368.

[7] Maria Gragera Garces, Chris Heunen, and Mahesh Marina. 2025. Towards Model Agnostic Distribution of Quantum Computation with Hybrid Dependency Hypergraphs. (2025). Manuscript under review.

[8] Ali Javadi-Abhari, Matthew Treinish, Kevin Krsulich, Christopher J Wood, Jake Lishman, Julien Gacon, Simon Martiel, Paul D Nation, Lev S Bishop, Andrew W Cross, et al. 2024. Quantum computing with Qiskit. *arXiv preprint arXiv:2405.08810* (2024).

[9] Xiaogang Qiang, Shixin Ma, and Haijing Song. 2024. Quantum Walk Computing: Theory, Implementation, and Application. *Intelligent Computing* 3 (2024), 0097.

[10] Robert Raussendorf and Hans J Briegel. 2001. A one-way quantum computer. *Physical review letters* 86, 22 (2001), 5188.

[11] Ranjani G Sundaram and Himanshu Gupta. 2023. Distributing quantum circuits using teleportations. In *2023 IEEE International Conference on Quantum Software (QSW)*. IEEE, 186–192.