Qdislib: Implementation and Evaluation of Circuit Cutting and Sampling Techniques for Quantum Circuits

Mar Tejedor, Rosa M. Badia - Barcelona Supercomputing Center (BSC)

Motivation

- Near-term devices are **limited by qubit capacity**. **Circuit cutting** splits large circuits into **smaller fragments** that run **independently** and are classically reconstructed.
- **No cutting strategy is optimal for every workload**: each trades off cost, hardware needs and accuracy.
- We **evaluate the four techniques in Qdislib** and derive **practical selection guidelines**.

What is Qdislib, and why is it useful?

Qdislib is an **open-source framework** for **scalable hybrid quantum–classical execution**: four **circuit-cutting and sampling techniques**, automatic cut discovery, fragment caching and **PyCOMPSs-based distributed execution** on **Qiskit, Qibo, CUDA-Q, PennyLane**, running on CPU/GPU simulators and real **QPUs**.

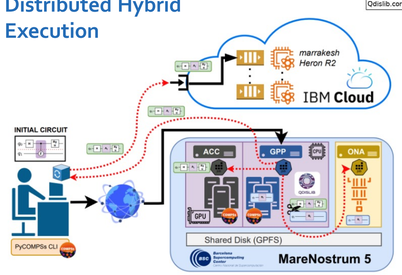
Wire Cutting

```python
# 1. Identify gates to cut
cut = qd.find_cut(circuit, gate_cut=False, wire_cut=True)
# 2. Generate subcircuits
subcircuits = qd.wire_cutting_subcircuits(circuit, cut, "qiskit")
# 3. Execute fragments
values = [qd.expectation_value(s) for s in subcircuits]
# 4. Reconstruct expectation value
result = subcircuits.reconstruct(values)
```

***Qdislib API in four steps**: find the cut, generate the subcircuits, execute them in parallel and reconstruct the expectation value.*

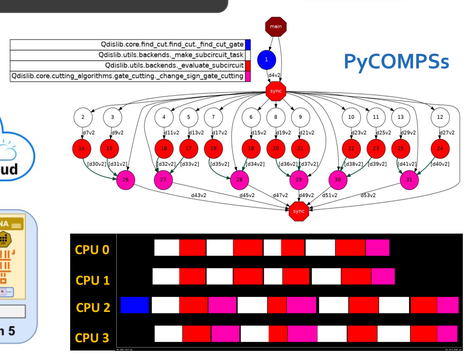
Try it live!

Distributed Hybrid Execution



*From the **PyCOMPSs CLI** on a laptop, the workflow runs on **MareNostrum 5**: Qdislib cuts the circuit on the **GPP** partition and dispatches subcircuits to **GPUs (ACC)**, the **Ona** quantum computer and **IBM Heron R2** in the cloud, sharing data through GPFS.*

PyCOMPSs



***PyCOMPSs** turns every subcircuit into an **independent task** (graph, top). The **Paraver trace** (bottom) shows them running **concurrently on 4 CPUs**; colours match the graph legend.*

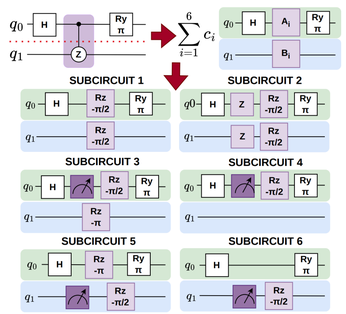
The Four Cutting Methods

Method	Mechanism	Subcircuits per k cuts	Mid-circuit measure?	Extra ancilla qubits?	Best use
(a) Gate cutting	QPD[1] of 2 qubit gate	6^k x (2 fragments)	Yes	No	No ancilla available
(b) Gate cutting (Hadamard Test)	QPD[1], ancilla based double Hadamard test	6^k x (2 fragments)	No	+2 per cut	Fast gate cutting
(c) Wire cutting	Measure-and-prepare on a qubit wire	8^k x (2 fragments)	No	+1 per cut	General wire cutting
(d) Wire cutting (optimized)	Measure-and-prepare on a qubit wire	$3^k + 4^k$	No	+1 per cut	Large number of cuts

[1]Quasi-Probability Decomposition

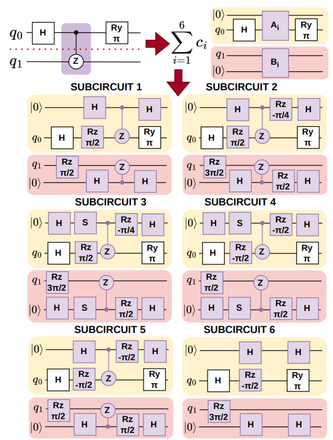
(a) Gate cutting a CZ Mid measures

A cut CZ gate is decomposed into **six fragment pairs** with classical coefficients using QDP; four of them need **mid-circuit measurements**.



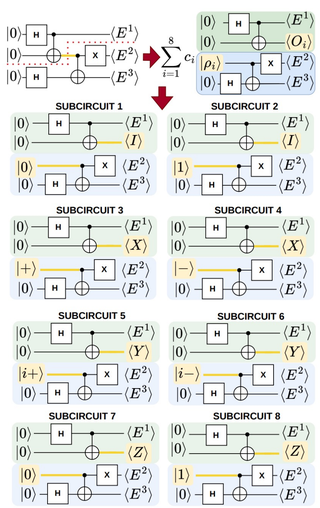
(b) Gate cutting a CZ Hadamard test

Replaces mid-circuit measurements with **ancilla-assisted Hadamard tests**: fragments are measured only at the end and **run independently**, still **requiring** $2 * 6^k$ configurations.



(c) Wire cutting a wire.

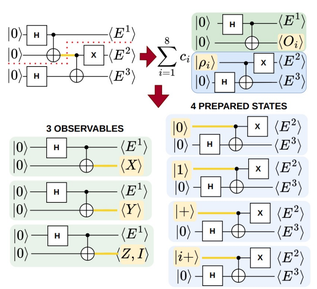
A wire cut **splits a qubit wire between two gates**: the upstream fragment measures a Pauli observable (I, X, Y, Z) and the downstream one prepares a Pauli eigenstate, giving **8 pairs per cut**.



(d) Optimized wire cutting

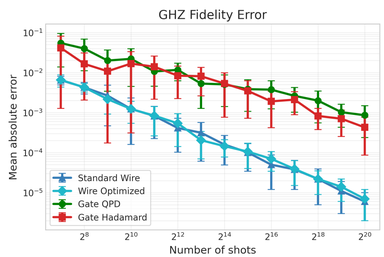
Using the **linearity of the reconstruction**, 4 Pauli measurements **reduce to 3** and 6 eigenstate preparations **reduce to 4**: **7 fragments per cut** instead of 8, with the **exact same result**.

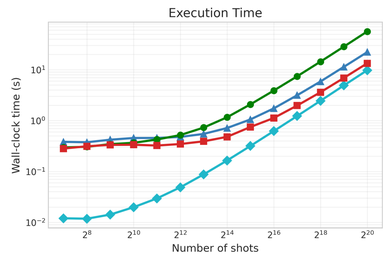
$$|-i\rangle\langle -i| = |0\rangle\langle 0| + |1\rangle\langle 1| - |+i\rangle\langle +i|$$
$$|-\rangle\langle -| = |0\rangle\langle 0| + |1\rangle\langle 1| - |+\rangle\langle +|$$



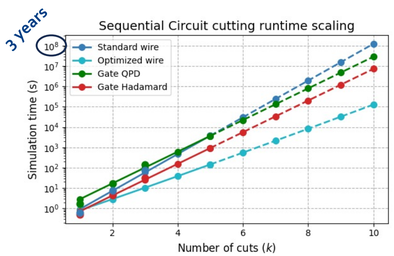
Experimental Evaluation

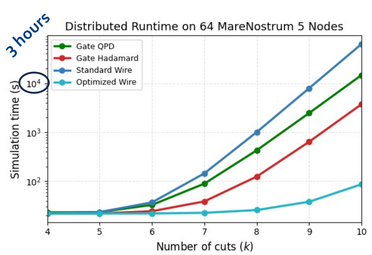
Accuracy and runtime (Qiskit Aer): wire-cutting methods reach the **lowest error** as shots increase; **Gate Hadamard matches Gate QPD accuracy** while running **faster**, with no mid-circuit measurements.





Scalability: optimized wire cutting has the **lowest runtime** as cuts grow, and running on **64 MareNostrum 5 nodes** cuts projected time **from years to hours** (dashed lines: extrapolated).





Expectation Value vs. Full-Distribution Reconstruction

Standard methods reconstruct a **single expectation value**; the **_sampling* variants in the library (i.e *gate_cutting_sampling*) recover the **full output distribution**. Both need the **same number of subcircuit evaluations** (6^k, 8^k or $3^k + 4^k$).

Resource	Expectation	Sampling
Output size	1 value	2^n probabilities
Reconstruction	Linear in configurations	Exponential in qubits
Shots	Fewer	More (rare outcomes)
Memory	**O(1)**	**O(2^n)**

More **shots → less sampling noise** and a lower **total variation distance** (**TVD**) to the exact distribution.



KEY FINDINGS:

- ✓ **Optimized wire cutting reduces fragment scaling from 8^k to $3^k + 4^k$.**
- ✓ **Gate Hadamard matches Gate QPD accuracy without mid-circuit measurements.**
- ✓ **Optimized wire cutting delivers the best runtime scaling.**
- ✓ **Distributed executions may reduce runtime from 3 years to 3 hours (64 MN5 nodes).**

Documentation **Github** **Previous Paper**

qdislib.readthedocs.io

```
pip install qdislib
```

[1] T. Peng, A. W. Harrow, M. Ozols, and X. Wu, "Simulating large quantum circuits on a small quantum computer," Physical Review Letters, vol. 125, no. 15, p. 150 504, 2020 **[2]** K. Mitarai and K. Fujii, "Constructing a virtual two-qubit gate by sampling single-qubit operations," New Journal of Physics, vol. 23, no. 2, p. 023 021, 2021 **[3]** K. Mitarai and K. Fujii, "Overhead for simulating a non-local channel with local channels by quasiprobability sampling," Quantum, vol. 5, p. 388, 2021 **[4]** A. W. Harrow and A. Lowe, "Optimal quantum circuit cuts with application to clustered Hamiltonian simulation," *PRX Quantum*, vol. 6, no. 1, p. 010316, 2025. **[5]** M. Tejedor, B. Casas, J. Conejero, A. Cervera-Lierta, and R. M. Badia, "Orchestrating quantum-HPC workflows with distributed quantum circuit cutting," *Proc. SC Workshops*, pp. 1898–1906, 2025.

Quantum SPAIN · "la Caixa" Foundation · EuroHPC-I