

Wave-Based Dispatch for Circuit Cutting in Hybrid HPC-Quantum Systems

Ricard S. Raigada-García¹ Josep Jorba¹ Sergio Iserte²¹ Universitat Oberta de Catalunya (UOC) ² Barcelona Supercomputing Center (BSC)

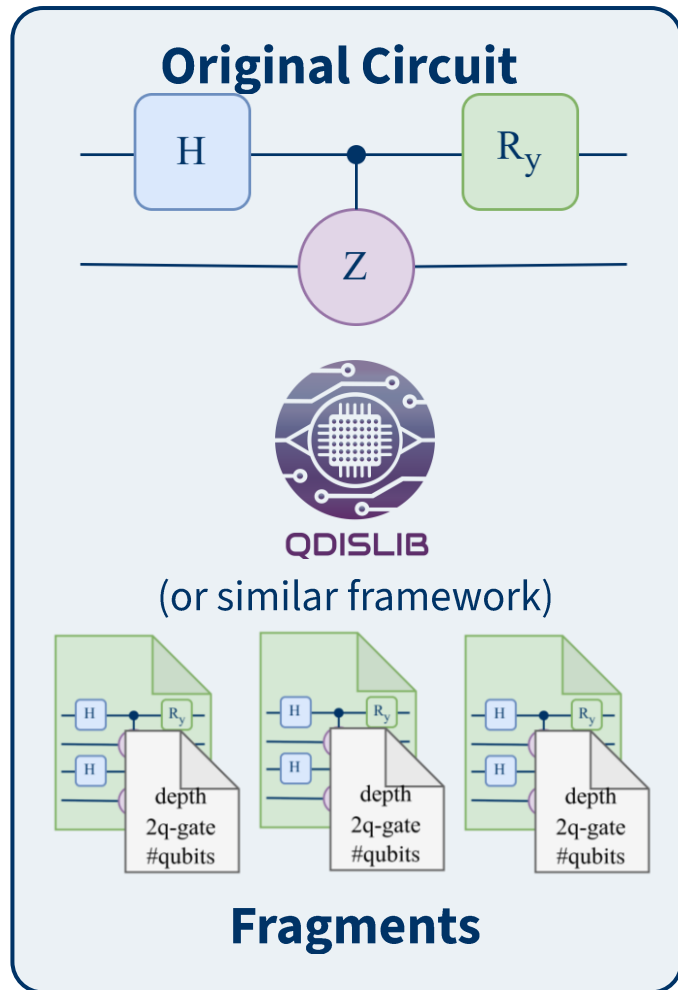
ABSTRACT: Hybrid HPC–quantum workloads based on circuit cutting decompose large quantum circuits into independent fragments, but existing frameworks tightly couple cutting logic to execution orchestration, preventing HPC centers from applying mature resource management policies to NISQ workloads. We present **DQR (Dynamic Queue Router)**, a runtime framework that treats circuit fragments as first-class schedulable units. DQR introduces a backend-agnostic fragment descriptor, a wave-based coordinator enabling pipeline concurrency through non-blocking polling, and a production-ready implementation on the **CESGA Qmio supercomputer** integrating local and cloud quantum backends. Experiments on a 32-qubit Hardware-Efficient Ansatz (HEA) demonstrate makespan improvements over a monolithic CPU baseline together with **transparent per-fragment failover recovery** without pipeline restart.

PRODUCTION HPC-QC STACK

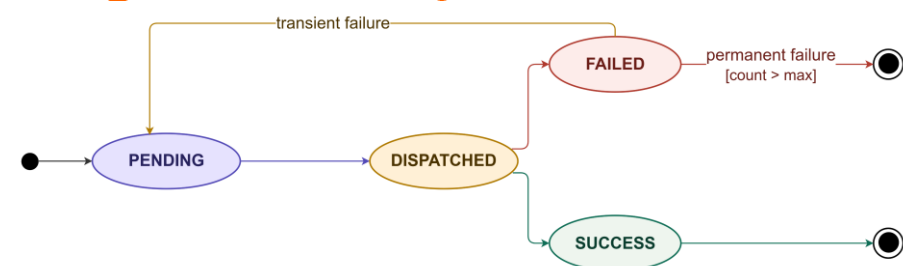


ARCHITECTURE

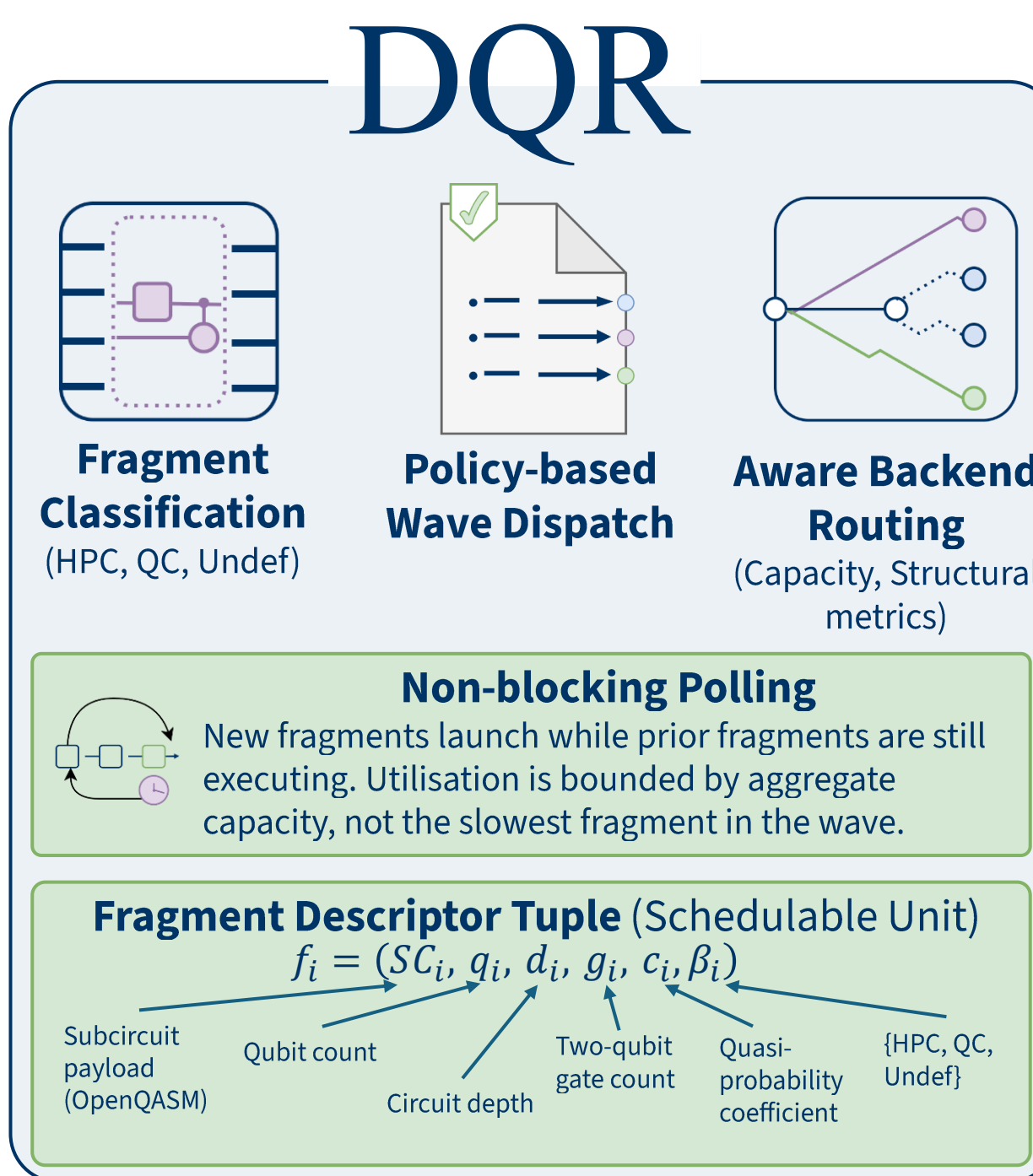
1. CIRCUIT CUTTING



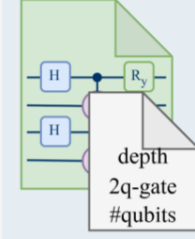
Fragment Lifecycle State Machine



2. RUNTIME ORCHESTRATION



KEY CONTRIBUTIONS



1. Fragment Abstraction
Backend-agnostic descriptor tuple exposing structural properties.

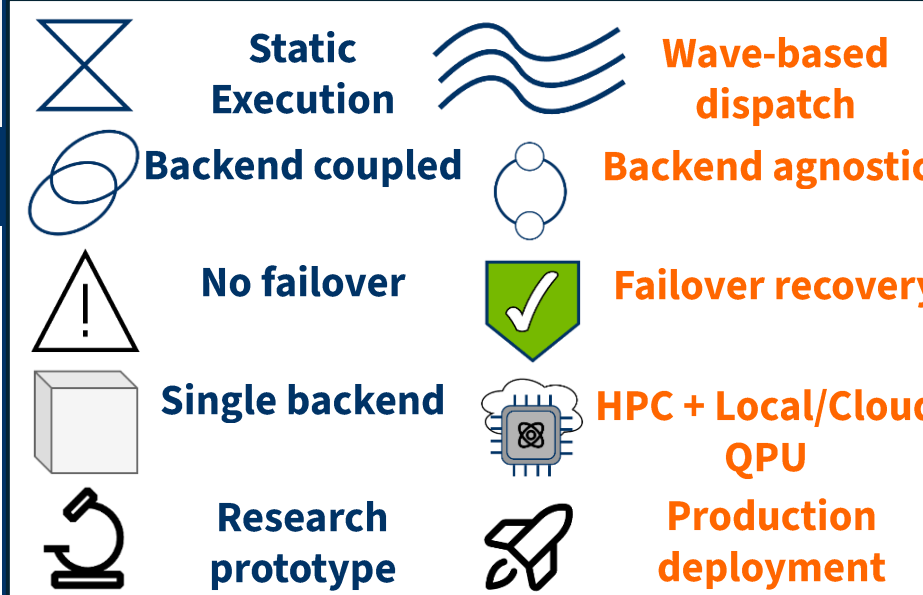


2. Wave-Based Dispatch
Pipeline concurrency, per-fragment retry and QPU→CPU failover.

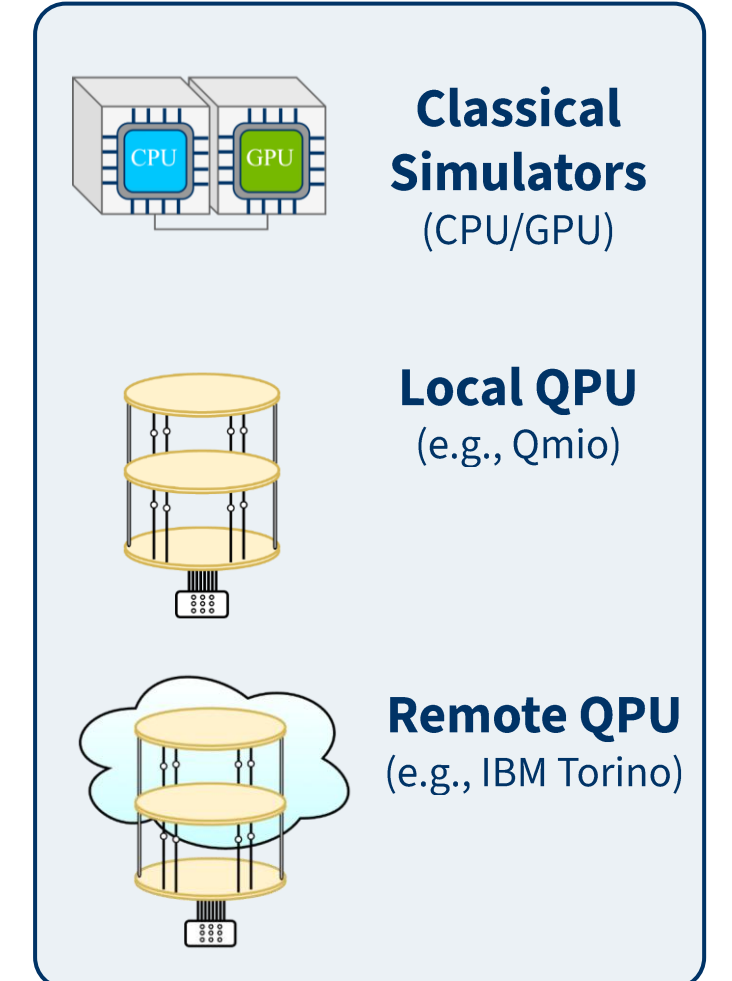


3. Production Implementation
CESGA Qmio, Qmio + IBM Torino, $k = 2$ and $k = 4$ cuts.

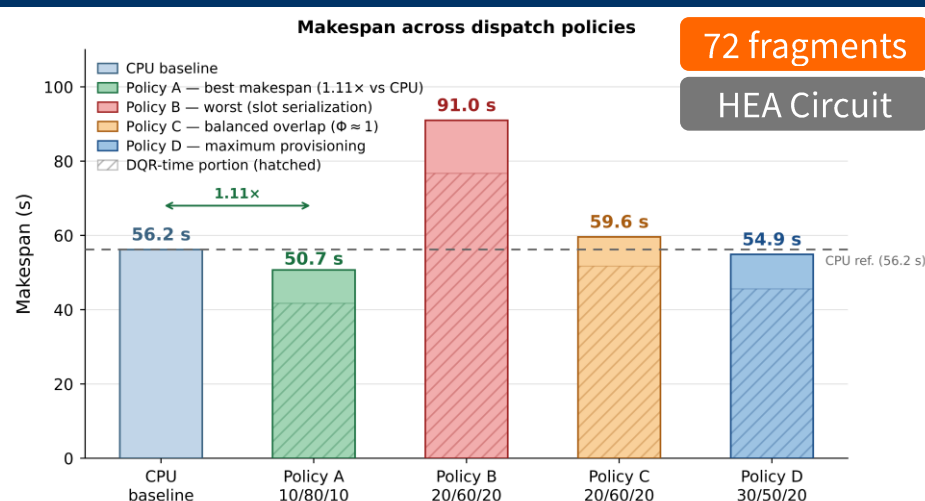
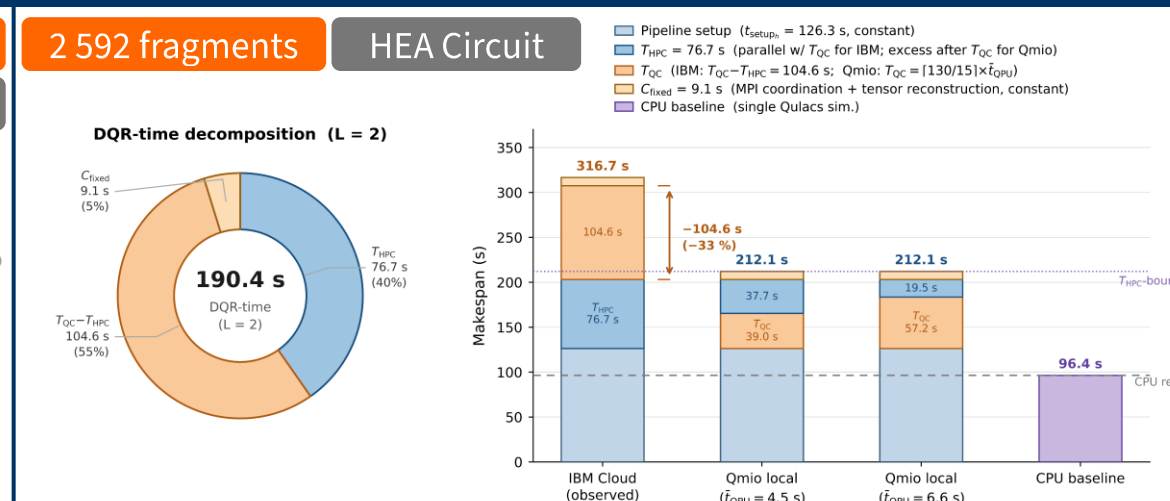
WHY DQR IS DIFFERENT



3. EXECUTION BACKENDS



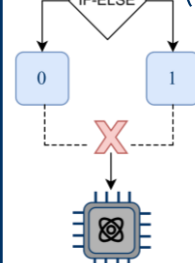
EXPERIMENTAL RESULTS & PERFORMANCE

72 fragments
HEA Circuit

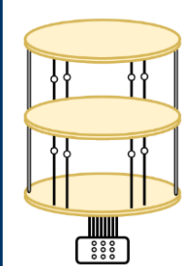
Makespan sensitivity to ϵ_{QPU} for the $L = 2$ run. Pipeline (t_{setup}) and T_{HPC} remain constant; substituting $\epsilon_{QPU} = 18.5$ s (IBM Cloud) with $\epsilon_{QPU} = 4.5$ or 6.6 s (Qmio) reduces makespan by 104.6s (33%).

Transparent Failover on Qmio QPU

Unsupported control-flow (Mid-circuit “if-else”)



After failing, DQR relabels and is successfully simulated on CPU



Qmio QPU → CESGA HPC Cluster (CPU)

ONGOING WORK

Adding malleability
DMR
dynamic management of resources

OTHERS

Acknowledgments
The authors thank the Accelcom Research Group at BSC for their support



L=1: Hybrid vs CPU Baseline

1.11×makespan improvement
over monolithic CPU baseline

Config	Makespan	Speedup
CPU only	56.2 s	—
Policy A (DQR)	50.7 s	1.11×

10/80/10 labelling (QC/HPC/U).
HPC + QPU execute concurrently.

L=2: Scalability & Fault Tolerance

95%of execution time is external or parallelized
only 5% remains as DQR coordination

Component	Share
IBM Cloud round-trip latency	55%
Parallel HPC critical path	40%
DQR coordination residual	5%

2592 fragments executed with transparent failover.