CUNQA

A DQC EMULATOR FOR HPC

Jorge Vázquez, Marta Losada, Álvaro Carballido,
Daniel Expósito, Andrés Gómez, Tomás F. Pena



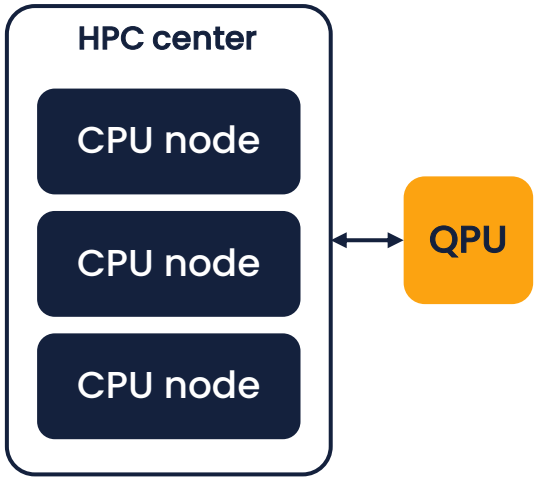
CESGA
GALICIA SUPERCOMPUTING CENTER

01

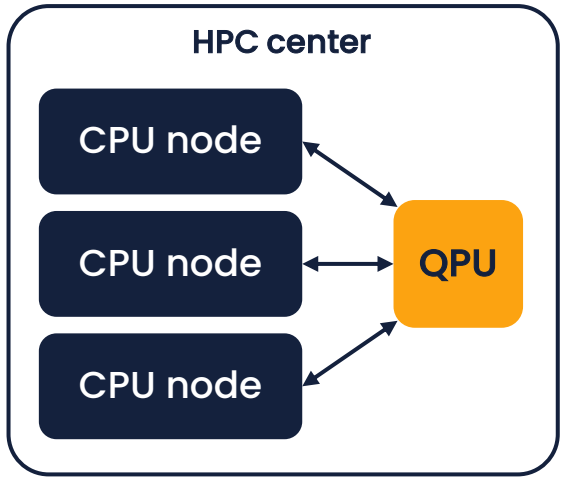
What is CUNQA?

HPC integration · DQC· Existing tools

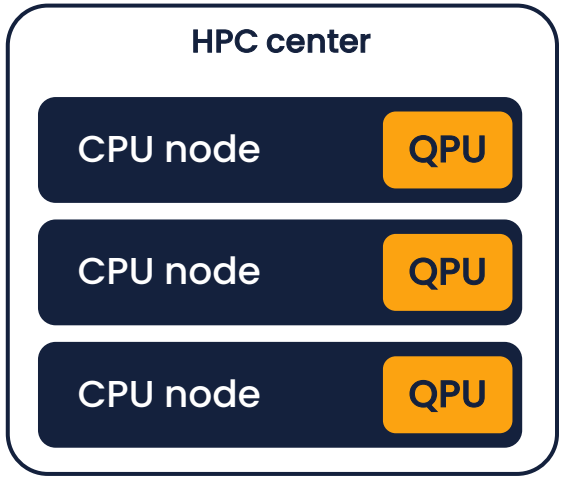
HPC INTEGRATION MODELS



STANDALONE

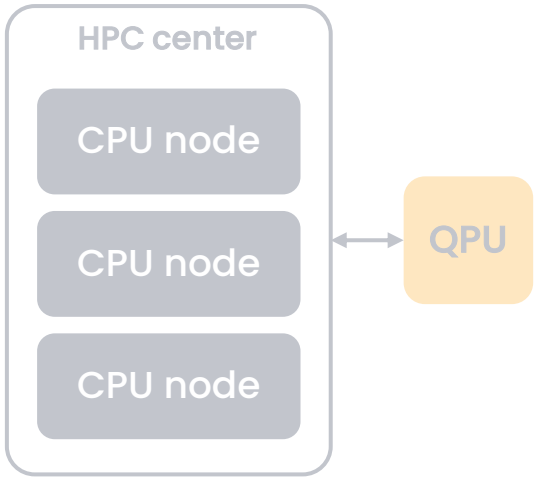


CO-LOCATED

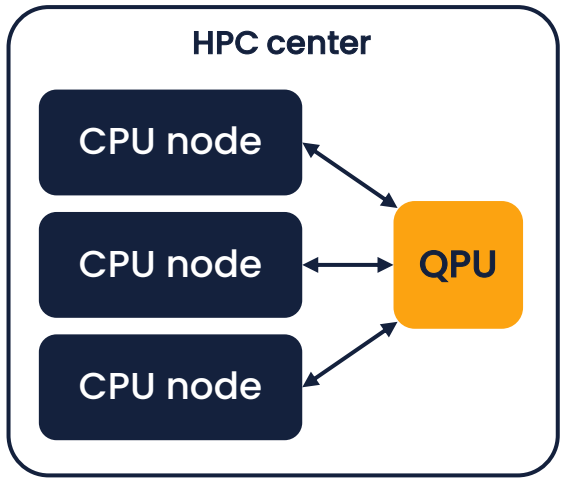


ON-NODE

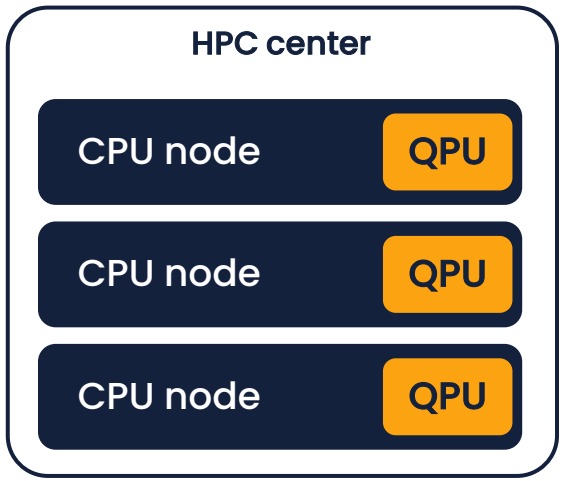
HPC INTEGRATION MODELS



STANDALONE



CO-LOCATED

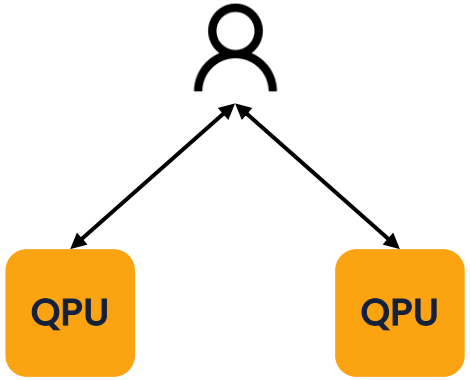


ON-NODE

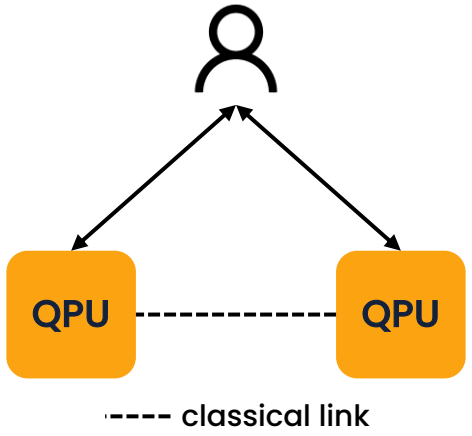
QPUs as accelerators

DISTRIBUTED QUANTUM COMPUTING

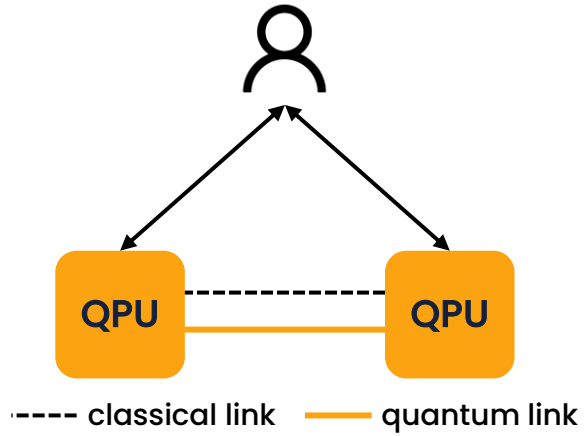
Distributed Quantum Computing (DQC) distributes quantum tasks along quantum nodes.



NO COMMUNICATIONS



CLASSICAL COMMUNICATIONS



QUANTUM COMMUNICATIONS

CUNQA VS EXISTING TOOLS

Tool family	Examples	HPC integration	No comm.	Classical comm.	Quantum comm.
HPC-QC integration frameworks	QFw, MQSS, CUDA-Q	✓	✓	✗	✗
Quantum-network tools	NetSquid, SeQUeNCe, SimulaQron	✗	~	✓	✓
MPI-style interfaces for DQC	QMPI, NetQMPI	~	~	✓	✓
Monolithic simulators	AerSimulator	✗	✓	✗	✗
CUNQA		✓	✓	✓	✓

CUNQA is the only tool that combines HPC integration with all three DQC communication models.

02

How does CUNQA do it?

vQPUs · Execution models · qraise

VIRTUAL QPUS (vQPUs)

A vQPU is an **HPC process with its own classical resources** that stands in for a real QPU.

vQPU =

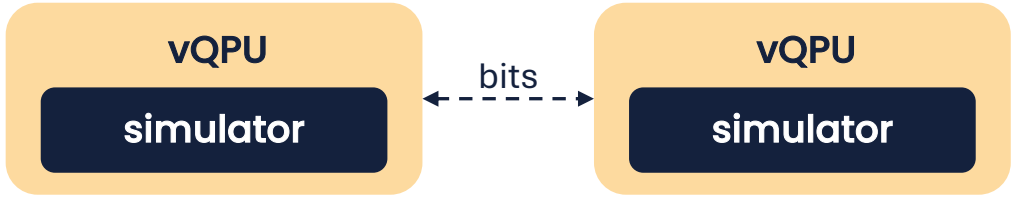
SERVER	Receives quantum tasks and returns the results Classical socket library
BACKEND	Describes the emulated QPU JSON file passed at launch
SIMULATOR	Executes the tasks AerSimulator, MQT-DDSIM, Maestro, Qulacs, Qsim, QuEST, CunqaSimulator

HOW vQPUs EXECUTE

NO COMMUNICATION



CLASSICAL COMMUNICATION



QUANTUM COMMUNICATION



RAISING vQPUs WITH QRAISE

1. **Simulator.** Seven available; default: AerSimulator

```
[user@login01 ~]$ qraise -n 4 -t 00:30:00 --sim=Qulacs
```

2. **Backend.** Qubits, coupling map, basis gates, noise

```
[user@login01 ~]$ qraise -n 4 -t 00:30:00 --backend=backend.json
```

3. **Model.** DQC communication and HPC integration

```
[user@login01 ~]$ qraise -n 4 -t 00:30:00 --quantum_comm --co-located
```

Other flags: **`--cores-per-qpu --mem-per-qpu --n-nodes --family`**

PYTHON API: DESIGNING CIRCUITS

```
# Classical communication
c0 = CunqaCircuit(2, 2, id="c0")
C0.h(0)
c0.measure(0, 0)

# Classical communication
c0 = CunqaCircuit(2, 2, id="c0")
c0.send(clbits=0, recving_circuit="c1")
c1.recv(clbits=0, sending_circuit="c0")

# Quantum communication: (data, comm) qubits
q0 = CunqaCircuit((3, 2), 3, id="q0")
q1 = CunqaCircuit((1, 1), 1, id="q1")
q0.gen_ent(comm_qubit=1, circuits="q1")
q1.gen_ent(comm_qubit=0, circuits="q0")
```

PYTHON API: RUNNING CIRCUITS

1

get_QPUs()
Obtaining the raised vQPUs

2

run(circuits, qpus)
Sending the circuits to the vQPUs

3

gather(qjobs)
Getting the results of the vQPUs

4

upgrade_parameters()
Variational loops send only the new parameters

```
from cunqa import get_QPUs, run, gather

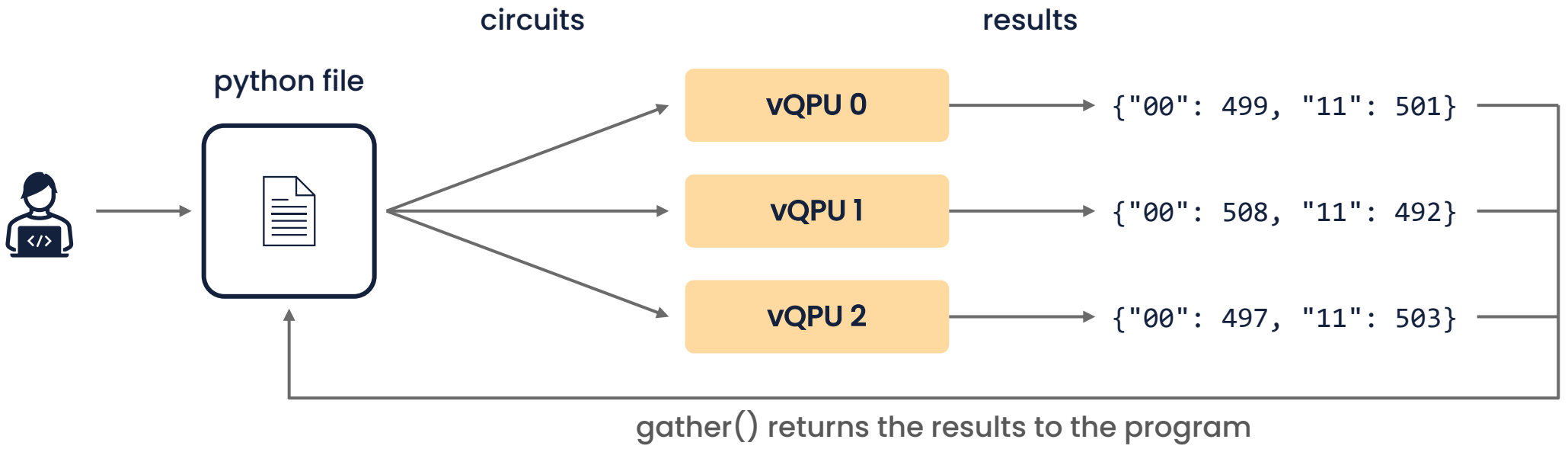
qpus    = get_QPUs(family="demo")
qjobs   = run([c0, c1], qpus[:2], shots=1000)
results = gather(qjobs)
results[0].counts      # {"00": 499, "11": 501}
```

WORKFLOW

1 Raise vQPUs
qraise from the login node

2 Set up environment
Load modules, export variables

3 Run the Python file
It calls the Python API



SUMMARY

01 WHAT IS CUNQA?

1 HPC integration

QPUs join HPC centers as accelerators, standalone or co-located

2 Distributed QC

Tasks split across QPUs with no, classical or quantum communication

3 CUNQA's place

The only tool that combines HPC integration with all three DQC models

02 HOW DOES CUNQA DO IT?

4 vQPUs

HPC processes that emulate a QPU: server, backend and simulator

5 qraise

One command raises vQPUs with a chosen simulator, backend and model

6 Python API

CunqaCircuit designs the circuits; get_QPUs, run and gather execute them

Raise vQPUs, write circuits, run them from Python: distributed quantum computing on today's HPC.

Thank you.

jvazquez@cesga.es

Code github.com/CESGA-Quantum-Spain/cunqa

This work was supported by the Xunta de Galicia under the “Programa de axudas á etapa predoutoral”, through the Axencia Galega de Innovación (GAIN), under the 2025 call [IN606A-2025/012], and through the Consellería de Cultura, Educación e Universidade, under the 2022 call [ED481A 2022/241]. It also was supported by project PID2024-159713OB-I00, funded by MICIU/AEI/10.13039/501100011033 and ERDF, EU.