Report Quantum Emulators at CINECA

Riccardo Mengoni, r.mengoni@cineca.it
Daniele Ottaviani, d.ottaviani@cineca.it

I. CINECA HPC SYSTEM: MARCONI 100

MARCONI 100 (M100) is the new accelerated cluster based on IBM Power9 architecture and Volta NVIDIA GPUs, acquired by Cineca within PPI4HPC European initiative. This system opens the way to the pre-exascale Leonardo Supercomputer expected to be installed in 2021.It is available from April 2020 to the Italian public and industrial researchers. Its computing capacity is about 32 PFlops.

More info here: https://indico.euro-fusion.org/event/341/attachments/293/664/M100.pdf
Quick Startup Giude : https://wiki.u-gov.it/confluence/pages/viewpage.action?pageId=336727645

A. M100 Features

- **Nodes**: 980
- **Processors**: 2x16 cores IBM POWER9 AC922 at 3.1 GHz
- **Accelerators**: 4 x NVIDIA Volta V100 GPUs, Nvlink 2.0, 16GB
- **Cores**: 32 cores/node
- **Hyper-Threading**: 128 (virtual) cpus [32 physical cores with 4 HTs each]
- **RAM**: 256 GB/node (242 usable)
- **Peak Performance**: 32 PFlop/s

II. OPEN SOURCE QUANTUM EMULATORS

- **Qiskit**, open-source python library by IBM for developing quantum algorithms. Qiskit supports multi-threading (using OpenMP) emulation methods and additional configurable options.
  Webpage: https://qiskit.org/
- **Cirq**, open-source python library by Google for the development of quantum algorithms.
  Webpage: https://quantumai.google/cirq
- **Qsim (Cirq)**: It uses gate fusion, AVX/FMA vectorized instructions and multi-threading using OpenMP to achieve state of the art simulations of quantum circuits. qsim is integrated with Cirq.
  Webpage: https://quantumai.google/qsim
- **QuTip**, open-source python library for circuit emulation and resolution of open quantum systems.
  Webpage: http://qutip.org/.

III. TESTING ALGORITHM: QUANTUM FOURIER TRANSFORM (QFT)

The quantum Fourier transform (QFT) is the quantum analogue of the classical discrete Fourier transform (DFT) which maps a vector $x = (x_0, x_1, \ldots, x_{N-1}) \in \mathbb{C}^N$ to a new vector $y = (y_0, y_1, \ldots, y_{N-1}) \in \mathbb{C}^N$, where $N = 2^n$, as follows

$$y_k = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} x_j e^{2\pi i \frac{kj}{N}} \tag{1}$$

The QFT acts on the 2^n amplitudes of the quantum state

$$|x\rangle = \sum_{j=0}^{N-1} x_j |j\rangle \tag{2}$$

and maps them to the new amplitudes of the quantum state $|y\rangle$

$$|y\rangle = \sum_{k=0}^{N-1} y_k |k\rangle \qquad \text{where} \qquad y_k = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} x_j e^{2\pi i \frac{kj}{N}} \tag{3}$$

The quantum gate associated to the QFT has the following matrix representation, where $\omega = e^{\frac{2\pi i}{N}}$

$$\mathcal{F} = \frac{1}{\sqrt{N}} \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \omega & \omega^2 & \cdots & \omega^{N-1} \\ 1 & \omega^2 & \omega^4 & \cdots & \omega^{2(N-1)} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & \omega^{N-1} & \omega^{2(N-1)} & \cdots & \omega^{(N-1)(N-1)} \end{bmatrix} \tag{4}$$

The discrete Fourier transform on 2^n amplitudes can be implemented as a quantum circuit of only $O(n^2)$ gates while the classical discrete Fourier transform takes $O(n2^n)$ on a classical computer. The exponential speed-up of the quantum Fourier makes it one of the most widely used subroutines of many quantum algorithms.

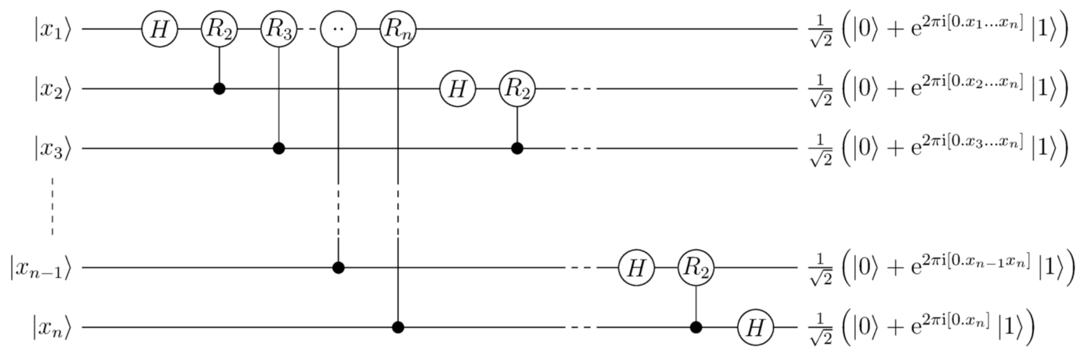
$|x_1\rangle$
$|x_2\rangle$
$|x_3\rangle$
$|x_{n-1}\rangle$
$|x_n\rangle$
H
R_2
R_3
R_n
$\frac{1}{\sqrt{2}}\left(|0\rangle + e^{2\pi i[0.x_1 \ldots x_n]}|1\rangle\right)$
$\frac{1}{\sqrt{2}}\left(|0\rangle + e^{2\pi i[0.x_2 \ldots x_n]}|1\rangle\right)$
$\frac{1}{\sqrt{2}}\left(|0\rangle + e^{2\pi i[0.x_3 \ldots x_n]}|1\rangle\right)$
$\frac{1}{\sqrt{2}}\left(|0\rangle + e^{2\pi i[0.x_{n-1} x_n]}|1\rangle\right)$
$\frac{1}{\sqrt{2}}\left(|0\rangle + e^{2\pi i[0.x_n]}|1\rangle\right)$

FIG. 1: Quantum circuit for Quantum-Fourier-Transform with n qubits (without rearranging the order of output states). It uses the binary fraction notation introduced below.

IV. PERFORMANCES M100

Performances are based on the computational time needed to emulate the full state-vector associated to the QFT algorithm on n-qubits using a single node of M100. In general, a state vector of n-qubits uses 2^n complex values (each complex number needs 16 Bytes in double precision or 8 Bytes in single precision). The memory usage is shown in the plot below. Remember that each node in M100 has 242 GB of available RAM.

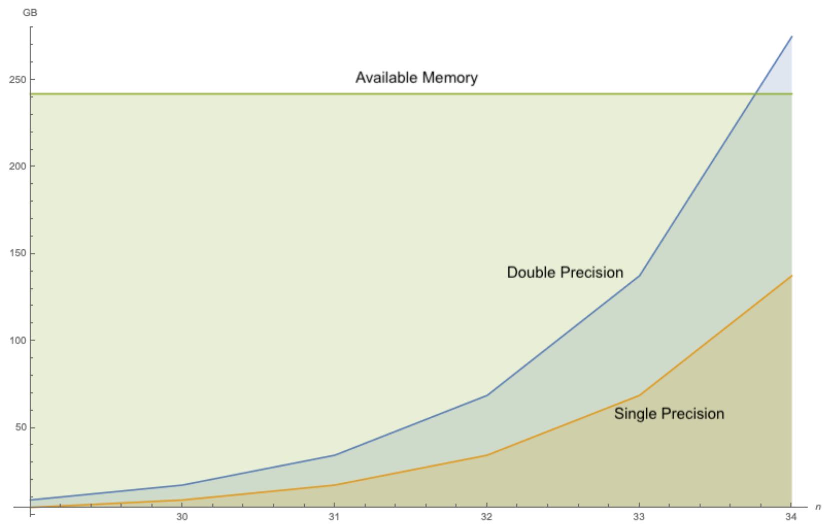

FIG. 2: Memory required (express in GB) by the state vector increasing the number of qubits n (x axis on the plot) Each node has a memory of 242 GB, the theoretical maximum number of qubits that the node can handle is 33 in double precision, 34 in single precision

A. QISKIT: StatevectorSimulator

StatevectorSimulator is an Ideal quantum circuit emulator of the state vector designed to run on a single machine. Supports Multi-Threading. For more info: https://qiskit.org/documentation/apidoc/aer_provider.html

Results up to $n = 30$

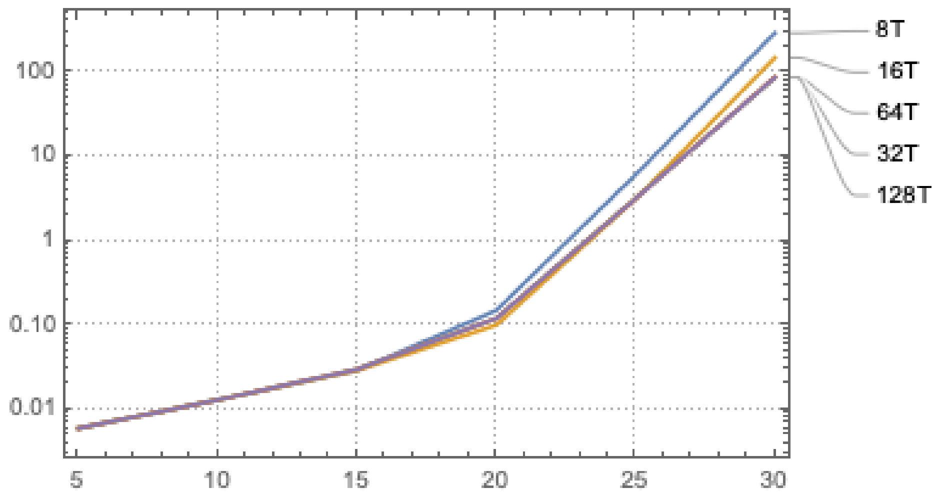

FIG. 3: The plot (Log-scale) shows on the y-axis the runtime (in seconds) of the QFT algorithm increasing the number of qubits (x- axis on the plot). We were able to run the QFT algorithm using a single node up to 30 qubits in under 100 seconds with a number of threads ≥ 32.

Results up to $n = 33$ in double precision and $n = 34$ in single precision

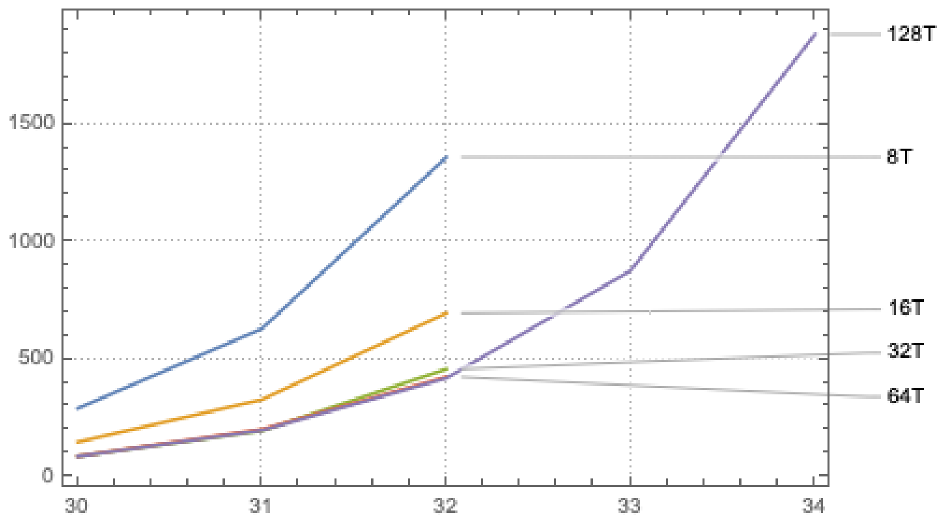

FIG. 4: The plot shows on the y-axis the runtime (in seconds) of the QFT algorithm increasing the number of qubits (x- axis on the plot). We were able to run the QFT algorithm using a single node up to 33 qubits (double precision) in under 1000 seconds and 34 qubits (single precision) in under 2000 seconds with 128 threads.

B. Cirq: cirq.Simulator

The Cirq Simulator is an Ideal quantum circuit emulator of the vector associated to the quantum state. Does not support Multi-Threading. For more info: https://quantumai.google/cirq/simulation

Results up to $n = 25$

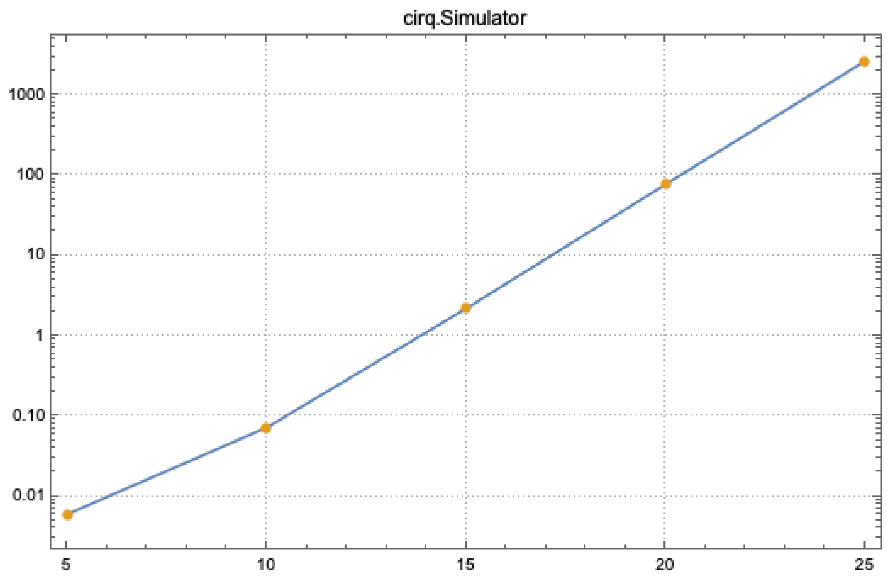

FIG. 5: The plot (Log-scale) shows on the y-axis the runtime (in seconds) of the QFT algorithm increasing the number of qubits (x- axis on the plot). We were able to run the QFT algorithm using a single node up to 25 qubits in under 2800 seconds.

C. Qsim: qsimcirq.QSimSimulator

Qsim is a state-vector simulator designed to run on a single machine. It produces the full state vector as output. Supports Multi-Threading. Qsim works using single precision arithmetic only.
For more info: https://quantumai.google/qsim/tutorials/qsimcirq

Results up to $n = 30$

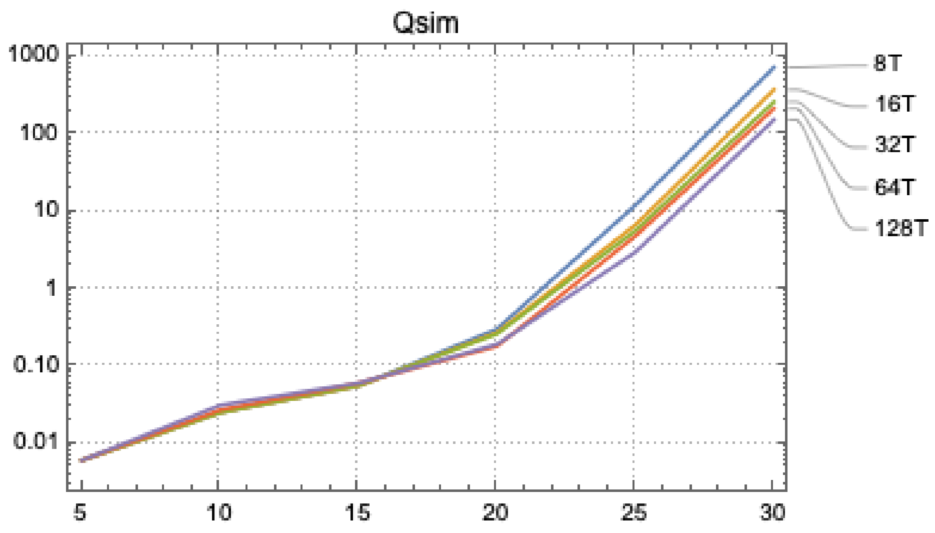

FIG. 6: The plot (Log-scale) shows on the y-axis the runtime (in seconds) of the QFT algorithm increasing the number of qubits (x- axis on the plot). We were able to run the QFT algorithm using a single node up to 30 qubits. The 30 qubits QFT was emulated in 152 seconds using 128 threads.

Results up to $n = 34$

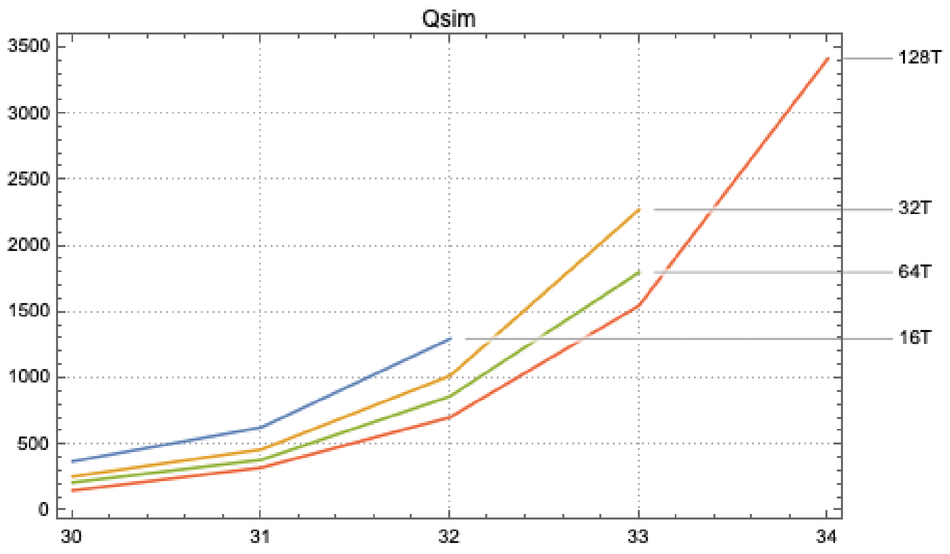

FIG. 7: The plot shows on the y-axis the runtime (in seconds) of the QFT algorithm increasing the number of qubits (x- axis on the plot). We were able to run the QFT algorithm using a single node up to 34 qubits (single precision) in under 3500 seconds with 128 threads.

D. QuTIP

QuTIP simulates the full state-vector. Does not support Multi-Threading. For more info: http://qutip.org/

Results up to $n = 15$

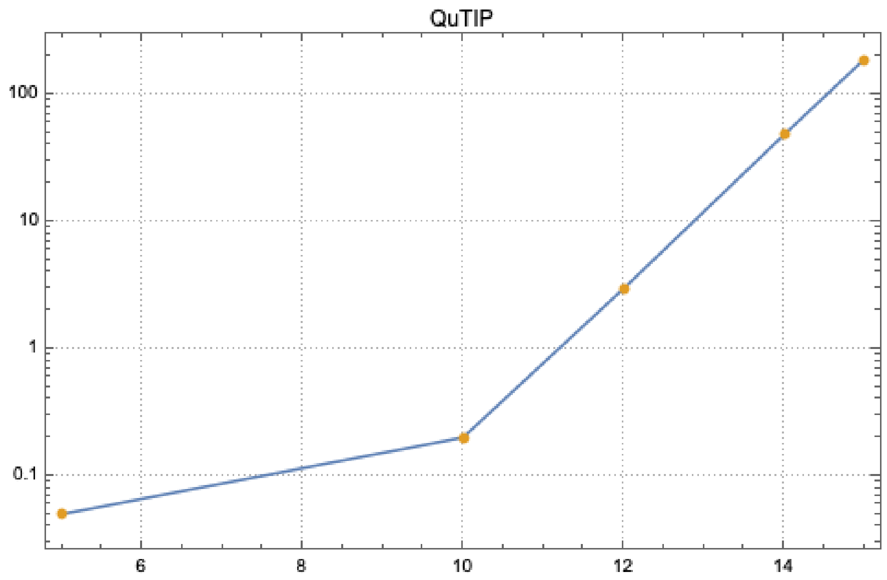

FIG. 8: The plot (Log-scale) shows on the y-axis the runtime (in seconds) of the QFT algorithm increasing the number of qubits (x- axis on the plot). We were able to run the QFT algorithm using a single node up to 15 qubits in under 200 seconds.