Benchmarking JUWELS Booster with the Jülich Universal Quantum Computer Simulator

January 20, 2021 I Dr. Dennis Willsch

CONTENTS

1. **Quantum computing**
2. **JUQCS:** Simulating quantum computers
3. **JUQCS-G:** Simulating quantum computers on GPUs
4. **JUQMES:** Simulating physical realizations of quantum computers
5. **Conclusion**

JÜLICH SUPERCOMPUTING CENTRE

Prof. Dr. Hans De Raedt

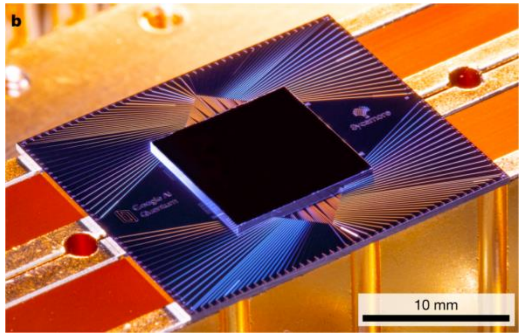
Dr. Dennis Willsch

Prof. Dr. Kristel Michielsen

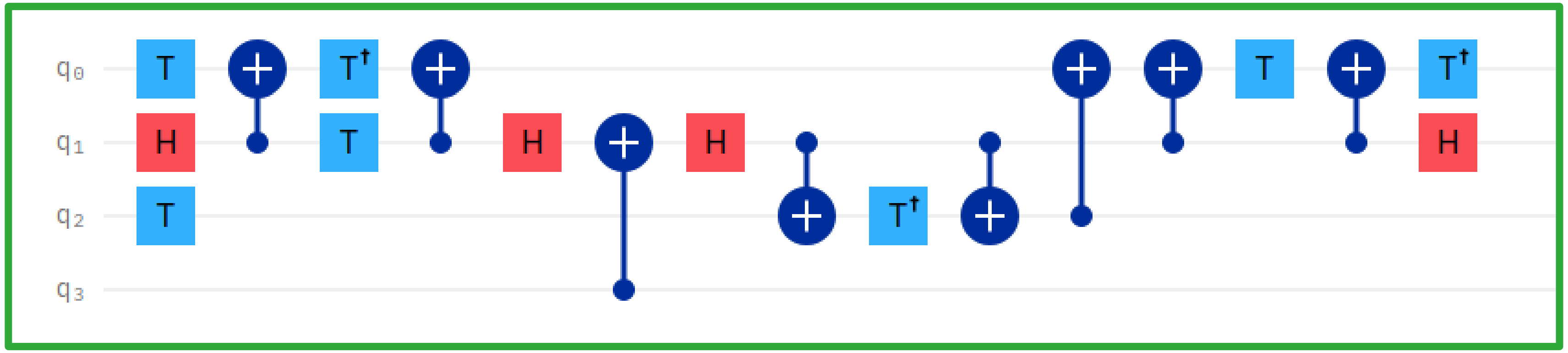
JÜLICH Forschungszentrum

QUANTUM COMPUTING

Ideal gate-based quantum computing



- What does a (gate-based) quantum computer do?
  - It runs a quantum circuit



- What does this mean, actually?
  - It performs **matrix-vector multiplications** that are

    sparse **complex** **unitary**

  - with **huge** vectors and **huge²** matrices

JÜLICH Forschungszentrum

QUANTUM COMPUTING

Ideal gate-based quantum computing

vector = state of the QC = 2^n complex numbers

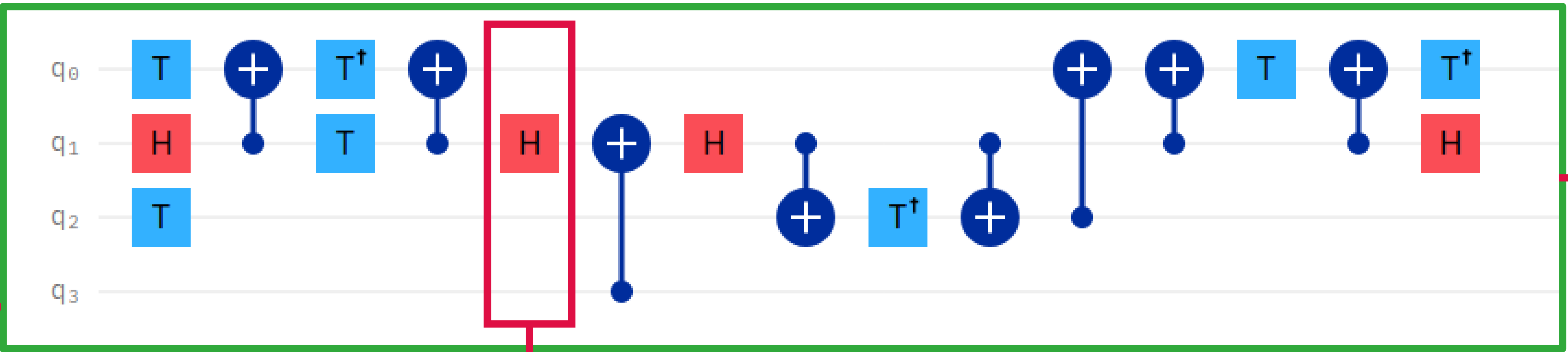
$$|\psi\rangle = \psi_{0\cdots 0}|0\cdots 0\rangle + \cdots + \psi_{1\cdots 1}|1\cdots 1\rangle = \begin{pmatrix}\psi_{0\cdots 0}\\ \vdots \\ \psi_{1\cdots 1}\end{pmatrix}$$

➢ What kind of sparse, unitary **matrix-vector multiplications**, precisely?

each quantum gate = 1 sparse, unitary **matrix**

➢ **Example:**

$n = 4$ qubits
$2^n = 16$ complex numbers



Initial state of QC:

$$|\psi\rangle = |0000\rangle = \begin{pmatrix}1\\0\\ \vdots \\0\end{pmatrix}$$

Example matrix-vector multiplication: H on qubit q_1

For each q_3, q_2, q_0 perform 2x2 update:

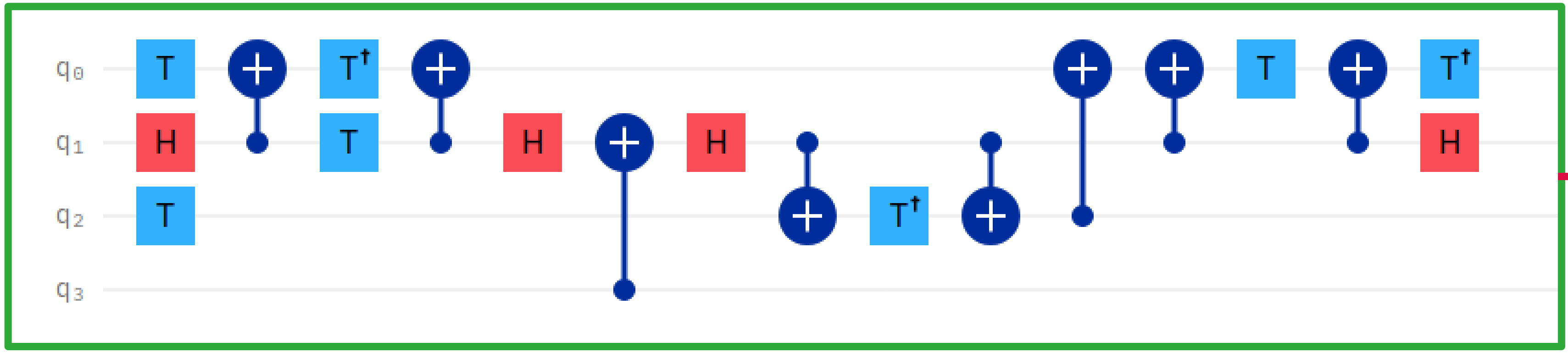
$$\begin{pmatrix}\psi_{q_3 q_2 0 q_0}\\ \psi_{q_3 q_2 1 q_0}\end{pmatrix} \leftarrow \frac{1}{\sqrt{2}}\begin{pmatrix}1 & 1\\ 1 & -1\end{pmatrix}\begin{pmatrix}\psi_{q_3 q_2 0 q_0}\\ \psi_{q_3 q_2 1 q_0}\end{pmatrix}$$

Final state of QC:

$$|\psi\rangle = \begin{pmatrix}\psi_{0000}\\ \vdots \\ \psi_{1111}\end{pmatrix}$$

QUANTUM COMPUTING

Ideal gate-based quantum computing



Final state of QC:

$$|\psi\rangle = \begin{pmatrix} \psi_{0000} \\ \vdots \\ \psi_{1111} \end{pmatrix}$$

- What does a **hardware realization** of a QC return?
  - The quantum state after all sparse matrix-vector multiplications?
  - No! That would be 2^n complex numbers.

    For 40 qubits: 2^{40} $\psi'\mathrm{s}$ = 16 TiB complex numbers

  - What then? Only a single bitstring $j_{n-1} \cdots j_1 j_0$ with n bits
  - The complex numbers only define the **probability**:

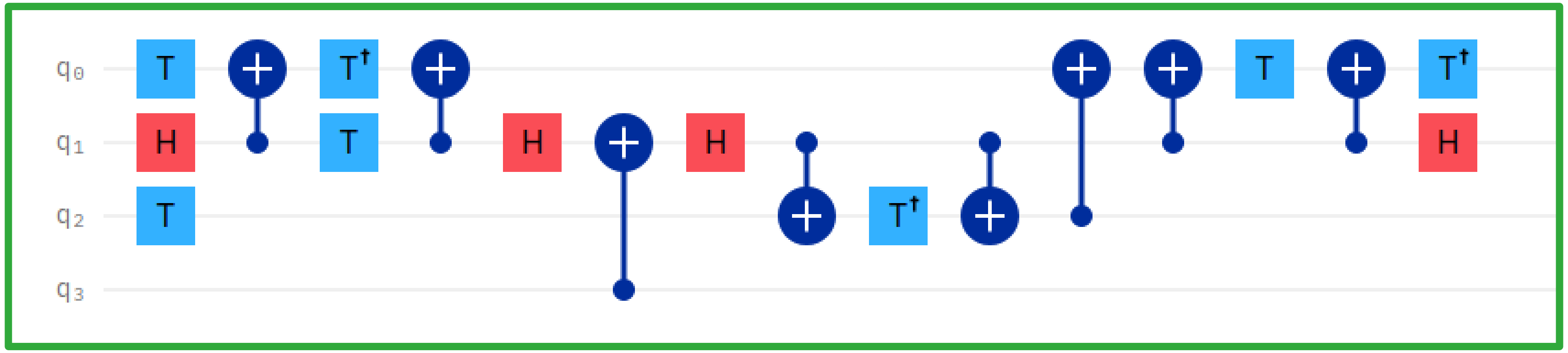
    $\left|\psi_{j_{n-1} \cdots j_1 j_0}\right|^2$ = probability to return bitstring $j_{n-1} \cdots j_1 j_0$

  - Need to run circuit multiple times to **sample** from the bitstring distribution

JUQCS

Jülich universal quantum computer simulator

- What does a quantum computer **simulator** do?
  - It runs a quantum circuit



- What does this mean, actually?
  - It performs **matrix-vector multiplications** that are

sparse **complex** **unitary**

  - with **huge** vectors and **huge²** matrices

JÜLICH Forschungszentrum

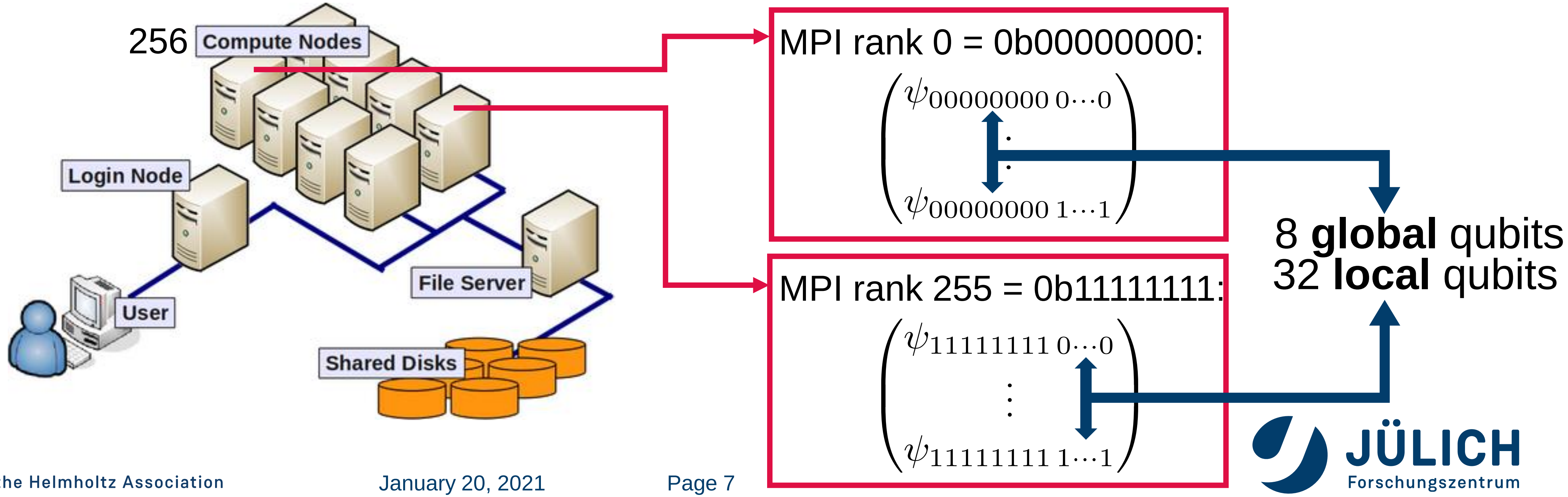
JUQCS

Distribution of the quantum state

How does the simulator manage all these complex numbers?

→ Distribute quantum state $|\psi\rangle = (\psi_{\cdots q_2 q_1 q_0})$ over multiple compute nodes

For 40 qubits: $2^{40}\ \psi'\mathrm{s}$ = 16 TiB complex numbers = 64 GiB complex numbers on 256 nodes

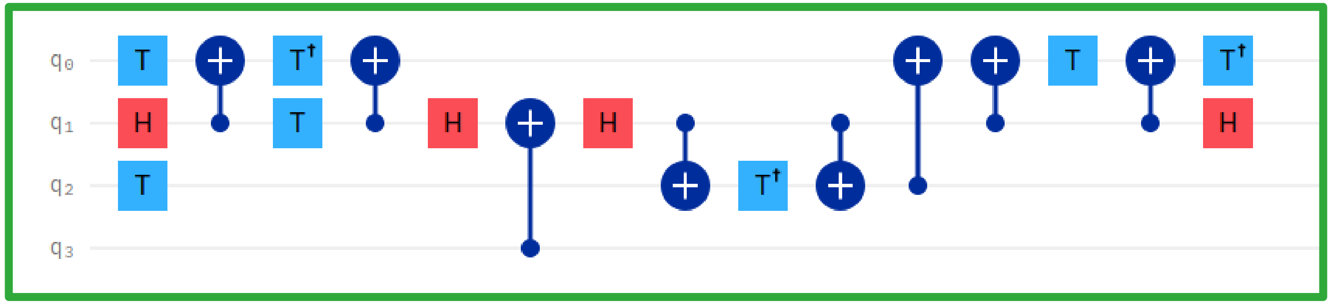


$$\begin{pmatrix} \psi_{00000000\,0\cdots0} \\ \vdots \\ \psi_{00000000\,1\cdots1} \end{pmatrix} \qquad \begin{pmatrix} \psi_{11111111\,0\cdots0} \\ \vdots \\ \psi_{11111111\,1\cdots1} \end{pmatrix}$$

JÜLICH Forschungszentrum

JUQCS

MPI communication scheme



How to implement these **matrix-vector multiplications** in the most efficient way?

Full quantum state:

$\psi_{00000000\,0\cdots0}$
MPI rank 0
$\vdots$
$\psi_{00000000\,1\cdots1}$

$\psi_{00000001\,0\cdots0}$
MPI rank 1
$\vdots$
$\psi_{00000001\,1\cdots1}$

$\vdots$

$\psi_{11111111\,0\cdots0}$
MPI rank 255
$\vdots$
$\psi_{11111111\,1\cdots1}$

Quantum gate on **local** qubits:

e.g. H on qubit q_{30}

→ Each MPI rank r performs local 2x2 updates of the form

$$\begin{pmatrix} \psi_{rrrrrrrr\,*0*\cdots*} \\ \psi_{rrrrrrrr\,*1*\cdots*} \end{pmatrix} \leftarrow \text{H} \begin{pmatrix} \psi_{rrrrrrrr\,*0*\cdots*} \\ \psi_{rrrrrrrr\,*1*\cdots*} \end{pmatrix}$$

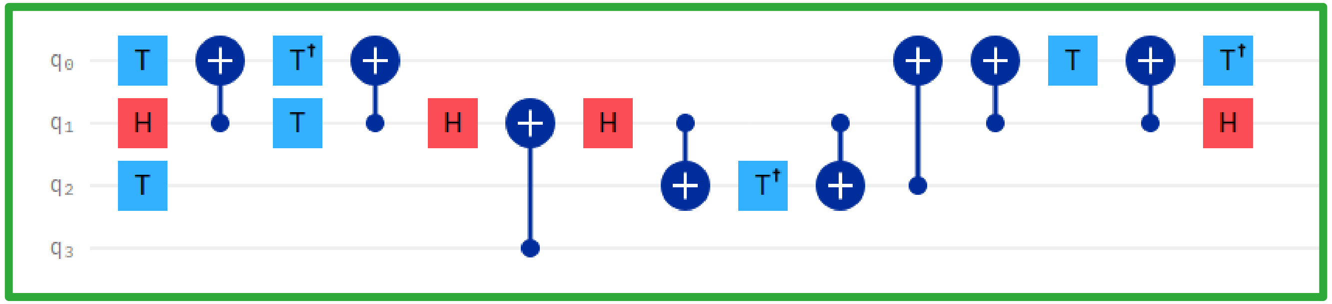
$$\uparrow$$

$$\frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

JÜLICH Forschungszentrum

JUQCS

MPI communication scheme



How to implement these **matrix-vector multiplications** in the most efficient way?

Full quantum state:

$\psi_{00000000\,0\cdots0}$ ⋮ $\psi_{00000000\,1\cdots1}$ — MPI rank 0

$\psi_{00000001\,0\cdots0}$ ⋮ $\psi_{00000001\,1\cdots1}$ — MPI rank 1

⋮

$\psi_{11111111\,0\cdots0}$ ⋮ $\psi_{11111111\,1\cdots1}$ — MPI rank 255

Quantum gate on **global** qubits:

e.g. H on qubit q_{32}

- Need to perform 2x2 updates of the form

$$H \begin{pmatrix} \psi_{rrrrrrr0\,*\cdots*\,r} \\ \psi_{rrrrrrr1\,*\cdots*\,r} \end{pmatrix}$$

- Problem: the numbers are on separate nodes
- Naïve solution:
  - Transfer $2^n/2$ ψ's (8 TiB), perform H, transfer back
- Optimal solution:
  - Exchange global and local qubit, e.g. $q_{32} \leftrightarrow q_0$
  - Transfer $2^n/2$ ψ's only **once**
  - Keep track of qubit assignment in a permutation

$\psi_{0000000\,0\cdots0\,0}$ ⋮ $\psi_{0000000\,1\cdots1\,0}$ — MPI rank 0

$\psi_{0000000\,0\cdots0\,1}$ ⋮ $\psi_{0000000\,1\cdots1\,1}$ — MPI rank 1

⋮

$\psi_{1111111\,0\cdots0\,1}$ ⋮ $\psi_{1111111\,1\cdots1\,1}$ — MPI rank 255

JÜLICH Forschungszentrum

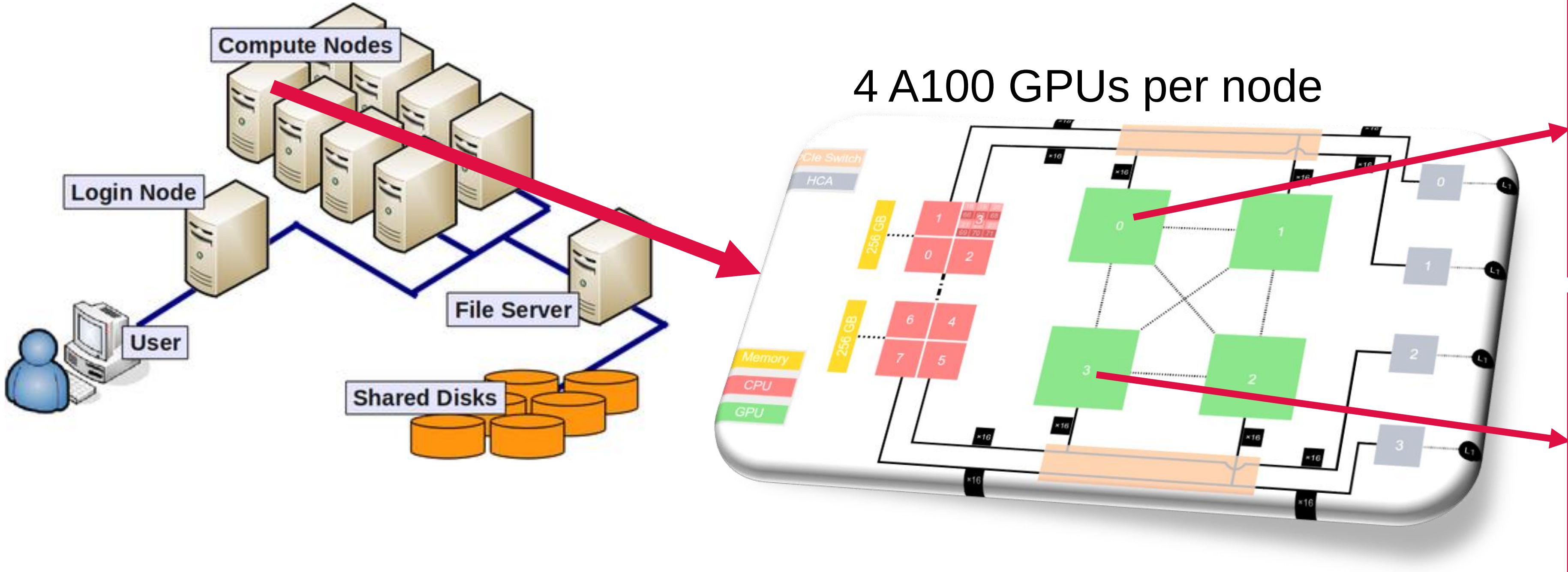
JUQCS-G

Simulating quantum computers on JUWELS Booster

- Distribute quantum state on GPUs (40GB per GPU) **CUDA MPI Fortran**

For 40 qubits: 2^{40} ψ's = 16 TiB complex numbers = 16 GiB complex numbers on 4*256 GPUs

4 A100 GPUs per node



MPI rank 0 = 0b0000000000:

$$\begin{pmatrix} \psi_{0000000000\,0\cdots0} \\ \vdots \\ \psi_{0000000000\,1\cdots1} \end{pmatrix}$$

MPI rank 3 = 0b0000000011:

$$\begin{pmatrix} \psi_{0000000011\,0\cdots0} \\ \vdots \\ \psi_{0000000011\,1\cdots1} \end{pmatrix}$$

- The MPI communication scheme and the 2x2 / 4x4 updates are the same

JUQCS-G

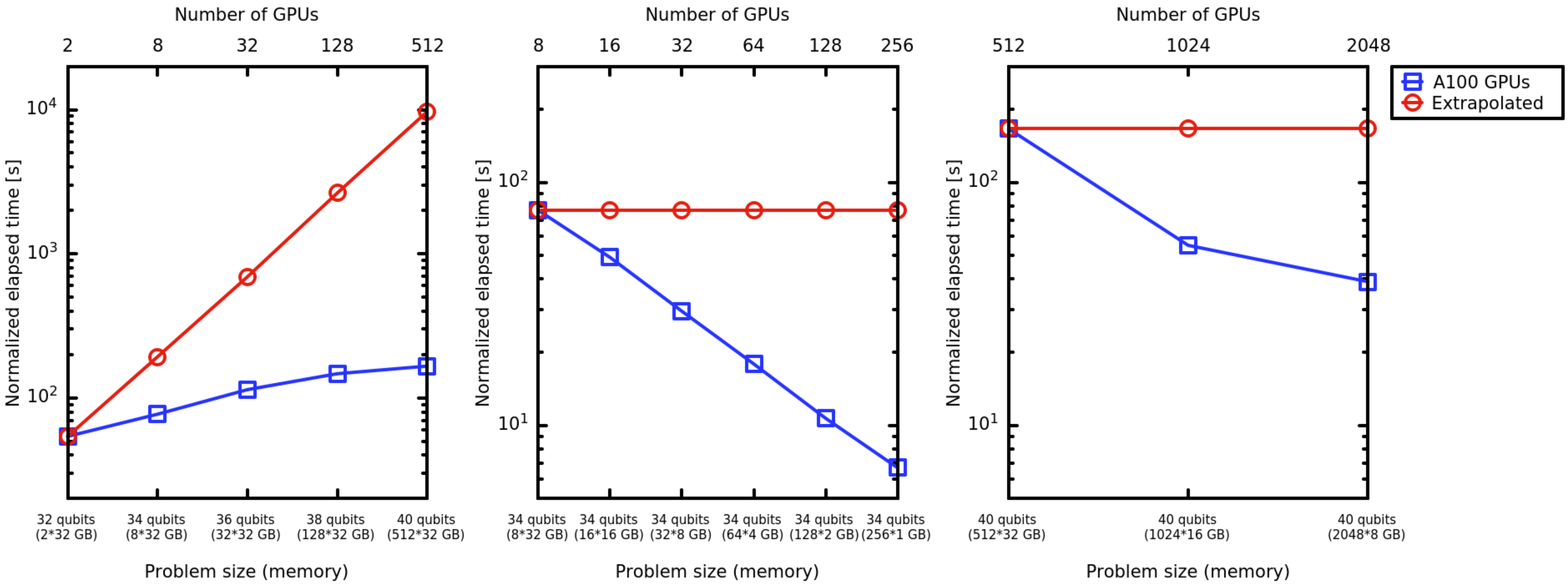
Why we can use it to benchmark JUWELS Booster

- Memory-intensive:
  - For 40 qubits: $2^{40}\ \psi'\mathrm{s}$ = 16 TiB memory
- Network-intensive:
  - Each global single-qubit gate requires transferring **one half** of all memory
  - For 40 qubits: $2^{40}/2\ \psi'\mathrm{s}$ = 8 TiB transfer
- High GPU utilization
  - For 40 qubits:
    - 32 GiB on 512 GPUs
    - 16 GiB on 1024 GPUs
    - 8 GiB on 2048 GPUs
- Using **GPUs** to simulate **universal QPUs**

QPU = Quantum Processing Unit

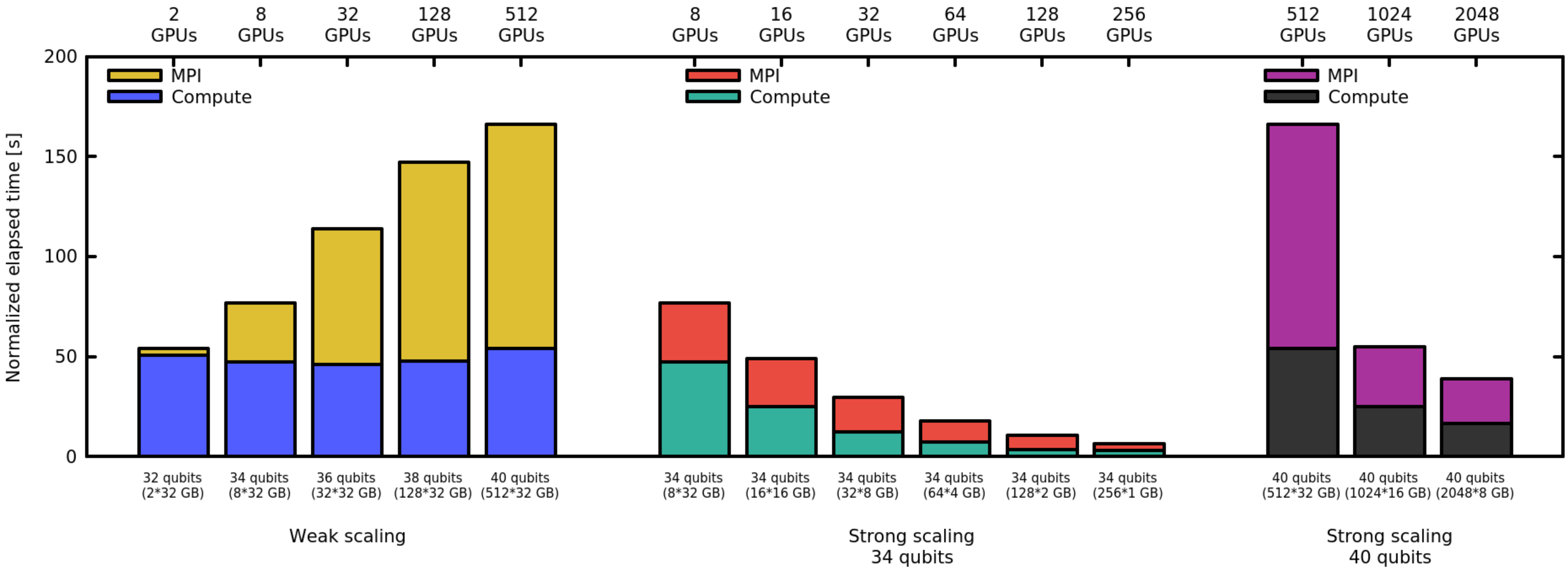
JUQCS-G

Weak and strong scaling results



JUQCS-G

Weak and strong scaling results: MPI vs. Compute Time

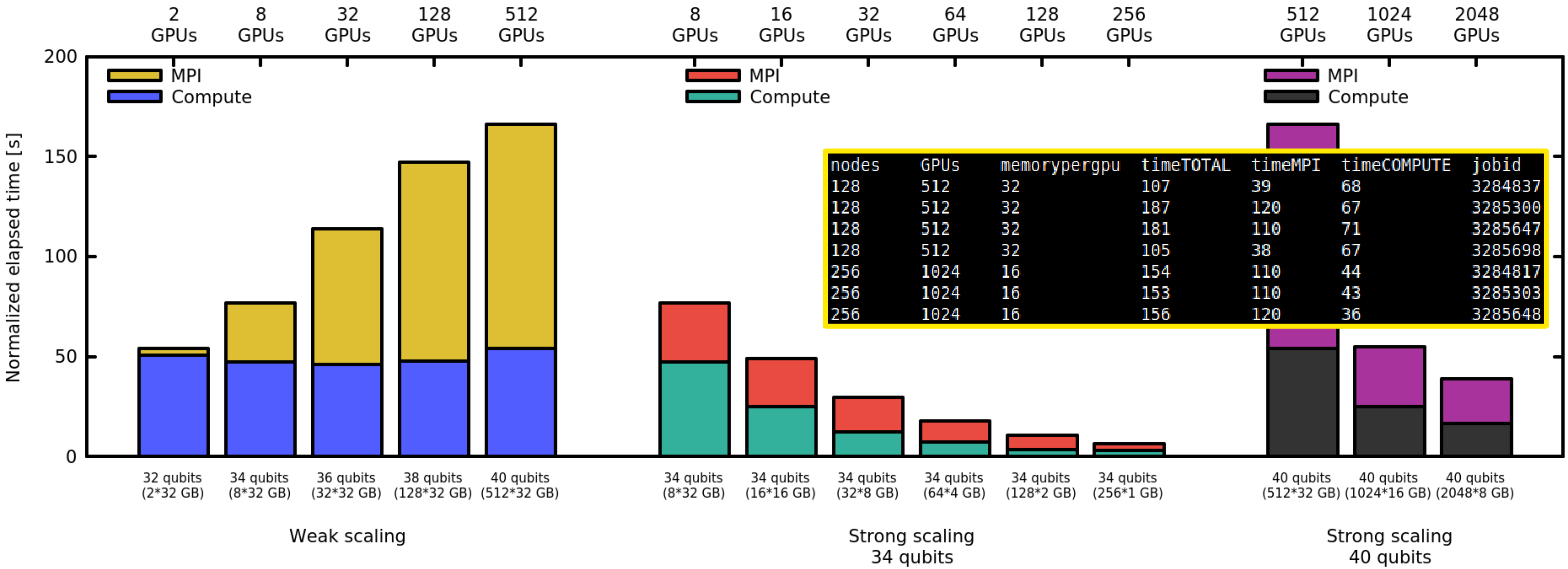


- **Speedup** (compute time per node)**:** JUWELS Cluster → JUWELS GPUs (V100): 10
- **Speedup** (compute time per node)**:** JUWELS GPUs → JUWELS Booster (A100): 2 – 3

JÜLICH Forschungszentrum

JUQCS-G

Weak and strong scaling results: MPI vs. Compute Time



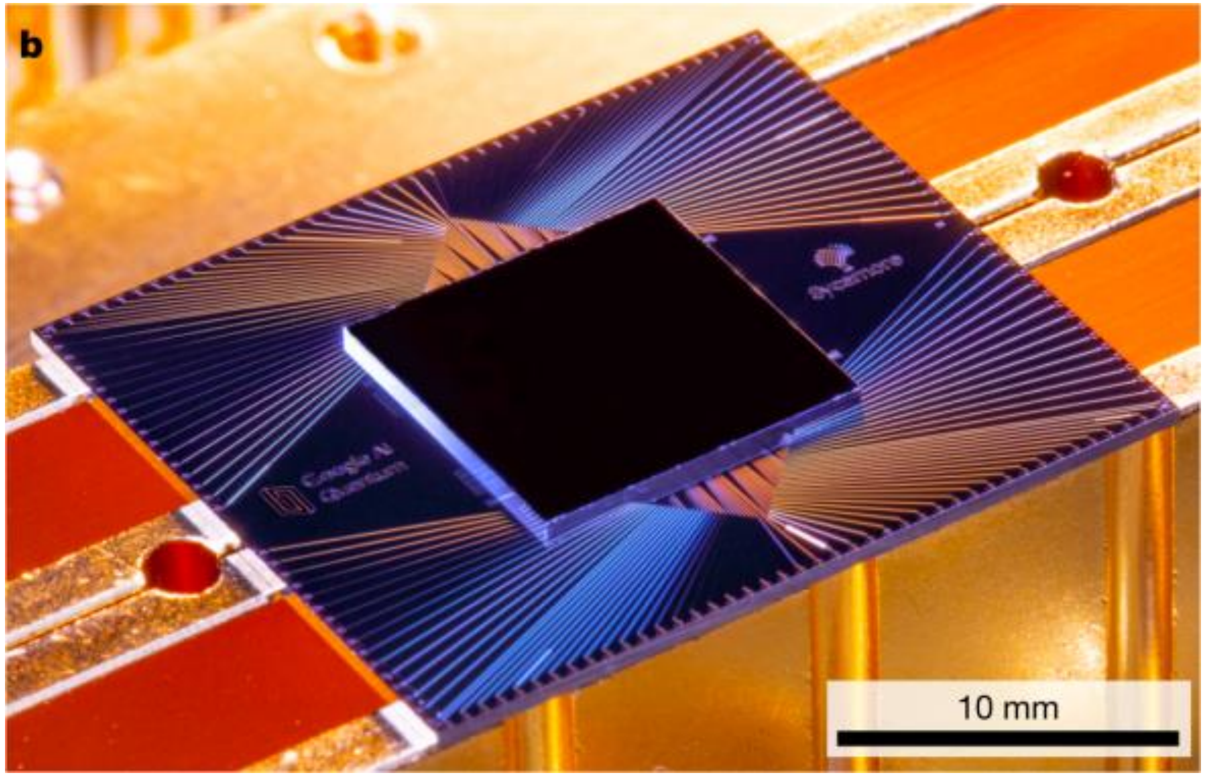
nodes	GPUs	memorypergpu	timeTOTAL	timeMPI	timeCOMPUTE	jobid
128	512	32	107	39	68	3284837
128	512	32	187	120	67	3285300
128	512	32	181	110	71	3285647
128	512	32	105	38	67	3285698
256	1024	16	154	110	44	3284817
256	1024	16	153	110	43	3285303
256	1024	16	156	120	36	3285648

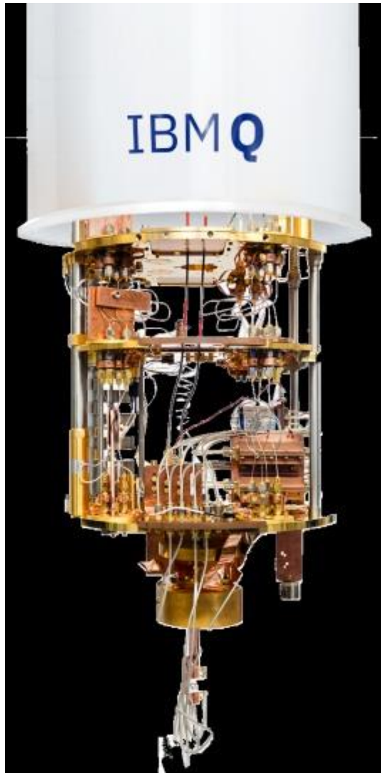
- **Speedup** (compute time per node)**:** JUWELS Cluster → JUWELS GPUs (V100): 10
- **Speedup** (compute time per node)**:** JUWELS GPUs → JUWELS Booster (A100): 2 – 3

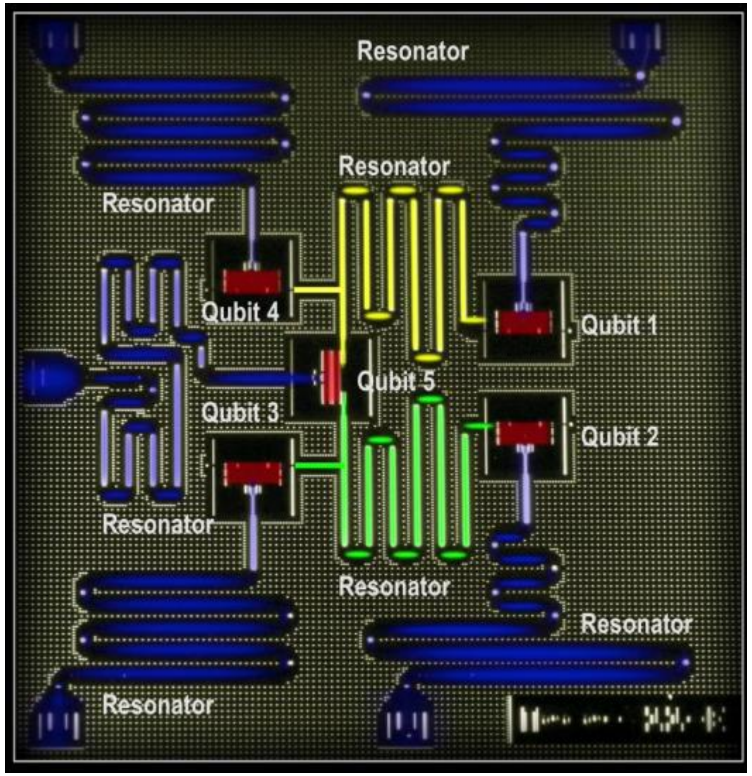
JÜLICH Forschungszentrum

JUQMES: QUANTUM MASTER EQUATION SIMULATOR

Simulating physical realizations of quantum computers







JÜLICH Forschungszentrum

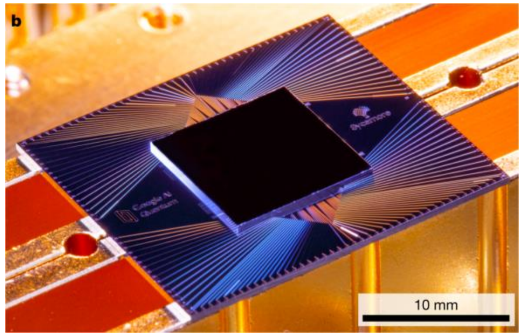
JUQMES: QUANTUM MASTER EQUATION SIMULATOR

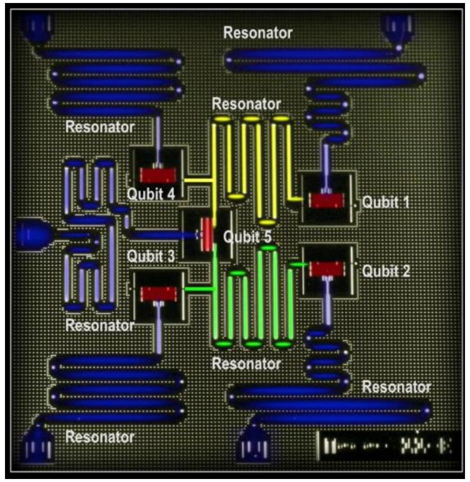
Simulating physical realizations of quantum computers

- Physical realization: Solve Schrödinger / Master Equation

$$\frac{\partial}{\partial t}|\psi\rangle = -iH|\psi\rangle \quad \text{or} \quad \frac{\partial}{\partial t}\rho = -i[H,\rho] + \mathcal{D}[\rho]$$

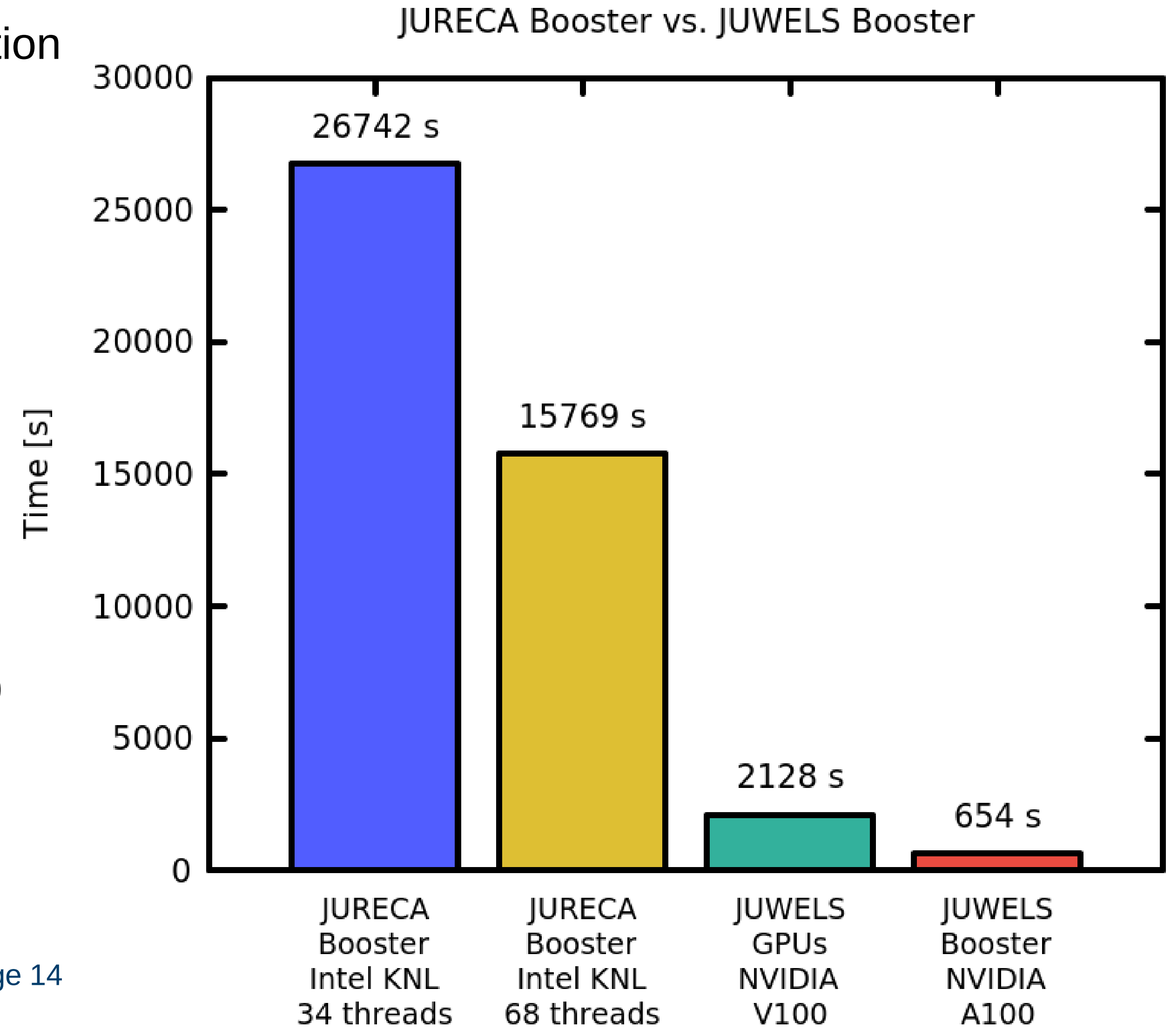
- Similar SPMV updates **but:**
  - More than 2 states per qubit





  - More complicated sparse matrices (sin, cos, exp, ...)
  - Many updates per time step
- Very computation-intensive (memory “only” 2 GiB)

  → Useful to measure single-GPU performance



CONCLUSION

JÜLICH SUPERCOMPUTING CENTRE

- Simulating QCs is a versatile approach to benchmark supercomputers
  - Memory-, network-, and computation-intensive
- Huge speedup on GPUs compared to CPU-based simulators



- JUWELS Booster is awesome ☺

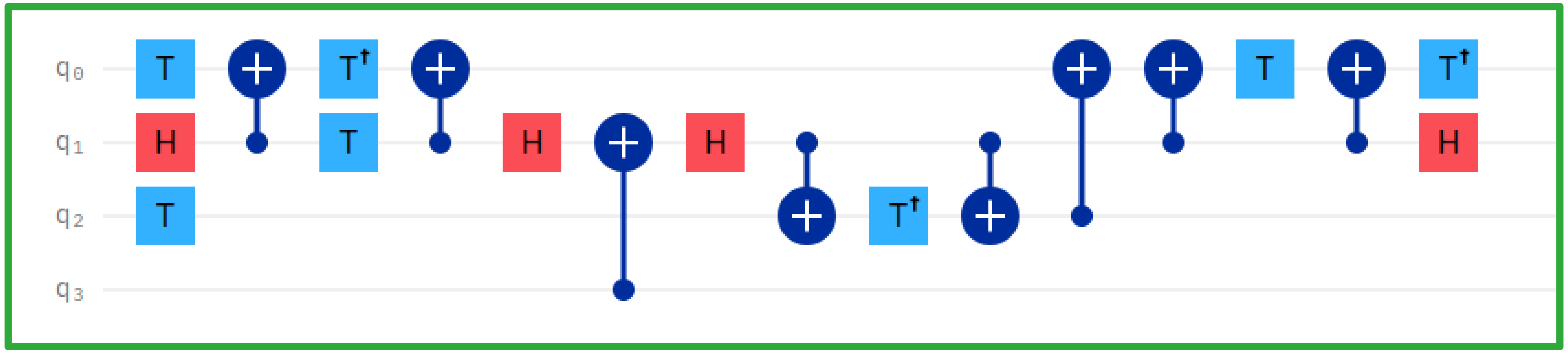
THANK YOU VERY MUCH!

- More information / references:
  - **MPI communication scheme:** De Raedt et al., Comp. Phys. Commun. 176, 121 (2007)
  - **JUQCS:** De Raedt et al., Comp. Phys. Commun. 237, 41 (2019)
  - **Quantum supremacy with JUQCS:** Arute et al., Nature 574, 505 (2019)
  - **Benchmarking supercomputers with JUQCS:** Willsch et al., NIC Series 50, 255 (2020)
  - **Benchmarks on JUWELS Booster and others:** Willsch et al., in preparation (2021)

BACKUP: QUANTUM COMPUTING

Ideal gate-based quantum computing

- In particular, what does **this quantum circuit** do?



- 2-qubit adder

$$|q_3q_2\rangle|q_1q_0\rangle \mapsto |q_3q_2\rangle|q_3q_2 + q_1q_0\rangle$$

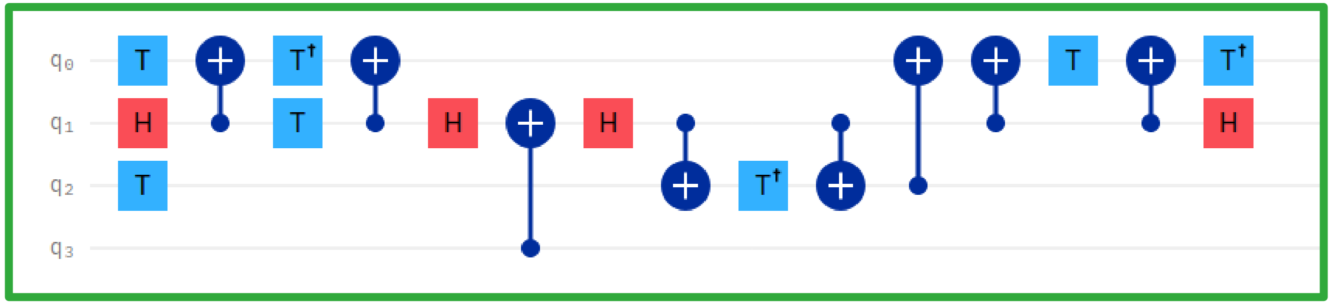
  - e.g. $|2\rangle|1\rangle \mapsto |2\rangle|3\rangle$
  - but also **superpositions**:

$$|2\rangle\frac{|0\rangle + |1\rangle + |2\rangle}{\sqrt{3}} \mapsto |2\rangle\frac{|2\rangle + |3\rangle + |0\rangle}{\sqrt{3}}$$

$$\begin{pmatrix} \vdots \\ \psi_{1000} = 1/\sqrt{3} \\ \psi_{1001} = 1/\sqrt{3} \\ \psi_{1010} = 1/\sqrt{3} \\ \psi_{1011} = 0 \\ \vdots \end{pmatrix} \mapsto \begin{pmatrix} \vdots \\ \psi_{1000} = 1/\sqrt{3} \\ \psi_{1001} = 0 \\ \psi_{1010} = 1/\sqrt{3} \\ \psi_{1011} = 1/\sqrt{3} \\ \vdots \end{pmatrix}$$

BACKUP

MPI Communication Scheme: Two-qubit gates



- General two-qubit gates:
  - 2 global, 0 local: exchange ¾
  - 1 global, 1 local: exchange ½
  - 0 global, 2 local: exchange 0
  - Then: each MPI rank does 4x4 update locally
- CNOT gate: ¾ cases: no communication necessary (in principle)
  - "2 global": no exchange, relabel MPI rank *10* ←→ *11*
  - "2 local": each MPI rank swaps *10* ←→ *11* locally (½ of all amplitudes)
  - "C global, T local": each MPI rank with control=1 (½ of all ranks) swaps *10* ←→*11* locally (½ of all ampl.)
  - Only in case "T global, C local": exchange ½ of all amplitudes
- CPHASE gate: each MPI rank multiplies *11* by -1 locally (¾ of all amplitudes)
- Toffoli: similar, in many cases no communication necessary
- For benchmarking purposes: do the exchange whenever one qubit in a multi-qubit gate is global

$$\sigma_1 = \begin{pmatrix} 3 & 2 & 1 & 0 \\ 3 & 2 & 1 & 0 \end{pmatrix}$$

	00	**01**	**10**	**11**
00	$a(0000)$	$a(0100)$	$a(1000)$	$a(1100)$
01	$a(0001)$	$a(0101)$	$a(1001)$	$a(1101)$
10	$a(0010)$	$a(0110)$	$a(1010)$	$a(1110)$
11	$a(0011)$	$a(0111)$	$a(1011)$	$a(1111)$