Jülich Universal Quantum Computer Simulator (JUQCS)
Past, present and future

Hans De Raedt

Jülich Supercomputing Centre, Forschungszentrum Jülich

and

Zernike Institute for Advanced Materials, University of Groningen

JSC – NVIDIA Lab 10 year anniversary, June 2022

The past: why simulating quantum computers?

Quantum-mechanical computers, if they can be constructed, will do things no ordinary computer can
S. Lloyd, Scientific American *October* 1995, 47.

To surpass classical simulations of quantum systems, however, would require only tens of bits followed for tens of steps, a goal that is more attainable (but has not been yet). And to use quantum logic to create strange, multiparticle quantum states and to explore their properties is a goal that lies in our current grasp.

What are they  about ?

1993 → 1995 → …

Solution of the time-dependent Schrödinger equation for two-dimensional spin-½ (= qubit nowadays) Heisenberg systems,
P. de Vries and H. De Raedt, Physical Review B 47, 1 (1993)
…
The algorithm has been implemented on a **CRAY-YMP** with a performance of 55 Mflops . A typical run for the isotropic Heisenberg model with 24 spins (= 24 qubits) and 1530 time steps takes about 10 hours on the **CRAY-YMP** (less on my current notebook).

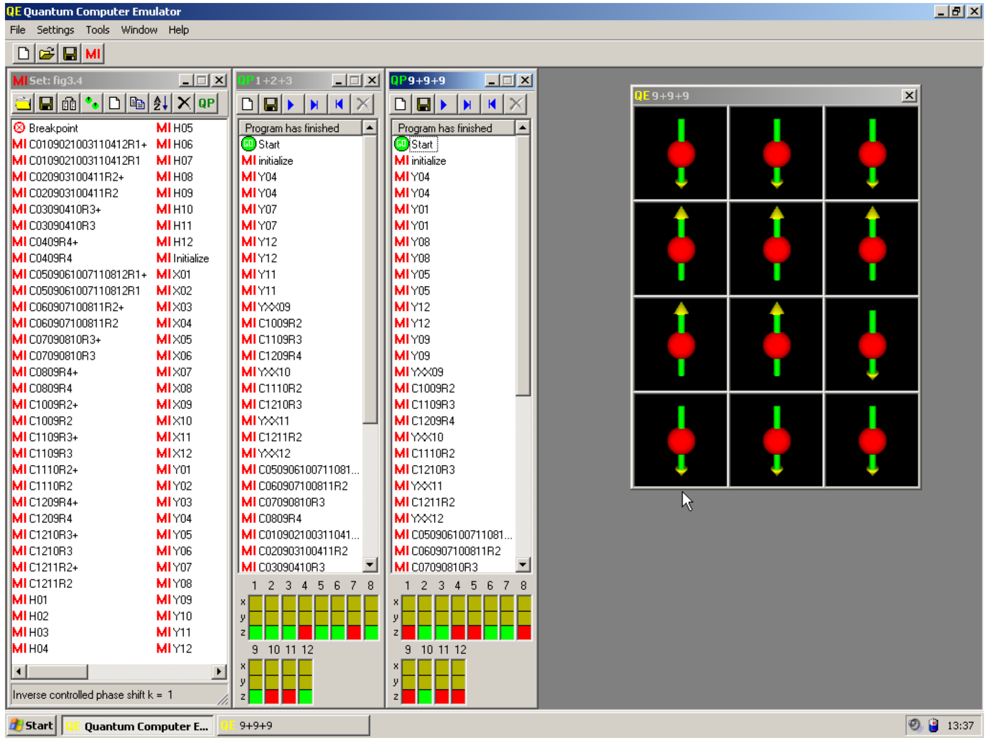
First steps: an emulator of real QC hardware

1993 → 1995 → 2000 → …

Quantum Computer Emulator*

H. De Raedt, A.H. Hams, K. Michielsen, and K. De Raedt,
Computer Physics Communications 132, 1 (2000)

A Quantum Computer Emulator for a generic, general purpose gated-based quantum computer, consisting of a graphical user interface to program and control the simulator of generic physical model of real quantum computers (up to 16 qubits, limited by graphics)



*Fully operational as a [VM VirtualBox](#) appliance (Win98 and Win XP)

https://www.compphys.org/QCE/

Take a step back: simulate large ideal universal quantum computers

1993 | 1995 | 2000 | 2007 | | | …

Massively parallel quantum computer simulator
K. De Raedt, K. Michielsen, H. De Raedt, B. Trieu, G. Arnold, M. Richter, Th. Lippert, H. Watanabe, N. Ito, Computer Physics Communications 176, 121 (2007).

Portable MPI/OpenMP software to simulate universal quantum computers on massive parallel computers.
Test of the software by running various quantum algorithms on IBM BlueGene/L (2005 Univ. Groningen), IBM Regatta p690+, Hitachi SR11000/J1, Cray X1E, SGI Altix 3700 and on clusters of PCs running Windows XP.
Simulate universal quantum computers with up to 36 qubits (1 TiB), using up to 4096 processor.

https://www.hpcwire.com/2010/06/28/quantum_computer_simulation_new_world_record_on_jugene/ : 42 qubits

Our main motivation: develop and test software components to speed up the simulations of physical models of quantum computers/annealers and quantum spin-1/2 systems in general (implemented)

Make noise…

1993 | 1995 | 2000 | 2007 | 2019 | …

Massively parallel quantum computer simulator, eleven years later
H. De Raedt, F. Jin, D. Willsch, M. Willsch, N. Yoshioka, N. Ito, S. Yuan, and K. Michielsen,
Computer Physics Communications 237, 47 (2019).

Simulator exhibits close-to ideal weak-scaling behavior on the Sunway TaihuLight, on the K computer, on an IBM Blue Gene/Q, and somewhat less ideal scaling on JUWELS and JURECA (Xeon based clusters). Results for simple quantum circuits and Shor's factorization algorithm on quantum computers containing up to 46 qubits and 48 qubits (with reduced precision).

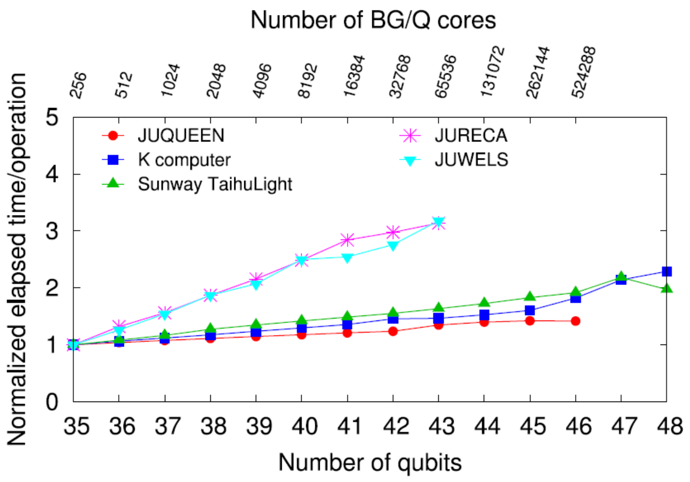
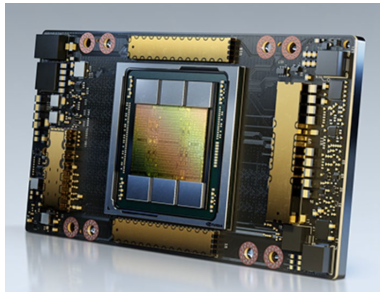


Fig. 4. The elapsed time per gate operation (normalized by the values 2.7 s (JUQUEEN), 3.8 s (K), 19.9 s (Sunway TaihuLight), 2.4 s (JURECA), and 2.2 s (JUWELS) to run the 35 qubits circuit) as a function of the number of qubits, obtained by JUQCS-A executing a Hadamard operation on qubit 0 and the sequence (CNOT 0 1), (CNOT 1 2), …, (CNOT N-2, N-1). This weak scaling plot shows that JUQCS-A beats the exponential scaling of the computational work by doubling the size of the machine with each added qubit.

https://www.hpcwire.com/2017/12/18/world-record-quantum-computer-46-qubits-simulated/

May 2020: NVIDIA A100 Tensor Core GPU, 40 GiB per ⇔31 qubits ☺, August 2020 JUWELS booster Early Access ...



1993 → 1995 → 2000 → 2007 → 2019 → **2021** ...

GPU-accelerated simulations of quantum annealing and the quantum approximate optimization algorithm
D. Willsch, M. Willsch, F. Jin, K. Michielsen, and H. De Raedt
Computer Physics Communications 278, 108411 (2022)

- Benchmarks + applications to several quantum algorithms, up to 42 qubits
- Large-scale applications using the GPU-accelerated version of the massively parallel Jülich universal quantum computer simulator (JUQCS–G)...

Table 1
Comparison of the GPU-based simulator JUQCS–G (first row) and the CPU-based simulator JUQCS–E [1] (second to last row) for the largest systems using the Hadamard benchmark circuits $(H^{\otimes N})^{11}$. The time t_{total} is the run time spent for the total simulation, normalized by the number of gates with respect to the 32-qubit case (see Eq. (3)). The time t_{MPI} is the elapsed time for communication plus the elapsed time to prepare and postprocess MPI buffers. JUQCS–E uses all cores of the CPUs on each node.

qubits	nodes	processes	hardware	normal.	t_{total} [s]	t_{MPI} [s]
42	256	1024	GPU	1.31	149.4	122.1
42	256	2048	CPU	1.31	2632.4	1297.7
42	512	4096	CPU	1.31	1500.4	763.4
43	512	4096	CPU	1.34	2714.4	1343.3

- A100 GPUs are about a factor 18 faster than JUWELS **booster CPUs**
- MPI communication is a dominating factor
- Amount of memory is the limiting factor

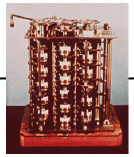
Quantum computing crash course: bits and qubits

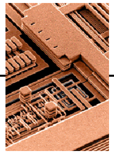
- Bit: 0 or (exclusive) 1
- Elementary operations:

  0 → 0 , 0 → 1

  1 → 0 , 1 → 1

  - Boolean algebra
- Read out:
  - Either 0 or 1
- Mechanical, electronic, … , realizations
  - Logical operation does not depend on realization





- Qubit:

$$\Psi = \begin{pmatrix} a_0 \\ a_1 \end{pmatrix} \quad \begin{matrix} \leftarrow \text{state } 0 \\ \leftarrow \text{state } 1 \end{matrix}$$

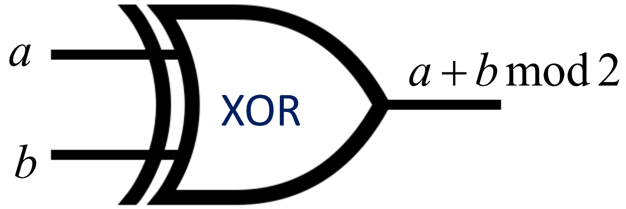
- Elementary operations: $\Psi_{\text{out}} = \mathbf{U}\,\Psi_{\text{in}}$
- Linear algebra: (unitary matrix) * (vector)
- Read out: the probability to find machine in state 0 (1) is

$$P_i = |a_i|^2, i = 0,1$$

- Quantum theory (= imagination): nature executes matrix-vector multiplication in one step
  - Postulated intrinsic parallelism
  - In reality: logical operation depends on the physical realization

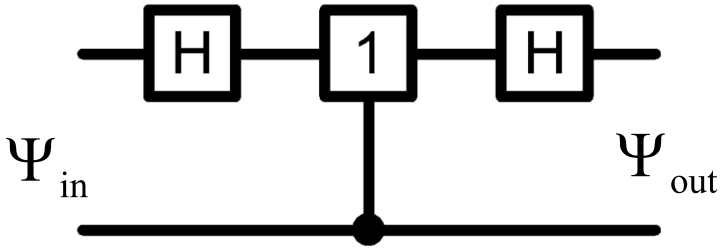
Computation: Add 2 (qu)bits

- 2-bit adder



- For given (a,b), XOR gate computes a+b mod 2
- 0+0 ➔ 0 , 1+0 ➔ 1
  0+1 ➔ 1 , 1+1 ➔ 0

- 2-qubit adder (CNOT)



$$\Psi_{\text{in}} = \begin{pmatrix} a_{00} \\ a_{10} \\ a_{01} \\ a_{11} \end{pmatrix} \quad \boxed{H} = \frac{1}{2}\begin{pmatrix} +1 & +1 & 0 & 0 \\ +1 & -1 & 0 & 0 \\ 0 & 0 & +1 & +1 \\ 0 & 0 & +1 & -1 \end{pmatrix} \quad \boxed{1} = \begin{pmatrix} +1 & 0 & 0 & 0 \\ 0 & +1 & 0 & 0 \\ 0 & 0 & +1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix}$$

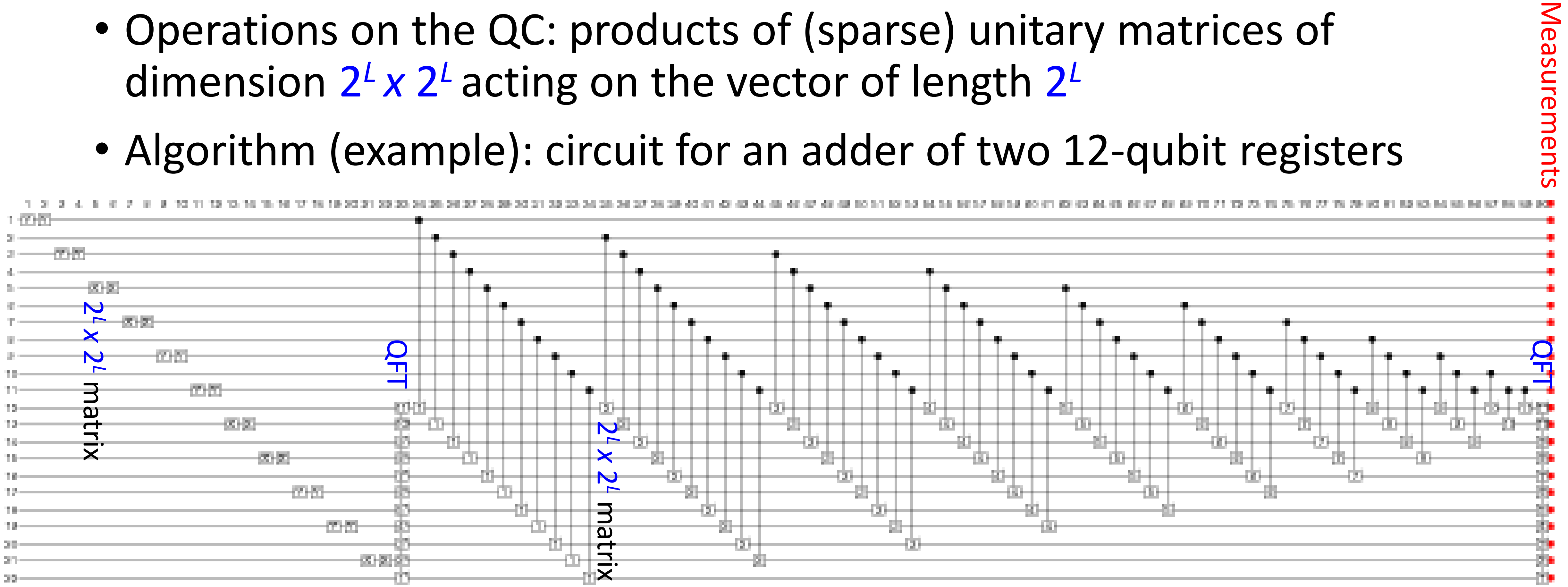
- Quantum theory: a QC performs such a matrix-vector multiplication in 1 step
- L qubits → vector has length 2^L
- L = 36 qubits → vector has length 68 719 476 736 ⇔ 1 TiB memory

What is a universal quantum computer?

- Any operation on a quantum computer can be decomposed in single-qubit (⇔NOT) and controlled-NOT (⇔ XOR) operations
- Length of the state vector increases exponentially with the number of qubits
    - L = 42 qubits ➔ needs at least 64 TiB memory
- Computation time to perform matrix-vector operations increases exponentially with the number of qubits and linearly with the number of gates in the quantum circuit

Universal quantum computer: summary

- State of an L-qubit QC: vector of length 2^L
- Operations on the QC: products of (sparse) unitary matrices of dimension $2^L \times 2^L$ acting on the vector of length 2^L
- Algorithm (example): circuit for an adder of two 12-qubit registers



A 3-qubit QC on a 2-processor computer with ~32 bytes/processor

- Example: CNOT on qubit 1 with control qubit 0

$$\begin{pmatrix} a_0 \\ a_3 \\ a_2 \\ a_1 \\ a_4 \\ a_7 \\ a_6 \\ a_5 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \\ a_7 \end{pmatrix} \qquad \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \end{pmatrix}$$

Proc. 0

Proc. 1

No communication between processors !

A 3-qubit QC on a 2-processor computer with ~32 bytes/processor

- Example: CNOT on qubit 2 with control qubit 0

$$\begin{pmatrix} a_0 \\ a_5 \\ a_2 \\ a_7 \\ a_4 \\ a_1 \\ a_6 \\ a_3 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \\ a_7 \end{pmatrix} \quad \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \end{pmatrix}$$

Proc. 0

Proc. 1

"Massive" communication between processors !

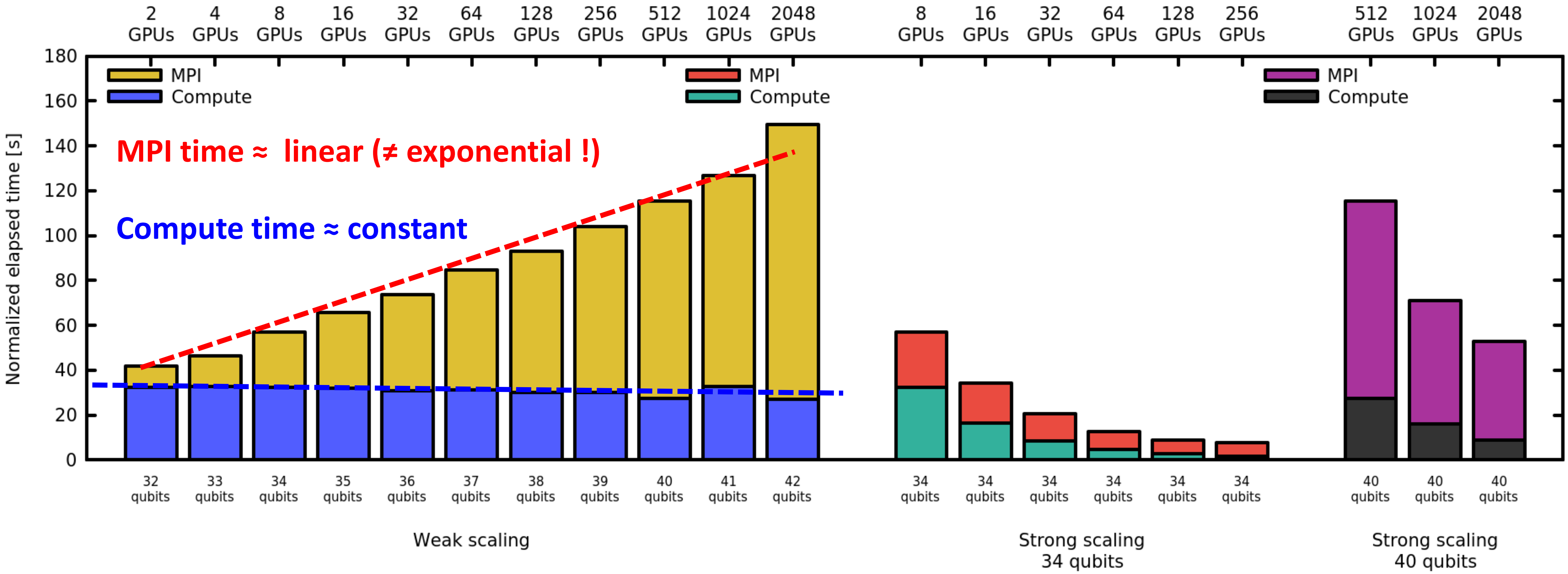
Minimize communication

- Solution 1: Transfer data back and forth
  - Advantage: Simple logic
  - Disadvantage: 2 data transfers per operation
- Solution 2: Transfer data and keep track of permutations of qubits
  - Advantage: Only 1 data transfer per operation
  - Disadvantage: Complicated logic

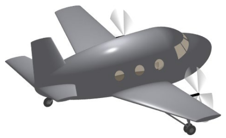
Performance

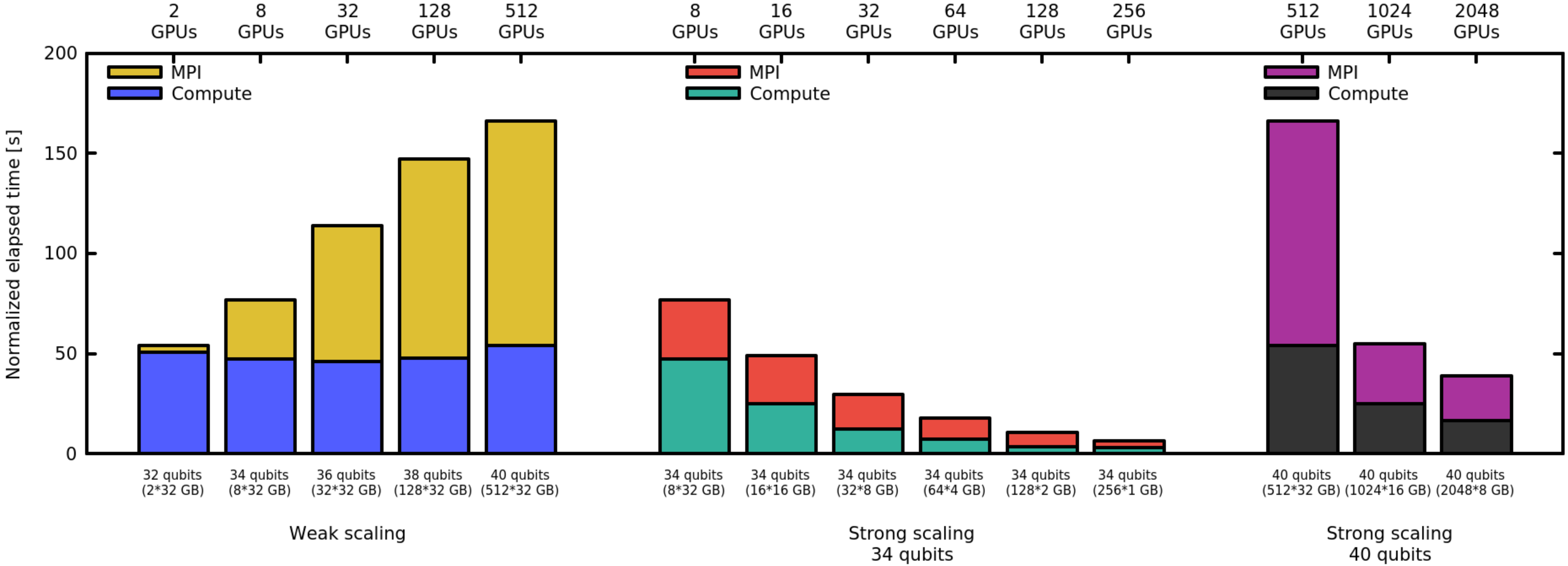
- Memory and CPU time increase exponentially with the problem size ⇔ exponentially with the number of qubits
  - In practice, it rarely makes sense to keep the problem size constant and increase the number of processing units
- Weak-scaling: If we add 1 qubit to the QC, we have to double the amount of memory and also double the number of CPUs
- Strong-scaling: keep memory constant and increase the number of processes

JUQCS-G(pu) benchmark: weak and strong scaling results for 11 Hadamard operations on each qubit*



***GPU-accelerated simulations of quantum annealing and the quantum approximate optimization algorithm,** D. Willsch, M. Willsch, F. Jin, K. Michielsen and H. De Raedt, Computer Physics Communications 278, 108411 (2022)

JUQCS-G(pu): application to a simplified version of  - tail assignment optimization*



***GPU-accelerated simulations of quantum annealing and the quantum approximate optimization algorithm,** D. Willsch, M. Willsch, F. Jin, K. Michielsen and H. De Raedt, Computer Physics Communications 278, 108411 (2022)

Future

- JUQCS-G: for applications with more than 31 qubits, communication
  - MPI via CPUs: slow
  - CUDA-aware MPI: ok
  - NVLINK (cudaMemcpyAsync) beyond more than 4 GPUs per node seems promising
- JUQCS-G single GPU performance
  - Bound by speed of memory access
  - Sebastian Schulz: can be improved a lot by clever GPU programming
- Back to physics:
  - Madita Willsch: GPU-based simulator of physical models of quantum annealers
    → next talk
  - Me: GPU version of generic quantum spin dynamics simulator →postponed ☺

THANK YOU

Jiri Kraus (NVIDIA), Markus Hrywniak (NVIDIA), Andreas Herten (JSC), and Damian Alvarez (JSC) for helping us out every time we got stuck

THANK YOU