Simulating Quantum Computers on Supercomputers

Madita Willsch[1,2], Dennis Willsch[1], Fengping Jin[1], Hans De Raedt[1,3], Kristel Michielsen[1,2,4]

[1] *Institute for Advanced Simulation, Jülich Supercomputing Centre, Forschungszentrum Jülich, 52425 Jülich, Germany*

[2] *AIDAS, 52425 Jülich, Germany*

[3] *Zernike Institute for Advanced Materials, University of Groningen, Nijenborgh 4, 9747 AG Groningen, The Netherlands*

[4] *RWTH Aachen University, 52056 Aachen, Germany*

Practical Aspects of Quantum Computer Simulations

Considerations

- N-qubit wave function:

$$|\psi\rangle = a_{0\ldots000}|0\ldots000\rangle + a_{0\ldots001}|0\ldots001\rangle + a_{0\ldots010}|0\ldots010\rangle + \cdots + a_{1\ldots111}|1\ldots111\rangle$$

$$= \begin{pmatrix} a_{0\ldots000} \\ a_{0\ldots001} \\ a_{0\ldots010} \\ \vdots \\ a_{1\ldots111} \end{pmatrix} \Bigg\} 2^N \text{ complex coefficients}$$

  - ➔ Storage requires (complex double precision) $16 \times 2^N = 2^{N+4}$ Bytes of RAM
- Large-scale simulations require handling of distributed memory
  - ➔ Local qubits: corresponding amplitudes stored on the same GPU
  - ➔ Global qubits: amplitudes distributed over different GPUs
  - ➔ MPI communication for data transfer

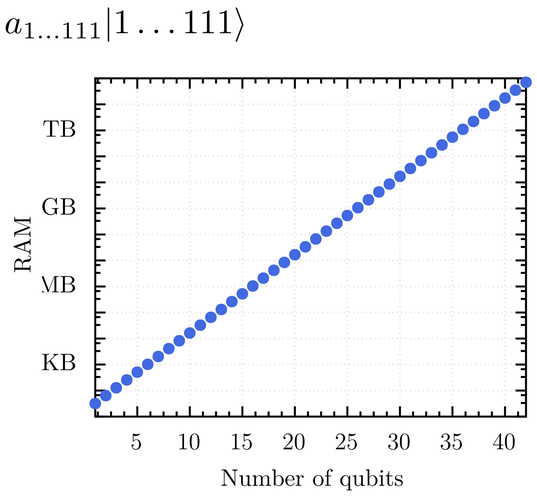

Fig. 1: RAM required to store the state vector.

Implementation

- 1-, 2- and 3-qubit gates require 2-, 4- and 8-component updates of state vector [1,2]
- For quantum annealing (QA), solve time-dependent Schrödinger equation via time stepping [3]
  - ➔ Efficient approximation of time-evolution operator yields 1-, 2- and 4-component updates
- Efficient MPI communication scheme [1,2]:
  - ➔ Swap local and global qubit by exchanging half of the amplitudes between pairs of GPUs
  - ➔ Keep track of local and global qubit indices instead of transferring data back and forth

Jülich Universal Quantum Computer Simulator - JUQCS

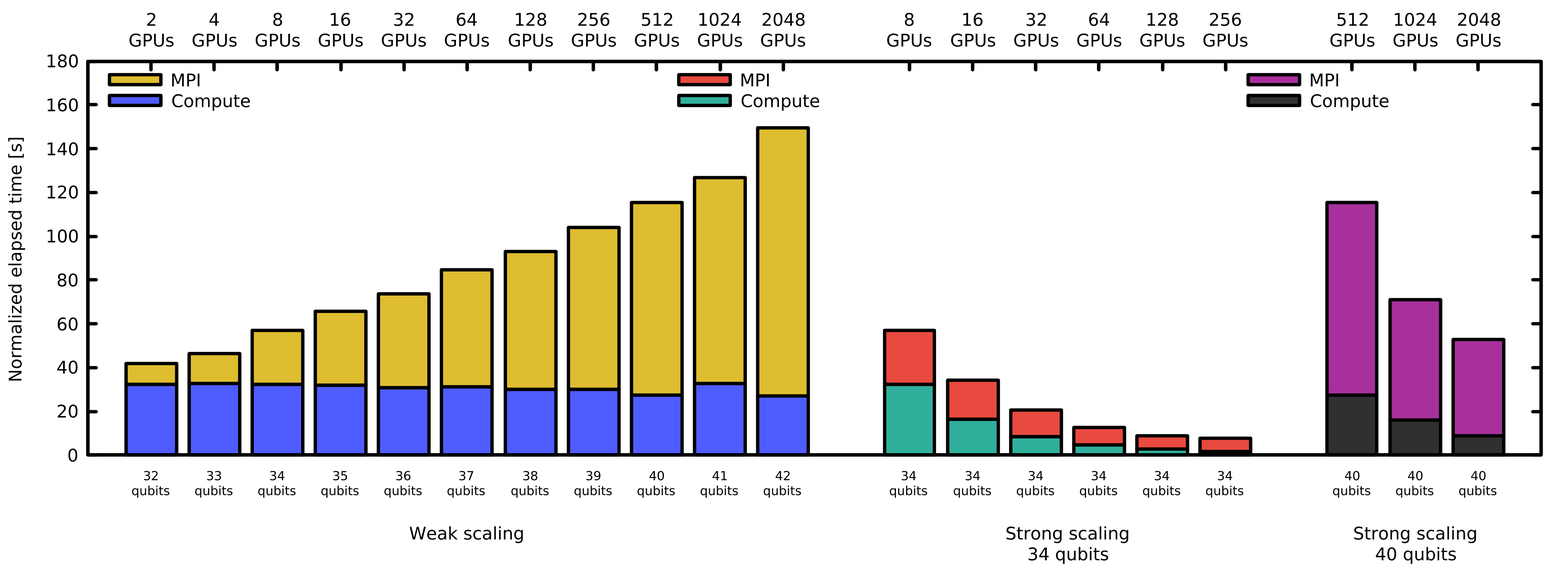

Fig. 2: Performance benchmark of JUQCS on JUWELS Booster [4]. Weak scaling (left): The number of GPUs is doubled with each added qubit. Strong scaling (middle and right): Number of qubits is constant but number of GPUs is increased.

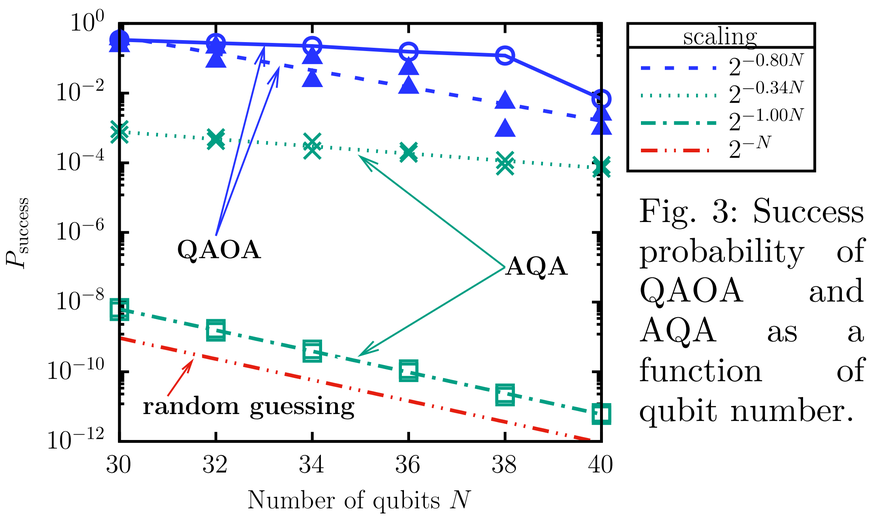

Fig. 3: Success probability of QAOA and AQA as a function of qubit number.

- High-performance quantum computer simulator; very good scaling
- Study quantum algorithms, e.g. QAOA [4]:
  - Initialize variational parameters according to discretized quantum annealing schedule (AQA [4])
    - ➔ Scaling with system size depends on choice of discretization parameters (but still exponential)
  - Optimize variational parameters for a "small" instance and reuse
  - Optimize variational parameters for each instance (standard QAOA) → costly, and optimization can get stuck

JUelich Quantum Annealing Simulator - JUQAS

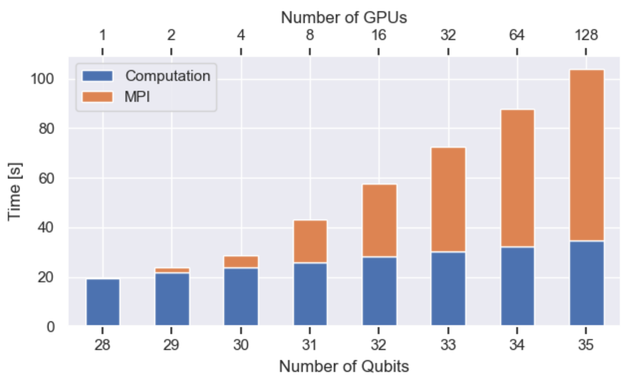

Fig. 3: Performance benchmark of JUQAS.

- Overall very good performance
- Can compare ideal QA to existing quantum annealers
  - ➔ Depending on problem instance, results can be very different
- Study various influences for ideal QA (annealing schedule, embedding, ...)

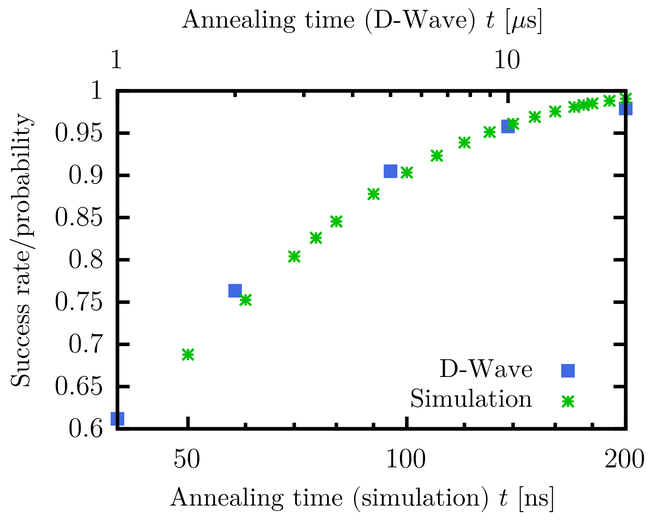

Fig. 4: Comparison of success rates/probabilities of (simulated) quantum annealing and the D-Wave DW_2000Q_6 quantum annealer.

References

[1] K. De Raedt et al., *Massively parallel quantum computer simulator*, Comput. Phys. Commun. 176, 121 (2007)

[2] H. De Raedt et al., *Massively parallel quantum computer simulator, eleven years later*, Comput. Phys. Commun. 237, 47 (2019)

[3] H. De Raedt, *Product formula algorithms for solving the time dependent Schrödinger equation*, Comput. Phys. Rep. 7, 1 (1987)

[4] D. Willsch et al., *GPU-accelerated simulations of quantum annealing and the quantum approximate optimization algorithm*, Comput. Phys. Commun. 278, 108411 (2022)

Acknowledgments

We acknowledge the Gauss Centre for Supercomputing e.V. for providing computing time on the Supercomputer JUWELS at JSC. D.W. and M.W. acknowledge support from the project JUNIQ that has received funding from BMBF and the Ministry of Culture and Science of NRW.