Chair for Design Automation
TUM School of Computation, Information and Technology
Technical University of Munich

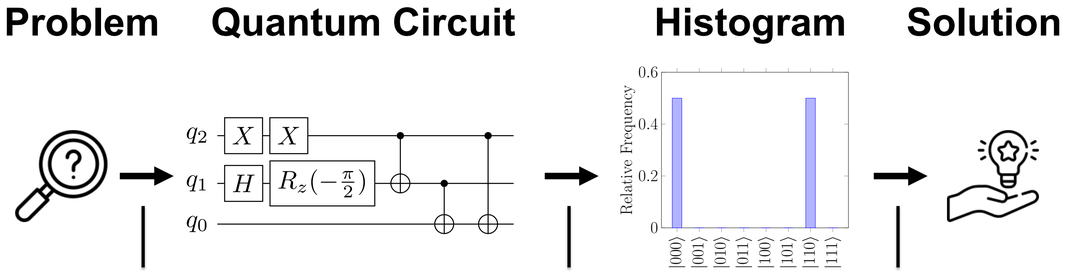
Supporting End-Users in Realizing Quantum Computing Applications

Nils Quetschlich, Lukas Burgholzer, and Robert Wille, Contact: nils.quetschlich@tum.de
http://www.cda.cit.tum.de/research/quantum

Abstract

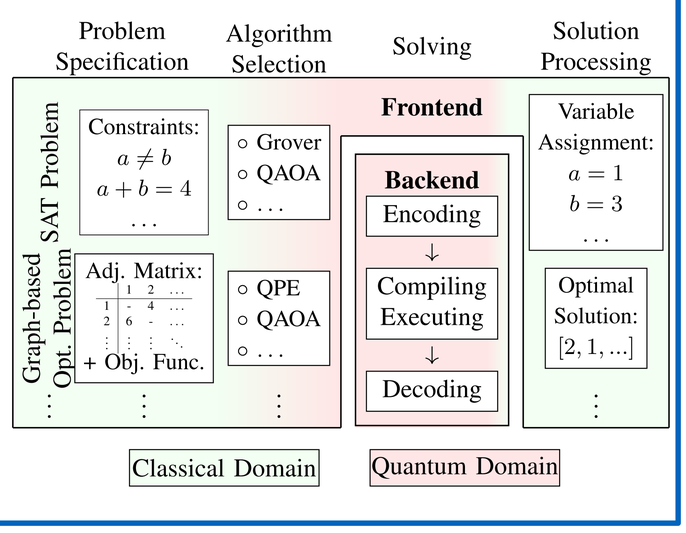
Realizing quantum computing applications requires (1) the selection of a suitable quantum algorithm, (2) the generation of a corresponding encoding, (3) the compilation/execution of the resulting quantum circuit, and (4) decoding the results. To date, these tasks still substantially rely on manual labour— creating a high entry barrier especially for end-users with little to no expertise in that domain. In our work, various methods and repositories are proposed to support end-users in conducting those tedious and error-prone steps. The resulting software is available as part of the *Munich Quantum Toolkit (MQT)*.

Background: Problem → Quantum Solution



MQT ProblemSolver [1]

- **Goal: Shielding end-users as much as possible from quantum**
- Automation of solving problems from various problem classes
- Providing the same interfaces as classical solvers



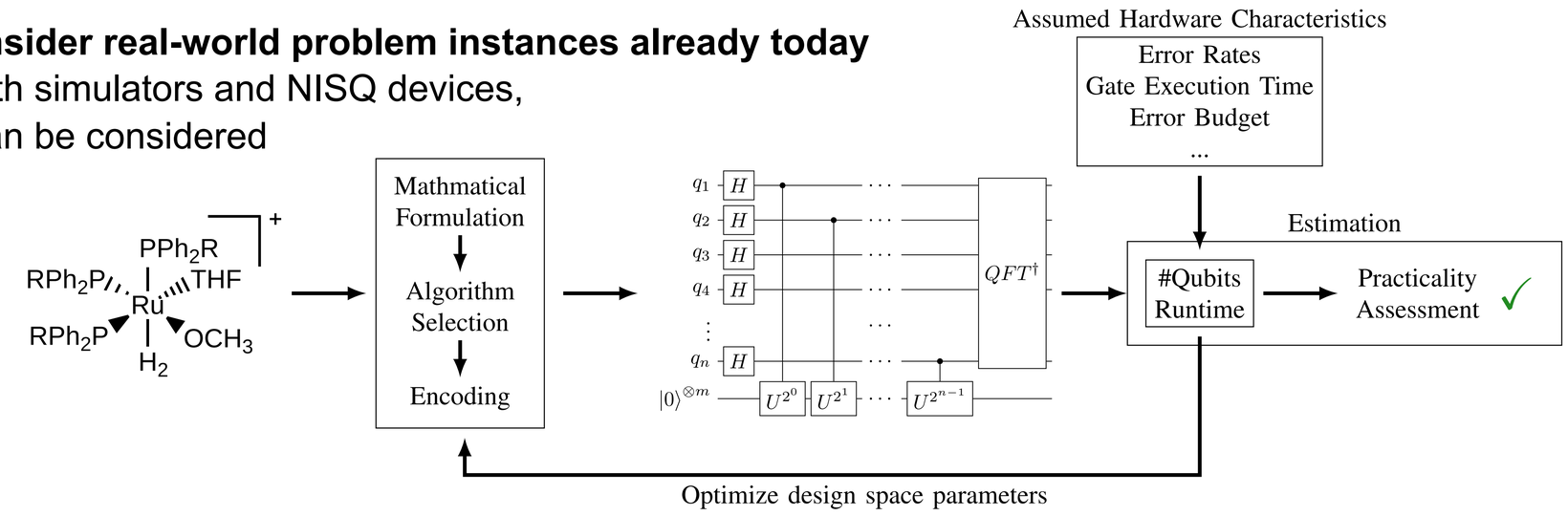
MQT Bench [2]

- **Goal: Increased comparability, reproducibility, and transparency**
- Huge benchmark suite with > 70,000 benchmarks of various algorithms on four abstraction levels
- To be used, e.g., to evaluate and compare software tools

Easy-to-use Website

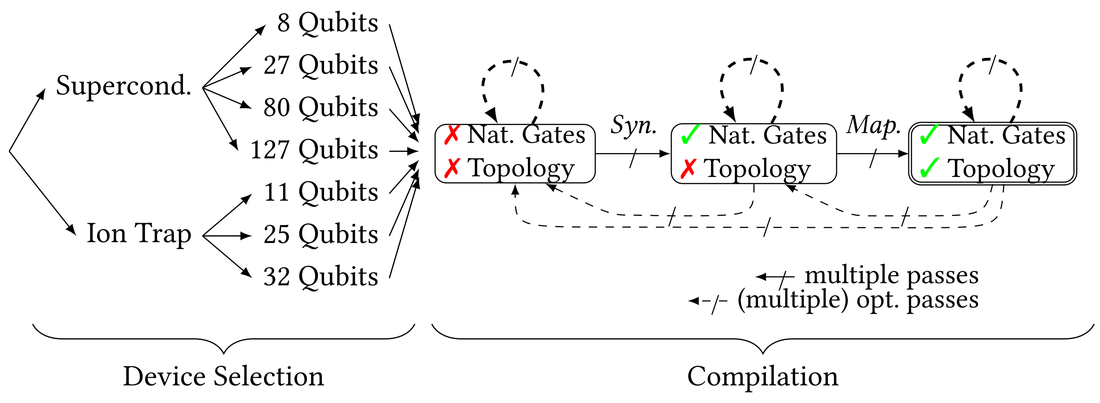
Extended MQT ProblemSolver Workflow Using Resource Estimation [3]

- **Goal: Allowing end-users to consider real-world problem instances already today**
- Due to the limited capacities of both simulators and NISQ devices, only toy-size problem instances can be considered
- Rather than executing a quantum circuit, its number of qubits and expected runtime can be estimated to explore possible optimizations of those real-world problem instances across the entire design space



MQT Predictor [4]

- **Goal: Guiding end-users through the compilation process**
- Executing an application realized in a quantum circuit requires the selection of a quantum technology, a respective device, and compilation options; often overwhelming end-users
- The MQT Predictor framework automatically makes those decisions for a given circuit using supervised machine learning [5] and reinforcement learning [6]



Selected References

[1] N. Quetschlich, L. Burgholzer, and R. Wille. Towards an Automated Framework for Realizing Quantum Computing Solutions. In *International Symposium on Multiple-Valued Logic (ISMVL)*. 2023.
[2] N. Quetschlich, L. Burgholzer, and R. Wille, *MQT Bench: Benchmarking software and design automation tools for quantum computing*. In *Quantum*. 2023.
[3] N. Quetschlich, M. Soeken, P. Murali und R. Wille, “Utilizing Resource Estimation for the Development of Quantum Computing Applications,” 2024. arXiv: 2402.12434.
[4] N. Quetschlich, L. Burgholzer und R. Wille, “MQT Predictor: Automatic Device Selection with Device-Specific Circuit Compilation for Quantum Computing,” 2023. arXiv: 2310.06889.
[5] N. Quetschlich, L. Burgholzer, and R. Wille. Predicting Good Quantum Circuit Compilation Options. In *International Conference on Quantum Software (QSW)*. 2023.
[6] N. Quetschlich, L. Burgholzer, and R. Wille. Compiler Optimization for Quantum Computing Using Reinforcement Learning. In Design Automation Conference (DAC). 2023.

Open-source Implementations

mqt-problemsolver
mqt-bench
mqt-predictor