Toolchains for Quantum Computing

Bettina Heim
Microsoft Research, QuArC
Programming a Quantum Computer

Quantum Teleportation

\[(|0\rangle + |1\rangle) \ldots \]

\[(|00\rangle + |11\rangle) \ldots \]

\[(|00\rangle + |11\rangle) \ldots \]

\[(|00\rangle + |11\rangle) \ldots \]
Programming a Quantum Computer

Quantum Teleportation

Quantum states:
\[ |\Phi_{ab}\rangle \]

Circuit diagram:

- MSG gate
- H gate
- X gate
- Z gate
Programming a Quantum Computer

Quantum Teleportation in a Steane7 Code

\[ a + b \]
Expectation for a software stack

Abstraction
- Hardware independent formulation of mathematical concepts
- Algorithm formulation on a logical level
- Encapsulation

Validation
- Resource requirements
- Correctness of the algorithm
- Verifiable behavior

Error Sources
- Algorithmic Errors
- Approximation Errors
- Hardware Errors

Resource Management
- Memory management
- Asynchronous execution
- Classical processing
- Hardware specific optimization

→ Hardware specifications
→ Classical/quantum coordination
→ Precision distribution
→ Available information

→ Context dependent dispatch
→ Performance metrics
→ Static vs. runtime
→ Heuristics
Quantum Software Framework

Development Environment

Quantum Libraries

User Code

Supporting Libraries

Compilation and Optimization

Compiler

Execution Model

Hardware Specifications

Classical Compiler

Executable Binary

Runtime Environment

Executable Binary

Execution Platform

Tools for verification and benchmarking of quantum algorithms

Quantum Processor

Classical Processor
Compilation process

- Library: variations for each quantum (sub-)routine
- User code defining an algorithm, optimization of algorithmic errors
- Dependency model of subroutines, constant folding, optimization of the overall error
- Subroutine dispatch based on hardware, erasure of subroutine boundaries
- Exploiting (de-facto) commutation relations to reduce algorithm cost
- Optimization of synthesis errors
- Determine state distillation routines (possibly dynamic)
- Physical layout, “routing” (dynamic and/or look-up)
- Applying or tracking error correction, communication for runtime compilation
- Choice of error correction code

Compiler
Expectation for a software stack

- **What is the relevant information?**
- **How do we obtain the necessary information?**
- **How do we represent that information?**
- **How do we use that information?**
- **How do we generalize this process?**
WWW.MICROSOFT.COM/QUANTUM

Learn more about our approach
Get started with Quantum
Invent the future