QPack: An application -oriented benchmark for NISQ computers

TQCI seminar, 11 May 2023 Thales TRT – Palaiseau

Matthias Möller, Zaid Al-Ars, Koen Mesman, Huub Donkers

TU Delft

About

Matthias Möller

Associate Professor of Numerical Analysis
Department of Applied Mathematics

Zaid Al-Ars

Associate Professor at the Computer Engineering Lab
Department of Quantum & Computer Engineering

Koen Mesmann

PhD candidate – QAIMS lab

Huub Donkers

former MSc student – QCE

TU Delft

About

Vision

QPUs as special-purpose hardware accelerators in future high-performance computing systems

Research

NISQ & FTQ algorithms, high-level quantum programming SDKs, quantum benchmarks

TU Delft

QPack benchmark

Motivation

Variety of quantum benchmarks but lack of an accepted standard, e.g., "QLINPACK"

Goal

Unbiased, vendor- and qubit-technology n… benchmark for NISQ computers

Design criteria

- Hardware agnostic implementation (write-once-run-anywhere)
- Variety of application-oriented scalable test cases
- Multiple metrics – single score

TU Delft

LibKet

application developer

algorithm developer

"we"

one-API high-level quantum programming SDKs for C, C++, and Python

building block layer: quantum primitives, NISQ algorithms

abstraction layer: filters, gates, and device-specific features

embedded Python engine

Atos QLM | Cirq | IBM-Q | IonQ | Rigetti | YOURS?

C++ engine

OpenQL | QuEST | QX | YOURS?

TU Delft

Möller, M. and Schalkers, M. (2020): LibKet – A Cross-Platform Programming Framework for Quantum-Accelerated Scientific Computing. ICCS 2020. Lecture Notes in Computer Science

LibKet – by example

```
#include <LibKet.hpp>

using namespace LibKet;
using namespace LibKet::circuits;

auto expr = measure(qft(init()));

try {
  QDevice<QDeviceType::qi_26_simulator, 6> qpu; qpu(expr);
  auto result = qpu.eval(1024);

  QInfo << result << std::endl;
  QInfo << "job ID     : " << qpu.get<QResultType::id>(result)        << std::endl;
  QInfo << "best       : " << qpu.get<QResultType::best>(result)      << std::endl;
  QInfo << "histogram  : " << qpu.get<QResultType::histogram>(result) << std::endl;
} catch(const std::exception &e) {
  QWarn << e.what() << std::endl;
}
```

TU Delft

LibKet – by example

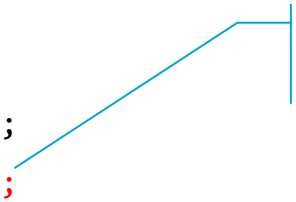
```
#include <LibKet.hpp>

using namespace LibKet;
using namespace LibKet::circuits;

auto expr = measure(qft(init()));

try {
  QDevice<QDeviceType::qi_26_simulator, 6> qpu; qpu(expr);
  auto result = qpu.eval(1024);

  QInfo << result << std::endl;
  QInfo << "job ID     : " << qpu.get<QResultType::id>(result)        << std::endl;
  QInfo << "best       : " << qpu.get<QResultType::best>(result)      << std::endl;
  QInfo << "histogram  : " << qpu.get<QResultType::histogram>(result) << std::endl;
} catch(const std::exception &e) {
  QWarn << e.what() << std::endl;
}
```



Create generic quantum expression

TU Delft

LibKet – by example

```
#include <LibKet.hpp>

using namespace LibKet;
using namespace LibKet::circuits;

auto expr = measure(qft(init()));

try {
  QDevice<QDeviceType::qi_26_simulator, 6> qpu; qpu(expr);
  auto result = qpu.eval(1024);

  QInfo << result << std::endl;
  QInfo << "job ID     : " << qpu.get<QResultType::id>(result)        << std::endl;
  QInfo << "best       : " << qpu.get<QResultType::best>(result)      << std::endl;
  QInfo << "histogram  : " << qpu.get<QResultType::histogram>(result) << std::endl;
} catch(const std::exception &e) {
  QWarn << e.what() << std::endl;
}
```

Create 6-qubit device on the 26-qubit QI simulator and upload the expression

TU Delft

LibKet – by example

```
#include <LibKet.hpp>

using namespace LibKet;
using namespace LibKet::circuits;

auto expr = measure(qft(init()));

try {
  QDevice<QDeviceType::qi_26_simulator, 6> qpu; qpu(expr);
  auto result = qpu.eval(1024);

  QInfo << result << std::endl;
  QInfo << "job ID     : " << qpu.get<QResultType::id>(result)        << std::endl;
  QInfo << "best       : " << qpu.get<QResultType::best>(result)      << std::endl;
  QInfo << "histogram  : " << qpu.get<QResultType::histogram>(result) << std::endl;
} catch(const std::exception &e) {
  QWarn << e.what() << std::endl;
}
```

Evaluate quantum expression with 1024 shots

TU Delft

LibKet – by example

Views

```
auto expr = all(qft(sel<0,3,4,6>(…)));
```

Switch to another device

```
QDevice<QDeviceType::ibmq_seattle, 433> qpu;
```

Non-blocking execution

```
auto job = qpu.execute_async(1024);

while (!job->query()) {
  // do something else
}

auto result = job->get();
```

Manual coding

```
Qprogram prog;

prog.rx      ( 3.141, {0,1,2} );
prog.h       ( {0,1,2} );
prog.h       ( 3 );
prog.rx      ( 3.141, {3,4,5} );
prog.cnot    ( {3,4,5}, {6,7,8} );
prog.measure ( {0,1,2,3,4,5,6,7,8} );

qpu(prog.to_string());
```

Advanced features

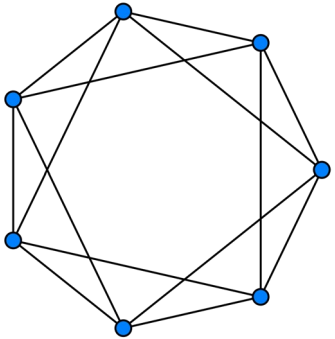
- CUDA-like streams
- JIT-compilation of quantum expressions
- Rule-based optimization

TU Delft

QPack : application-oriented scalable test cases

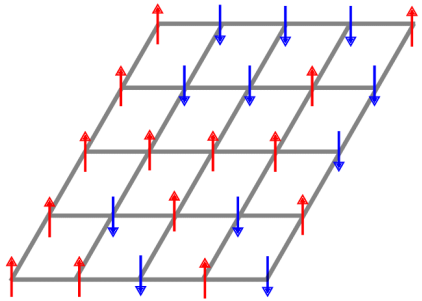
Quantum Approximate Optimization Algorithm (QAOA)

- Max-cut problem (MCP)
- Dominating set problem (DSP)
- Travelling salesperson problem (TSP)
- Maximum independent set problem (MIS)



Variational Quantum Eigensolver (VQE)

- Random diagonal Hamiltonian (RH)
- Transverse Ising chain (IC)



TU Delft

Mesman, K., Al-Ars, Z., and Möller, M. (2022): QPack – Quantum Approximate Optimization Algorithms as universal benchmark for quantum computers. arXiv: 2103.17193

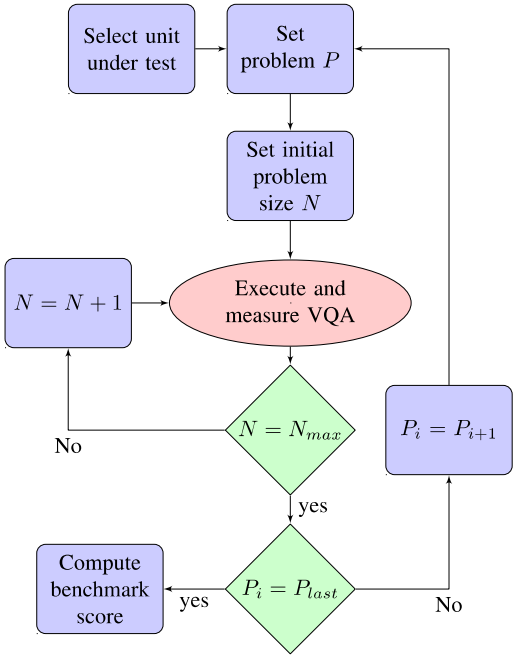
QPack : application-oriented scalable test cases

Quantum Approximate Optimization Algorithm (QAOA)

- Max-cut problem (MCP)
- Dominating set problem (DSP)
- Travelling salesperson problem (TSP)
- Maximum independent set problem (MIS)

Variational Quantum Eigensolver (VQE)

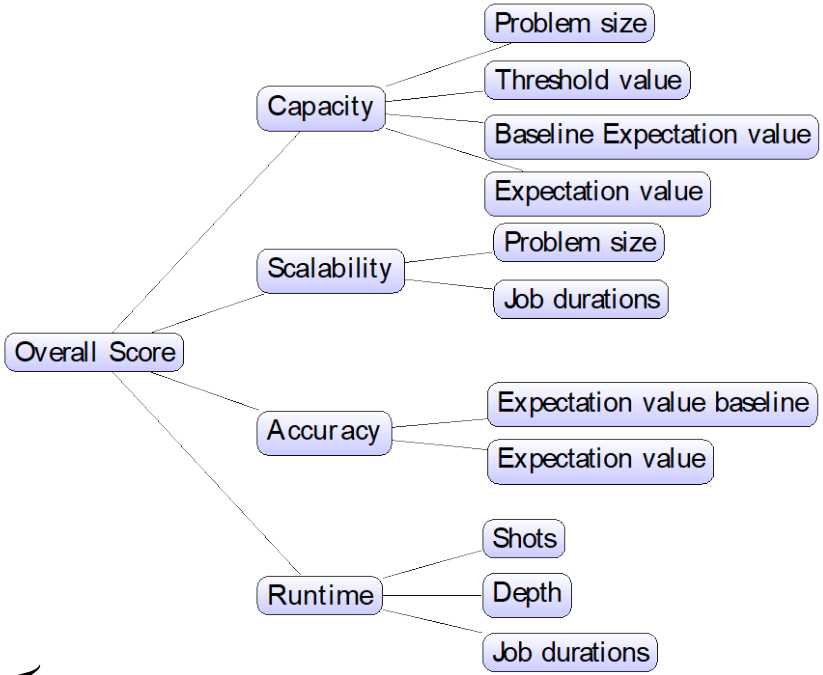
- Random diagonal Hamiltonian (RH)
- Transverse Ising chain (IC)

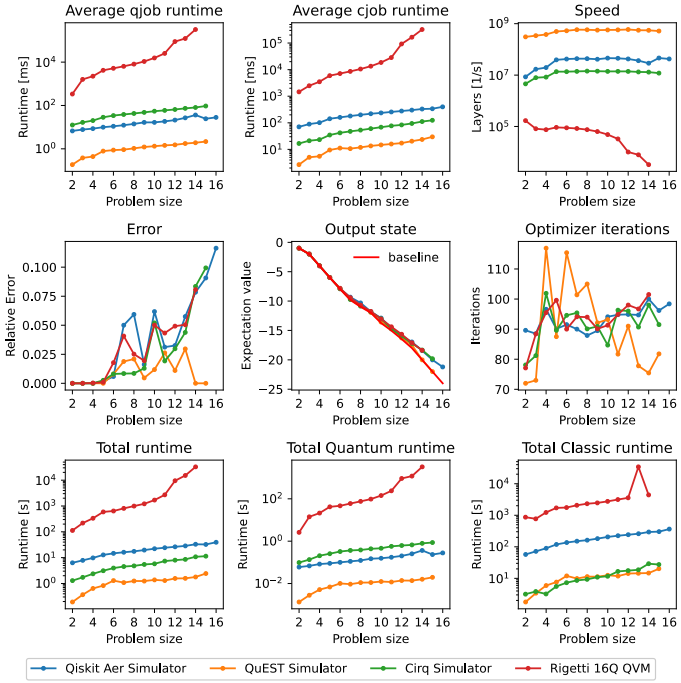


TU Delft

Donkers, H., Mesman, K., Al-Ars, Z. and Möller, M. (2022): QPack Scores – Quantitative Performance Metrics for Application-oriented Quantum Computer Benchmarking. arXiv: 2205.12142

QPack: multiple metrics





TU Delft

Donkers, H., Mesman, K., Al-Ars, Z. and Möller, M. (2022): QPack Scores – Quantitative Performance Metrics for Application-oriented Quantum Computer Benchmarking. arXiv: 2205.12142

QPack: multiple metrics

Capacity

- maximal number of qubits for which QPU achieves prescribed relative error relative to QuEST simulator

Scalability

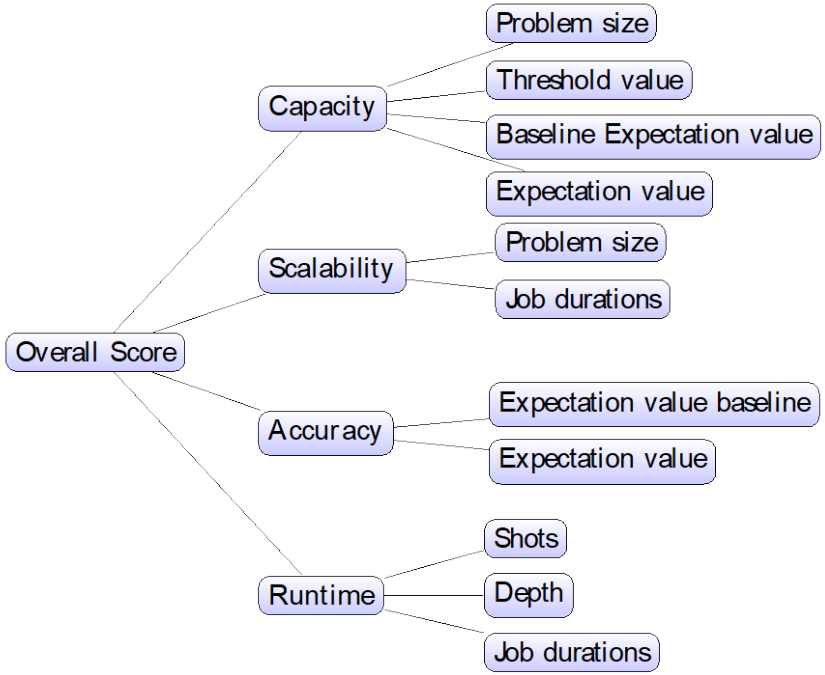
- power law fitting $\mathrm{T}_q = (\text{problem size})^{\alpha}$

Accuracy

- average relative error between expectation value of ideal simulator (QuEST) and QPU under testing

Runtime

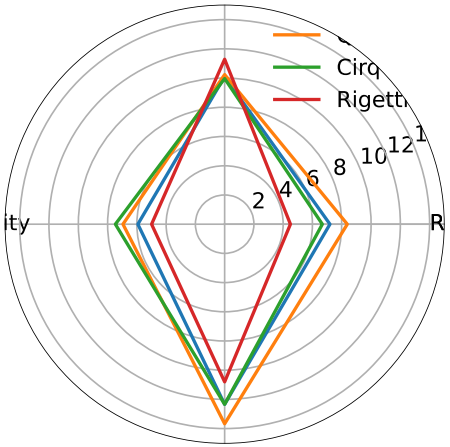
- average #gates per second over all problem sizes



TU Delft

Donkers, H., Mesman, K., Al-Ars, Z. and Möller, M. (2022): QPack Scores – Quantitative Performance Metrics for Application-oriented Quantum Computer Benchmarking. arXiv: 2205.12142

QPack: single score

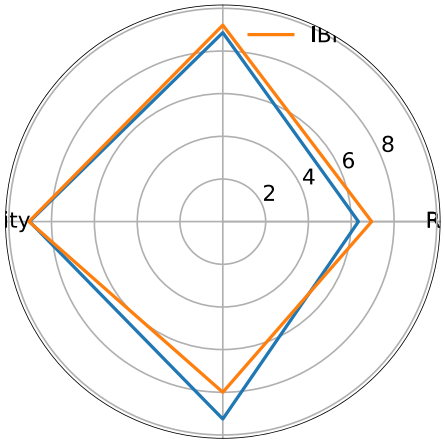


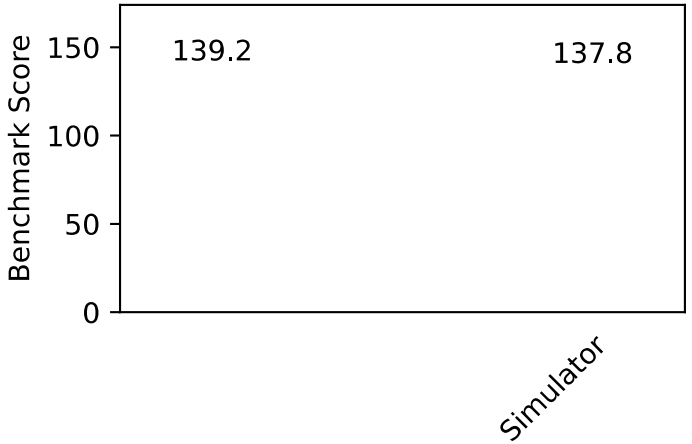
$$S = \frac{1}{2}(s_{\text{runtime}} + s_{\text{scalability}})(s_{\text{accuracy}} + s_{\text{capacity}})$$

TU Delft

Donkers, H., Mesman, K., Al-Ars, Z. and Möller, M. (2022): QPack Scores – Quantitative Performance Metrics for Application-oriented Quantum Computer Benchmarking. arXiv: 2205.12142

QPack: preliminary result on remote simulators

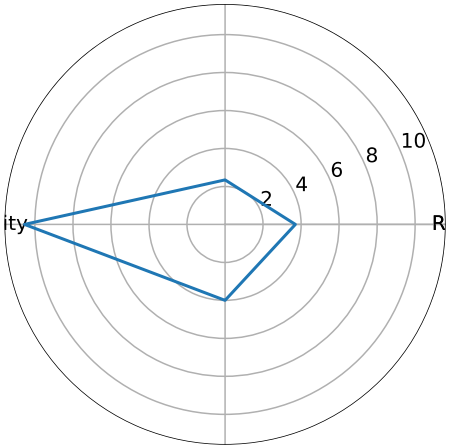


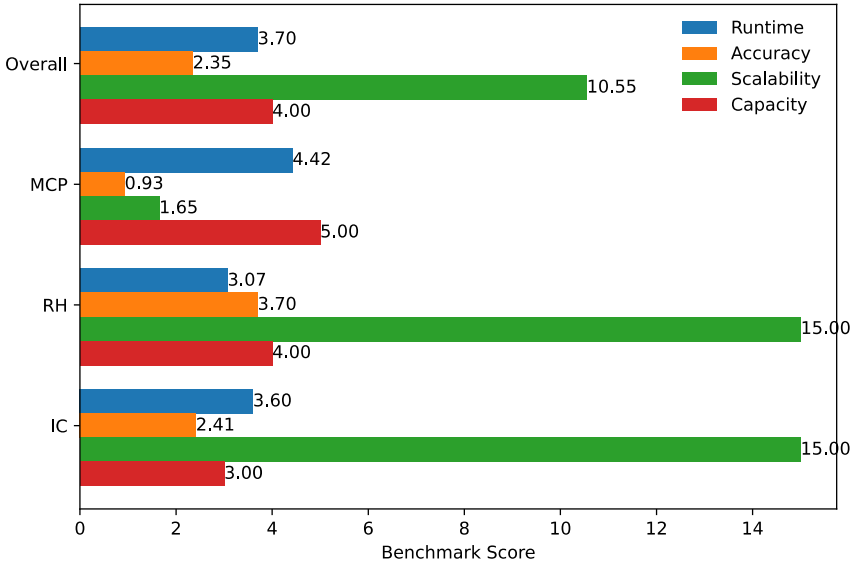


TU Delft

Donkers, H., Mesman, K., Al-Ars, Z. and Möller, M. (2022): QPack Scores – Quantitative Performance Metrics for Application-oriented Quantum Computer Benchmarking. arXiv: 2205.12142

QPack: preliminary result on hardware QPU





TU Delft

Donkers, H., Mesman, K., Al-Ars, Z. and Möller, M. (2022): QPack Scores – Quantitative Performance Metrics for Application-oriented Quantum Computer Benchmarking. arXiv: 2205.12142

Summary and outlook

- QPack is an application-oriented scalable benchmark for NISQ computers
- Extension of test suite and benchmarking of other QPUs is ongoing (support & access is welcome!)




TU Delft

References

- Möller, M. and Schalkers, M. (2020): LibKet – A Cross-Platform Programming Framework for Quantum-Accelerated Scientific Computing. ICCS 2020. Lecture Notes in Computer Science
- Mesman, K., Al-Ars, Z., and Möller, M. (2022): QPack – Quantum Approximate Optimization Algorithms as universal benchmark for quantum computers. arXiv: 2103.17193
- Donkers, H., Mesman, K., Al-Ars, Z. and Möller, M. (2022): QPack Scores – Quantitative Performance Metrics for Application-oriented Quantum Computer Benchmarking. arXiv: 2205.12142

Thank you!