Toward a High-Performance Batched Quantum Circuit Simulator for Heterogeneous Architectures

Koki Kawamura[1], Kazuki Yamauchi[1], Yuki Nakaya[1], Ryo Matsumoto[1], Keichi Takahashi[1], Toshio Mori[1,2], Yasunari Suzuki[2], Keisuke Fujii[1,2]

[1]The University of Osaka [2]RIKEN Center for Quantum Computing k-koki@ist.osaka-u.ac.jp

Background

- **Quantum circuit simulators** (e.g., Qulacs[S+21], cuQuantum[B+23]) are widely utilized for quantum computing research.
- **State-vector simulation** is a prominent approach, which stores n-qubit quantum states as 2^n-length complex vector.
- Multiple **parallel computing architectures** (e.g., SIMD, OpenMP, CUDA) are typically used.
- **Batch execution** of quantum circuits: simulate multiple quantum circuits which share the same structure.
- Writing code in multiple architectures makes software maintenance difficult and increases the risk of bugs.
- Only few simulators support high-performance batch execution.

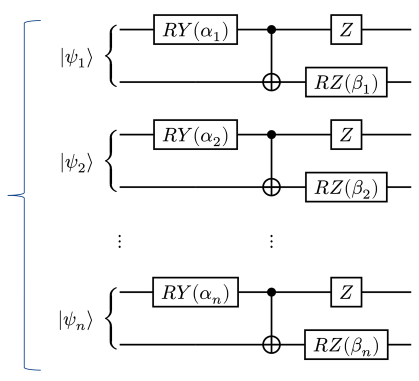
State-vector Method

$$a_{00}|00\rangle + a_{01}|01\rangle + a_{10}|10\rangle + a_{11}|11\rangle \Rightarrow \begin{pmatrix} a_{00} \\ a_{01} \\ a_{10} \\ a_{11} \end{pmatrix}$$

$$\alpha|0\rangle\langle 0| + \beta|0\rangle\langle 1| + \gamma|1\rangle\langle 0| + \delta|1\rangle\langle 1| \Rightarrow \begin{pmatrix} \alpha & \beta \\ \gamma & \delta \end{pmatrix}$$

Batch Execution of Quantum Circuit

Simulate these circuits in parallel



Our Contribution

We present a new high-performance state-vector-type simulator supporting batch simulation on both CPU and GPU.

Supporting Heterogeneous Architectures

- We utilize Kokkos[T+22], a parallel programming library for CPU and GPU from a single codebase with minimum divergence.
- This abstraction imposes almost no overhead as original functions are inline-expanded into the device function for each architecture.

```
Kokkos::parallel_for(Kokkos::RangePolicy<Space>(0, n),
KOKKOS_LAMBDA(int i) {…});
```

Abstraction using C++ templates and macros

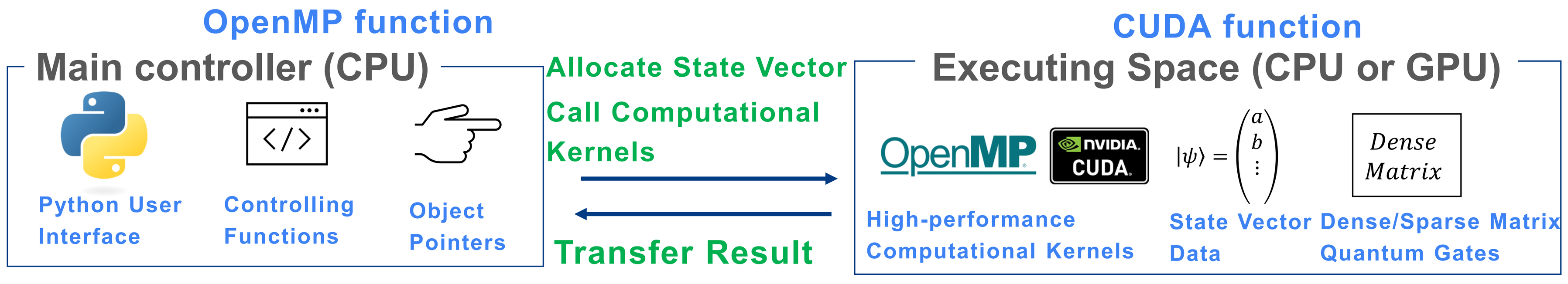
Call architecture-specific functions

```
#pragma omp parallel for
for(int i = 0; i < n; i++)
{…}
```

OpenMP function

```
__device__ void func() {
  int i = …; …
}
```

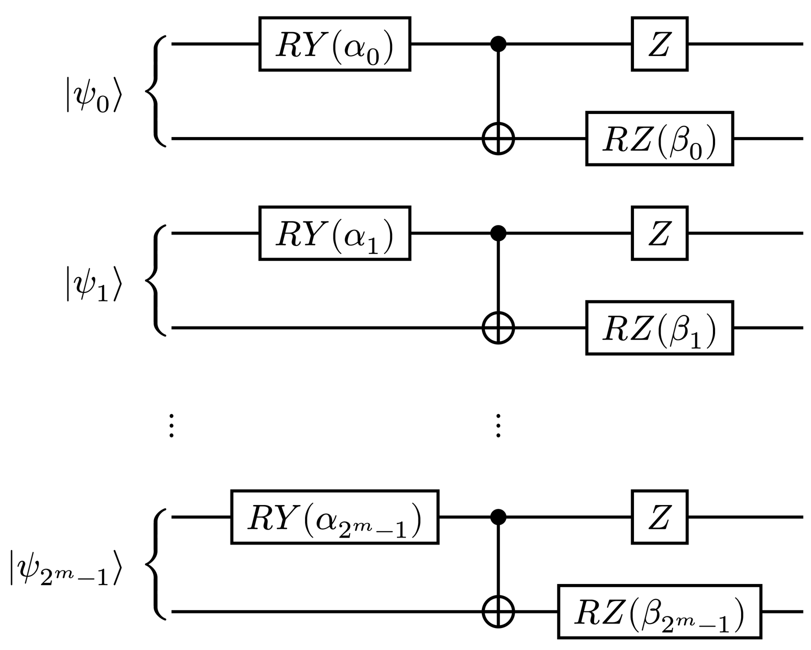
CUDA function

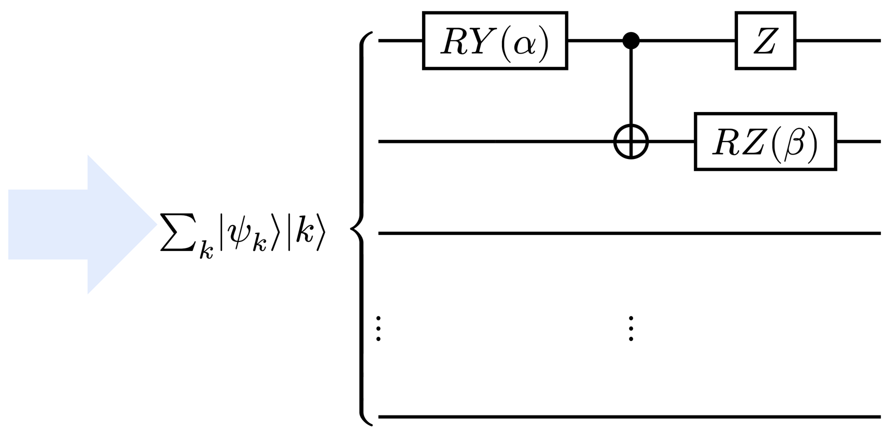


Enabling Batch Execution

- n-qubit state vectors of batch size 2^m is allocated with the same structure as a $(n+m)$-qubit state vector.
- We can simulate bathed quantum circuits in the same way as normal state vector even if the batch size is not a power of 2. (single for-loop for OpenMP, 1D thread parallelization for CUDA)

a_{00}	a_{01}	a_{10}	a_{11}	b_{00}	b_{01}	b_{10}	b_{11}	c_{00}	c_{01}	c_{10}	c_{11}



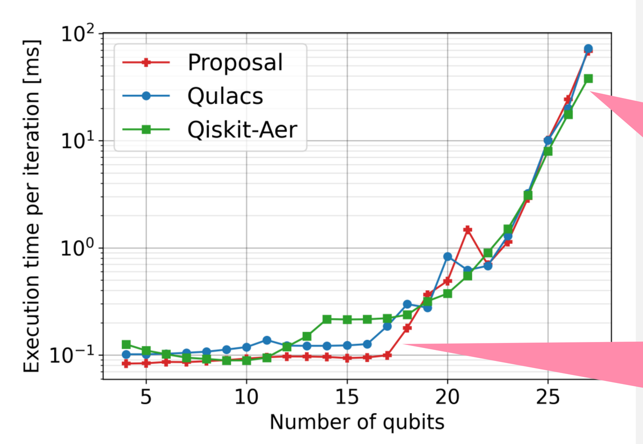


Although $RY(\alpha)$ is written as $\sum_k RY(\alpha_k) \otimes |k\rangle\langle k|$ in more precise form, we can regard this as simple RY gate where rotation angle vary by batch index.

Performance Evaluation

- CPU: Intel Xeon Platinum 9242 x 2 sockets (2.30GHz, 96 cores), GPU: NVIDIA A100 40 GB
- Method: Benchmark execution time to apply CX, RX and RZ gates (averaged over every target qubit)
- Compared to Qulacs[S+21], cuQuantum[B+23], and Qiskit-Aer[J+24]

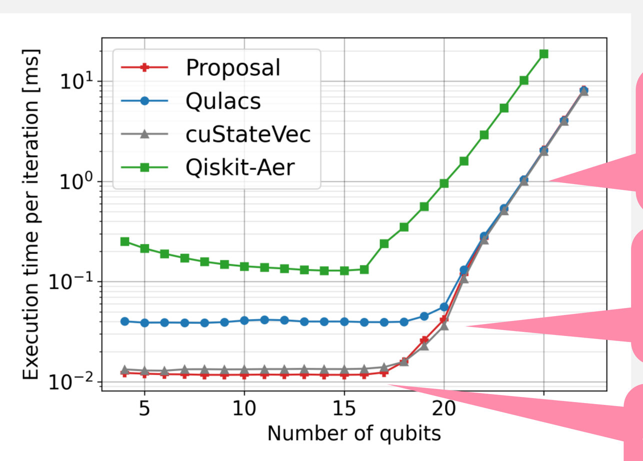
1. Single State Vector Update



Qiskit-Aer is faster than others at #qubits=26

Our simulator is the fastest at #qubits≤18 (can be much faster by disabling OpenMP)

CPU result



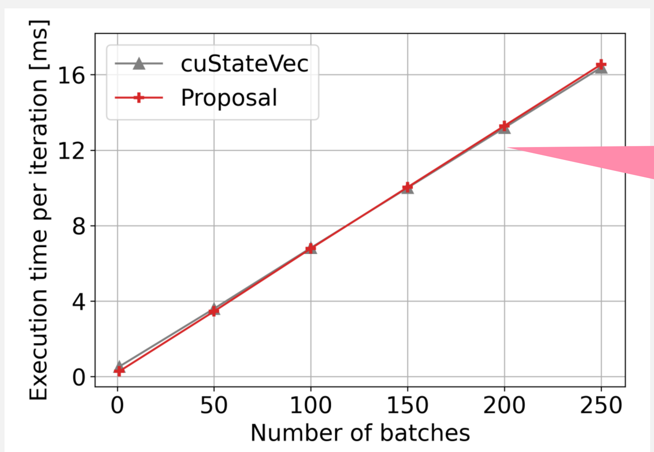
Almost equivalent to the fastest (Qulacs, cuStateVec)

A little slower than cuStateVec at #qubits=19, 20

A little faster than cuStateVec at #qubits≤17

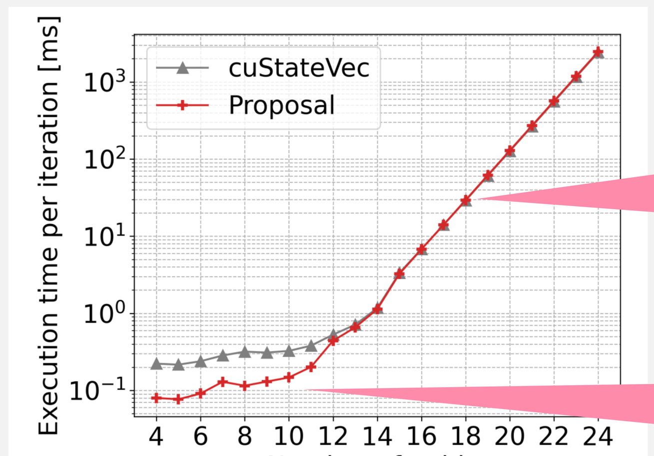
GPU result

2. Batched State Vector Update



Equivalent to cuStateVec

varying batch size (#qubits=16)



Equivalent to cuStateVec at #qubits ≥ 14

Up to 2.8x faster than cuStateVec at #qubits≤11

varying #qubits (batch size=100)

Advanced Usage

Our simulator shows 6.8x speed-up compared to Qulacs in a practical usage.

Task: solve maximum cut problem by PCE (Pauli Correlation Encoding)[S+24].

OMP threads	Qulacs (it/s)	Proposal (it/s)
4	1.54	7.40
8	1.53	9.30
16	1.46	9.96
32	1.30	6.70

Each iteration consists of gradient calculations of expectation values for parametric circuits.

Circuit size details:

- 13-qubit circuit
- 10,400 parametric gates
- 800 Pauli-term observable

Conclusion & Future Work

- We developed a state-vector-type quantum circuit simulator which work on both CPU and GPU.
- Our simulator also features batch execution of circuits.
- Our simulator performs as well as the fastest existing simulator.
- We are planning to support diverse platforms such as AMD and Intel GPUs and release our simulator officially in the future.

Learn More

Posting any issues (questions / bug reports / feature requests) is welcome!

GitHub: https://github.com/qulacs/scaluq

Python Tutorial: https://scaluq.readthedocs.io/en/latest/tutorials/python/index.html

Acknowledgment

This work was supported by the JST COI-NEXT Program Grant No. JPMJPF2014.

References

[B+23] Harun Bayraktar et al. cuQuantum SDK: A High-Performance Library for Accelerating Quantum Science, 2023 IEEE International Conference on Quantum Computing and Engineering (QCE).
[S+21] Yasunari Suzuki et al. Qulacs: a fast and versatile quantum circuit simulator for research purpose. Quantum, 5:559, October 2021.
[T+22] Christian R. Trott et al. Kokkos 3: Programming model extensions for the exascale era. IEEE Transactions on Parallel and Distributed Systems, 33(4):805–817, 2022
[J+24] Ali Javadi-Abhari et al. Quantum computing with Qiskit. arXiv:2405.08810.
[S+24] Marco Sciorilli et al. Towards large-scale quantum optimization solvers with few qubits. arXiv preprint arXiv:2401.09421.