Quantum Computer Simulator

Scope of this presentation

Item	Description
Simulation Method	Statevector
Computation Resource	GPU (Graphics Processing Unit)
Language, Framework	C++, CUDA, MPI (Message Passing Interface), NCCL (NVIDIA Collective Communications Library)

Not: Actual QCs, Noise Mitigation Methods, Tensor network based, CPU based

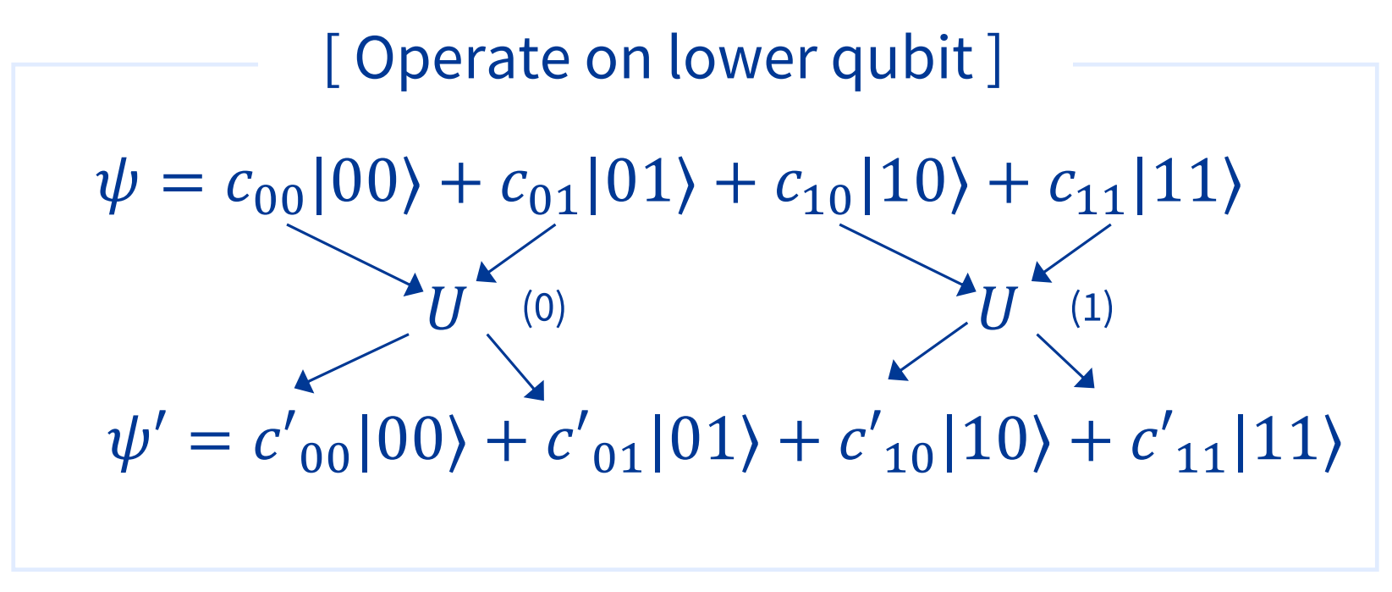
Statevector method

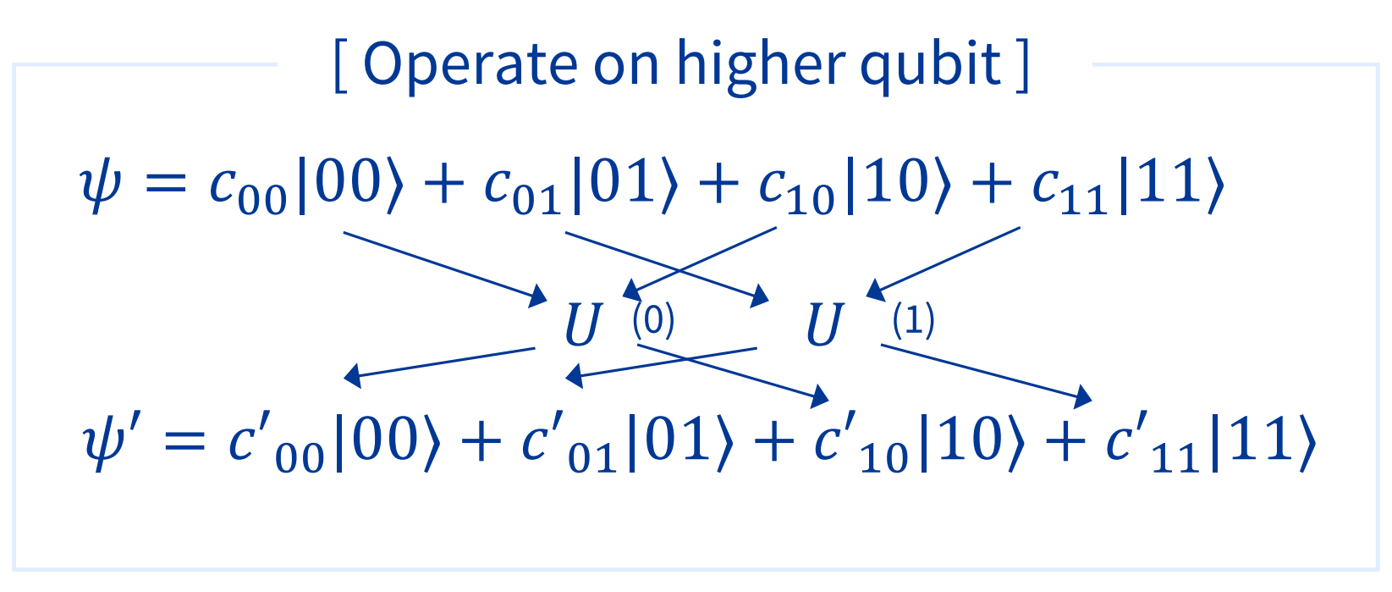
The state-vector method stores the full quantum state in memory and applies quantum gates as matrix–vector multiplications.

Quantum State $\psi = c_{00}|00\rangle + c_{01}|01\rangle + c_{10}|10\rangle + c_{11}|11\rangle$

Memory $\boxed{c_{00}}\boxed{c_{01}}\boxed{c_{10}}\boxed{c_{11}}$ This is Statevector

Quantum Gate


[Operate on lower qubit]
ψ = c00|00⟩ + c01|01⟩ + c10|10⟩ + c11|11⟩
U (0)
U (1)
ψ′ = c′00|00⟩ + c′01|01⟩ + c′10|10⟩ + c′11|11⟩


[Operate on higher qubit]
ψ = c00|00⟩ + c01|01⟩ + c10|10⟩ + c11|11⟩
U (0)
U (1)
ψ′ = c′00|00⟩ + c′01|01⟩ + c′10|10⟩ + c′11|11⟩

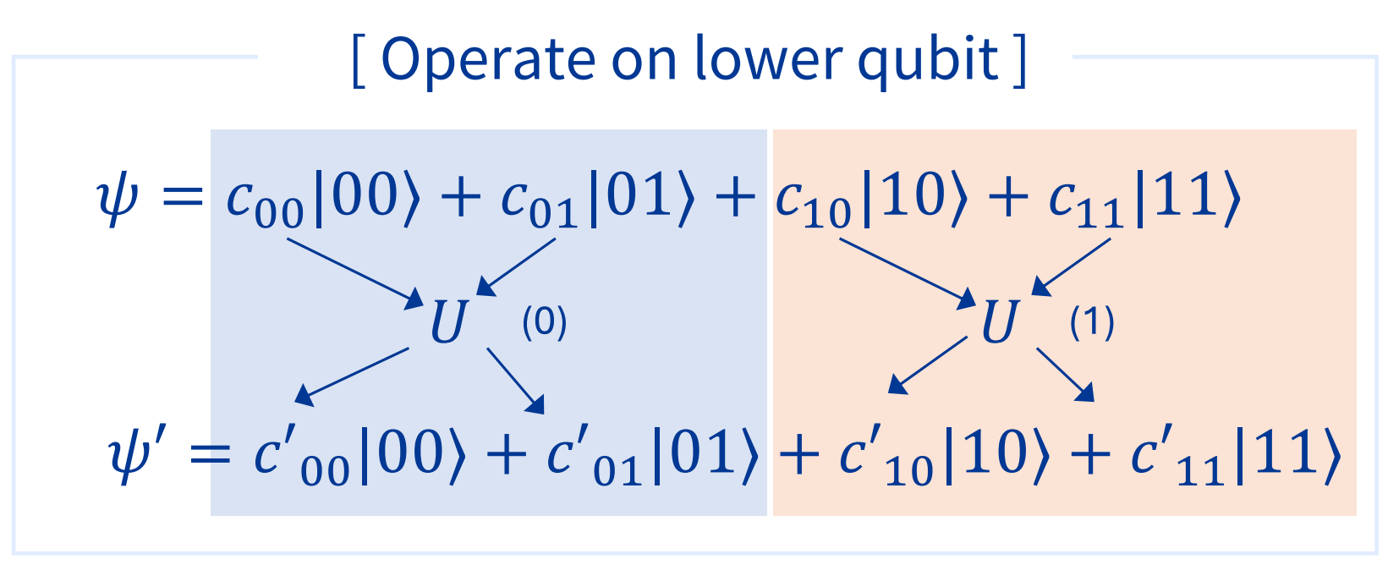
Parallel Computation

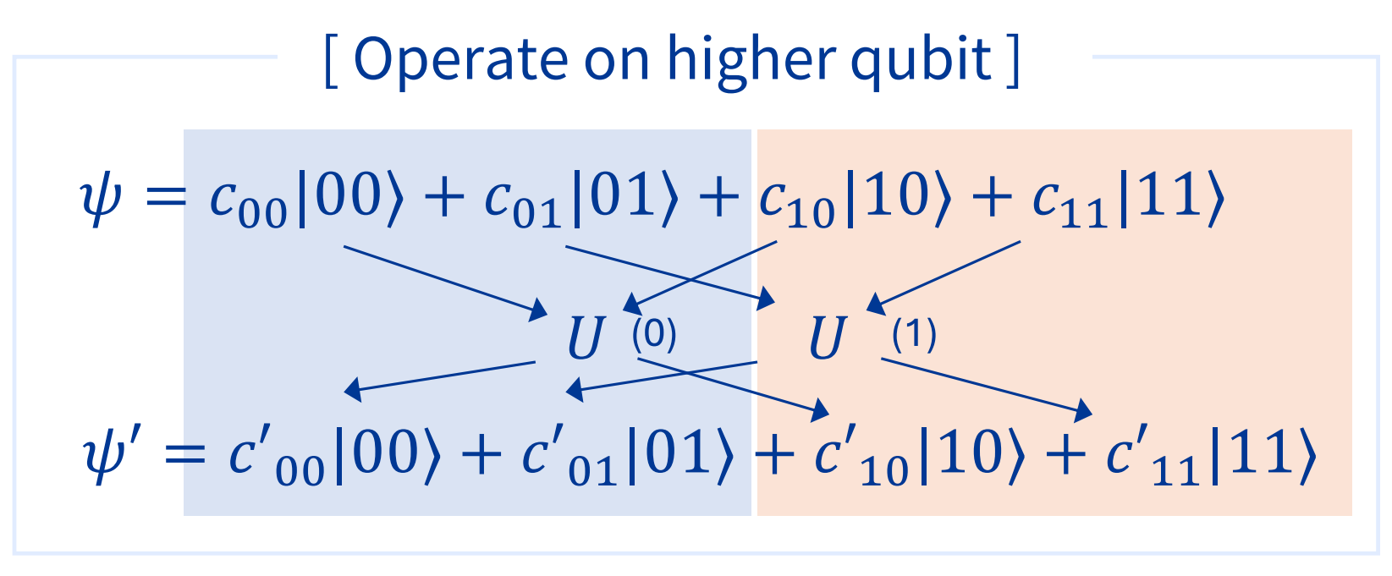
In multi-GPU systems, the state vector is partitioned and distributed across devices.

$\boxed{c_{00}}\boxed{c_{01}}\boxed{c_{10}}\boxed{c_{11}} \Longrightarrow \boxed{c_{00}}\boxed{c_{01}} \quad \boxed{c_{10}}\boxed{c_{11}}$

GPU ID	Memory
0	c_{00}, c_{01}
1	c_{10}, c_{11}

When gates act on **high-order qubits**, the corresponding operations span multiple partitions and require **inter-GPU**.


[Operate on lower qubit]
ψ = c00|00⟩ + c01|01⟩ + c10|10⟩ + c11|11⟩
U (0)
U (1)
ψ′ = c′00|00⟩ + c′01|01⟩ + c′10|10⟩ + c′11|11⟩


[Operate on higher qubit]
ψ = c00|00⟩ + c01|01⟩ + c10|10⟩ + c11|11⟩
U (0)
U (1)
ψ′ = c′00|00⟩ + c′01|01⟩ + c′10|10⟩ + c′11|11⟩

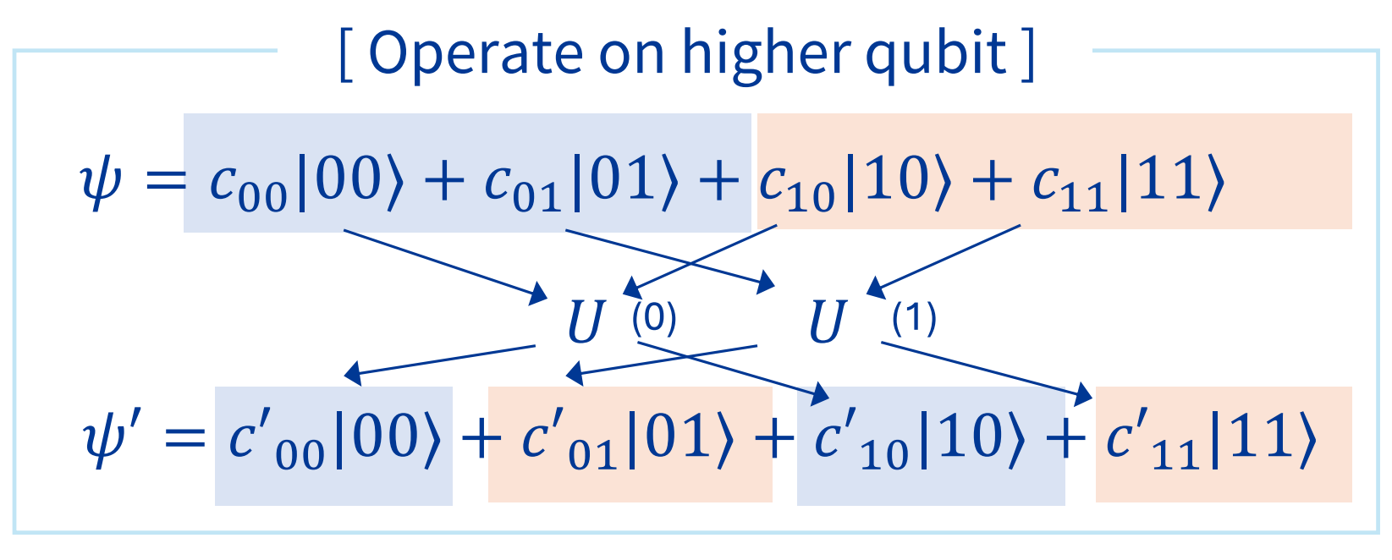
Communication occurs when the bits being operated on include any bits at or above the upper $\log_2$(number of GPUs) bits. Such bits are referred to as global bits, while the lower bits are called local bits. When communication occurs, the communication volume corresponds to the entire state vector. As the number of qubits increases, this volume becomes **extremely large** (2^n sized).

Qubit Reordering

Qubit Reordering (QR) [1] mitigates this issue by dynamically modifying the mapping between logical qubits and indices, thereby halving the communication volume.

Relocation of Partitioned Memory

GPU ID	Memory Before	Memory After
0	c_{00}, c_{01}	c_{00}, c_{10}
1	c_{10}, c_{11}	c_{01}, c_{11}


[Operate on higher qubit]
ψ = c00|00⟩ + c01|01⟩ + c10|10⟩ + c11|11⟩
U (0)
U (1)
ψ′ = c′00|00⟩ + c′01|01⟩ + c′10|10⟩ + c′11|11⟩

As a result, the communication volume is reduced by half. However, it remains extremely large, and communication still becomes the performance bottleneck.

Since QR changes the data layout, all GPUs must communicate in a synchronized manner to ensure that data is not lost due to overwriting.

	With QR	Without QR
Advantages	Communication volume is halved	**A design in which each GPU operates independently is possible**
Disadvantages	Coordinated operation among processes (synchronization required)	Communication volume increases

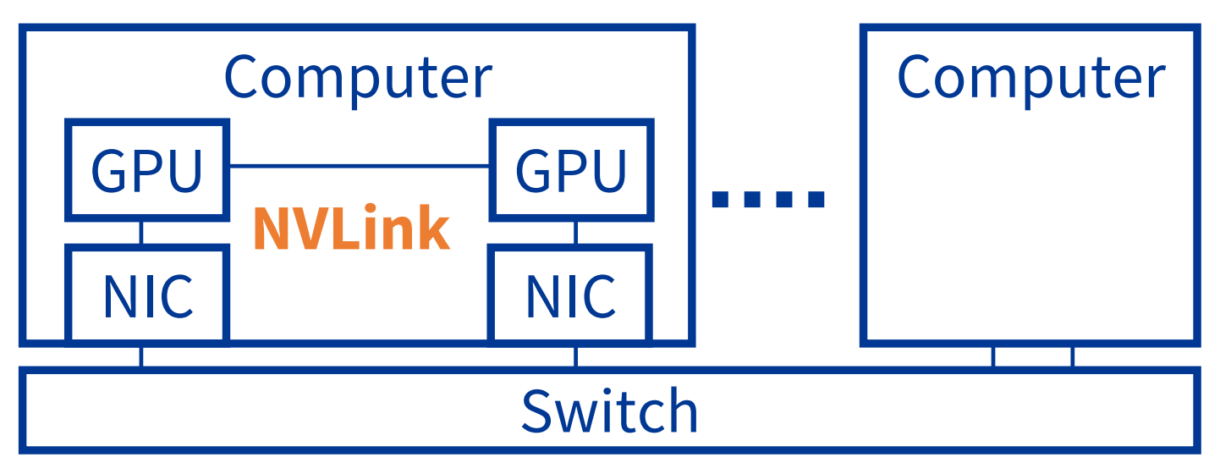
Communication Hierarchy & NVLink

The characteristics of NVLink and UVA suggest that, in environments with fast interconnects, a QR-free approach can achieve competitive or superior performance.

GPUs are interconnected via high-speed NVLink as well as slower network interface cards (NICs) such as InfiniBand.

NVLink is integrated into the GPU memory system, **enabling low-overhead access to the memory of other GPUs**.

⇒ This can make the approach without QR more advantageous.



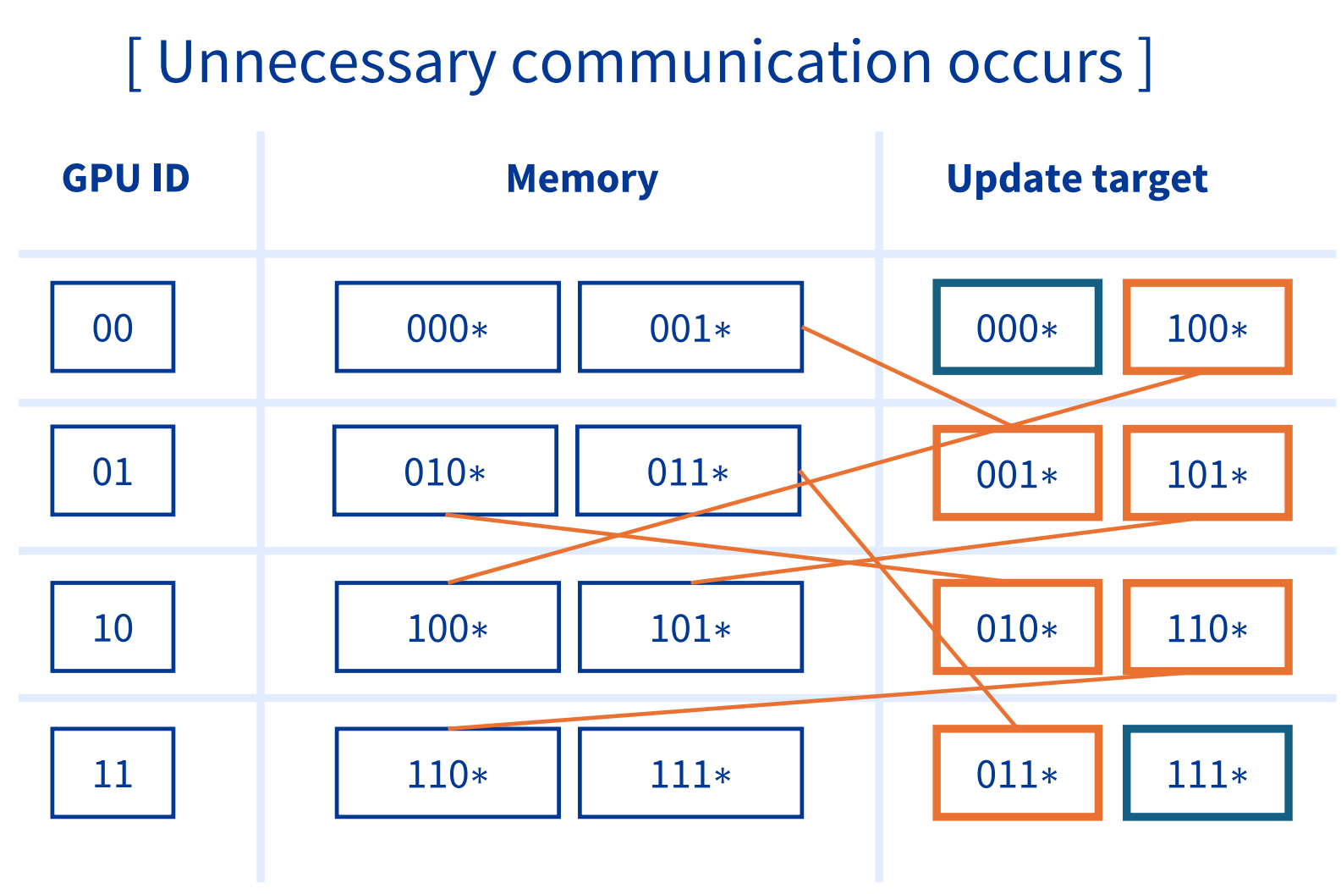
Furthermore, between GPUs connected via NVLink, **Unified Virtual Addressing (UVA)** allows access to another GPU's memory without explicitly writing program instructions for the counterpart GPU to send or receive data. This access model is similar to one-sided access, and as a result, each GPU can **operate independently**.

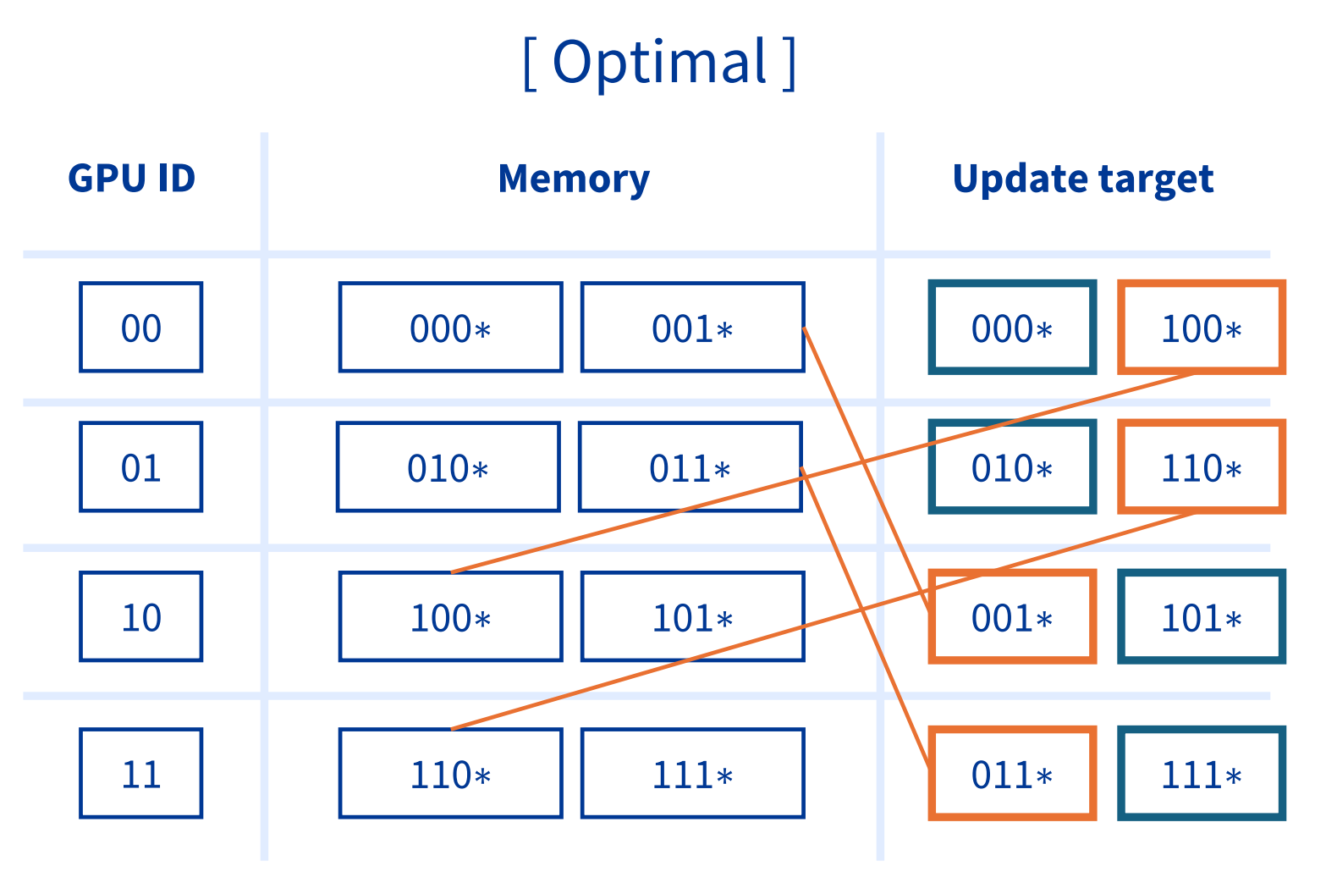
Suggestion of Memory Sharing Method

Based on the above discussion, we propose a memory-sharing method that enables efficient processing without QR in environments where NVLink is available.

Importance of Access Location Design

When processing without QR, the pairs assigned to each GPU must be carefully designed; otherwise, unnecessary communication will be incurred.





Access Location Design

** For simplicity of explanation, we assume that the number of qubits being operated on is one; however, the same approach can be extended straightforwardly to larger cases.*

[Notation]

The indices representing pairs of state vector elements are written in binary form, such as **00|00…0 to 11|11…1**. Here, the vertical bar "|" separates the **GPU identifier** on the left from the remaining bits. Similarly, **the positions of state vector elements** are expressed in binary, with the GPU identifier on the left and the index of the state vector element within the GPU on the right: **00|000…0 to 11|111…1**.
Note that the total number of bits here is one greater than that of the pair indices.

[Design]

Suppose a pair index is specified as **ab|x…x**. To achieve the optimal behavior described earlier, the corresponding state vector element positions are computed as follows:

- **When operating on a global bit**: Set the corresponding global bit position to 0 or 1, and then insert the original value into the local portion.
- **When operating on a local bit**: Insert 0 or 1 at the corresponding local position.

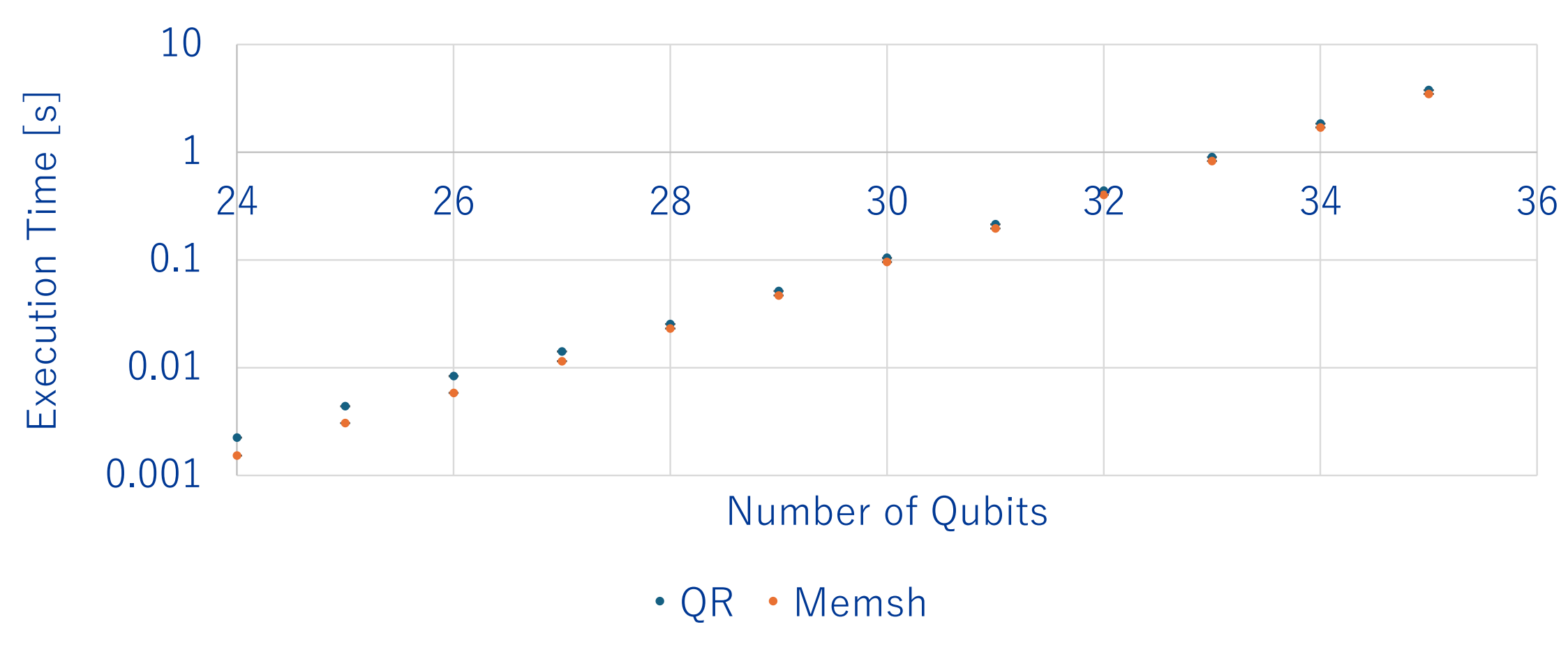
Operation Position	State Vector Element Positions
Most significant bit	0b|ax…x, 1b|ax…x
Second most significant bit	a0|bx…x, a1|bx…x
Lower bits (local)	ab|x..0..x, ab|x..1..x

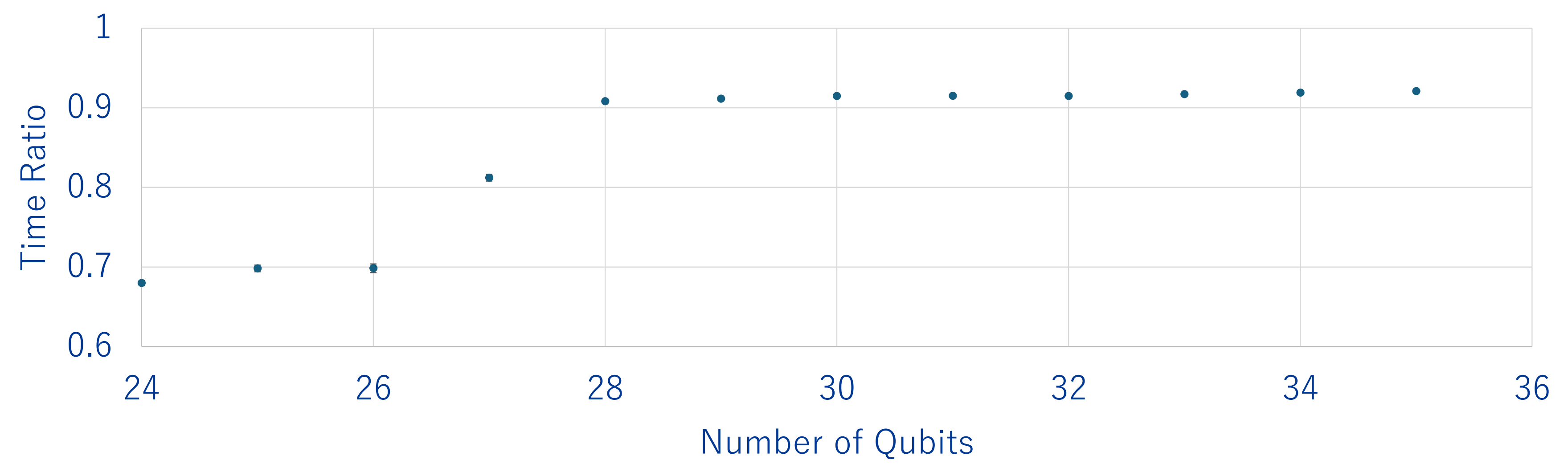
With this design, it is guaranteed that at least one element of each pair (combination) is a state vector element stored on the same GPU.

Evaluation of Memory Sharing Method

Compared with QR, an approximately **8% speedup** was observed.

Item	Description
GPU	8 × NVIDIA A100
NVLink bandwidth	1.2 TB/s bidirectional
Benchmark	Applying Hadamard gates to all qubits





Conclusions

A QR-free state-vector simulation method for NVLink-connected GPUs with Peer-to-Peer Memory Access was presented. Evaluation on an eight-A100 system showed approximately 8% improvement over a QR-based method, indicating that removing QR can be beneficial for quantum simulation on high-bandwidth GPU platforms.

Acknowledgements

The author expresses sincere gratitude to Naoki Yoshioka and Nobuyasu Ito for their valuable guidance on fundamental concepts. This presentation is based on results obtained from a project, JPNP20017, commissioned by the New Energy and Industrial Technology Development Organization (NEDO).

References

[1] K. De Raedt, K. Michielsen, H. De Raedt, B. Trieu, G. Arnold, M. Richter, Th. Lippert, H. Watanabe, and N. Ito. 2007. Massively parallel quantum computer simulator. *Comput. Phys. Commun.* 176, 2 (Jan, 2007), 121–136. DOI: https://doi.org/10.1016/j.cpc.2006.08.007