A Community Detection–Based Parallel Algorithm for Quantum Circuit Simulation

Presented By:

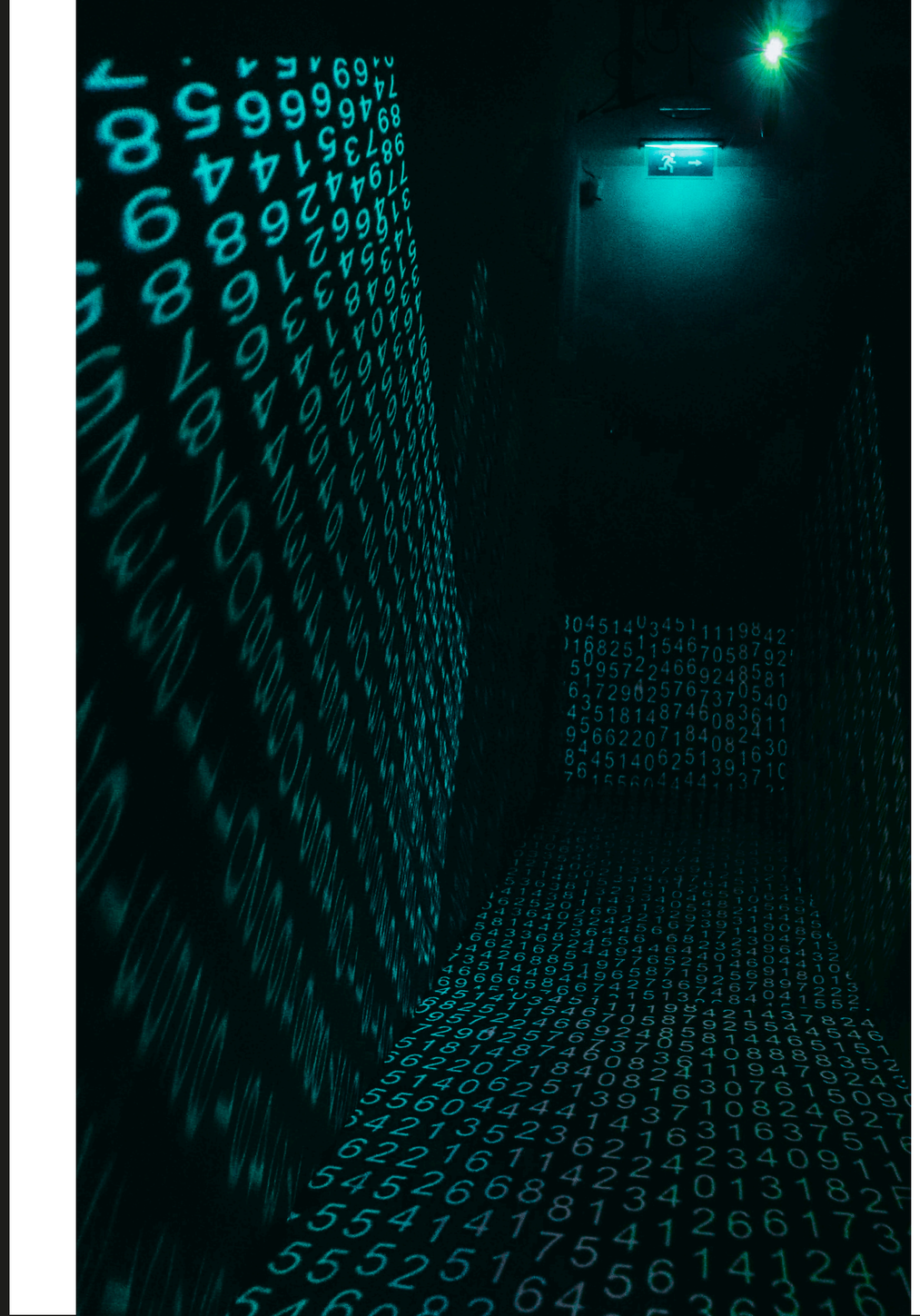
Muhammad Usman
Abdul Wahab
Ahmed Ali Zaidi

Why Simulate Quantum Circuits Classically ?

In the current era of quantum computing (NISQ), devices are noisy and have limited qubit counts.

Classical simulators validate quantum algorithms and testing circuit behavior.

However, simulation becomes exponentially more difficult as the number of qubits increases.



Motivation

- Quantum computing is making an impact in optimization, drug discovery, and cryptography.
- Because quantum hardware is not yet scalable, we rely on simulations to explore quantum solutions. (e.g., 50 qubits ≈ 9 PB)
- AnalysisHigh computational cost of tensor network contraction.

Tensor Networks Overview

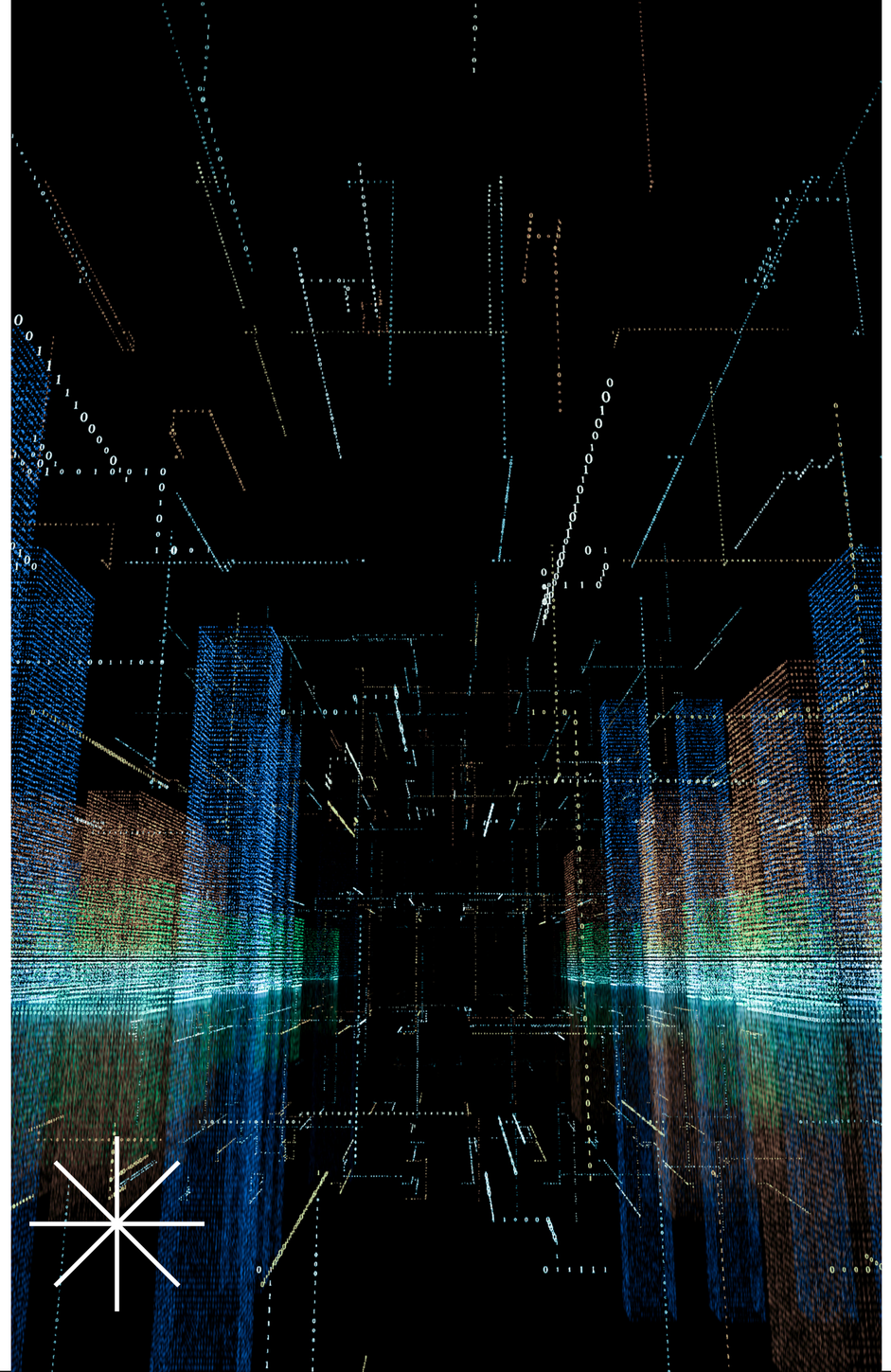
Tensor networks efficiently represent quantum circuits where tensors are nodes and qubit interactions are edges.

- Nodes = quantum gates (tensors)
- Edges = qubits (indices)

Simulation involves contracting these tensors, similar to multiplying matrices

Merge tensors pairwise to compute amplitudes

Choosing the right contraction order can drastically reduce cost but finding it is NP-hard.



Existing Parallelization Strategies

MPI with Slicing :

- Slice indices to split the network → Contract sub-networks in parallel.
- High communication overhead.

GPU-Based Contraction :

- Parallelize tensor-pair contractions on GPU (e.g., cuTENSOR)
- Limited by GPU memory and tensor ranks.

Limitation :

- Spatial cost dominates for large circuits

Proposed Algorithm

Balances spatial and temporal costs via community-level parallelism.

Stage 1 :
Community Detection

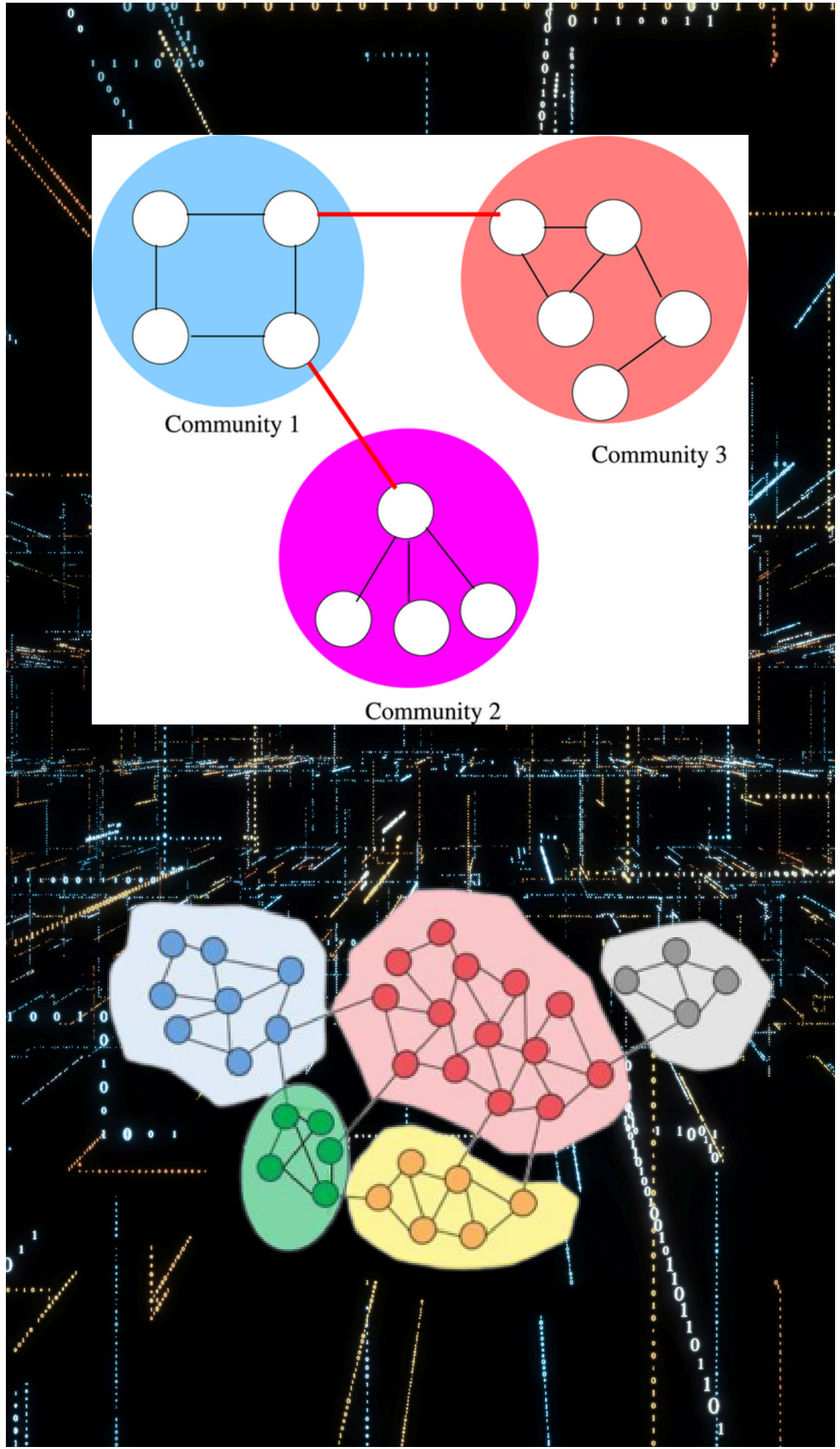
Use the Girvan–Newman (GN) algorithm to detect weakly-connected communities.

Stage 2 :
Parallel Community Contraction

Parallel contraction of these communities on CPU cores.

Stage 3 :
Final Network Contraction

Sequentially contract the reduced network.



Girvan–Newman Algorithm

Steps :

- Calculate edge betweenness centrality
- Remove edges with highest centrality → Split into communities
- Repeat until desired partitions

Advantages :

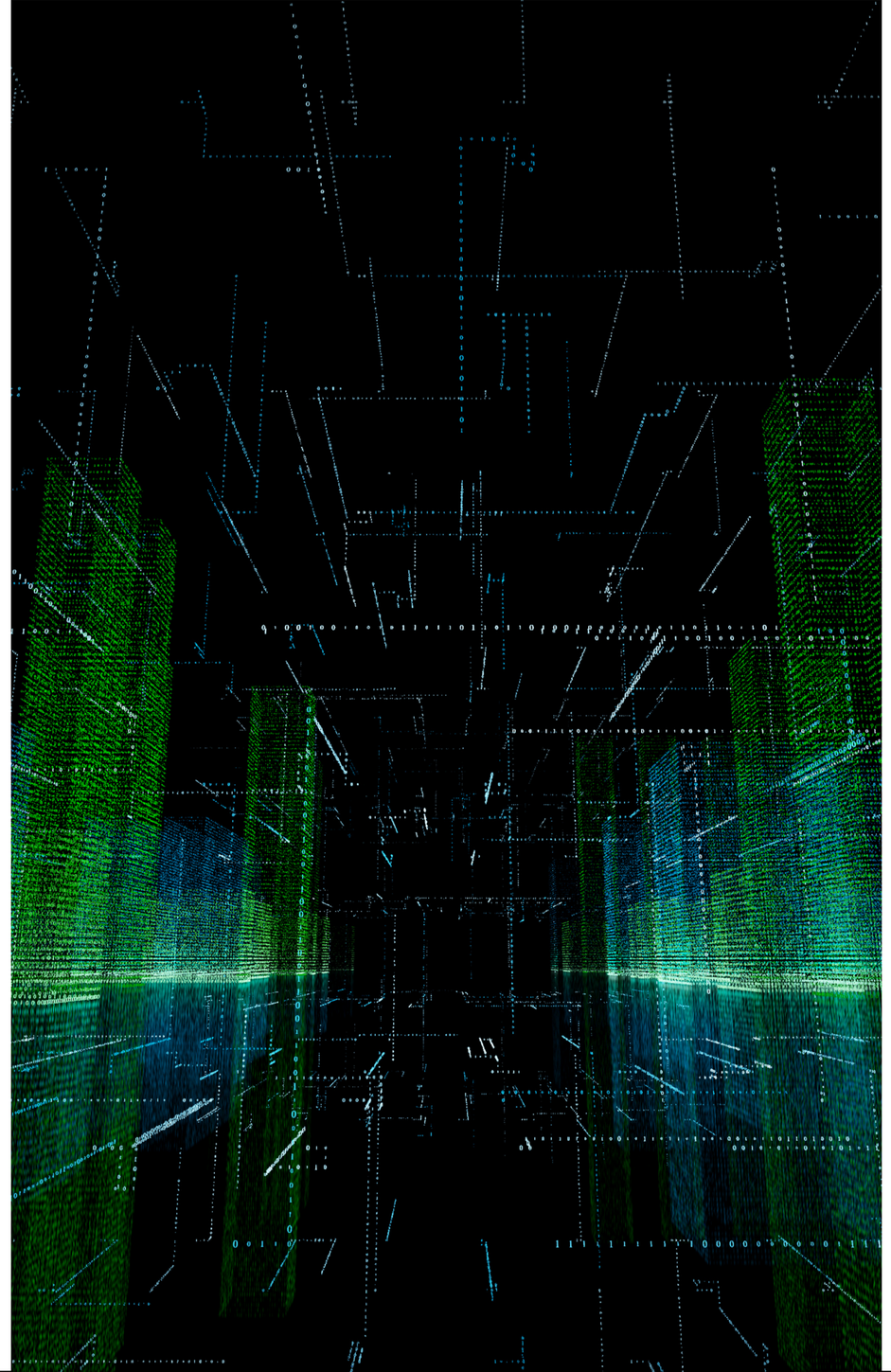
- High-quality communities for tensor networks
- Generates contraction plans automatically

Proposed Parallelization Strategy

Partition tensor graph with **METIS**

Distribute communities with **MPI**

Contract with **OpenMP/OpenCL**



Why METIS ?

Advantages Over Girvan–Newman

01 **Faster for Larger Graphs.**

02 **Scalable for Distributed Systems.**

03 **Minimizes Edge Cuts**

Expected Results

Comparison

Sequential vs. MPI + METIS vs. Hybrid

Speedup curves for QFT, RQC

Metrics:

- Speedup: Tsequential/Tparallel.
- Scalability: Strong vs. weak scaling.
- Memory Efficiency: Reduced spatial cost.

Challenges & Mitigation

Challenges:

- Load imbalance
- MPI communication overhead
- GPU memory limits

Mitigation:

- METIS balancing constraints
- Non-blocking MPI
- Optimized OpenCL kernels

Conclusion

Hybrid parallelism + community detection reduces costs.

METIS enables scalable partitioning.

Any Questions?