Synthesis on Atom Computation

Dingchao Gao

Institute of Software Chinese Academy of Sciences

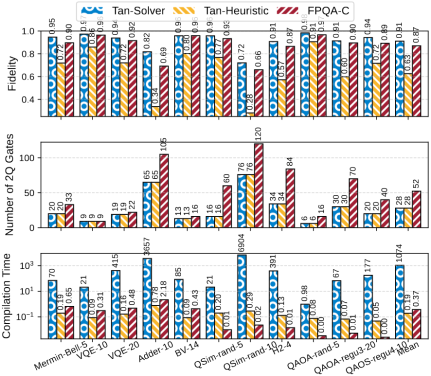
June 4, 2024

Outline

1. Compilation for Dynamically Field-Programmable Qubit Arrays with Efficient and Provably Near-Optimal Scheduling

Related Works

- ***Compiling Quantum Circuits for Dynamically Field-Programmable Neutral Atoms Array Processors***: Utilizes Z3 MST, but lacks scalability and fidelity considerations.
- ***FPQA-C: A Compilation Framework for Field Programmable Qubit Array***: Employs a rule-based algorithm, offering good scalability, but does not achieve the optimal count of 2Q gates.



Overview

- Background: Quantum computing with neutral atoms has advanced rapidly.
- Fidelity:

$$f = (f_1)^{g_1} \cdot \overbrace{(f_2)^{g_2} \cdot (f_{\text{exc}})^{|Q|S-2g_2}}^{\text{two-qubit gate}} \cdot \overbrace{(f_{\text{trans}})^{N_{\text{trans}}}}^{\text{atom transfer}} \cdot \overbrace{\prod_{q \in Q} (1 - T_q / T_2)}^{\textit{decoherence}}.$$

- Significant scalability: experiments with up to 6,100 qubits.
- Challenges in fully leveraging hardware flexibility while respecting constraints.
- The compilation process is broken down into three tasks: scheduling, placement, and routing.

Scheduling

- Scheduling is crucial for determining the sequence of operations.
- The approach uses graph edge coloring for optimal scheduling.
- Ensures near-optimal stage count for two-qubit gates.
- Proven to be at most one stage more than the optimal solution.
- This reduces the fidelity bottleneck in this platform.

Scheduling: Graph Edge Coloring

- Graph edge coloring is used to model the scheduling problem.
- Each edge represents a two-qubit gate.
- Colors represent different stages.
- The goal is to minimize the number of stages while ensuring no two adjacent edges share the same color.

Placement

- Placement refers to assigning qubits to physical locations.
- Optimal placement minimizes the distance between interacting qubits.
- This reduces the need for long-distance routing, which can lower fidelity.
- Placement strategies must handle large qubit arrays efficiently.

Placement Strategies

- Use of heuristic algorithms to find near-optimal solutions.
- Consideration of hardware constraints and qubit connectivity.
- Balancing between computational efficiency and placement quality.

Routing

- Routing involves determining paths for qubits to move during computation.
- Efficient routing algorithms are essential to maintain high fidelity.
- Routing must handle constraints such as limited movement and interaction range.
- Ensuring minimal delay and avoiding congestion are key goals.

Routing Algorithms

- Use of shortest path algorithms to determine efficient routes.
- Dynamic adaptation to changing qubit positions and interactions.
- Balancing the trade-off between routing efficiency and computational overhead.

Results and Comparison

- The compiler, Enola, shows significant improvements in performance.
- Achieves 3.7X stage reduction compared to existing works.
- Demonstrates 5.9X improvement in fidelity on benchmark sets.
- Highly scalable, capable of compiling circuits with up to 10,000 qubits within 30 minutes.
- Outperforms the current state of the art, OLSQ-DPQA.

Performance Metrics

- Stage count reduction: Enola vs. OLSQ-DPQA.
- Fidelity improvement: Quantitative comparisons.
- Scalability: Handling large-scale quantum circuits efficiently.

Conclusion

- The compilation process for dynamically field-programmable qubit arrays involves scheduling, placement, and routing.
- The method provide near-optimal solutions for scheduling and efficient strategies for placement and routing.
- Enola compiler achieves significant improvements in stage reduction and fidelity.
- Future work includes further optimization and exploring additional constraints.
- Open source availability: `https://github.com/UCLA-VAST/Enola`