UCLA Samueli Computer Science

Qubit Mapping and Scheduling for Quantum Computing: Gap Analysis and Optimal Solutions

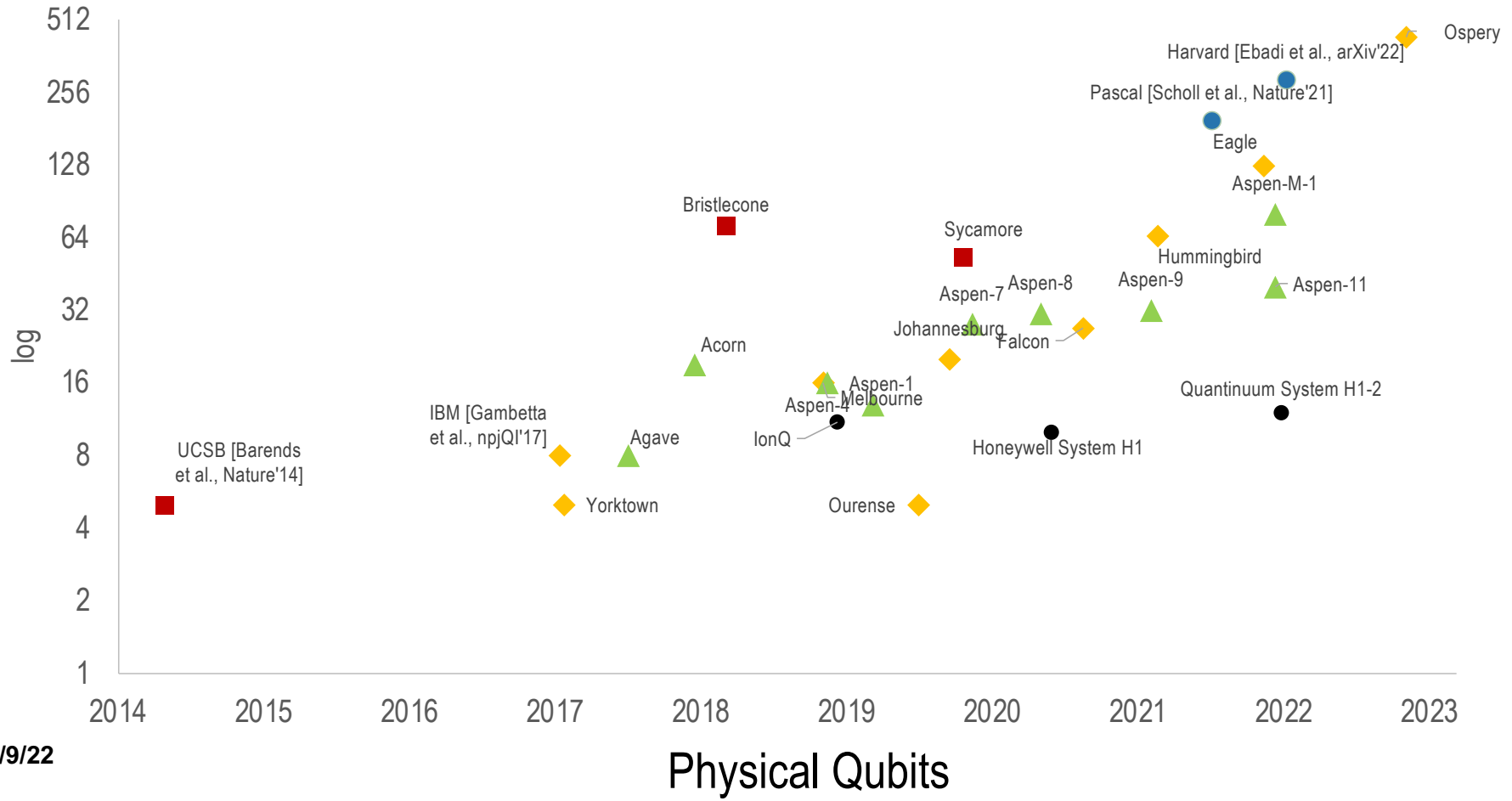
Jason Cong

UCLA Computer Science

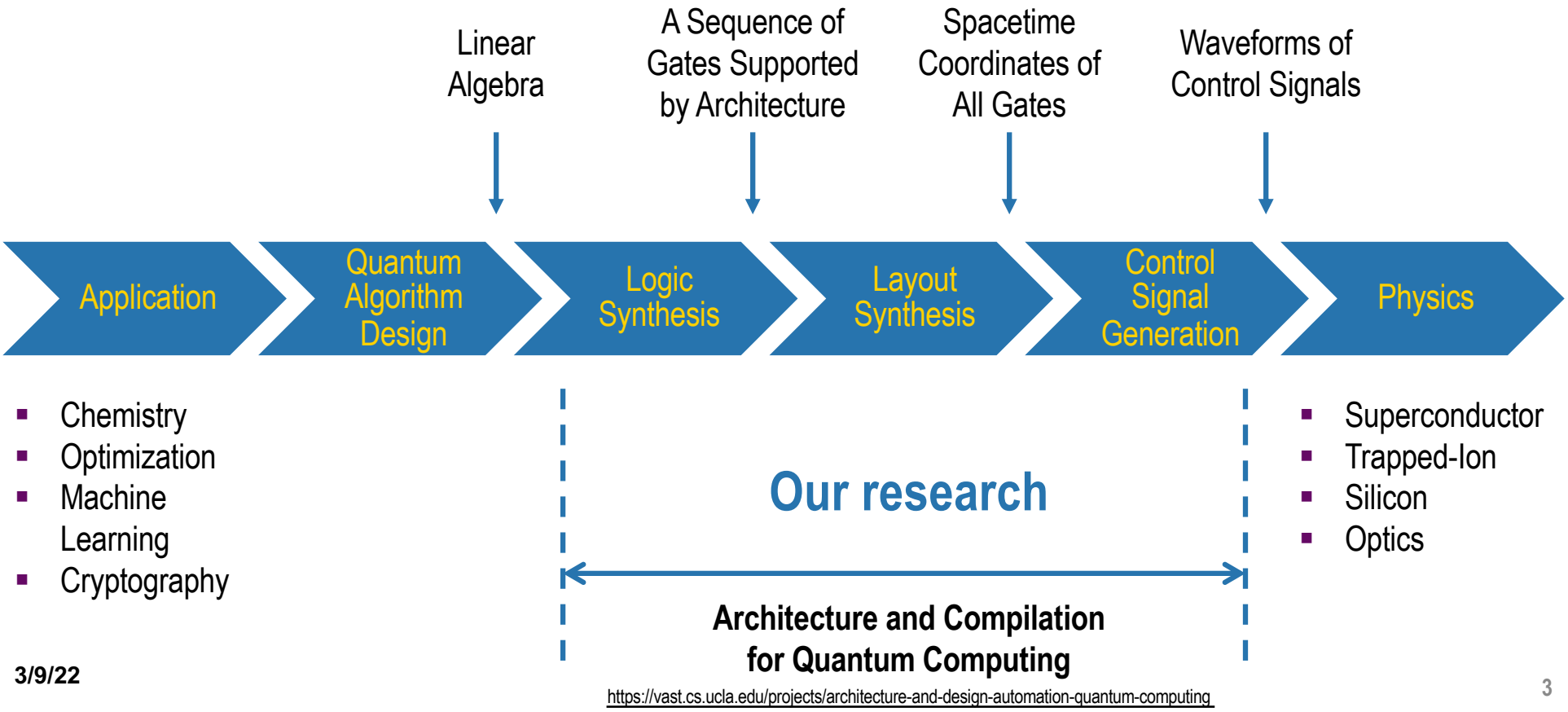
PhD Students: Daniel Bochen Tan, Wan-Hsuan Lin, and Jason Kimko

Other Collaborators: Murphy Niu (Google Quantum), Dolev Bluvstein and Mikhail D. Lukin (Harvard Physics)

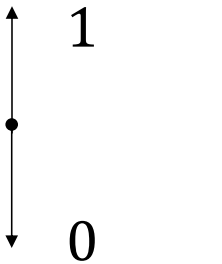
Advances of Quantum Computing



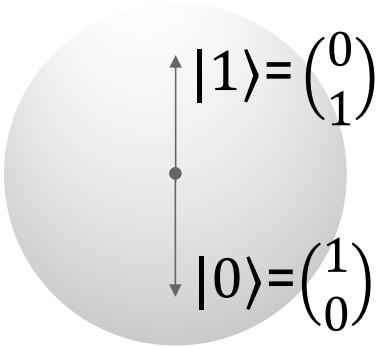
Flow of Quantum Computing



Gate Model of Quantum Computation: Qubits



A bit is either 0 or 1



A qubit can be visualized as a point on the Bloch sphere.

I.e., any unit vector in the Hilbert space spanned by basis vectors |0> and |1>.

$$00 \quad 01 \quad 10 \quad 11$$

Multiple bits become a bit string.

$$\begin{bmatrix} \alpha \\ \beta \end{bmatrix} \otimes \begin{bmatrix} \alpha' \\ \beta' \end{bmatrix} = \begin{bmatrix} \alpha\alpha' \\ \alpha\beta' \\ \beta\alpha' \\ \beta\beta' \end{bmatrix}$$

Multiple qubits become a vector in a higher dimensional Hilbert space.

The dimension grows exponential to the number of qubits!

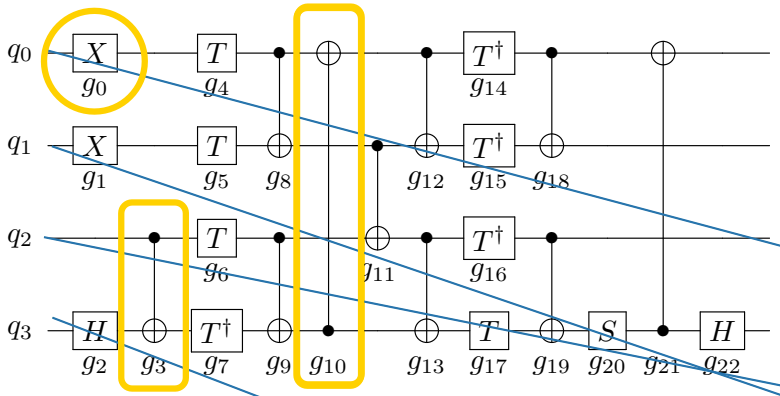
Gate Model of Quantum Computation: Quantum Gates

- **Single-qubit gate**
  - "Bitflip" gate X: $\begin{pmatrix}\alpha\\ \beta\end{pmatrix} \mapsto \begin{pmatrix}\beta\\ \alpha\end{pmatrix}$ i.e., $\begin{pmatrix}0 & 1\\ 1 & 0\end{pmatrix}$
  - Hadamard gate $H = \frac{1}{\sqrt{2}}\begin{pmatrix}1 & 1\\ 1 & -1\end{pmatrix}$
  - Phase shift gate $R_\phi = \begin{pmatrix}1 & 0\\ 0 & e^{i\phi}\end{pmatrix}$
    - $P \equiv R_{\frac{\pi}{2}},\ T \equiv R_{\frac{\pi}{4}}$
  - Qiskit U_3 gate $U3(\theta, \phi, \lambda) = \begin{pmatrix}\cos(\theta/2) & -e^{i\lambda}\sin(\theta/2)\\ e^{i\phi}\sin(\theta/2) & e^{i(\phi+\lambda)}\cos(\theta/2)\end{pmatrix}$
- **Two-qubit gate**
  - Controlled-not gate CX:

$$\begin{pmatrix}\alpha\\ \beta\\ \gamma\\ \delta\end{pmatrix} \mapsto \begin{pmatrix}\alpha\\ \beta\\ \delta\\ \gamma\end{pmatrix}$$

- All gates are unitary: $AA^+ = I$
- **The Solovay-Kitaev Theorem:** the gate set $\{H, P, T, CX\}$ is universal for quantum computing! [Nielsen&Chuang, QCQI]

Qubit Mapping and Scheduling (Layout Synthesis) for QC (LSQC)



Input Circuit of Adder

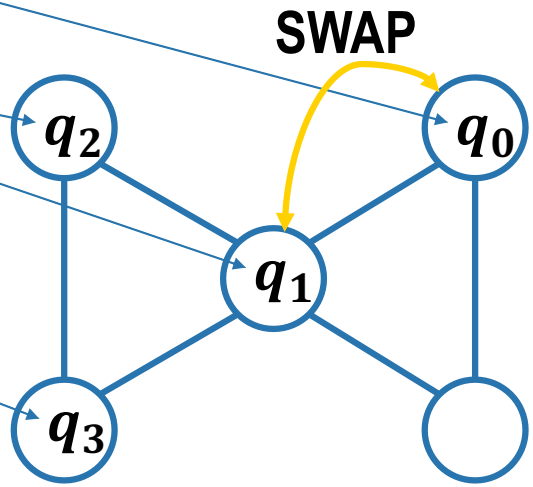
```
# Input quantum program
x q[0];
x q[1];
h q[3];
cx q[2], q[3];
t q[0];
…
```

† means Hermitian conjugate, which is straightforward once we have the original gate implementation.

CX on a pair of adjacent qubits, OK.

CX on a pair of non-adjacent qubits!

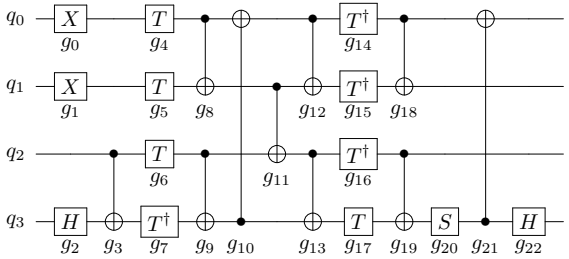
Insert SWAP gate to change the mapping



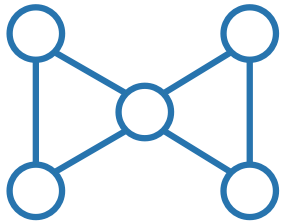
Coupling Graph of IBM QX 2

Layout Synthesis for Quantum Computing (LSQC)

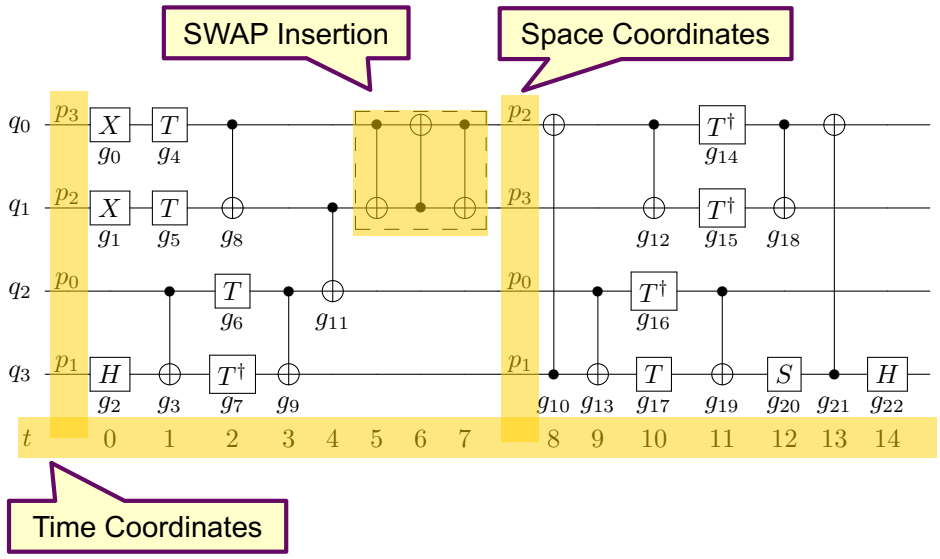
- **Input: quantum circuit/program, coupling graph**
- **Output: spacetime coordinates of all gates, including inserted SWAPs**
- **Objectives: depth, additional SWAP count, fidelity, …**
- **Constraints:**
  - Execute all gates
  - Respect dependencies



Input Circuit of Adder



Coupling Graph of IBM QX 2



Outline

- Introduction
- Gap analysis for quantum compilation
- Optimal layout synthesis for quantum computing (OLSQ)
- Applications to quantum architecture customization
- Compilation for reconfigurable atom array (OLSQ-RAA)

Landscape of QC Layout Synthesis in Early 2020

- **Layer-by-layer:**
  - [Maslov et al., TCAD'08], [Zulehner et al., DATE'18]: lookahead search guided by heuristic cost function
  - [Shafaei et al., ASPDAC'14]: optimize the 'total distance'
- **Gate-by-gate:**
  - [Siraichi et al., CGO'18]: heuristic search for min #SWAPs
  - [Wille et al., DAC'19]: optimize #SWAPs
- **Use dependency:**
  - [Murali et al., ISCA'19]: optimize fidelity upper bound
  - [Li et al., ASPLOS'19]: bi-directional search with cost function concerning both #SWAPs and depth
- **Industry tools: Quilc, Qiskit, t|ket>, Cirq, …**

Are they good enough?

Construction of Placement Examples with Known Optimal (PEKO) Wirelength [Chang et al., TCAD'04]

- **Up to 2 million placeable objects**
  - *Initial WL gap: 1.6x ~ 2.5x (2003)*
- **Multiple EE Times articles coverage, e.g.**
  - Placement tools criticized for hampering IC designs [Feb'03]
- **Many downloads from our website**
  - Cadence, IBM, Intel, Magma, Mentor Graphics, Synopsys, …
  - CMU, MIT, SUNY, UCB, UCSB, UCSD, UIC, UMichgan, UWaterloo, …
- **Optimality gap on PEKO was narrowed down to ~20% as of 2007 (from 60% - 150%)**
- **Improvement on real circuits as well**
  - 30+% improvement by mPL placer 2003-06

http://cadlab.cs.ucla.edu/~pubbench

Quantum Mapping Examples with Known Optimal (QUEKO)

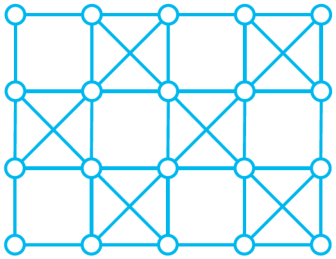
QUEKO: depth and gate count optimal benchmarks tailored to arbitrary devices for LSQC

- Input: device graph, target depth, gate density
- Backbone construction: grow a dependency chain
- Sprinkling: match the gate density profile
- Scrambling: challenge the LSQC tools
- Output: OpenQASM file

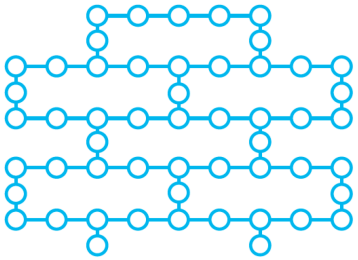
Evaluating Existing LSQC Tools with QUEKO

- **Devices:** Google Sycamore, Rigetti Aspen-4, IBM Q Tokyo, and IBM Q Rochester
- **Circuits:** QUEKO benchmarks
  - **Depth:**
    - 5-45 as near-term feasible,
    - 100-900 as scalability study
  - **Gate density:** profile of Toffoli gate and quantum supremacy experiment [Arute et al., Nature'19]
- **Tools:**
  - **Cirq (Google)**
  - **Qiskit (IBM)**
  - **t|ket⟩ (Cambridge QC, now Quantinuum)**
  - **[Zulehner et al., DATE'18]**

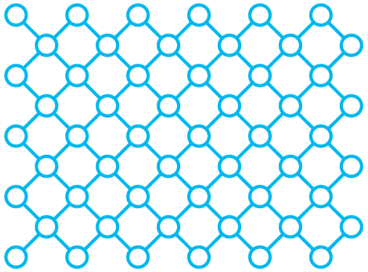
(b) Rigetti's Aspen-4 device graph



(c) IBM's Tokyo device graph

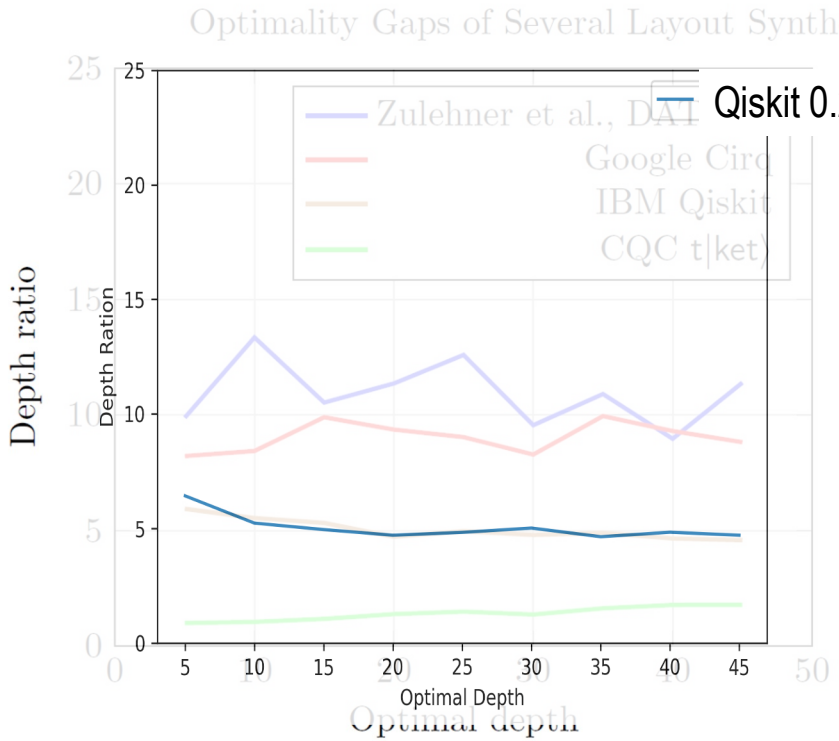


(d) IBM's Rochester device graph

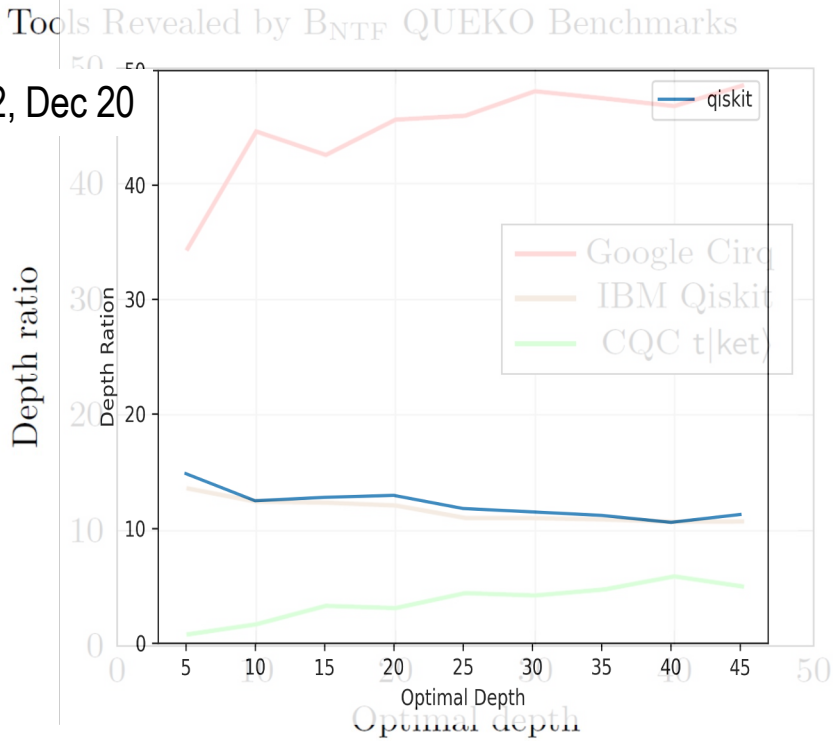


(e) Google's Sycamore device graph

QUEKO Results: Near-Term Feasible



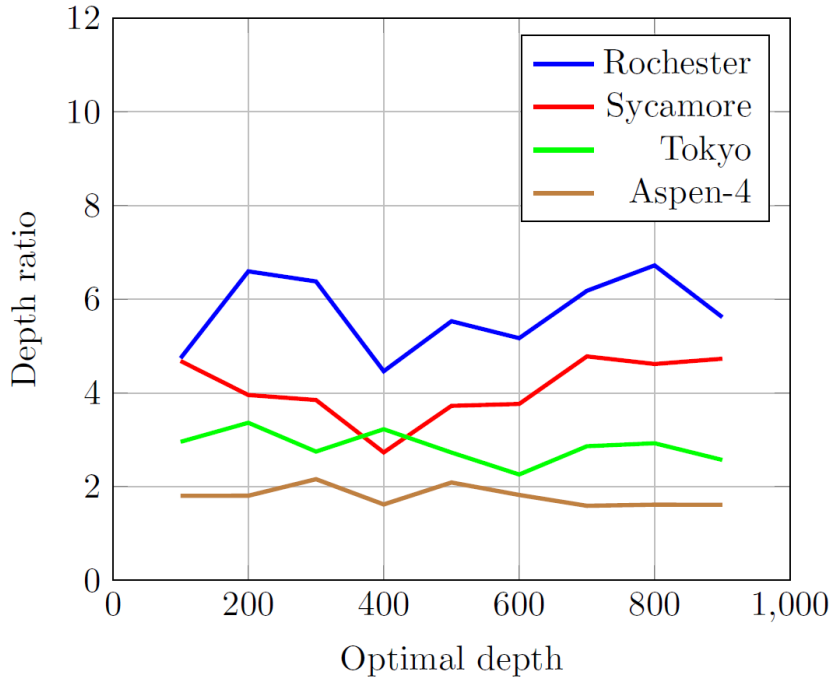
Toffoli gate density
Rigetti Aspen-4 Device



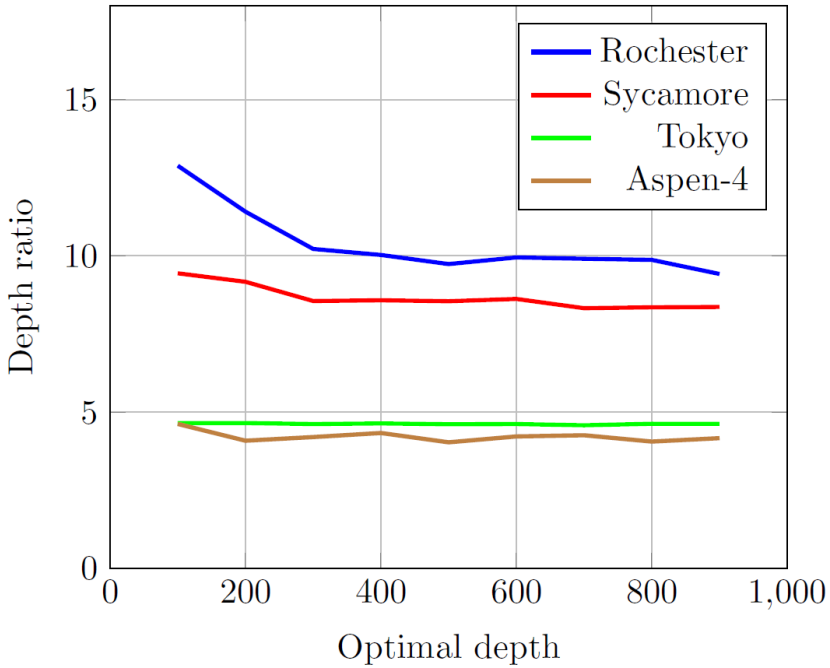
Quantum supremacy experiment gate density
Google Sycamore device

QUEKO Results: Scalability Study

Optimality Gaps of Two Layout Synthesis Tools Revealed by B_{SS} QUEKO Benchmarks



CQC t|ket⟩ Performance



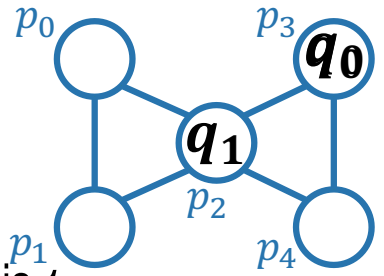
IBM Qiskit Performance

Outline

- Introduction
- Gap analysis for quantum compilation
- Optimal layout synthesis for quantum computing (OLSQ)
- Applications to quantum architecture customization
- Compilation for reconfigurable atom array (OLSQ-RAA)

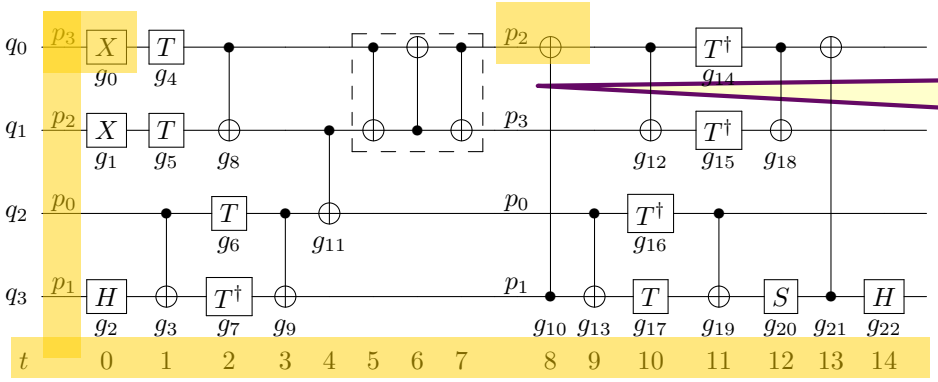
Our Approach: OLSQ (Optimal Layout Synthesis for Quantum Computing)

- **Variables in OLSQ**
  - Time Coordinate t_l for every gate g_l: $t_l = t$ iff. g_l is executed at time t
  - Mapping π_q^t: at time t, logical qubit q is mapped to the physical qubit π_q^t
  - Use of SWAP σ_e^t: $\sigma_e^t = 1$ iff. there is a SWAP on edge e and its last time step is t
  - More efficient encoding of search space*: N^{MT}



Mapping of logical qubit q_0

$\pi_0^0 = 3,$
$\pi_0^1 = 3,$
...
$\pi_0^8 = 2,$
...
$\pi_0^{14} = 2$



Result of OLSQ

The SWAP insertion $\sigma_{(p_2,p_3)}^7 = 1$

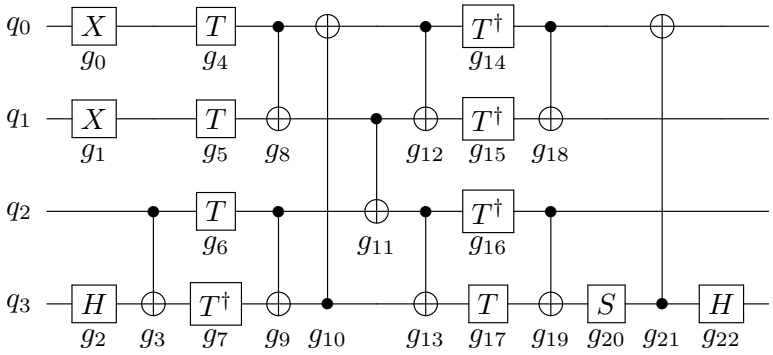
For all other e's and t's: $\sigma_e^t = 0$

*N physical qubits,
M logical qubits,
T time steps

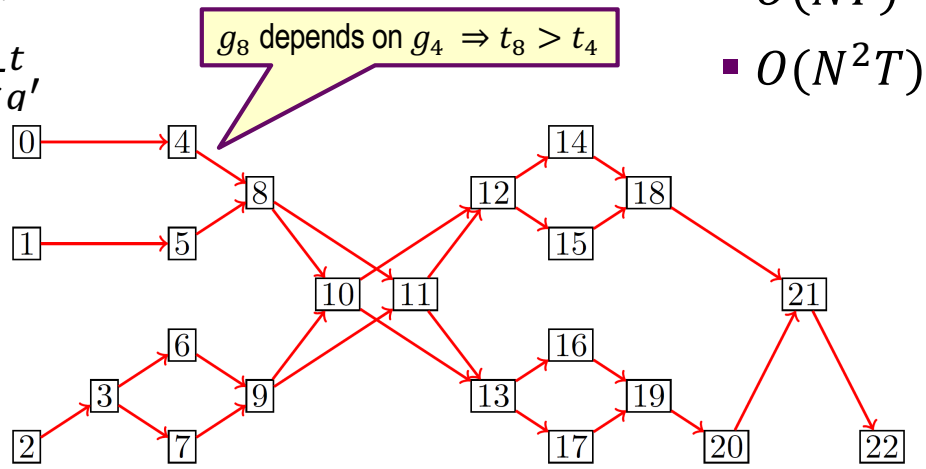
OLSQ: Constraints (Examples)

- **Validity**
  - Valid mapping targets: $\forall t, q \quad \pi_q^t \in P$ (all nodes in the coupling graph G)
  - Valid time coordinates: $\forall l \quad 0 \leq t_l < T$ (increase T if no solution)
  - Valid two-qubit gate scheduling: if $g_l(q, q')$ is a two-qubit gate and $t_l = t$, $(\pi_q^t, \pi_{q'}^t) \in E$ (all edges in G)
- **Dependencies:** if g_l depends on $g_{l'}$, then $t_l > t_{l'}$
- **Injective mapping:** $\forall t, q, q' \quad q' \neq q \Rightarrow \pi_q^t \neq \pi_{q'}^t$

- $O(LT)$
  - $O(NT)$
  - $O(L)$
  - $O(LT)$
- $O(NT)$
- $O(N^2T)$

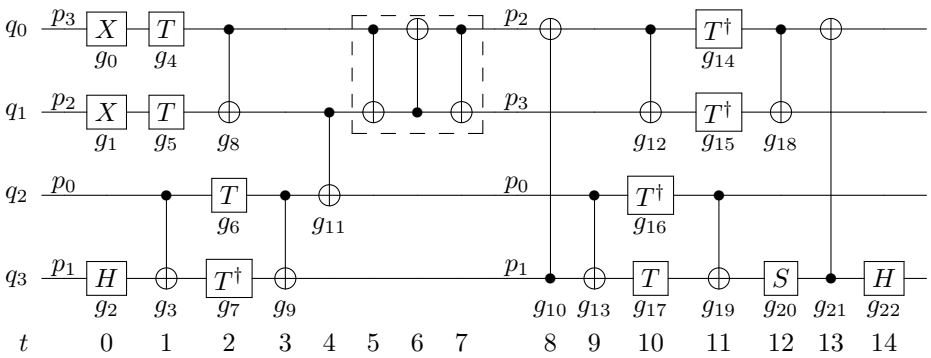


Input Circuit of Adder



OLSQ Optimization Objectives

- **Depth = max t_l**
  - Iterative increase depth bound T : $\bigwedge_{g_l \in G} t_g \leq T$
- **#SWAP = $\sum \sigma_e^t$**
  - Iterative decrease SWAP bound S: $\sum_{\substack{0 \leq t \leq T \\ e \in E}} \sigma_e^t \leq S$



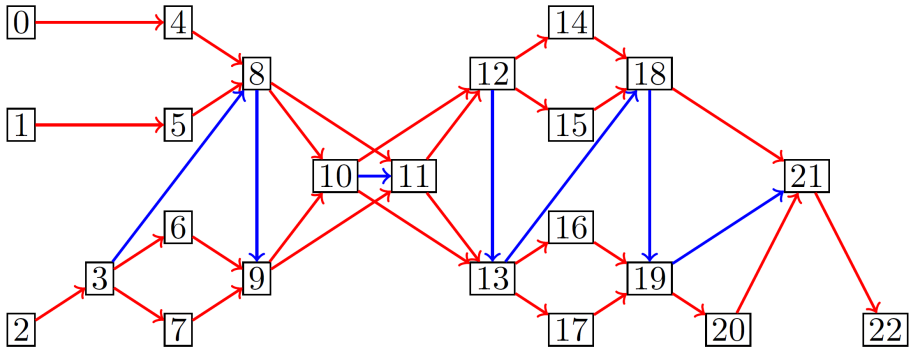
Result of OLSQ

Solved Using SMT (Satisfiability Modulo Theories)

- **SAT (Boolean Satisfiability): given a conjunctive normal form, whether there is an assignment such that it is true. E.g.,**
  - $a \wedge (\bar{a} \vee b) \wedge c$
  - **Solution**: $a = b = c = \mathbf{True}$
- **SMT generalizes SAT to more complex formulas involving real numbers, integers, lists, arrays, bit-vectors, etc. E.g.**
  - $a \coloneqq x + y < 3,\ b \coloneqq x < 4 - y,\ c \coloneqq x > 0$.
  - Then, $x = y = 1$ makes the model satisfiable.
- **SMT is very expressive, widely used in compilation, programming language, formal verification, etc.**
- **There are efficient SMT solvers, such as Z3 (and we further customize for OLSQ)**

Key Advantages of OLSQ

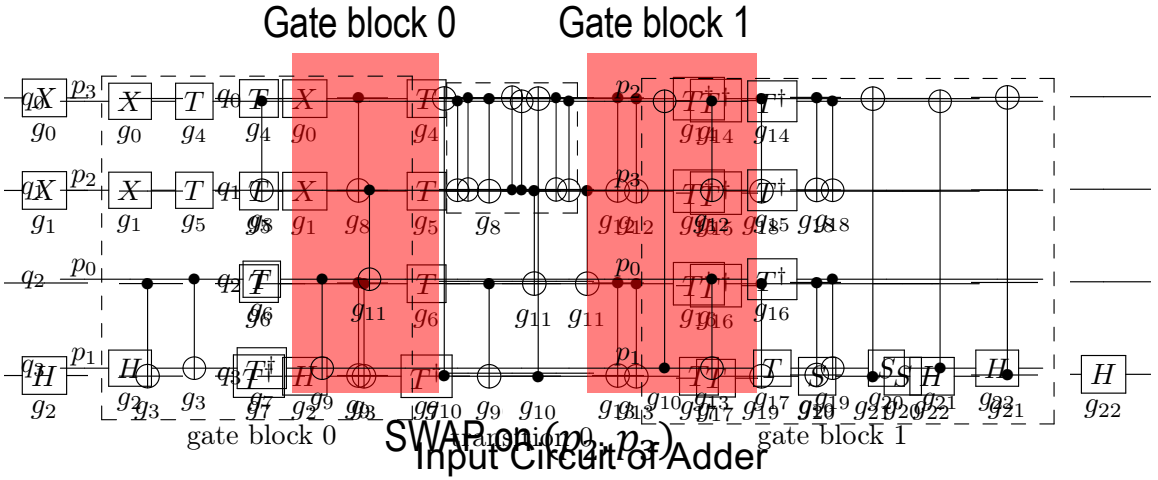
- **Efficiency: $O(NT)$ vars & $O(NTL)$ constraints versus $O(L_2N!)$ & $O(L_2MN!)$**
- **Complexity result: a polynomial certificate → quantum LS is in NP**
- **Optimality: independent from input gate order**



Side-effect of gate-by-gate processing by [Wille et al., DAC'19]

Transition-Based OLSQ

- **Motivation:** many mapping variables are redundant in the lack of SWAPs.
- **Solution:** gate blocks + transitions.
- **Variables:** mapping, spacetime, SWAP *for each block* instead for each time step
  - 2 blocks versus 14 time steps
- After SWAP insertion, we can use ASAP (as soon as possible) scheduling



Input Circuit of Adder

Summary of Constraints for TB-OLSQ

Constraints	OLSQ	[Wille et al., DAC'19]
Validity	$O(LT)$	
Injective Mapping	$O(N^2T)$	
Dependency	$O(NT)$	
No Overlap with Other SWAPs	$O(NT)$	
No Overlap with Original Gates	$O(NTL)$	
Mapping transformed by SWAPs	$O(N^2T)$	
In total	$O(NTL)$	$O(L_2MN!)$

E.g., for QAOA circuits, TB-OLSQ can reduce variables by 49% and constraints by 84%!

TB-OLSQ Evaluation

- Comparison with OLSQ >400x speedup (geomean)

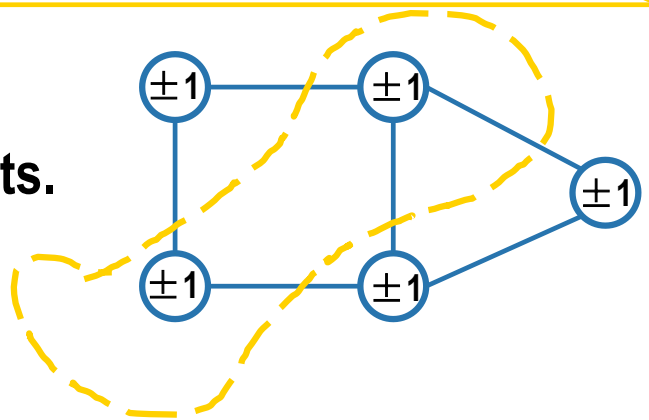
Benchmarks
Small circuits to verify optimality
Larger arithmetic circuits
QUEKO circuits

- A more recent work [Zhang et al., ASPLOS'21] uses A* search with an admissible heuristic, which runs faster with depth-optimal solutions (but cannot optimize other objectives, e.g. fidelity).

* TriQ is from [Murali et al., ISCA'19]

Quantum Approximate Optimization Algorithm (QAOA)

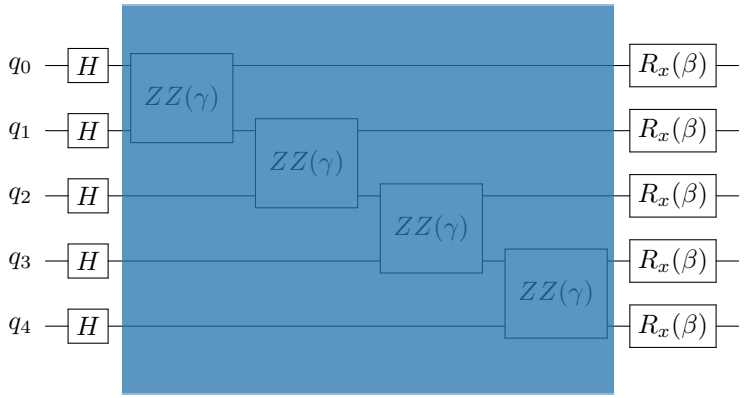
- Aiming optimization with binary variables
- Quantize the problem by changing variables to qubits.
- Example: MAX-CUT problem on $G = (V, E)$
- Assign ± 1 variables z_i to vertices

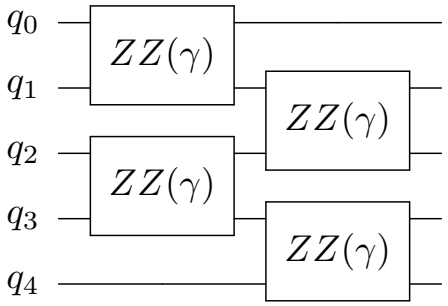


- MAX-CUT = Maximize $\sum_{(v_j, v_k) \in E} \frac{1 - z_j z_k}{2}$
- $z_j z_k$ has a corresponding two-qubit gate, ZZ-Phase.

The ZZ-Phase Gate for Each Edge

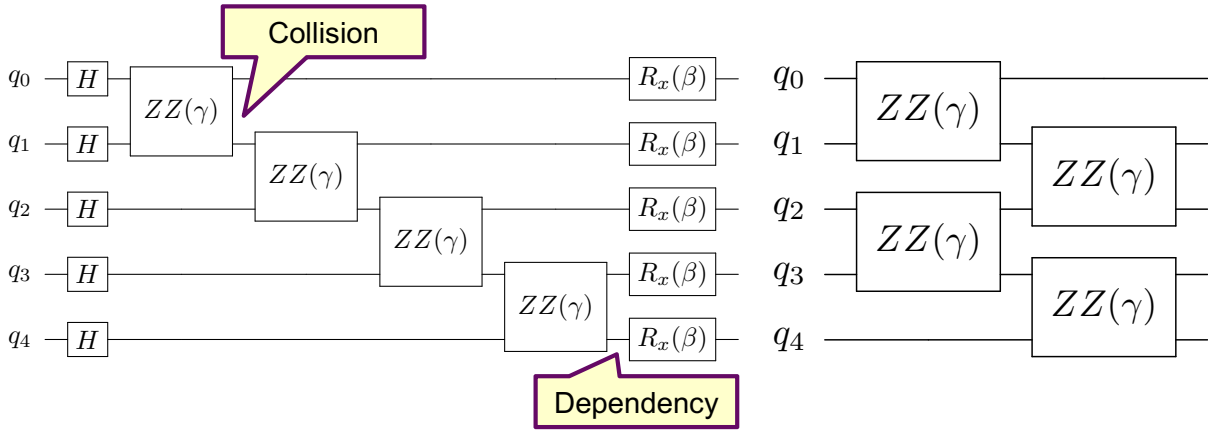
- $\begin{pmatrix} e^{-i\gamma} & 0 & 0 & 0 \\ 0 & e^{i\gamma} & 0 & 0 \\ 0 & 0 & e^{i\gamma} & 0 \\ 0 & 0 & 0 & e^{-i\gamma} \end{pmatrix}$
- **Commutable, i.e., AB=BA, since diagonal**





QAOA-OLSQ

- **Observation:** some ‘dependencies’ are not real, according to commutation.



- **Solution:** make a distinction between dependency and collision
- **Result:** 70% depth reduction, 54% SWAP reduction compared to tket.

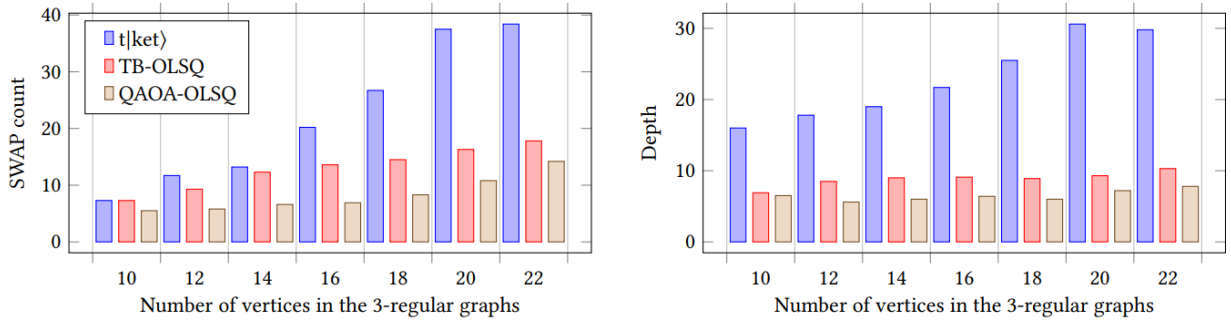
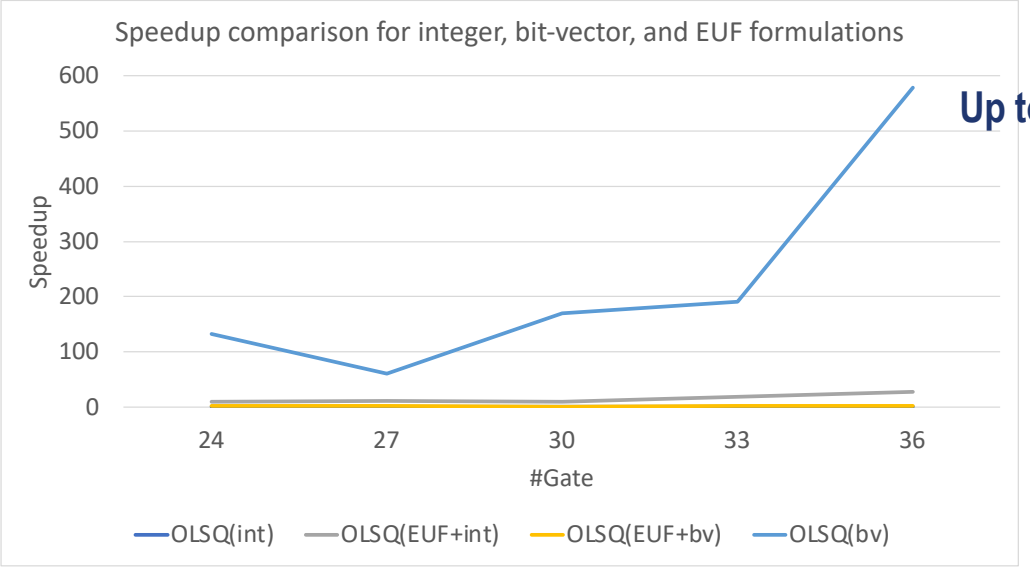


Figure 9. Evaluation of QAOA-OLSQ

Exploration of Different Encoding Schemes in SMT Solver [DAC'23]

- **Variables can be expressed by integers, bit-vectors or equality and uninterpreted functions (EUF)**
  - Invoke different theory solvers



Up to 580x speedup!

Exploration of Different Encoding Schemes in SMT Solver

- **Cardinality constraint can be expressed by AtMost function or conjunction normal form (CNF)**
  - Cardinality constraint: summation of Boolean variables. $\sum_{\substack{0 \le t \le T \\ e \in E}} \sigma_e^t \le S$
  - Use AtMost → invoke pseudo-Boolean solver.
  - Use CNF → invoke SAT solver.

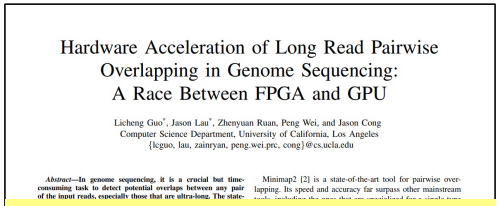
Grid	Qubit	Gate	OLSQ(AtMost)		OLSQ(CNF)		TB-OLSQ(AtMost)		TB-OLSQ(CNF)	
			Runtime(s)	Ratio	Runtime(s)	Ratio	Runtime(s)	Ratio	Runtime(s)	Ratio
5	16	24	404.98	1	65.03	6.23	8.37	48.38	2.59	156.36
	18	27	944.71	1	703.20	1.34	9.25	102.13	2.73	346.05
	20	30	8308.12	1	1493.32	5.56	13.1	634.21	3.08	2697.44
	22	33	TO	--	19037.63	--	12.42	--	3.40	--
	24	36	78555.91	1	10776.14	7.29	12.09	6497.59	3.79	20727.15
Avg. Ratio			1		4.38		261.57		1066.62	

Outline

- Introduction
- Gap analysis for quantum compilation
- Optimal layout synthesis for quantum computing (OLSQ)
- Applications to quantum architecture customization
- Compilation for reconfigurable atom array (OLSQ-RAA)

Benefit of Domain-Specific Architectures in Classical Computing
[From UCLA CDSC since 2009]

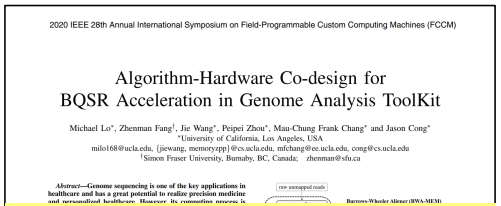
Medical Applications



Hardware Acceleration of Long Read Pairwise Overlapping in Genome Sequencing: A Race Between FPGA and GPU

Licheng Guo*, Jason Lau*, Zhenyuan Ruan, Peng Wei, and Jason Cong
Computer Science Department, University of California, Los Angeles
{lcguo, lau, zainryan, peng.wei.prc, cong}@cs.ucla.edu

Accelerate genome sequencing
- Chaining
- 28x speed-up



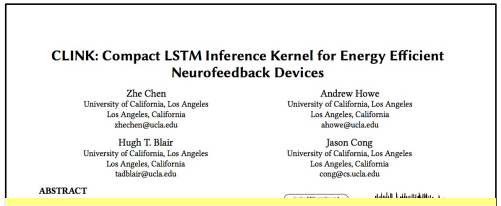
2020 IEEE 28th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM)

Algorithm-Hardware Co-design for BQSR Acceleration in Genome Analysis ToolKit

Michael Lo*, Zhenman Fang†, Jie Wang*, Peipei Zhou*, Mau-Chung Frank Chang* and Jason Cong*
*University of California, Los Angeles, USA
†Simon Fraser University, Burnaby, BC, Canada; zhenman@sfu.ca

Accelerate Genome Analysis
- BQSR
- 8.5x speed-up

Machine Learning

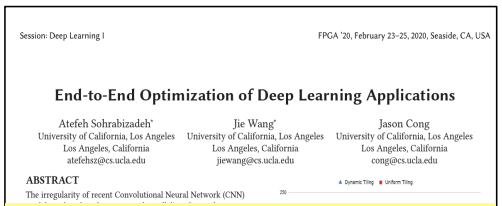


CLINK: Compact LSTM Inference Kernel for Energy Efficient Neurofeedback Devices

Zhe Chen, University of California, Los Angeles, Los Angeles, California, zhechen@ucla.edu
Andrew Howe, University of California, Los Angeles, Los Angeles, California, ahowe@ucla.edu
Hugh T. Blair, University of California, Los Angeles, Los Angeles, California, tadblair@ucla.edu
Jason Cong, University of California, Los Angeles, Los Angeles, California, cong@cs.ucla.edu

ABSTRACT

Accelerate Neural Network
- LSTM-RNN
- 215x energy-efficient



Session: Deep Learning I — FPGA '20, February 23–25, 2020, Seaside, CA, USA

End-to-End Optimization of Deep Learning Applications

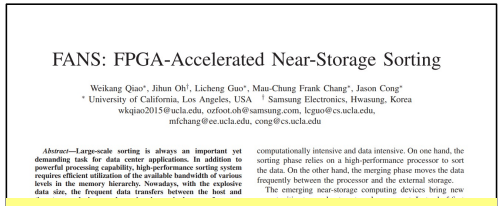
Atefeh Sohrabizadeh*, University of California, Los Angeles, Los Angeles, California, atefehsz@cs.ucla.edu
Jie Wang*, University of California, Los Angeles, Los Angeles, California, jiewang@cs.ucla.edu
Jason Cong, University of California, Los Angeles, Los Angeles, California, cong@cs.ucla.edu

ABSTRACT

The irregularity of recent Convolutional Neural Network (CNN)

Accelerate Neural Network
- End-to-end CNN application
- 11.5x speed-up

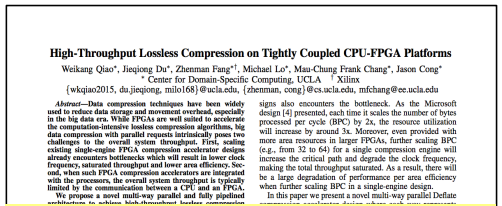
Big-Data



FANS: FPGA-Accelerated Near-Storage Sorting

Weikang Qiao*, Jihun Oh†, Licheng Guo*, Mau-Chung Frank Chang*, Jason Cong*
* University of California, Los Angeles, USA † Samsung Electronics, Hwasung, Korea
wkqiao2015@ucla.edu, ozfoot.oh@samsung.com, lcguo@cs.ucla.edu, mfchang@ee.ucla.edu, cong@cs.ucla.edu

Accelerate sorting
- Merge sort
- 3.2x speed-up

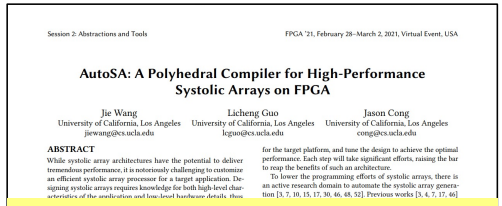


High-Throughput Lossless Compression on Tightly Coupled CPU-FPGA Platforms

Weikang Qiao*, Jieqiong Du*, Zhenman Fang*†, Michael Lo*, Mau-Chung Frank Chang*, Jason Cong*
* Center for Domain-Specific Computing, UCLA † Xilinx
{wkqiao2015, du.jieqiong, milo168}@ucla.edu, {zhenman, cong}@cs.ucla.edu, mfchang@ee.ucla.edu

Accelerate Compression
- Deflate
- 39x speed-up

Systolic arrays & stencils



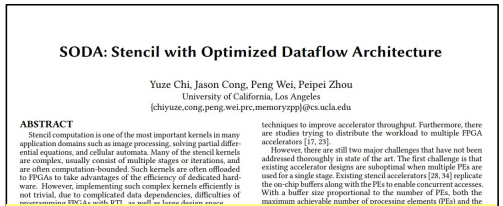
Session 2: Abstractions and Tools — FPGA '21, February 28–March 2, 2021, Virtual Event, USA

AutoSA: A Polyhedral Compiler for High-Performance Systolic Arrays on FPGA

Jie Wang, University of California, Los Angeles, jiewang@cs.ucla.edu
Licheng Guo, University of California, Los Angeles, lcguo@cs.ucla.edu
Jason Cong, University of California, Los Angeles, cong@cs.ucla.edu

ABSTRACT

Accelerate affine-function loops
- GeMM CNN etc.
- 6.8x speed-up



SODA: Stencil with Optimized Dataflow Architecture

Yuze Chi, Jason Cong, Peng Wei, Peipei Zhou
University of California, Los Angeles
{chiyuze,cong,peng.wei.prc,memoryzpp}@cs.ucla.edu

ABSTRACT

Stencil computation

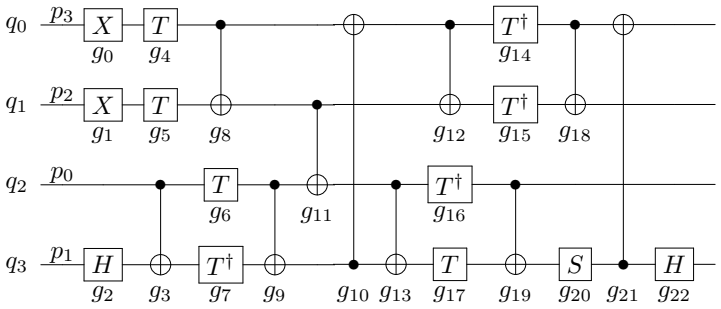
- 2D and 3D stencils
- 3.3x speed-up

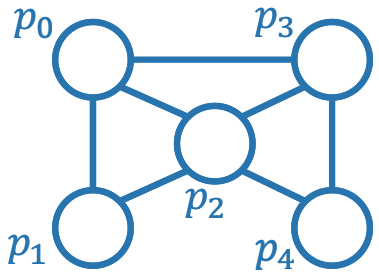
Domain-Specific Quantum Architecture Design?

- Architecture customization has led to the biggest accelerations in classical computing.

How can architecture customization benefit quantum computing?

Improve circuit fidelity by reduce compilation overhead!

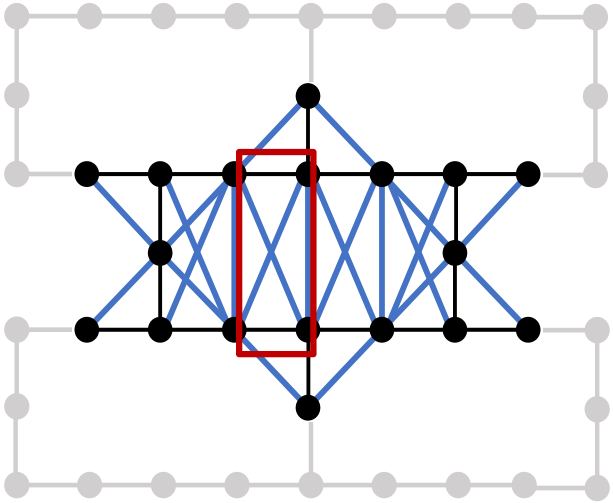




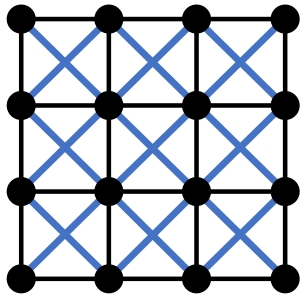
No swap gates is needed!

Example QC Architecture Design Space

- **Definition:** (G_b, E_f, C)
  - G_b : Coupling graph, E_f : Flexible edges, C : Constraints
- **Heavy-hexagon architecture space:**

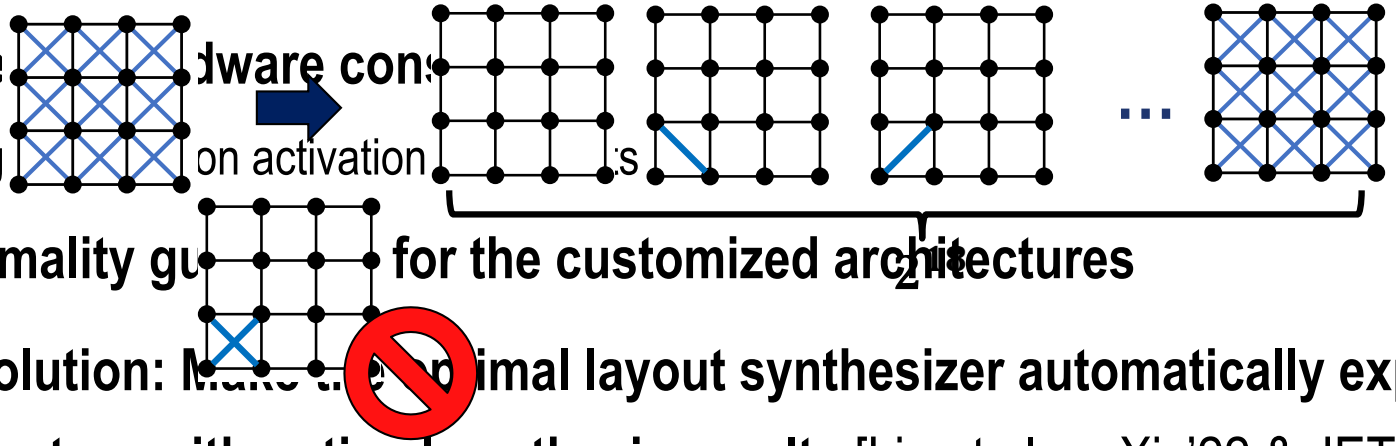


- **Grid architecture space:**



Challenges to Perform Architecture Customization

- **Large design space**
- **Diffe dware cons**
  - E.g on activation s
- **Optimality gu for the customized architectures**



Our solution: M imal layout synthesizer automatically explore the architecture with optimal synthesis results [Lin et al., arXiv'22 & JETCAS'22]

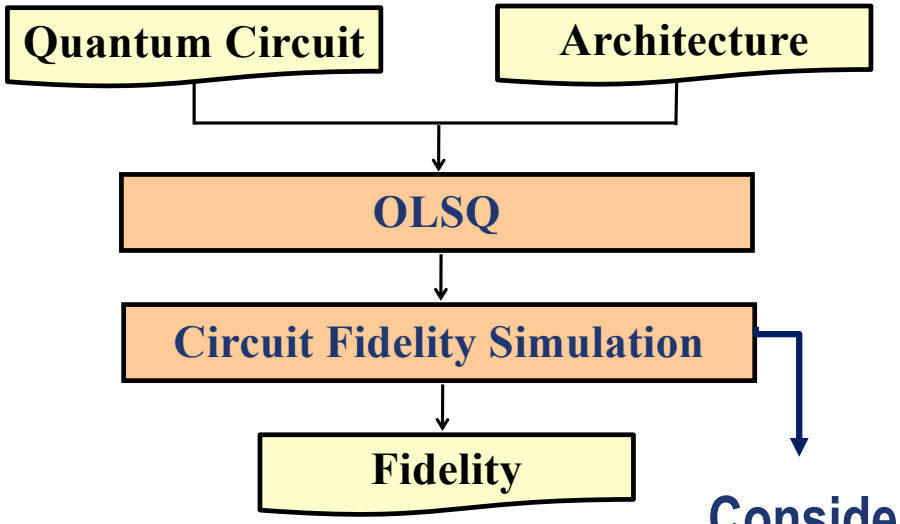
Our Approach

- **Integrate layout synthesis into architecture customization**
  - Choose the formulation of TB-OLSQ due to better scalability and optimality in terms of SWAP count
- **Circuit-related variables and constraints: derived from TB-OLSQ**
- **New addition: architecture-related variables and constraints:**
  - Flexible edge u_e: $u_e = 1$ iff at least one gate is scheduled to be executed on the flexible edge e
  - Limit the number of activated flexible edges: $\sum_{e \in E_f} u_e \leq \alpha$

Reduce (1) hardware fabrication/calibration cost, and (2) potential crosstalk noises

Evaluation Flow

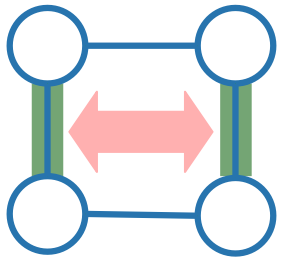
- Utilize a realistic fidelity model to perform the architecture evaluation



Consider

1. gate error
2. crosstalk noise
3. qubit idling error

Crosstalk noise

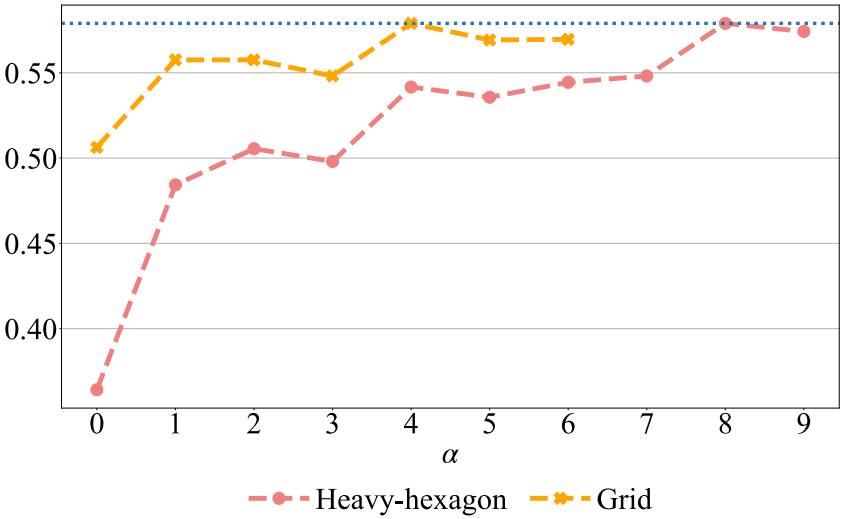


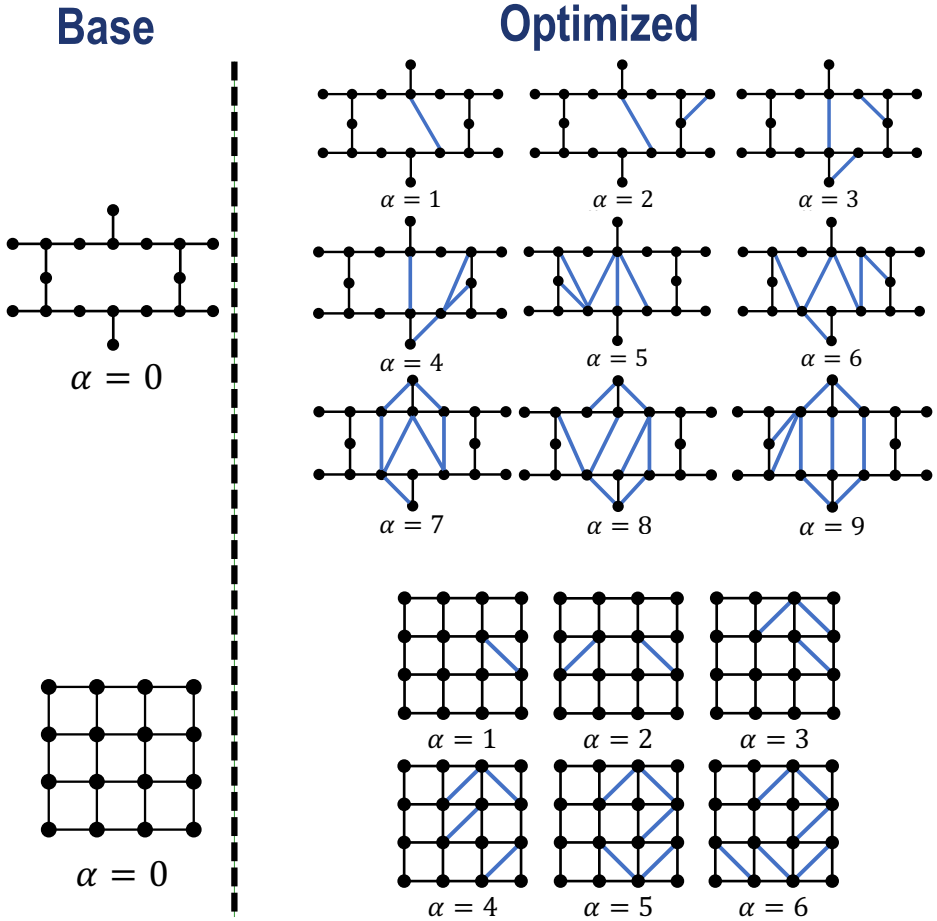
Qubit decoherence

$$|\alpha\rangle \xrightarrow{\text{loss information}} ?$$

Architecture Customization Results for QAOA-10

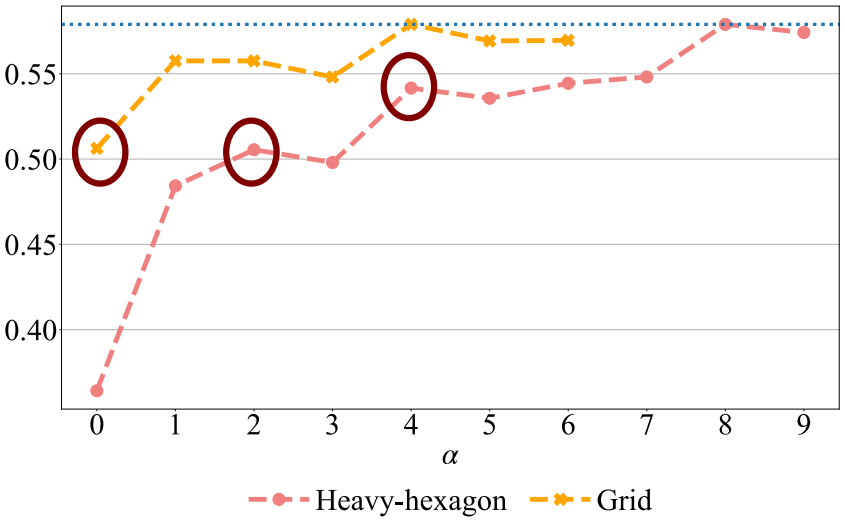
- **Fidelity improvement:**
  - Heavy-hexagon architecture space: 59%
  - Grid architecture space: 14%

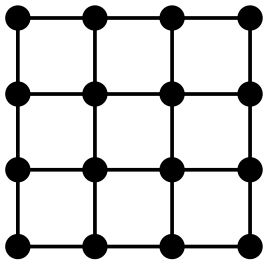




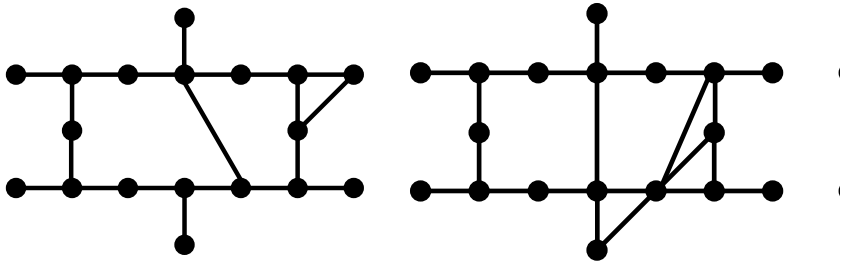
Architecture Customization Results for QAOA-10

- **Fidelity improvement:**
  - Heavy-hexagon architecture space: 59%
  - Grid architecture space: 14%





Base grid architecture
Avg degree: 3



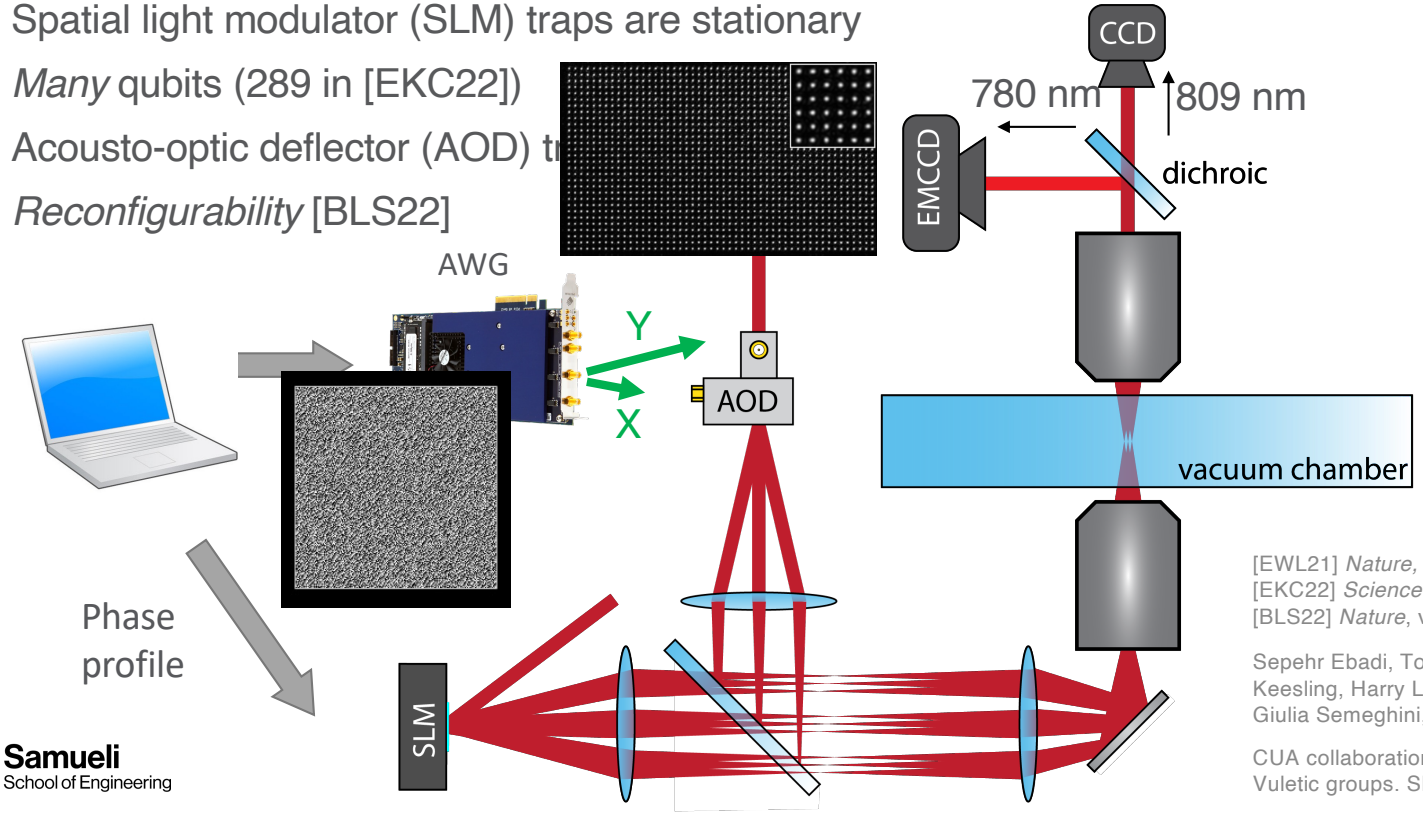
Optimized heavy-hexagon architecture
Avg degree: 2.22, 2.44

Outline

- Introduction
- Gap analysis for quantum compilation
- Optimal layout synthesis for quantum computing (OLSQ)
- Applications to quantum architecture customization
- Compilation for reconfigurable atom array (OLSQ-RAA)

Reconfigurable Atom Arrays (RAA) Platform

- Spatial light modulator (SLM) traps are stationary
- *Many* qubits (289 in [EKC22])
- Acousto-optic deflector (AOD) t
- *Reconfigurability* [BLS22]



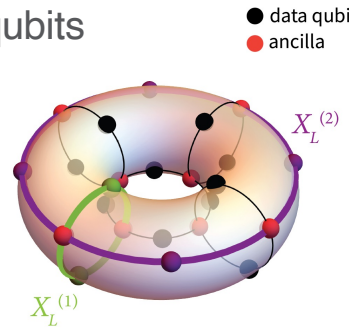
[EWL21] *Nature,* vol. 595, no. 7866
[EKC22] *Science*, vol. 376, no. 6598
[BLS22] *Nature*, vol. 604, no. 7906

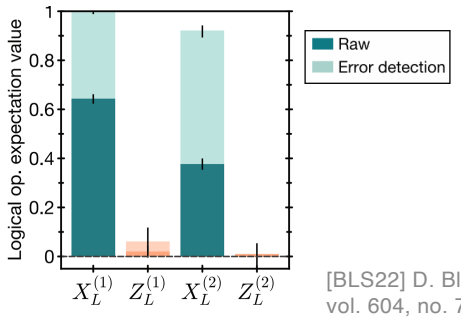
Sepehr Ebadi, Tout Wang, Alexander Keesling, Harry Levine, Ahmed Omran, Giulia Semeghini, Dolev Bluvstein (2020)

CUA collaboration of Lukin, Greiner & Vuletic groups. Slide Credit: M. D. Lukin

RAA Manual Construction Example

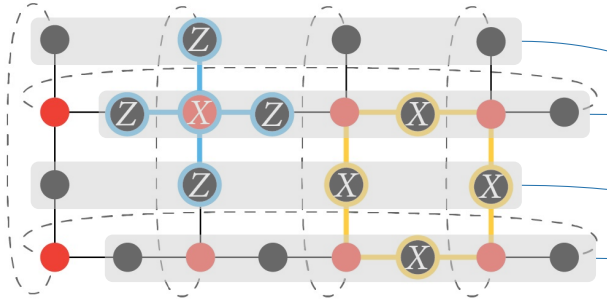
Toric code on 2 qubits

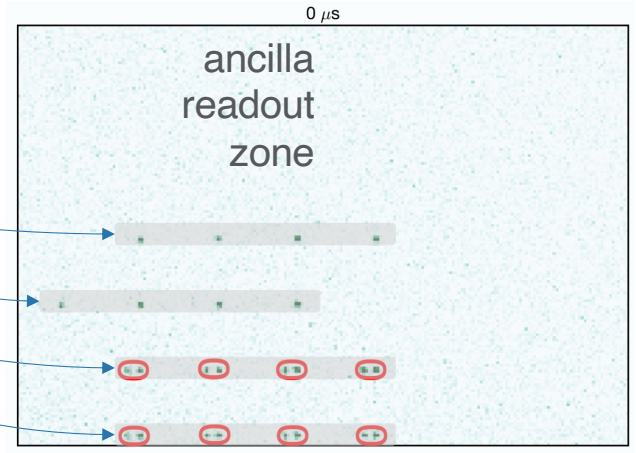




[BLS22] D. Bluvstein et al, *Nature*, vol. 604, no. 7906

Equivalent to graph state with non-local connectivity

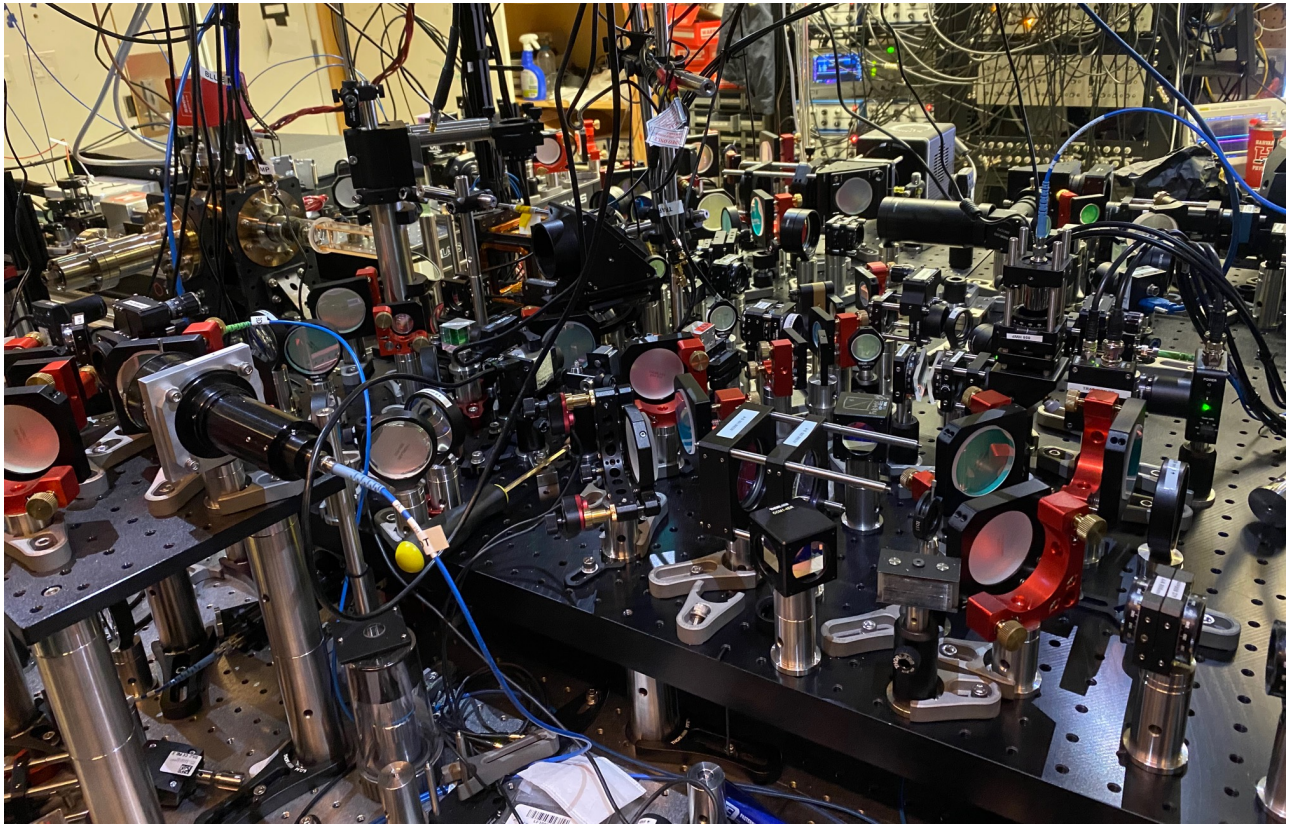




UCLA Samueli School of Engineering

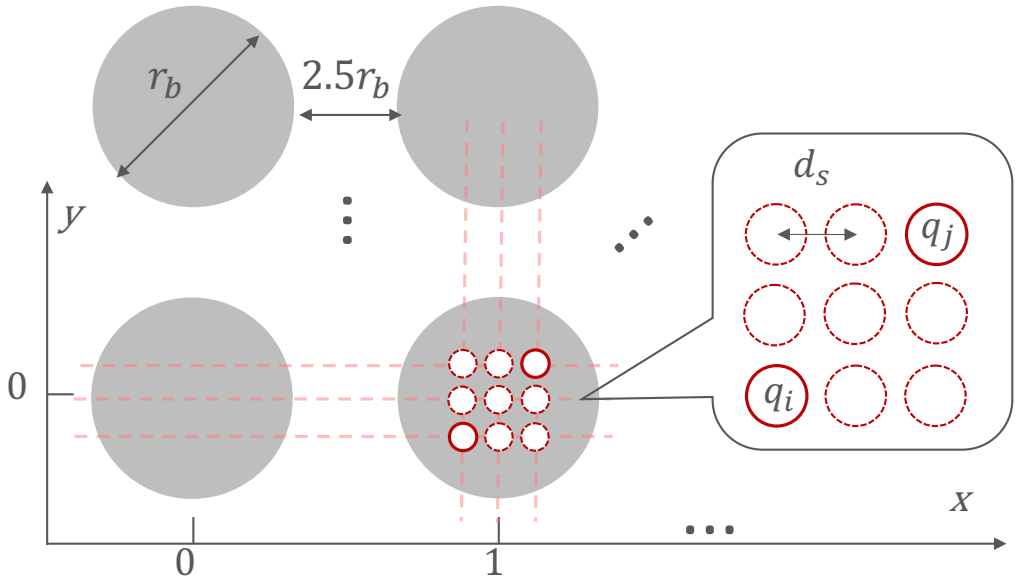
Slide Credit: M. D. Lukin

Reconfigurable Atom Arrays (RAA) Platform



From Lukin's lab

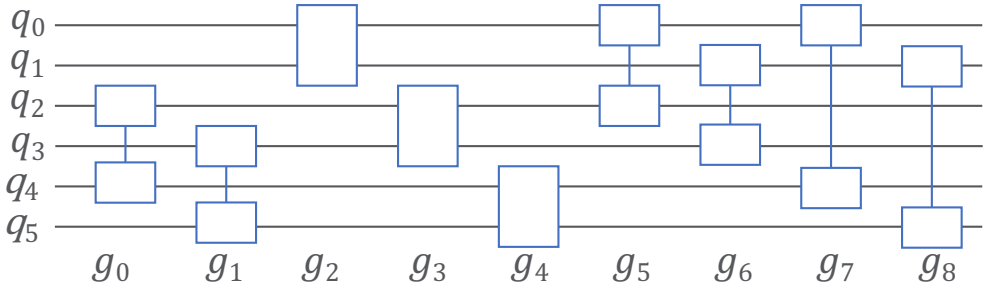
Discretization of Architecture State



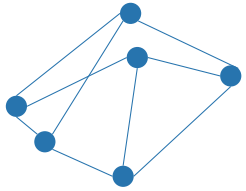
RAA architecture

- Time: stages (when we turn on Rydberg laser) + interpolation for movement
- Sufficient separation: $> 2.5r_b$
- Space: interaction sites (sufficiently separated for parallel two-qubit gates)
  - 0 or 1 SLM per site
  - AOD moves between sites
- 'Stacking' of AOD rows/columns
- Minimal AOD separation: d_s -> maximal row/col stacking factors

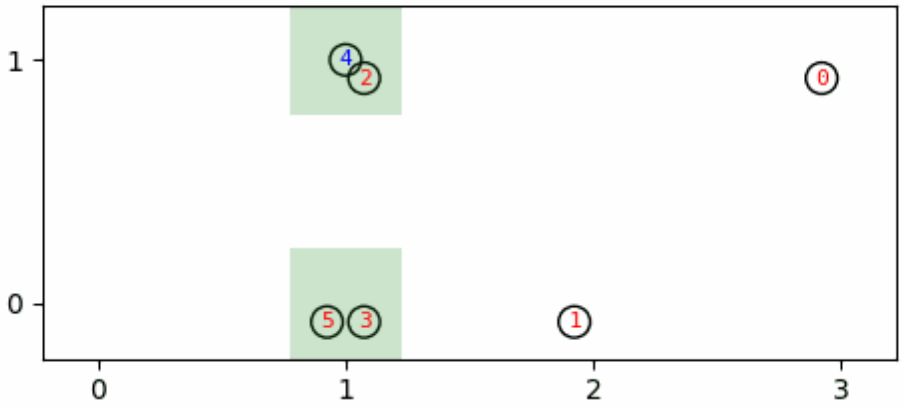
Running Example



Problem unitary $U_C(\gamma)$ in QAOA for 3-regular graph Max-Cut



The 3-regular graph



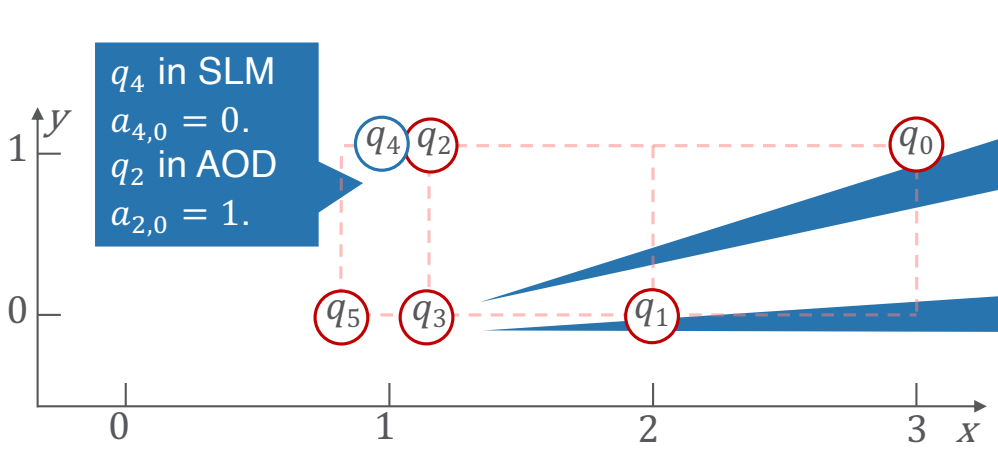
3 SWAPs on Sycamore, none required on RAA.

Formulation: Variables

- Site indices $(x_{i,t}, y_{i,t})$: at stage t, qubit i is at $(x_{i,t}, y_{i,t})$.
- Array indices $a_{i,t}$: $= 0$ if qubit i is in SLM at t, and $= 1$ if it is in AOD.
- AOD col/row indices $(c_{i,t}, r_{i,t})$: qubit i is in AOD col c_i and row r_i at t.
  - These variables only appear in constraints when $a_{i,t} = 1$
- Gate scheduling t_j: gate j is executed at stage t_j.

g_0 on q_2 and q_4
g_1 on q_3 and q_5
g_2 on q_0 and q_1
g_3 on q_2 and q_3
g_4 on q_4 and q_5
g_5 on q_0 and q_2
g_6 on q_1 and q_3
g_7 on q_0 and q_4
g_8 on q_1 and q_5

For gate 0, $t_0 = 0$.



RAA state at stage $t = 0$

Formulation: Constraints

- Valid architecture states:
  - Bound variable values to put qubits in the spacetime we consider;
  - Some constant number of AOD rows/cols can be at the same site.
- Valid movements:
  - Qubits in SLM are stationary in one step;
  - Qubits in AOD are moved by rows or columns;
  - AOD rows/cols cannot change order.
- Valid atom transfer:
  - No atom transfer when a site has both AOD and SLM qubits.
- Gate execution:
  - Dependency or collision (when all gates are commutable);
  - Connectivity: two qubits must be at the same site for a gate;
  - Exact computation: any two qubits avoid each other when there is not a gate.

Optimization Procedure

- Objective: #stages, i.e., # turning on Rydberg laser
  - Main error source is from Rydberg laser
- Lower bound of #stages by gate dependences T_0
- Satisfiability modulo theories (SMT) model: variables + constraints
- SMT solver: given a model, output a solution if satisfiable, or 'UNSAT'

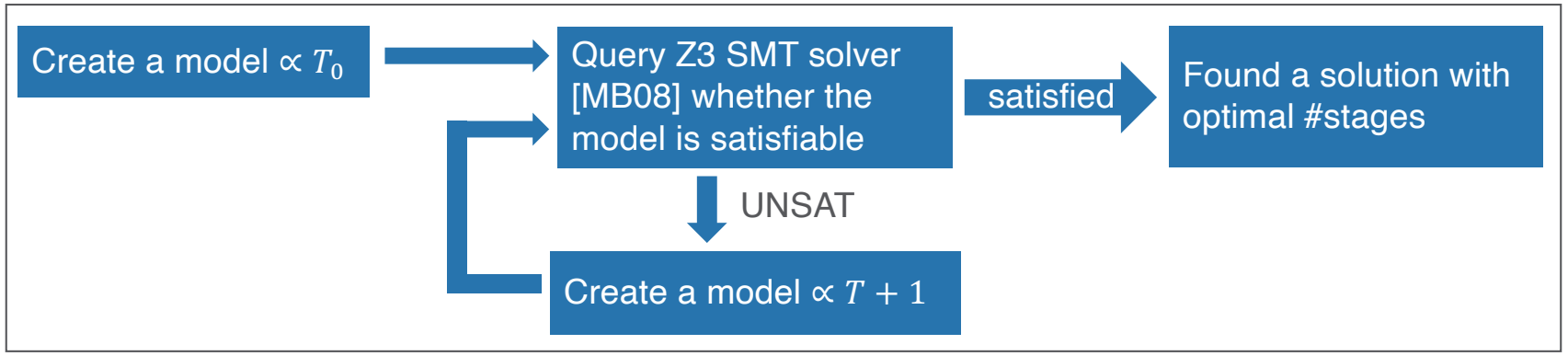
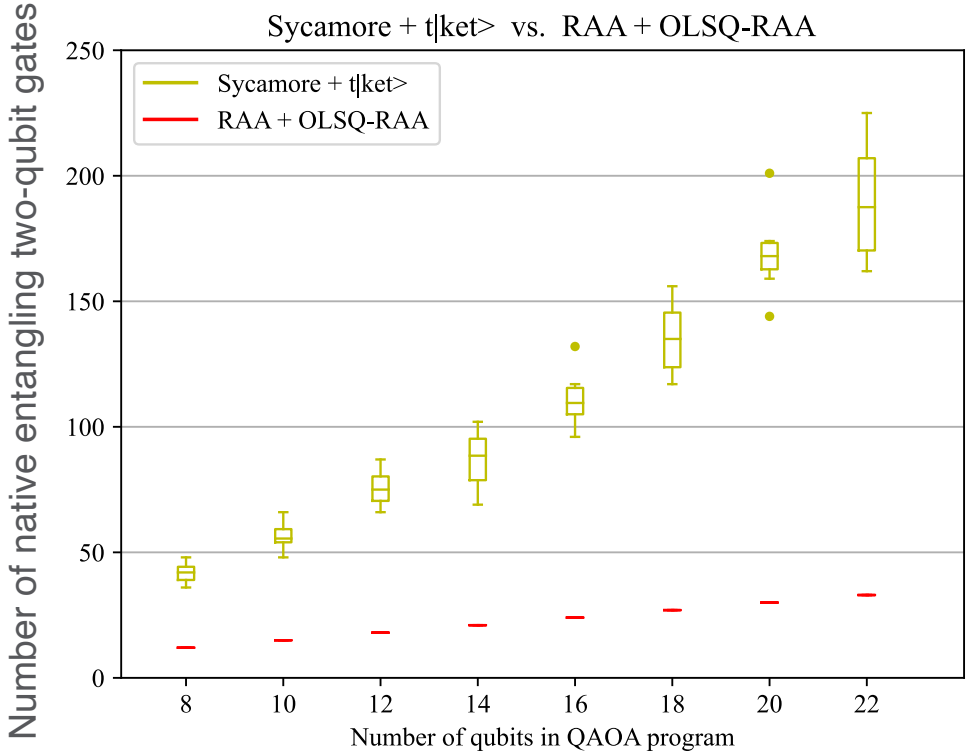


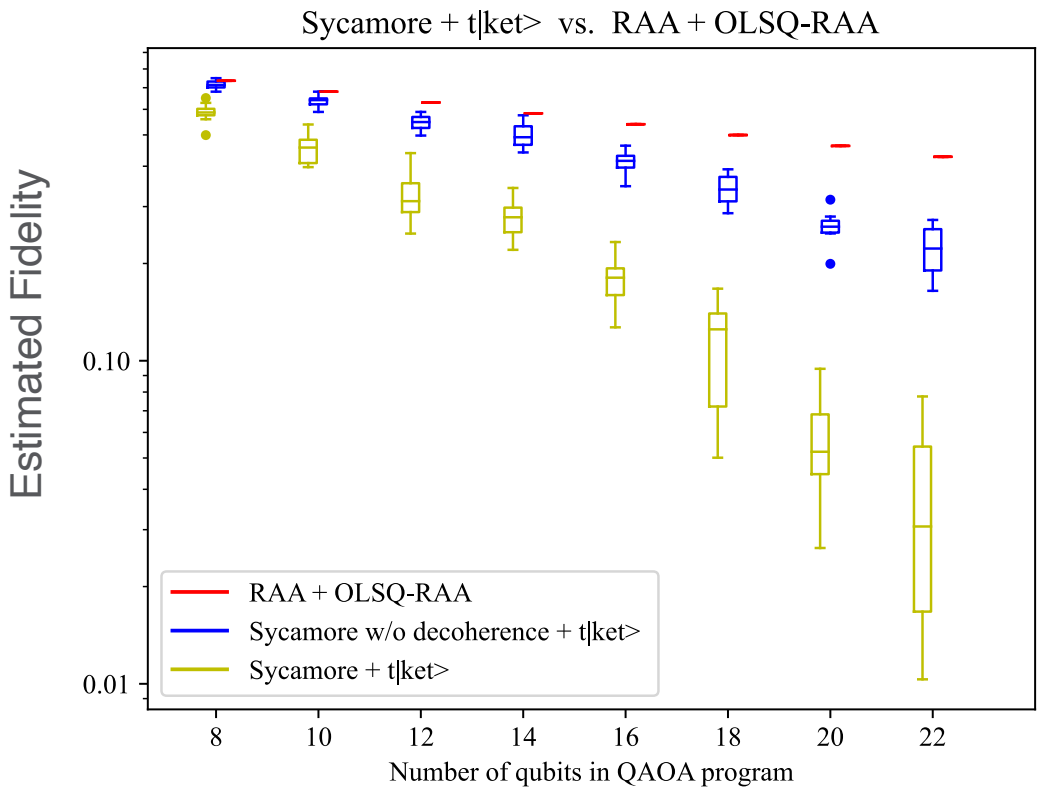
Diagram of OLSQ-RAA (https://github.com/UCLA-VAST/OLSQ/tree/RAA) optimization procedure

OLSQ-RAA: Evaluation on Number of Gates



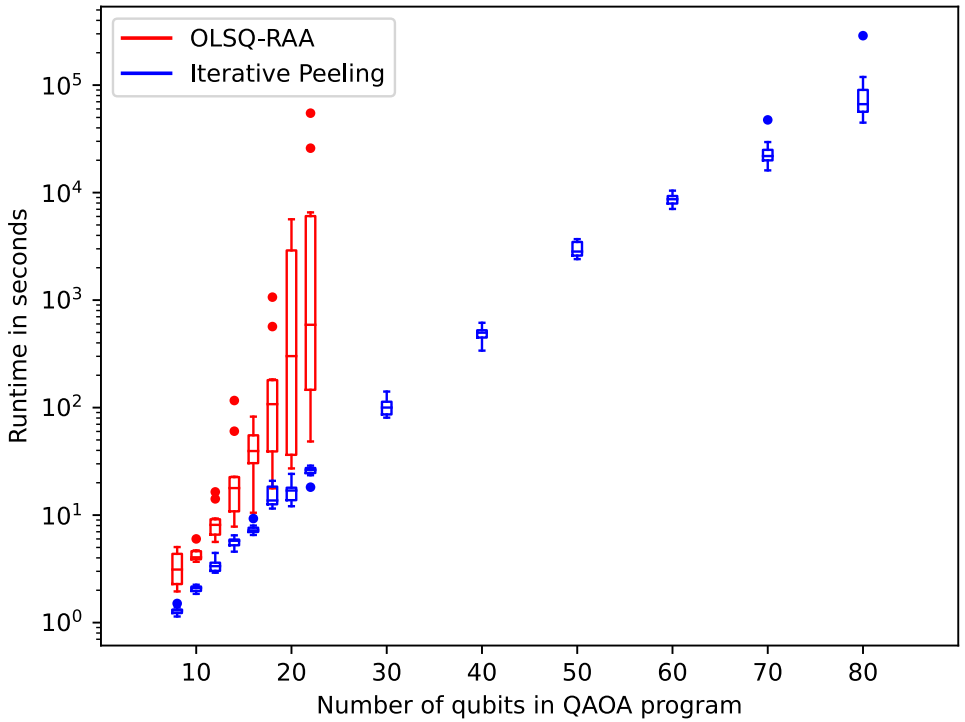
- Sycamore + tlket> represents the previous leading experiment [HSN21].
- For RAA, all the 'routing' is done by array movements.
- 5.72x less gates for QAOA-22.

OLSQ-RAA: Evaluation on Fidelity



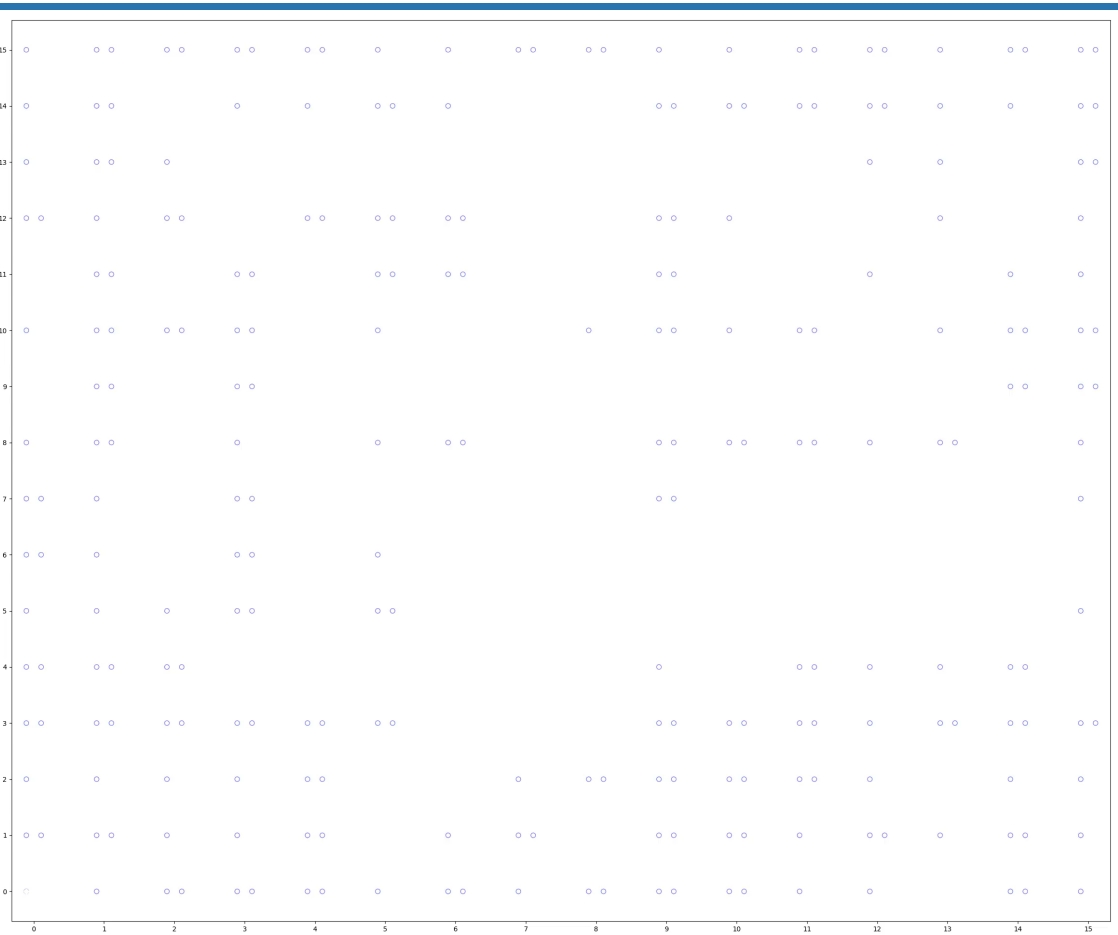
- f_2 two-qubit gate fidelity, T coherence time
- Currently, on QAOA-22, RAA + OLSQ-RAA has 14.4x higher fidelity than Sycamore + t|ket>.
- The pure effect of reconfigurability is 1.96x.

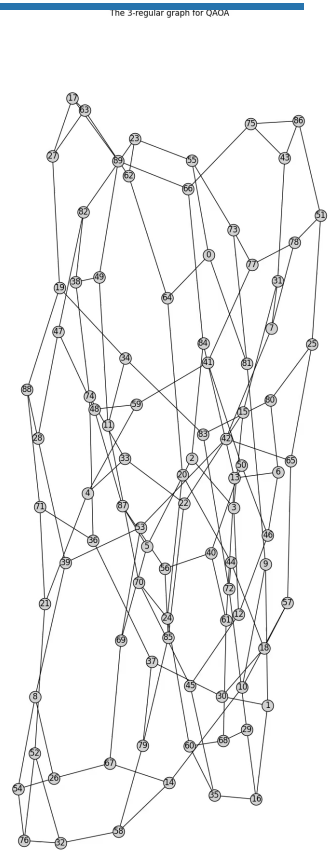
Classical Circuit Design Comes to Help



- Iterative peeling: routing for multilayer classical circuit layouts with a provable performance bound of $1 - 1/e$ [CHS93]
- Solving unit-step SMT models to execute as many CZ gates as possible
- Still exponential, but much faster
- Less variation in size-runtime scaling

90-qubit QAOA example





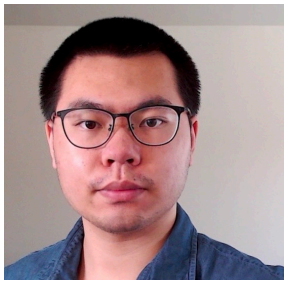
UCLA Samueli School of Engineering

Concluding Remarks

- **There is a great need for better design automation or compilation tools for QC**
  - E.g. as measured by the QUEKO circuits
- **OLSQ provides a framework for optimal solution for layout synthesis**
  - Can be applied to QC devices from different technologies
- **Further opportunities to combine layout synthesis with logic synthesis**
- **Promising results on quantum architecture customization**
- **Many opportunities of applying scalable optimization methodology from VLSI CAD**

Acknowledgements

Supports from the Industrial Partners of the Center for Domain-Specific Computing (CDSC)



Daniel Bochen Tan

Wan-Hsuan Lin

Jason Kimko

Murphy Niu
(Google Quantum)

Dolev Bluvstein
(Harvard Physics)

Mikhail D. Lukin
(Harvard Physics)