Recent Approaches for Efficient Compiling of Quantum Circuits

Michele Amoretti[1,2]

1. Department of Engineering and Architecture - University of Parma, Italy
2. Quantum Information Science @ University of Parma, Italy

www.qis.unipr.it

Contact: michele.amoretti@unipr.it

Quantum Computing and HPC

CINECA, 19 December 2019

Quantum Software @ UniPr

Team

- Michele Amoretti (Associate Professor)
- Davide Ferrari (PhD student)

Topics

- high performance computing (classical and quantum)
- quantum compiling
- quantum networking (protocols for distributed monitoring, anonymity, leader election, etc.)

Source code

https://github.com/qis-unipr

UNIVERSITÀ DI PARMA

QIS.UNIPR.IT
QUANTUM INFORMATION SCIENCE @ UNIPR

Molecular Magnetism Group @ UniPr

MOLECULAR MAGNETISM group

Team

- Stefano Carretta (Full Professor)
- Paolo Santini (Full Professor)
- Elena Garlatti (Research Associate)
- Alessandro Chiesa (Post-doc)
- Roberto De Renzi (Full Professor)
- Francesco Petiziol (Post-Doc)
- Emilio Macaluso (PhD student)
- Simone Chicco (PhD student)
- Luca Crippa (PhD student / IBM)

Topics

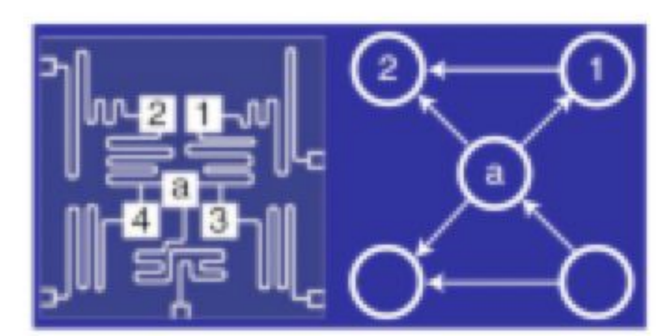

nature physics

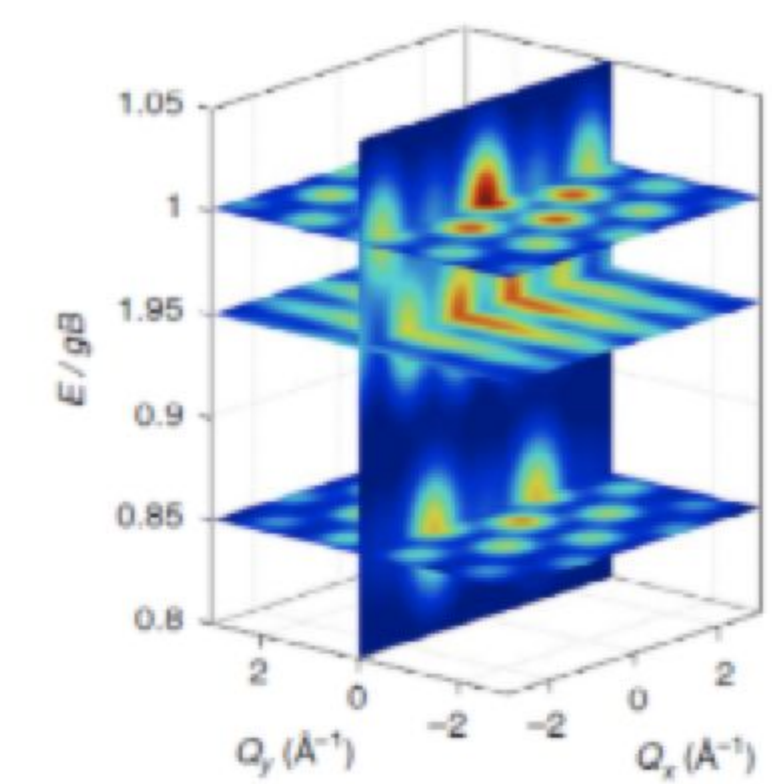
LETTERS

https://doi.org/10.1038/s41567-019-0437-4

Quantum hardware simulating four-dimensional inelastic neutron scattering

A. Chiesa[1,5], F. Tacchino[2,5], M. Grossi[2,3], P. Santini[1], I. Tavernelli[4], D. Gerace[2] and S. Carretta[1*]






JACS JOURNAL OF THE AMERICAN CHEMICAL SOCIETY

Cite This: J. Am. Chem. Soc. 2018, 140, 9814−9818

Coherent Manipulation of a Molecular Ln-Based Nuclear Qudit Coupled to an Electron Qubit

Riaz Hussain,[†] Giuseppe Allodi,[†] Alessandro Chiesa,[†] Elena Garlatti,[†,‡] Dmitri Mitcov,[¶] Andreas Konstantatos,[¶] Kasper S. Pedersen,[¶,§] Roberto De Renzi,[†] Stergios Piligkos,[¶] and Stefano Carretta[*,†,∥]

Multilevel structure exploited to encode a **qubit with embedded Quantum Error Correction**.

Outline

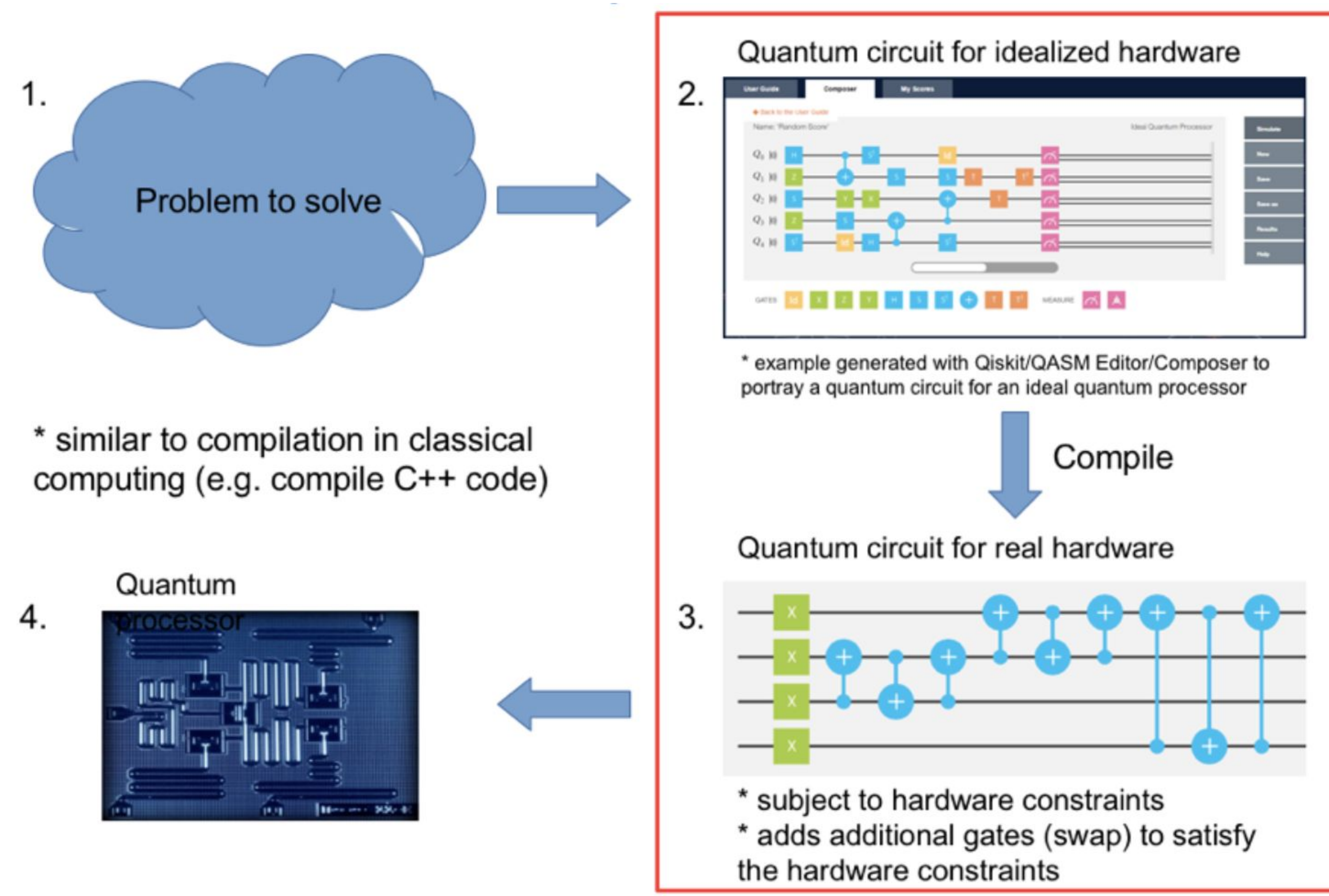
- Quantum Compilation Problem
- State of the Art
- Deterministic Pattern-oriented Quantum Compiler (DPQC)
  - Performance Evaluation

Quantum Compilation Problem

Fast, device-aware implementation of quantum algorithms

A good quantum compiler must translate an input program into the most efficient equivalent of itself, getting the most out of the available hardware

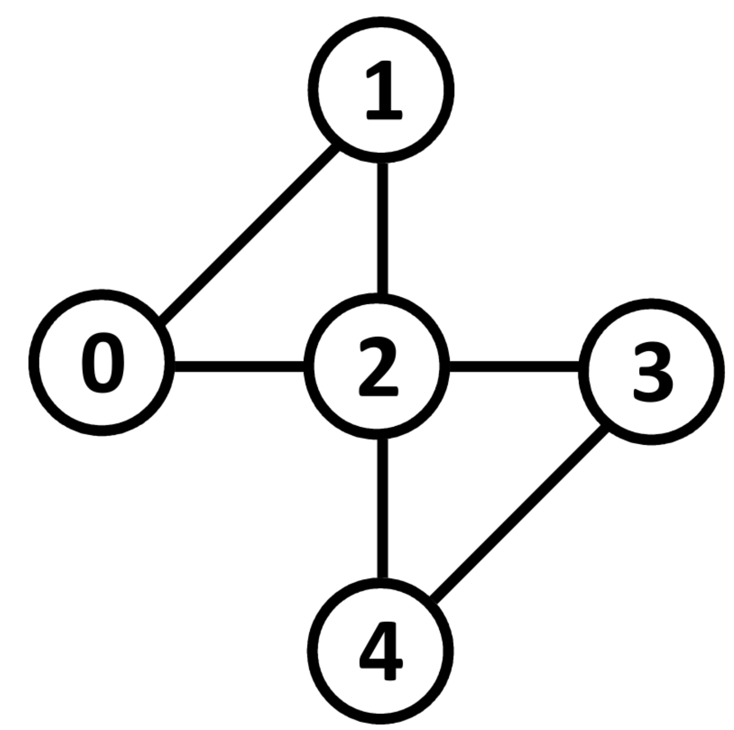
The quantum compilation problem in general is **NP-Hard**

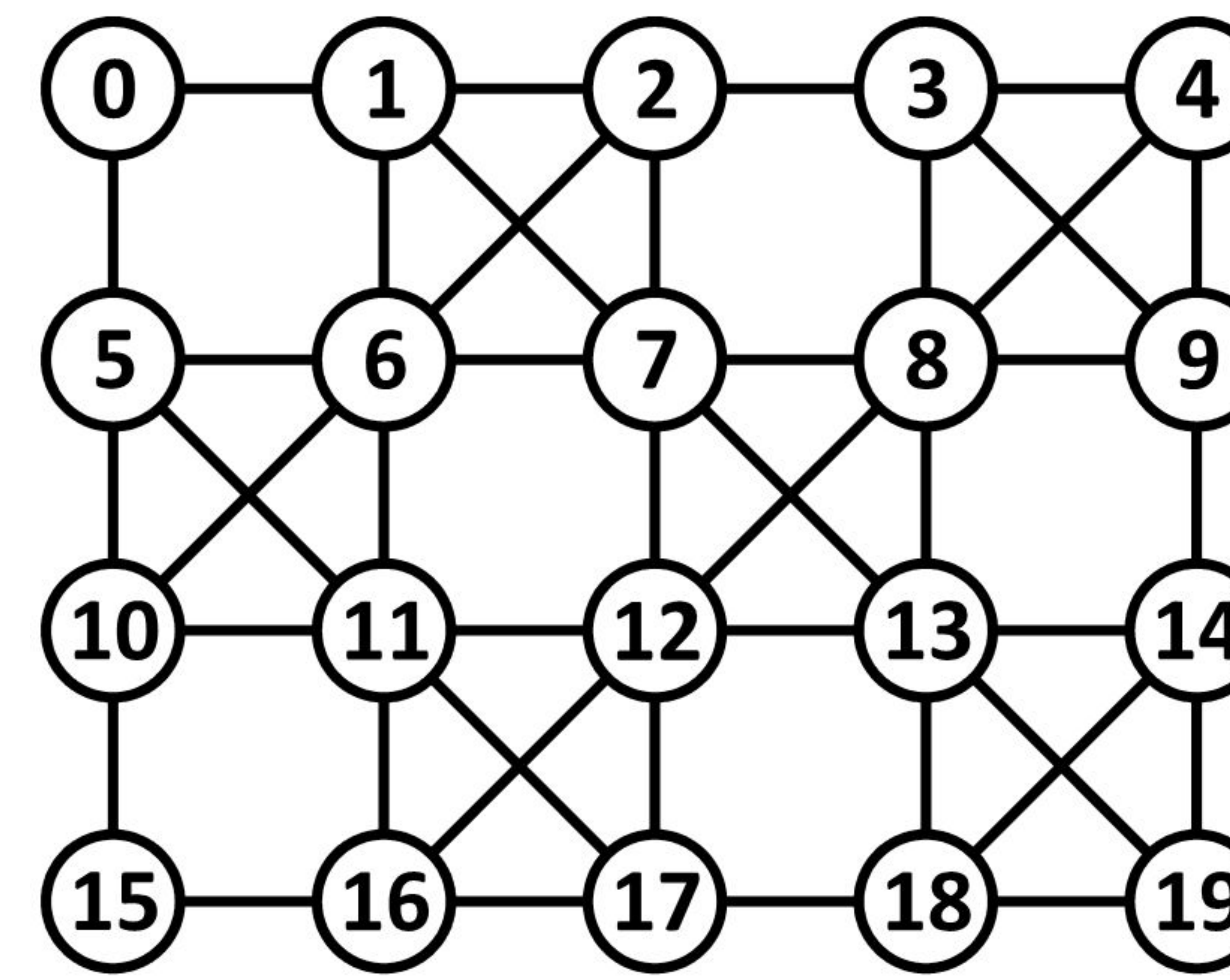


Quantum Compilation Problem

On noisy intermediate-scale quantum (NISQ) devices:

- **Gate synthesis** = decomposition of an arbitrary unitary operation into a sequence of gates from a discrete set
- Compliance with the **coupling map**
- Noise awareness



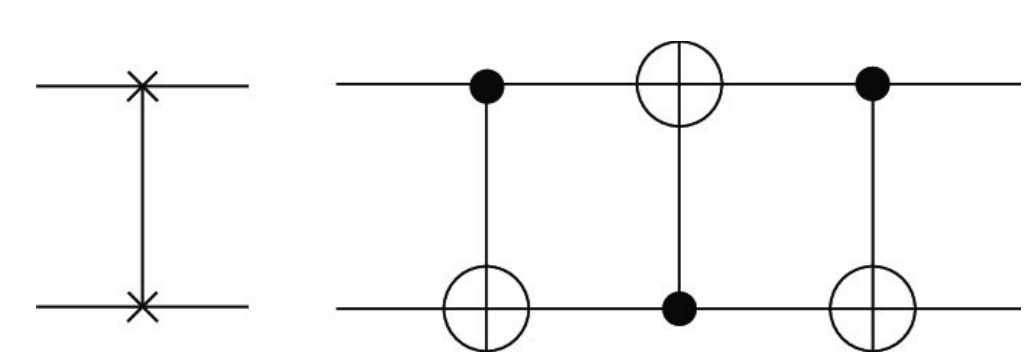


Quality indicators:

- Circuit **depth**
- **Gate count**
- **Fidelity** of quantum states

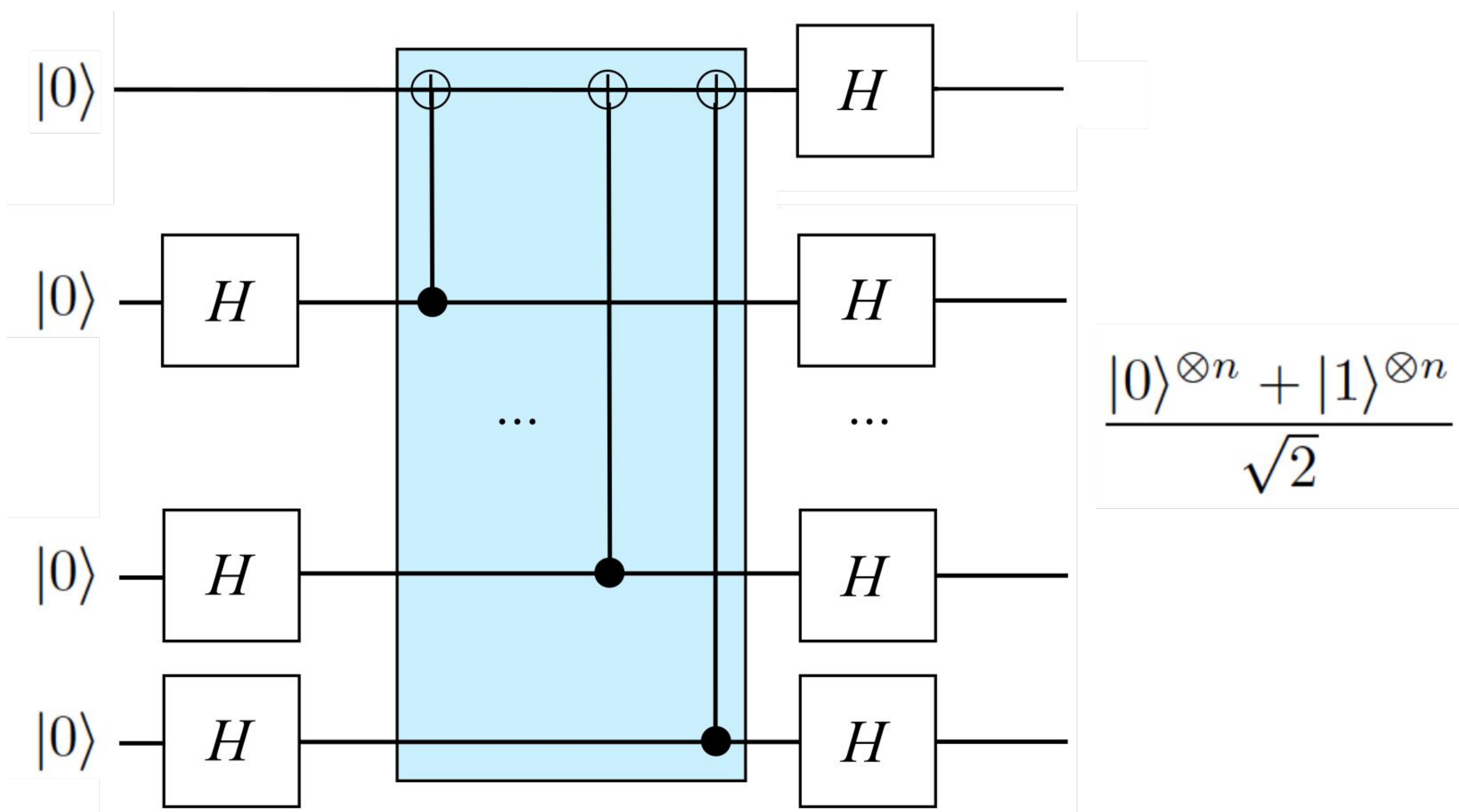
Actions:

- Decide **initial mapping**
- Exchange qubit states using **SWAP gates**



Example: GHZ State

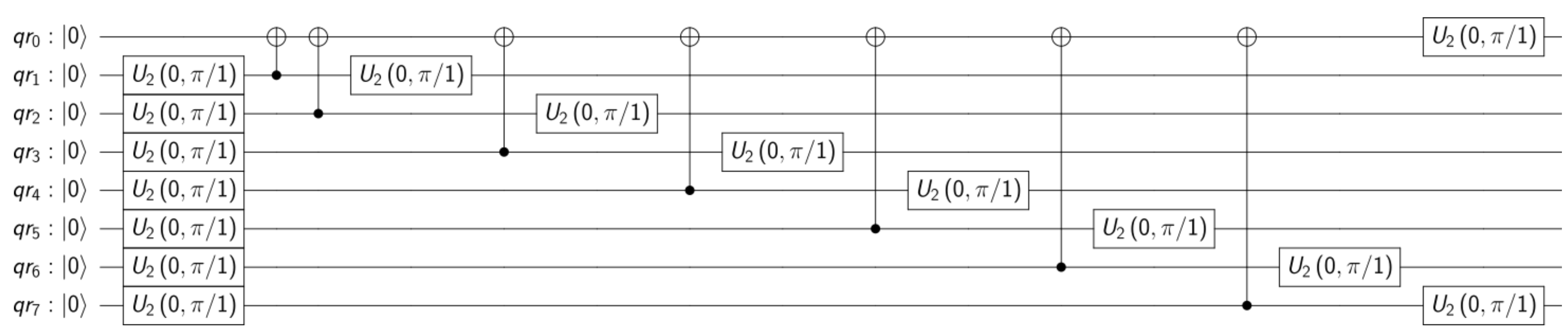
The ideal quantum circuit that generates the GHZ state requires only H and CNOT gates

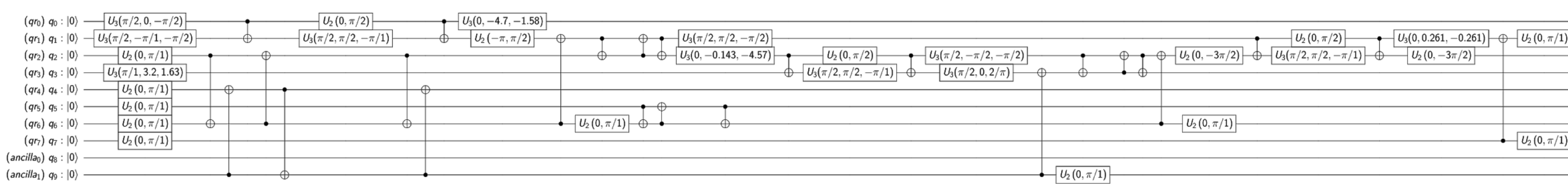


D. M. Greenberger, M. A. Horne, A. Zeilinger, Going Beyond Bell's Theorem, in 'Bell's Theorem, Quantum Theory, and Conceptions of the Universe', M. Kafatos (Ed.), Kluwer, Dordrecht, 69-72 (1989)

Example: GHZ State

Ideal vs. compiled 8-qubit GHZ circuit

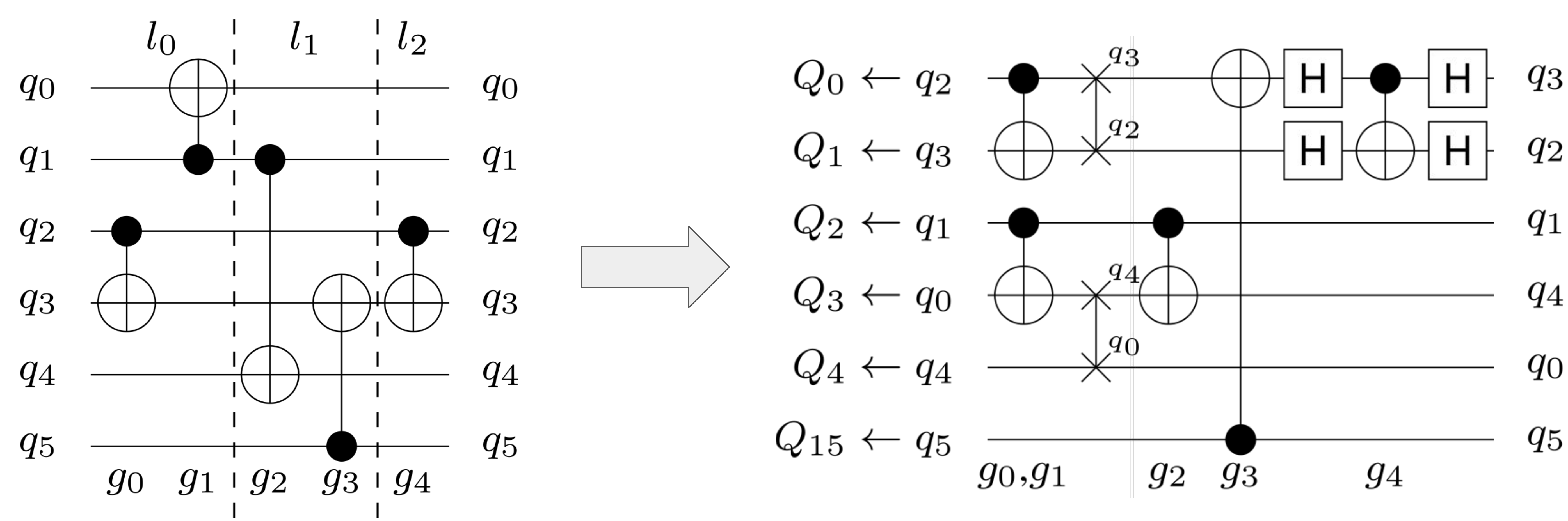




State of the Art

Zulehner *et al.*, An Efficient Methodology for Mapping Quantum Circuits to the IBM QX Architectures, IEEE TCAD vol.38 no.7, 2019

- Partition circuit into layers
- For each layer find a CNOT compliant mapping
  - *m!/(m-n)!* possible mappings with *m* physical qubits and *n* logical qubits
  - **A* search algorithm** to find less expensive swap sequence
  - minimize additional operations to switch between subsequent mappings



State of the Art

Jandura, 2018 (LookaheadSwap in Qiskit)

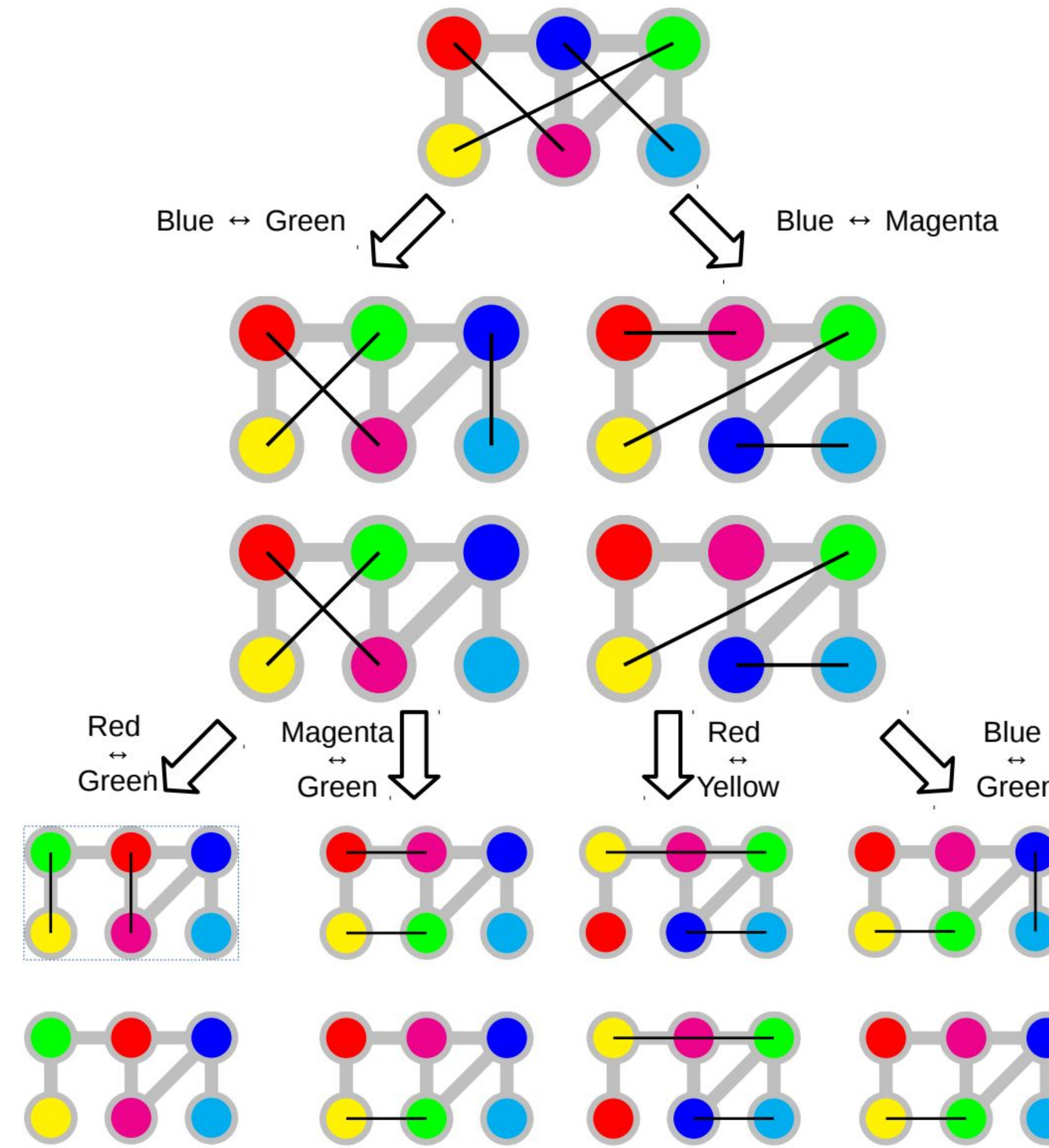
Mapping = logical qubits ↔ physical qubits

Given a mapping and a list of upcoming CNOTs, repeat 4 times:

1. Find 4 most promising swaps and calculate the new mapping for each of them
2. For each mapping count how many CNOTs can be executed and remove them from the list

From the 4^4=256 final mappings, choose the one that allowed for the most CNOTs to be executed and execute the fist swap on the path to this mapping

Then start the whole algorithm again, until the circuit is completed

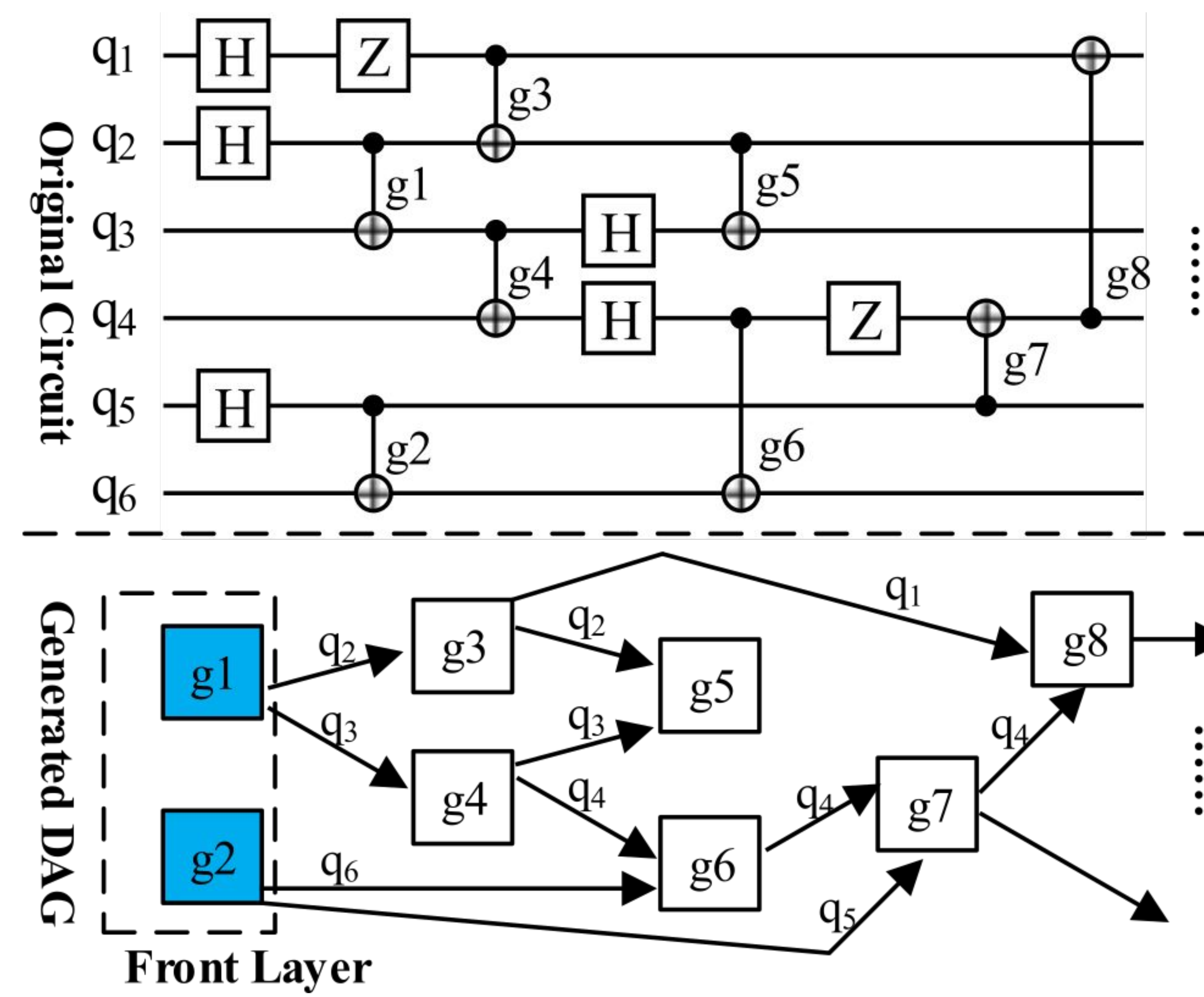


State of the Art

***Li et al., Tackling the Qubit Mapping Problem for NISQ-Era Quantum Devices*, ASPLOS '19, 2019**

SW**A**P-based **B**idi**RE**ctional heuristic search (**SABRE**)

- **Preprocessing**
  - Compute distance matrix D over the coupling map
  - Generate DAG from circuit to represent two-qubit gates dependencies
  - Initialize the front layer F as the set of two-qubit gates without unexecuted predecessors
  - Generate random initial mapping
- **Iterate over front layer F, stop when F is empty**
  - Remove executable gates from F and add their successor to F
  - For those gates in F that cannot be executed, select best SWAP sequence using an heuristic cost function based on distance matrix D



State of the Art

qiskit.transpiler

```
qc_comp = transpile(qc, backend=backend, coupling_map=coupling_map, pass_manager=pm)
```

The PassManager schedules the **passes**:

- Layout selection
- Unrolling
- **Swap Mapping**
  - **BasicSwap**
  - **StochasticSwap**
  - **LookaheadSwap**
- Gate optimizations
- etc.

Deterministic Pattern-oriented Quantum Compiler (DPQC)

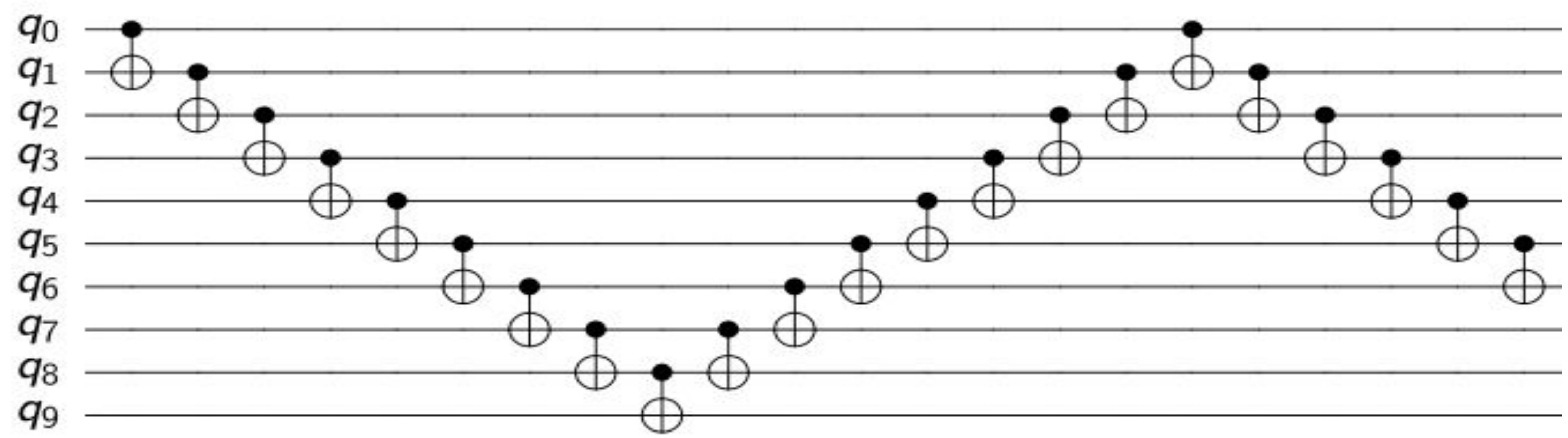
The DPQC software is the implementation of **efficient and deterministic strategies** for the compilation of quantum circuits characterized by peculiar sequences (patterns) of two-qubit operators.

By compilation we mean the transformation of abstract quantum circuits into circuits suited to the characteristics of the available hardware device, preserving their correctness and trying to maximize their efficiency.

Among the possible characteristics of a quantum computer, the DPQC software considers the coupling map and tries to **minimize circuit depth and gate count**.

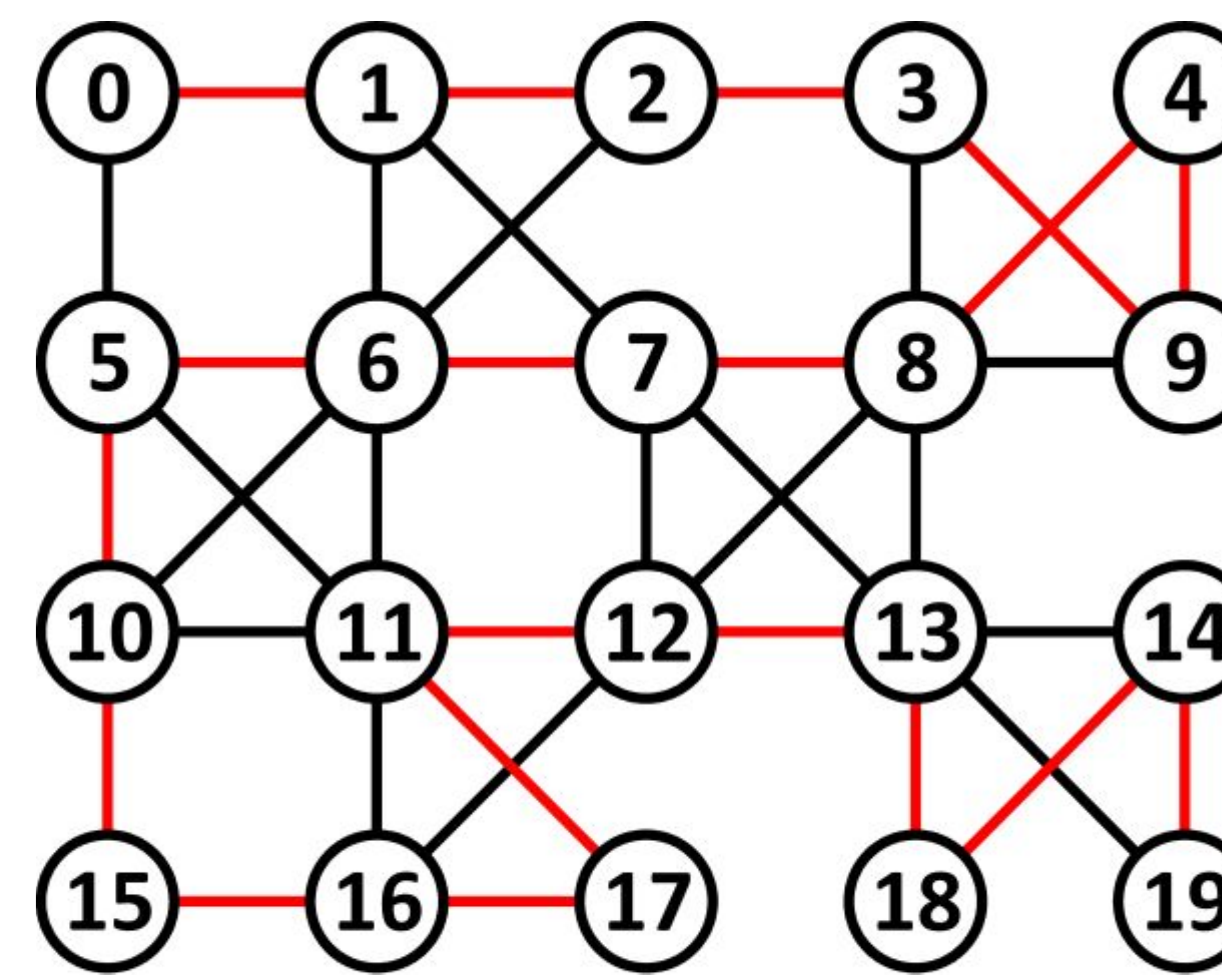
SIAE filing n.2019002328, 11/9/2019

CNOT Sequences Without Gaps

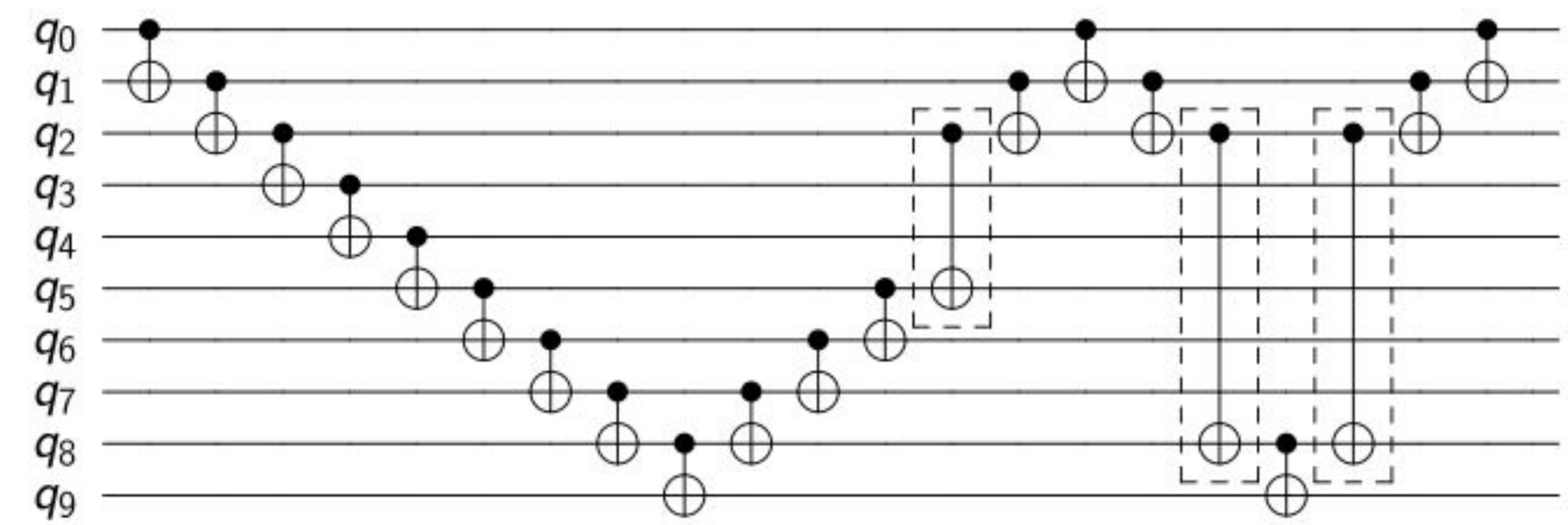


- Algorithm ***chain_layout()*** analyzes the coupling map and orders the qubits in a *chain*
- Every logical qubit $\mathbf{q_i}$ is associated with the **i-th** physical qubit in the chain
- Initial mapping satisfies all CNOTs

- In general, the problem of finding a path that visits every node only once is NP-complete
- *chain_layout()* exploits the qubit indexing and the graph structure to find a path with a deterministic and efficient approach
- Computational complexity ***O(m)***, where *m* is the number of physical qubits



CNOT Sequences With Gaps

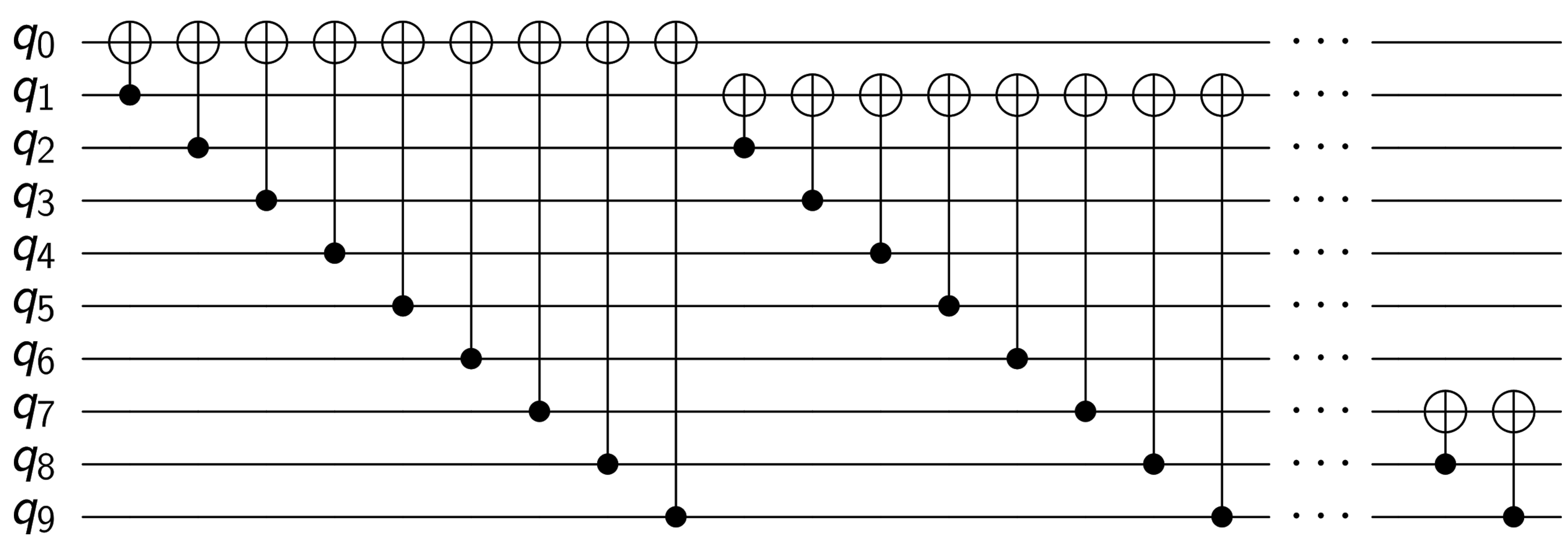


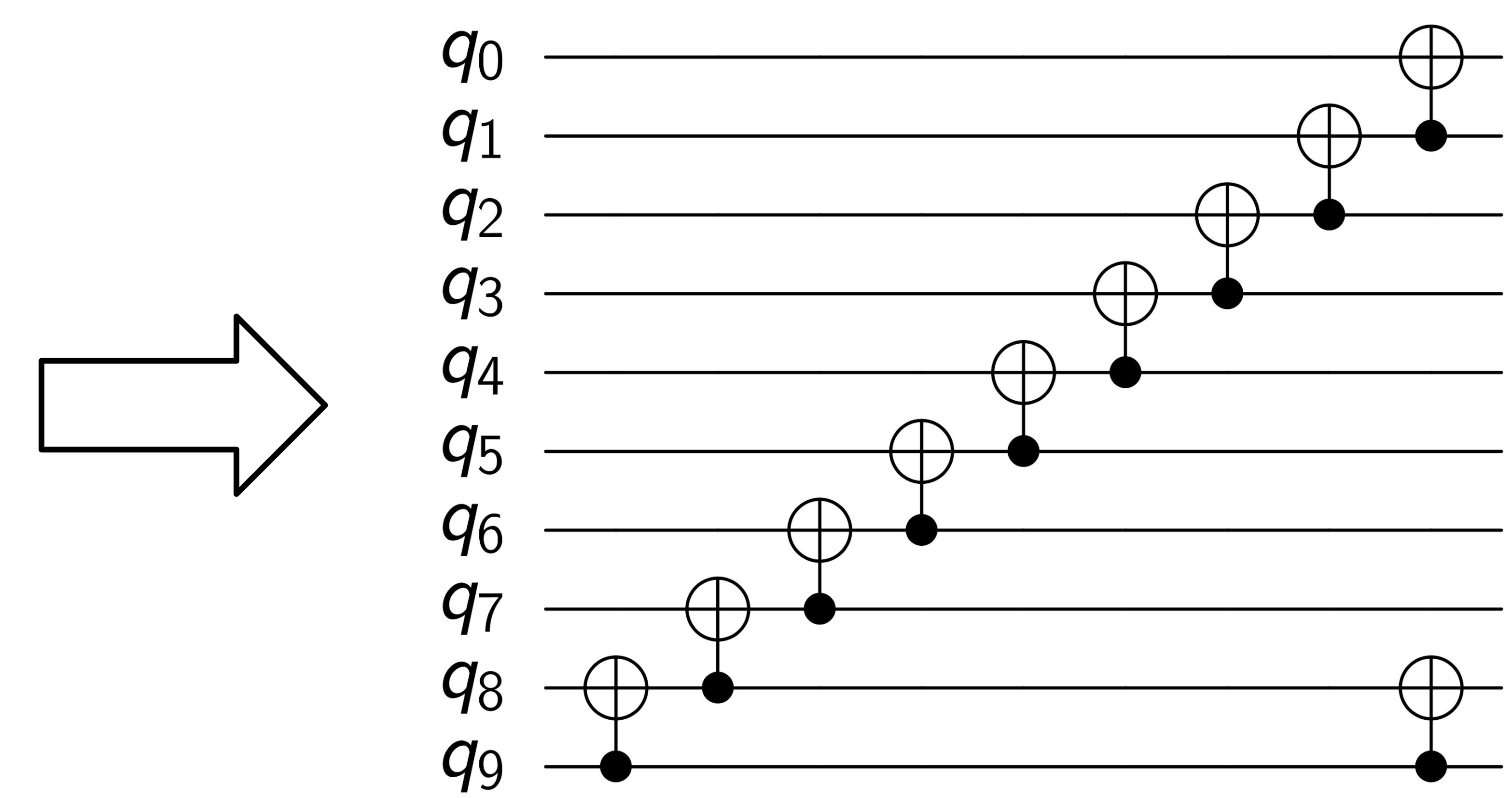
- Initial mapping is not enough
- Algorithm ***path()*** moves the qubits $\boldsymbol{q_1}$ and $\boldsymbol{q_2}$ involved in a remote CNOT close to each other:
  - iteratively computes the SWAP path between $\boldsymbol{q_1}$ and $\boldsymbol{q_2}$ neighbors
  - evaluates the distance between $\boldsymbol{q_x}$ and $\boldsymbol{q_y}$ as $|x - y|$ with x and y being their physical qubit indices in the coupling map
  - uses a cost function to choose which qubit to add to the path

$$f(x, y) = |x - y| * SWAP\_DEPTH$$

  - updates $\boldsymbol{q_1}$ and $\boldsymbol{q_2}$ mapping after every iteration
- Computational complexity ***O(m)***

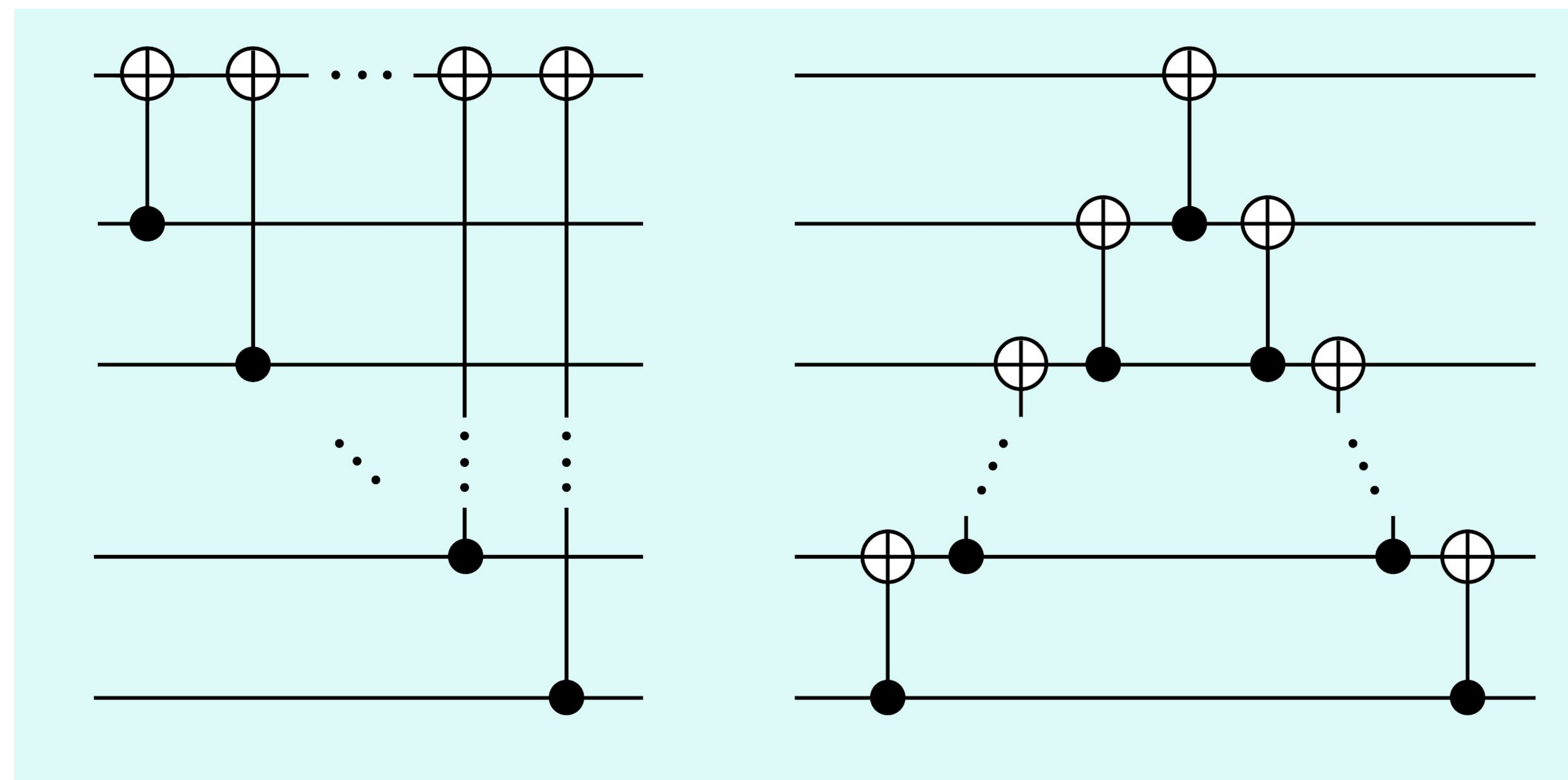
CNOT Cascades



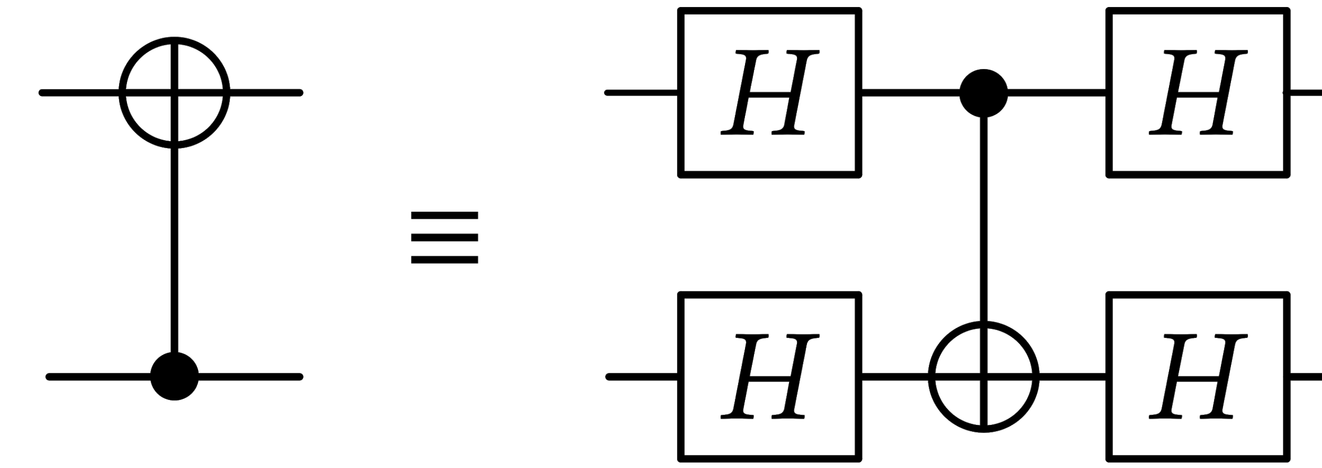


Repeated sequences of inverted CNOT cascades

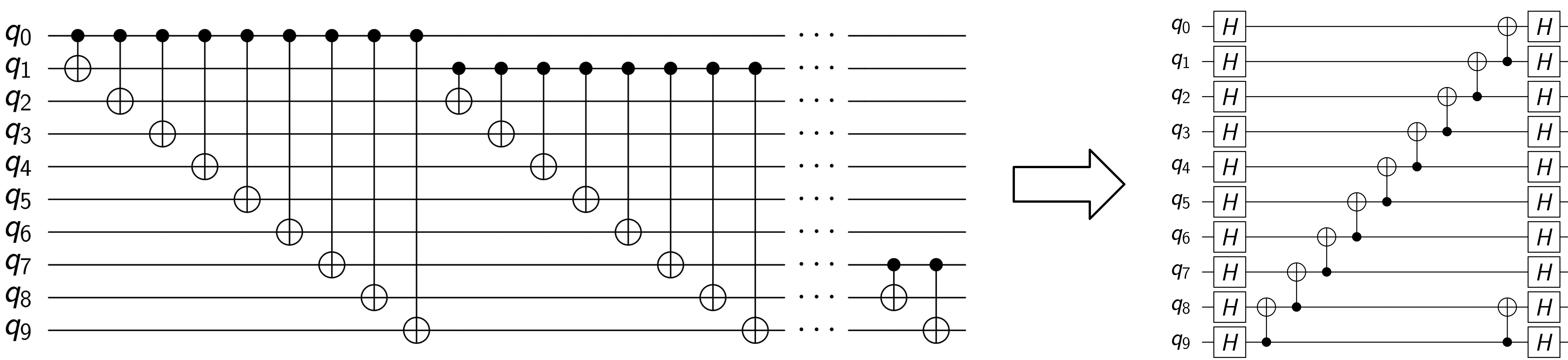
- CNOT cascades are equivalent to CNOT sequences
- Algorithm ***analyze_circuit()*** detects CNOT cascades and turns them into CNOT sequences
- Computational complexity ***O(gn)*** with *g* being the number of gates in the circuit



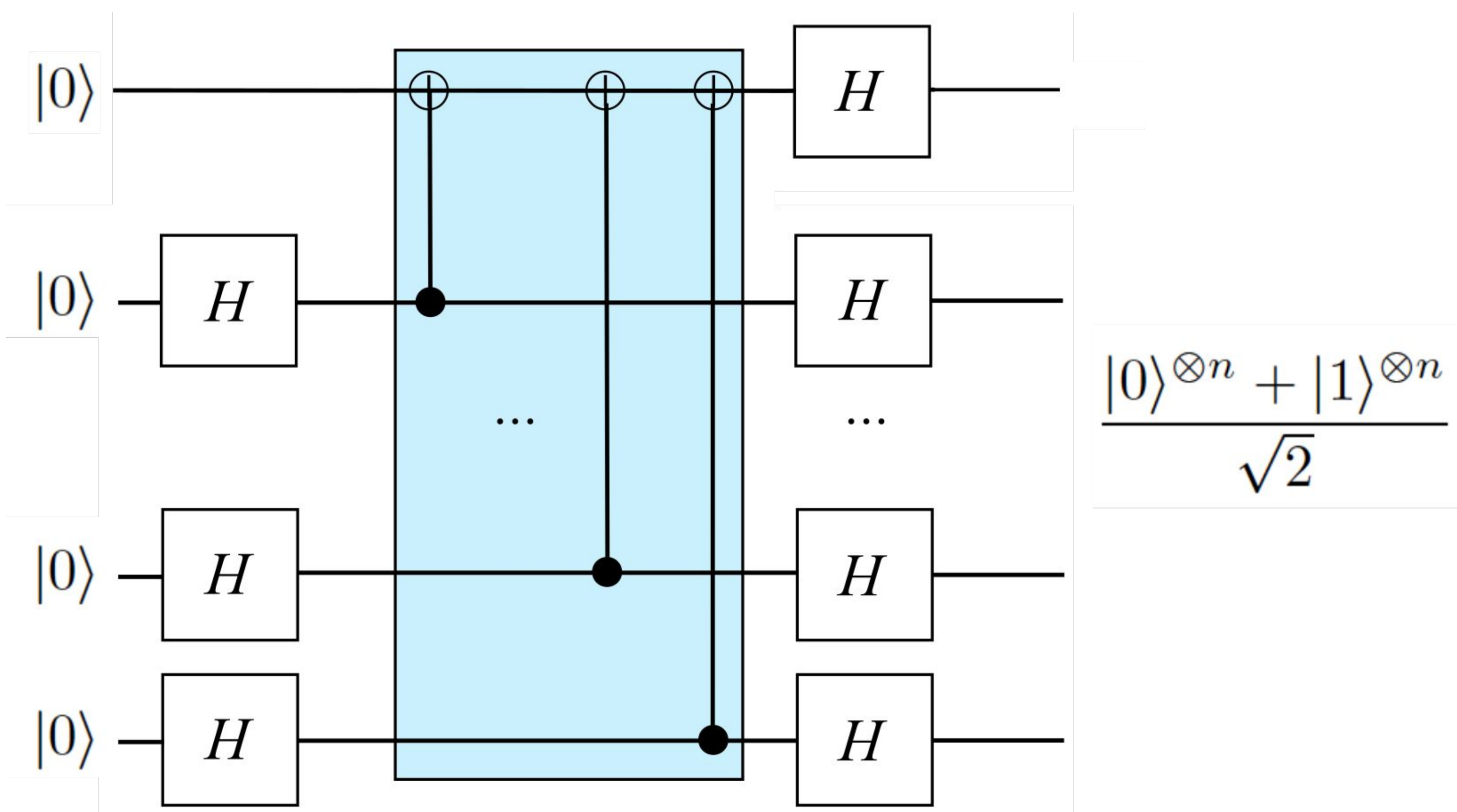
Inverted CNOT Cascades

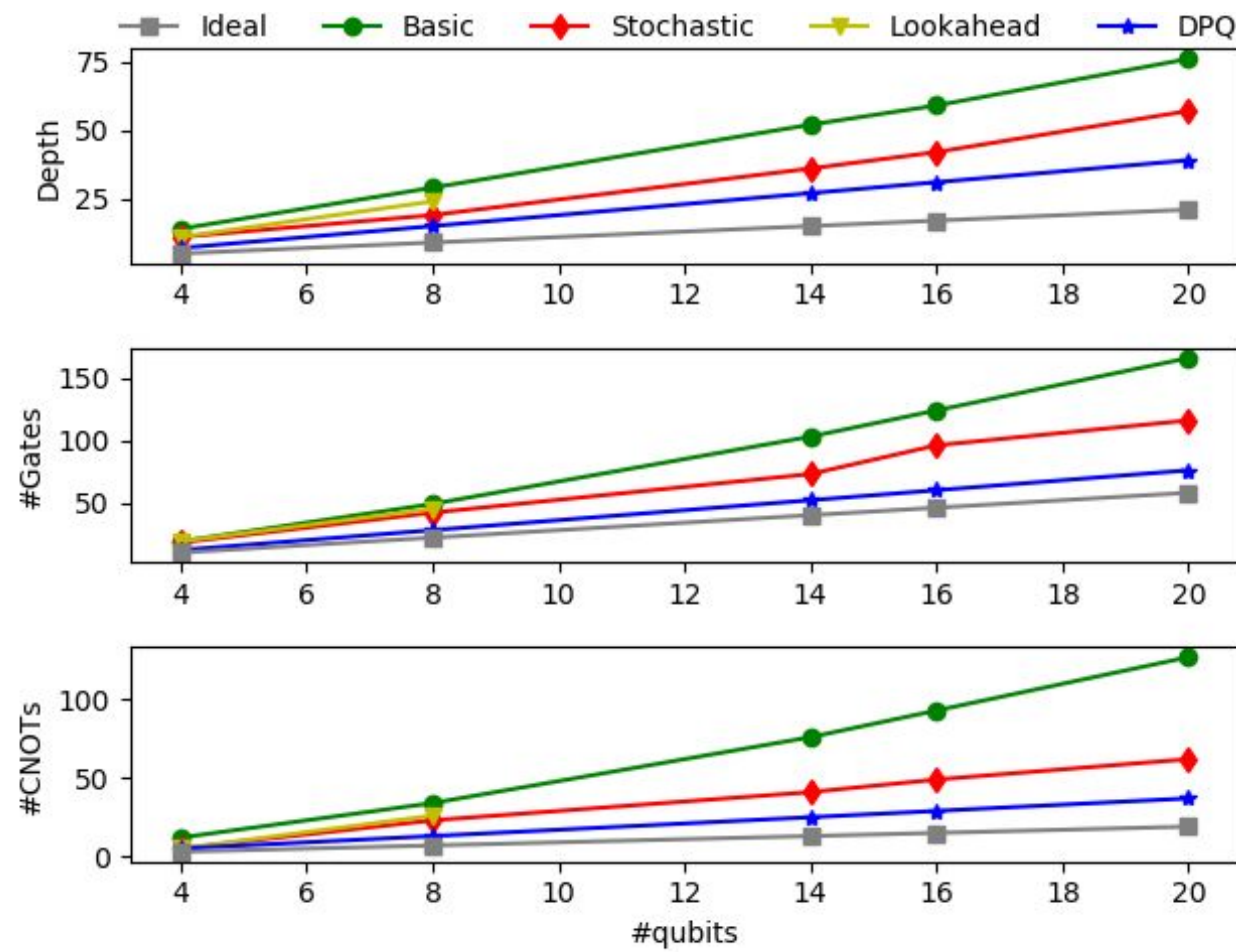


- CNOT gates can be inverted by applying H gates before and after involved qubits
- ***analyze_circuit()*** also detects inverted CNOT cascades and turns them into CNOT cascades
- Algorithm ***gate_cancellation()*** deletes double CNOT and double H gates
- Computational complexity ***O(dn²)***



GHZ Results





Backend: QX20 Tokyo

Quantum Chemistry Circuits

Used to compute the ground state wave function of simple molecular systems such as:

- Hydrogen **H_2 with 4 qubits**
- Lithium Hydride **LiH with 12 qubits**
- Water **H_2O with 14 qubits**

IBM Research | Zurich

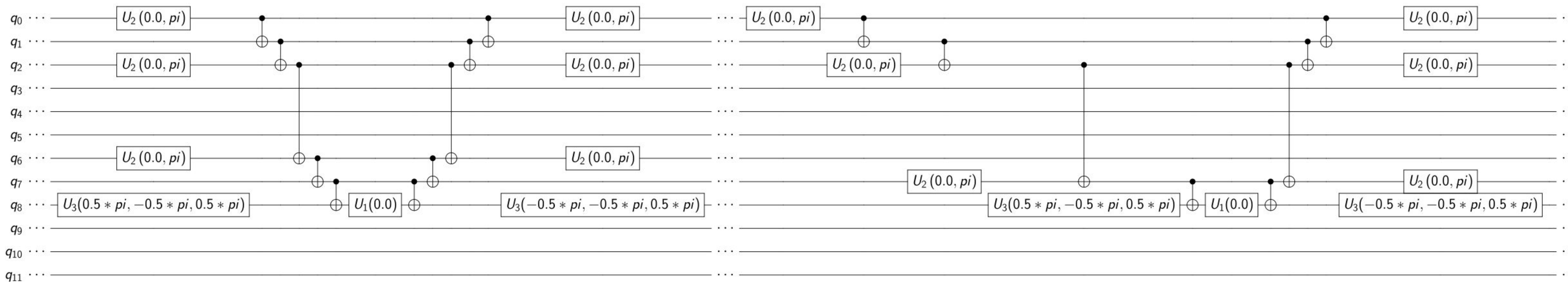
Ivano Tavernelli
Theoretical Quantum Computing @ IBM Research - Zurich

Efficient Quantum Compiling for Quantum Chemistry Simulation on IBM Q

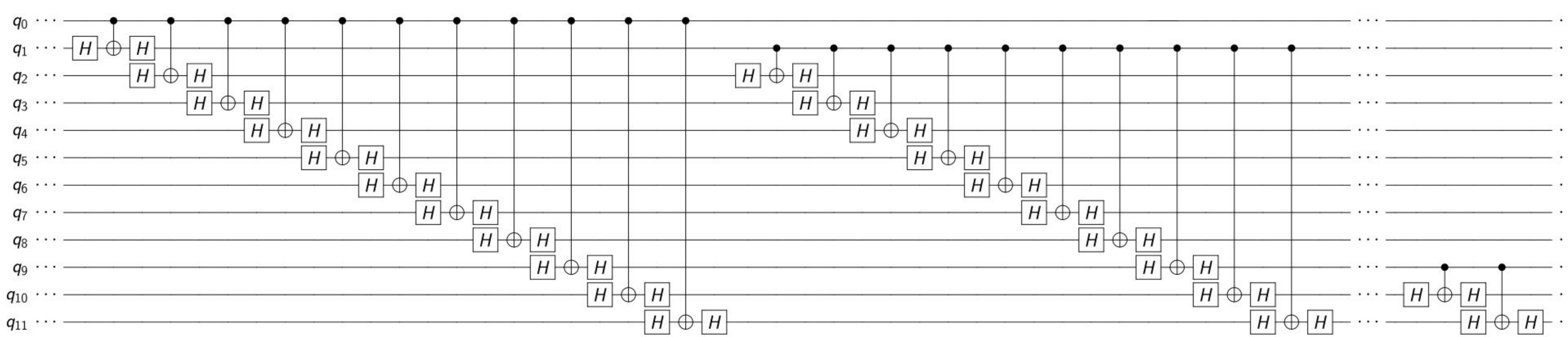
D. Ferrari,* I. Tavernelli,† M. Amoretti@*

@michele.amoretti@unipr.it, *Department of Engineering and Architecture - University of Parma, Italy, †IBM Zurich Research Laboratory

UCCSD


q0
q1
q2
q3
q4
q5
q6
q7
q8
q9
q10
q11
U2 (0.0, pi)
U3(0.5 * pi, −0.5 * pi, 0.5 * pi)
U1(0.0)
U3(−0.5 * pi, −0.5 * pi, 0.5 * pi)

RyRz


q0
q1
q2
q3
q4
q5
q6
q7
q8
q9
q10
q11
H

UCCSD Results

	Circuit Depth			
Circuit Name	Input Circuit	DPQC	Basic	Stochastic
H2_UCCSD	82	64	64	64
LiH_UCCSD	8845	6717	12201	8026
H2O_UCCSD	15388	13413	12797	14789
Random20_UCCSD	125638	132865	204927	173672

	Compilation Time (s)		
Circuit Name	DPQC	Basic	Stochastic
H2_UCCSD	1	1	1
LiH_UCCSD	221	247	314
H2O_UCCSD	575	393	756
Random20_UCCSD	18448	11230	20127

Backend: QX20 Tokyo

RyRz Results

	Circuit Depth			
Circuit Name	Input Circuit	DPQC	Basic	Stochastic
H2_RyRz	73	40	45	45
LiH_RyRz	233	160	898	615
H2O_RyRz	273	190	1258	841
Random20_RyRz	393	279	3552	1629

	Compilation Time (s)		
Circuit Name	DPQC	Basic	Stochastic
H2_RyRz	1	1	1
LiH_RyRz	4	20	24
H2O_RyRz	5	36	28
Random20_RyRz	8	93	65

Backend: QX20 Tokyo

Other Benchmarks

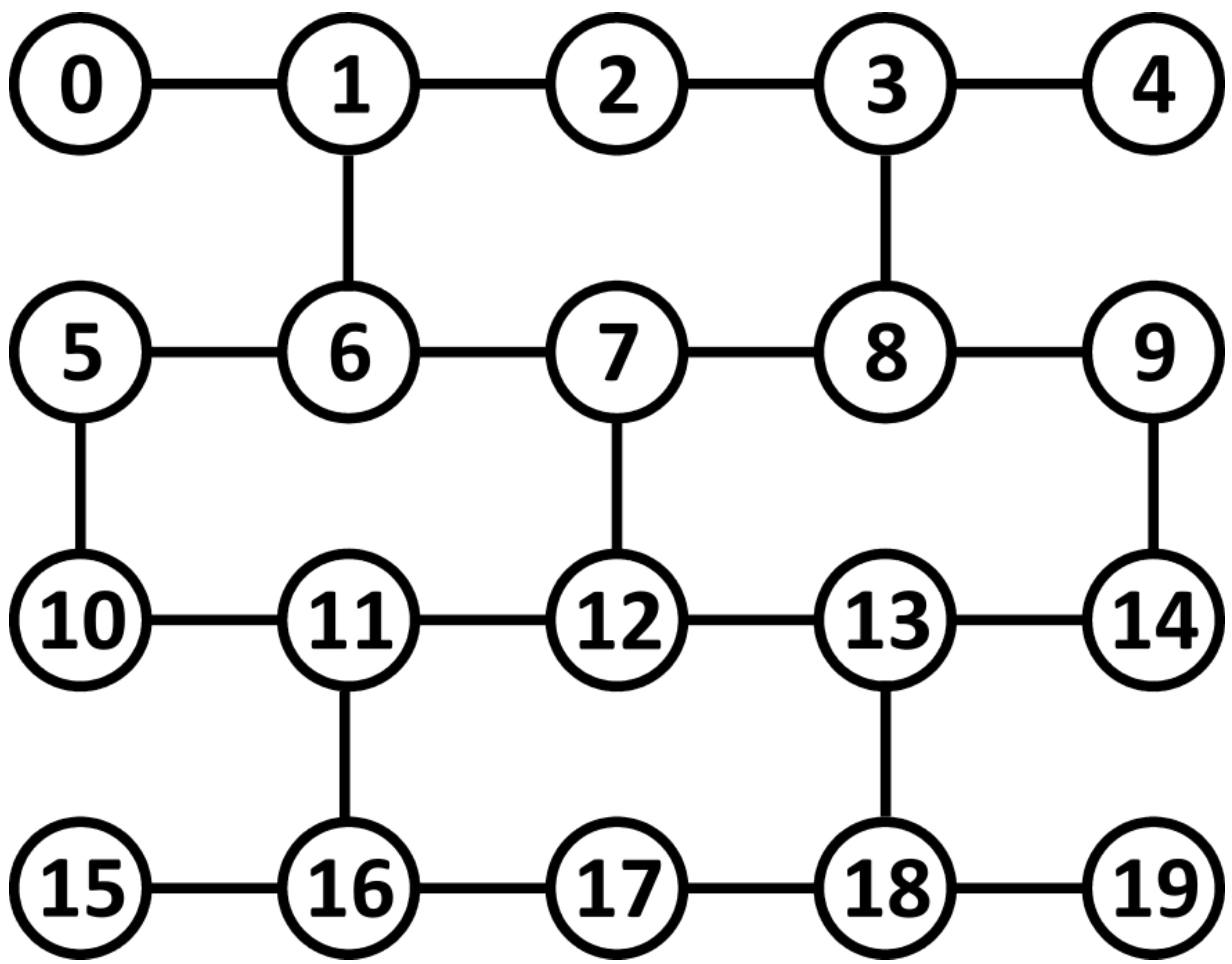
	Circuit Depth			
Circuit Name	Input Circuit	DPQC	Basic	Stochastic
9symml_195	19235	35406	38489	40002
clip_206	17879	34715	33877	38726
co14_215	8570	18294	18551	19458
dist_223	19694	36144	36488	42289
life_238	12511	23319	24974	25990
max46_240	14257	27466	27550	29650
sao2_257	19563	39268	40202	42988
sqn_258	5458	9999	10599	11271
sym10_262	35575	67362	64885	75168

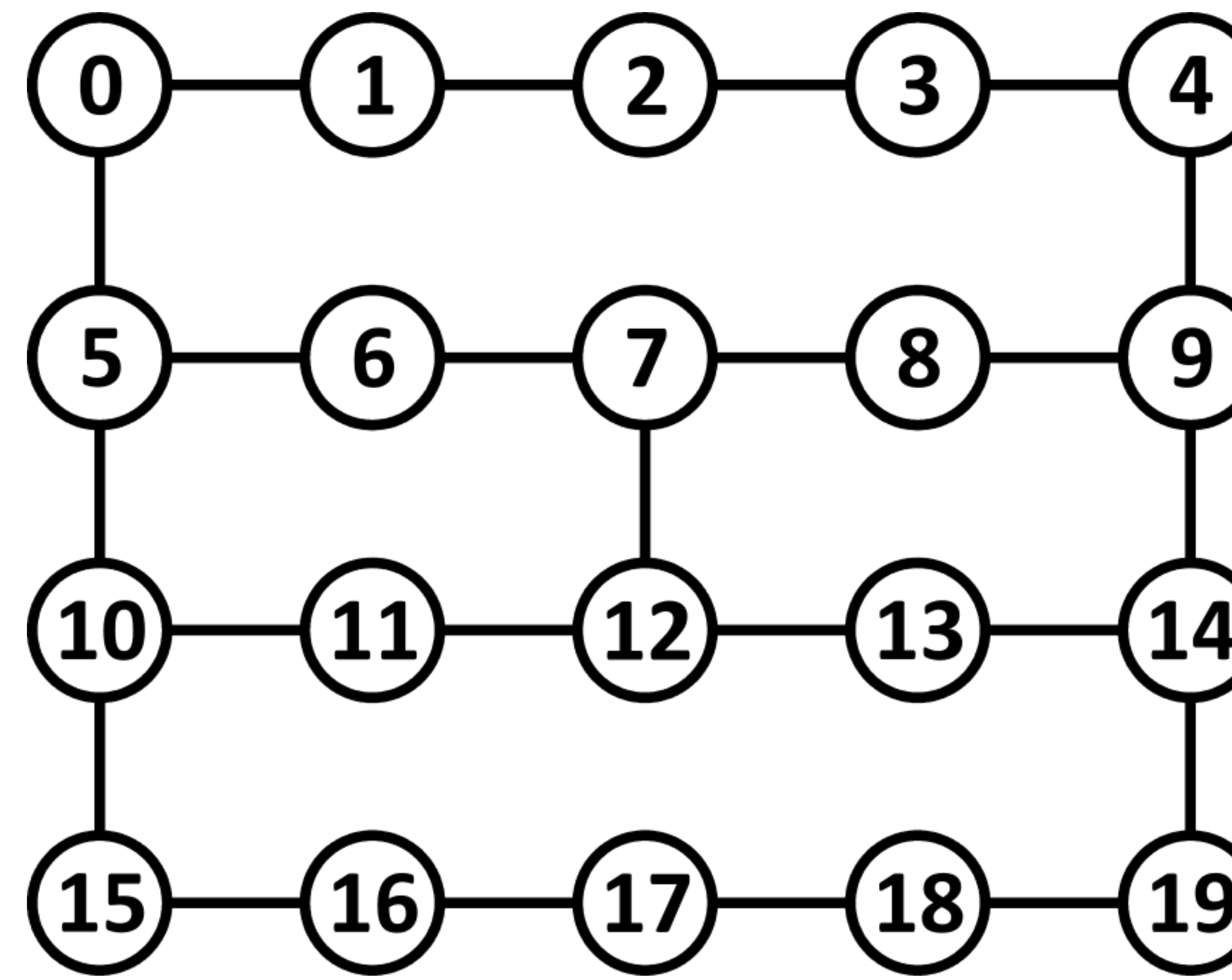
	Compilation Time (s)		
Circuit Name	DPQC	Basic	Stochastic
9symml_195	612	626	802
clip_206	759	570	817
co14_215	377	316	450
dist_223	783	626	951
life_238	358	397	538
max46_240	431	429	660
sao2_257	871	691	1036
sqn_258	148	170	239
sym10_262	1543	1119	1987

Backend: QX20 Tokyo

Future Work

- Testing the proposed approach with the **new coupling maps**
- Using extra qubits
- Studying **other circuit patterns**





Future Work

While effective, circuit depth and gate count are only pseudo-objectives. In reality, what matters is the **fidelity** of a computation when run on actual quantum hardware.

NISQ compilers excel when more **information** is made available to them **from the device**.

IBM Q **device properties** are shared openly and can be benchmarked, resulting in a bunch of compiler innovations.

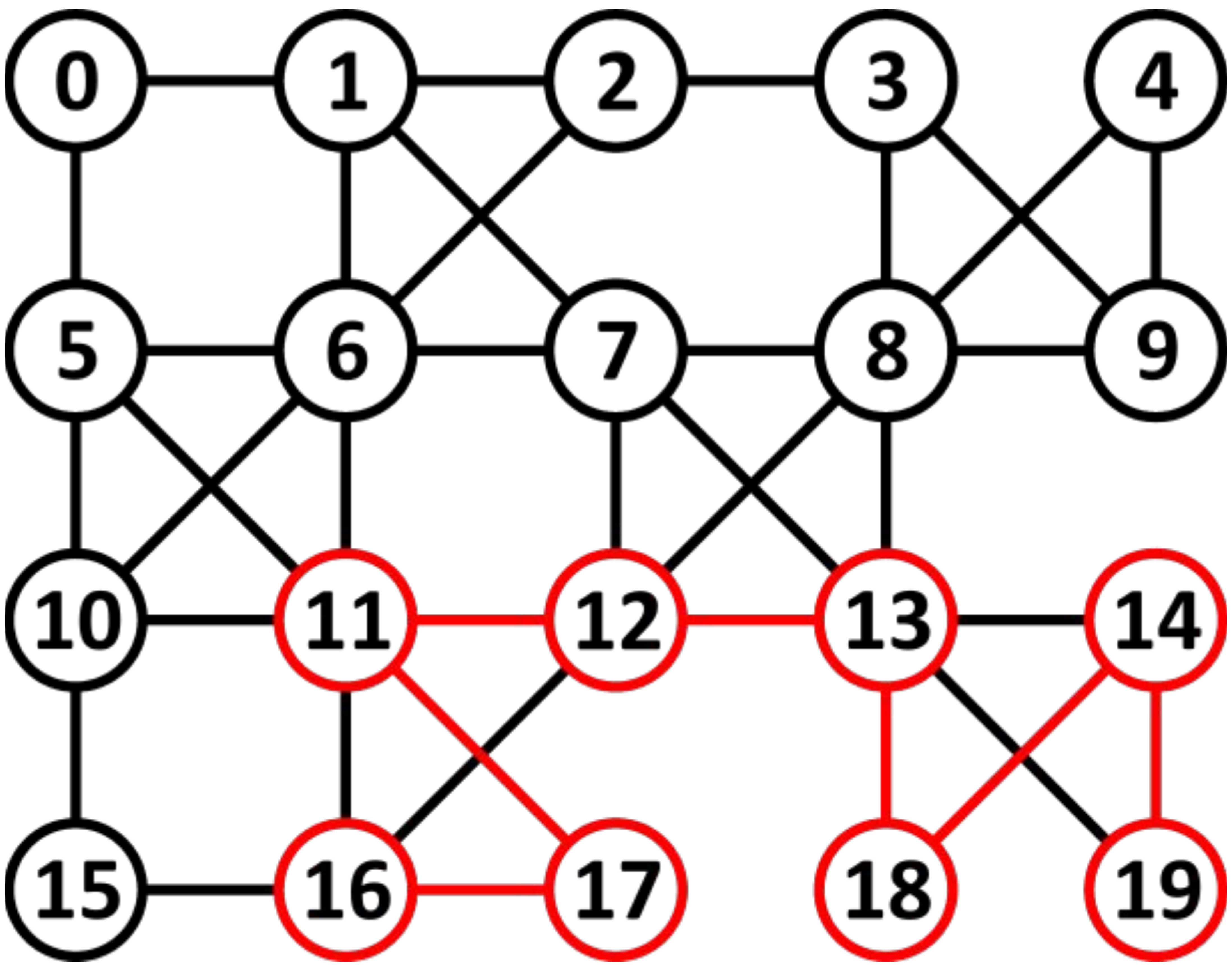
P. Murali *et al.*, Noise-Adaptive Compiler Mappings for Noisy Intermediate-Scale Quantum Computers, arXiv:1901.11054, 2019

S. Nishio *et al.*, Extracting Success from IBM's 20-Qubit Machines Using Error-Aware Compilation, arXiv:1903.10963, 2019

Thank You!

Questions?

Sequence Offset



8 qubits chain with *offset*=12

- 0 ≤ ***offset*** < (*m*-*n*)
- Used to map the qubits in the circuit to the qubits in the chain in the interval {0,..,*n* + *offset*} instead of {0,..,*n*}

Impact of Sequence Offset