Designing Optimal Quantum Circuits with Mixed-Integer Nonlinear Programming

Harsha Nagarajan

Theoretical Division, Los Alamos National Laboratory

CPAIOR: Quantum Computing for CP, AI, and OR, and vice-versa

Uppsala, Sweden

May 28, 2024

LA-UR-22-32426

Los Alamos NATIONAL LABORATORY EST. 1943

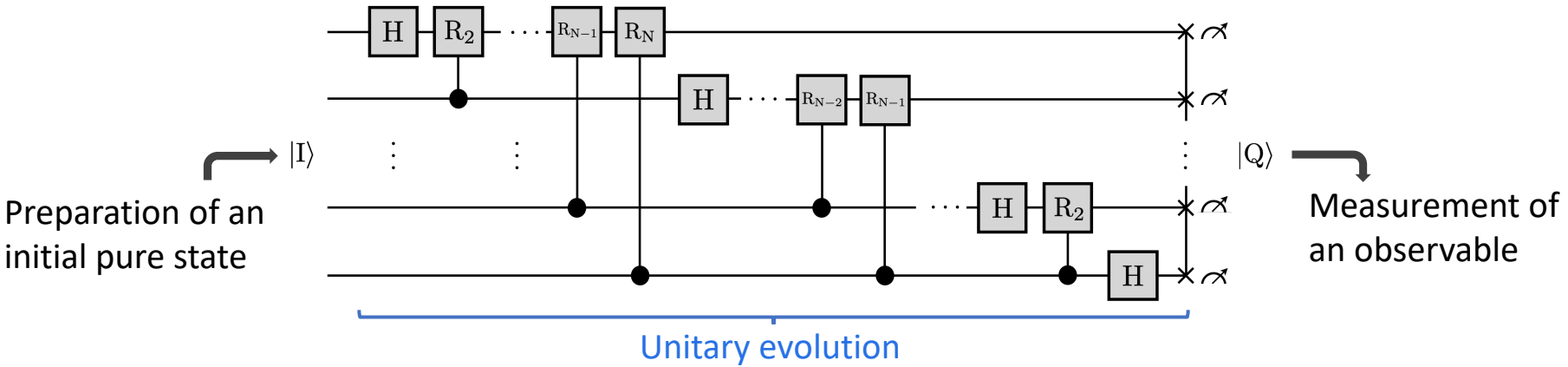
Outline

- A short introduction
- Underlying mathematical programs:
  - Mixed-integer linear programming (MIP) models
- QuantumCircuitOpt's framework
- Compact circuit realizations and MIP performance
- Concluding remarks

Background on Quantum Circuits

Quantum algorithm as a circuit

Quantum algorithm is a sequence of unitary one- and two-qubit elementary gates, with arbitrary entangled states.

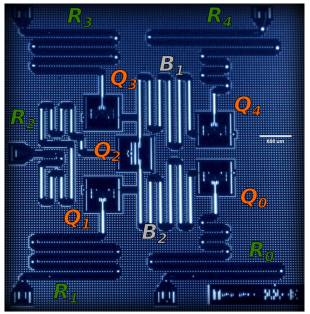


Research goal: To implement any quantum algorithm, also represented using a target unitary, into a shortest possible sequence of one- and two-qubit gates.

Hardware-based native gate set

IBMQX4 5-qubit processor's native gate set[a]

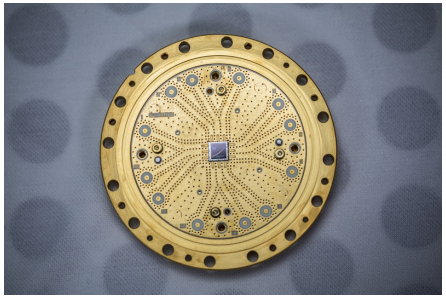
$$\{\mathrm{U_3}(\theta=0,\phi=0,\lambda),\mathrm{R_x}(\pi/2),\mathrm{CNOT_{i,j}}\}$$



One-qubit gates **Two-qubit gates**

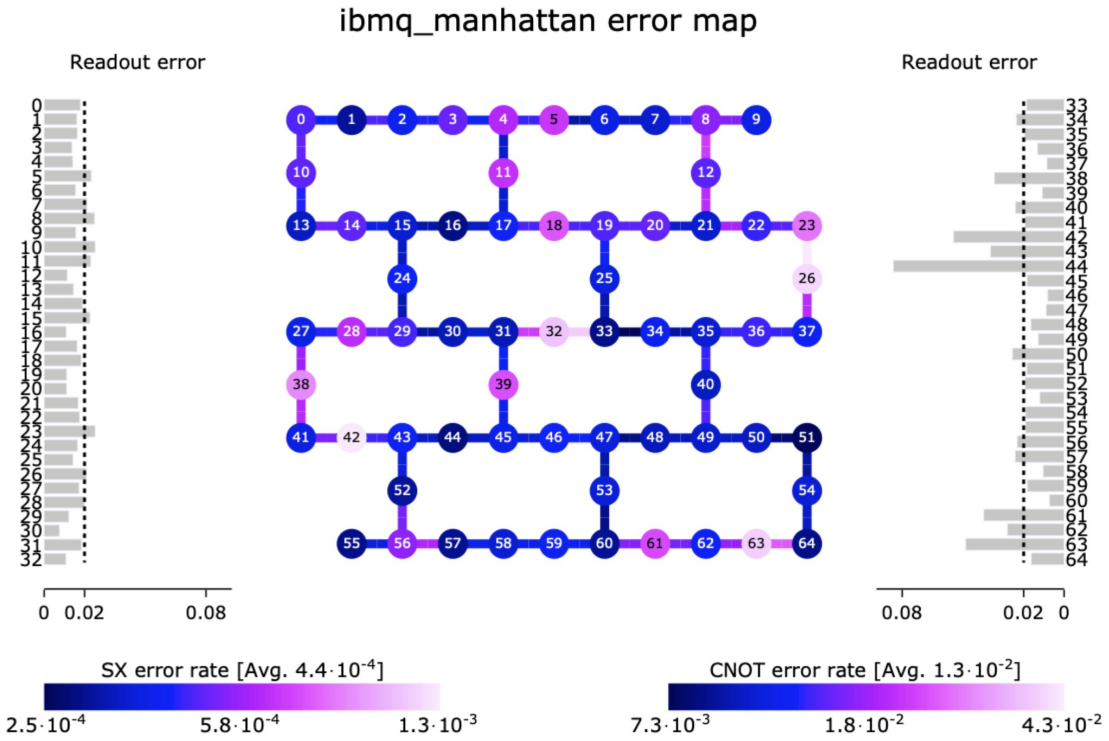
Rigettis' 8Q-Agave 8-qubit processor's native gate set[a]

$$\{\mathrm{U_3}(\theta=0,\phi=0,\lambda),\mathrm{R_x}(\pm\pi/2),\mathrm{CZ_{i,j}}\}$$



a. Khatri, S., LaRose, R., Poremba, A., Cincio, L., Sornborger, A.T. and Coles, P.J., Quantum-assisted quantum compiling. Quantum, 2019.

Hardware connectivity/topology constraints



What do we care about?

Reducing noise in the quantum processor!

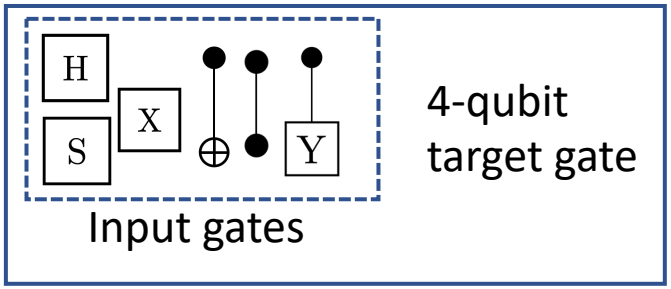
The key properties that define the quality of the transpiled circuit are:

- Error rate of one and two-qubit gates - entangling two-qubit gates (like CNOTs) are more error prone
- Total depth of the circuit
- Inter-qubit connectivity
- Qubit coherence times

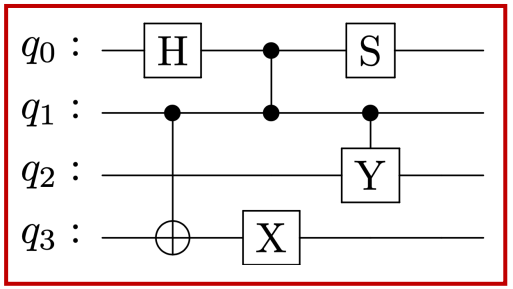
Circuit compilation: Software libraries

Qiskit Cirq Amazon Braket Azure Quantum

- Various libraries to design optimized circuits with desired levels of accuracy.
- Provide an interface to quantum computers or simulators.

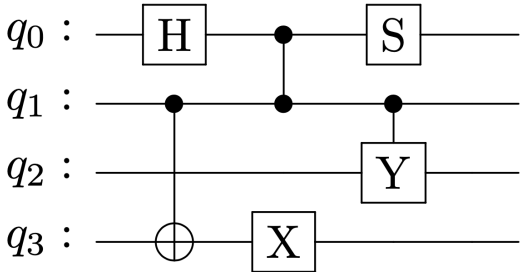


Heuristics in Cirq

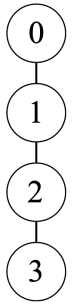


No mechanism to measure how close they are to the best possible circuit.

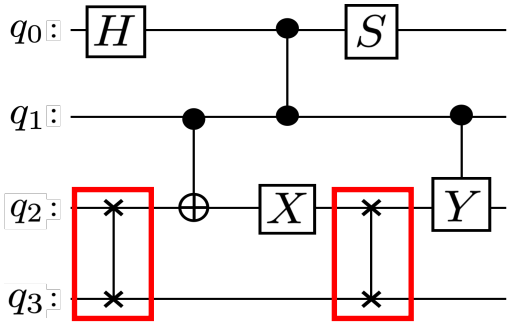
Circuit optimization: Integer programming



Input circuit



Hardware connectivity



Output circuit

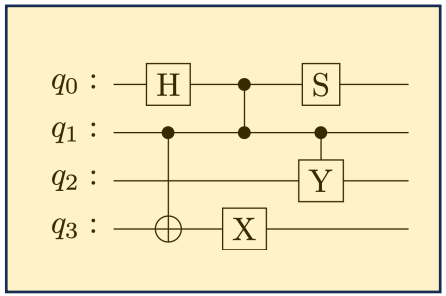
6 CNOTs?

Optimal qubit routing for hardware connectivity

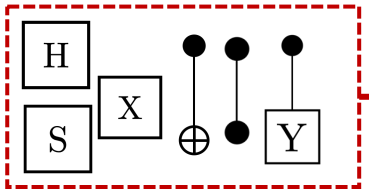
- Nannicini, G., Bishop, L.S., Günlük, O. and Jurcevic, P., *Optimal qubit assignment and routing via integer programming*. ACM Transactions on Quantum Computing, 4(1), pp.1-31, 2022.

- Wagner, Friedrich, Andreas Bärmann, Frauke Liers, and Markus Weissenbäck. "*Improving quantum computation by optimized qubit routing*." Journal of Optimization Theory and Applications 197, no. 3, 1161-1194, 2023.

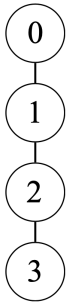
An unified approach!



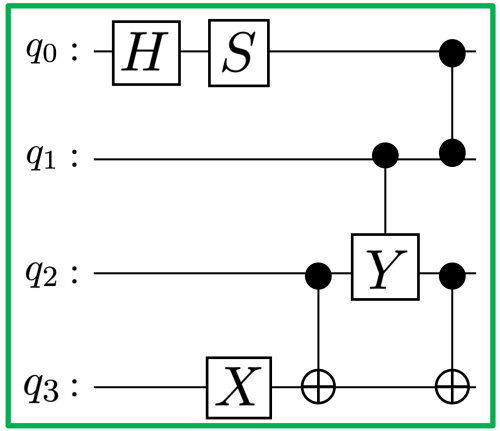
Target unitary



Input gates



Hardware connectivity

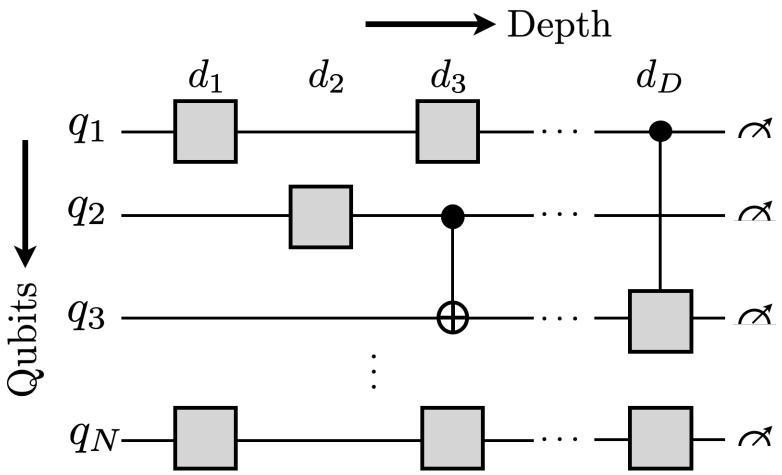


Optimal circuit

✓ Use Mathematical Programming tools to provide theoretical guarantees on the solution quality of quantum circuit decompositions

Underlying Mathematical Programs

Let's formulate it!



N – Number of qubits

D – Maximum allowable depth

$$\mathrm{U}_3(\hat{\theta},\hat{\phi},\hat{\lambda}) = \begin{bmatrix} \cos(\hat{\theta}) & -e^{\mathrm{i}\hat{\lambda}}\sin(\hat{\theta}) \\ e^{\mathrm{i}\hat{\phi}}\sin(\hat{\theta}) & e^{\mathrm{i}(\hat{\phi}+\hat{\lambda})}\cos(\hat{\theta}) \end{bmatrix}$$

$\hat{\theta}, \hat{\phi} \in \{0\}$,

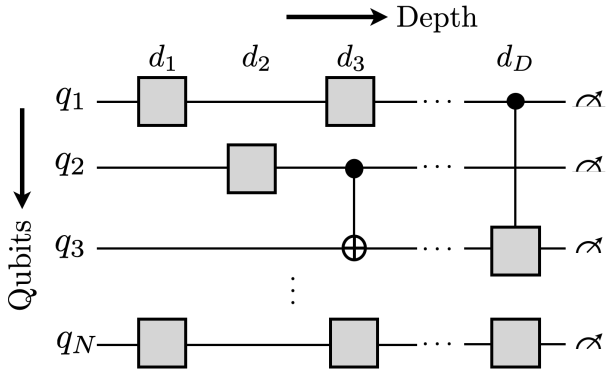
$\hat{\lambda} \in \mathcal{L}$ (discretized angles)

$$\mathrm{R_x}(\pi/2) = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & -\mathrm{i} \\ -\mathrm{i} & 1 \end{bmatrix}, \quad \mathrm{I} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$\mathrm{CNOT}_{c,t} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

Native gates set with constant parameters

Let's formulate it!



N – Number of qubits

D – Maximum allowable depth

Binary variables for gate locations

$$z^{u}_{n,d,k} \in \{0,1\},\ z^{r}_{n,d} \in \{0,1\},\ z^{cnot}_{n_c,n_t,d} \in \{0,1\},\ z^{id}_{d} \in \{0,1\}$$

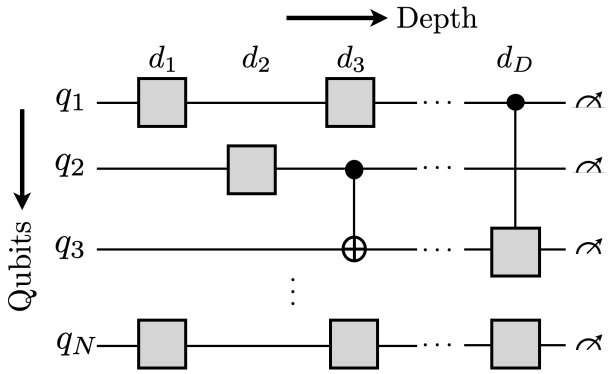
$z^{u}_{n,d,k} = 1$ *iff* $\mathrm{U}_3(\hat{\lambda}_k)$ is placed on qubit n, depth d

$z^{r}_{n,d} = 1$ *iff* $\mathrm{R}_{\mathrm{x}}(\pi/2)$ is placed on qubit n, depth d

$z^{cnot}_{n_c,n_t,d} = 1$ *iff* CNOT_{n_c,n_t} is placed on qubits (n_c, n_t), depth d,

$z^{id}_{d} = 1$ *iff* I is placed at depth d

Let's formulate it!



N – Number of qubits

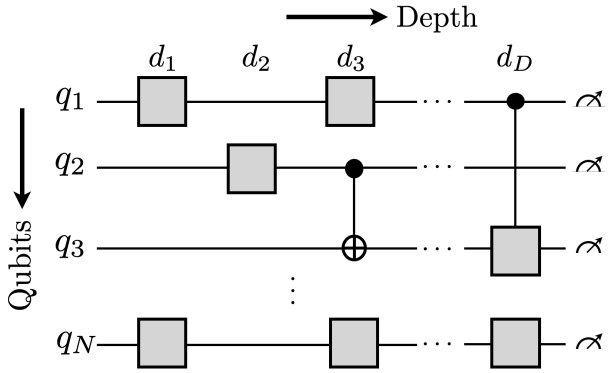
D – Maximum allowable depth

Gate-choice constraints per depth (in 2 qubits)

$$\mathrm{G}_d = \sum_{\hat{\lambda}_k \in \mathcal{L}} z^u_{1,d,k}(\mathrm{U_3}(\hat{\lambda}_k) \otimes \mathrm{I}) + \sum_{\hat{\lambda}_k \in \mathcal{L}} z^u_{2,d,k}(\mathrm{I} \otimes \mathrm{U_3}(\hat{\lambda}_k))$$
$$+ z^r_{1,d}(\mathrm{R_x}(\pi/2) \otimes \mathrm{I}) + z^r_{2,d}(\mathrm{I} \otimes \mathrm{R_x}(\pi/2))$$
$$+ z^{cnot}_{c_{1,2,d}}(\mathrm{CNOT}_{1,2}) + z^{cnot}_{c_{2,1,d}}(\mathrm{CNOT}_{2,1}) + z^{id}_d(\mathrm{I} \otimes \mathrm{I}),$$

$$\sum_{n=1}^{2}\left(\sum_{\hat{\lambda}_k \in \mathcal{L}} z^u_{n,d,k} + z^r_{n,d}\right) + z^{cnot}_{c_{1,2,d}} + z^{cnot}_{c_{2,1,d}} + z^{id}_d = 1$$

Let's formulate it!



N – Number of qubits

D – Maximum allowable depth

Multiplicative property constraints of unitary evolution

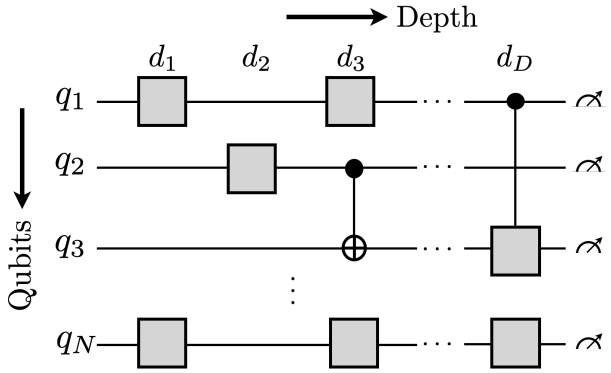
Initial state

$$\mathrm{G}_0 \prod_{d=1}^{D} \mathrm{G}_d = \mathrm{T}^{\mathrm{g}}$$

Target gate

Non-linear, non-convex constraints
Hard to compute!

Let's formulate it!



N – Number of qubits

D – Maximum allowable depth

Multiplicative property constraints of unitary evolution

Initial state → $\mathrm{G}_0 \prod_{d=1}^{D} \mathrm{G}_d = \mathrm{T}^{\mathrm{g}}$ ← Target gate

Recursive matrix linearizations

$$\widehat{\mathrm{G}}_d = \widehat{\mathrm{G}}_{d-1}\mathrm{G}_d \quad \forall d = 2, \ldots, (D-1),$$

$$\widehat{\mathrm{G}}_1 = \mathrm{G}_0\mathrm{G}_1, \quad \widehat{\mathrm{G}}_{d-1}\mathrm{G}_D = \mathrm{T}^{\mathrm{g}}$$

Apply exact bilinear product linearizations (Disjunctive programming)

Enforcing global phase equivalence

$$\underbrace{\widehat{\mathrm{G}}_{d-1}\mathrm{G}_D}_{\widehat{\mathrm{G}}_D} = e^{i\phi}\cdot T^g \quad \Longleftrightarrow \quad \frac{\left[\widehat{\mathrm{G}}_D\right]_{i,j}}{[T^g]_{i,j}} = e^{i\phi} \quad \forall i,j = 1,\ldots,2^N$$

Linear constraints

(1) Choose an I, J such that $[T^g]_{I,J} \neq 0$

(2) $$\frac{\left[\widehat{\mathrm{G}}_D\right]_{I,J}}{[T^g]_{I,J}} = \frac{\left[\widehat{\mathrm{G}}_D\right]_{i,j}}{[T^g]_{i,j}} \quad \forall i,j = 1,\ldots,2^N, \quad \text{such that } i \neq I,\ j \neq J,\ [T^g]_{i,j} \neq 0$$

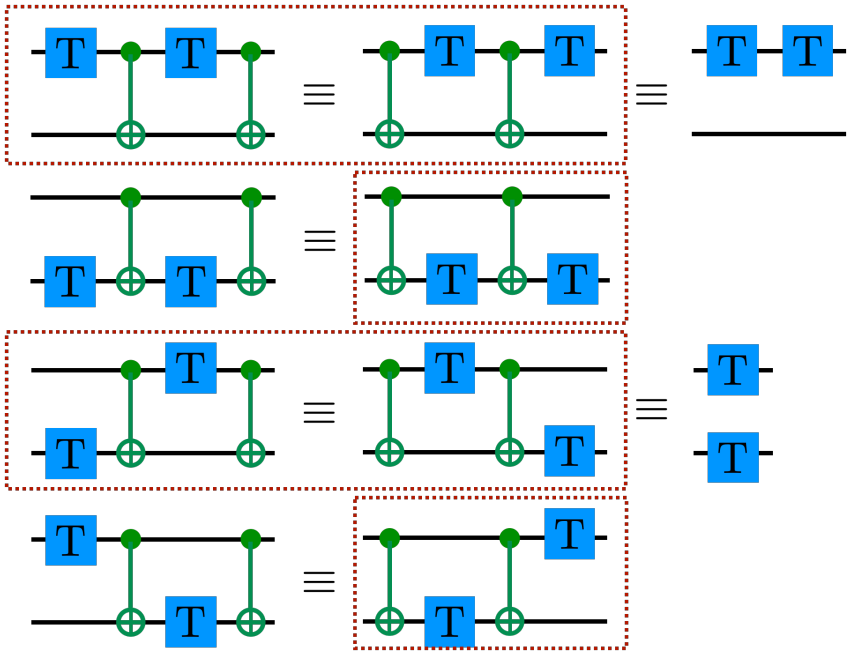
Symmetry elimination via valid constraints

Gate pair equivalence

- Commuting gates: AB = BA
- Involutory gates: $A^2 = I$
- Idempotent gates: $A^2 = A$
- Redundant pairs: AB = CD

Very effective in MIP convergence

Larger depth equivalence



Objective function

- Minimize the total number of one and two-qubit gates

$$\text{minimize} \sum_{d=1}^{D} \left(\sum_{n=1}^{2} \left(\sum_{\hat{\lambda}_k \in \mathcal{L}} z^{u}_{n,d,k} + z^{r}_{n,d} \right) + z^{cnot}_{c_{1,2,d}} + z^{cnot}_{c_{2,1,d}} \right)$$

- Minimize the total number of entangling CNOT gates

$$\text{minimize} \sum_{d=1}^{D} \left(z^{cnot}_{c_{1,2,d}} + z^{cnot}_{c_{2,1,d}} \right)$$

Putting everything together

Inputs

- Number of qubits
- Maximum allowable depth for the circuit
- A target gate
- Set of gates native to the quantum processor

MIP warm-start

Mixed-Integer Program (MIP)

- **Variables:** Binary choices for gate locations, cumulative product unitaries
- **Objective**: Linear function in binaries
- **Constraints**: Linearized, recursive matrix multiplications, Commutative-group symmetry.

Output

Optimal Solution: An exact circuit to implement the target gate up to *global phase* and *machine precision.*

QuantumCircuitOpt (*QCOpt*)

QCOpt's framework

An open-source toolkit for quantum circuit design

From Julia package manager

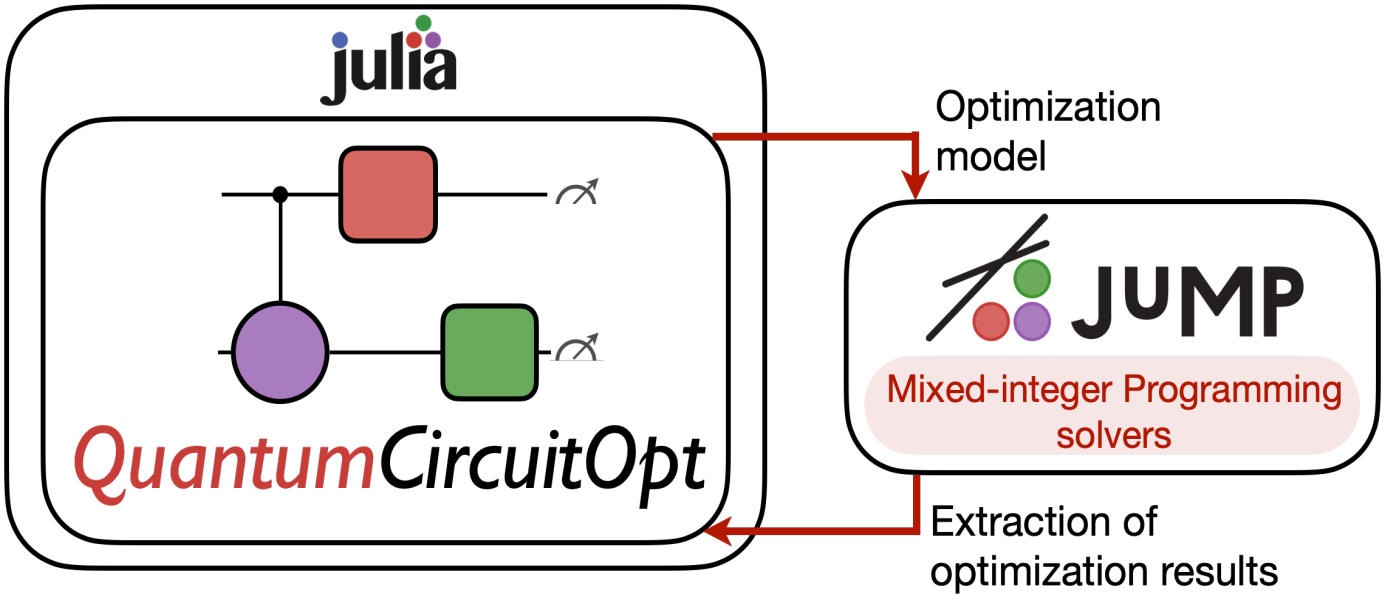
```
julia> import Pkg

julia> Pkg.add("QuantumCircuitOpt")
```

From Github

https://github.com/harshangrjn/QuantumCircuitOpt.jl

QCOpt's framework



A menagerie of quantum gates in QCOpt

Native gates set options

One-qubit gates

$$\{\mathrm{H}, \mathrm{S}, \mathrm{S}^\dagger, \mathrm{X}, \sqrt{\mathrm{X}}, \sqrt{\mathrm{X}}^\dagger, \mathrm{Y}, \mathrm{Z}\}, \{\mathrm{T}, \mathrm{T}^\dagger\}$$

One-qubit gates with angle parameters

$$\{(\mathrm{R_x}(\theta), \mathrm{R_y}(\theta), \mathrm{R_z}(\theta), \mathrm{U_3}(\theta, \phi, \lambda)\}$$

Examples to invoke a gate

```
QCOpt.TdaggerGate()
QCOpt.U3Gate(π,π/2,0)
QCOpt.CNotGate()
```

Two-qubit gates

$$\{\mathrm{CNOT}, \mathrm{Swap}, \mathrm{iSwap}, \mathrm{Magic}, \mathrm{GroverOperator}, \mathrm{Sycamore}, \mathrm{QFT2}\}$$

$$\text{Controlled versions of } \{\mathrm{V}, \mathrm{V}^\dagger\}$$

$$\text{Controlled versions of all one qubit gates}$$

Multi-qubit gates: Multi-control Toffoli, Global rotation gates

Sample circuit decomposition

```
import QuantumCircuitOpt as QCOpt
using JuMP, Gurobi

# Target: Controlled-Z Gate
function target_gate()
    return Matrix{ComplexF64}([1 0 0 0
                               0 1 0 0
                               0 0 1 0
                               0 0 0 -1])
end

params = Dict{String, Any}(
"num_qubits" => 2,
"maximum_depth" => 4,
"elementary_gates" => ["U3_1", "U3_2", "CNot_1_2", "Identity"],
"target_gate" => target_gate(),

"U3_θ_discretization" => -π:π/2:π,
"U3_ϕ_discretization" => -π:π/2:π,
"U3_λ_discretization" => -π:π/2:π,

"objective" => "minimize_depth")

optimizer = JuMP.optimizer_with_attributes(Gurobi.Optimizer)
results = QCOpt.run_QCModel(params, optimizer)
```

Import QCOpt, JuMP and an MIP solver

Input a target gate to be decomposed

Input parameters, including your favorite set of native gates

Euler angle discretizations for the U_3 gate

Choose an objective function

Let QCOpt build and run the optimization model

Output of QCOpt

```
Detected 180 non-unique gates (after discretization)
Detected 1377 input elementary gate pairs which commute
Detected 26 input elementary gate pairs whose product is Identity
Detected 19 involutory elementary gates
Detected 960 redundant input elementary gate pairs
```

QCOpt's pre-preprocessing

```
================================================================================
Quantum Circuit Model Data

  Number of qubits: 2
  Total number of elementary gates (after presolve): 72
  Maximum depth of decomposition: 4
  Input elementary gates: ["U3_1", "U3_2", "CNot_1_2", "Identity"]
    U3_θ discretization: [-180.0, -90.0, 0.0, 90.0, 180.0]
    U3_ϕ discretization: [-180.0, -90.0, 0.0, 90.0, 180.0]
    U3_λ discretization: [-180.0, -90.0, 0.0, 90.0, 180.0]
  Type of decomposition: exact
  MIP optimizer: Gurobi

Optimal Circuit Decomposition

  U3_2(-90.0,0.0,0.0) * CNot_1_2 * U3_2(90.0,0.0,0.0) = Target gate
  Minimum optimal depth: 3
  Optimizer run time: 4.06 sec.
================================================================================
```

controlled-Z gate

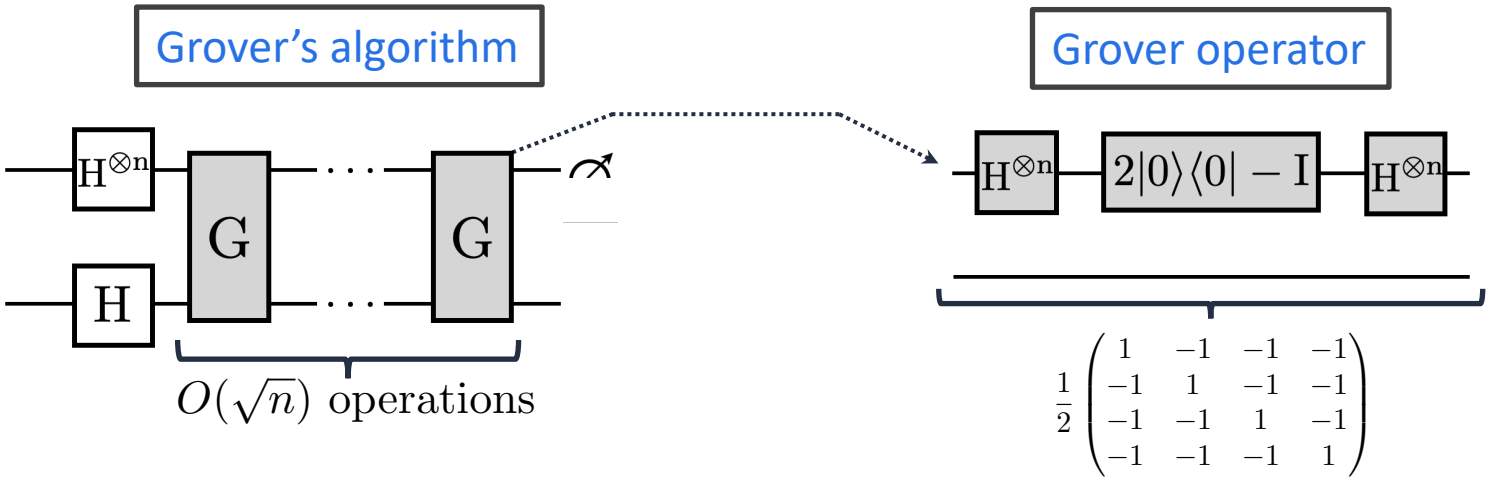
Compact Circuit Realizations

Effect of symmetry breaking valid constraints

N_g: Total number of native gates
D : Maximum allowable depth

			MIP run times (sec.)		
Target gate	N_g	D	**w/o VCs**	**with VCs**	**Speedup**
controlled-Z	72	4	188.4	3.9	48.3x
controlled-V	9	7	63.8	12.2	5.2x
controlled-H	32	5	31.1	4.6	6.7x
Magic basis	72	5	597.7	177.2	3.4x
iSwap	9	10	170.9	6.1	28.0x
Grover	21	10	180.1	1.0	180.1x

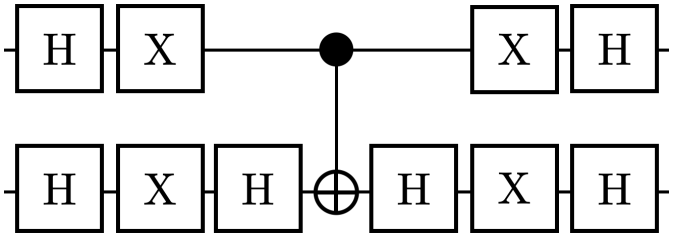
Grover diffusion operator



A key building block of the Glover's algorithm used to find an element (with probability > 0.5) within a randomly ordered database of *n* elements in $O(\sqrt{n})$ operations[a].

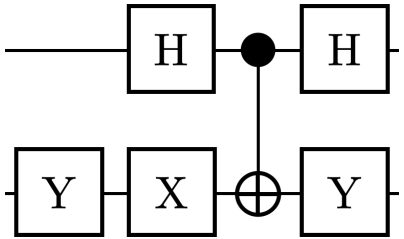
a. Grover, L.K., *A fast quantum mechanical algorithm for database search*. In Proceedings of the twenty-eighth annual ACM symposium on Theory of computing, 1996.

Grover diffusion operator: Equivalent circuits

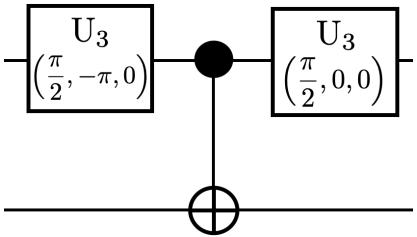


a. https://qiskit.org/documentation/tutorials/algorithms/06_grover.html

Optimal realizations in QCOpt

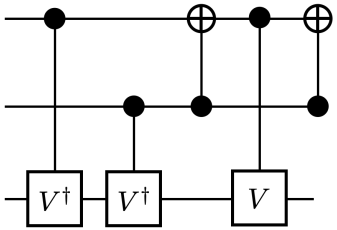


Using Clifford, T & CNOT gates
(42.8% reduction in depth)

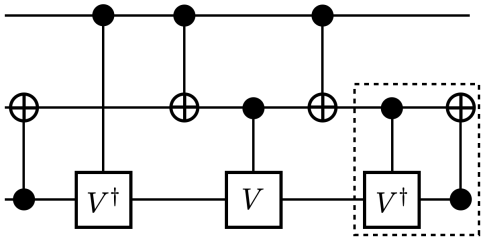


Using U3 & CNOT gates
(57.1% reduction in depth)

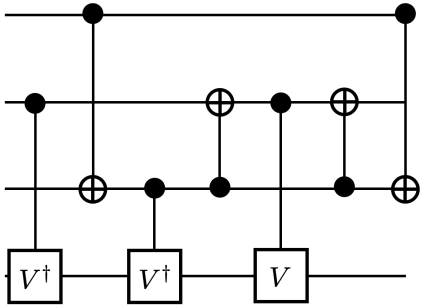
Optimal larger circuits



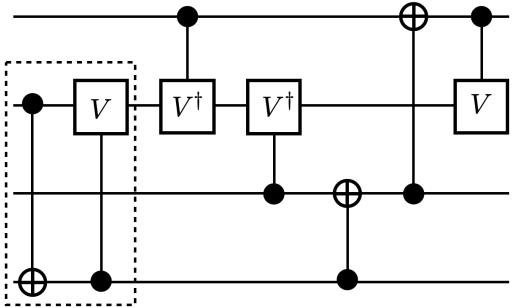
(a) Toffoli (quantum cost = 5)



(b) Fredkin (quantum cost = 5)



(c) Double Toffoli (quantum cost = 7)



(d) Full adder (quantum cost = 6)

QCOpt run times are <= 3 minutes.

For multi-controlled Toffoli gates as elementary gates, QCOpt scales up to 7 qubit circuits.

Collaborators

- Software: Owen Lockwood (RPI), Zsolt Szabo (Macquarie University), Carleton Coffrin (LANL)
- Modeling: Hassan Hijazi (while @LANL), Marc Vuffray (LANL)

Package paper:

Nagarajan, H., Lockwood, O. and Coffrin, C., "*QuantumCircuitOpt: An Open-source Framework for Provably Optimal Quantum Circuit Design*". In *Second International Workshop on Quantum Computing Software,* SC21: The International Conference for High Performance Computing, Networking, Storage, and Analysis (pp. 55-63). IEEE, 2021.

Source: https://doi.org/10.1109/QCS54837.2021.00010

Concluding remarks

- Proposed MIPs, which implements provably optimal methods to compile any arbitrary unitary gate into a sequence of hardware-native gates.

- The MIP formulations are hard to solve; we introduced symmetry-breaking constraints which significantly speed up run times.

- Function-based architecture of the well-documented QuantumCircuitOpt package can enable a robust software infrastructure.

Thank You!

harsha@lanl.gov

https://github.com/harshangrjn/QuantumCircuitOpt.jl