NCA 2025

The 23rd IEEE International Symposium on Network Computing and Applications

Hypergraphic partitioning framework for static and adaptive quantum circuits

Waldemir Cambiucci and Regina Melo Silveira

Department of Computer Engineering and Digital Systems

Polytechnic School

Universidade de São Paulo (USP)

São Paulo – SP – Brazil

waldemir.cambiucci@usp.br

regina.silveira@usp.br

November 5– 7th, 2025

Outline

1. Introduction on Distributed Quantum Computing (**DQC**)
2. Differences between **static** and **adaptive** quantum circuits
3. **Hypergraphic** representation for static and adaptive quantum circuit
4. HyPAQ framework, experiments and results
5. Conclusions and future works

Outline

1. **Introduction on Distributed Quantum Computing (DQC)**
2. Differences between static and adaptive quantum circuits
3. Hypergraphic representation for static and adaptive quantum circuit
4. HyPAQ framework, experiments and results
5. Conclusions and future works

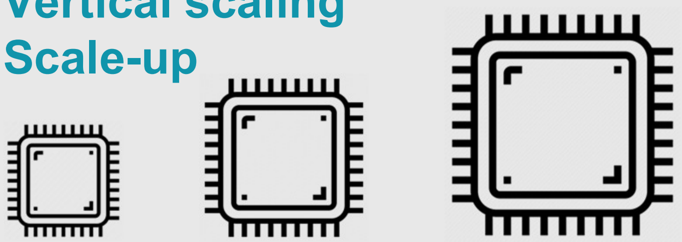
How do we scale quantum machines?

- Through **monolithic** and **distributed** approaches:

Monolithic approach / scale-up

Adding qubits to a single QPU

Vertical scaling
Scale-up



v1 → v2 → v3

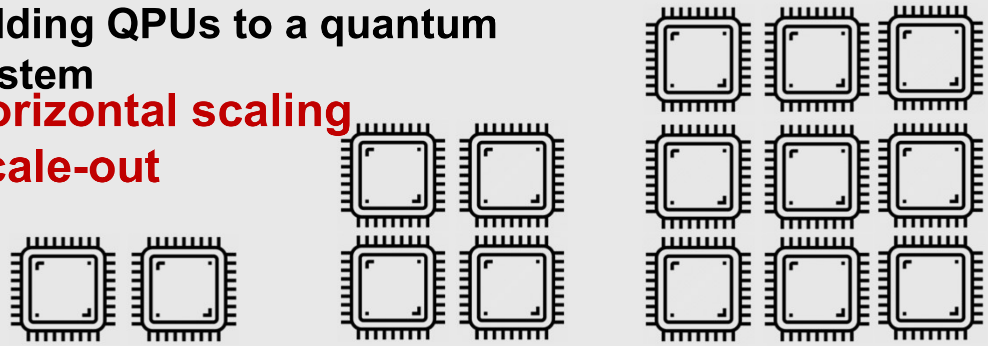
Various challenges depending on qubit technology:

- Photonic: Higher complexity.
- Superconducting: Greater decoherence.
- Trapped-ions: Lower gate fidelity.

Distributed approach / scale-out

Adding QPUs to a quantum system

Horizontal scaling
Scale-out



v1 → v2 → v3

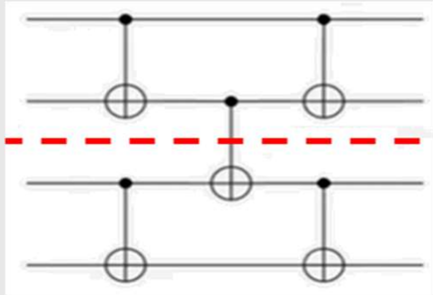
Various challenges for communication, partitioning, and quantum parallelism:

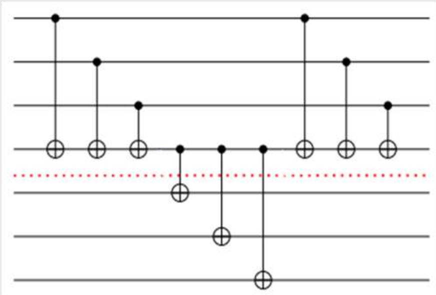
- Need for classical or quantum communication links.
- Teleportation is an expensive and noisy resource.
- Determining the best circuit partitioning points to reduce communication costs and generate smaller subcircuits for distribution is an NP-hard problem.

How can we cut quantum circuits?

- Through a **spatial** or **temporal** approach:

Spatial approach (*gate-cut*)





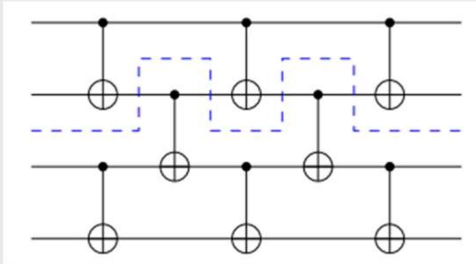
Spatial approach, or gate-cut:

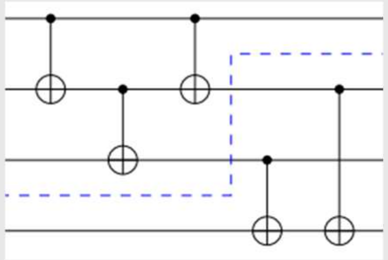
- The circuit is partitioned through binary gates, such as CNOTs or CZs.

- Subcircuits are generated with local operations and non-local operations requiring classical communication.

- The challenge is to minimize the number of cut gates, reducing the communication cost between segments.

Temporal approach (*wire-cut*)





Temporal approach, or wire-cut:

- The circuit is partitioned through qubits, adding a movement qubit in one of the segments.

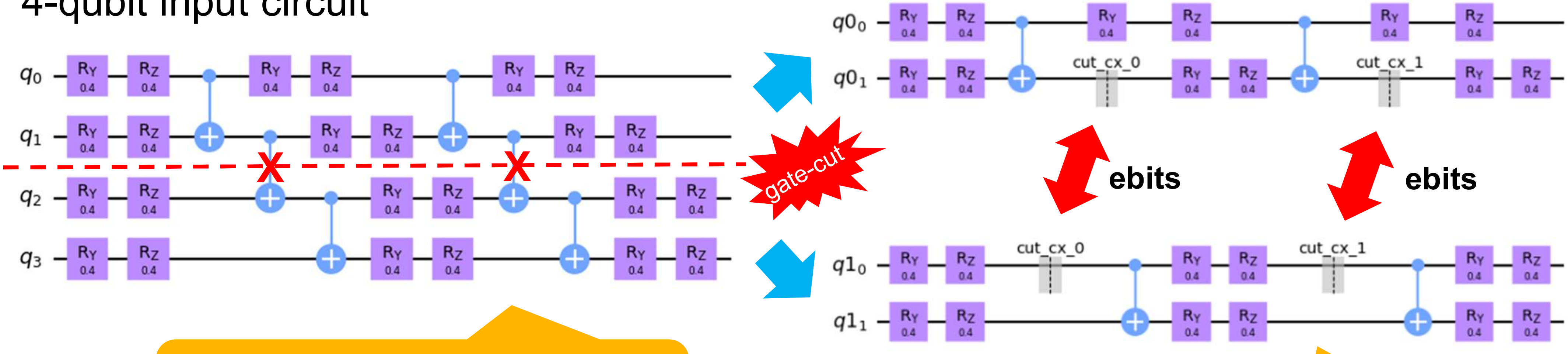
- Temporal cuts can be used to segment a circuit for a single QPU.

- The challenge is to initialize states between segments using simulations in classical post-processing.

Spatial approach (gate-cut) for circuit width reduction

2-qubit subcircuit

4-qubit input circuit



2-qubit subcircuit

Circuit width reduction through quantum gate cutting

ebits = qubits dedicated to communication

PIVETEAU, Christophe; SUTTER, David. Circuit knitting with classical communication. IEEE Transactions on Information Theory, 2023.

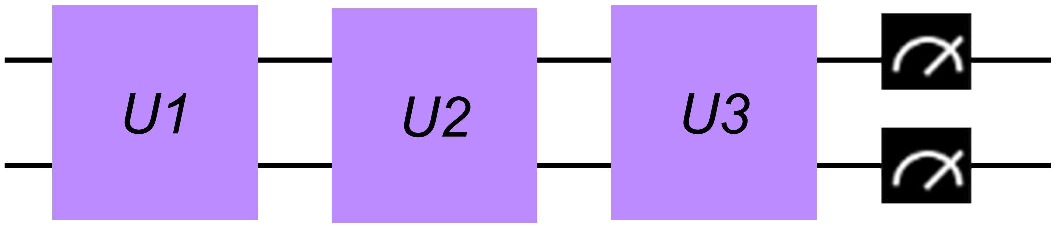
*Simulating large quantum circuits on small quantum computers https://www.ibm.com/quantum/blog/**circuit-knitting**-with-classical-communication*

Outline

1. Introduction on Distributed Quantum Computing (DQC)
2. **Differences between static and adaptive quantum circuits**
3. Hypergraphic representation for static and adaptive quantum circuit
4. HyPAQ framework, experiments and results
5. Conclusions and future works

Static and **Adaptive** Quantum Circuits

a) Static Quantum Circuit

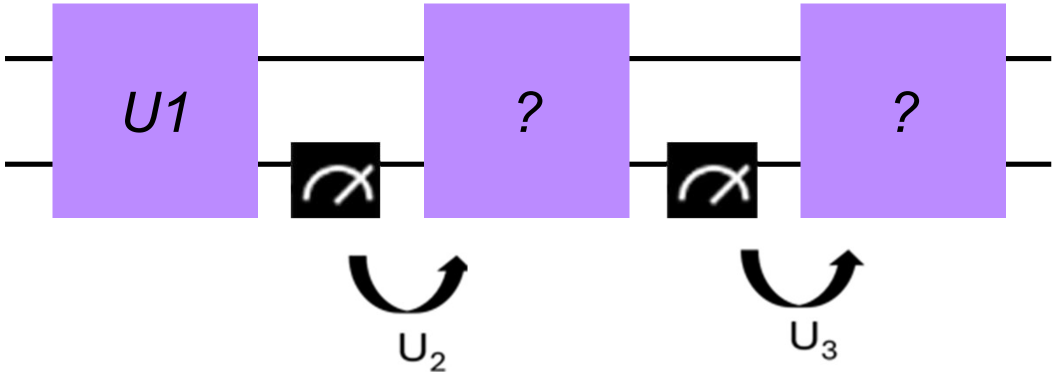


A static circuit C can be represented as:

$$\mathbf{C = U_N U_{N-1} \ldots U_2 U_1}$$

where U_i are unitary operations (quantum gates) applied sequentially on the quantum state $|\psi\rangle$.

b) Adaptive Quantum Circuit



A dynamic quantum circuit C' can be expressed as a sequence of conditional operations:

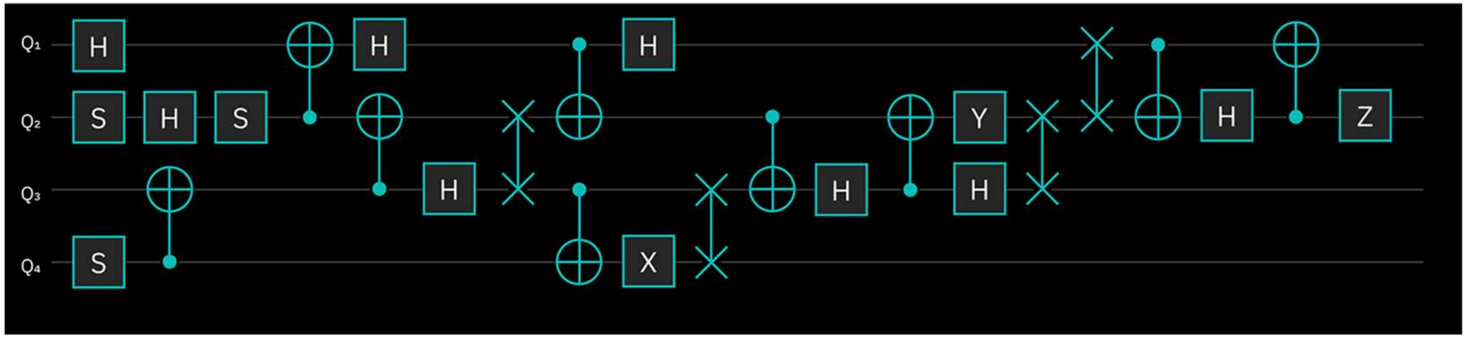
$$\mathbf{C' = (U_N|m_N\rangle)(U_{N-1}|m_{N-1}\rangle)\ldots(U_2|m_2\rangle)(U_1|m_1\rangle)}$$

where $|m_i\rangle$ represents the measurement outcomes that influence the subsequent gate operations U_{i+1}.

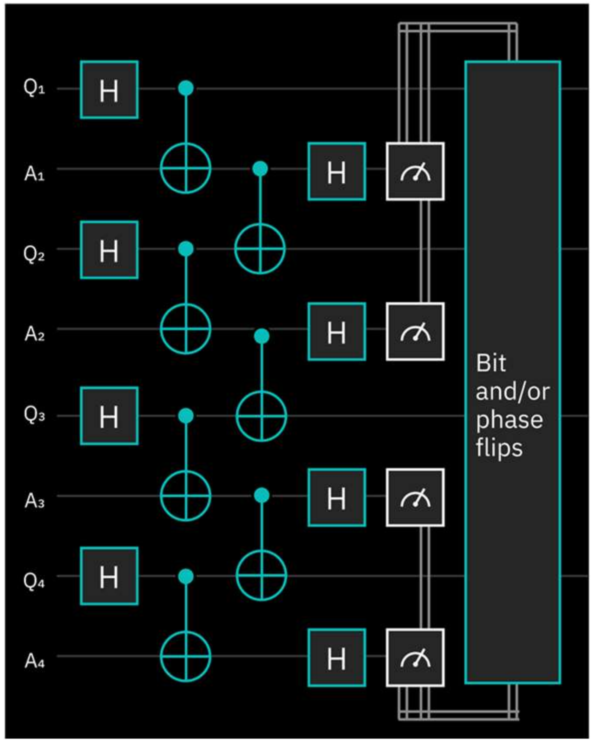
FOSS-FEIG, Michael et al. Experimental demonstration of the advantage of adaptive quantum circuits. arXiv preprint arXiv:2302.03029, 2023. https://arxiv.org/abs/2302.03029

Some QPU processors already support adaptive or dynamic circuits: IBM and Quantinuum.

Static quantum circuit



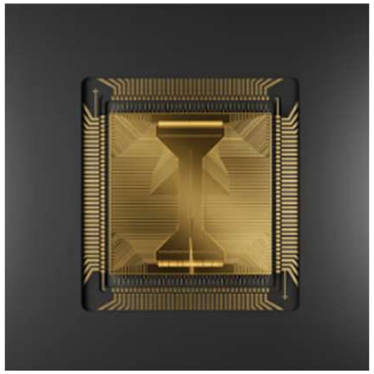
Equivalent adaptive (dynamic) quantum circuit



IBM Processors Heron

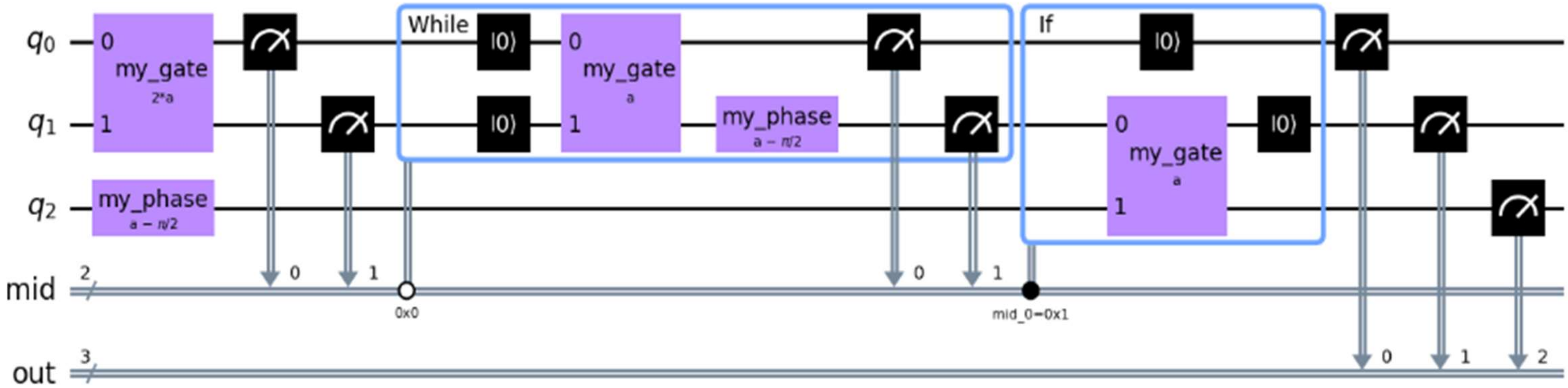
https://www.ibm.com/quantum/blog/quantum-roadmap-2033

Quantinuum Processors H1-1



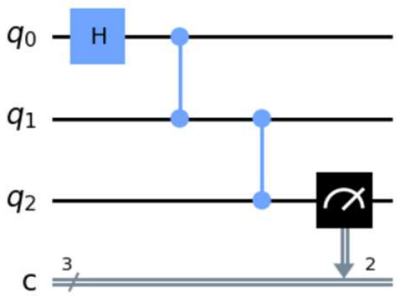
https://www.quantinuum.com/products-solutions/quantinuum-systems/system-model-h1

Example of Adaptive Quantum Circuits

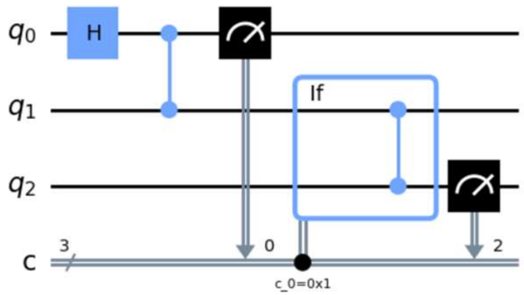


a) example of 3-qubits
adaptive quantum circuit

Adaptive Quantum Circuits can reduce depth, supporting classical control instructions in runtime, such as IF, WHILE, LOOPs



b) example of static circuit

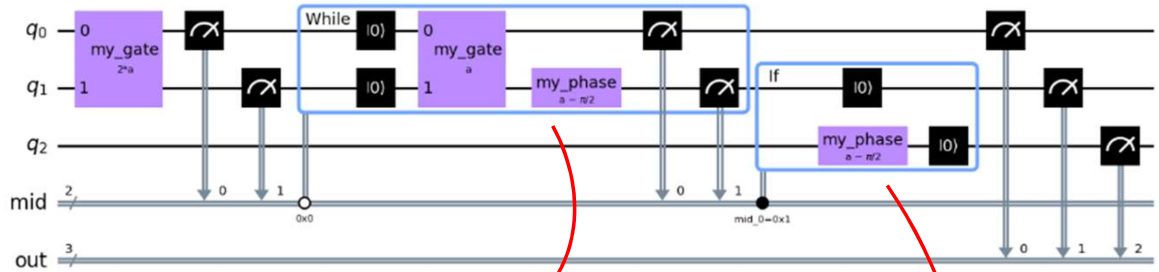


c) example of adaptive circuit

Using OPENQASM3 instructions

```
import qiskit.qasm3

# original circuit - good program
program = """
    OPENQASM 3.0;
    include "stdgates.inc";

    input float[64] a;
    qubit[3] q;
    bit[2] mid;
    bit[3] out;

    let aliased = q[0:1];

    gate my_gate(a) c, t {
      gphase(a / 2);
      ry(a) c;
      cx c, t;
    }
    gate my_phase(a) c {
      ctrl @ inv @ gphase(a) c;
    }

    my_gate(a * 2) aliased[0], q[{1, 2}][0];
    measure q[0] -> mid[0];
    measure q[1] -> mid[1];
```

Adatptive Quantum Circuit




```
    while (mid == "00") {
      reset q[0];
      reset q[1];
      my_gate(a) q[0], q[1];
      my_phase(a - pi/2) q[1];
      mid[0] = measure q[0];
      mid[1] = measure q[1];
    }

    if (mid[0]) {
      let inner_alias = q[{1, 2}];
      my_phase(a - pi/2) q[2];
      reset inner_alias;
    }

    out = measure q;
"""
```

Outline

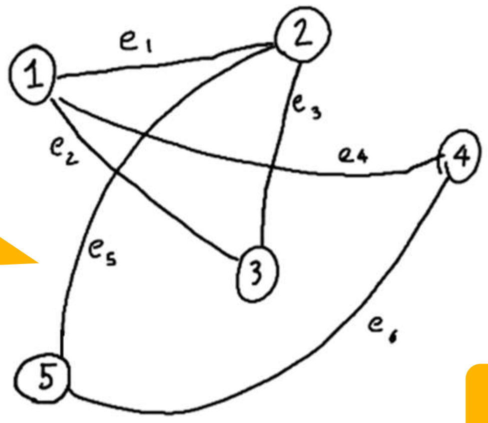
1. Introduction on Distributed Quantum Computing (DQC)
2. Differences between static and adaptive quantum circuits
3. **Hypergraphic representation for static and adaptive quantum circuit**
4. HyPAQ framework, experiments and results
5. Conclusions and future works

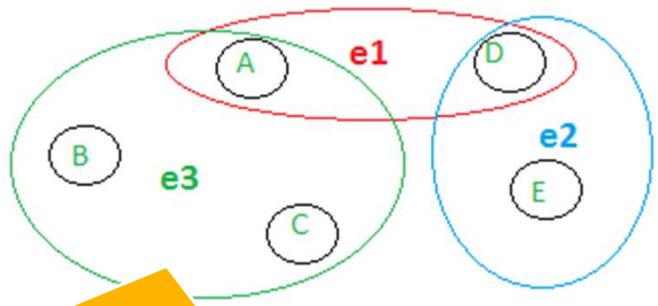
Graphs and Hypergraph

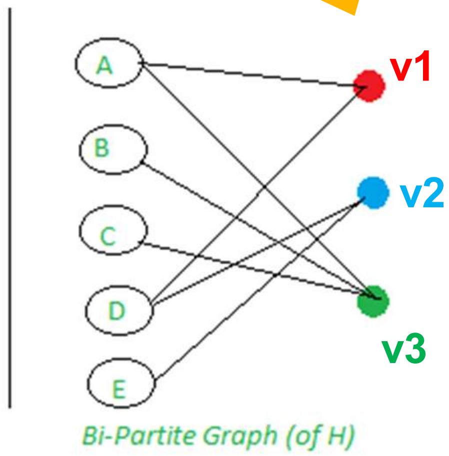
e1 could be CZ or CX

Expanded vertices in the hypergraph

e5 can be **CX** for **q2** and **q5**





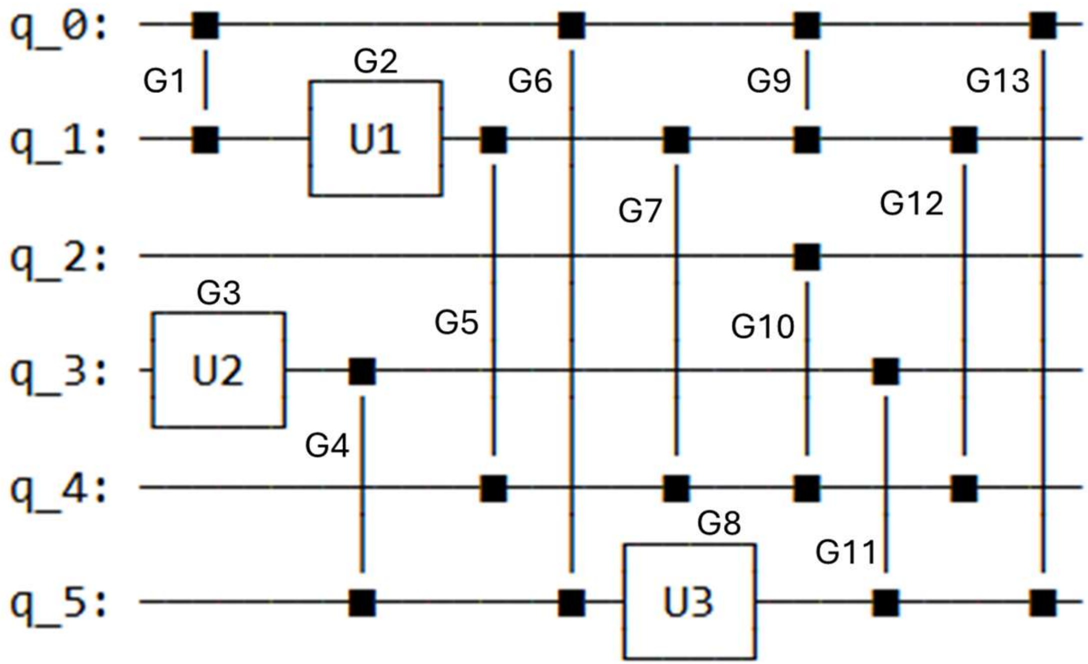


e3 can be a CCX gate

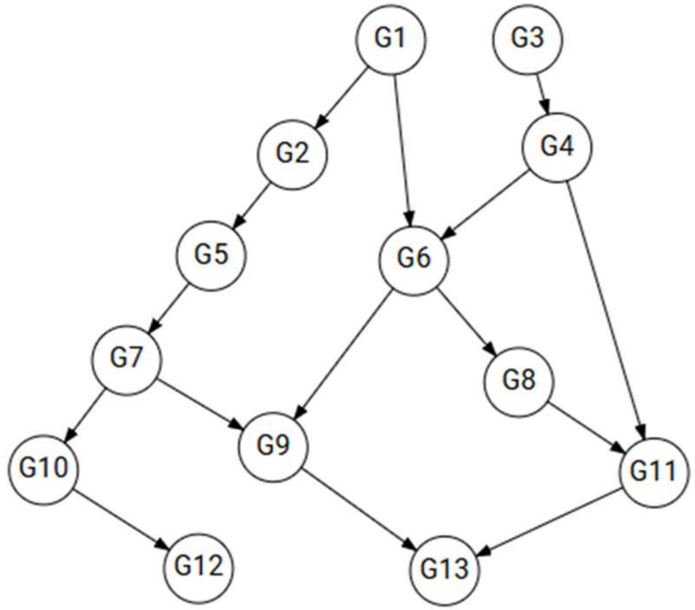
- A **graph** is a mathematical model where **G = (V, E)**, with vertices in **V** and edges in **E,** representing **binary relationships**, where an edge connects only **two elements**.

- A **hypergraph** is a mathematical model where **H = (N, A)**, with nodes in **N** and hyperedges in **A**, representing **non-binary relationships**.
- For example, a hyperedge can involve **more than two elements.**

Graph representation of a quantum circuit



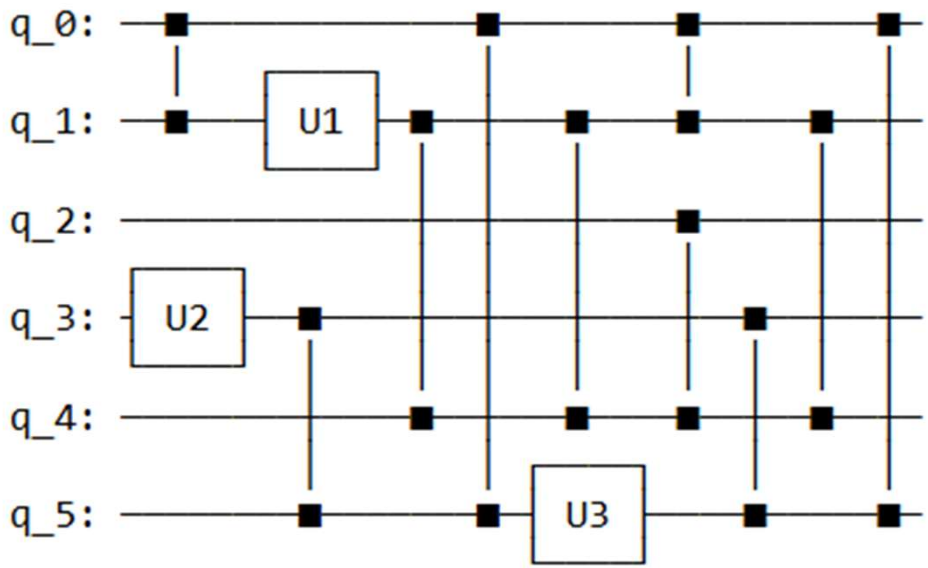
a) Static quantum circuit with 6 qubits, 3 unitary gates, and 10 binary gates.



b) DAG graph representing the static quantum circuit.

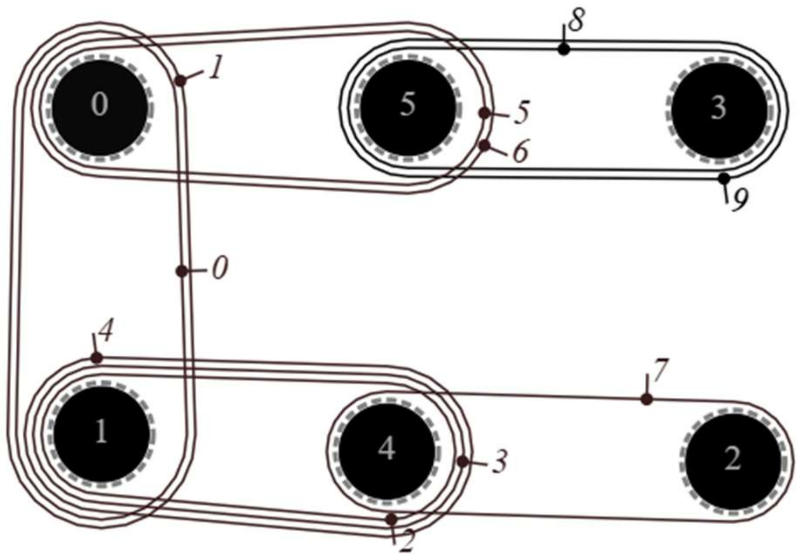
CAMBIUCCI, W., SILVEIRA, R. M., and RUGGIERO, W. V., "Hypergraphic Partitioning of Quantum Circuits for Distributed Quantum Computing," 2023 IEEE International Conference on Quantum Computing and Engineering (QCE), Bellevue, WA, USA, 2023, pp. 268-269, doi: 10.1109/QCE57702.2023.10237.

Hypergraph representation of a quantum circuit



a) Static quantum circuit with 6 qubits and 10 binary gates.

This circuit had only binary gates

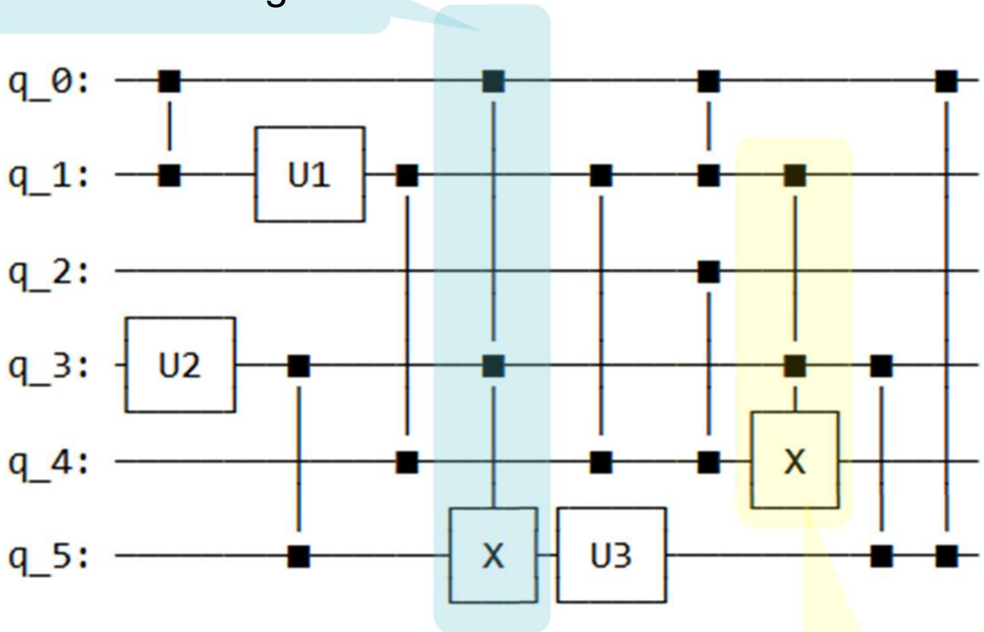


b) Hypergraph representation of the circuit with 6 vertices and 10 hyperedges.

c) Hypergraph H = { "0": ["0", "1"], "1": ["0", "1"], "2": ["1", "4"],
"3": ["1", "4"], "4": ["1", "4"], "5": ["0", "5"],
"6": ["0", "5"], "7": ["2", "4"], "8": ["3", "5"],
"9": ["3", "5"] }

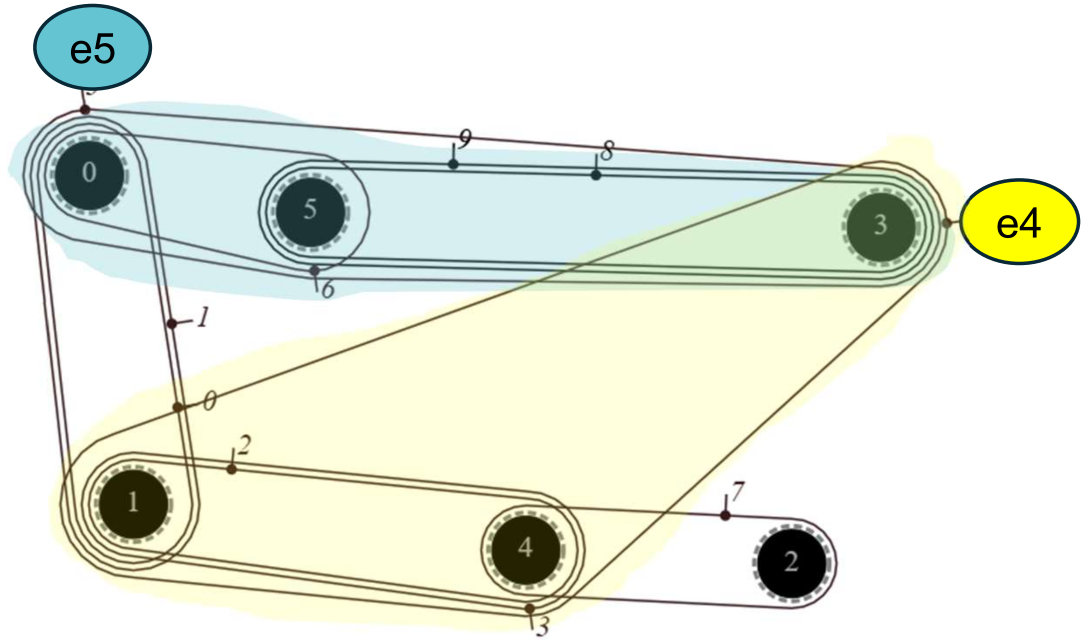
Hypergraph representation of a quantum circuit

e5 is a CCX gate



a) Static quantum circuit with 6 qubits, 8 binary gates, and 2 ternary gates.

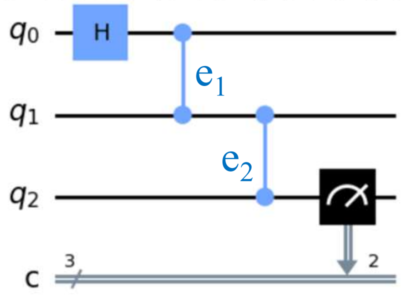
e4 is a CCX gate



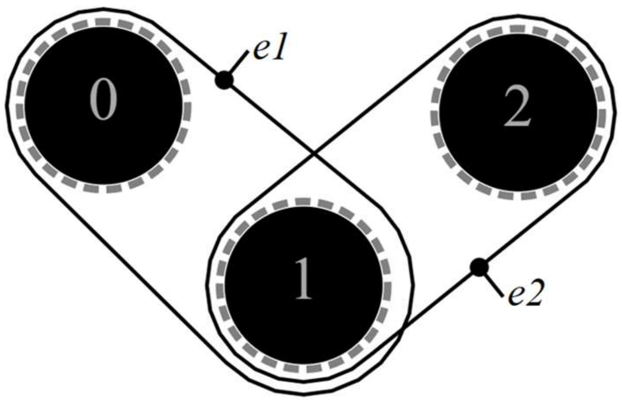
b) Hypergraph representation of the circuit with 6 qubits, 8 binary gates, and 2 ternary gates.

c) Hypergraph H = { "0": ["0", "1"], "1": ["0", "1"], "2": ["1", "4"],
"3": ["1", "4"], "4": ["1", "3", "4"], "5": ["0", "3", "5"],
"6": ["0", "5"], "7": ["2", "4"], "8": ["3", "5"],
"9": ["3", "5"] }

Translating **static circuits** into hypergraphic representation



a) example of static circuit



From the static circuit (a) we have the following components:

- Qubits: q_0, q_1, q_2;
- Gates: G1 (on q_0, q_1), G2 (on q_1, q_2)

The hypergraph representation can be described as:

- Vertices: $V = \{q_0, q_1, q_2\}$;
- Edges: $E = \{e_1$ (G_1 connecting q_0, q_1), e_2 (G_2 connecting q_1, q_2)$\}$

```
H = {
  “e1": ["q0", "q1"],        # e1: Gate G1 on q0, q1
  “e2": ["q1", "q2"]         # e2: Gate G2 on q1, q2
}
```

Translating **adaptive circuits** into hypergraphic representation

In **hypergraph representations for adaptive circuits**, **the idea is to "group" everything that happens inside a single classical control flow block** (like an IF or WHILE) into one hyperedge.

That way, the hyperedge explicitly includes:

- The qubits used by the gates in that block.
- The classical register(s) controlling the block (measurements).
- A label describing the condition structures (for example "WHILE mid==0“, “IF mid==0”, etc.).

This keeps the **quantum+classical dependencies clear** and shows that these operations happen together under a single adaptive control flow.

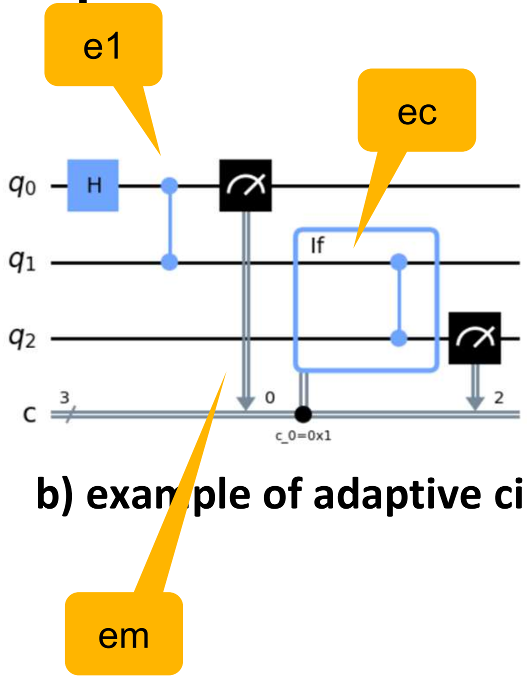
Translating **adaptive circuits** into hypergraphic representation

- Vertices (V): qubits in the adaptive circuit.
- Hyperedges (E): based on three possible sources:
  - *Standard hyperedges: Represent gates that are always executed.*
  - *Conditional hyperedges: represent gates executed only when certain conditions are met.*
  - ***Measurement hyperedges**: represent measurements that influence subsequent gates.*

- **Grouping Gates**:
  - ***Super-Hyperedges:** logical grouping of gates into modules for better partitioning efficiency. Those groups can be part of a classical instruction, such as While or IF.*

- **Weights (w):** reflecting cost of cutting hyperedges, considering execution probability, predefined constraints, or co-design directions.

Representing **adaptive circuits** in hypergraphic representation



b) example of adaptive circuit

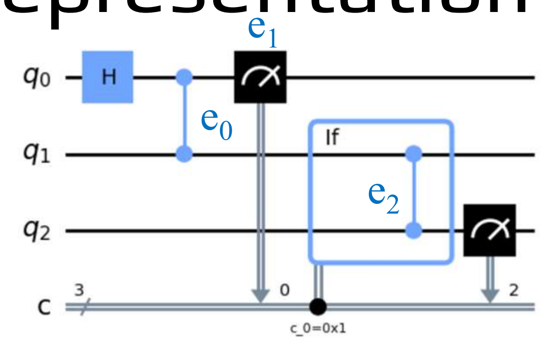
From the adaptive circuit (b) we have the following components:

- Qubits: q_0, q_1, q_2
- Gates:
  - *G1 (on q0,q1); Gc (conditional gate on q1, q2) executed if M(q0) == 1*
- Measurement: $M(q_0)$

The hypergraph representation for this adaptive circuit can be described as:

- Vertices: $V = \{q_0, q_1, q_2\}$
- Hyperedges:
  - *e1: Hyperedge for G1 connecting q0, q1;*
  - *em: Measurement hyperedge for M(q0) connecting q0*
  - *ec: Conditional hyperedge for Gc connecting q1, q2 with condi. Cgc (M(q0) == 1)*
- Weights:
  - *w(e1): Based on G1 properties*
  - *w(em): Based on measurement impact*
  - *w(ec): Weighted by the probability that M(q0) == 1*

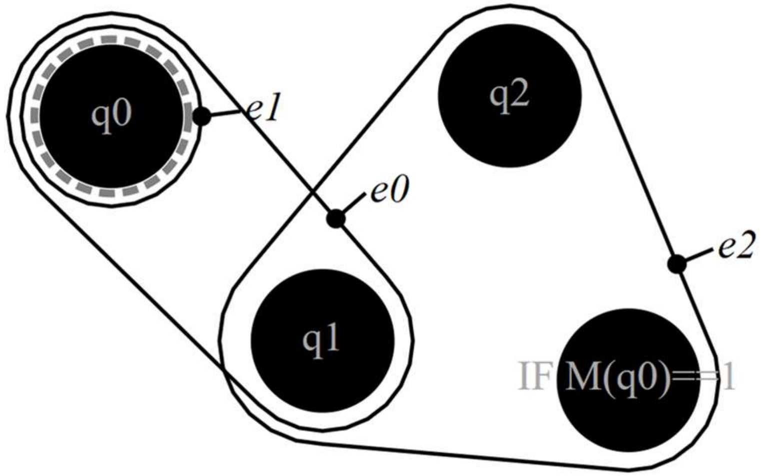
Representing **adaptive circuits** in hypergraphic representation



b) example of adaptive circuit

From the adaptive circuit (b) we have the following components:

- Qubits: q_0, q_1, q_2
- Gates:
  - *G1 (on q0,q1); Gc (conditional gate on q1, q2) executed if M(q0) == 1*
- Measurement: $M(q_0)$

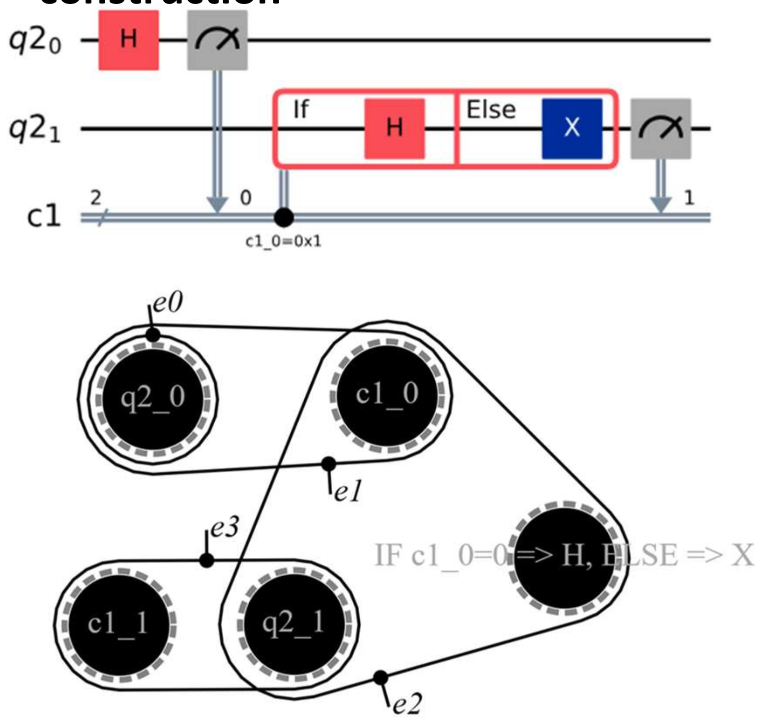



```
H = {
  “e0": ["q0", "q1"],             # e1: Gate G1 on q0, q1
  “e1": ["q0"],                    # em: Measurement on q0
  “e2": ["q1", "q2", "IF M(q0)==1“] # ec: Cond. gate on q1, q2 (exec if
M(q0)=1)
}
```

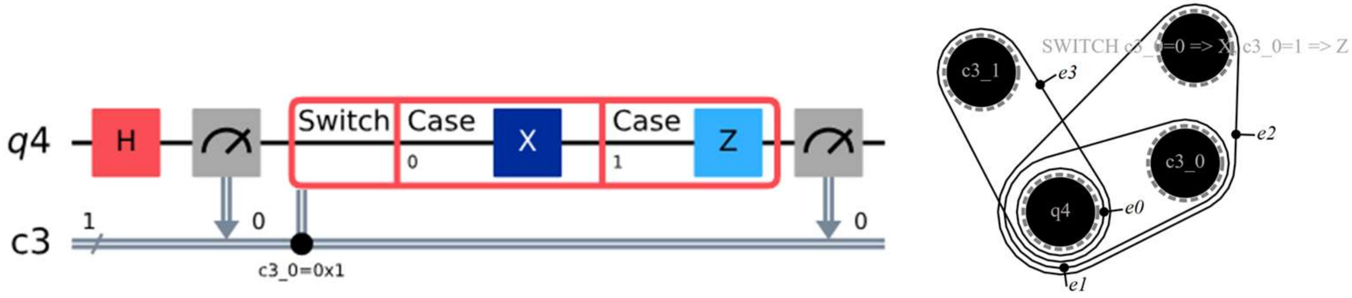
This way, we have additional hyperedges, that can have a bigger weight, grouping gates during the hypergraph partitioning heuristics

Templates for hypergraphic representation of adaptive qc

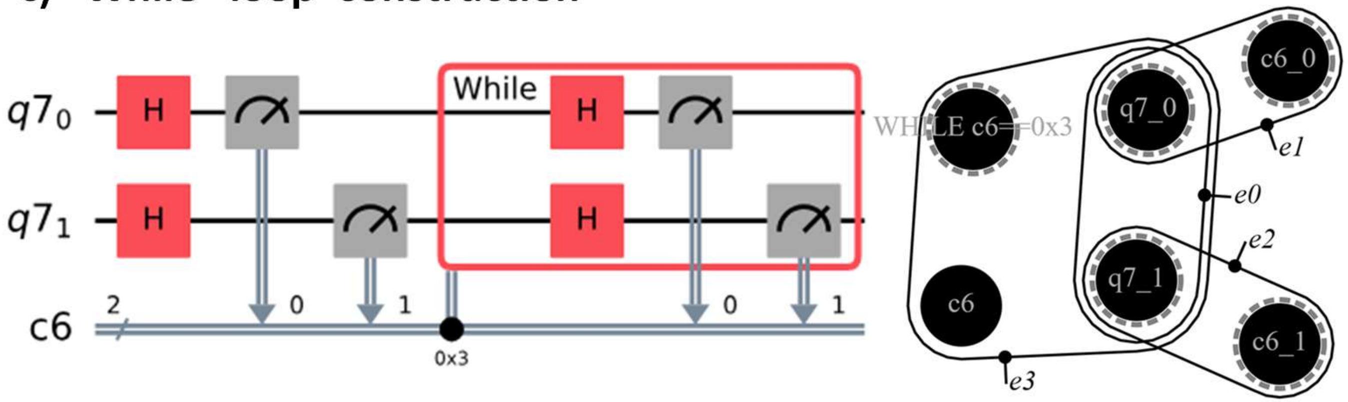
a) Conditional "If Else" construction



b) "Switch Case" construction

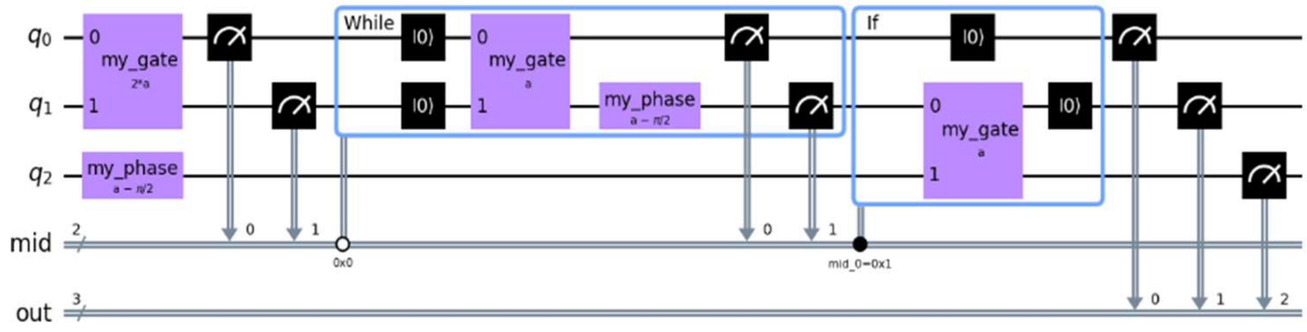


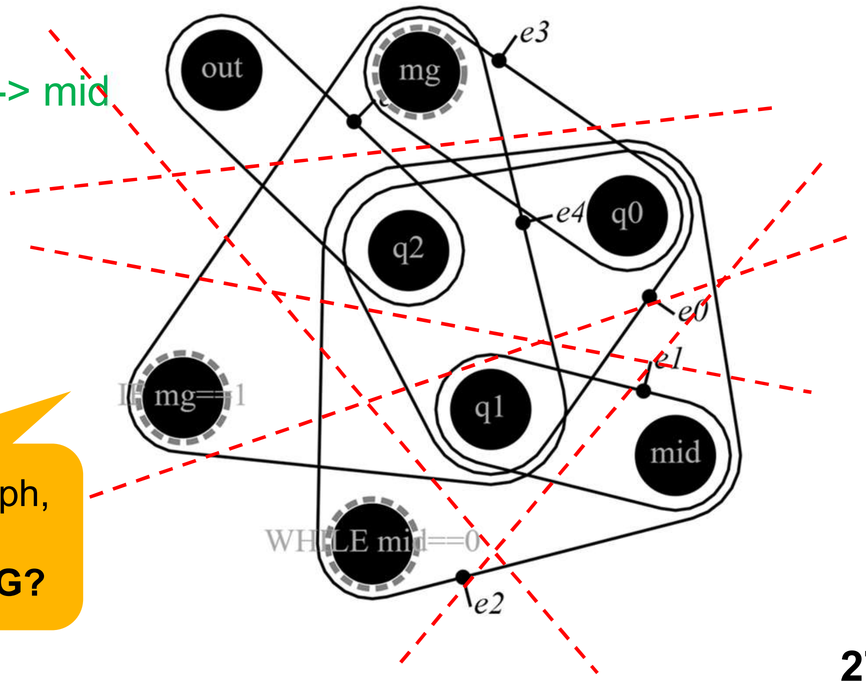
c) "While" loop construction



Representing **adaptive circuits** in hypergraphic representation

```
H = {
    # e0: Initial gates (my_gate on q0,q1;
    # my_phase on q2)
    "e0": ["q0", "q1", "q2"],

    # e1: Measure q1 -> mid
    "e1": ["q1", "mid"],

    # e2: WHILE mid==0 block:
    #     Re-apply my_gate(q0,q1) and my_phase(q2), measure q1 -> mid
    "e2": ["q0", "q1", "q2", "mid", "WHILE mid==0"],

    # e3: Measure q0 -> mg (a second classical bit)
    "e3": ["q0", "mg"],

    # e4: IF mg==1 => apply my_gate on q1,q2
    "e4": ["q1", "q2", "mg", "IF mg==1"],

    # e5: Final measurement of q2 -> out
    "e5": ["q2", "out"]
}
```





With this Hypergraph, how to run the **CIRCUIT CUTTING?**

Circuit Cutting with Hypergraph Partitioning Heuristics

- **Graph** and **hypergraph partitioning** are optimization problems while optimizing certain objective functions, such as minimizing the **number of cut edges** or **balancing the number of vertices in each partition**.
- Several heuristics can be employed for partitioning, such as:

KL is considered the first "good" partitioning heuristic but is very slow.

Big-O (n^2 .log n)

Big-O (n^3 . log n)

Graph Partitioning Heuristics (n is the number of vertices)	**Hypergraph Partitioning Heuristics (n is the number of nodes)**
Spectral Bisection	Recursive Bisection
Kernighan-Lin (KL)	K-way Partitioning
Metis	**Fiduccia-Mattheyses (FM)**
Diffusion-Based Methods	Multilevel Hypergraph Parti oning
Greedy Algorithms	Hypergraph Balancing
Genetic Algorithms	Min-cut algorithm
Stoer-Wagner (SW)	Hybrid Method
Force-Directed Methods	

Big-O(n): Linear time, flexible, and fast.

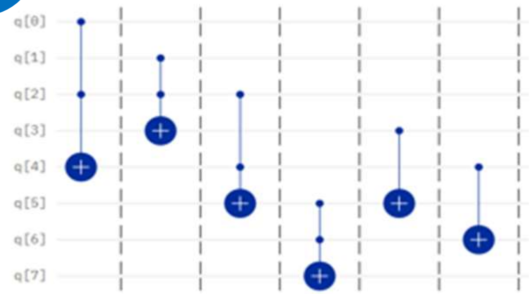
FM is a heuristic designed for hypergraphs and adopted in research.

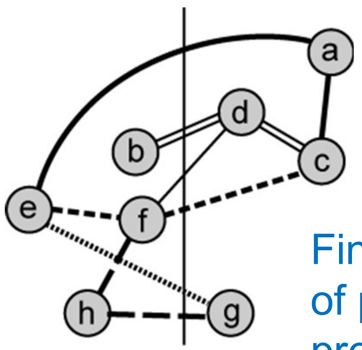
Work very well with unbalanced partitions

CAMBIUCCI, W., SILVEIRA, R. M., and RUGGIERO, W. V., "Hypergraphic Partitioning of Quantum Circuits for Distributed Quantum Computing," 2023 IEEE International Conference on Quantum Computing and Engineering (QCE), Bellevue, WA, USA, 2023, pp. 268-269, doi: 10.1109/QCE57702.2023.10237.

Fiduccia-Mattheyses (FM) heuristic for hypergraph partitioning process

1 8-qubits input circuit



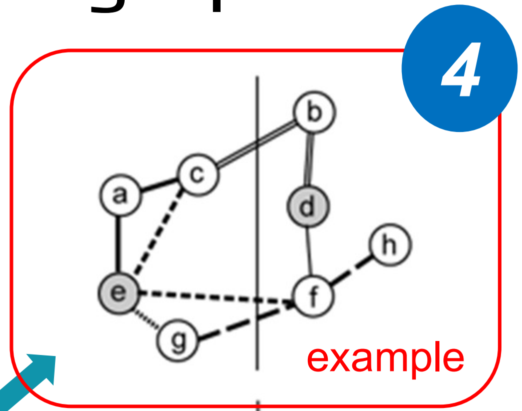


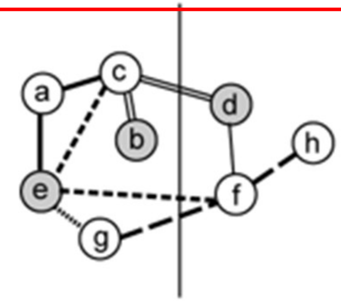
2 Final representation of partitioning process with FM

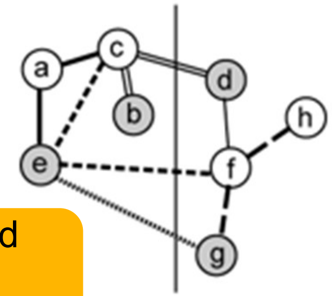
3 Three optimal solutions were found:

i	cell	$g(i)$	$\sum g(i)$	cutsize
0	-	-	-	6
1	e	2	2	4
2	$\boldsymbol{d}$	**1**	**3**	**3**
3	$\boldsymbol{b}$	**0**	**3**	**3**
4	$\boldsymbol{g}$	**0**	**3**	**3**
5	a	-1	2	4
6	f	-1	1	5
7	h	0	1	5
8	c	-1	0	6

4







The number of cuts was reduced from 6 to 3 in three possible configurations

FIDUCCIA,C.M., MATTHEYSES, R. M., "A Linear-Time Heuristic for Improving Network Partitions," 19th Design Automation Conference, 1982, pp. 175-181, doi: 10.1109/DAC.1982.1585498.

Outline

1. Introduction on Distributed Quantum Computing (DQC)
2. Differences between static and adaptive quantum circuits
3. Hypergraphic representation for static and adaptive quantum circuit
4. **HyPAQ framework, experiments and results**
5. Conclusions and future works

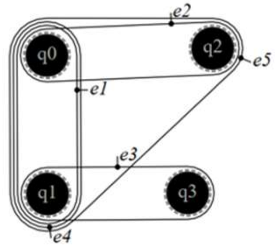
Hypergraphic partitioning method

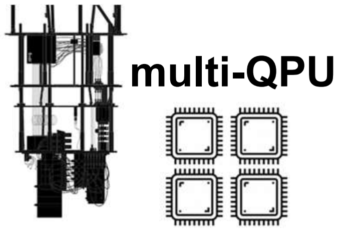
Receiving the input circuit (Static or Adaptive)

Pattern recognition for gate grouping (Optimization and Adaptive)

Translating the circuit as a primal hypergraph

Generation of a hypergraph representation





Hypergraph partitioning (FM heuristic variation)

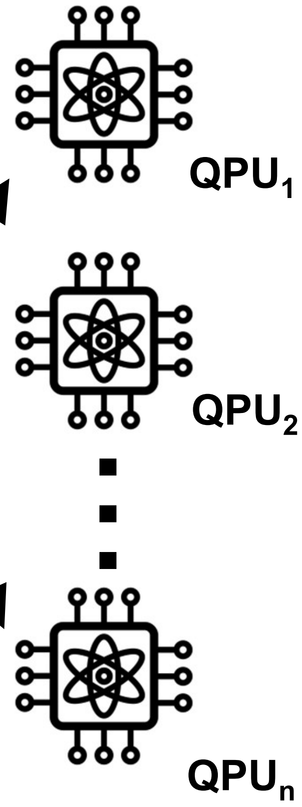
QPU allocation and scheduling (QRM*)

Parameters from the multi-QPU env. (configuration)

Capabilities of each QPU, desired balance between partitions, co-design, constraints,…

QRM – Quantum Resource Manager

Distributed execution on multiple QPUs



W. CAMBIUCCI, R. M. SILVEIRA and W. V. RUGGIERO, "Hypergraphic partitioning for spatial and temporal quantum circuit cutting," 2024 IEEE International Conference on Quantum Computing and Engineering (QCE), Montreal, Canada, 2024, IEEE Catalog Number: CFP24W18-ART, ISBN: 979-8-3315-4137-8.

hypaq / hypaq

Type / to search

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

HyPAQ

HyPAQ - Hypergraph Partitioning for Static and Adaptive Quantum Circuits

PyPI downloads | rate limited by upstream service | stackoverflow | Ask questions | DOI | 10.1109/QCE57702.2023.00055 | Contribute | Good First Issue

🚀 Results for Quantum Circuit Cutting with HyPAQ

This repository presents key results from applying the **HyPAQ (Hypergraphic Partitioning for Static and Adaptive Quantum Circuits)** methodology for **quantum circuit cutting.** HyPAQ enables large quantum circuits to be executed on smaller quantum hardware by partitioning the circuit into subcircuits using advanced **graph** and **hypergraph partitioning techniques**.

The following sections illustrate the performance of HyPAQ in terms of **entanglement reduction** and **partitioning time**.

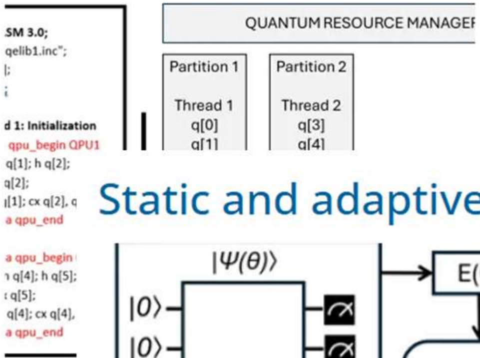
---

https://github.com/hypaq/hypaq/tree/main

IEEEDataPort™ DATASETS | SUBMIT A DATASET | COMPETITIONS | SEARCH IEEE

Search

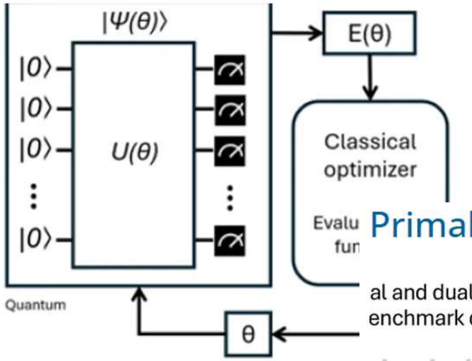
Static and Adaptive Quantum Circuits for Co-Design and Multi-threading Partitioning Approach



Quantum computing stands at the forefront of technological innovation, promising to revolutionize fields ranging from cryptography to material science by leveraging the unique properties of quantum mechanics. Central to the advancement of quantum computing is the development of efficient and scalable quantum circuits, which serve as the fundamental building blocks for quantum algorithms. Traditional static quantum circuits, while powerful, often face limitations in flexibility and efficiency, particularly as the complexity of quantum algorithms increases.

[Sem título]

Static and adaptive quantum circuits for VQE ansatz in hypergraphic partitioning

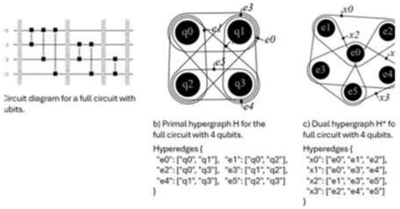


As quantum computing advances, there is a growing need for sophisticated computational methods that efficiently leverage quantum resources. This paper investigates the integration of **adaptive quantum circuits** with the **Variational Quantum Eigensolver (VQE)** algorithm, proposing **Adaptive VQE** as an enhanced approach for dynamically constructing quantum ansätze.

Categories: Other

Primal and dual hypergraphic representation for quantum circuits in hypergraphic partitioning

al and dual hypergraphic representation
enchmark quantum circuits.



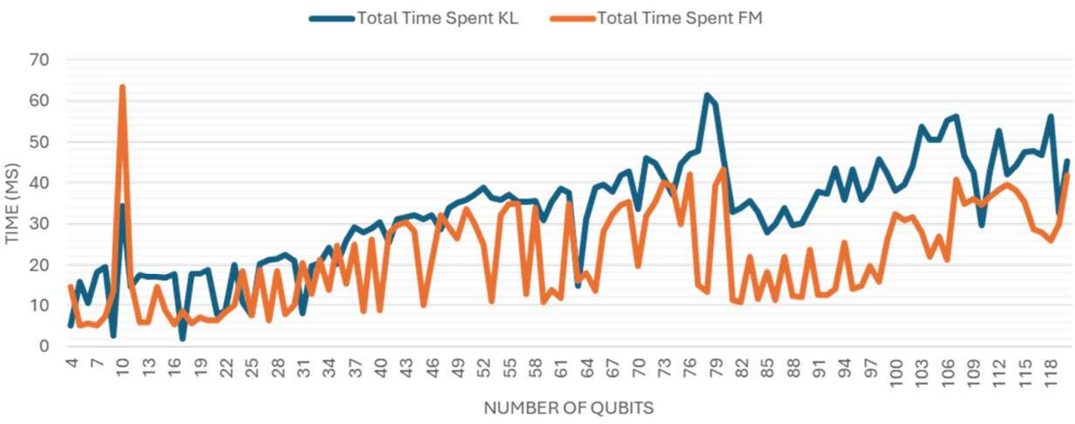
Quantum computing holds the promise of revolutionizing the way we solve complex problems by harnessing the principles of quantum mechanics. However, current noisy intermediate-scale quantum (NISQ) computers face significant limitations due to their small number of qubits and high error rates, making it challenging to execute large quantum circuits that require greater depth, size, or width.

Categories: Other

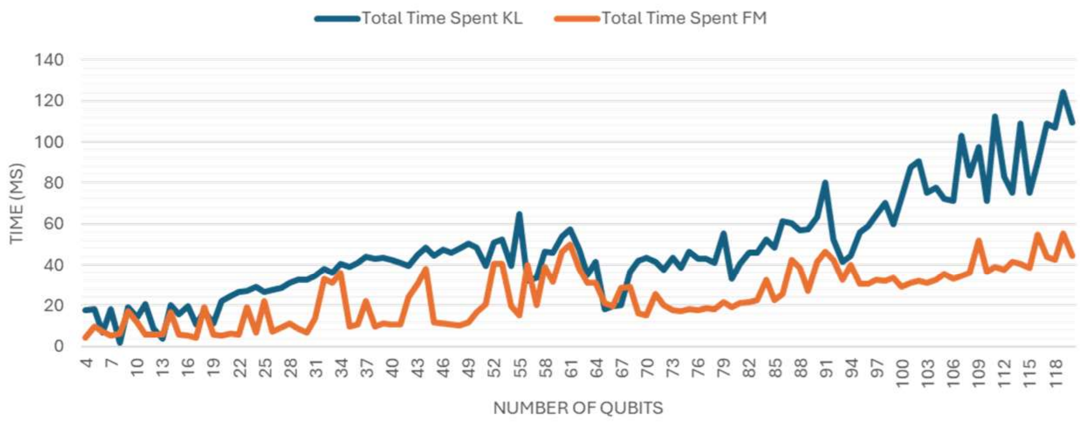
https://ieee-dataport.org/

Comparative performance of FM against KL heuristics

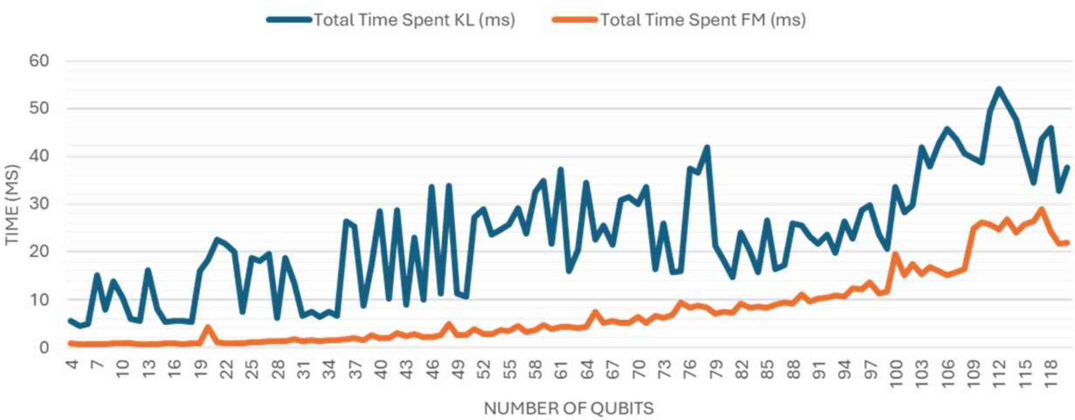
a) Static Full Circuit – Balanced partition for KL & reduced FM



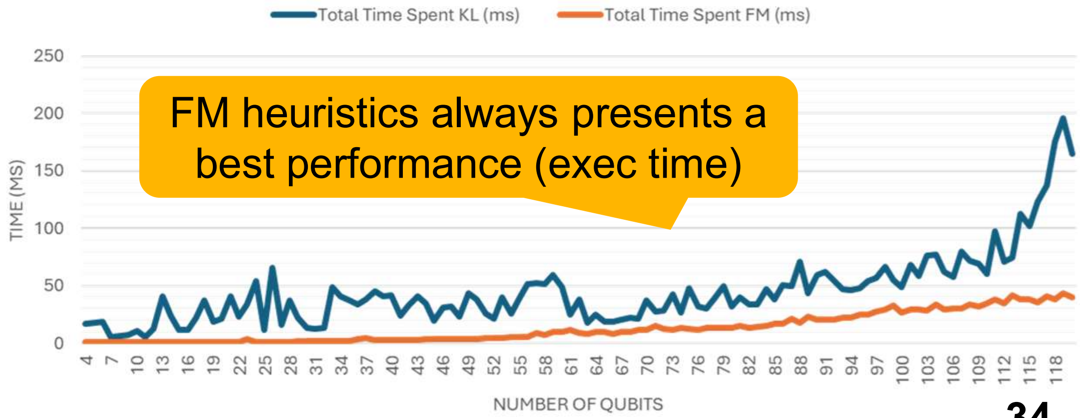
b) Static Random Circuit – Balanced partition for KL & reduced FM



c) Static Full Circuit – Unbalanced partition with reduced FM



d) Static Random Circuit – Unbalanced partition with reduced FM



FM heuristics always presents a best performance (exec time)

The Agenda

1. Introduction on Distributed Quantum Computing (DQC)
2. Static and Adaptive quantum circuits
3. The hypergraphic representation for static and adaptive quantum circuit
4. Experiments and results
5. **Conclusions and future works**

Conclusions

1. **Hypergraphic representation** with groups of gates from classical structures **can support spatial approach** with static and **adaptive quantum circuits**.
2. **Fiduccia-Mattheyses** heuristic reduce communication costs compared to mid-central bipartition scenarios, always with a **better execution time**. Big-O(n) for #nodes.
3. **Hypergraphic partitioning framework** can be a new layer in a full-stack quantum computer, with multi-QPU systems.

Future Work (on going)

1. We are exploring the impact of hypergraph representation for more **complex adaptive quantum circuits**.
2. It´s also possible to combine a **co-design approach** with automatic hypergraphic representation.
3. We are considering the impact of circuit partitioning for applications in chemistry and optimization, using **QPE, VQE, QAOA algorithms** in distributed quantum computing environment.

NCA 2025

The 23rd IEEE International Symposium
on Network Computing and Applications

Thank you for your attention!

UNIVERSIDADE DE SÃO PAULO
SCIENTIA VINCES

UNIVERSIDADE DE SÃO PAULO
ESCOLA POLITÉCNICA

November 5– 7th, 2025