Seminar course

Quantum Software Systems

(aka "qc-systems-seminar")

Kick-off meeting

https://dse.in.tum.de/

Aleksandra Świerkowska
Francisco Romão
Emmanouil (Manos) Giortamis
Prof. Pramod Bhatotia

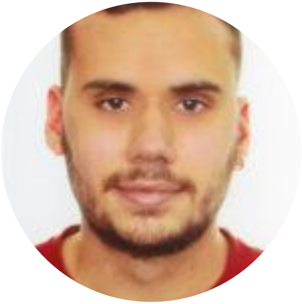
Course instructors

Chair of Computer Systems

https://dse.in.tum.de/team/

Aleksandra Świerkowska

Francisco Romão

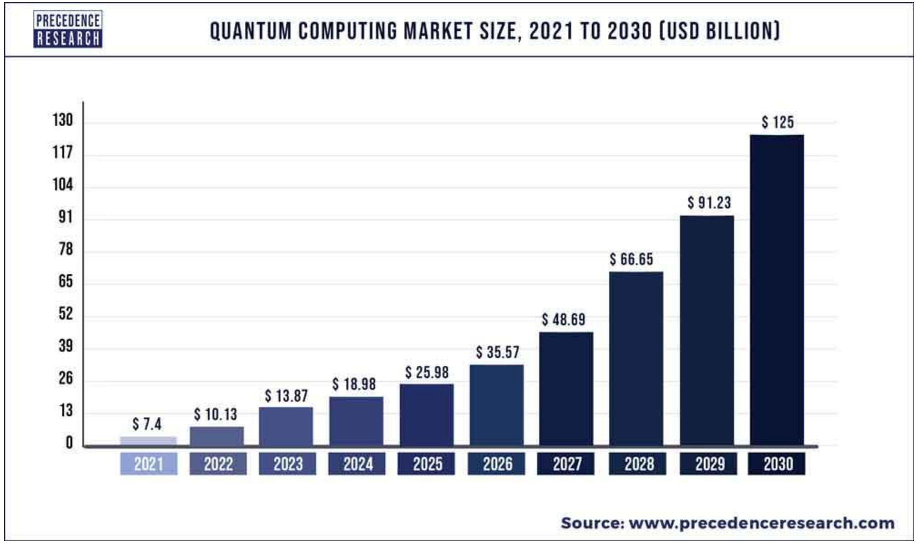


Manos Giortamis

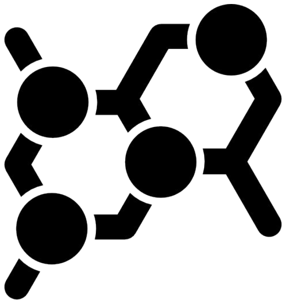
Prof. Pramod Bhatotia

Motivation for quantum computing (QC)

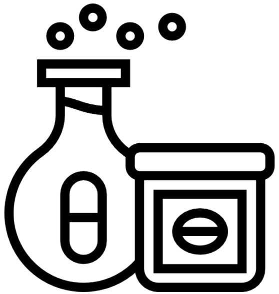
- Quantum computers will be the world's fastest computing devices
  - They can solve problems intractable for classical computers
- QC is still at an infant stage
  - Many open problems and opportunities for research exist
  - Exciting field for exploration and discovery



Applications of QC

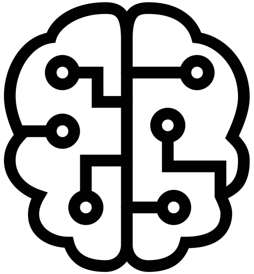


Chemistry

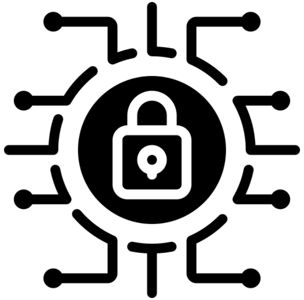


Pharmaceuticals

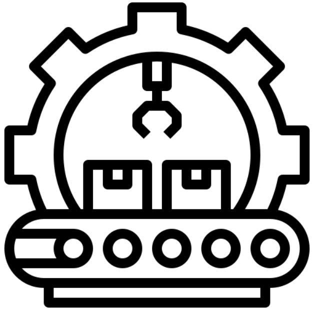
Finance



AI



Cybersecurity



Manufacturing

Source: https://icon-library.com/

Tech giants + startups adopt QC

IBM Q™

Google AI Quantum

aws Amazon Braket

rigetti

IONQ

D-Wave The Quantum Computing Company™

intel®

Atos

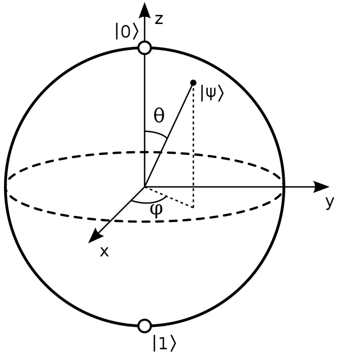
Microsoft

Quantum vs classical computing

	Classical	**Quantum**
Bit	0 or 1	Superposition of 0 and 1
Hardware	$>10^9$ of bits, “perfect”	10^2-10^3 of qubits, noisy
Programming	High-level	Qubit/Gate level
Determinism	Yes	Inherently probabilistic

Qubits

- Quantum devices perform computations on qubits
- Qubit: A two-state quantum-mechanical system
- State of a qubit: Represented as a two dimensional complex vector



Bloch sphere

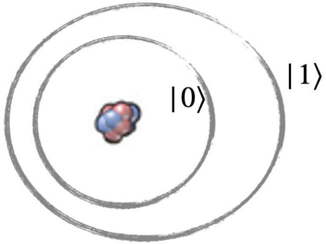
$$|\psi\rangle = a_0|0\rangle + a_1|1\rangle$$

$$a_0, a_1 \in \mathbb{C}, \quad \sum_{k=0}^{n-1} |a_k|^2 = 1$$

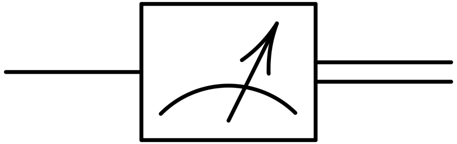
Linear combination of $|0\rangle$ and $|1\rangle$

Superposition and measurement

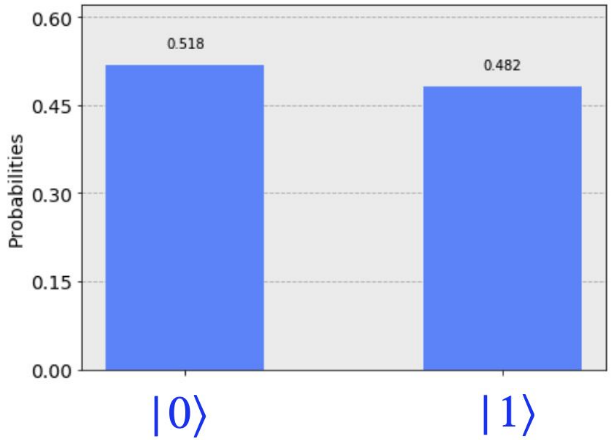
- Superposition: Qubit in state 'between' $|0\rangle$ and $|1\rangle$
- State of a qubit in superposition cannot be observed/copied
- After measuring a qubit, it decoheres to either $|0\rangle$ or $|1\rangle$



Superposition



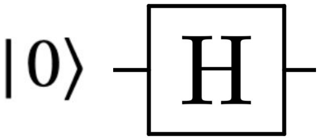
Measurement



Quantum operations

- Quantum Operator = Gate
- Gates are reversible

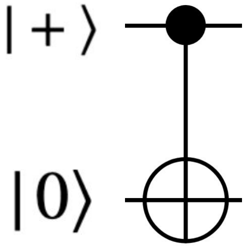
Unary Gates
(e.g. Hadamard Gate)



$$\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) = |+\rangle$$

$$H = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

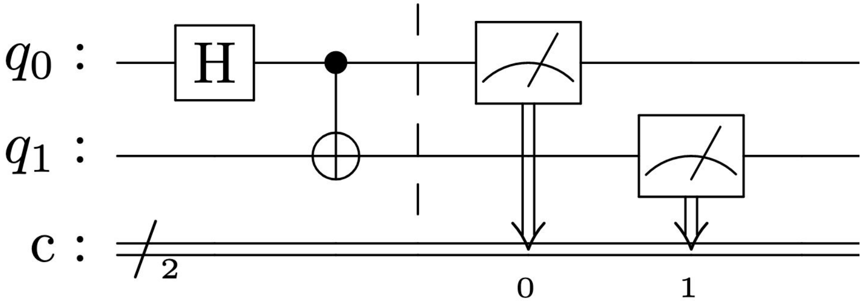
Binary Gates
(e.g. CNOT Gate)



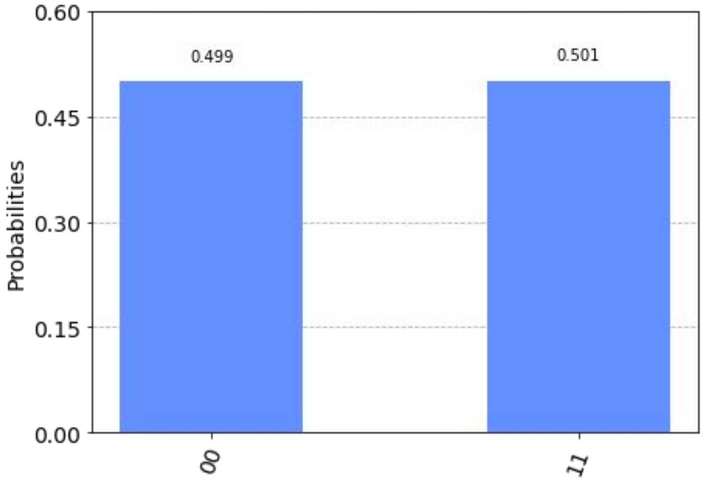
$$\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$$

$$CNOT = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

Quantum circuits

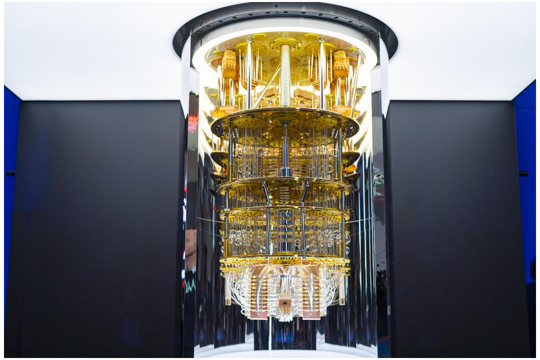


Quantum circuit with 2 qubits, 2 gates and 2 measurements



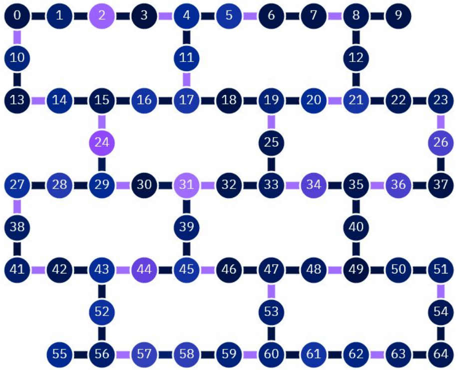
$1e^3$ - $1e^5$ runs

Quantum Computers



Quantum computer

Quantum chip



Qubit topology

Source: https://www.ibm.com/quantum

Programming quantum computers

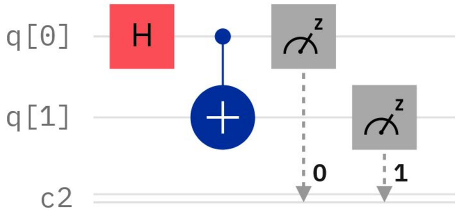
```python
# Importing standard Qiskit libraries
from qiskit import QuantumCircuit, transpile, IBMQ
from qiskit.visualization import plot_histogram

provider = IBMQ.load_account()
backend = provider.backend.ibm_oslo

qc = QuantumCircuit(2)

qc.h(0)
qc.cx(0,1)
qc.measure_all()

qc = transpile(qc, backend)
job = backend.run(qc)
counts = job.result().get_counts()
plot_histogram(counts)
```



Visualised circuit

```
OPENQASM 2.0;
include "qelib1.inc";

qreg q[2];
creg c[2];

h q[0];
cx q[0], q[1];
measure q[0] -> c[0];
measure q[1] -> c[1];
```

Assembly

Programming quantum computers

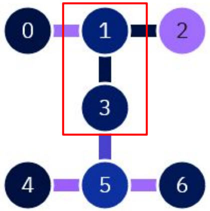
```python
# Importing standard Qiskit libraries
from qiskit import QuantumCircuit, transpile, IBMQ
from qiskit.visualization import plot_histogram

provider = IBMQ.load_account()
backend = provider.backend.ibm_oslo

qc = QuantumCircuit(2)

qc.h(0)
qc.cx(0,1)
qc.measure_all()

qc = transpile(qc, backend)
job = backend.run(qc)
counts = job.result().get_counts()
plot_histogram(counts)
```



QPU

```
OPENQASM 2.0;
include "qelib1.inc";
qreg q[7];
creg meas[2];
rz(pi/2) q[0];
sx q[0];
rz(pi/2) q[0];
cx q[0],q[1];
barrier q[0],q[1];
measure q[0] -> meas[0];
measure q[1] -> meas[1];
```

New assembly

Programming quantum computers

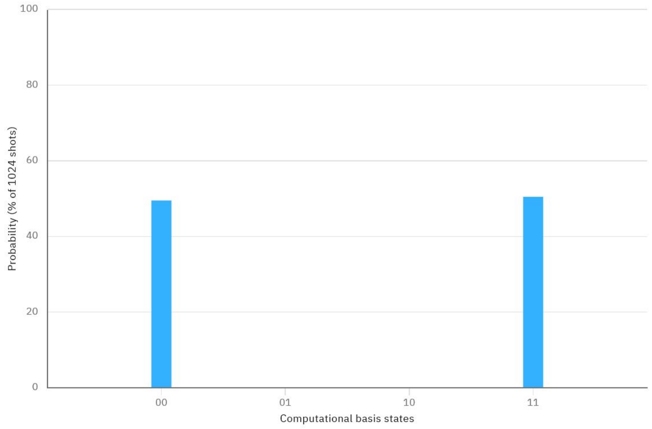
```python
# Importing standard Qiskit libraries
from qiskit import QuantumCircuit, transpile, IBMQ
from qiskit.visualization import plot_histogram

provider = IBMQ.load_account()
backend = provider.backend.ibm_oslo

qc = QuantumCircuit(2)

qc.h(0)
qc.cx(0,1)
qc.measure_all()

qc = transpile(qc, backend)
job = backend.run(qc)
counts = job.result().get_counts()
plot_histogram(counts)
```

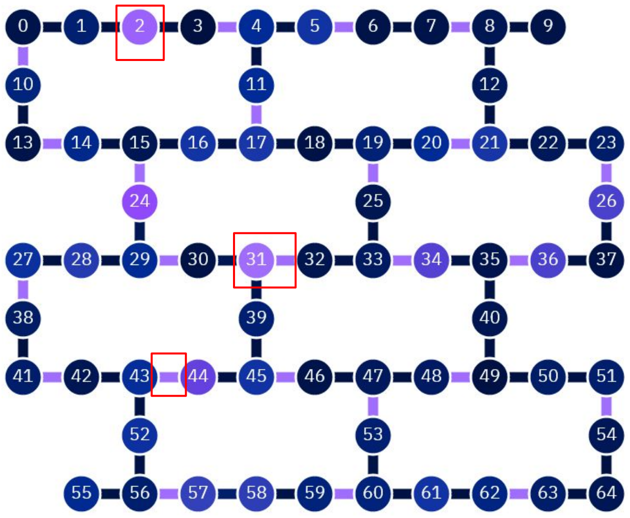


Results

Current state: NISQ era

Noisy Intermediate-Scale Quantum (NISQ) era

- Noisy hardware:
    - Prone to environmental noise
    - Prone to decoherence errors and cross-talk noise
    - Limited error mitigation/correction



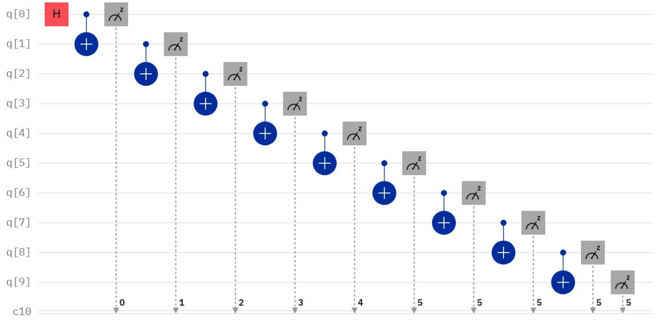
Purple nodes and links have higher noise !

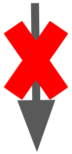
Large quantum circuits on NISQ computers give low-quality results !

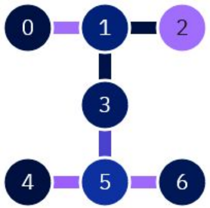
Current state: NISQ era

- Intermediate-Scale:
  - Currently up to a few 100s of qubits
    - 10.000s needed for quantum advantage
  - Low quantum-volume
  - Limited qubit connectivity

Quantum circuit







QPU

Large quantum circuits can’t be transpiled on the QPUs !

Research questions

Existing QC hardware is limited in terms of quantity and quality

Which are the research challenges for software systems in quantum computing?

Tentative topics

Papers from top conferences (e.g., ASPLOS, HPCA, MICRO, PLDI)

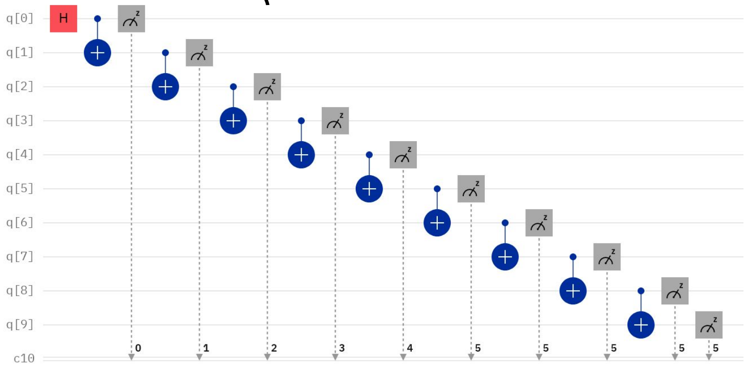
Tentative topics
#1: Transpilation (qubit mapping)
#2: Quantum resource management
#3: Circuit cutting & knitting
#4: Circuit multiprogramming
#5: Circuit transformations
#6: Quantum Error Correction

Challenge #1: Transpilation (qubit mapping)

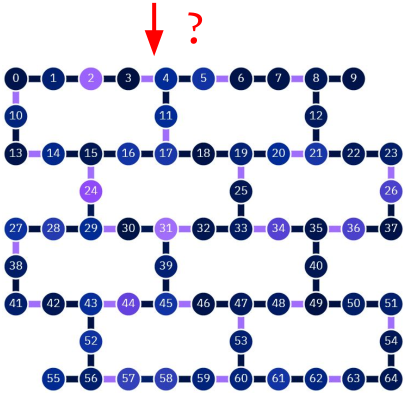
Transpilation modifies a given circuit to match the topology of a specific quantum backend

- How can we optimally map logical qubits to physical qubits?
  - Avoid noisy qubits
  - Avoid noisy qubit links
  - Minimise SWAP operations
- How can we do it fast?
  - NP-hard problem
  - Greedy approaches
  - Heuristics

Quantum circuit



?

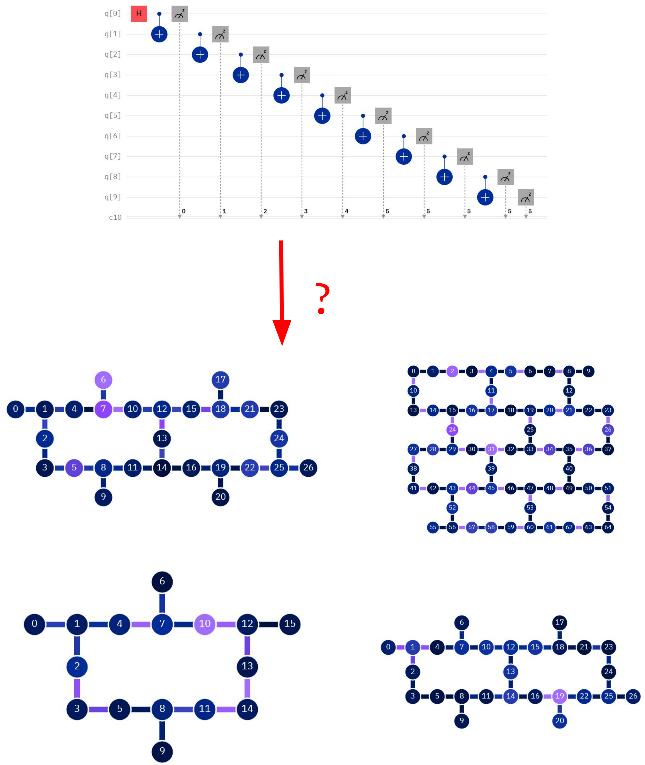


Qubit topology

Challenge #2: Quantum resource management

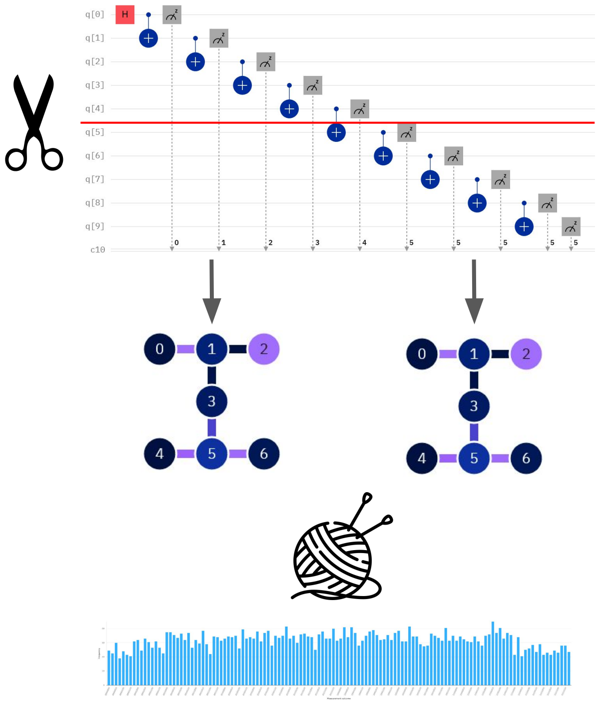
Selecting the best machine for a given quantum circuit is challenging

- Which machines does the circuit fit into?
- Which machines support the circuit's operations?
- Which topology best fits our circuit?
- Which machine has the best noise properties?



Challenge #3: Circuit cutting and knitting

Circuit cutting and knitting is a method of dividing a quantum circuit into smaller fragments, executing them and merging the results back

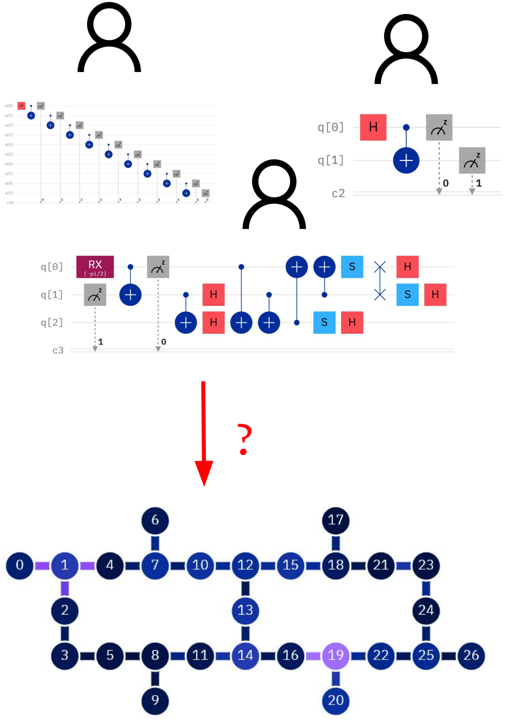


Results

- Which are the optimal cut locations?
- What is the additional quantum cost ?
- What is the additional classical cost?
- How can we mitigate the (exponential) costs?

Challenge #4: Circuit multiprogramming

Multiprogramming enables multiple circuits to be executed on the same QPU in parallel

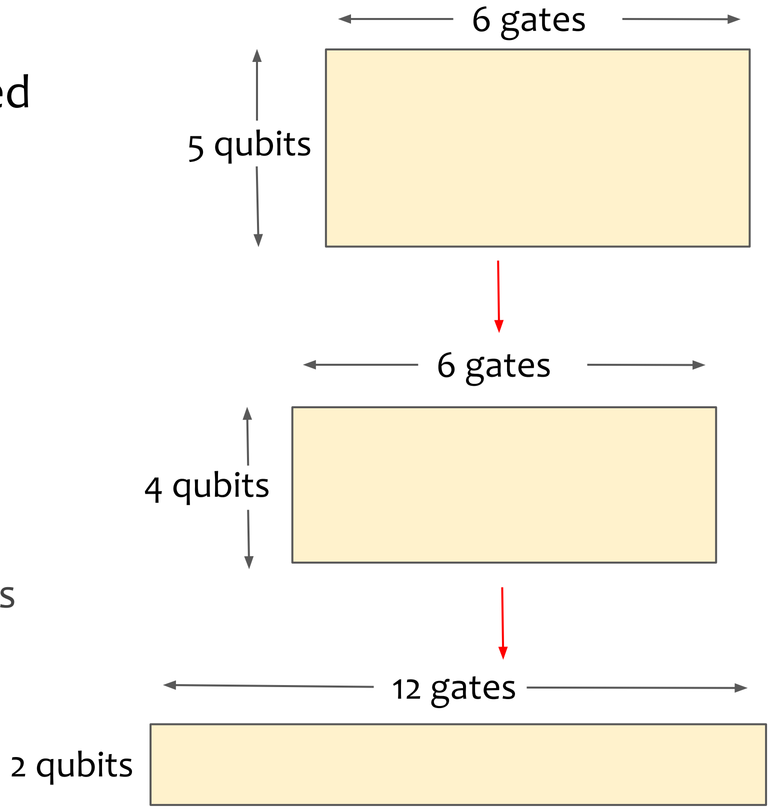


- How can we optimally map multiple circuits on a QPU?
  - Equally good partitions
- How can we minimise circuit interference ?
  - Crosstalk noise
  - Measurement interference
  - Unequal depths

Challenge #5: Circuit transformations

There are circuit transformations that aim to change a circuit in a way that its fidelity is improved

- Circuit compaction
  - Width (# of qubits) reduced
  - Depth (# of consecutive gates) increased
- How to use it optimally ?
  - Tradeoff between allocating less qubits but adding intermediate measurement operations
  - Possible decoherence errors

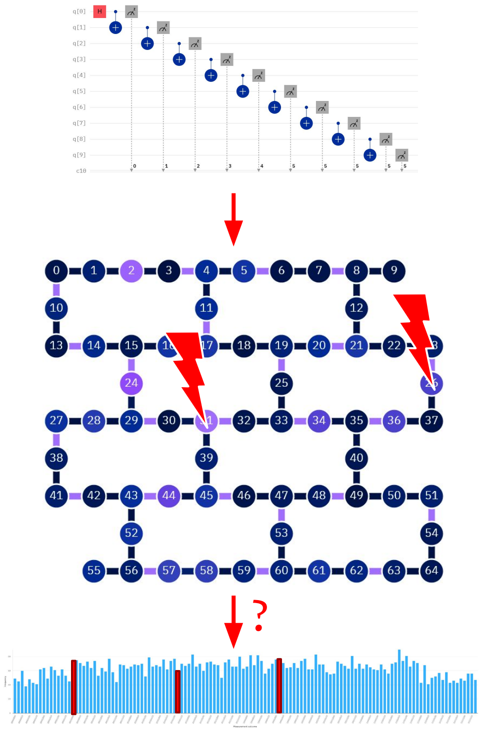


Challenge #6: Quantum Error Correction

Quantum Error Correction is a technique allowing for detecting and correcting errors occurring during execution:

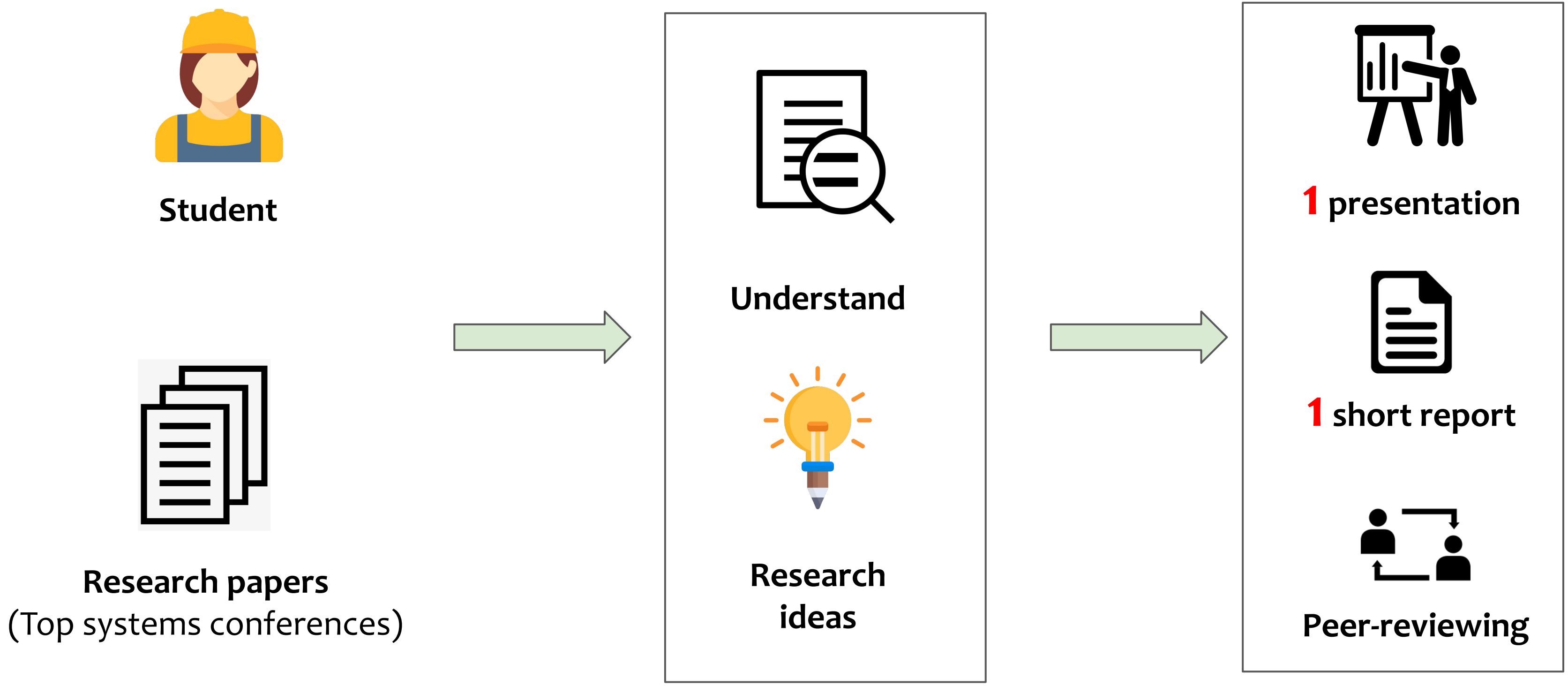
- How can we protect a circuit against errors of the device (e.g., gate errors)?
  - Encode circuit with a QEC code
  - Cyclically check if the states were not changes
  - Use heuristic to predict the error patterns and propose correction strategy

- How can we provide strongest protection?
- How can we predict the patterns most accurately?
- How can we do it with limited number of qubits?
- How can we do it fast?

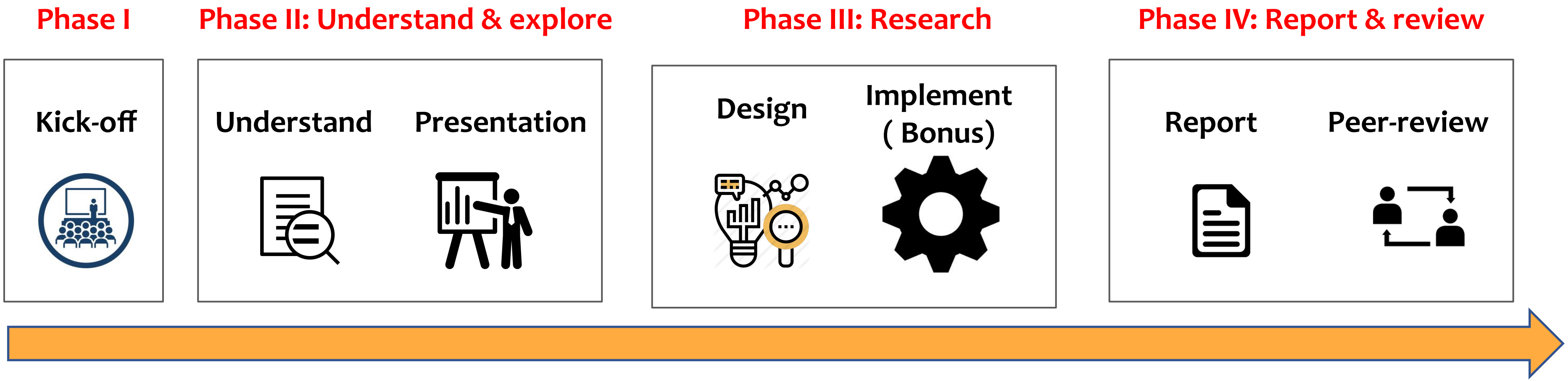


Format

Bird's eyes view

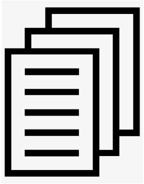


Overview


Phase I
Phase II: Understand & explore
Phase III: Research
Phase IV: Report & review
Kick-off
Understand
Presentation
Design
Implement
(Bonus)
Report
Peer-review

Phase I: Kick-off meeting

Format and motivation
(all participants meeting)



Paper selection
(Top systems conferences)

The first week

NOTE

1. A list of papers will be provided for FCFS bidding
2. Paper presentation guidelines will be provided for the next phase

Phase II: Understand & explore

Understand the paper(s)

Focus

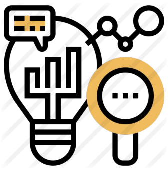
1. **Understand** the paper and related work
2. Also **explore** a “laundry list” of research ideas/directions

Paper presentation

Focus

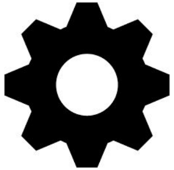
1. Explain the work/related work (**“why?”** and **“how?”**)
2. Explain and discuss all possible research directions
3. Pick a research direction

Phase III: Research



Research work

Focus:
Indepth research work to nail-down the problem and detailed approach to solve it!



Research prototype

Bonus:
(Optional)
“Build the system to solve it!” and show us the working idea and associated results

Phase IV: Report & review

Report

Focus

Prepare a single “short & sweet” report summarizing

(a) Paper

(b) Research work

Peer-review

Focus

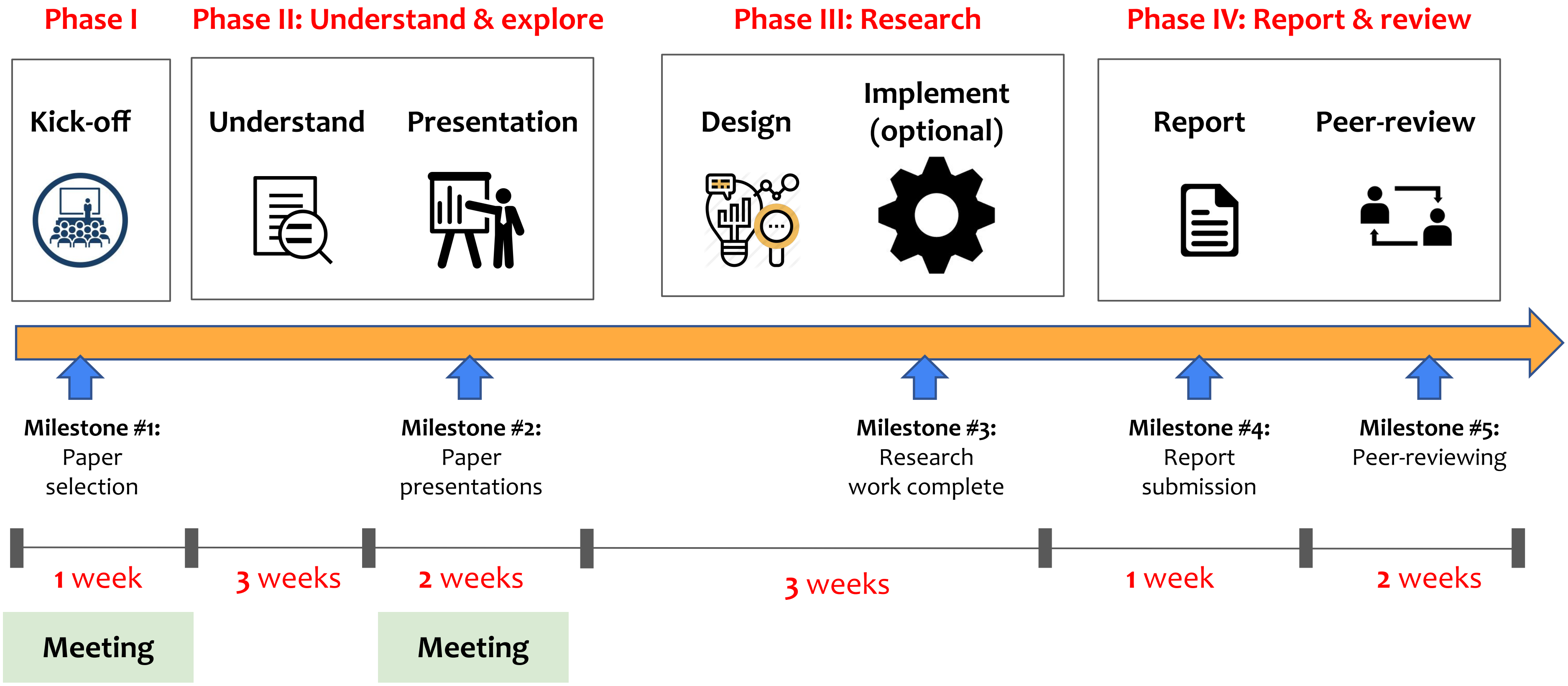
Give constructive (positive and critical) feedback for

(a) Paper summary

(b) Research work

END.

Overall timeline



Organization

- Format
  - Seminar course
- Communication
  - Slack for announcements and information sharing
  - Hotcrp for report submission and peer-reviewing
- Meetings (**in-person, attendance is compulsory**)
  - **Meeting #1:** Kick-off
  - **Meeting #2:** Paper presentation

Learning goals

- Learn about the cutting-edge research in quantum computing systems
- Promote critical thinking
- Cultivate an environment for innovation
  - To push the boundaries by advancing the state-of-the-art
- Improve scientific skills
  - Presentation
  - Writing
  - Communication: discussion and arguing
  - Mentorship: giving feedback and moderating discussion
- Encourage system building and evaluation
  - Learn by building, breaking, and benchmarking systems
- Importantly, to have fun!

Code of conduct

- University plagiarism policy
  - https://www.cit.tum.de/en/cit/studies/students/examination-matters-modules/informatics/practical-courses-seminar-courses/

- Decorum
  - Promote freedom of thoughts and open exchange of ideas
  - Cultivate dignity, understanding and mutual respect, and embrace diversity
  - Racism and bullying will not be tolerated

Contact

- Aleksandra Świerkowska
  - aleksandra.swierkowska@tum.de
- Francisco Romão
  - francisco.romao@tum.de
- Manos Giortamis
  - emmanouil.giortamis@in.tum.de
- Prof. Pramod Bhatotia
  - pramod.bhatotia@in.tum.de

- All seminar-related info: https://github.com/TUM-DSE/seminars

slack

Communication:
Join us with TUM email address (@tum.de)
ls1-courses-tum.slack.com
#ss-25-qc-systems