Quantum Software

Aleks Kissinger, Stefano Gogiosso

Oxford, Hillary Term 2024
Quantum software

1. := the code that runs on a quantum computer
   - factoring
   - search
   - physical simulation
   - optimisation problems
   - linear systems & codes
   - network flows
   - natural language processing
   ...

2. := the code that makes that code (better)
   - • compilers
   - • optimisation
   - • verification
Quantum software

1. := the code the runs on a quantum computer
Quantum software

1. := the code the runs on a quantum computer

- factoring
- search

- physical simulation
- optimisation problems
- linear systems & codes
- network flows
- natural language processing
- ...

Aleks Kissinger

Quantum Software

HT 2022
Quantum software

1. := the code the runs on a quantum computer

   - factoring
   - search

2. := the code that makes that code

   - physical simulation
   - optimisation problems
   - linear systems & codes
   - network flows
   - natural language processing
   ...
Quantum software

1. := the code the runs on a quantum computer
   - factoring
   - search
   - physical simulation
   - optimisation problems
   - linear systems & codes
   - network flows
   - natural language processing
   - ...

2. := the code that makes that code
   - compilers
Quantum software

1. := the code that runs on a quantum computer

   - factoring
   - search

2. := the code that makes that code
   - compilers
   - optimisation
   - verification
Quantum software

1. := the code the runs on a quantum computer
   
   **factoring**
   
   **search**

2. := the code that makes that code (better)
   
   - **compilers**
   - **optimisation**
   - **verification**

physical simulation
optimisation problems
linear systems & codes
network flows
natural language processing
...
Problem: no quantum computers
Problem: no quantum computers

(+ Oxford, Vienna, Delft, Sussex, Maryland...)}
Problem: no quantum computers

[Diagram showing various companies and providers in quantum computing]

1 https://quantumcomputingreport.com/review-of-the-cirq-quantum-software-framework
Problem: limited quantum computers

NISQ := “noisy intermediate-scale quantum”
**Problem:** limited quantum computers

NISQ := “noisy intermediate-scale quantum”

NISQ devices have:

- short coherence times
- low numbers of qubits
- noisy operations
- limited connectivity
- ...
**Problem:** limited quantum computers

NISQ := “noisy intermediate-scale quantum”

NISQ devices have:
- short coherence times
- low numbers of qubits
- noisy operations
- limited connectivity
- ...

⇒ **small** advances in software give **big** gains on NISQ hardware!
Quantum circuits

\[
\begin{array}{c}
\text{INIT} 5 \\
\text{CNOT} 1 0 \\
\text{H} 2 \\
\text{Z} 3 \\
\text{H} 0 \\
\text{H} 1 \\
\text{CNOT} 4 2 \\
\end{array}
\]

\[
\begin{array}{ccc}
\oplus & 0 & 0 \\
\text{H} & 0 & 0 \\
\oplus & \oplus & \oplus \\
\text{H} & \text{Z} & \text{X} \\
\oplus & \oplus & \oplus \\
\text{S} & \text{T} & \text{S} \\
\end{array}
\]
Quantum circuits

• := the ‘assembly language’ of quantum computation
Quantum circuits

:= the ‘assembly language’ of quantum computation, e.g.

\[
\begin{align*}
\text{INIT} & \ 5 \\
\text{CNOT} & \ 1 \ 0 \\
\text{H} & \ 2 \\
\text{Z} & \ 3 \\
\text{H} & \ 0 \\
\text{H} & \ 1 \\
\text{CNOT} & \ 4 \ 2 \\
\ldots
\end{align*}
\]
* Selinger 2015
* Amy, Chen, & Ross 2018
* Nam et al 2018
ZX-diagram
ZX-diagrams

...are like circuits, but made from spiders instead of gates: 

\[
\begin{align*}
X \alpha &= \alpha \\
Z \alpha &= \alpha 
\end{align*}
\]
ZX-diagrams

...are like circuits, but made from spiders instead of gates:

\[ Z_\alpha = \quad \alpha \]

\[ Z_\alpha = \quad \alpha \]
ZX-diagrams

...are like circuits, but made from **spiders** instead of gates:

\[ Z_\alpha = \begin{array}{c}
\vdots \\
\circ
\end{array} \]

\[ X_\alpha = \begin{array}{c}
\vdots \\
\circ
\end{array} \]
ZX-diagrams are bendy
ZX-diagrams are bendy
ZX-diagrams are bendy
Why spiders?

- They generate all linear maps $\mathbb{C}^{2^m} \rightarrow \mathbb{C}^{2^n}$.
Why spiders?

• They generate **all** linear maps $\mathbb{C}^{2^m} \rightarrow \mathbb{C}^{2^n}$.
• Handy for building most common gates, e.g.

\[
S = \begin{array}{c}
\pi \\
\hline
\end{array} \quad T = \begin{array}{c}
\pi/4 \\
\hline
\end{array} \\
H = \begin{array}{c}
\pi/2 \\
\hline
\end{array} \quad = \begin{array}{c}
\pi/2 \\
\hline
\end{array} = \begin{array}{c}
\pi/2 \\
\hline
\end{array}
\]

\[
\begin{array}{c}
\bullet \\
\hline
\end{array} \quad = \begin{array}{c}
\circ \\
\hline
\end{array} \quad = \begin{array}{c}
\circ \\
\hline
\end{array} = \begin{array}{c}
\bullet \\
\hline
\end{array}
\]

\[
\begin{array}{c}
\bullet \\
\hline
\end{array} = \begin{array}{c}
\circ \\
\hline
\end{array} = \begin{array}{c}
\circ \\
\hline
\end{array} = \begin{array}{c}
\bullet \\
\hline
\end{array}
\]
Why spiders?

- They generate **all** linear maps $\mathbb{C}^{2^m} \rightarrow \mathbb{C}^{2^n}$.
- Handy for building most common gates, e.g.

\[
\begin{align*}
S & = - \frac{\pi}{2} \\
T & = - \frac{\pi}{4} \\
H & = - \frac{\pi}{2} \quad \frac{\pi}{2} \quad \frac{\pi}{2} \\
\end{align*}
\]

\[
\begin{align*}
\text{square} & = \text{circle} \\
\text{circle} & = \text{diamond} \\
\text{diamond} & = \text{square}
\end{align*}
\]

- ....and....
ZX calculus
ZX calculus

these 8 rules \( \implies \) everything before
Q: How do we scale up?

A: Automation.
Q: How do we scale up?

~30 nodes

10^2-10^5 nodes

Automation.
Q: How do we scale up?

A: Automation.
In this course...

- We'll look at fundamental structures underlying quantum computations:
  - ZX-diagrams
  - Graph states
  - Stabiliser groups
  - Exponentiated Paulis
  - Phase polynomials
  - .....

- And apply them to...
In this course...

- We’ll look at **fundamental structures** underlying quantum computations:

  \[ ZX\text{-diagrams} \iff \{\text{graph states}, \text{stabiliser groups}, \text{exponentiated Paulis}, \text{phase polynomials}, \ldots\} \]
In this course...

• We’ll look at **fundamental structures** underlying quantum computations:

\[
\text{ZX-diagrams} \quad \leftrightarrow \quad \left\{ \begin{array}{c}
\text{graph states} \\
\text{stabiliser groups} \\
\text{exponentiated Paulis} \\
\text{phase polynomials} \\
\vdots
\end{array} \right\}
\]

• And apply them to...
Classical simulation

\[ \psi = C \]

\[ \text{Prob}(i \mid \psi) = ??? \]
Circuit optimisation

![Diagram of quantum circuit optimisation](image-url)
Circuit routing

\[
\begin{array}{c}
\begin{array}{ccc}
4 & 3 & 1 \\
2 & 15 & 13 \\
5 & 6 & 17 \\
0 & 10 & 12 \\
\end{array}
\end{array}
\Rightarrow
\begin{array}{c}
\begin{array}{ccc}
0 & 1 & 2 \\
5 & 4 & 3 \\
6 & 7 & 8 \\
\end{array}
\end{array}
\]

\[
\begin{array}{ccccc}
\oplus & \oplus & \oplus & \oplus & \oplus \\
0 & 4 & 1 & 2 & 6 & 7 & 8 & 5 & 3 \\
\end{array}
\]

\[
t_1 \quad t_2 \quad t_3 \quad t_4 \quad t_5
\]

\[
\begin{array}{c}
\begin{array}{c}
0 \\
5 \\
6 \\
\end{array}
\end{array}
\Rightarrow
\begin{array}{c}
\begin{array}{c}
0 \\
1 \\
2 \\
3 \\
4 \\
5 \\
6 \\
7 \\
8 \\
\end{array}
\end{array}
\]

Aleks Kissinger

Quantum Software

HT 2022
(NISQ) quantum algorithms

https://www.nature.com/articles/s42254-021-00348-9
Implementation

ibmq_casablanca

Details

7

7
Quibits

Total pending jobs: 23 jobs

Processor type (Q): Falcon r4H

Version: 1.2.49

Basis gates: CX, ID, RZ, SX, X

Your usage: 0 jobs

Avg. CNOT Error: 1.057e-2

Avg. T1: 92.35 us

Avg. T2: 120.42 us

Providers with access: 2 Providers

Supports Qiskit Runtime: Yes

Your upcoming reservations: 0

New reservation +

Calibration data

Last calibrated: 11 minutes ago

Map view

Graph view

Table view

Qubit:
Frequency (GHz)

Avg: 4.911

min: 4.76

max: 5.177

Connection:
CNOT error

Avg: 1.057e-2

min: 5.012e-3

max: 1.965e-2


Aleks Kissinger

Quantum Software

HT 2022

21 / 22
Format of the course

8 weeks, 24 lectures, 6 problem sheets + classes (week 3, 4, 5, 6, 7, 8)

5 weeks theory (Aleks)
- basics, ZX, classical simulation, quantum compilation, quantum error correction

3 weeks live coding (Stefano)
- algorithms, implementation on IBMQ

Materials:
- handwritten lecture notes, matching Aleks' lectures
- textbook: preprint of Picturing Quantum Software. Kissinger & van de Wetering
- Jupyter notebooks from live coding
- Picturing Quantum Processes. Coecke & Kissinger. CUP (optional)
- Quantum Computation and Quantum Information. Nielsen & Chuang. CUP (optional)

Exam is by take-home miniproject
- expect a theory and a (Python) coding component
Format of the course

- 8 weeks, 24 lectures, 6 problem sheets + classes (week 3, 4, 5, 6, 7, 8)
- 5 weeks theory (Aleks)
  - basics, ZX, classical simulation, quantum compilation, quantum error correction
- 3 weeks live coding (Stefano)
  - algorithms, implementation on IBMQ

Materials:
- handwritten lecture notes, matching Aleks' lectures
- textbook: preprint of Picturing Quantum Software. Kissinger & van de Wetering
- Jupyter notebooks from live coding
- Picturing Quantum Processes. Coecke & Kissinger. CUP (optional)
- Quantum Computation and Quantum Information. Nielsen & Chuang. CUP (optional)

Exam is by take-home miniproject
- expect a theory and a (Python) coding component
Format of the course

- 8 weeks, 24 lectures, 6 problem sheets + classes (week 3, 4, 5, 6, 7, 8)
- 5 weeks theory (Aleks)
  - basics, ZX, classical simulation, quantum compilation, quantum error correction
- 3 weeks live coding (Stefano)
  - algorithms, implementation on IBMQ

Materials:
- handwritten lecture notes, matching Aleks' lectures
- textbook: preprint of Picturing Quantum Software. Kissinger & van de Wetering
- Jupyter notebooks from live coding
- Picturing Quantum Processes. Coecke & Kissinger. CUP (optional)
- Quantum Computation and Quantum Information. Nielsen & Chuang. CUP (optional)

Exam is by take-home miniproject
- expect a theory and a (Python) coding component
Format of the course

• 8 weeks, 24 lectures, 6 problem sheets + classes (week 3, 4, 5, 6, 7, 8)
• 5 weeks theory (Aleks)
  • basics, ZX, classical simulation, quantum compilation, quantum error correction
• 3 weeks live coding (Stefano)
  • algorithms, implementation on IBMQ

Materials:
• handwritten lecture notes, matching Aleks' lectures
• textbook: preprint of Picturing Quantum Software. Kissinger & van de Wetering
• Jupyter notebooks from live coding
• Picturing Quantum Processes. Coecke & Kissinger. CUP (optional)
• Quantum Computation and Quantum Information. Nielsen & Chuang. CUP (optional)

Exam is by take-home miniproject
• expect a theory and a (Python) coding component
Format of the course

• 8 weeks, 24 lectures, 6 problem sheets + classes (week 3, 4, 5, 6, 7, 8)
• 5 weeks theory (Aleks)
  • basics, ZX, classical simulation, quantum compilation, quantum error correction
• 3 weeks live coding (Stefano)
  • algorithms, implementation on IBMQ
• Materials:
  • handwritten lecture notes, matching Aleks’ lectures
  • textbook: preprint of Picturing Quantum Software. Kissinger & van de Wetering
  • Jupyter notebooks from live coding

Exam is by take-home miniproject

expect a theory and a (Python) coding component
Format of the course

- 8 weeks, 24 lectures, 6 problem sheets + classes (week 3, 4, 5, 6, 7, 8)
- 5 weeks theory (Aleks)
  - basics, ZX, classical simulation, quantum compilation, quantum error correction
- 3 weeks live coding (Stefano)
  - algorithms, implementation on IBMQ
- Materials:
  - handwritten lecture notes, matching Aleks’ lectures
  - textbook: preprint of Picturing Quantum Software. Kissinger & van de Wetering
  - Jupyter notebooks from live coding
  - Picturing Quantum Processes. Coecke & Kissinger. CUP (optional)
  - Quantum Computation and Quantum Information. Nielsen & Chuang. CUP (optional)
- Exam is by take-home miniproject
- expect a theory and a (Python) coding component
Format of the course

• 8 weeks, 24 lectures, 6 problem sheets + classes (week 3, 4, 5, 6, 7, 8)
• 5 weeks theory (Aleks)
  • basics, ZX, classical simulation, quantum compilation, quantum error correction
• 3 weeks live coding (Stefano)
  • algorithms, implementation on IBMQ
• Materials:
  • handwritten lecture notes, matching Aleks’ lectures
  • textbook: preprint of Picturing Quantum Software. Kissinger & van de Wetering
  • Jupyter notebooks from live coding
  • Picturing Quantum Processes. Coecke & Kissinger. CUP (optional)
  • Quantum Computation and Quantum Information. Nielsen & Chuang. CUP (optional)
• Exam is by take-home miniproject
  • expect a theory and a (Python) coding component