Resources for quantum computation

Oliver Reardon-Smith

Uniwersytet Jagielloński

oliver.reardon-smith@uj.edu.pl

July 6, 2020

Background

Bell 1964

Can a quantum system be probabilistically simulated by a classical (probabilistic, I assume) universal computer?...The answer is certainly, 'No!'...It is impossible to represent the result of quantum mechanics with a classical universal device.

Feynman 1981

Nature isn't classical, dammit, and if you want to make a simulation of Nature, you'd better make it quantum mechanical.

Classically simulable subtheories

Clifford gates + stabiliser states

Gottesman (1998)

Aaronson and Gottesman (2004)

Bravyi, Browne, Calpin, Campbell, Gosset and Howard (2019)

Classically simulable subtheories

Clifford gates + stabiliser states

Gottesman (1998)
Aaronson and Gottesman (2004)
Bravyi, Browne, Calpin, Campbell, Gosset and Howard (2019)

Match gates + product states

Valiant (2001)
Knill (2001)
Terhal and DiVincenzo (2002)
Jozsa and Miyake (2008)

Classically simulable subtheories

Clifford gates + stabiliser states

Gottesman (1998)

Aaronson and Gottesman (2004)

Bravyi, Browne, Calpin, Campbell, Gosset and Howard (2019)

Match gates + product states

Valiant (2001)
Knill (2001)
Terhal and DiVincenzo (2002)

Jozsa and Miyake (2008)

Spekkens' toy model

Spekkens (2004)

Catani and Browne (2018)

Why do we care about these things?

Foundations

Understanding classically simulable subtheories gives insight into what “special sauce” makes quantum computation different

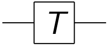
Applications

These subtheories can be used to design “efficient” classical simulations of quantum computers

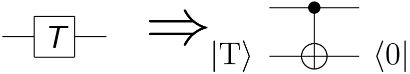
Turning simulable subtheories into universal simulators

1. Find some “resourceful” operations that promote your subtheory to quantum universality
2. “Gadgetize” these operations
3. Write gadgetized state as a (typically big!) combination of subtheory states
4. Sample enough elements of the big combination to approximate the true sum

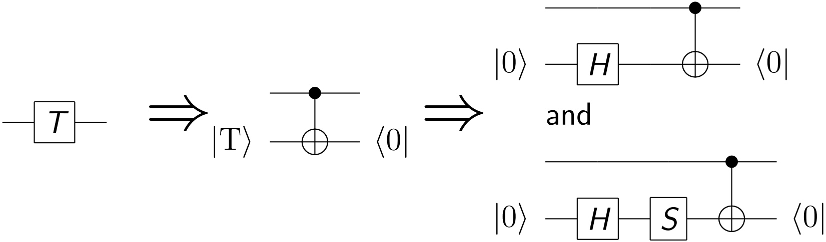
Example



Example



Example



CH-form

What is it?

Phase-sensitive representation of a stabiliser state

$$|\xi\rangle = \omega U_C U_H |s\rangle \tag{1}$$

where

- ω is a complex number
- $U_C|00\ldots0\rangle = |00\ldots0\rangle$
- U_H is a tensor product of identity and Hadamard matrices
- $|s\rangle$ is a computational basis state

What have I been doing (1)

Factorizing CH-form states

Given a CH-form representing the state $|0\rangle \otimes |\xi\rangle$ can we find a CH-form for the state $|\xi\rangle$?

What have I been doing (1)

Factorizing CH-form states

Given a CH-form representing the state $|0\rangle \otimes |\xi\rangle$ can we find a CH-form for the state $|\xi\rangle$?

Yes - in $O(n^2)$ time!

What have I been doing (2)

Precomputation

Do we have to recompute the whole circuit for every sample in the big linear combination?

What have I been doing (2)

Precomputation

Do we have to recompute the whole circuit for every sample in the big linear combination?

No - the work we have to do each sample is independent of the length of the circuit!

What have I been doing (2)

Precomputation

Do we have to recompute the whole circuit for every sample in the big linear combination?

No - the work we have to do each sample is independent of the length of the circuit!

Turn $O(cn^2 2^t)$ into $O(cn^2 + n^2 2^t)$

Future directions

More work on the CH-form

- Complexity improvements in our algorithms
- Some implementation details
- Different resources

Broader questions

- Other simulable subtheories
- Equivalence of the subtheories we have

Applications

- Finish the (open source) python implementation
- Verification&validation of NISQ machines
- Parallelisation!

Please ask a lot of questions

Please ~~ask a lot of~~ axolotl questions