Highly Efficient Bell State Purification Through Genetic Algorithms

…

Stefan Krastanov, Liang Jiang, and many others at

Yale Quantum Institute

The Problem:
Designing an Efficient (Quantum) Circuit

A Solution: Discrete Optimization

Outline: Quantum Circuits for Bell State Purification

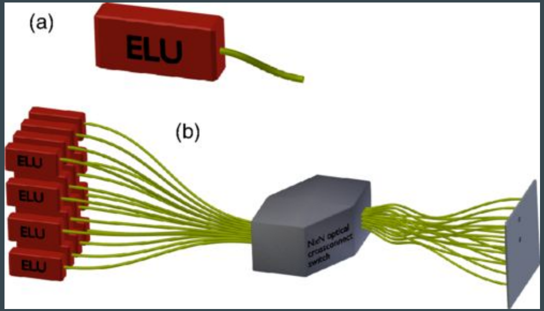
- Why do we need Bell states?
  - A primer on modular architecture for quantum computers
- Purification
- Optimized Purification
- Some Applications (if you ask)
  - GHZ states
  - Stabilizer measurements

Why Do We Need Bell States?

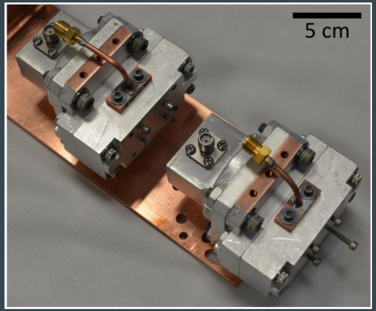
Quantum Teleportation

Quantum Key Distribution

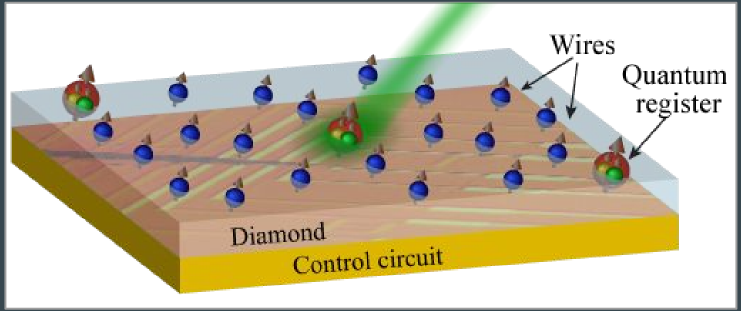
Modular architecture for quantum computers
(Trapped Ions, Superconducting Circuits, NV centers, etc.)



Monroe, PRA 89, 022317 (2014)



Shankar, Narla, YQI, Yale



Prof. Cappellaro, MIT

Purification

$$|\Phi+\rangle = |00\rangle + |11\rangle$$
$$|\Psi-\rangle = |01\rangle - |10\rangle$$
$$|\Psi+\rangle = |01\rangle + |10\rangle$$
$$|\Phi-\rangle = |00\rangle - |11\rangle$$

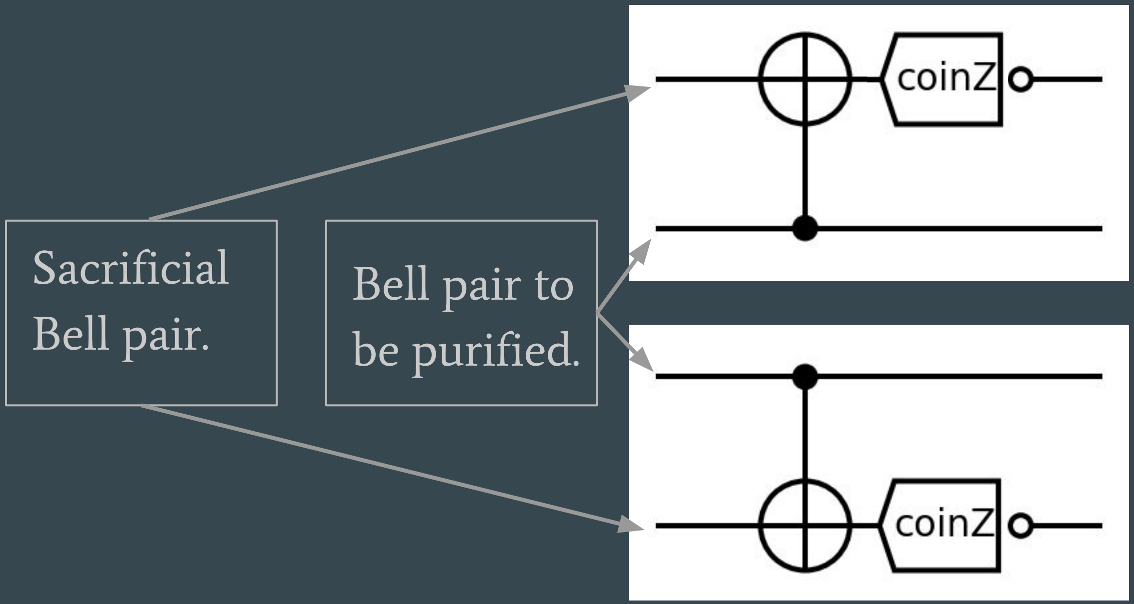
The four possible Bell states are Φ+, Ψ-, Ψ+, and Φ-:

Φ+ at 90%

Ψ-, Ψ+, and Φ- at 3.3%

Start with two Bell pairs and obtain one higher quality pair.

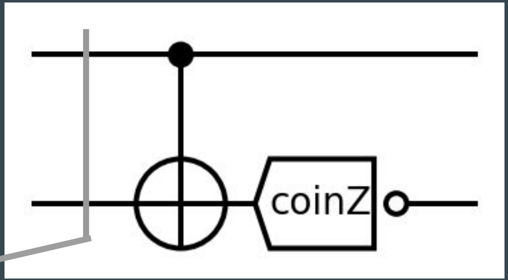
Purification



Alice's side of the circuit.

Bob's side of the circuit. Alice does the same to her "half pairs".

Purification

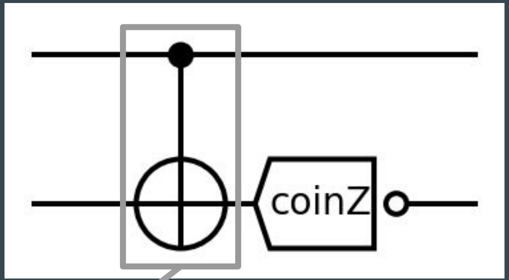


Bob's side of the circuit. Alice does the same to her "half pairs".

Φ+Φ+	81%
Φ+Ψ-	3%
Φ+Ψ+	3%
Φ+Φ-	3%
others	10%

Tracing over the second pair returns the same quality pair as the one we started with.

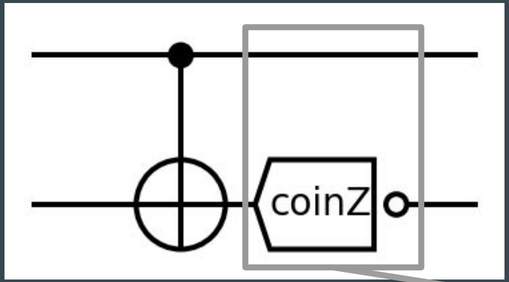
Purification



Φ+Φ+ remains Φ+Φ+
Φ-Φ- becomes Φ+Φ-
other permutations happen

Bob’s side of the circuit. Alice does the same to her “half pairs”.

Purification

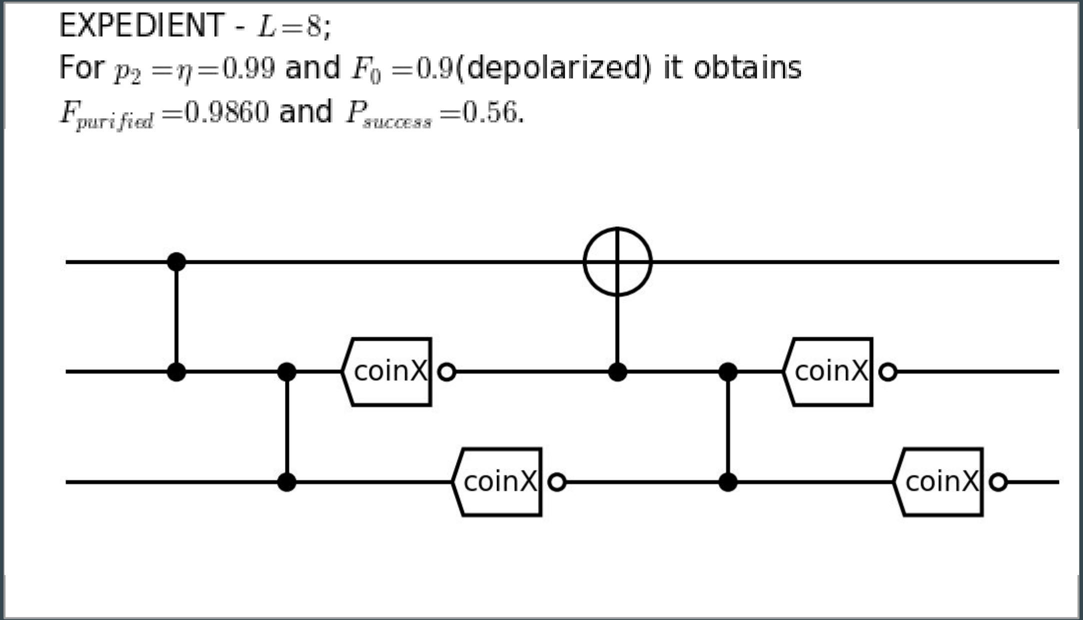


Bob's side of the circuit.
Alice does the same to her "half pairs".

Trace over the second pair, selecting only Φ+ or Φ-.

Result:
93% Φ+ in the first pair.

More Advanced Purification Example



The EXPEDIENT purification circuit, Nickerson, Nature Comm. 4, 1756 (2014)

Combinatorial Explosion of Possible Circuits

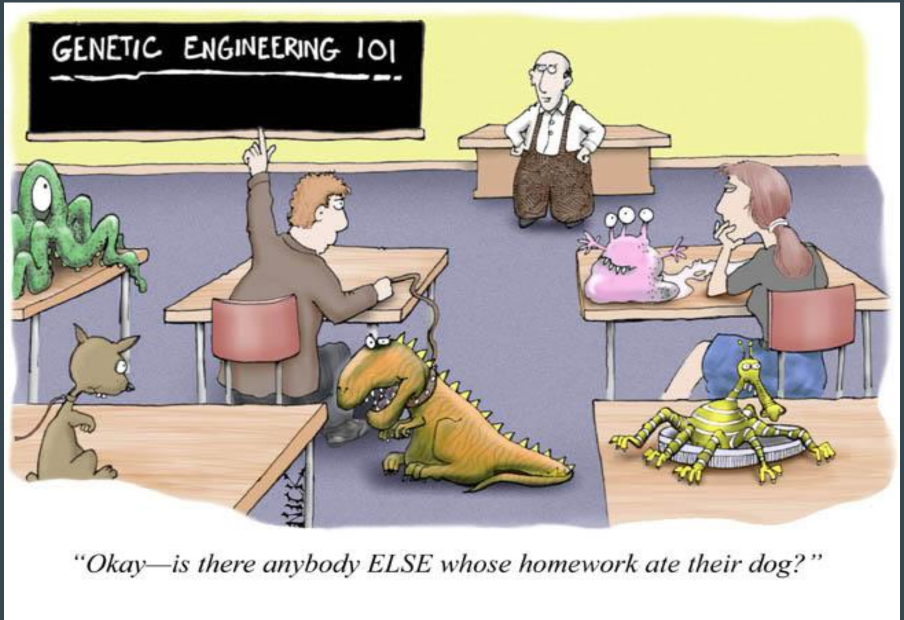
Circuits of length L (number of operations)
and depth N (number of resettable Bell pairs):

$$\text{possible circuits} \approx \left(\text{gates' fraction} \times 2\binom{N}{2} + \text{measurements' fraction} \times 3N\right)^L$$

Optimized Purification (Genetic Algorithms)

Represent the circuit as a “DNA string” where each gene is a gate/measurement.

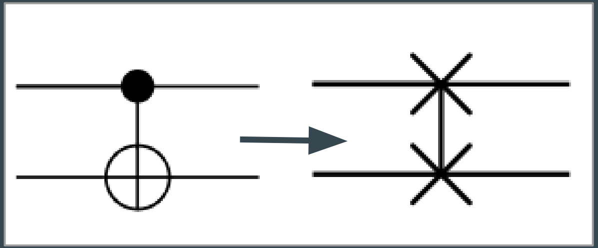
Get many random circuits and subject them to merciless simulated evolution.



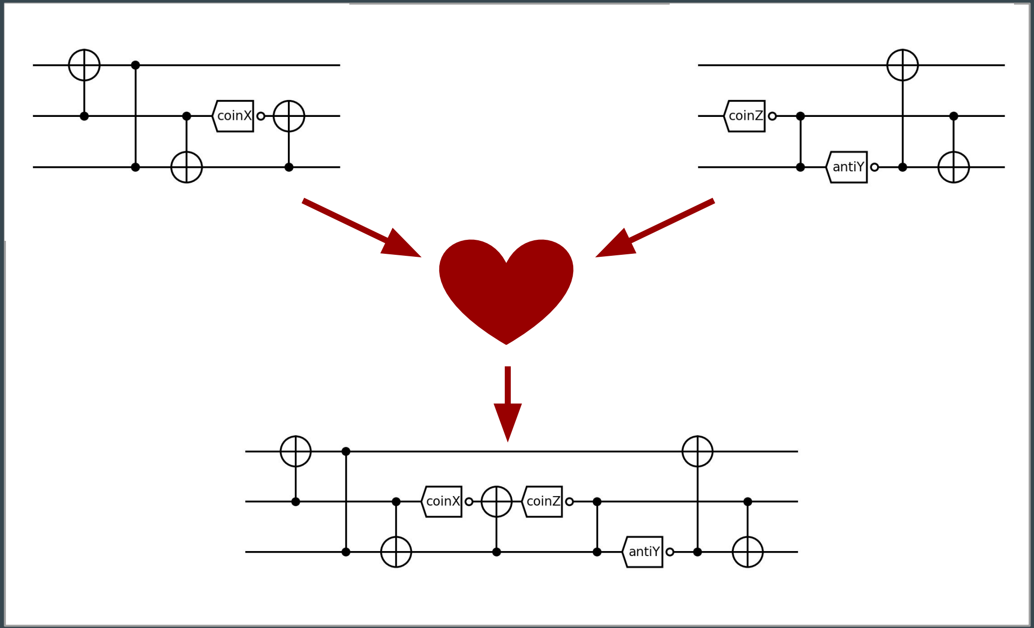
nearingzero.net

Two "Modes" of Evolution

Mutations



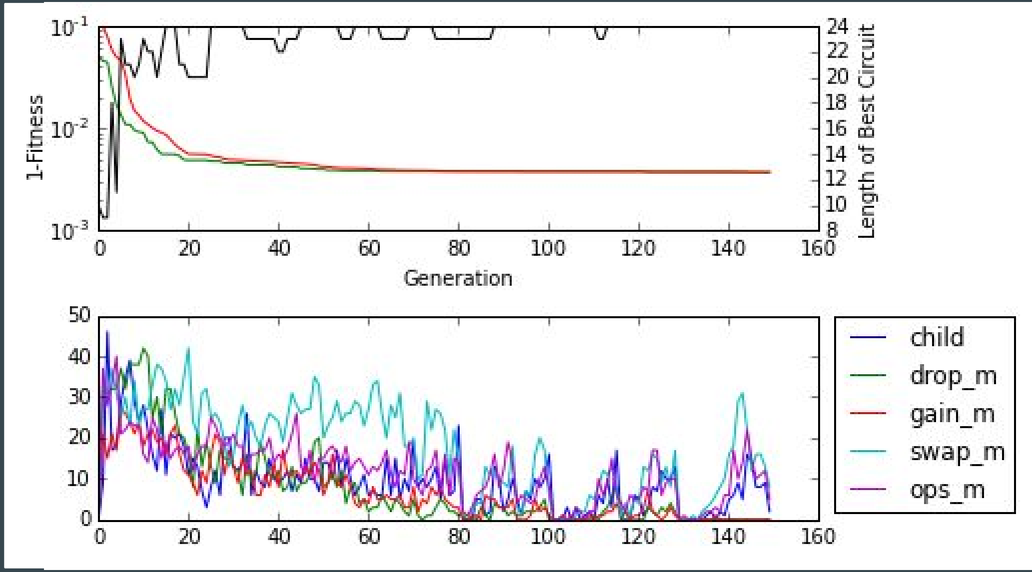
Sexual reproduction



A Hundred Generations Later (5min laptop time)

For a given error model and specified error parameters quickly find near-optimal purification circuits.

Various optimizations related to the structure of the problem were implemented.



Real-time trace of the optimization process.

(it is pretty, this is all about it)

Better Than Prior Art

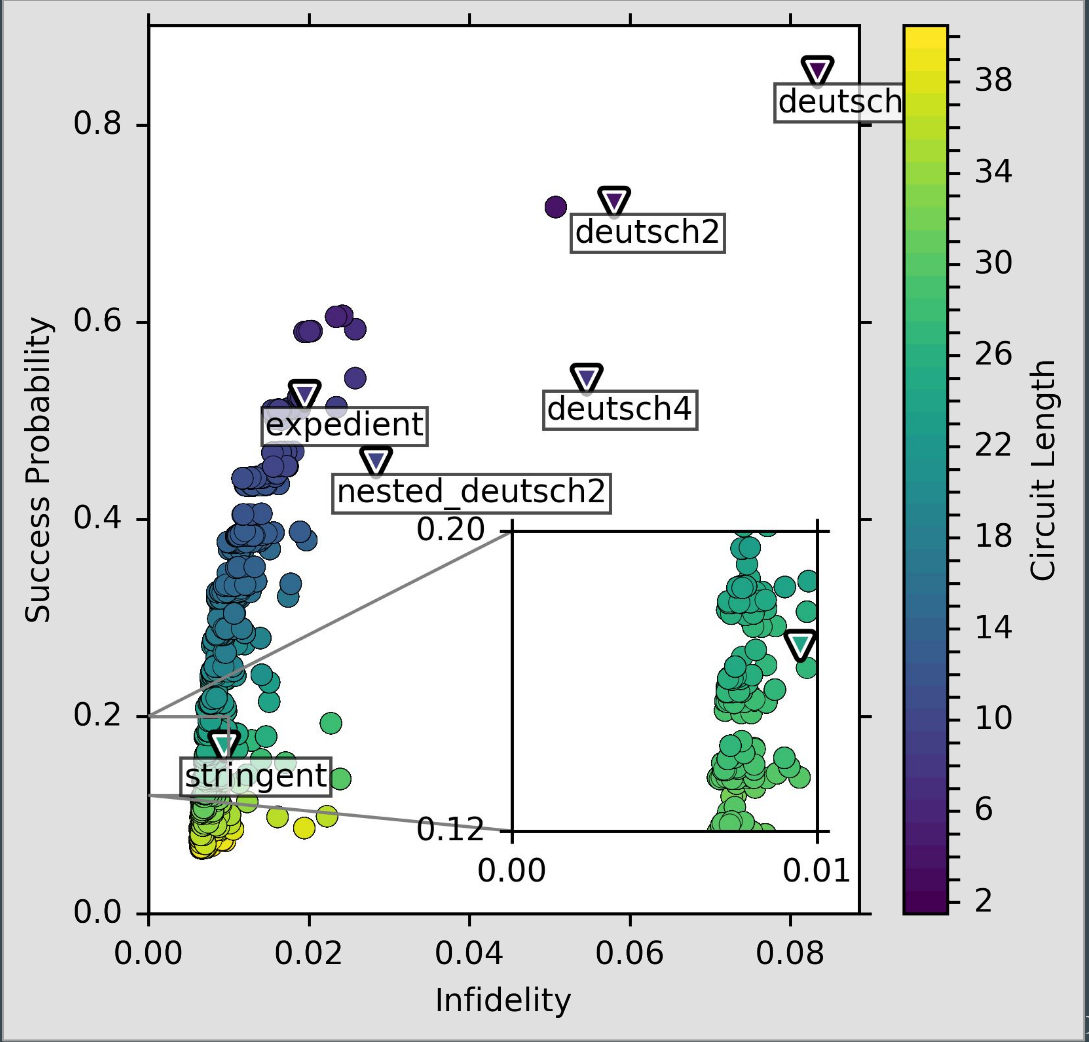
Simulated for

- Initial Bell Pair Fidelity 0.9
- Local Operations Fidelity 0.99
- Depolarization Error Model

Available at

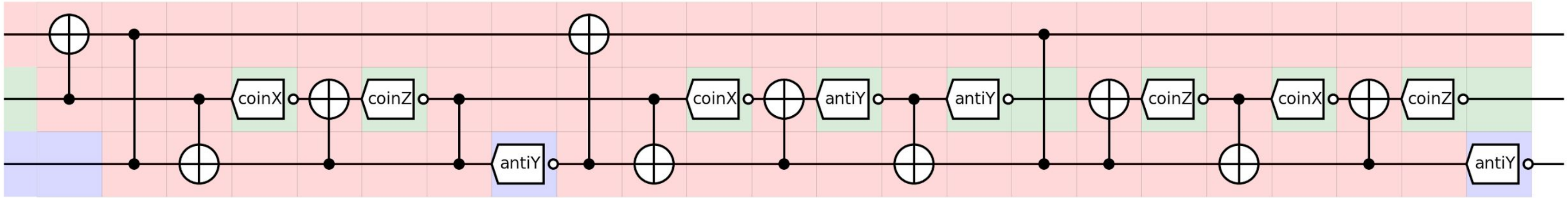
qevo.krastanov.org

Circuits From:
- Nickerson, Nature Comm. 4, 1756 (2014)
- Deutsch, PRL 77, 2818 (1996)
- Dür, PRA, 59, 169 (1999)

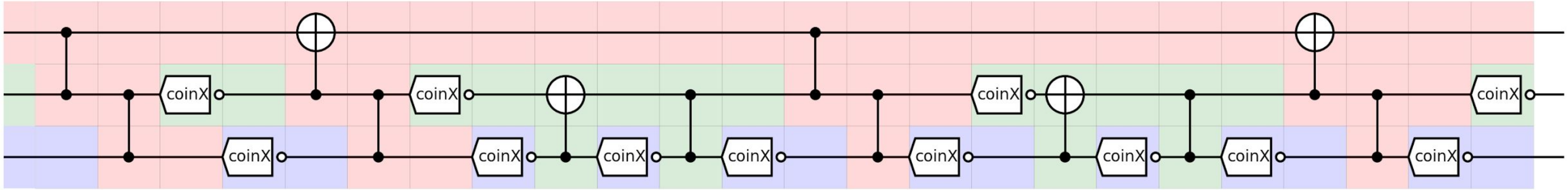


A Particular Example

L23



STRINGENT



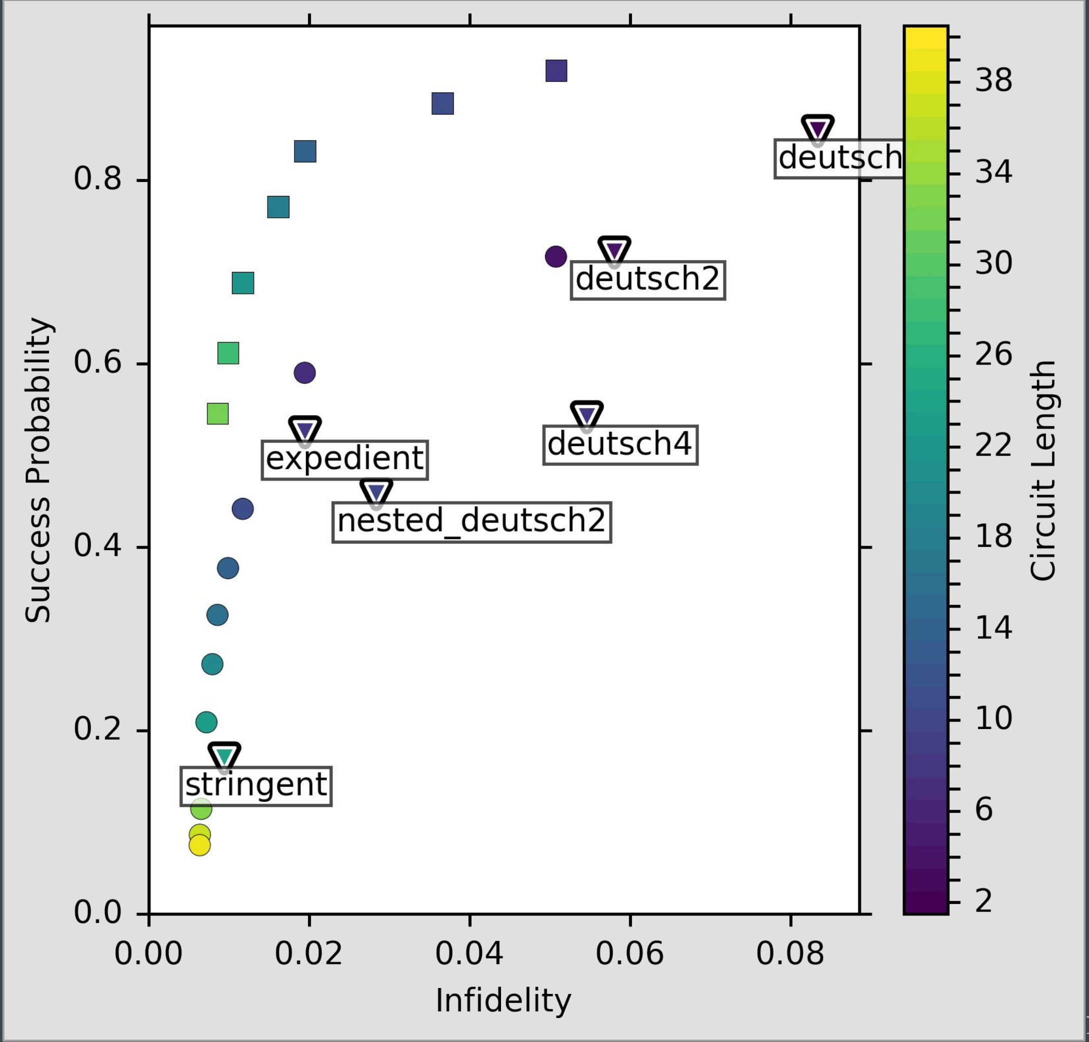
L23 is one of the many circuits we designed. STRINGENT is from Nickerson, Nature Comm. 4, 1756 (2014)

With Naive Feed-Forward

Chain two circuits together.

If the first succeeds, accept.

If the first fails, try the second circuit.



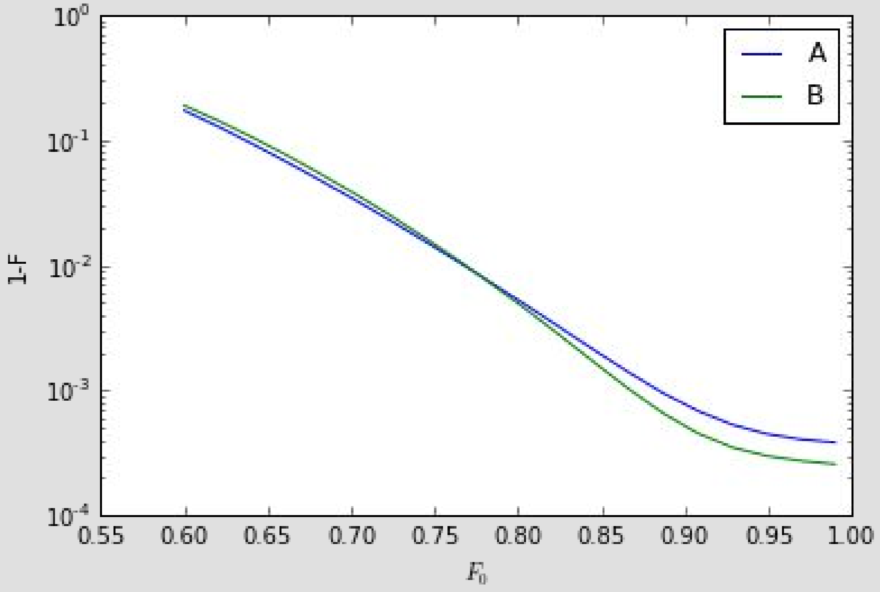
Conclusions (Thanks for listening)

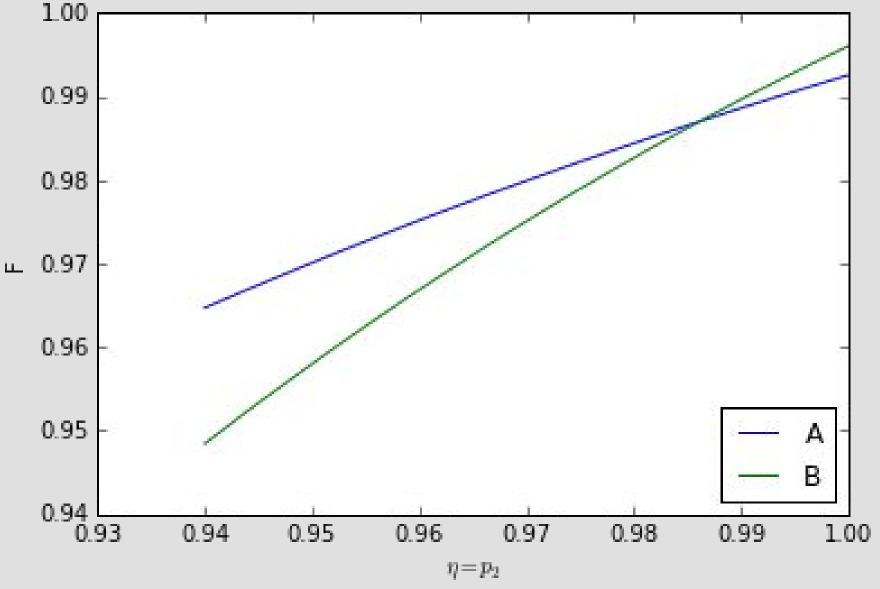
- Optimizing a “building-block” protocol to work for particular hardware is now easy and automated
- ~20% gain in fidelity for same-length (or even shorter) circuits
- A new, much bigger than previously available collection of circuits to try
  - Available at qevo.krastanov.org
- Applications in quantum communication and fault-tolerant computing
- Caveats
  - Do not over rely on magic! Prune and inspect your circuits!
  - Set cost function properly!
  - Having a canonical form can be important!

Different Circuits Work Better for Different Parameters

Different Network Efficiency

Different Local Gates' Quality





A and B are circuits obtained after optimizing for different hardware parameters.