Quantum Cryptanalysis in the RAM Model: Claw-Finding Attacks on SIKE

Samuel Jaques and John M. Schanck

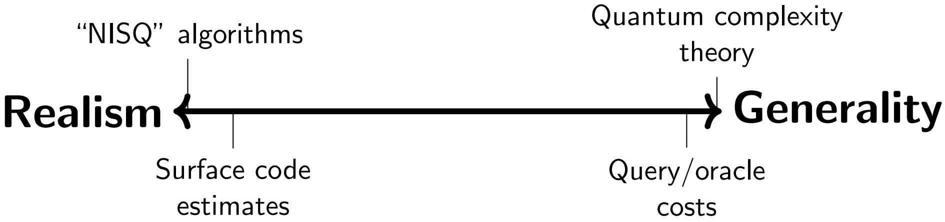
UNIVERSITY OF WATERLOO

Models of quantum computers

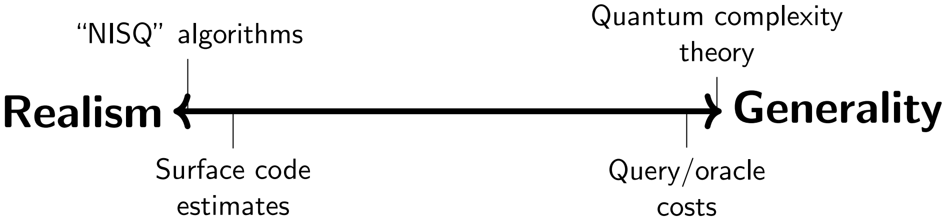
- NIST is working on post-quantum public key standards
- This requires **quantum cryptanalysis**
- This requires models of quantum computers

How do you imagine a quantum computer?

Quantum cost analysis

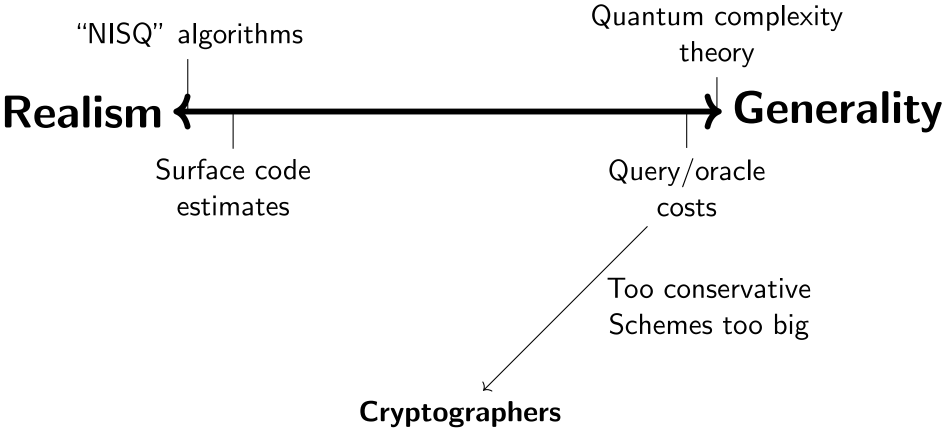


Quantum cost analysis

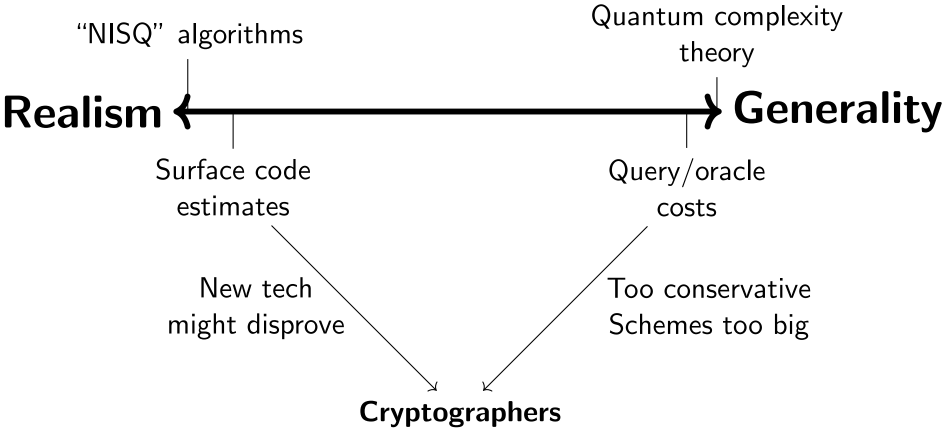


Cryptographers

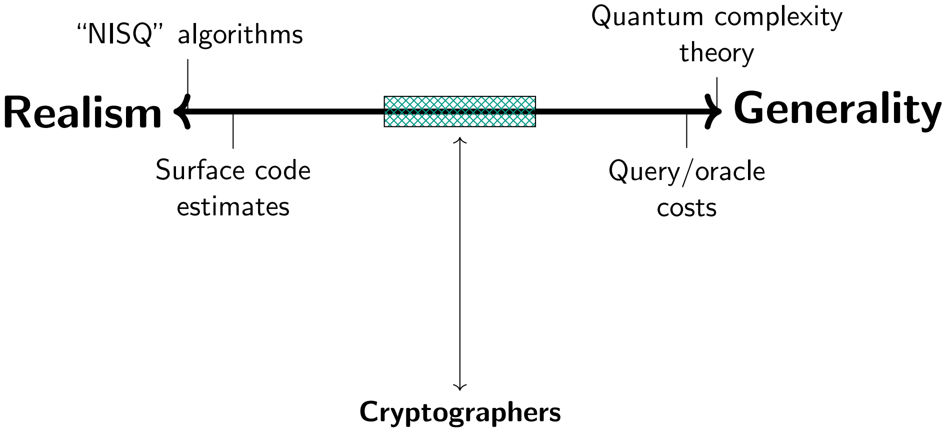
Quantum cost analysis



Quantum cost analysis



Quantum cost analysis



Outline

Goal 1: Fairly compare classical and quantum resources

How do we compare a quantum bit of security to a classical bit of security?

How do we cost mixed classical/quantum algorithms like Kuperberg's sieve?

Goal 1: Fairly compare classical and quantum resources

How do we compare a quantum bit of security to a classical bit of security?
How do we cost mixed classical/quantum algorithms like Kuperberg's sieve?

Previous work: Analysis of Brassard-Høyer-Tapp (BHT)

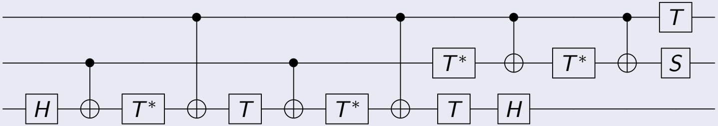
- BHT provided a quantum collision-finding algorithm with quantum access to classical memory.
- Bernstein argued van Oorschot-Wiener is more efficient after fully accounting for memory costs.

Brassard, Høyer, Tapp. 1997. Quantum Algorithm for the Collision Problem

Bernstein. 2009. Cost analysis of hash collisions: Will quantum computers make SHARCS obsolete?

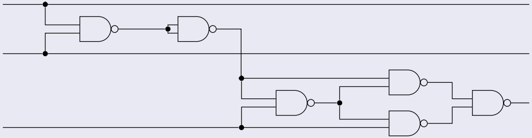
Goal 2: View gates as processes

Quantum



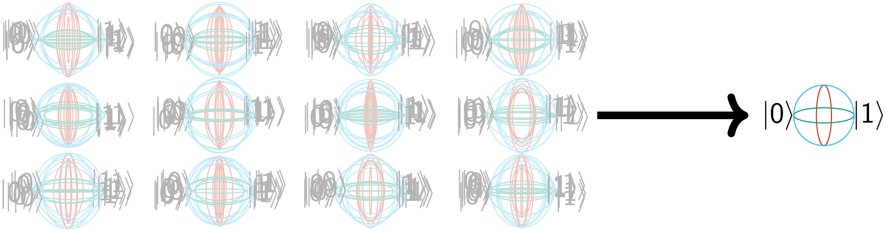
Time→

Classical



Time/Space→

Goal 3: Include error correction



Many physical qubits One logical qubit

Memory peripheral framework

Main Idea

Model computation as "memory" acted on by a "memory controller".

Examples:

- Turing machine: head + tape
- RAM: CPU + random access memory
- Quantum circuit: Random access machine + qubits

Premises:

1. **Memory** is a physical system that changes over time
2. A **memory controller** interacts with a memory
3. The **cost** of a computation is the number of interactions

Premise 1: Memory is a physical system

Free evolution

Caused by:

- Noise
- Ballistic computation

Costly evolution

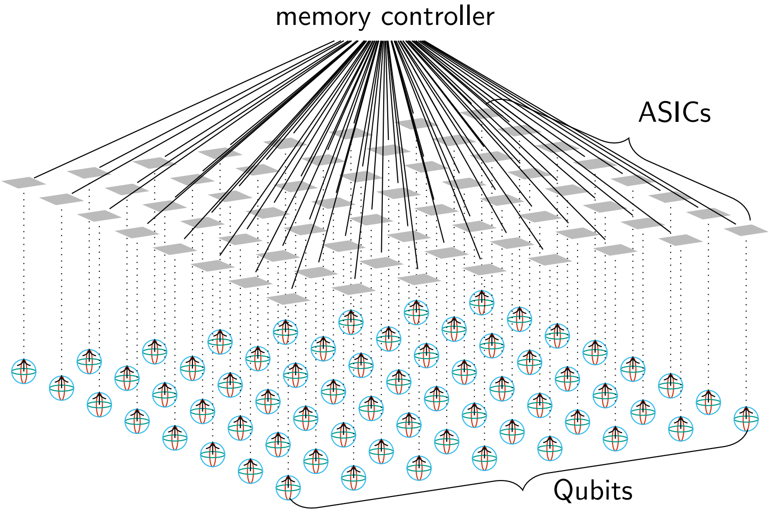
Caused by the controller.

We model a quantum computer as a **parallel random access machine** with new instructions for quantum gates

- e.g.: apply gate x to qubit y at time t

Result: quantum algorithms are classical programs

Premise 2: Memory controller



Premise 3: Cost

The **cost** of a computation is the number of interactions.

- We ignore the construction cost
- We focus on the cost to the controller

There are opportunity costs: What else could the controller do?

Cost models

We provide physical justifications for two cost models: **G-cost** and **DW-cost**.

Both are qubit memories with a standard universal gate set (Clifford + T).

Differences:

- *G*-cost: **Passive** error correction.
- *DW*-cost: **Active** error correction.

Error correction

Passive/Non-volatile memory

To preserve: keep cool.

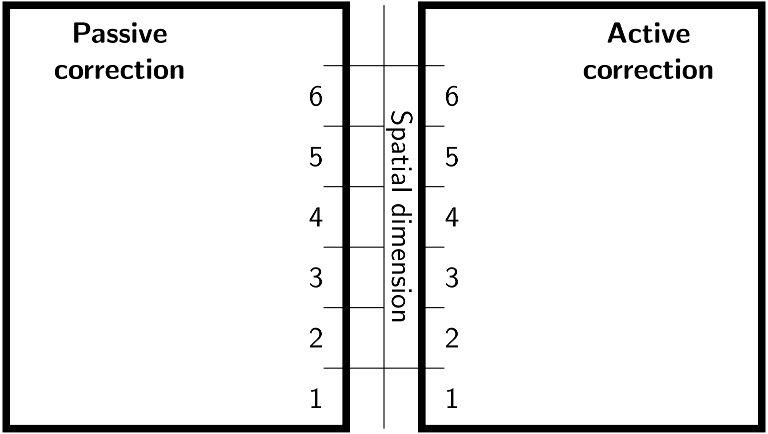
- Paper
- Magnetic discs

Active/Volatile memory

To preserve: continuously refresh.

- DRAM
- Surface codes (quantum)

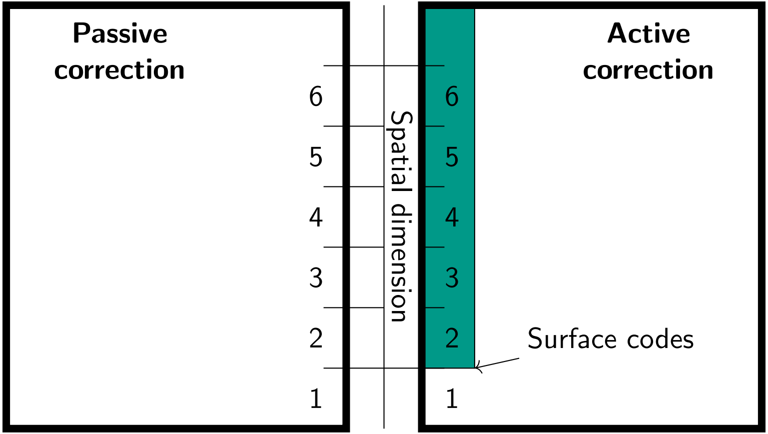
Quantum error correction theory



[1] Bravyi and Terhal. 2009. A no-go theorem for a two-dimensional self-correcting quantum memory based on stabilizer codes.
[2] Kitaev. 2003. Fault-tolerant quantum computation by anyons.

Dennis, Kitaev, Landahl, Preskill. 2002. Topological Quantum Memory.

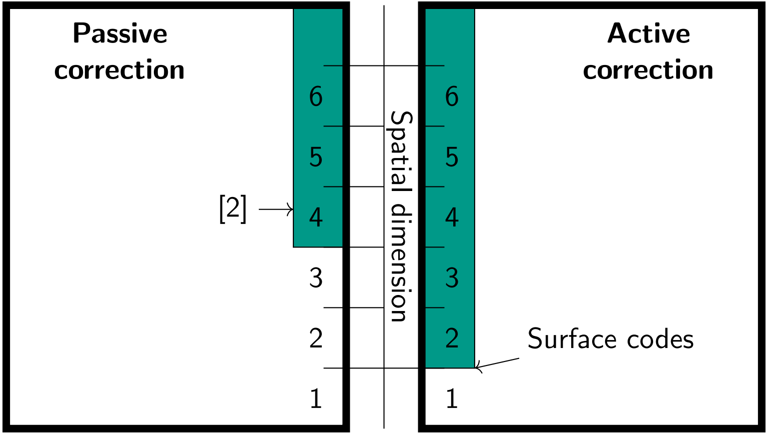
Quantum error correction theory



[1] Bravyi and Terhal. 2009. A no-go theorem for a two-dimensional self-correcting quantum memory based on stabilizer codes.
[2] Kitaev. 2003. Fault-tolerant quantum computation by anyons.

Dennis, Kitaev, Landahl, Preskill. 2002. Topological Quantum Memory.

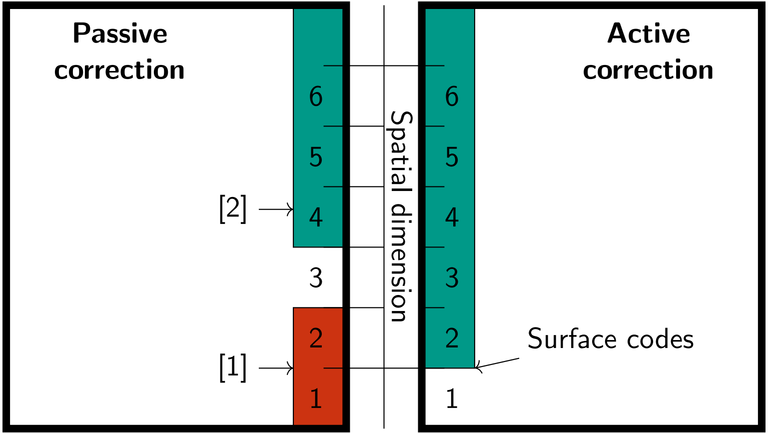
Quantum error correction theory



[1] Bravyi and Terhal. 2009. A no-go theorem for a two-dimensional self-correcting quantum memory based on stabilizer codes.
[2] Kitaev. 2003. Fault-tolerant quantum computation by anyons.

Dennis, Kitaev, Landahl, Preskill. 2002. Topological Quantum Memory.

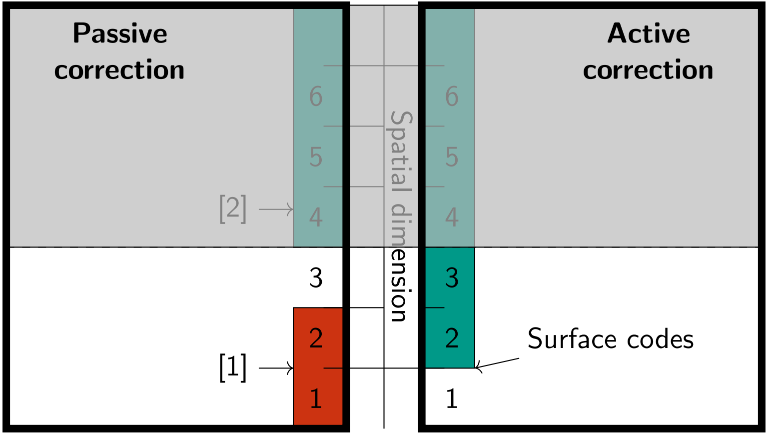
Quantum error correction theory



[1] Bravyi and Terhal. 2009. A no-go theorem for a two-dimensional self-correcting quantum memory based on stabilizer codes.
[2] Kitaev. 2003. Fault-tolerant quantum computation by anyons.

Dennis, Kitaev, Landahl, Preskill. 2002. Topological Quantum Memory.

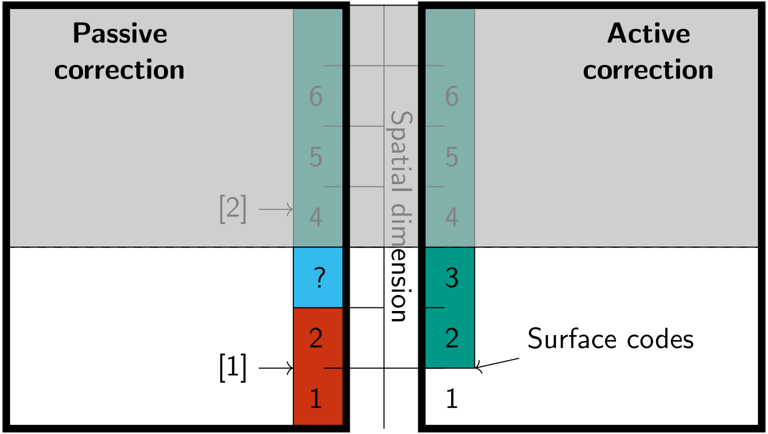
Quantum error correction theory



[1] Bravyi and Terhal. 2009. A no-go theorem for a two-dimensional self-correcting quantum memory based on stabilizer codes.
[2] Kitaev. 2003. Fault-tolerant quantum computation by anyons.

Dennis, Kitaev, Landahl, Preskill. 2002. Topological Quantum Memory.

Quantum error correction theory



[1] Bravyi and Terhal. 2009. A no-go theorem for a two-dimensional self-correcting quantum memory based on stabilizer codes.
[2] Kitaev. 2003. Fault-tolerant quantum computation by anyons.

Dennis, Kitaev, Landahl, Preskill. 2002. Topological Quantum Memory.

Costs

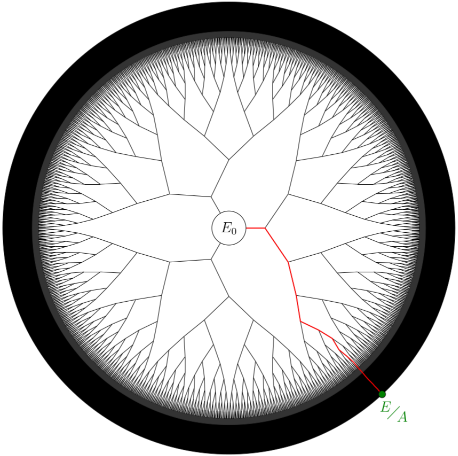
G-cost

- Assumption: **Passive** error correction.
  (Physical, not just technological, assumption)
- Cost: 1 RAM operation per gate
- Total cost: Number of gates (“G”)

DW-cost

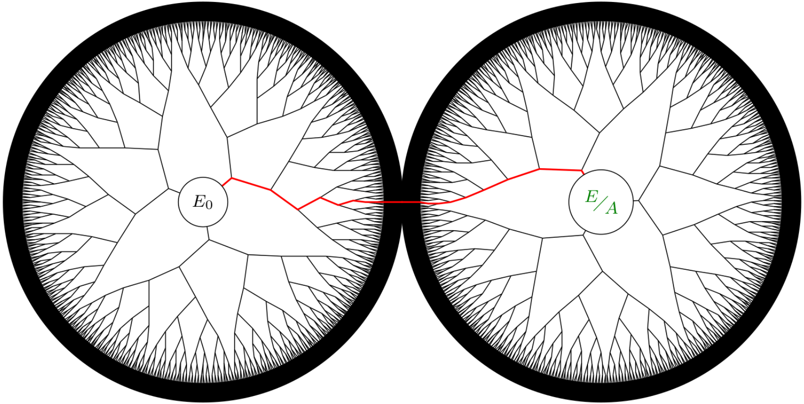
- Assumption: **Active** error correction.
- Cost: 1 RAM operation per qubit per time step
- Total cost: Depth$\times$Width (“DW”)

Analysis of SIKE



- E_0 is public parameter, E/A is public key
- Parameterized by a large prime p (e.g. $p \approx 2^{434}$)
- Red path is secret key (length $\log p/2$)

Meet-in-the-middle



Tani's collision-finding algorithm

To find a collision between two functions $f : X \to S$ and $g : Y \to S$:

- Random walk on two Johnson graphs: one over X, the other over Y
- Check for collisions at each step
- Make it quantum!

Johnson graph over X

Vertices: R-element subsets of a fixed set X.
Vertices u and v are adjacent iff $|u \cap v| = R - 1$.

Tani. 2007. An improved claw finding algorithm using quantum walk.

Tani's collision-finding algorithm

To find a collision between two functions $f : X \to S$ and $g : Y \to S$:

- Random walk on two Johnson graphs: one over X, the other over Y
- Check for collisions at each step
- Make it quantum!

Johnson graph over X

Vertices: R-element subsets of a fixed set X.
Vertices u and v are adjacent iff $|u \cap v| = R - 1$.

Query-optimal parameters:

$$R = \# \text{ queries } = \text{ time}$$

Tani. 2007. An improved claw finding algorithm using quantum walk.

Tani's collision-finding algorithm

To find a collision between two functions $f : X \to S$ and $g : Y \to S$:

- Random walk on two Johnson graphs: one over X, the other over Y
- Check for collisions at each step
- Make it quantum!

Johnson graph over X

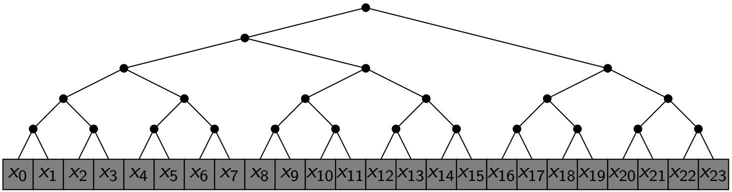
Vertices: R-element subsets of a fixed set X.
Vertices u and v are adjacent iff $|u \cap v| = R - 1$.

Query-optimal parameters to attack SIKE:

$$R = \text{ \# queries } = \text{ time } = p^{1/6+o(1)}$$

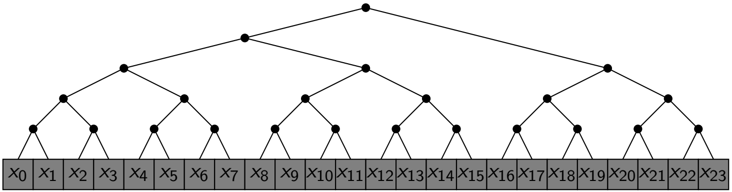
Tani. 2007. An improved claw finding algorithm using quantum walk.

Memory access



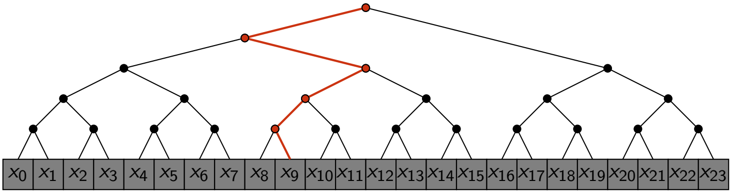
Memory access

Classical Query: 9



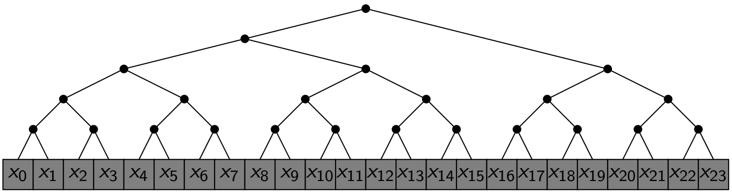
Memory access

Classical Query: 9



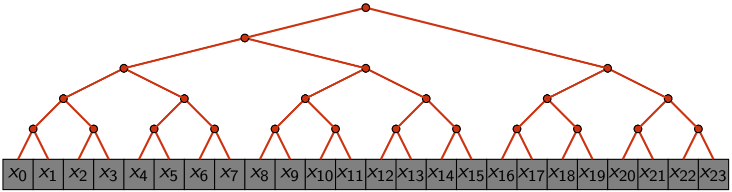
Memory access

Quantum Query: [illegible]



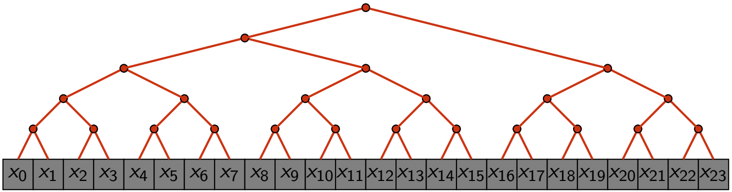
Memory access

Quantum Query:



Memory access

Quantum Query:



Analogy for Cryptographers

- Any physical “side channel” leaks information
- Any leaked information decoheres (destroys) the state
- Controller must implement circuits for all possible inputs

Memory costs

For N bits of random-access quantum memory:

Idle memory

- G-cost: Free
- DW-cost: $O(N)$ RAM ops per time step

Random access

- G-cost: $O(N)$ RAM ops
- DW-cost: $O(N \log N)$ RAM ops

Johnson vertex data structure

History independence

For quantum interference between random walk paths, the representation of a vertex must be independent of the path taken.

Bernstein, Jeffery, Lange, Meurer. 2013. Quantum algorithms for the subset-sum problem

Johnson vertex data structure

History independence

For quantum interference between random walk paths, the representation of a vertex must be independent of the path taken.

History-dependent:

- Binary tree as linked list

Bernstein, Jeffery, Lange, Meurer. 2013. Quantum algorithms for the subset-sum problem

Johnson vertex data structure

History independence

For quantum interference between random walk paths, the representation of a vertex must be independent of the path taken.

History-dependent:

- Binary tree as linked list

History-independent:

- Quantum radix tree: superposition over all layouts
- Sorted array: physically in order

Bernstein, Jeffery, Lange, Meurer. 2013. Quantum algorithms for the subset-sum problem

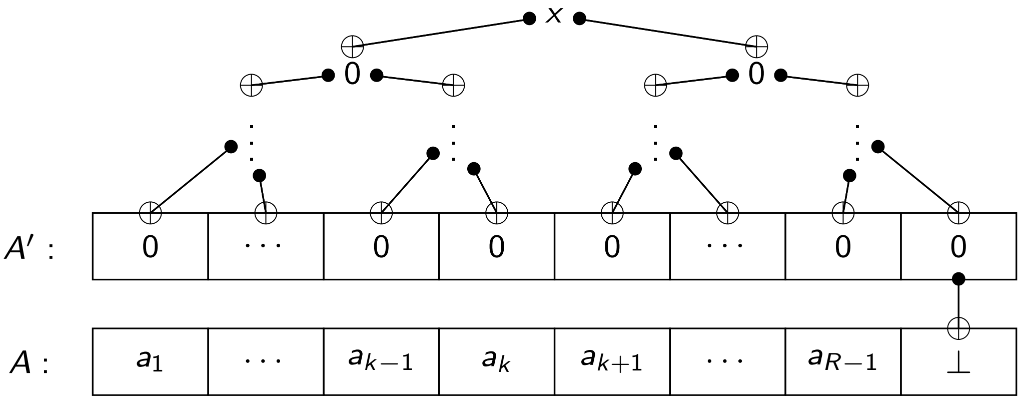
Johnson vertex insertion

Idea: We already pay $O(N)$ for memory access, so pay $O(N)$ to physically sort array:

A' :	0	$\cdots$	0	0	0	$\cdots$	0	0
A :	a_1	$\cdots$	a_{k-1}	a_k	a_{k+1}	$\cdots$	a_{R-1}	$\perp$

Johnson vertex insertion

1. "Fan out" an input x



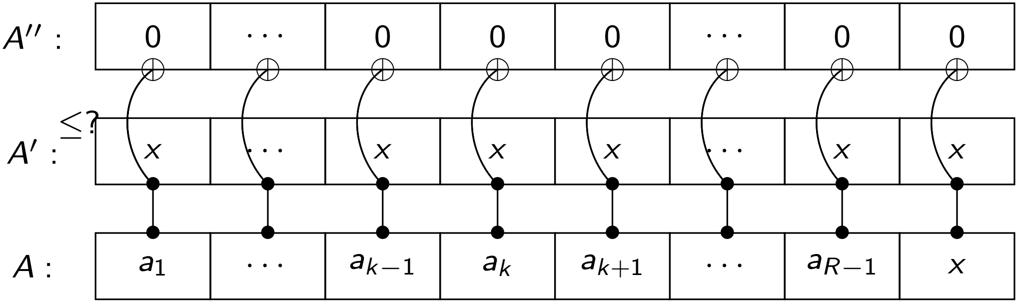
Johnson vertex insertion

1. "Fan out" an input x

A' :	x	$\cdots$	x	x	x	$\cdots$	x	x
A :	a_1	$\cdots$	a_{k-1}	a_k	a_{k+1}	$\cdots$	a_{R-1}	$\bot$

Johnson vertex insertion

2. Compare all elements simultaneously



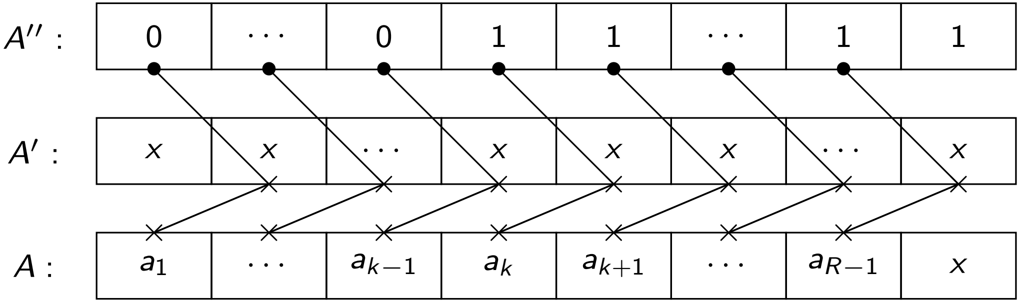
Johnson vertex insertion

2. Compare all elements simultaneously

A'' :	0	$\cdots$	0	1	1	$\cdots$	1	1
A' :	x	$\cdots$	x	x	x	$\cdots$	x	x
A :	a_1	$\cdots$	a_{k-1}	a_k	a_{k+1}	$\cdots$	a_{R-1}	x

Johnson vertex insertion

3. Conditionally swap "up"



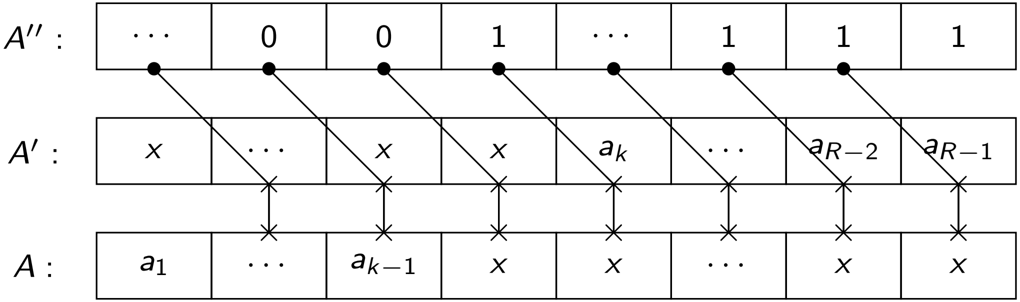
Johnson vertex insertion

3. Conditionally swap "up"

A'' :	0	$\cdots$	0	1	1	$\cdots$	1	1
A' :	x	$\cdots$	x	x	a_k	$\cdots$	a_{R-2}	a_{R-1}
A :	a_1	$\cdots$	a_{k-1}	x	x	$\cdots$	x	x

Johnson vertex insertion

4. Conditionally swap "down"



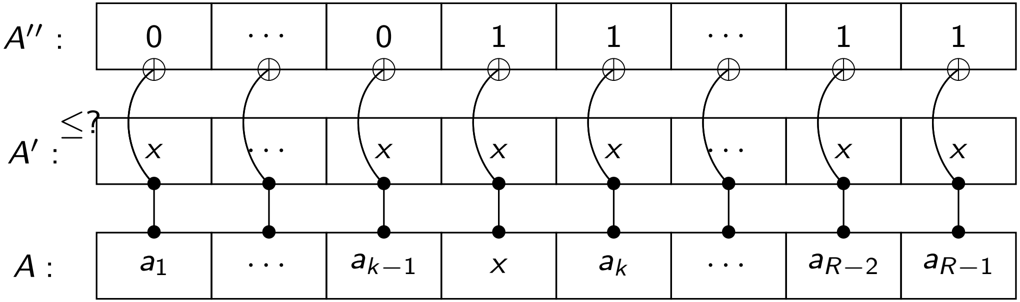
Johnson vertex insertion

4. Conditionally swap "down"

A'' :	$\cdots$	0	0	1	$\cdots$	1	1	1
A' :	x	$\cdots$	x	x	x	$\cdots$	x	x
A :	a_1	$\cdots$	a_{k-1}	x	a_k	$\cdots$	a_{R-2}	a_{R-1}

Johnson vertex insertion

5. Clear comparison bit



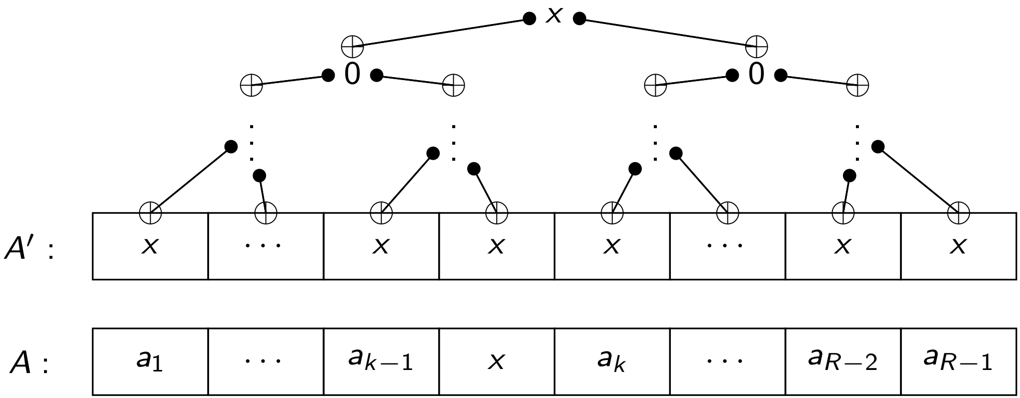
Johnson vertex insertion

5. Clear comparison bit

A'' :	0	$\cdots$	0	0	0	$\cdots$	0	0
A' :	x	$\cdots$	x	x	x	$\cdots$	x	x
A :	a_1	$\cdots$	a_{k-1}	x	a_k	$\cdots$	a_{R-2}	a_{R-1}

Johnson vertex insertion

7. Clear fan-out



Johnson vertex insertion

8. Insertion complete

A' :	0	$\cdots$	0	0	0	$\cdots$	0	0
A :	a_1	$\cdots$	a_{k-1}	x	a_k	$\cdots$	a_{R-2}	a_{R-1}

Costs of Tani's algorithm for SIKE

Previous analyses focused on the $p^{1/6}$ query cost of Tani's algorithm.

Using the Johnson vertex data structure, we find the SIKE secret at cost:

	Gates	Depth	Width	DW
Tani (query-optimal)	$p^{1/3+o(1)}$	$p^{1/6+o(1)}$	$p^{1/6+o(1)}$	$p^{1/3+o(1)}$

$$2^{434} \leq p \leq 2^{951}$$

Costs of Tani's algorithm for SIKE

Previous analyses focused on the $p^{1/6}$ query cost of Tani's algorithm.

Using the Johnson vertex data structure, we find the SIKE secret at cost:

	Gates	Depth	Width	DW
Tani (query-optimal)	$p^{1/3+o(1)}$	$p^{1/6+o(1)}$	$p^{1/6+o(1)}$	$p^{1/3+o(1)}$
Tani (G-optimal)	$p^{1/4+o(1)}$	$p^{1/4+o(1)}$	$p^{o(1)}$	$p^{1/4+o(1)}$
Tani (DW-optimal)	$p^{1/4+o(1)}$	$p^{1/4+o(1)}$	$p^{o(1)}$	$p^{1/4+o(1)}$

$$2^{434} \le p \le 2^{951}$$

Costs of Tani's algorithm for SIKE

Previous analyses focused on the $p^{1/6}$ query cost of Tani's algorithm.

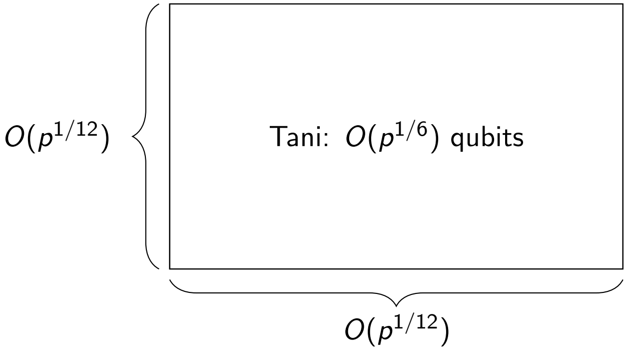
Using the Johnson vertex data structure, we find the SIKE secret at cost:

	Gates	Depth	Width	DW
Tani (query-optimal)	$p^{1/3+o(1)}$	$p^{1/6+o(1)}$	$p^{1/6+o(1)}$	$p^{1/3+o(1)}$
Tani (G-optimal)	$p^{1/4+o(1)}$	$p^{1/4+o(1)}$	$p^{o(1)}$	$p^{1/4+o(1)}$
Tani (DW-optimal)	$p^{1/4+o(1)}$	$p^{1/4+o(1)}$	$p^{o(1)}$	$p^{1/4+o(1)}$
Grover (G-optimal)	$p^{1/4+o(1)}$	$p^{1/4+o(1)}$	$p^{o(1)}$	$p^{1/4+o(1)}$

$$2^{434} \leq p \leq 2^{951}$$

Comparison with parallel Grover

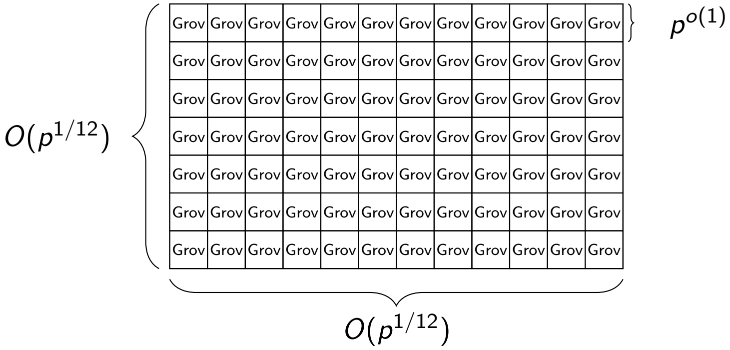
The classical controller can apply gates to every qubit to run Tani's algorithm.
It could instead group them together and run Grover's search algorithm.



Grover and Rudolph. 2004. How significant are the known collision and element distinctness quantum algorithms

Comparison with parallel Grover

The classical controller can apply gates to every qubit to run Tani's algorithm.
It could instead group them together and run Grover's search algorithm.



Grover and Rudolph. 2004. How significant are the known collision and element distinctness quantum algorithms

Comparison with parallel Grover

$O(p^{1/6})$ copies of Grover finds isogeny in time $O(p^{1/6})$.

$p^{o(1)}$

$O(p^{1/12})$

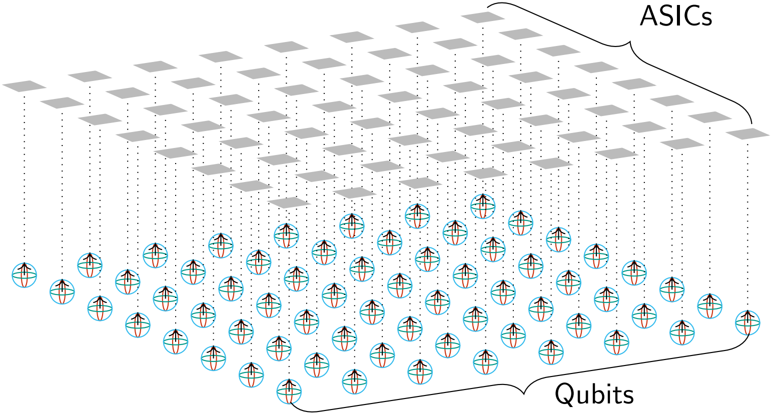
Grov	Grov	Grov	Grov	Grov	Grov	Grov	Grov	Grov	Grov	Grov	Grov
Grov	Grov	Grov	Grov	Grov	Grov	Grov	Grov	Grov	Grov	Grov	Grov
Grov	Grov	Grov	Grov	Grov	Grov	Grov	Grov	Grov	Grov	Grov	Grov
Grov	Grov	Grov	Grov	Grov	Grov	Grov	Grov	Grov	Grov	Grov	Grov
Grov	Grov	Grov	Grov	Grov	Grov	Grov	Grov	Grov	Grov	Grov	Grov
Grov	Grov	Grov	Grov	Grov	Grov	Grov	Grov	Grov	Grov	Grov	Grov
Grov	Grov	Grov	Grov	Grov	Grov	Grov	Grov	Grov	Grov	Grov	Grov

$O(p^{1/12})$

Grover and Rudolph. 2004. How significant are the known collision and element distinctness quantum algorithms

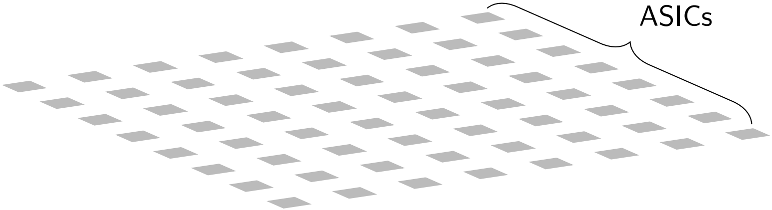
Comparison with van Oorschot-Wiener

- Time/query-optimal Tani has $O(p^{1/6})$ classical control processors.
- We could reprogram these to run van Oorschot-Wiener (VW)



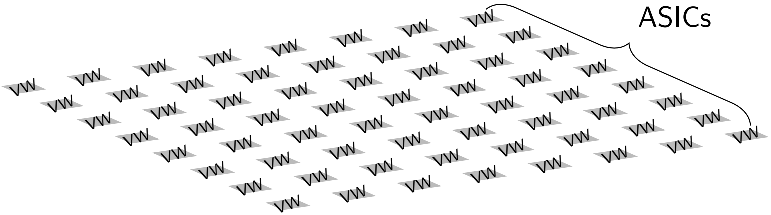
Comparison with van Oorschot-Wiener

- Time/query-optimal Tani has $O(p^{1/6})$ classical control processors.
- We could reprogram these to run van Oorschot-Wiener (VW)



Comparison with van Oorschot-Wiener

- Time/query-optimal Tani has $O(p^{1/6})$ classical control processors.
- We could reprogram these to run van Oorschot-Wiener (VW)

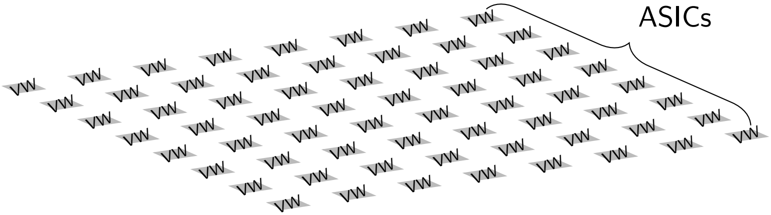


Comparison with van Oorschot-Wiener

- Time/query-optimal Tani has $O(p^{1/6})$ classical control processors.
- We could reprogram these to run van Oorschot-Wiener (VW)

Conclusion

$O(p^{1/6})$ parallel instances of van Oorschot-Wiener find isogeny in time $O(p^{1/8})$, faster than the quantum algorithms.



Memory peripheral framework

1. **Memory** is a physical system that changes over time
2. A **memory controller** interacts with a memory
3. The **cost** of a computation is the number of interactions

Conclusions

- In a quantum computer, qubits are a peripheral of a classical computer.
- Quantum memory access has a linear gate cost.
- Active error correction gives cost to the identity gate.
- SIKE is more secure than previously thought.

samuel.jaques@materials.ox.ac.uk jmschank@uwaterloo.ca

UNIVERSITY OF WATERLOO