# Cost analysis of hash collisions: will quantum computers make SHARCS obsolete?

D. J. Bernstein
University of Illinois at Chicago

NSF ITR–0716498

# Quantum vs. SHARCS

Exactly how expensive is it to break RSA-1024, ECC-160, etc.?

Many papers on the topic.
Widespread interest today.

## Quantum vs. SHARCS

Exactly how expensive is it to break RSA-1024, ECC-160, etc.?

Many papers on the topic.
Widespread interest today.

But quantum computing says:
"All your circuit designs will soon be obsolete!
Our quantum computers will break RSA and ECC in polynomial time."

Exactly how expensive is it
to invert a hash function,
find a cipher key, etc.?
$2^b$ “operations” for $b$-bit key;
how expensive is an “operation”?

Many papers on the topic.
Widespread interest today.

Exactly how expensive is it
to invert a hash function,
find a cipher key, etc.?
$2^b$ "operations" for $b$-bit key;
how expensive is an "operation"?

Many papers on the topic.
Widespread interest today.

But quantum computing says:
"All your circuit designs
will soon be obsolete!
Our quantum computers
will find a $b$-bit key
in time only $2^{b/2}$."

Exactly how expensive is it
to find *collisions*
in a hash function?
$2^{b/2}$ "operations" for $b$-bit hash;
how expensive is an "operation"?

Many papers on the topic.
Widespread interest today.

Exactly how expensive is it to find *collisions* in a hash function?
$2^{b/2}$ "operations" for $b$-bit hash; how expensive is an "operation"?

Many papers on the topic. Widespread interest today.

But quantum computing says: "All your circuit designs will soon be obsolete! Our quantum computers will find a $b$-bit collision in time only $2^{b/3}$."

Main point of my paper:
All known quantum algorithms are fundamentally *slower* than traditional collision circuits, despite optimistic assumptions re quantum-computer speed.

Main point of my paper:
All known quantum algorithms are fundamentally *slower* than traditional collision circuits, despite optimistic assumptions re quantum-computer speed.

Extra point of this talk:
Optimization experience for ASICs/FPGAs/other meshes will be even more valuable in a quantum-computing world.
"Quantum SHARCS"?

# Two quantum algorithms

1994 Shor:
Fast quantum period-finding.
Gives polynomial-time
quantum solution to DLP.

1996 Grover, 1997 Grover:
Fast quantum search.

Practically all quantum algorithms
are Shor/Grover applications.
See 2003 Shor, "Why haven't
more quantum algorithms been
found?"; 2004 Shor.

Grover explicitly constructs
a quantum circuit $\mathrm{Gr}(F)$
to find a root of $F$,
assuming root is unique.

"Only $\sqrt{N}$ steps."
$N = 2^b$ if $F$ maps
$b$-bit input to 1-bit output.

Success probability $\geq 1/2$.
Can use fewer steps but
probability degrades quadratically.

$F$: any computable function.

Can specify $F$ by a
classical combinatorial circuit:
a directed acyclic graph
of NAND computations
from $b$ input bits
to 1 output bit.

$F$: any computable function.

Can specify $F$ by a
classical combinatorial circuit:
a directed acyclic graph
of NAND computations
from $b$ input bits
to 1 output bit.

Without serious overhead
(and maybe reducing power!)
can replace NAND gates by
reversible "Toffoli gates"
$r, s, t \mapsto r, s, t \oplus rs$.
Obtain $x, t \mapsto x, F(x) \oplus t$.

The basic quantum conversion: replace each Toffoli gate by a quantum Toffoli gate. Resulting quantum circuit computes $x, t \mapsto x, F(x) \oplus t$ where $x$ is a *quantum superposition* of $b$-bit inputs.

The basic quantum conversion: replace each Toffoli gate by a quantum Toffoli gate. Resulting quantum circuit computes $x, t \mapsto x, F(x) \oplus t$ where $x$ is a *quantum superposition* of $b$-bit inputs.

Grover builds a superposition of all possible strings $x$; applies this circuit; applies an easy quantum flip to build a new result $x$; repeats $\Theta(2^{b/2})$ times.

What if $F$ has more roots?

1996 Boyer–Brassard–Høyer–Tapp, generalizing Grover:
"time in $O(\sqrt{N/t})$"
if there are $t$ roots.

What if $F$ has more roots?

1996 Boyer–Brassard–Høyer–Tapp, generalizing Grover: "time in $O(\sqrt{N/t})$" if there are $t$ roots.

Don't need generalization. Can simply apply Grover to $x \mapsto F(R(x))$ where $x$ has $\approx b - \lg t$ bits, $R$ is random affine map.

What if $F$ has more roots?

1996 Boyer–Brassard–Høyer–Tapp, generalizing Grover: "time in $O(\sqrt{N/t})$" if there are $t$ roots.

Don't need generalization. Can simply apply Grover to $x \mapsto F(R(x))$ where $x$ has $\approx b - \lg t$ bits, $R$ is random affine map.

Unknown $t$? Simply guess. . . . but BBHT is more streamlined.

# Grover space and time

Don't have to unroll $F$
into a combinatorial circuit.

Take any circuit of area $A$
(using reversible gates!)
that reads $x, t$ at the top,
ends with $x$, $F(x) \oplus t$ at the top,
where $x$ is a $b$-bit string.

Convert gates to quantum gates.
Obtain quantum circuit
that reads $x, t$ at the top,
ends with $x$, $F(x) \oplus t$ at the top,
where $x$ is a quantum
superposition of $b$-bit strings.

Don't unroll Grover iterations.

Need some extra space
for quantum flip etc.,
but total Grover circuit size
will be essentially $A$.

Don't unroll Grover iterations.

Need some extra space
for quantum flip etc.,
but total Grover circuit size
will be essentially $A$.

"Aren't quantum gates
much larger than classical gates?"
— Yes. Constants matter!
But this talk makes
best-case assumption
that the overhead
doesn't grow with $A$.

“Time in $O(\sqrt{N})$”
fails to account for $F$ time.

Assume that original circuit
computes $F$ in time $T$.

Each Grover iteration
takes time essentially $T$.
Total time essentially $T\sqrt{N}$.

"Time in $O(\sqrt{N})$"
fails to account for $F$ time.

Assume that original circuit
computes $F$ in time $T$.

Each Grover iteration
takes time essentially $T$.
Total time essentially $T\sqrt{N}$.

"Aren't quantum gates much
slower than classical gates?"
— Yes, but again assume
no $(A, T)$-dependent penalty.

"Can quantum gates
operate with just as much
parallelism as original gates?"

— Best-case assumption: Yes.

Example: RAM lookup $x \mapsto A[x]$
is actually computing
$A[0](x = 0) + A[1](x = 1) + \cdots$;
$n$ terms if $A$ has size $n$.
The basic quantum conversion
produces $\Omega(n)$ quantum gates
... which, presumably,
can all operate in parallel.
Realistic mesh/speed of light
$\Rightarrow$ wire delay $\Rightarrow$ time $\Omega(\sqrt{n})$.

# Guessing a collision

Consider a hash function
$H : \mathbf{F}_2^{b+1} \to \mathbf{F}_2^b$.

Define $F : \mathbf{F}_2^{b+1} \times \mathbf{F}_2^{b+1} \to \mathbf{F}_2$
as follows: $F(x, y) =$
0 if $x \neq y$ and $H(x) = H(y)$;
1 if $x = y$ or $H(x) \neq H(y)$.

A collision in $H$ is,
by definition, a root of $F$.

Easiest way to find collision:
search randomly for root of $F$.

Assume circuit of area $A$ computes $H$ in time $T$.

Then circuit of area $\approx A$ computes $F$ in time $\approx T$.
("You mean $2A$?" — Roughly.)

Collision chance $\geq 1/2^{b+1}$ for a uniform random pair $(x, x')$.

Trying $2^{b+1}$ pairs
takes time $\approx 2^b T$
on circuit of area $\approx A$.

Grover takes time $\approx 2^{b/2} T$
on quantum circuit of area $\approx A$.

# Table lookups

Generate many random inputs $x_1, x_2, \ldots, x_M$; e.g. $M = 2^{b/3}$.

Compute and sort $M$ pairs $(H(x_1), x_1)$, $(H(x_2), x_2)$, $\ldots$, $(H(x_M), x_M)$ in lex order.

Generate a random input $y$. Check for $H(y)$ in sorted list. Keep trying more $y$'s until collision is found.

Collision chance $\approx M/2^b$
for each $y$.

Naive free-communication model:
Table lookup takes time $\approx 1$.
Total time $\approx (M + 2^b/M)(T + 1)$
on circuit of area $\approx A + M$.

e.g. time $\approx 2^{2b/3}T$
on circuit of area $\approx A + 2^{b/3}$.

Realistic model:
Table lookup takes time $\approx \sqrt{M}$.
Total time
$\approx (M + 2^b/M)(T + \sqrt{M})$
on circuit of area $\approx A + M$.

Define $F(y)$ as 0 iff there is a collision among $(x_1, y), (x_2, y), \ldots, (x_M, y)$.

We're guessing root of $F$.

1998 Brassard–Høyer–Tapp: Instead use quantum search; "time" $2^{b/3}$ if $M = 2^{b/3}$.

Wow, faster than $2^{b/2}$! Many people say this is scary. ECRYPT Hash Function Website: "For collision resistance at least 384 bits are needed."

Let's look at the actual costs
of 1998 Brassard–Høyer–Tapp.

Naive free-communication model:
Total time $\approx (M+\sqrt{2^b/M})(T+1)$
on quantum circuit
of area $\approx A + M$.

(Realistic model: Slower.
See paper for details.)

e.g. $M = 2^{b/3}$:
time $\approx 2^{b/3}T$,
area $\approx A + 2^{b/3}$.

2003 Grover–Rudolph,
"How significant are the known collision and element distinctness quantum algorithms?":

With such a huge machine, can simply run $2^{b/3}$ parallel quantum searches for collisions $(x, x')$.

High probability of success within "time" $2^{b/3}$.

But these algorithms are giant steps backwards!

Standard collision circuits, 1994 van Oorschot–Wiener: time $\approx 2^{b/4}T$, area $\approx 2^{b/4}A$.

This is much faster than 1998 Brassard–Høyer–Tapp, on a much smaller circuit.

My paper presents newer, faster quantum collision algorithms, but I conjecture optimality for the standard circuits.