The quantum threat to cryptography

Ashley Montanaro

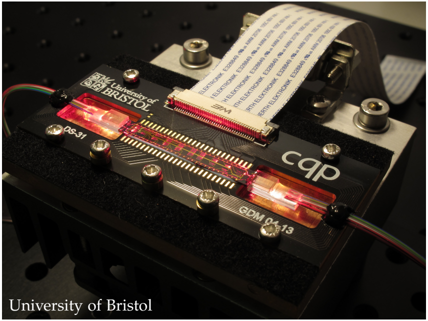
School of Mathematics,
University of Bristol

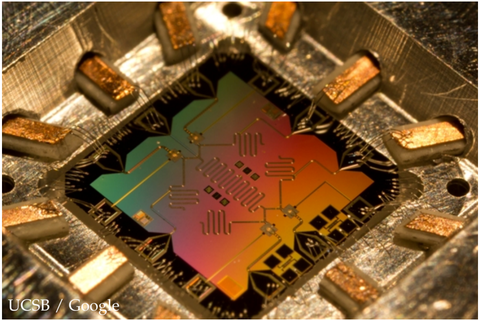
20 October 2016

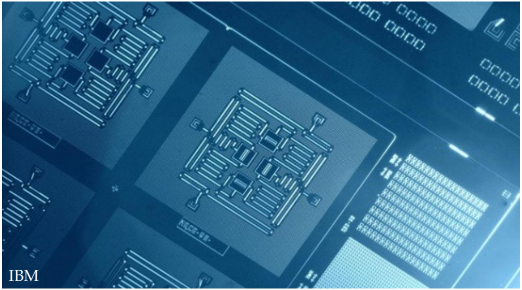
University of Bristol

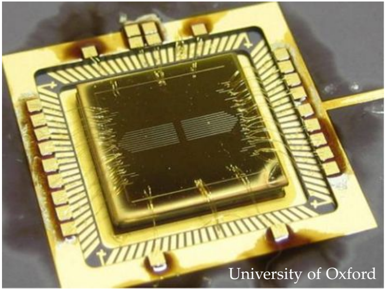
EPSRC
Engineering and Physical Sciences Research Council

Quantum computers


University of Bristol


UCSB / Google


IBM


University of Oxford

Experimental progress

Important aspects of a quantum computer are:

- The number of qubits (quantum bits) it has;
- The number of quantum gates (elementary operations) it can execute, and the speed with which it does so;
- Whether it is fault-tolerant and scalable.

Experimental progress

Important aspects of a quantum computer are:

- The number of qubits (quantum bits) it has;
- The number of quantum gates (elementary operations) it can execute, and the speed with which it does so;
- Whether it is fault-tolerant and scalable.

Current experiments have 5-10 qubits, can implement a few hundred gates, and are not fault-tolerant.

Experimental progress

Important aspects of a quantum computer are:

- The number of qubits (quantum bits) it has;
- The number of quantum gates (elementary operations) it can execute, and the speed with which it does so;
- Whether it is fault-tolerant and scalable.

Current experiments have 5-10 qubits, can implement a few hundred gates, and are not fault-tolerant.

For practical (e.g. crypto) applications, we would need 100-10,000+ fault-tolerant qubits and to perform 10^8-10^{12} gates.

Experimental progress

Important aspects of a quantum computer are:

- The number of qubits (quantum bits) it has;
- The number of quantum gates (elementary operations) it can execute, and the speed with which it does so;
- Whether it is fault-tolerant and scalable.

Current experiments have 5-10 qubits, can implement a few hundred gates, and are not fault-tolerant.

For practical (e.g. crypto) applications, we would need 100-10,000+ fault-tolerant qubits and to perform 10^8-10^{12} gates.

Estimates for when this will be achieved vary, but only a pessimist would bet \$1M on it taking >20 years...

Introduction

One of the important applications of quantum computers is expected to be attacking cryptosystems that are designed to be secure against classical adversaries.

The rest of this talk:

1. Efficient quantum attacks on public-key cryptosystems;
2. General-purpose quantum algorithms and applications to cryptographic tasks.

Integer factorisation

Problem

Given an n-digit integer $N = p \times q$ for primes p and q, determine p and q.

Integer factorisation

Problem

Given an n-digit integer $N = p \times q$ for primes p and q, determine p and q.

- The best (classical!) algorithm we have for factorisation (the number field sieve) runs in time

$$\exp(O(n^{1/3}(\log n)^{2/3}))$$

Integer factorisation

Problem

Given an n-digit integer $N = p \times q$ for primes p and q, determine p and q.

- The best (classical!) algorithm we have for factorisation (the number field sieve) runs in time

$$\exp(O(n^{1/3}(\log n)^{2/3}))$$

- The RSA cryptosystem is based around the hardness of this task. If we can factorise large integers efficiently, we can break RSA.

Integer factorisation

Problem

Given an n-digit integer $N = p \times q$ for primes p and q, determine p and q.

- The best (classical!) algorithm we have for factorisation (the number field sieve) runs in time

$$\exp(O(n^{1/3}(\log n)^{2/3}))$$

- The RSA cryptosystem is based around the hardness of this task. If we can factorise large integers efficiently, we can break RSA.

Theorem [Shor '97]

There is a quantum algorithm which finds the prime factors of an n-digit integer in time $O(n^3)$.

Shor's algorithm: complexity comparison

Very roughly (ignoring constant factors!):

Number of digits	Timesteps (quantum)	Timesteps (classical)
100	10^6	$\sim 4 \times 10^5$
1,000	10^9	$\sim 5 \times 10^{15}$
10,000	10^{12}	$\sim 1 \times 10^{41}$

Shor's algorithm: complexity comparison

Very roughly (ignoring constant factors!):

Number of digits	Timesteps (quantum)	Timesteps (classical)
100	10^6	$\sim 4 \times 10^5$
1,000	10^9	$\sim 5 \times 10^{15}$
10,000	10^{12}	$\sim 1 \times 10^{41}$

Based on these figures, a 10,000-digit number could be factorised by:

- A quantum computer with a clock speed of 1MHz in 11 days.

Shor's algorithm: complexity comparison

Very roughly (ignoring constant factors!):

Number of digits	Timesteps (quantum)	Timesteps (classical)
100	10^6	$\sim 4 \times 10^5$
1,000	10^9	$\sim 5 \times 10^{15}$
10,000	10^{12}	$\sim 1 \times 10^{41}$

Based on these figures, a 10,000-digit number could be factorised by:

- A quantum computer with a clock speed of 1MHz in 11 days.
- The fastest computer on the Top500 supercomputer list ($\sim 9.3 \times 10^{16}$ operations per second) in $\sim 3.4 \times 10^{16}$ years.

(see e.g. [Van Meter et al '05] for a more detailed comparison)

But a cautionary note…

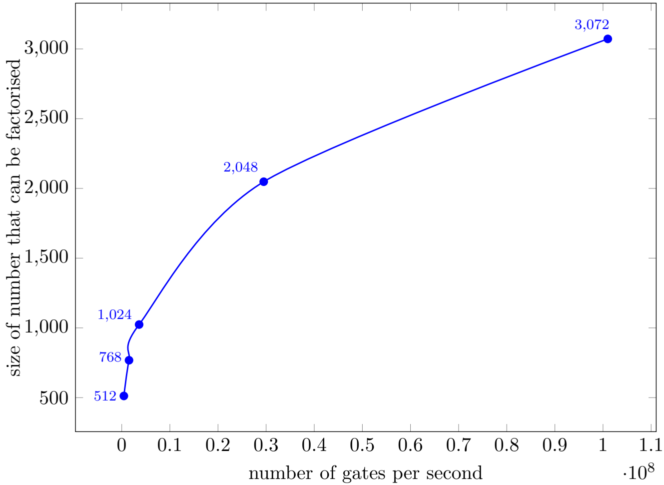


Figure 58: The relation between speed of a quantum computer and the size of number that can be factorised in 1 day

Pic: Nicharee Techatanerut, 2014

Hidden subgroup problems

Hidden subgroup problem (e.g. [Boneh and Lipton '95])

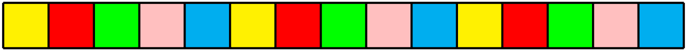
Let G be a group. Given oracle access to a function $f : G \to X$ such that f is constant on the cosets of some subgroup $H \leqslant G$, and distinct on each coset, identify H.

Hidden subgroup problems

Hidden subgroup problem (e.g. [Boneh and Lipton '95])

Let G be a group. Given oracle access to a function $f : G \to X$ such that f is constant on the cosets of some subgroup $H \leqslant G$, and distinct on each coset, identify H.

Integer factorisation reduces to the case $G = \mathbb{Z}_M$ for some integer M. This is the problem of determining the period of a periodic function:

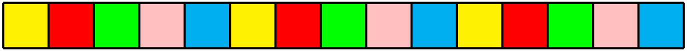


Hidden subgroup problems

Hidden subgroup problem (e.g. [Boneh and Lipton '95])

Let G be a group. Given oracle access to a function $f : G \to X$ such that f is constant on the cosets of some subgroup $H \leqslant G$, and distinct on each coset, identify H.

Integer factorisation reduces to the case $G = \mathbb{Z}_M$ for some integer M. This is the problem of determining the period of a periodic function:



On a quantum computer, the HSP can be solved using $O(\log |G|)$ queries to f for all groups G [Ettinger et al. '04]. Classically, some groups require $\Omega(\sqrt{|G|})$ queries [Simon '97].

Hidden subgroup problems

The HSP is related to many other problems and cryptosystems:

Problem	Group	Complexity	Cryptosystem
Factorisation	$\mathbb{Z}_N$	Polynomial[1]	RSA
Discrete log	$\mathbb{Z}_{p-1} \times \mathbb{Z}_{p-1}$	Polynomial[1]	Diffie-Hellman, DSA, ...
Elliptic curve d. log	Elliptic curve	Polynomial[2]	ECDH, ECDSA, ...
Principal ideal	$\mathbb{R}$	Polynomial[3]	Buchmann-Williams
Principal ideal	$\mathbb{R}^m$	Polynomial[4]	Smart-Vercauteren, ...
Shortest lattice vector	Dihedral grp	Subexp.[5]	NTRU, Ajtai-Dwork, ...
Graph isomorphism	Symmetric grp	Exponential	—

[1]Shor '97, [2]Proos et al. '03, [3]Hallgren '07, [4]Eisenträger et al. '14, Biasse and Song '15, [5]Kuperberg '05, Regev '04

Hidden subgroup problems

The HSP is related to many other problems and cryptosystems:

Problem	Group	Complexity	Cryptosystem
Factorisation	$\mathbb{Z}_N$	Polynomial[1]	RSA
Discrete log	$\mathbb{Z}_{p-1} \times \mathbb{Z}_{p-1}$	Polynomial[1]	Diffie-Hellman, DSA, ...
Elliptic curve d. log	Elliptic curve	Polynomial[2]	ECDH, ECDSA, ...
Principal ideal	$\mathbb{R}$	Polynomial[3]	Buchmann-Williams
Principal ideal	$\mathbb{R}^m$	Polynomial[4]	Smart-Vercauteren, ...
Shortest lattice vector	Dihedral grp	Subexp.[5]	NTRU, Ajtai-Dwork, ...
Graph isomorphism	Symmetric grp	Exponential	—

[1]Shor ’97, [2]Proos et al. ’03, [3]Hallgren ’07, [4]Eisenträger et al. ’14, Biasse and Song ’15, [5]Kuperberg ’05, Regev ’04

A significant amount of other work on the HSP has resolved its complexity for many other groups.

Other cryptosystems?

As RSA is insecure against quantum attack, can we switch to something else?

Other cryptosystems?

As RSA is insecure against quantum attack, can we switch to something else?

The field of post-quantum cryptography tries to develop cryptosystems which are secure against quantum attack.

Other cryptosystems?

As RSA is insecure against quantum attack, can we switch to something else?

The field of post-quantum cryptography tries to develop cryptosystems which are secure against quantum attack.

- May 2006: first conference on post-quantum crypto held.
- 2014-2016: post-quantum crypto companies emerge: e.g. Post-Quantum, ISARA, . . . ?
- Aug 2015: NSA states that "we anticipate a need to shift to quantum-resistant cryptography in the near future"
- July 2016: Google announces that a candidate post-quantum cryptosystem ("New Hope") has been implemented as an experiment in Chrome.

“Post-quantum” cryptosystems

Some examples of cryptosystems which have thus far resisted quantum attack:

- The McEliece cryptosystem, which is (roughly) based on the hardness of finding transformations between equivalent linear codes.

“Post-quantum” cryptosystems

Some examples of cryptosystems which have thus far resisted quantum attack:

- The McEliece cryptosystem, which is (roughly) based on the hardness of finding transformations between equivalent linear codes.
- There can be no efficient quantum attack on this cryptosystem based on simple Fourier sampling (the key ingredient in Shor’s algorithm) [Dinh et al ’10].

"Post-quantum" cryptosystems

Some examples of cryptosystems which have thus far resisted quantum attack:

- The McEliece cryptosystem, which is (roughly) based on the hardness of finding transformations between equivalent linear codes.
- There can be no efficient quantum attack on this cryptosystem based on simple Fourier sampling (the key ingredient in Shor's algorithm) [Dinh et al '10].
- Lattice-based cryptosystems dependent on the hardness of solving closest/shortest vector problems in lattices.

“Post-quantum” cryptosystems

Some examples of cryptosystems which have thus far resisted quantum attack:

- The McEliece cryptosystem, which is (roughly) based on the hardness of finding transformations between equivalent linear codes.
- There can be no efficient quantum attack on this cryptosystem based on simple Fourier sampling (the key ingredient in Shor’s algorithm) [Dinh et al ’10].
- Lattice-based cryptosystems dependent on the hardness of solving closest/shortest vector problems in lattices.
- No polynomial-time quantum algorithm for these problems has been found for general lattices (but there is an efficient algorithm for lattices with more structure [e.g. Campbell et al. ’14, Biasse and Song ’15]).

Grover's algorithm

One of the most basic problems in computer science is unstructured search.

Grover's algorithm

One of the most basic problems in computer science is unstructured search.

- Imagine we have access to a function $f : \{0,1\}^n \to \{0,1\}$ which we treat as a black box.

Grover's algorithm

One of the most basic problems in computer science is unstructured search.

- Imagine we have access to a function $f : \{0,1\}^n \to \{0,1\}$ which we treat as a black box.

- We want to find an x such that $f(x) = 1$.

0	0	1	0	0	0	1	0

Grover's algorithm

One of the most basic problems in computer science is unstructured search.

- Imagine we have access to a function $f : \{0,1\}^n \rightarrow \{0,1\}$ which we treat as a black box.

- We want to find an x such that $f(x) = 1$.

0	0	1	0	0	0	1	0

- On a classical computer, this task could require 2^n queries to f in the worst case. But on a quantum computer, Grover's algorithm [Grover '97] can solve the problem with $O(\sqrt{2^n})$ queries to f (and bounded failure probability).

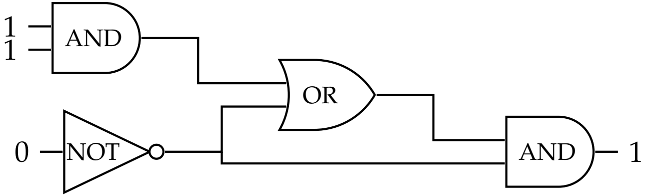
Applications of Grover's algorithm

Grover's algorithm gives a speedup over naïve algorithms for any decision problem in the complexity class NP, i.e. where we can verify the solution efficiently.

Applications of Grover's algorithm

Grover's algorithm gives a speedup over naïve algorithms for any decision problem in the complexity class NP, i.e. where we can verify the solution efficiently.

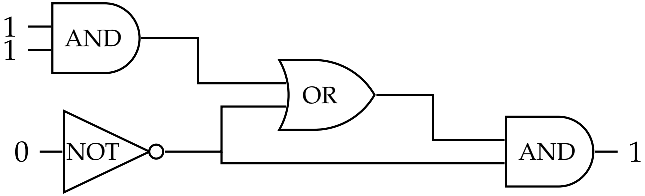
- For example, in the Circuit SAT problem we would like to find an input to a circuit on n bits such that the output is 1:



Applications of Grover's algorithm

Grover's algorithm gives a speedup over naïve algorithms for any decision problem in the complexity class NP, i.e. where we can verify the solution efficiently.

- For example, in the Circuit SAT problem we would like to find an input to a circuit on n bits such that the output is 1:



- Grover's algorithm improves the runtime from $O(2^n)$ to $O(2^{n/2}\,\mathrm{poly}(n))$: applications to design automation, circuit equivalence, model checking, ...

Quadratic speedup

Is a quadratic speedup significant?

Quadratic speedup

Is a quadratic speedup significant?

A concrete example: Circuit SAT with different clock speeds.

	Classical		**Quantum**		
Input bits	1MHz	1GHz	1KHz	10KHz	1MHz
30	18s	1s	32s	3s	0.03s
40	13d	18m	17m	104s	1s
50	36y	13d	9h	55m	33s
60	37M	36y	12d	1d	18m

Speeds listed are approximate, effective speeds (i.e. number of circuit evaluations per second) after overhead for fault-tolerance.

Cryptographic applications of Grover's algorithm

Password checking (preimage-finding) is an obvious application of Grover's algorithm: if there are N possible passwords, we can crack a password with $O(\sqrt{N})$ checks.

Cryptographic applications of Grover's algorithm

Password checking (preimage-finding) is an obvious application of Grover's algorithm: if there are N possible passwords, we can crack a password with $O(\sqrt{N})$ checks.

Grover's algorithm can also be used as a subroutine to:

- Speed up the information-set method for breaking the McEliece cryptosystem [Bernstein '10];
- Find short lattice vectors more efficiently [Laarhoven '15].

Cryptographic applications of Grover's algorithm

Password checking (preimage-finding) is an obvious application of Grover's algorithm: if there are N possible passwords, we can crack a password with $O(\sqrt{N})$ checks.

Grover's algorithm can also be used as a subroutine to:

- Speed up the information-set method for breaking the McEliece cryptosystem [Bernstein '10];
- Find short lattice vectors more efficiently [Laarhoven '15].

In all these cases, one need only increase the key length by a constant factor to achieve the same level of security as was the case classically.

Other notes on Grover's algorithm

Grover's algorithm is not parallelisable in the following sense:

- Imagine we have K quantum or classical computers solving a search problem in a space of size N.
- Classical complexity: $O(N/K)$ per computer $\Rightarrow$ total effort $O(N)$.
- Quantum complexity: $O(\sqrt{N/K})$ per computer $\Rightarrow$ total effort $O(\sqrt{NK})$.

Finding hash function collisions

Quantum computers can also be used to find collisions in hash functions etc. more efficiently than classically:

- [Brassard, Høyer and Tapp '98] gave a quantum algorithm finding a collision in an 2-to-1 function $f : [N] \to X$ using the function $O(N^{1/3})$ times.

Finding hash function collisions

Quantum computers can also be used to find collisions in hash functions etc. more efficiently than classically:

- [Brassard, Høyer and Tapp '98] gave a quantum algorithm finding a collision in an 2-to-1 function $f : [N] \to X$ using the function $O(N^{1/3})$ times.
- This beats the best possible classical complexity of $O(N^{1/2})$ function evaluations.

Finding hash function collisions

Quantum computers can also be used to find collisions in hash functions etc. more efficiently than classically:

- [Brassard, Høyer and Tapp '98] gave a quantum algorithm finding a collision in an 2-to-1 function $f : [N] \to X$ using the function $O(N^{1/3})$ times.
- This beats the best possible classical complexity of $O(N^{1/2})$ function evaluations.
- However, the quantum algorithm uses $O(N^{1/3})$ space and might not be faster than the best known classical algorithms in practice [Bernstein '09].

Finding hash function collisions

Quantum computers can also be used to find collisions in hash functions etc. more efficiently than classically:

- [Brassard, Høyer and Tapp '98] gave a quantum algorithm finding a collision in an 2-to-1 function $f : [N] \to X$ using the function $O(N^{1/3})$ times.
- This beats the best possible classical complexity of $O(N^{1/2})$ function evaluations.
- However, the quantum algorithm uses $O(N^{1/3})$ space and might not be faster than the best known classical algorithms in practice [Bernstein '09].

Finding a collision without the $2\to 1$ promise can be done with $O(N^{2/3})$ function evaluations [Ambainis '04], and this is tight.

Quantum speedup of backtracking algorithms

Backtracking algorithms can be used to solve constraint satisfaction problems (CSPs) where we are able to determine efficiently whether partial assignments to the variables can be extended to full solutions.

Quantum speedup of backtracking algorithms

Backtracking algorithms can be used to solve constraint satisfaction problems (CSPs) where we are able to determine efficiently whether partial assignments to the variables can be extended to full solutions.

- These algorithms explore a tree of partial solutions until they find a complete solution to the problem.

Quantum speedup of backtracking algorithms

Backtracking algorithms can be used to solve constraint satisfaction problems (CSPs) where we are able to determine efficiently whether partial assignments to the variables can be extended to full solutions.

- These algorithms explore a tree of partial solutions until they find a complete solution to the problem.
- [AM '15, informal]: if there is a classical backtracking algorithm which solves a CSP on n variables in time T, there is a quantum algorithm which solves the same problem in time $O(\sqrt{T}\,\mathrm{poly}(n))$.

Quantum speedup of backtracking algorithms

Backtracking algorithms can be used to solve constraint satisfaction problems (CSPs) where we are able to determine efficiently whether partial assignments to the variables can be extended to full solutions.

- These algorithms explore a tree of partial solutions until they find a complete solution to the problem.
- [AM '15, informal]: if there is a classical backtracking algorithm which solves a CSP on n variables in time T, there is a quantum algorithm which solves the same problem in time $O(\sqrt{T}\,\mathrm{poly}(n))$.
- Can be applied e.g. to speed up enumeration attacks on lattice-based cryptosystems [del Pino et al. '16, Alkim et al. '16].

Summary and further reading

- Quantum computation has already had a substantial impact on the world of cryptography…

Summary and further reading

- Quantum computation has already had a substantial impact on the world of cryptography…
- …the actual development of a large-scale quantum computer could have significantly more impact.

Summary and further reading

- Quantum computation has already had a substantial impact on the world of cryptography…
- …the actual development of a large-scale quantum computer could have significantly more impact.
- Very few people have worked on quantum algorithms for breaking cryptosystems!

Summary and further reading

- Quantum computation has already had a substantial impact on the world of cryptography…
- …the actual development of a large-scale quantum computer could have significantly more impact.
- Very few people have worked on quantum algorithms for breaking cryptosystems!

See the Quantum Algorithm Zoo for over 320 papers on quantum algorithms: http://math.nist.gov/quantum/zoo/

Quantum algorithms: an overview,
AM, *npj Quantum Information* 2, 2016
www.nature.com/articles/npjqi201523