Quantum algorithms: From foundations to applications

Ashley Montanaro

Department of Computer Science,
University of Bristol

8 November 2018

University of Bristol

EPSRC
Engineering and Physical Sciences Research Council

Quantum algorithms

- Quantum computers are designed to use quantum mechanics to outperform any possible standard computer based only on the laws of classical physics.

- If built, a large-scale quantum computer would find applications to fields as diverse as number theory, computational chemistry, and electronic design automation.

- These applications are driven by quantum algorithms: algorithms that run on a quantum computer.

This talk

In this talk, I'll discuss two famous quantum algorithms:

- Shor's algorithm for integer factorisation;
- Grover's algorithm for fast quantum search.

This talk

In this talk, I'll discuss two famous quantum algorithms:

- Shor's algorithm for integer factorisation;
- Grover's algorithm for fast quantum search.

And some more recent work:

- A quantum algorithm for accelerating backtracking algorithms;
- Applications to solving hard constraint satisfaction problems.

This talk

In this talk, I'll discuss two famous quantum algorithms:

- Shor's algorithm for integer factorisation;
- Grover's algorithm for fast quantum search.

And some more recent work:

- A quantum algorithm for accelerating backtracking algorithms;
- Applications to solving hard constraint satisfaction problems.

Disclaimer 1: The talk will focus on ideas and omit most/all technical details.

This talk

In this talk, I'll discuss two famous quantum algorithms:

- Shor's algorithm for integer factorisation;
- Grover's algorithm for fast quantum search.

And some more recent work:

- A quantum algorithm for accelerating backtracking algorithms;
- Applications to solving hard constraint satisfaction problems.

Disclaimer 1: The talk will focus on ideas and omit most/all technical details.

Disclaimer 2: The Quantum Algorithm Zoo (http://math.nist.gov/quantum/zoo/) cites 392 papers on quantum algorithms, so this is necessarily a partial view…

Hidden subgroup problems

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

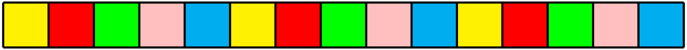
Let G be a group. Given access to a function $f : G \to X$ such that f is constant on the cosets of some subgroup $H \leqslant G$, and distinct on each coset, identify H.

Hidden subgroup problems

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

Let G be a group. Given 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.

A particularly interesting, and simple, case: $G = \mathbb{Z}_M$ for some integer M. This is the problem of determining the period of a periodic function which is one-to-one on each period:

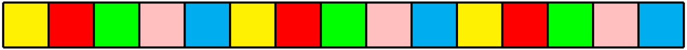


Hidden subgroup problems

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

Let G be a group. Given 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.

A particularly interesting, and simple, case: $G = \mathbb{Z}_M$ for some integer M. This is the problem of determining the period of a periodic function which is one-to-one on each period:



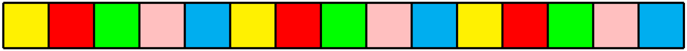
On a quantum computer, the HSP can be solved using $O(\log |G|)$ queries to f for all groups G [Ettinger et al. '04].

Classically, some groups require $\Omega(\sqrt{|G|})$ queries [Simon '97].

Periodicity in pictures

The quantum algorithm proceeds as follows:

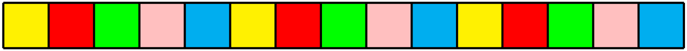
1. Query all function values in superposition:



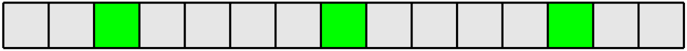
Periodicity in pictures

The quantum algorithm proceeds as follows:

1. Query all function values in superposition:



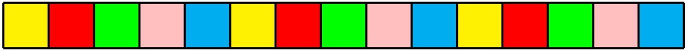
2. Measure the function value, receiving a random answer:



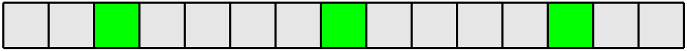
Periodicity in pictures

The quantum algorithm proceeds as follows:

1. Query all function values in superposition:



2. Measure the function value, receiving a random answer:

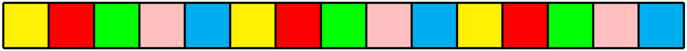


3. Apply the quantum Fourier transform. If the period was t we get a superposition with period M/t (ignoring phases):

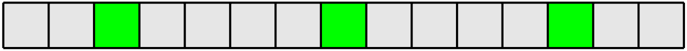
Periodicity in pictures

The quantum algorithm proceeds as follows:

1. Query all function values in superposition:



2. Measure the function value, receiving a random answer:



3. Apply the quantum Fourier transform. If the period was t we get a superposition with period M/t (ignoring phases):

4. Measure, getting a random outcome $r = kM/t$. Simplify the fraction r/M and output the denominator.

Periodicity and factorisation

Claim **[Miller '76]**

To find the prime factors of an integer N, it is sufficient to determine the period of the function $f(x) = a^x \bmod N$ for arbitrary integers a.

Periodicity and factorisation

Claim **[Miller '76]**

To find the prime factors of an integer N, it is sufficient to determine the period of the function $f(x) = a^x \bmod N$ for arbitrary integers a.

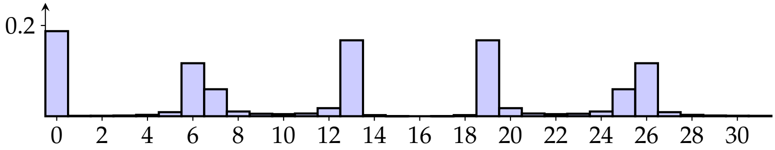
- For any a, this function is periodic over $\mathbb{Z}$. We truncate it to the integers mod M for some $M > N$.
- If the period t does not divide M, the distribution on measurement outcomes is peaked around integer multiples of M/t.

Periodicity and factorisation

Claim [Miller '76]

To find the prime factors of an integer N, it is sufficient to determine the period of the function $f(x) = a^x \bmod N$ for arbitrary integers a.

- For any a, this function is periodic over $\mathbb{Z}$. We truncate it to the integers mod M for some $M > N$.
- If the period t does not divide M, the distribution on measurement outcomes is peaked around integer multiples of M/t.
- e.g. if f has period 5 on domain size $M = 32$:



Hidden subgroup problems

Many cryptosystems and other problems reduce to the HSP, e.g.:

Problem	**Group**	**Complexity**	**Cryptosystem**
Integer factorisation	$\mathbb{Z}$	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
Shortest lattice vector	Dihedral grp	Subexp.[4]	NTRU, Ajtai-Dwork, ...
Graph isomorphism	Symmetric grp	Exponential	—

[1]Shor '97, [2]Proos et al. '03, [3]Hallgren '07, [4]Kuperberg '05, Regev '04

Hidden subgroup problems

Many cryptosystems and other problems reduce to the HSP, e.g.:

Problem	Group	Complexity	Cryptosystem
Integer factorisation	$\mathbb{Z}$	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
Shortest lattice vector	Dihedral grp	Subexp.[4]	NTRU, Ajtai-Dwork, ...
Graph isomorphism	Symmetric grp	Exponential	—

[1] Shor '97, [2] Proos et al. '03, [3] Hallgren '07, [4] Kuperberg '05, Regev '04

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

Open problem

For which groups G can the HSP be solved efficiently?

Unstructured search

Problem

Given access to a function $f : \{1, \ldots, n\} \to \{0, 1\}$ such that $f(x) = 1$ if and only if $x = x_0$ for some x_0, output x_0.

Unstructured search

Problem

Given access to a function $f : \{1, \ldots, n\} \to \{0, 1\}$ such that $f(x) = 1$ if and only if $x = x_0$ for some x_0, output x_0.

- Write $|\psi\rangle = \frac{1}{\sqrt{n}} \sum_{i=1}^{n} |i\rangle = (1, 1, \ldots, 1)^T / \sqrt{n}$.

Unstructured search

Problem

Given access to a function $f : \{1, \ldots, n\} \to \{0, 1\}$ such that $f(x) = 1$ if and only if $x = x_0$ for some x_0, output x_0.

- Write $|\psi\rangle = \frac{1}{\sqrt{n}} \sum_{i=1}^{n} |i\rangle = (1, 1, \ldots, 1)^T / \sqrt{n}$.
- Grover's algorithm: starting with $|\psi\rangle$, alternately reflect about $\left|x_0^{\perp}\right\rangle$ and about $|\psi\rangle$:

$$R_{|\psi\rangle} R_{|x_0^{\perp}\rangle} R_{|\psi\rangle} R_{|x_0^{\perp}\rangle} \ldots R_{|\psi\rangle} R_{|x_0^{\perp}\rangle},$$

where for a state $|\chi\rangle$, $R_{|\chi\rangle} = 2|\chi\rangle\langle\chi| - I$.

Unstructured search

Problem

Given access to a function $f : \{1, \ldots, n\} \to \{0, 1\}$ such that $f(x) = 1$ if and only if $x = x_0$ for some x_0, output x_0.

- Write $|\psi\rangle = \frac{1}{\sqrt{n}} \sum_{i=1}^{n} |i\rangle = (1, 1, \ldots, 1)^T/\sqrt{n}$.
- Grover's algorithm: starting with $|\psi\rangle$, alternately reflect about $|x_0^\perp\rangle$ and about $|\psi\rangle$:

$$R_{|\psi\rangle} R_{|x_0^\perp\rangle} R_{|\psi\rangle} R_{|x_0^\perp\rangle} \ldots R_{|\psi\rangle} R_{|x_0^\perp\rangle},$$

  where for a state $|\chi\rangle$, $R_{|\chi\rangle} = 2|\chi\rangle\langle\chi| - I$.
- $R_{|x_0^\perp\rangle}$ can be implemented by mapping $|i\rangle \mapsto (-1)^{f(i)}|i\rangle$, which can be done using one evaluation of f.

Grover's algorithm

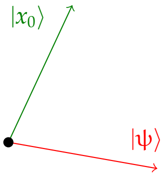
- The composition of two reflections is a rotation: $R_{|\psi\rangle} R_{|x_0^\perp\rangle}$ rotates by angle 2ϕ from $|\psi\rangle$ to $|x_0\rangle$, where

$$\sin\phi = \langle\psi|x_0\rangle = 1/\sqrt{n}.$$

Grover's algorithm

- The composition of two reflections is a rotation: $R_{|\psi\rangle} R_{|x_0^\perp\rangle}$ rotates by angle 2ϕ from $|\psi\rangle$ to $|x_0\rangle$, where

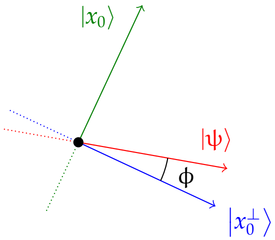
$$\sin\phi = \langle\psi|x_0\rangle = 1/\sqrt{n}.$$



Grover's algorithm

- The composition of two reflections is a rotation: $R_{|\psi\rangle} R_{|x_0^\perp\rangle}$ rotates by angle 2ϕ from $|\psi\rangle$ to $|x_0\rangle$, where

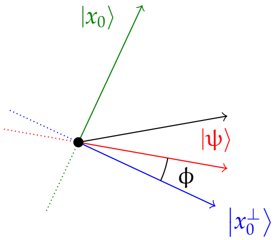
$$\sin \phi = \langle \psi | x_0 \rangle = 1/\sqrt{n}.$$



Grover's algorithm

- The composition of two reflections is a rotation: $R_{|\psi\rangle} R_{|x_0^\perp\rangle}$ rotates by angle 2ϕ from $|\psi\rangle$ to $|x_0\rangle$, where

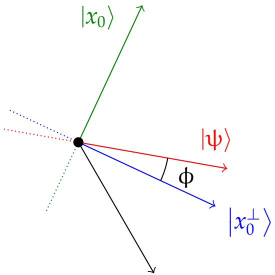
$$\sin\phi = \langle\psi|x_0\rangle = 1/\sqrt{n}.$$



Grover's algorithm

- The composition of two reflections is a rotation: $R_{|\psi\rangle} R_{|x_0^\perp\rangle}$ rotates by angle 2ϕ from $|\psi\rangle$ to $|x_0\rangle$, where

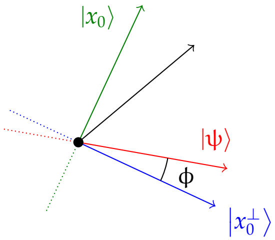
$$\sin\phi = \langle\psi|x_0\rangle = 1/\sqrt{n}.$$



Grover's algorithm

- The composition of two reflections is a rotation: $R_{|\psi\rangle} R_{|x_0^\perp\rangle}$ rotates by angle 2ϕ from $|\psi\rangle$ to $|x_0\rangle$, where

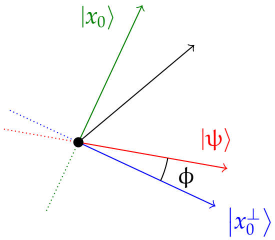
$$\sin\phi = \langle\psi|x_0\rangle = 1/\sqrt{n}.$$



Grover's algorithm

- The composition of two reflections is a rotation: $R_{|\psi\rangle} R_{|x_0^\perp\rangle}$ rotates by angle 2ϕ from $|\psi\rangle$ to $|x_0\rangle$, where

$$\sin\phi = \langle\psi|x_0\rangle = 1/\sqrt{n}.$$



- Thus the algorithm uses f $O(\sqrt{n})$ times to reach $|x_0\rangle$.

Applications of Grover's algorithm

This can be used to obtain many speedups over classical algorithms, e.g.:

- Finding the minimum of n numbers in $O(\sqrt{n})$ time [Dürr and Høyer '96]

Applications of Grover's algorithm

This can be used to obtain many speedups over classical algorithms, e.g.:

- Finding the minimum of n numbers in $O(\sqrt{n})$ time [Dürr and Høyer '96]
- Determining connectivity of an n-vertex graph in $O(n^{3/2})$ time [Dürr et al '04]

Applications of Grover's algorithm

This can be used to obtain many speedups over classical algorithms, e.g.:

- Finding the minimum of n numbers in $O(\sqrt{n})$ time [Dürr and Høyer '96]
- Determining connectivity of an n-vertex graph in $O(n^{3/2})$ time [Dürr et al '04]
- Finding a collision in a 2-1 function $f : [n] \to [n]$ in $O(n^{1/3})$ time [Brassard et al '98]

Applications of Grover's algorithm

This can be used to obtain many speedups over classical algorithms, e.g.:

- Finding the minimum of n numbers in $O(\sqrt{n})$ time [Dürr and Høyer '96]
- Determining connectivity of an n-vertex graph in $O(n^{3/2})$ time [Dürr et al '04]
- Finding a collision in a 2-1 function $f : [n] \to [n]$ in $O(n^{1/3})$ time [Brassard et al '98]
- Finding a maximal matching in a bipartite graph with V vertices and E edges in $O(V\sqrt{E}\log V)$ time [Ambainis and Špalek '05]

Applications of Grover's algorithm

This can be used to obtain many speedups over classical algorithms, e.g.:

- Finding the minimum of n numbers in $O(\sqrt{n})$ time [Dürr and Høyer '96]
- Determining connectivity of an n-vertex graph in $O(n^{3/2})$ time [Dürr et al '04]
- Finding a collision in a 2-1 function $f : [n] \to [n]$ in $O(n^{1/3})$ time [Brassard et al '98]
- Finding a maximal matching in a bipartite graph with V vertices and E edges in $O(V\sqrt{E}\log V)$ time [Ambainis and Špalek '05]
- Approximating the ℓ_1 distance between probability distributions on n elements in $O(\sqrt{n})$ time [Bravyi et al '09]
- ...

Accelerating other algorithms

Grover's algorithm accelerates a particular classical algorithm: unstructured search.

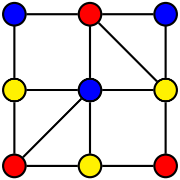
Can we speed up other algorithms too?

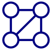
Accelerating other algorithms

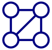
Grover's algorithm accelerates a particular classical algorithm: unstructured search.

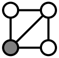
Can we speed up other algorithms too?

Another case where we can achieve a speedup: backtracking ("trial and error").

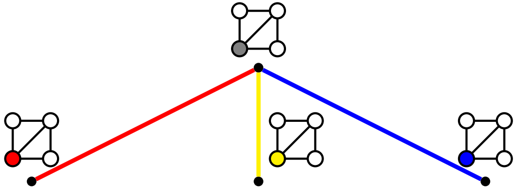


Colouring  by backtracking

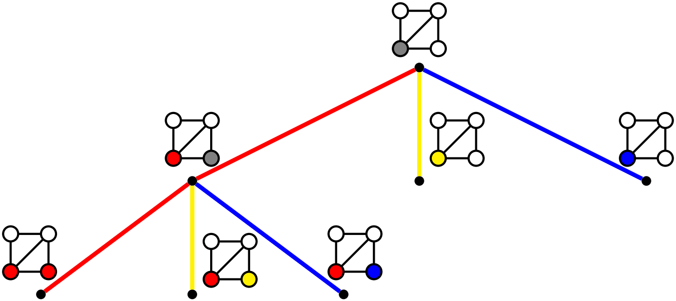
Colouring  by backtracking

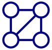


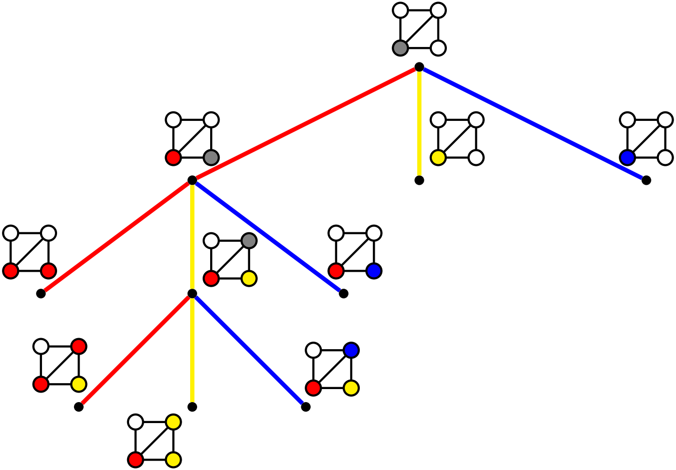
Colouring by backtracking

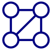


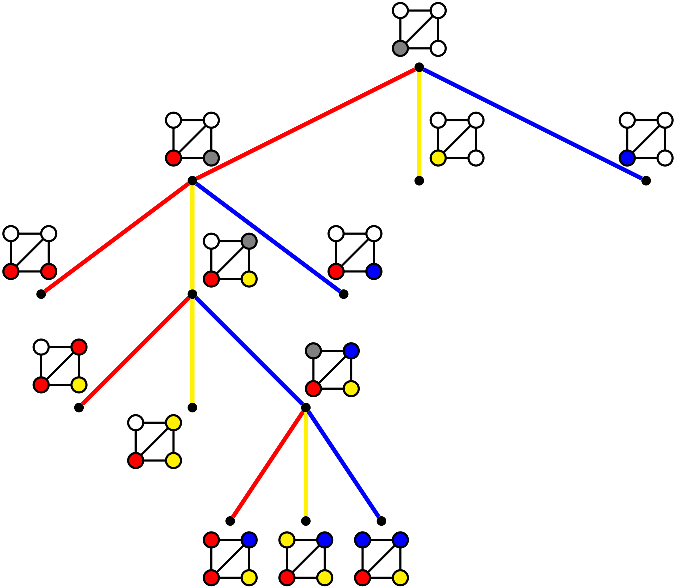
Colouring by backtracking



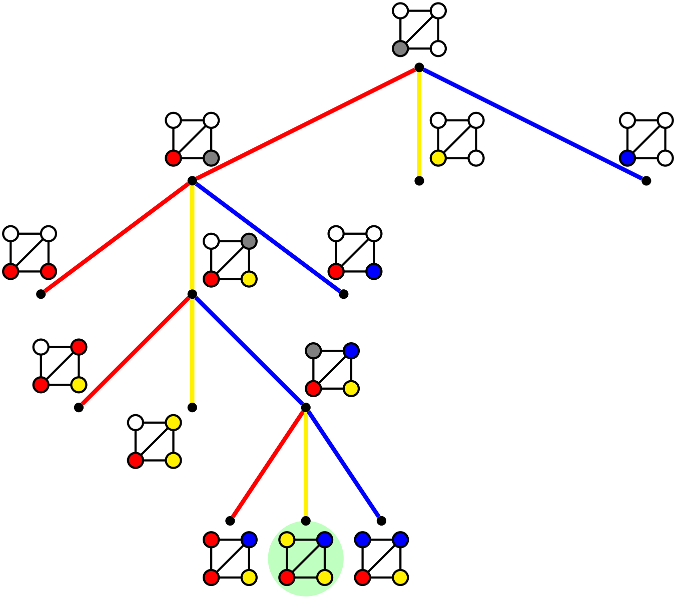
Colouring  by backtracking



Colouring  by backtracking



Colouring by backtracking



Quantum backtracking algorithm

Theorem (informal) [AM '15]

Assume there exists a classical algorithm which solves a constraint satisfaction problem on n variables via backtracking, with a backtracking tree containing T nodes.

Then there is a quantum algorithm which solves the same problem in time $O(\sqrt{T}\,\mathsf{poly}(n))$.

Quantum backtracking algorithm

Theorem (informal) [AM '15]

Assume there exists a classical algorithm which solves a constraint satisfaction problem on n variables via backtracking, with a backtracking tree containing T nodes.

Then there is a quantum algorithm which solves the same problem in time $O(\sqrt{T}\,\mathsf{poly}(n))$.

- We usually think of T as being exponentially large in n. In this regime, this is a near-quadratic separation.

How the backtracking algorithm works

The algorithm is based on the use of a quantum walk to search for a marked vertex in the backtracking tree.

How the backtracking algorithm works

The algorithm is based on the use of a quantum walk to search for a marked vertex in the backtracking tree.

- We can view the algorithm as being like Grover's algorithm – but searching in a way that respects the structure of the tree.

How the backtracking algorithm works

The algorithm is based on the use of a quantum walk to search for a marked vertex in the backtracking tree.

- We can view the algorithm as being like Grover's algorithm – but searching in a way that respects the structure of the tree.
- We replace the $R_{|\psi\rangle}$ operations with operations of the form $R_{|\psi_x\rangle}$, for a node x, where $|\psi_x\rangle \propto |x\rangle + \sum_{y \text{ child of } x} |y\rangle$.

How the backtracking algorithm works

The algorithm is based on the use of a quantum walk to search for a marked vertex in the backtracking tree.

- We can view the algorithm as being like Grover's algorithm – but searching in a way that respects the structure of the tree.
- We replace the $R_{|\psi\rangle}$ operations with operations of the form $R_{|\psi_x\rangle}$, for a node x, where $|\psi_x\rangle \propto |x\rangle + \sum_{y \text{ child of } x} |y\rangle$.
- The state $|\psi_r\rangle$ corresponding to the root r is different: $|\psi_r\rangle \propto |r\rangle + \sqrt{n} \sum_{y \text{ child of } r} |y\rangle$.

Quantum walk in a tree

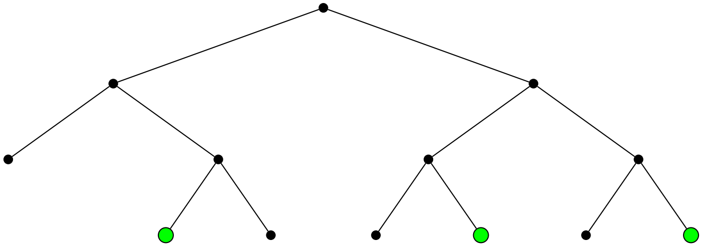
Let A and B be the sets of vertices an even and odd distance from the root, respectively.

Then a step of the walk consists of applying the operator $R_B R_A$, where $R_A = \bigoplus_{x \in A} R_{|\psi_x\rangle}$ and $R_B = -|r\rangle\langle r| + \bigoplus_{x \in B} R_{|\psi_x\rangle}$.

Quantum walk in a tree

Let A and B be the sets of vertices an even and odd distance from the root, respectively.

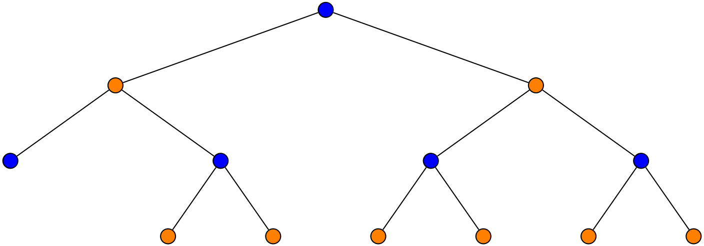
Then a step of the walk consists of applying the operator $R_B R_A$, where $R_A = \bigoplus_{x \in A} R_{|\psi_x\rangle}$ and $R_B = -|r\rangle\langle r| + \bigoplus_{x \in B} R_{|\psi_x\rangle}$.



Quantum walk in a tree

Let A and B be the sets of vertices an even and odd distance from the root, respectively.

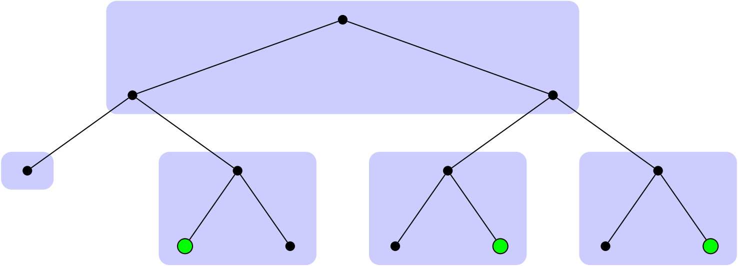
Then a step of the walk consists of applying the operator $R_B R_A$, where $R_A = \bigoplus_{x \in A} R_{|\psi_x\rangle}$ and $R_B = -|r\rangle\langle r| + \bigoplus_{x \in B} R_{|\psi_x\rangle}$.



Quantum walk in a tree

Let A and B be the sets of vertices an even and odd distance from the root, respectively.

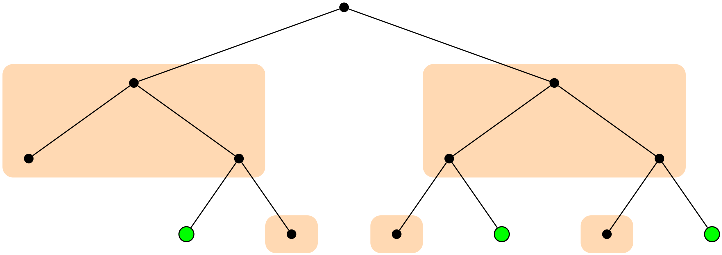
Then a step of the walk consists of applying the operator $R_B R_A$, where $R_A = \bigoplus_{x \in A} R_{|\psi_x\rangle}$ and $R_B = -|r\rangle\langle r| + \bigoplus_{x \in B} R_{|\psi_x\rangle}$.



Quantum walk in a tree

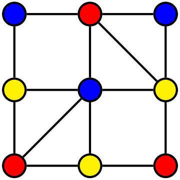
Let A and B be the sets of vertices an even and odd distance from the root, respectively.

Then a step of the walk consists of applying the operator $R_B R_A$, where $R_A = \bigoplus_{x \in A} R_{|\psi_x\rangle}$ and $R_B = -|r\rangle\langle r| + \bigoplus_{x \in B} R_{|\psi_x\rangle}$.



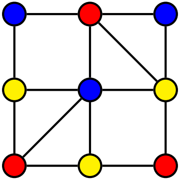
Two applications

We applied Grover's algorithm and backtracking to two important problems: graph colouring and boolean satisfiability [Campbell, Khurana, AM '18].



Two applications

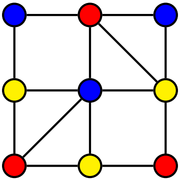
We applied Grover's algorithm and backtracking to two important problems: graph colouring and boolean satisfiability [Campbell, Khurana, AM '18].



$$(x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee x_2 \vee x_4) \wedge (x_2 \vee \neg x_3 \vee \neg x_4)$$

Two applications

We applied Grover's algorithm and backtracking to two important problems: graph colouring and boolean satisfiability [Campbell, Khurana, AM '18].



$$(x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee x_2 \vee x_4) \wedge (x_2 \vee \neg x_3 \vee \neg x_4)$$

Each problem is NP-complete and has a huge number of direct applications:

- SAT: verification of electronic circuits; planning; computer-aided mathematical proofs; ...
- Colouring: register allocation; scheduling; frequency assignment problems; ...

Summary of results: good and bad news

- In the most optimistic hardware parameter regime we considered, we could see speedup factors of $> 10^5$ (compared with a standard desktop PC) for k-SAT (via Grover's algorithm) and $> 10^4$ for graph colouring (via backtracking) for instances that can be solved in 1 day.

Summary of results: good and bad news

- In the most optimistic hardware parameter regime we considered, we could see speedup factors of $> 10^5$ (compared with a standard desktop PC) for k-SAT (via Grover's algorithm) and $> 10^4$ for graph colouring (via backtracking) for instances that can be solved in 1 day.
- This speedup gets substantially smaller when considering hardware available today (e.g. $\sim 10^3$ for k-SAT).

Summary of results: good and bad news

- In the most optimistic hardware parameter regime we considered, we could see speedup factors of $> 10^5$ (compared with a standard desktop PC) for k-SAT (via Grover's algorithm) and $> 10^4$ for graph colouring (via backtracking) for instances that can be solved in 1 day.
- This speedup gets substantially smaller when considering hardware available today (e.g. $\sim 10^3$ for k-SAT).
- If we additionally take into account the cost of classical error-correction processing, this speedup essentially disappears.

Summary of results: good and bad news

- In the most optimistic hardware parameter regime we considered, we could see speedup factors of $> 10^5$ (compared with a standard desktop PC) for k-SAT (via Grover's algorithm) and $> 10^4$ for graph colouring (via backtracking) for instances that can be solved in 1 day.
- This speedup gets substantially smaller when considering hardware available today (e.g. $\sim 10^3$ for k-SAT).
- If we additionally take into account the cost of classical error-correction processing, this speedup essentially disappears.
- The number of physical qubits used is very large (e.g. $> 10^{12}$), almost all of which are used for fault-tolerance.

Summary of results: good and bad news

- In the most optimistic hardware parameter regime we considered, we could see speedup factors of $> 10^5$ (compared with a standard desktop PC) for k-SAT (via Grover's algorithm) and $> 10^4$ for graph colouring (via backtracking) for instances that can be solved in 1 day.
- This speedup gets substantially smaller when considering hardware available today (e.g. $\sim 10^3$ for k-SAT).
- If we additionally take into account the cost of classical error-correction processing, this speedup essentially disappears.
- The number of physical qubits used is very large (e.g. $> 10^{12}$), almost all of which are used for fault-tolerance.
- This strongly motivates the design of improved fault-tolerance techniques!

Outlook

- I've described only a few quantum algorithms: there are many others known.

Outlook

- I've described only a few quantum algorithms: there are many others known.
- Even in 2018, there are still quantum speedups to be found for very basic problems…

Outlook

- I've described only a few quantum algorithms: there are many others known.
- Even in 2018, there are still quantum speedups to be found for very basic problems…
- …and another challenge is to make the algorithms we know truly practical.

Outlook

- I've described only a few quantum algorithms: there are many others known.
- Even in 2018, there are still quantum speedups to be found for very basic problems…
- …and another challenge is to make the algorithms we know truly practical.

Advert

A workshop on Quantum Computing Theory in Practice will take place in Bristol from 8–10 April 2019.

www.bristolmathsresearch.org/meeting/qctip/

Thanks!

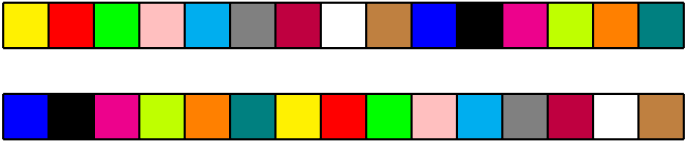
Some further reading:

- "Quantum algorithms: an overview" [AM 1511.04206]
- "Quantum algorithms for algebraic problems" [Childs and van Dam 0812.0380]
- "New developments in quantum algorithms" [Ambainis 1006.4014]

The dihedral hidden subgroup problem

The dihedral HSP turns out to be equivalent to a hidden shift problem:

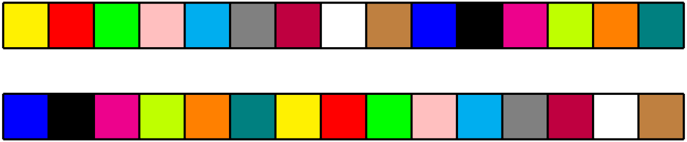
- Given two injective functions $f, g : \mathbb{Z}_N \to X$ such that $g(x) = f(x+s)$ for some $s \in \mathbb{Z}_N$, determine s.



The dihedral hidden subgroup problem

The dihedral HSP turns out to be equivalent to a hidden shift problem:

- Given two injective functions $f, g : \mathbb{Z}_N \to X$ such that $g(x) = f(x+s)$ for some $s \in \mathbb{Z}_N$, determine s.

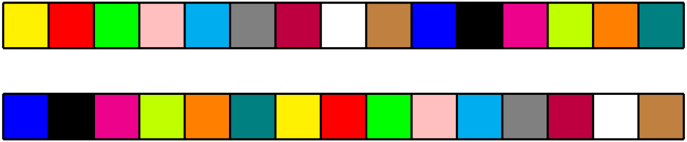


Implies applications to pattern matching problems in strings.

The dihedral hidden subgroup problem

The dihedral HSP turns out to be equivalent to a hidden shift problem:

- Given two injective functions $f, g : \mathbb{Z}_N \to X$ such that $g(x) = f(x+s)$ for some $s \in \mathbb{Z}_N$, determine s.



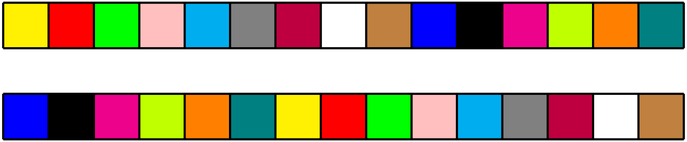
Implies applications to pattern matching problems in strings.

- The best known algorithm for the dihedral HSP uses time $2^{O(\sqrt{\log N})}$ [Kuperberg '05] …can this be improved?

The dihedral hidden subgroup problem

The dihedral HSP turns out to be equivalent to a hidden shift problem:

- Given two injective functions $f, g : \mathbb{Z}_N \to X$ such that $g(x) = f(x+s)$ for some $s \in \mathbb{Z}_N$, determine s.



Implies applications to pattern matching problems in strings.

- The best known algorithm for the dihedral HSP uses time $2^{O(\sqrt{\log N})}$ [Kuperberg '05] …can this be improved?
- A $\mathrm{poly}(\log N)$-time algorithm would give an efficient quantum algorithm for the shortest vector problem in lattices [Regev '04].