Quantum algorithms for PDEs

Ashley Montanaro

Talk based on joint work with Noah Linden and Changpeng Shao

School of Mathematics, University of Bristol

27 November 2019

University of BRISTOL

EPSRC
Engineering and Physical Sciences Research Council

European Research Council
Established by the European Commission

Quantum computers

Quantum computers are designed to do things that classical computers cannot. But to achieve a quantum speedup requires a quantum algorithm.

Quantum computers

Quantum computers are designed to do things that classical computers cannot. But to achieve a quantum speedup requires a quantum algorithm.

Most quantum algorithms can be divided into 5 categories:

Algorithm	**Speedup**	**Example**
Simulation of quantum systems	Exponential	Lloyd
Breaking cryptographic codes	Exponential	Shor
Optimisation / combinatorial search	Square-root	Grover
High-dimensional linear algebra	Exponential?	HHL
Quantum heuristics	Unknown	QAOA

Quantum computers

Quantum computers are designed to do things that classical computers cannot. But to achieve a quantum speedup requires a quantum algorithm.

Most quantum algorithms can be divided into 5 categories:

Algorithm	Speedup	Example
Simulation of quantum systems	Exponential	Lloyd
Breaking cryptographic codes	Exponential	Shor
Optimisation / combinatorial search	Square-root	Grover
High-dimensional linear algebra	Exponential?	HHL
Quantum heuristics	Unknown	QAOA

The Quantum Algorithm Zoo currently lists 404 papers on quantum algorithms.

Solving PDEs with a quantum computer

One plausible problem domain where quantum computers could be applied is solving PDEs:

$$\frac{\partial u}{\partial t} = \alpha \left(\frac{\partial^2 u}{\partial x_1^2} + \cdots + \frac{\partial^2 u}{\partial x_d^2} \right)$$

Solving PDEs with a quantum computer

One plausible problem domain where quantum computers could be applied is solving PDEs:

$$\frac{\partial u}{\partial t} = \alpha \left(\frac{\partial^2 u}{\partial x_1^2} + \cdots + \frac{\partial^2 u}{\partial x_d^2} \right)$$

One reason this seems plausible is that:

- PDEs are often solved by discretisation to produce a system of linear equations;
- Quantum computers can have an exponential advantage over classical computers for linear equation problems.

Solving PDEs with a quantum computer

One plausible problem domain where quantum computers could be applied is solving PDEs:

$$\frac{\partial u}{\partial t} = \alpha \left(\frac{\partial^2 u}{\partial x_1^2} + \cdots + \frac{\partial^2 u}{\partial x_d^2} \right)$$

One reason this seems plausible is that:

- PDEs are often solved by discretisation to produce a system of linear equations;
- Quantum computers can have an exponential advantage over classical computers for linear equation problems.

Some indications there could be an advantage:
e.g. [Leyton+Osborne 0812.4423] [Berry 1010.2745] [Cao et al 1207.2485] [Clader et al 1301.2340]

“Solving” linear equations

A basic task in mathematics and engineering:

Solving linear equations

Given access to a d-sparse $N \times N$ matrix A, and $b \in \mathbb{R}^N$, output x such that $Ax = b$.

“Solving” linear equations

A basic task in mathematics and engineering:

Solving linear equations

Given access to a d-sparse $N \times N$ matrix A, and $b \in \mathbb{R}^N$, output x such that $Ax = b$.

One “quantum” way of thinking about the problem:

“Solving” linear equations

Given the ability to produce the quantum state $|b\rangle = \sum_{i=1}^{N} b_i |i\rangle$, and access to A as above, produce the state $|x\rangle = \sum_{i=1}^{N} x_i |i\rangle$.

“Solving” linear equations

A basic task in mathematics and engineering:

Solving linear equations

Given access to a d-sparse $N \times N$ matrix A, and $b \in \mathbb{R}^N$, output x such that $Ax = b$.

One “quantum” way of thinking about the problem:

“Solving” linear equations

Given the ability to produce the quantum state $|b\rangle = \sum_{i=1}^{N} b_i |i\rangle$, and access to A as above, produce the state $|x\rangle = \sum_{i=1}^{N} x_i |i\rangle$.

Theorem: If A has condition number κ ($= \|A^{-1}\| \|A\|$), $|x\rangle$ can be approximately produced in time $\mathsf{poly}(\log N, d, \kappa)$ [Harrow et al 0811.3171]

Some challenges for the quantum algorithm

- How to produce the initial state $|x\rangle$?

Some challenges for the quantum algorithm

- How to produce the initial state $|x\rangle$?
- How to get information out of the final state $|b\rangle$?

Some challenges for the quantum algorithm

- How to produce the initial state $|x\rangle$?
- How to get information out of the final state $|b\rangle$?
- How to access the matrix A?

Some challenges for the quantum algorithm

- How to produce the initial state $|x\rangle$?
- How to get information out of the final state $|b\rangle$?
- How to access the matrix A?
- How to bound the condition number κ?

Some challenges for the quantum algorithm

- How to produce the initial state $|x\rangle$?
- How to get information out of the final state $|b\rangle$?
- How to access the matrix A?
- How to bound the condition number κ?
- How to bound the level of accuracy achieved?

Some challenges for the quantum algorithm

- How to produce the initial state $|x\rangle$?
- How to get information out of the final state $|b\rangle$?
- How to access the matrix A?
- How to bound the condition number κ?
- How to bound the level of accuracy achieved?

Taking these into account, and making some assumptions about the problem solved, in [AM+Pallister 1512.05903] it was shown that using the HHL algorithm to solve PDEs discretised with the finite element method (FEM) can achieve at most a polynomial speedup (in fixed "spatial" dimension).

This talk

Today I will discuss recent work on quantum algorithms solving the heat equation in d dimensions in the region $[0, L]^d \times [0, T]$ with periodic spatial boundary conditions:

$$\frac{\partial u}{\partial t} = \alpha \left(\frac{\partial^2 u}{\partial x_1^2} + \cdots + \frac{\partial^2 u}{\partial x_d^2} \right)$$

Problem

Let $u(\mathbf{x}, t)$ be a solution to the heat equation. Given an initial condition $u(\mathbf{x}, 0) = u_0(\mathbf{x})$, a time t, and a subset $S \subseteq [0, L]^d$, compute $\int_S u(\mathbf{x}, t) d\mathbf{x} \pm \epsilon$.

Will quantum algorithms outperform classical ones for this problem?

Classical and quantum results

We compared various classical and quantum methods for solving the heat equation:

Method	$\mathbf{d = 1}$	$\mathbf{d = 2}$	$\mathbf{d = 3}$	$\mathbf{d > 3}$
* Classical linear equations	$\widetilde{O}(\epsilon^{-2})$	$\widetilde{O}(\epsilon^{-2.5})$	$\widetilde{O}(\epsilon^{-3})$	$\widetilde{O}(\epsilon^{-d/2-1.5})$
* Classical time-stepping	$\widetilde{O}(\epsilon^{-1.5})$	$\widetilde{O}(\epsilon^{-2})$	$\widetilde{O}(\epsilon^{-2.5})$	$\widetilde{O}(\epsilon^{-d/2-1})$
* Classical FFT	$\widetilde{O}(\epsilon^{-0.5})$	$\widetilde{O}(\epsilon^{-1})$	$\widetilde{O}(\epsilon^{-1.5})$	$\widetilde{O}(\epsilon^{-d/2})$
Classical random walk	$\widetilde{O}(\epsilon^{-3})$	$\widetilde{O}(\epsilon^{-3})$	$\widetilde{O}(\epsilon^{-3})$	$\widetilde{O}(\epsilon^{-3})$
HHL	$\widetilde{O}(\epsilon^{-2.5})$	$\widetilde{O}(\epsilon^{-2.5})$	$\widetilde{O}(\epsilon^{-2.75})$	$\widetilde{O}(\epsilon^{-d/4-2})$
Diagonalisation	$\widetilde{O}(\epsilon^{-1.25})$	$\widetilde{O}(\epsilon^{-1.5})$	$\widetilde{O}(\epsilon^{-1.75})$	$\widetilde{O}(\epsilon^{-d/4-1})$
Coherent rw acceleration	$\widetilde{O}(\epsilon^{-1.75})$	$\widetilde{O}(\epsilon^{-2})$	$\widetilde{O}(\epsilon^{-2.25})$	$\widetilde{O}(\epsilon^{-d/4-1.5})$
Rw amplitude estimation	$\widetilde{O}(\epsilon^{-2})$	$\widetilde{O}(\epsilon^{-2})$	$\widetilde{O}(\epsilon^{-2})$	$\widetilde{O}(\epsilon^{-2})$

Only dependence on the accuracy ϵ is shown.

Starred methods use space $\mathrm{poly}(1/\epsilon)$, others use space $\mathrm{poly}(\log 1/\epsilon)$.

Methods

All of the classical and quantum algorithms are based on discretising space and time via the finite difference method (FTCS):

$$\frac{du}{dx} = \frac{u(x+h) - u(x)}{h} + O(h)$$

$$\frac{d^2u}{dx^2} = \frac{u(x+h) + u(x-h) - 2u(x)}{h^2} + O(h^2)$$

Methods

All of the classical and quantum algorithms are based on discretising space and time via the finite difference method (FTCS):

$$\frac{du}{dx} = \frac{u(x+h) - u(x)}{h} + O(h)$$

$$\frac{d^2u}{dx^2} = \frac{u(x+h) + u(x-h) - 2u(x)}{h^2} + O(h^2)$$

Leads to the set of linear constraints

$$\frac{\widetilde{u}(\mathbf{x},t+\Delta t) - \widetilde{u}(\mathbf{x},t)}{\Delta t} = \frac{\alpha}{\Delta x^2} \sum_{i=1}^{d} \widetilde{u}(\ldots, x_i + \Delta x, \ldots, t) + \widetilde{u}(\ldots, x_i - \Delta x, \ldots, t) - 2\widetilde{u}(\mathbf{x}, t)$$

Methods

All of the classical and quantum algorithms are based on discretising space and time via the finite difference method (FTCS):

$$\frac{du}{dx} = \frac{u(x+h) - u(x)}{h} + O(h)$$

$$\frac{d^2u}{dx^2} = \frac{u(x+h) + u(x-h) - 2u(x)}{h^2} + O(h^2)$$

Leads to the set of linear constraints

$$\frac{\widetilde{u}(\mathbf{x},t+\Delta t) - \widetilde{u}(\mathbf{x},t)}{\Delta t} = \frac{\alpha}{\Delta x^2} \sum_{i=1}^{d} \widetilde{u}(\dots, x_i + \Delta x, \dots, t) + \widetilde{u}(\dots, x_i - \Delta x, \dots, t) - 2\widetilde{u}(\mathbf{x}, t)$$

To achieve final accuracy ϵ, can take $\Delta t = O(\epsilon)$, $\Delta x = O(\sqrt{\epsilon})$.

Linear equation methods

We have a system of $N = O(\epsilon^{-d/2-1})$ linear equations to solve.

Linear equation methods

We have a system of $N = O(\epsilon^{-d/2-1})$ linear equations to solve.

- Condition number: $\kappa = O(\epsilon^{-1})$.

Linear equation methods

We have a system of $N = O(\epsilon^{-d/2-1})$ linear equations to solve.

- Condition number: $\kappa = O(\epsilon^{-1})$.
- Classical complexity: $\widetilde{O}(\sqrt{\kappa}N) = O(\epsilon^{-d/2-1.5})$.

Linear equation methods

We have a system of $N = O(\epsilon^{-d/2-1})$ linear equations to solve.

- Condition number: $\kappa = O(\epsilon^{-1})$.
- Classical complexity: $\widetilde{O}(\sqrt{\kappa}N) = O(\epsilon^{-d/2-1.5})$.
- Quantum complexity: $\widetilde{O}(\kappa) = O(\epsilon^{-1})\ldots$

Linear equation methods

We have a system of $N = O(\epsilon^{-d/2-1})$ linear equations to solve.

- Condition number: $\kappa = O(\epsilon^{-1})$.
- Classical complexity: $\widetilde{O}(\sqrt{\kappa}N) = O(\epsilon^{-d/2-1.5})$.
- Quantum complexity: $\widetilde{O}(\kappa) = O(\epsilon^{-1})\dots$

$\dots$but this algorithm only produces a quantum state which is equal to $\widetilde{u}/\|\widetilde{u}\|_2$.

Linear equation methods

We have a system of $N = O(\epsilon^{-d/2-1})$ linear equations to solve.

- Condition number: $\kappa = O(\epsilon^{-1})$.
- Classical complexity: $\widetilde{O}(\sqrt{\kappa}N) = O(\epsilon^{-d/2-1.5})$.
- Quantum complexity: $\widetilde{O}(\kappa) = O(\epsilon^{-1})\dots$

...but this algorithm only produces a quantum state which is equal to $\widetilde{u}/\|\widetilde{u}\|_2$.

To approximate $\int_S u(\mathbf{x}, t)d\mathbf{x}$, we need to know $\|\widetilde{u}\|_2$; achieving high enough accuracy takes time $\widetilde{O}(\epsilon^{-d/4-2})$.

Other classical methods

We can rewrite the discretised heat equation as

$$\widetilde{u}(\mathbf{x}, t+\Delta t) = \left(1 - \frac{2d\alpha\Delta t}{\Delta x^2}\widetilde{u}(\mathbf{x}, t)\right) + \frac{\alpha\Delta t}{\Delta x^2}\sum_{i=1}^{d}\widetilde{u}(\dots, x_i+\Delta x, \dots, t) + \widetilde{u}(\dots, x_i-\Delta x, \dots, t).$$

Other classical methods

We can rewrite the discretised heat equation as

$$\widetilde{u}(\mathbf{x}, t+\Delta t) = \left(1 - \tfrac{2d\alpha\Delta t}{\Delta x^2}\widetilde{u}(\mathbf{x}, t)\right) + \tfrac{\alpha\Delta t}{\Delta x^2}\textstyle\sum_{i=1}^{d}\widetilde{u}(\dots, x_i+\Delta x, \dots, t) + \widetilde{u}(\dots, x_i - \Delta x, \dots, t).$$

- We can simply step forward in time using sparse matrix multiplication: time $\widetilde{O}(\epsilon^{-d/2} \cdot \epsilon^{-1})$.

Other classical methods

We can rewrite the discretised heat equation as

$$\widetilde{u}(\mathbf{x}, t + \Delta t) = \left(1 - \frac{2d\alpha\Delta t}{\Delta x^2}\widetilde{u}(\mathbf{x}, t)\right) + \frac{\alpha\Delta t}{\Delta x^2}\sum_{i=1}^{d}\widetilde{u}(\dots, x_i + \Delta x, \dots, t) + \widetilde{u}(\dots, x_i - \Delta x, \dots, t).$$

- We can simply step forward in time using sparse matrix multiplication: time $\widetilde{O}(\epsilon^{-d/2} \cdot \epsilon^{-1})$.
- We can diagonalise the discretised linear system with the FFT: time $\widetilde{O}(\epsilon^{-d/2})$.

Other classical methods

We can rewrite the discretised heat equation as

$$\widetilde{u}(\mathbf{x}, t+\Delta t) = \left(1 - \frac{2d\alpha\Delta t}{\Delta x^2}\widetilde{u}(\mathbf{x}, t)\right) + \frac{\alpha\Delta t}{\Delta x^2}\sum_{i=1}^{d}\widetilde{u}(\dots, x_i+\Delta x, \dots, t) + \widetilde{u}(\dots, x_i-\Delta x, \dots, t).$$

- We can simply step forward in time using sparse matrix multiplication: time $\widetilde{O}(\epsilon^{-d/2}\cdot\epsilon^{-1})$.
- We can diagonalise the discretised linear system with the FFT: time $\widetilde{O}(\epsilon^{-d/2})$.
- We can observe that this corresponds to a random walk and sample from the output distribution in time $\widetilde{O}(\epsilon^{-1})$.

Other classical methods

We can rewrite the discretised heat equation as

$$\widetilde{u}(\mathbf{x}, t+\Delta t) = \left(1 - \frac{2d\alpha\Delta t}{\Delta x^2}\widetilde{u}(\mathbf{x}, t)\right) + \frac{\alpha\Delta t}{\Delta x^2}\sum_{i=1}^{d}\widetilde{u}(\dots, x_i+\Delta x, \dots, t) + \widetilde{u}(\dots, x_i-\Delta x, \dots, t).$$

- We can simply step forward in time using sparse matrix multiplication: time $\widetilde{O}(\epsilon^{-d/2}\cdot\epsilon^{-1})$.
- We can diagonalise the discretised linear system with the FFT: time $\widetilde{O}(\epsilon^{-d/2})$.
- We can observe that this corresponds to a random walk and sample from the output distribution in time $\widetilde{O}(\epsilon^{-1})$.
- Gives an algorithm for approximating $\int_S u(\mathbf{x}, t)d\mathbf{x} \pm \epsilon$ in time $\widetilde{O}(\epsilon^{-1}\cdot\epsilon^{-2})$.

Other quantum methods

Analogous to the classical ones:

Other quantum methods

Analogous to the classical ones:

- We can start with an initial state $|u_0\rangle$ and try to produce the state $|u_t\rangle$ corresponding to time t.

Other quantum methods

Analogous to the classical ones:

- We can start with an initial state $|u_0\rangle$ and try to produce the state $|u_t\rangle$ corresponding to time t.
- We can use a coherent acceleration method for random walks due to [Apers+Sarlette '18], [Gilyen et al '18] which gives a square-root improvement in t: time $\widetilde{O}(\epsilon^{-d/4-1.5})$.

Other quantum methods

Analogous to the classical ones:

- We can start with an initial state $|u_0\rangle$ and try to produce the state $|u_t\rangle$ corresponding to time t.
- We can use a coherent acceleration method for random walks due to [Apers+Sarlette '18], [Gilyen et al '18] which gives a square-root improvement in t: time $\widetilde{O}(\epsilon^{-d/4-1.5})$.
- We can diagonalise the discretised linear system and solve the diagonalised system: time $\widetilde{O}(\epsilon^{-d/4-1})$.

Other quantum methods

Analogous to the classical ones:

- We can start with an initial state $|u_0\rangle$ and try to produce the state $|u_t\rangle$ corresponding to time t.
- We can use a coherent acceleration method for random walks due to [Apers+Sarlette '18], [Gilyen et al '18] which gives a square-root improvement in t: time $\widetilde{O}(\epsilon^{-d/4-1.5})$.
- We can diagonalise the discretised linear system and solve the diagonalised system: time $\widetilde{O}(\epsilon^{-d/4-1})$.
- We can speed up the classical random walk using amplitude estimation.

Other quantum methods

Analogous to the classical ones:

- We can start with an initial state $|u_0\rangle$ and try to produce the state $|u_t\rangle$ corresponding to time t.
- We can use a coherent acceleration method for random walks due to [Apers+Sarlette '18], [Gilyen et al '18] which gives a square-root improvement in t: time $\widetilde{O}(\epsilon^{-d/4-1.5})$.
- We can diagonalise the discretised linear system and solve the diagonalised system: time $\widetilde{O}(\epsilon^{-d/4-1})$.
- We can speed up the classical random walk using amplitude estimation.
- Gives an algorithm for approximating $\int_S u(\mathbf{x},t)d\mathbf{x} \pm \epsilon$ in time $\widetilde{O}(\epsilon^{-1} \cdot \epsilon^{-1})$.

Conclusions

Some intuition we gained from this work:

- Quantum computers might achieve a speedup over classical algorithms for solving the heat equation, but this speedup is likely to be only polynomial (in fixed spatial dimension).

Conclusions

Some intuition we gained from this work:

- Quantum computers might achieve a speedup over classical algorithms for solving the heat equation, but this speedup is likely to be only polynomial (in fixed spatial dimension).
- Quantum algorithms might still offer an advantage in terms of flexibility over their classical counterparts.

Conclusions

Some intuition we gained from this work:

- Quantum computers might achieve a speedup over classical algorithms for solving the heat equation, but this speedup is likely to be only polynomial (in fixed spatial dimension).
- Quantum algorithms might still offer an advantage in terms of flexibility over their classical counterparts.
- The best quantum algorithms for solving PDEs might not be based on solving a system of linear equations.

Conclusions

Some intuition we gained from this work:

- Quantum computers might achieve a speedup over classical algorithms for solving the heat equation, but this speedup is likely to be only polynomial (in fixed spatial dimension).
- Quantum algorithms might still offer an advantage in terms of flexibility over their classical counterparts.
- The best quantum algorithms for solving PDEs might not be based on solving a system of linear equations.

Thanks!