Limitations of the Macaulay matrix approach for using the HHL algorithm to solve multivariate polynomial systems

Jianqiang Li (Penn State University)

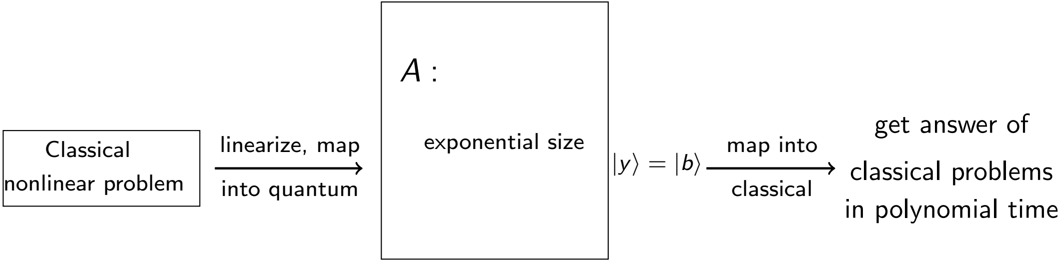
Joint work with

Jintai Ding, Vlad Gheorghiu, András Gilyén, Sean Hallgren

February 5, 2021

Finding new quantum algorithms based on HHL?

One approach:

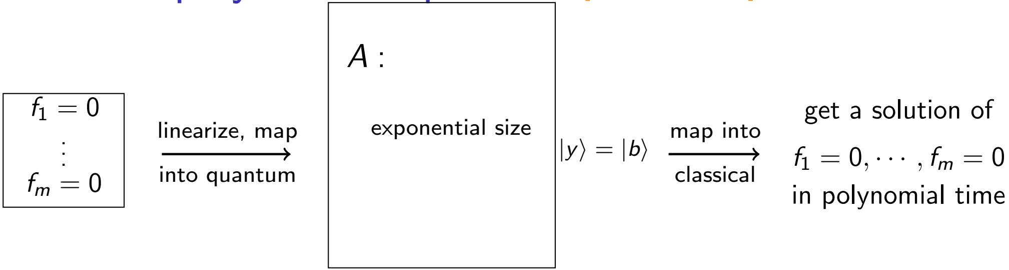


Classical nonlinear problem $\xrightarrow[\text{into quantum}]{\text{linearize, map}}$ A : exponential size $|y\rangle = |b\rangle$ $\xrightarrow[\text{classical}]{\text{map into}}$ get answer of classical problems in polynomial time

using HHL in polynomial time

Where to find such problems?

Linearize polynomial equations [Chen, Gao 17]



$f_1 = 0$

$\vdots$

$f_m = 0$

linearize, map

into quantum

A :

exponential size

$|y\rangle = |b\rangle$

map into

classical

get a solution of

$f_1 = 0, \cdots, f_m = 0$

in polynomial time

- The two mapping steps can be done in polynomial time;
- But: the HHL time depends on the condition number of A.

  - Chen and Gao did not analyze the condition number of A;

  - Instead, they pose the question that any of the following problems, including 3SAT, Polynomial systems over finite fields, SVP, and CVP, GI, SIS, LWE, the block cipher AES, and the stream cipher Trivum might have small condition number when reducing to solving polynomial equations [Chen, Gao 17; Chen, Gao, Yuan 18].

This work:

Question: Given $f_1, \cdots, f_m$ with unique nonzero solution $s \in \{0,1\}^n$, linearize into A.
The running time of Chen/Gao's algorithm $=$?

$H(s)$: the Hamming weight of s.

Theorem 1: The lower bound of the condition number of A is exponential($H(s)$).
So the algorithm takes exponential time in $H(s)$.

This implies that Grover's algorithm is faster.

Theorem 2: We refine the algorithm so that $H(s) \approx \log n$ might have a speedup.

Motivation for mapping polynomial systems to linear algebra

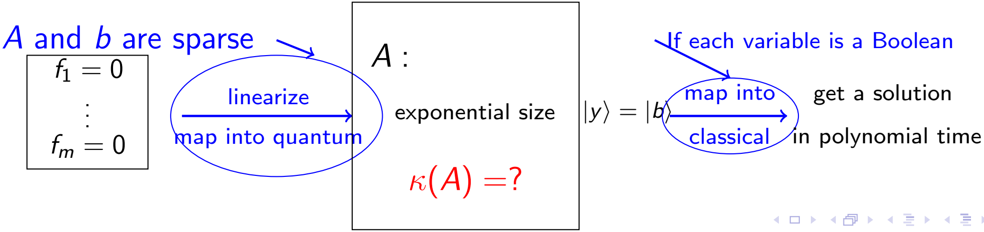
Bad news: Solving polynomial systems is NP-hard ;
→ So, the condition number better be big!

Good news:

- Classically, people have used this approach for some polynomial systems.

  For example: Bilinear polynomial systems; Overdefined systems of polynomial equations [Courtois, Klimov, Patarin, Shamir 00; Ars, Faugère, Imai, Kawazoe, Sugita 04] Polynomial systems over finite fields [Bettale, Faugère, Perret 09; Bardet, Faugère, Imai, Salvy, Spaenlehauer 13].

- Quantumly, [Chen, Gao 17]



A and b are sparse

$f_1 = 0$
$\vdots$
$f_m = 0$

linearize

map into quantum

A :

exponential size

$\kappa(A) = ?$

$|y\rangle = |b\rangle$

If each variable is a Boolean

map into

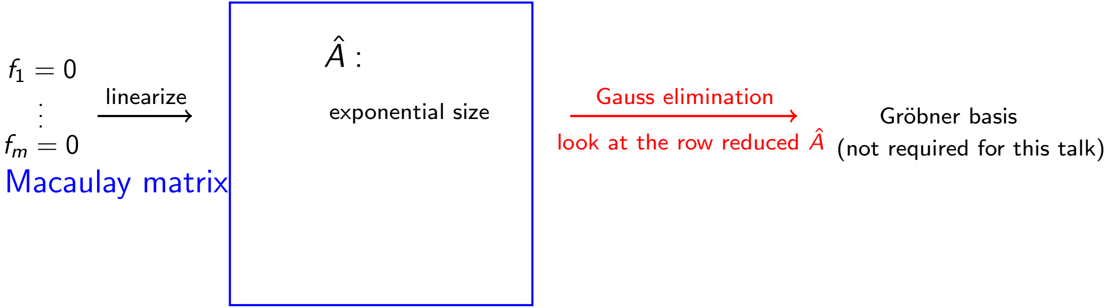
classical

get a solution

in polynomial time

Difference between the classical and quantum approach

- Classically,



$f_1 = 0$

$\vdots$

$f_m = 0$

$\xrightarrow{\text{linearize}}$

Macaulay matrix

$\hat{A}$:

exponential size

$\xrightarrow[\text{look at the row reduced } \hat{A}]{\text{Gauss elimination}}$

Gröbner basis

(not required for this talk)

- But: this is not directly possible with the quantum algorithm. Why?
  - Can't do Gauss elimination;
  - Can't look at the row reduced $\hat{A}$.
- Instead, something related but different works: solve $A\,|y\rangle = |b\rangle$, $\hat{A} = [A \quad -b]$.

Where does the matrix A come from?

Reduce polynomial systems to Macaulay linear systems[Chen,Gao17]

Input : $\mathcal{F}_1 = \{f_1, \ldots, f_m\} \subseteq \mathbb{C}[x_1, \ldots, x_n]$, $\deg(f_i) = 2$

$\mathcal{F}_2 = \{x_1^2 - x_1, \ldots, x_n^2 - x_n\}$.

Output : $s \in \{0,1\}^n$: $f_1(s) = \cdots = f_m(s) = 0$

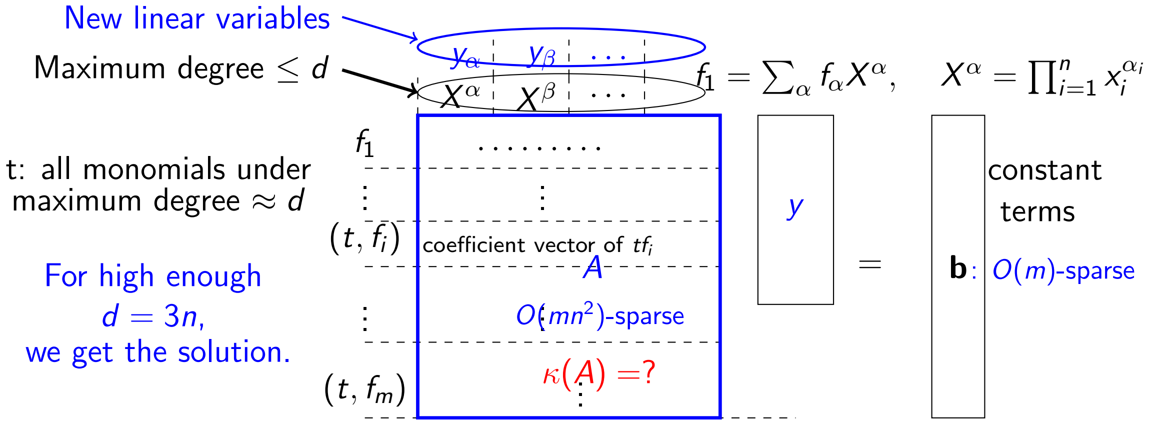
$$\boxed{\begin{array}{c}\text{Solving}\\ \text{Boolean}\\ \text{polynomial}\\ \text{systems}\end{array}} \longrightarrow \underbrace{\begin{bmatrix} \quad & & \quad \\ & A & \\ & & \end{bmatrix}}_{(3n)^n} \left.\vphantom{\begin{bmatrix} \\ \\ \\ \end{bmatrix}}\right\}{\scriptstyle (3n)^n} \begin{bmatrix} \\ y \\ \\ \end{bmatrix} = \begin{bmatrix} \\ b \\ \\ \end{bmatrix}$$

Can connect the HHL algorithm to this classical problem. But the running time**?**

Linearize polynomial equations for HHL

For example: $f = x_1^2 + x_2^2 + x_1x_2 - 1$ can be represented by

$$\begin{array}{cccccccc} & x_1 & x_2 & x_1^2 & x_1x_2 & x_2^2 & 1 \\ f = [& 0 & 0 & 1 & 1 & 1 & -1] \end{array}$$



New linear variables

Maximum degree $\leq d$

y_α y_β $\cdots$

X^α X^β $\cdots$

$f_1 = \sum_\alpha f_\alpha X^\alpha$, $X^\alpha = \prod_{i=1}^n x_i^{\alpha_i}$

t: all monomials under maximum degree $\approx d$

f_1 $\cdots\cdots\cdots$

$\vdots$ $\vdots$

(t, f_i) coefficient vector of tf_i

A

$\vdots$ $O(mn^2)$-sparse

(t, f_m) $\kappa(A) = ?$ $\vdots$

y $=$

constant terms

$\mathbf{b}$: $O(m)$-sparse

For high enough $d = 3n$, we get the solution.

The lower bound of the condition number $\kappa(A)$

Based on the all the setup we have, let

- $s \in \{0,1\}^n$: the unique non-zero solution of polynomial equations;
- $h = H(s)$: the Hamming weight of s.

Theorem 1 The lower bound of the condition number of A exponential in the Hamming weight of the solution : $(3n)^{h/2}$.

This implies that Chen/Gao's algorithm takes time $\Omega((3n)^{h/2})$.

- $h = O(1)$, classical algorithm can solve it efficiently;
- $h = \Theta(\log n)$, superpolynomially large, Grover is faster;
- $h = \Theta(n)$, superexponentially large, Grover is faster.

Proof of Theorem 1:

$$\kappa(A) = ||A|| \cdot ||A^+|| \geq ||A|| \frac{||A^+b||}{||b||} \geq ||A^+b|| = ||\hat{s}|| \tag{1}$$

For $\hat{s}$: $${\scriptstyle (3n)^n}\left\{\left[\begin{array}{c}\vdots\\ \hdashline 1\\ \hdashline \vdots\end{array}\right]\right. \leftarrow \Pi_{x_i \in S} x_i^{e_i}$$

the number of 1's in $\hat{s}$: $(d+1)^h - 1$

$$A\hat{s} = b$$

Therefore $||\hat{s}||^2 = (d+1)^h - 1$ and [Lemma 4.1][Chen, Gao 17] uses $d = 3n$, we have

$$\kappa(A) \geq \sqrt{(3n{+}1)^h - 1} \geq (3n)^{h/2}.$$

Can we do better?

Chen/Gao's algorithm takes time $\Omega((3n)^{h/2})$.

So Grover's algorithm is faster.

Is this the end of the story? No!

Next:

We can refine the algorithm so that the refined algorithm takes time $\Omega(2^{h/2})$.

- $h = O(1)$, classical algorithm can solve it efficiently;
- $h = \Theta(\log n)$, open again for potential quantum polynomial time;
- $h = \Theta(n)$, exponentially large.

How can we do better?

Variables are Boolean $\implies$ all monomials can be replaced by square-free versions.

Based on this observation:

- We reduce polynomial systems to better Boolean Macaulay linear systems with a smaller size;
- Solve the Boolean Macaulay linear system $Ax = b$ using HHL and the lower bound of $\kappa(A)$ is $\Omega(2^{h/2})$;
- We provide an alternative approach to get the solution of polynomial equations.

Reduce polynomial systems to Boolean Macaulay linear systems

Here we extend the definition of Boolean Macaulay matrix $\hat{A}$ from $\mathbb{F}_2$ [Bardet, Faugère, Imai, Salvy, Spaenlehauer 13] to $\mathbb{C}$, where $\hat{A} = [A \quad -b]$.

$$\boxed{\begin{array}{c} f_1 = 0 \\ f_2 = 0 \\ \vdots \\ f_m = 0 \end{array}} \longrightarrow \left.\underbrace{\begin{bmatrix} \quad & & \quad \\ & A & \\ \quad & & \quad \end{bmatrix}}_{2^n}\right\}2^n \begin{bmatrix} y \end{bmatrix} = \begin{bmatrix} b \end{bmatrix}$$

Can we get the correct solution with the Boolean Macaulay linear system? Yes!

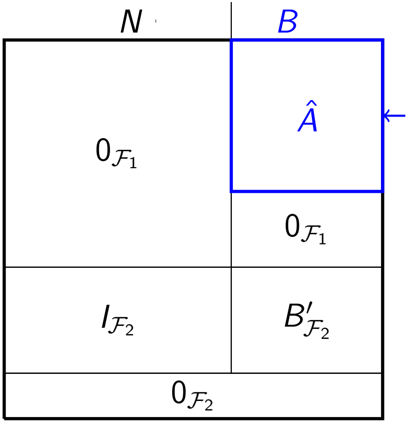
The row reduced Macaulay matrix

Because solving the Boolean Macaulay linear system is equivalent to solving the Macaulay linear system:

N: the non-square-free monomials

B: the square-free monomials

The row reduced Macaulay matrix



Boolean Macaulay matrix

$$\hat{A} = [A \quad -b]$$

Connect the Boolean Macaulay linear system to HHL

Goal: solve $Ay = b$ using the HHL algorithm. The running time depends on the condition number.

First: Are the two mapping steps still efficient? Yes!

- A: $O(mn^2)$-sparse matrix; $\checkmark$
- b : can be efficiently prepared as a quantum state (sparse); $\checkmark$
- $|y\rangle$: get the solution of polynomial equations in polynomial time; $\checkmark$
- The rank of the matrix A: high (full) column rank; $\checkmark$

Second: How about the condition number? $\kappa(A) : \Omega(2^{h/2})$

The lower bound of the condition number $\kappa(A)$

- $s \in \{0,1\}^n$: the unique non-zero solution of polynomial equations

- $\hat{s}$: $2^n \left\{ \begin{bmatrix} \vdots \\ \hdashline 1 \\ \hdashline \vdots \end{bmatrix} \right. \leftarrow \Pi_{x_i \in S} x_i^{e_i}$

the number of 1's in $\hat{s}$: $2^h - 1$

$$A\hat{s} = b$$

Theorem 2: by Theorem 1, $\kappa(A) \geq ||\hat{s}|| = 2^{h/2}$

The lower bound of the condition number of A is polynomial when $h = \log n$.

Open question: Are there any polynomial systems with the corresponding condition number also upper bounded by a polynomial?

The quantum solution of the Boolean Macaulay linear system

- $s \in \{0,1\}^n$: the unique non-zero solution of polynomial equations
- $\hat{s}$: $2^n \left\{ \begin{bmatrix} \vdots \\ \hline 1 \\ \hline \vdots \end{bmatrix} \right. \leftarrow \prod_{x_i \in S} x_i^{e_i}$

the number of 1's in $\hat{s}$: $2^h - 1$

$A\hat{s} = b$

$U = \{x_1, x_2, \ldots, x_n\}$, let $S \subseteq U$ be an unknown subset where all the variables in S have value 1.

Then the quantum state

$$|\hat{s}\rangle = \frac{1}{\sqrt{2^{|S|} - 1}} \sum_{R \in 2^S \setminus \{\emptyset\}} |R\rangle$$

is the uniform superposition on the support of the solution vector $\hat{s}$.

Question: How to compute S given copies of $|\hat{s}\rangle$?

The variant of the quantum coupon collector problem

$U = \{x_1, x_2, \ldots, x_n\}$, let $S \subseteq U$ be an unknown subset we want to compute.

The variant of the quantum coupon collector problem: Given copies of the state

$$|\hat{s}\rangle = \frac{1}{\sqrt{|S_d|}} \sum_{R \in S_d} |R\rangle ,$$

which is a superposition of subsets of S of size at most d. The goal is to compute S.

Theorem 3 Measuring $r = O((|S|/d)\log|S|)$ copies of the quantum superposition state $|\hat{s}\rangle$, we can compute the set S with probability at least $1 - 1/\mathrm{poly}(|S|)$.

- $d = |S|$, $r = O(\log|S|)$
- $d = \log|S|$, $r = O(|S|)$
- $d = 1$, $r = O(|S|\log|S|)$ [Arunachalam, Belovs, Childs, Kothari, Rosmanis, Wolf 20]

Summary

- We show Chen/Gao's algorithm is exponential in the Hamming weight of the solution;
- Our refined algorithm opens the possibility of quantum speedup for polynomial systems with solution with Hamming weight $h = \log n$;

  For problems, like polynomial system over finite fields, SVP, CVP, GI, SIS, LWE, 3SAT, GI [Chen, Gao, Yuan 18] and Factoring [Burges 03], the Hamming weight of the solution is going to be large when reducing to polynomial equations.
- Our analysis also applies to the truncated HHL algorithm;

  - Only inverted on well-conditioned subspace.
- We give an alternative approach to extract the classical solution via a generalization of the quantum coupon collector problem.

Open problems

- Find some special polynomial systems to exhibit the potential superpolynomial speedup when $h = \log n$?

- The lower bound of the condition number in the multiple solutions case?

  The refined Boolean Macaulay linear system also works for the multiple solutions case, this might open more possibilities of quantum speedup.

- Prove the matching lower bound of the variant of the quantum coupon collector problem?