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

Jintai Ding[*][†][‡]   Vlad Gheorghiu[§][¶]   András Gilyén[‖][**]   Sean Hallgren[††]   Jianqiang Li[‡‡]

11/02/21

Abstract

Recently Chen and Gao [CG17b] proposed a new quantum algorithm for Boolean polynomial system solving, motivated by the cryptanalysis of some post-quantum cryptosystems. The key idea of their approach is to apply a Quantum Linear System (QLS) algorithm to a Macaulay linear system over $\mathbb{C}$, which is derived from the Boolean polynomial system. The efficiency of their algorithm depends on the condition number of the Macaulay matrix. In this paper, we give a strong lower bound on the condition number as a function of the Hamming weight of the Boolean solution, and show that in many (if not all) cases a Grover-based exhaustive search algorithm outperforms their algorithm. Then, we improve upon Chen and Gao's algorithm by introducing the Boolean Macaulay linear system over $\mathbb{C}$ by reducing the original Macaulay linear system. This improved algorithm could potentially significantly outperform the brute-force algorithm, when the Hamming weight of the solution is logarithmic in the number of Boolean variables.

Furthermore, we provide a simple and more elementary proof of correctness for our improved algorithm using a reduction employing the Valiant-Vazirani affine hashing method, and also extend the result to polynomial systems over $\mathbb{F}_q$ improving on subsequent work by Chen, Gao and Yuan [CGY18]. We also suggest a new approach for extracting the solution of the Boolean polynomial system via a generalization of the quantum coupon collector problem [ABC+20].

1 Introduction

Solving systems of multivariate polynomial equations is a fundamental problem that is NP-complete even when the polynomials are restricted over $\mathbb{F}_2$. The problem can be reduced to solving an exponential number of linear equations via the so-called Macaulay matrix, which holds coefficients of linear equations that come from the input polynomials, and multiples of them (multiplying each polynomial by each monomial up to a certain degree). Each monomial is represented by a new variable, recasting the polynomial equations and their multiples as linear equations. The usual

---

[*]University of Cincinnati, OH, USA `jintai.ding@gmail.com`

[†]Yau Math. Sci. Center, Tsinghua University

[‡]Ding Lab, Beijing Institute of Mathematical Sciences and Applications

[§]softwareQ Inc. `vlad.gheorghiu@uwaterloo.ca`

[¶]Institute for Quantum Computing / Dept. of Combinatorics & Optimization, University of Waterloo, ON, Canada

[‖]Institute for Quantum Information and Matter, Caltech, Pasadena CA, USA

[**]Alfréd Rényi Institute of Mathematics, Budapest, Hungary `gilyen@renyi.hu`

[††]Department of Computer Science and Engineering, Pennsylvania State University, PA, USA `hallgren@cse.psu.edu`

[‡‡]Department of Computer Science and Engineering, Pennsylvania State University, PA, USA `jxl1842@psu.edu`

classical approach to solve a polynomial system is based on computing the Gröbner basis of the corresponding polynomial ideal by triangularizing the Macaulay matrix. There is a vast literature on characterizing and improving the complexity of solving various types of polynomial systems using the Macaulay matrix [AFI+04, CG17a, CKPS04, DS13, Die04, Per16, WW15].

In quantum computing, the HHL [HHL09] Quantum Linear System (QLS) algorithm outputs a quantum state $|x\rangle$ such that $\mu A |x\rangle = |b\rangle$ for an exponentially large matrix A with certain properties, and a quantum state $|b\rangle$, in time $\widetilde{\mathcal{O}}(\kappa^2 s^2)$,[1] where κ is the *condition number* of A, μ is a normalization factor, and s is the sparsity of the matrix A, while state-of-the-art QLS algorithms [Amb12, CKS17, CGJ19, GSLW19, SSO19, LT20] have complexity $\widetilde{\mathcal{O}}(\kappa s)$. Although, the QLS algorithm is BQP-complete [HHL09], meaning that it captures all essential features of quantum computing, a natural "killer-application" is still to be discovered – showing the difficulty of finding a practically interesting problem instance that satisfies all stringent conditions. For example, to efficiently solve the classical equation $Ax = b$ using the original HHL algorithm, where implicit access is given to an exponentially large matrix A and b, the following must be satisfied: the state $|b\rangle$ can be efficiently prepared, the sought data can be efficiently extracted from the output state $|x\rangle$, and the matrix A should be sparse and well-conditioned [Aar15].

Chen and Gao [CG17b] made an interesting connection between the exponential size Macaulay matrix and the HHL algorithm. While they use Gröbner bases in their proof of correctness, they do not explicitly compute the Gröbner basis and instead use the HHL algorithm to solve the exponentially large system of linear equations resulting from the Macaulay matrix. They show that the access requirements that usually cause so much trouble, can all be resolved for this application, namely: they can efficiently compute the entries of an appropriate sparse matrix A, prepare $|b\rangle$, and extract the answer from $|x\rangle$. However, a major question was left open: what is the condition number of the matrices, driving the running time? Intuitively, for worst case instances of polynomial systems, the condition number of the resulting matrix should be large because the approach can solve NP-complete problems. This being said, the analysis of the condition number was left open, both in general, and for special cases such as breaking cryptosystems which have distributions over specific problem instances that might be easier than the worst case.

In this paper we prove an exponential lower bound on the condition number κ of the matrix A related to the Boolean polynomial system, which shows that the quantum algorithm in [CG17b] takes exponential time in the worst case. We also give a Grover-based brute-force search algorithm that outperforms their quantum algorithm for solving Boolean polynomial systems when there is a unique solution or all solutions have the same Hamming-weight. Specifically, in the unique solution case we give a simple proof that the condition number κ is $\Omega((3n)^{h/2})$, where h is the Hamming weight of the solution to the original n-variable Boolean polynomial system. Meanwhile, a simple Grover-based brute-force search algorithm over the possible assignments to the variables takes time $\mathcal{O}\left(\sqrt{\binom{n}{h}}\right)$, where $\sqrt{\binom{n}{h}} \le \left(\frac{en}{h}\right)^{h/2} \le (3n)^{h/2}$.

In fact, we give "robust" lower bounds on the condition number by also considering "truncated" QLS algorithms [HHL09, GSLW19]. Namely, if the singular-values of A are only inverted on a *well-conditioned* subspace and the overlap of the solution x with such a subspace is large enough, then a "truncated" QLS algorithm can provide a sufficiently accurate solution $\tilde{x}$. In order to give a bound on the performance of such "truncated" versions of the QLS algorithm, we define the concept of the *truncated QLS condition number* $\kappa_b(A) := \|A\|\|A^+b\|/\|b\|$,[2] which is also a lower bound on

---

[1] We denote $\mathcal{O}(T \cdot \text{poly} \log(T) \cdot \text{poly}(1/\varepsilon))$ by $\widetilde{\mathcal{O}}(T)$, where ε is the required precision of the solution.

[2] A^+ stands for the pseudoinverse of A or more precisely the Moore-Penrose inverse of A, and we use the notation $\|\cdot\|$ for the ℓ_2 norm of vectors and for the corresponding induced operator norm, i.e., the spectral norm.

$\kappa = \|A\|\|A^+\|$. All of our lower bounds also apply to the truncated QLS condition number, ruling out further improvement by truncated QLS algorithms.

Finally, we refine Chen and Gao's algorithm to the point that our lower bound does not always rule out the possibility of a superpolynomial quantum speedup even for unique solutions. In particular, the lower bound changes from $(3n)^{h/2}$ to $2^{h/2}$ on our refined algorithm, so for $h = \Theta(\log n)$ the lower bound is only a polynomial, while the brute-force algorithm takes quasi-polynomial time. Thus, it is conceivable that the condition number is upper bounded by poly(n) for some set of interesting input equations, potentially yielding a superpolynomial speedup. We leave it open to find a problem instance whose associated Macaulay matrix has a small enough condition number so that the running time of our refined quantum algorithm gives a speedup over the best classical or Grover-based algorithm. Such an example could result in a new type of quantum speedup and one that uses the HHL algorithm in a novel way.

The core ingredient of our refined algorithm is to show that the Macaulay matrix can be simplified to what we call the Boolean Macaulay matrix over $\mathbb{C}$ by exploiting that the input consists of quadratic polynomials over $\mathbb{C}$, but restricted to $0/1$ solutions. The Boolean Macaulay matrix is a submatrix of the original Macaulay matrix that can be obtained via Gaussian elimination over $\mathbb{C}$. This construction of the Boolean Macaulay matrix over $\mathbb{C}$ is different from the Boolean Macaulay matrix over $\mathbb{F}_2$ as defined in [BFSS13] since they are over different fields. This matrix preserves the solution set while its size is much smaller compared to the original Macaulay matrix – ultimately leading to a smaller lower bound $\Omega(2^{h/2})$ on the condition number.

The correctness of our refined algorithm can be shown via the equivalence between the Boolean Macaulay linear system and the Macaulay linear system, where the correctness of the latter has been proven in [CG17b]. For completeness, we also provide a simple self-contained proof of correctness for our improved algorithm, which is more elementary than that of the original algorithm proposed by Chen and Gao [CG17b], since our proof does not require Gröbner bases. Instead, our proof combines a special case of the reduction in [CGY18] with the Valiant-Vazirani affine hashing method, reducing any Boolean polynomial system with more than one solution to one that has a unique solution. For a Boolean Macaulay linear system that has a unique solution, we also provide an alternative approach for extracting the Boolean solution of the corresponding Boolean polynomial system from the output quantum state, i.e., the normalized monomial solution vector of the Boolean Macaulay linear system. Specifically, we reformulate this problem as a generalization of the quantum coupon collector problem, and prove that $O(\log n)$ iterations suffice for extracting a solution, whereas Chen and Gao's algorithm uses $O(n)$ iterations. On the other hand, the affine hashing reduction introduces $\mathcal{O}(n)$ extra rounds, so the total number of iterations in our algorithm can be bounded as $\mathcal{O}(n \log n)$.

2   Quantum algorithms for solving polynomial systems

Chen and Gao [CG17b] proposed using the HHL algorithm to solve a Boolean polynomial system via solving an exponentially large linear system of equations. Now we discuss the two main parameters that appear in the complexity of the QLS algorithm, and their relevance in our case.

$\kappa_b(A):$ The *truncated QLS condition number* (tQLScn) $\kappa_b(A)$ of the QLSP $Ax = b$ is an important parameter related to the time-complexity of "truncated" QLS algorithms.[3] (For simplicity let

---

[3]Note that the truncated QLS condition number gives a lower bound on the performance of truncated QLS alorithms, but it does not characterize their complexity, i.e., there might not exist any truncated QLS algorithm with complexity matching the truncated QLS condition number.

us assume without loss of generality that $\|A\| \leq 1$ and $\|b\| = 1$.) We use a simple Markov-type inequality showing that inverting A via a truncated variant of the QLS algorithm, with (condition number) truncation much below $\kappa_b(A)$, must give a highly inaccurate solution. Indeed, let S be a subspace, which is spanned by right singular vectors of A that have singular value at least $c/\kappa_b(A)$ for some $c > 1$. Let Π_S be the orthogonal projector on S, then $\|\Pi_S x\| = \|\Pi_S A^+ b\| \leq \|\Pi_S A^+\| \|b\| \leq \kappa_b(A)/c$. On the other hand $\|x\| = \|A^+ b\| \geq \|A\| \|A^+ b\|/\|b\| = \kappa_b(A)$, thereby the overlap of x with the subspace S must be relatively small for large c. This argument shows, that giving a lower bound on the truncated QLS condition number makes our results stronger, as it does not only bound the complexity of standard QLSP solvers, but also lower-bounds the complexity of fine-tuned truncated variants.

s : In order to efficiently solve the Quantum Linear System Problem (QLSP), we need a succinct representation of the vector b and the matrix A. In our case both b and A are s-sparse for some polynomially large s, i.e., they have at most s nonzero entries in every column and row. In order to utilize their sparsity, we also need to be able to efficiently compute the locations and the values of the nonzero entries; this is easy to do in our case, since the vector b is sparse and the (Boolean) Macaulay matrix A has a quasi-Toeplitz structure.

More precisely, it suffices to have efficient (quantum) circuits computing the locations and values of the nonzero elements of A, allowing us to perform the transformations

$$|i, k\rangle \longrightarrow |i, \bar{c}(i, k)\rangle \tag{1}$$

$$|i, j, 0\rangle \longrightarrow |i, j, \bar{A}_{ij}\rangle, \tag{2}$$

where i labels the row indices of the matrix $\bar{A}$ standing for A and A^T, k labels the nonzero entries of $\bar{A}$ (which is assumed to be s-sparse), $\bar{c}(i, k)$ represents the column index of the k-th nonzero entry of the matrix $\bar{A}$ in row i, and $|0\rangle$ in (2) represents a (large enough) ancillary system in which the matrix element $\bar{A}_{ij}$ can be stored (as a bit-string with sufficient precision so that errors can be neglected). Note that the transformations (1)-(2) are essentially only used to build an efficient quantum circuit C_A [GSLW18, Lemma 48], which implements a unitary U, such that the top left $N \times M$ corner of U equals A – such a unitary is called a block-encoding of A [CGJ19, GSLW19]. Similarly, the sparsity assumption for b is only used in order to build an efficient quantum circuit C_b for preparing the quantum state $|b\rangle := \left(\sum_{i=1}^{N} b_i |i\rangle\right)/\left\|\sum_{i=1}^{N} b_i |i\rangle\right\|$. Thereby the QLSP problem can also be efficiently solved for non-sparse b or A, whenever we can build efficient quantum circuits C_b, C_A. For example, if we have both b and A stored in a quantum random access memory (qRAM) using an appropriate data structure, then we can build the efficient quantum circuits C_b, C_A as described in [KP17, CGJ19, GSLW19].

For completeness, let us mention two additional types of quantum algorithms that have been proposed for solving polynomial systems:

1. QAOA : In 2002, Burges [Bur02] reformulated RSA factoring problem as a polynomial system solving problem that has a unique solution, which is a special case of Problem 3.2. Later, Anschuetz, Olson, Aspuru-Guzik and Cao [AOAGC18] reformulated the polynomial system solving problem as a Local Hamiltonian Problem (LHP) that has a corresponding unique ground state, and they applied the QAOA algorithm for finding this ground state; the exact complexity of this algorithm is unknown.

2. Grover : In 2017, Faugère, Horan, Kahrobaei, Kaplan, Kashefi, and Perret [FHK+17] presented a quantum version of the classical algorithm in [BFSS13], that applies Grover search on both

the exhaustive search and the consistency check subroutines. Under certain assumptions, the running time of this quantum algorithm is slightly better than $\mathcal{O}(2^{n/2})$, which is the running time of trivial Grover search algorithm. Similarly, Bernstein and Yang [BY18] gave a quantum algorithm (GroverXL) for random polynomial systems over a finite field $\mathbb{F}_q$.[4]

While QAOA is only a heuristic algorithm and Grover search only represents a quadratic speedup, the QLS algorithm – being BQP-complete – captures the power of quantum computation and promises rigorous superpolynomial speedups, giving strong motivation for the study of the Macaulay matrix approach.

3 Reducing polynomial system solving over a finite field $\mathbb{F}_q$ to polynomial system solving over $\mathbb{C}$

First, we present the $\mathbb{F}_q = \mathbb{F}_2$ special case of Chen, Gao and Yuan's approach [CGY18]. In this special case there is a bijection between the solution sets of the corresponding polynomial systems over $\mathbb{F}_2$ and $\mathbb{C}$.

Problem 3.1. *Solve a system of n-variate quadratic polynomials with Boolean variables over $\mathbb{F}_2$.*
    Input : $\mathcal{F} = \{f_1, \ldots, f_m\} \subseteq \mathbb{F}_2[x_1, \ldots, x_n]$ *with* $\deg(f_i) \leq 2$ *for* $i = 1, 2, \ldots, m$.
    Output : *Find a solution* $s \in \mathbb{F}_2^n$ *such that* $f_1(s) = \cdots = f_m(s) = 0$, *when one exists.*

Problem 3.2. *Solve a system of n-variate quadratic polynomials over $\mathbb{C}$, together with the $\mathbb{F}_2$ field equations that force variables to be Boolean.*
    Input : $\mathcal{F} = \mathcal{F}_1 \cup \mathcal{F}_2 \subseteq \mathbb{C}[x_1, \ldots, x_n]$ *where* $\mathcal{F}_1 = \{f_1, \ldots, f_m\}$ *with* $\deg(f_i) = 2$ *for* $i = 1, \ldots, m$, *and* $\mathcal{F}_2 = \{x_1^2 - x_1, \ldots, x_n^2 - x_n\}$.
    Output : *Find an* $s \in \{0, 1\}^n$ *such that* $f_1(s) = \cdots = f_m(s) = 0$ *over* $\mathbb{C}$, *when one exists.*

Let $\#\mathcal{F}$ denote the maximum number of nonzero terms in any polynomial in $\mathcal{F}$. Also let $\#f$ be a shorthand for $\#\{f\}$.

Lemma 3.3. *There is a polynomial-time reduction from Problem 3.1 on n variables and a set of m equations $\mathcal{F}$ to Problem 3.2 on $n + m \cdot \lfloor \log_2 \#\mathcal{F} \rfloor$ variables and $n + m \cdot (\lfloor \log_2 \#\mathcal{F} \rfloor + 1)$ equations.*

Proof. Solving Problem 3.1 is equivalent to solving the following polynomial system in variables $x_1, \ldots, x_n, z_1, \ldots, z_m$ over $\mathbb{C}$:

$$\forall i \in [m]: \qquad f_i(x_1, \ldots, x_n) - z_i = 0, \tag{3}$$

$$\forall i \in [m]: \qquad z_i/2 \in \mathbb{Z}, \tag{4}$$

$$\forall j \in [n]: \qquad x_j^2 - x_j = 0. \tag{5}$$

The field equations (5) force each x_j to be 0 or 1, and therefore each f_i evaluates to an even or odd integer. Equations (3)-(4) force f_i to evaluate to an even integer, so it is 0 mod 2. It is easy to see that each integer z_i is in $[0, \#f_i]$ for $i = 1, \ldots m$.

Chen, Gao and Yuan [CGY18] then represent each $z_1, \ldots, z_m$ by the bits in its binary expansion. For each variable $z_i \in [0, \#f_i]$ a polynomial is introduced with Boolean variables y_{ib} to represent its

---

[4]The generalization of the Boolean Macaulay matrix method in [BFSS13] is equivalent to the reduced XL approach that first appeared in [CKPS04] and then defined in [Die04].

value in binary, i.e.,

$$z_i = \sum_{b=1}^{\lfloor \log_2 \#f_i \rfloor} 2^b y_{ib}, \text{ and } y_{ib}^2 - y_{ib} = 0 \text{ for } b = 1, \ldots, \lfloor \log_2 \#f_i \rfloor.$$

Substituting the polynomials and Boolean constraints corresponding to each z_i into the polynomial equations (3)-(5), we get a following polynomial system $\mathcal{F}$ over $\mathbb{C}$:

$$\forall i \in [m]: \qquad\qquad\qquad f_i(x_1, \ldots, x_n) - \sum_{b=1}^{\lfloor \log_2 \#\mathcal{F} \rfloor} 2^b y_{ib} = 0, \qquad (6)$$

$$\forall i \in [m] \forall b \in [\lfloor \log_2 \#\mathcal{F} \rfloor]: \qquad\qquad\qquad y_{ib}^2 - y_{ib} = 0, \qquad (7)$$

$$\forall j \in [n]: \qquad\qquad\qquad x_j^2 - x_j = 0. \qquad (8)$$

It is easy to see that there is a bijection between the set of solutions of $\mathcal{F} \subseteq \mathbb{F}_2[x_1, x_2 \ldots, x_n]$ and the set of solutions of (6)-(8) over $\mathbb{C}$. On one hand, given a solution $(s_1, \ldots, s_n)$ of $\mathcal{F} \subseteq \mathbb{F}_2[x_1, x_2 \ldots, x_n]$, evaluating $f_i(s_1, \ldots, s_n)$ over $\mathbb{C}$ gives an even number z_i, and its binary expansion gives the values of the y_{ib} variables. On the other hand, let $(s_j), (t_{ib})$ be a solution to (6)-(8). For each j, $s_j \in \{0, 1\}$ by (8), and for each i, $f_i(s_1, \ldots, s_n) = \sum_{b=1}^{\lfloor \log_2 \#\mathcal{F} \rfloor} 2^b y_{ib}$ by (6). Due to (7) this implies that $f_i(s_1, \ldots, s_n) \equiv 0 \mod 2$, so (s_j) is a solution of $\mathcal{F}$. $\qquad\square$

Given a set of polynomials $\mathcal{F} \subseteq \mathbb{F}_q[X]$, Chen, Gao and Yuan [CGY18] propose an analogous approach for reducing the polynomial system $\mathcal{F}$ to a polynomial system $\mathcal{F}_\mathbb{C} = 0$, where $\mathcal{F}_\mathbb{C} \subseteq \mathbb{C}[X, Y]$, in the form of Problem 3.2. However, in general even if $\mathcal{F}$ has a unique solution, the constructed polynomial systems $\mathcal{F}_\mathbb{C}$ may have multiple solutions.

Now we present two additional reduction steps that make the polynomial systems in Problem 3.2 easier to handle:

Red1: In order to ensure that the polynomial system $\mathcal{F} \subseteq \mathbb{C}[X]$ in Problem 3.2 has no more than one solution, we employ the Valiant-Vazirani affine hashing method [VV86]. Suppose that the polynomial system $\mathcal{F}$ has $S \in [2^n]$ different solutions. The main idea of the affine hashing method is the following [BKW19]: if one introduces $\lfloor \log_2(S) \rfloor + 2$ random linear equations $\mathcal{F}_R$ with $\mathcal{F}_R \subseteq \mathbb{F}_2[X]$, then they isolate a unique solution with probability at least $\frac{1}{8}$. Even if we don't know the number of solutions apriory, we can loop over all possible values of $\lfloor \log_2(S) \rfloor \in \{0, 1, \ldots, n\}$; making $\mathcal{O}(\ln(1/\varepsilon))$ trials for all possible choices of $\lfloor \log_2(S) \rfloor$ gives at least one system $\mathcal{F}_R = 0$ with a unique solution with probability at least $1 - \varepsilon$. This amounts to $\mathcal{O}(n \log(1/\varepsilon))$ different polynomial systems to check. Remember that $\mathcal{F}_R \subseteq \mathbb{F}_2[X]$ whereas $\mathcal{F} \subseteq \mathbb{C}[X]$. By Lemma 3.3, we can reduce the new polynomials $\mathcal{F}_R$ to polynomials $\mathcal{F}_{R\mathbb{C}} \subseteq \mathbb{C}[X, Y]$, where Y is the set of new variables introduced during the reduction. Finally, if $\mathcal{F}_R$ isolated a unique solution of $\mathcal{F}$, then the polynomial system $\mathcal{F}_{R\mathbb{C}} \cup \mathcal{F}$ has a unique solution. Thus, without loss of generality, we can always assume that Problem 3.2 has a unique solution.

Red2: Any polynomial system $\mathcal{F} = \{f_1, f_2, \ldots, f_m\} \subseteq \mathbb{C}[X]$ can be rewritten as $\mathcal{F}' = \{f_1', f_2', \ldots, f_m'\}$, where f_1' has constant term -1, while $f_2', \ldots, f_m'$ have no constant terms. In case no polynomial in $\mathcal{F}$ has a constant term, the all-zero vector is a trivial solution. Otherwise, let c_i denote the constant term of f_i, and let us assume without loss of generality that $c_1 \neq 0$. Then we can simply set $f_1' := -f_1/c_1$, and $f_i' := f_i + c_i f_1'$ for all $i \in \{2, 3, \ldots, m\}$.

The above two reductions increase the parameters considered in this paper only moderately. Indeed, Red1 introduces at most $\mathcal{O}(n \log(n))$ new equations and variables, while Red2 only affects

the number of nonzero terms in the polynomial system. Moreover, Red2 increases $\#\mathcal{F}$ and the total number of nonzero terms $\sum \#f_i$ by at most a factor of 2 (for the latter we shall choose f_1 to be the polynomial with a nonzero constant term that also has the fewest nonzero coefficients).

4 Macaulay linear systems and their tQLScn

In this section we define the Macaulay linear system of a set of polynomials $\mathcal{F} \subseteq \mathbb{C}[x_1, \ldots, x_n]$ and show that when $\mathcal{F}$ has a unique solution, the condition number of the matrix is $\Omega(\sqrt{\binom{n}{h}})$, where h is the Hamming weight of the solution. We show that the lower bound also holds when using max degree instead of total degree in the definition, as was done in [CG17b], showing that their proposed quantum algorithm for solving polynomial equations by using the QLS algorithm to solve a Macaulay linear system in general takes time $\Omega((3n)^{h/2})$. We also show that if there are t different solutions, but they have the same Hamming weight h, then the above lower bound can reduce by at most a factor of $\sqrt{t}$. Finally, we give a formula that can be used for giving a lower bound on the condition number for any number of solutions and present computational evidence that this analytical lower bound is exponentially large in terms of the smallest Hamming weight among the solutions.

4.1 Macaulay linear systems

There is a well-known approach for solving polynomial systems by linearizing them with the help of introducing new latent auxiliary variables. The advantage of this approach is that the problem becomes linear, but the downside is that the new problem is exponentially large. The matrix of the resulting linear system is called the Macaulay matrix.

Definition 4.1. *The* Macaulay matrix $\hat{\mathcal{M}}$ *of degree* d *of* $\mathcal{F} = \{f_1, \ldots, f_m\} \subseteq \mathbb{C}[X]$ *is the matrix where each row is labeled by a pair of polynomials* $(\hat{m}, f)$ *and contains the corresponding coefficient vector of the polynomial* $\hat{m}f$. *The rows range over all* $f \in \mathcal{F}$ *and monomials* $\hat{m}$ *such that* $\hat{m}f$ *has degree at most* d. *The columns are labeled by the set of monomials in* $x_1, \ldots, x_n$ *of degree at most* d *and are ordered with respect to a specified monomial ordering. The element in the row corresponding to* $(\hat{m}, f)$ *and the column corresponding to the monomial* $\hat{m}'$ *is the coefficient of* $\hat{m}'$ *in the polynomial* $\hat{m}f$.

In the above definition one can interpret the degree as either the total degree or the max degree (the maximum degree of any variable) of multivariate polynomials resulting in different notions of the Macaulay matrix. When it is necessary we will always clarify which definition is being used. For example, [CG17b] uses the max degree, so all references to that paper refer to the max degree version of this definition.

Row operations on the matrix $\hat{\mathcal{M}}$ correspond to polynomial addition, subtraction and scalar multiplication in the polynomial ideal $\langle \mathcal{F} \rangle$ [Bat13, B$^+$18], and these operations preserve the common roots of the system. Classically, Gaussian elimination can be performed on this matrix, and the entries can be read out from the row-reduced matrix [Die04, CKPS04]. However, in the quantum case, we cannot directly do Gaussian elimination on this matrix and look at the row-reduced matrix. Instead, Chen and Gao showed that the QLS algorithm can be used for sampling from nonzero solutions of the following related linear system.

Definition 4.2. *Let $\hat{\mathcal{M}}$ be the Macaulay matrix of a given polynomial system, with the last column $-\vec{b}$ corresponding to the constant terms of the polynomials. Let $\hat{\mathcal{M}} = [\mathcal{M}| -\vec{b}\,]$. Then the equation $\mathcal{M}\vec{y} = \vec{b}$ is called the* Macaulay linear system.

In other words, the Macaulay matrix is the augmented matrix from the Macaulay linear system $\mathcal{M}\vec{y} = \vec{b}$. Due to reduction Red2 in the previous section, it can be assumed that exactly one of the input polynomials has a nonzero constant term. So we may assume without loss of generality that $\vec{b} = [1 \quad \mathbf{0}]^T$, where the vector $\vec{b}$ corresponds to the column vector indexed by the degree 0 monomial 1.

Chen and Gao's algorithm applies the QLS algorithm to output the quantum state $|\vec{y}\rangle$ that can be measured in order to sample from monomials with nonzero value in a valid assignment corresponding to a solution. We will lower bound the condition number of $\mathcal{M}$, which will in turn lower bound the running time of the proposed algorithm.

Chen and Gao proposed to set the max degree to $3n$ in the Macaulay linear system, and they showed with this choice if a set of polynomials $\mathcal{F}$ has a unique solution, then the linear system also has a unique solution [CG17b, Lemma 4.1]. The the output state $|\vec{y}\rangle$ of the QLS algorithm then corresponds to this unique solution of the linear system, and the solution of $\mathcal{F}$ can be efficiently obtained from measuring the state $|\hat{y}\rangle$.

Classically, the solving degree of $\mathcal{F}$ is used, which is at most $n + 2$ as shown by Caminata and Gorla [CG17a, Theorem 3.26]. This allows computing the Gröbner basis of the polynomial ideal $\langle \mathcal{F} \rangle$ via Gaussian elimination of the linear system, and the solution of $\mathcal{F}$ can be obtained from the Gröbner basis. However, for some polynomial systems, the affine subspace of all solutions of the linear system has no well-understood structure even though $\mathcal{F}$ has a unique solution. In this case, the QLS algorithm outputs a state $|\hat{y}\rangle$ that corresponds to the smallest ℓ_2-norm solution of the linear system. In general we don't know how to extract the solution of $\mathcal{F}$ from such states $|\hat{y}\rangle$.

When $\mathcal{F}$ has more than one solution and the max degree is set to be $3n$, the dimension of the affine subspace of all solutions of the linear system equals the number of solutions of $\mathcal{F}$ minus 1. For each solution $a \in \{0, 1\}^n$ of $\mathcal{F}$, there is a corresponding solution $\hat{y}_a$ of the linear system. Those solutions $\hat{y}_a$ are linearly independent and any solution of the linear system is an affine combination of the solutions $\hat{y}_a$ [CG17b, Lemma 3.18]. Again, the QLS algorithm outputs a state $|\hat{y}\rangle$ corresponding to the smallest ℓ_2 norm vector $\hat{y}$ in this affine subspace.

Chen and Gao [CG17b] showed that $\mathcal{M}$ is an $\mathcal{O}(m \cdot \#\mathcal{F})$-sparse, row computable matrix and $\vec{b}$ can be efficiently prepared as a quantum state. In particular, assuming $|\mathcal{F}| = \mathcal{O}(\text{poly}(n))$, they show that the QLS algorithm can be run in time $\widetilde{\mathcal{O}}(\text{poly}(n)\kappa(\mathcal{M}))$. There is strong complexity theoretic evidence that in general running the QLS algorithm requires time $\widetilde{\Omega}(\kappa(\mathcal{M}))$ [HHL09], so a lower bound on the condition number also lower bounds the running time.

4.2 Lower bound on the truncated QLS condition number $\kappa_{\vec{b}}(\mathcal{M})$

In this section we give a lower bound on the tQLScn $\kappa_{\vec{b}}(\mathcal{M})$. Since $\kappa_{\vec{b}}(\mathcal{M}) \leq \kappa(\mathcal{M})$, this also implies a lower bound on the time complexity of Chen and Gao's [CG17b] algorithm.

In order to prove a lower bound on $\kappa_{\vec{b}}(\mathcal{M})$, it suffices to lower bound the length of the solution vector $\vec{y} = \mathcal{M}^+\vec{b}$, since

$$\kappa_{\vec{b}}(\mathcal{M}) = \|\mathcal{M}\| \frac{\left\|\mathcal{M}^+\vec{b}\right\|}{\left\|\vec{b}\right\|} \geq \left\|\mathcal{M}^+\vec{b}\right\| = \|\vec{y}\|. \tag{9}$$

Here, the first equality is the definition of $\kappa_{\vec{b}}$, and the inequality follows from $\left\|\vec{b}\right\| = 1$, and because

$\|\mathcal{M}\| \geq 1$, as $\mathcal{M}$ has at least one matrix element which has absolute value at least 1.[5]

In order to understand the length $\|\vec{y}\|$ of the solution vector $\vec{y} = \mathcal{M}^+\vec{b}$, let us first study the monomial solution vector $\vec{y}^{(a)}$ corresponding to a binary solution a. For a degree-d Macaulay linear system a monomial exponent $0 \neq e \in \mathbb{N}^n$, is a "valid" coordinate of $\vec{y}^{(a)}$ if $e \in \{0, 1, \ldots, d\}^n$ (and $\sum e_i \leq d$ for total degree), moreover the solution vector satisfies $\vec{y}_e^{(a)} = \Pi_i a_i^{e_i}$, which is 1 if and only if $a_i = 1$ for all variables x_i in the monomial $\Pi_i x_i^{e_i}$ indexed by e. If the Hamming weight of a is h and $\mathcal{M}$ is constructed with max degree, then the number of such non-zero coordinates (monomials) is $(d+1)^h - 1$, thus

$$\left\|\vec{y}^{(a)}\right\|^2 = (d+1)^h - 1. \tag{10}$$

When $\mathcal{M}$ is constructed with total degree, the number of such non-zero coordinates (monomials) is $\binom{d+h}{h} - 1$, so

$$\left\|\vec{y}^{(a)}\right\|^2 = \binom{d+h}{h} - 1. \tag{11}$$

Suppose $a_1, a_2, \cdots, a_t \in \{0,1\}^n$ are the t solutions of $\mathcal{F} = \mathcal{F}_1 \cup \mathcal{F}_2$, where $\mathcal{F}_1 = \{f_1, \ldots, f_m\}$ and $\mathcal{F}_2 = \{x_1^2 - x_1, \ldots, x_n^2 - x_n\}$. The t solutions $a_1, a_2, \cdots, a_t$ of $\mathcal{F}$ must be nonzero because the first equation has constant term $b_1 = 1$. Let $\vec{y}_1, \vec{y}_2, \cdots, \vec{y}_t$ be the $0/1$ solution vectors of the linear system $\mathcal{M}\vec{y} = \vec{b}$ under the assignments $a_1, a_2, \cdots, a_t$ respectively. When the max degree of the linear system is set to be $3n$, the affine subspace of all solutions of the linear system is spanned by the monomial solution vectors $\vec{y}_1, \vec{y}_2, \cdots, \vec{y}_t$ [CG17b, Theorem 3.21 and Lemma 4.1], but this property might also hold for lower degrees. From now on we assume that degree d is such that the linear system has this property, then $\vec{y} = \mathcal{M}^+\vec{b}$ has the minimum ℓ_2 norm in the affine subspace spanned by $\vec{y}_1, \vec{y}_2, \cdots, \vec{y}_t$.

If all the t solutions $a_1, a_2, \cdots, a_t \in \{0,1\}^n$ have the same Hamming-weight, we can lower bound the length $\|\vec{y}\|$ of the solution vector $\vec{y} = \mathcal{M}^+\vec{b}$ by the following lemma.

Lemma 4.3. *Suppose $\vec{y}_1, \vec{y}_2, \cdots, \vec{y}_t$ are vectors with 0 and 1 entries such that their Hamming weights are equal. Then every vector in their (complex) affine hull A has length (ℓ_2 norm) at least $\|\vec{y}_1\|/\sqrt{t}$.*

Proof. Every entry of the vectors $\vec{y}_i$ is either 0 or 1, therefore $\langle \vec{y}_i, \mathbf{1} \rangle = \|\vec{y}_i\|_1$ (we denote by $\mathbf{1}$ the all-1 vector). Since the vectors have the same Hamming weight we also have $\|\vec{y}_1\|_1 = \|\vec{y}_i\|_1$ for all $i \in [t]$. Let $\vec{y}$ be any vector in A, then $\langle \vec{y}, \mathbf{1} \rangle = \langle \vec{y}_1, \mathbf{1} \rangle$, and in particular $\|\vec{y}\|_1 \geq \langle \vec{y}, \mathbf{1} \rangle = \|\vec{y}_1\|_1 = \|\vec{y}_1\|_2^2$. Let $\|\vec{y}\|_0$ denote the support size of $\vec{y}$. Then $\|\vec{y}\|_0 \leq \sum_{i=1}^t \|\vec{y}_i\|_0 = t\|\vec{y}_1\|_0 = t\|\vec{y}_1\|_1 = t\|\vec{y}_1\|_2^2$. By the Cauchy-Schwarz inequality we have that

$$\|\vec{y}\|_1 \leq \|\vec{y}\|_2 \sqrt{\|\vec{y}\|_0} \implies \|\vec{y}\|_2 \geq \frac{\|\vec{y}\|_1}{\sqrt{\|\vec{y}\|_0}} \geq \frac{\|\vec{y}_1\|_2^2}{\sqrt{t}\|\vec{y}_1\|_2} = \frac{\|\vec{y}_1\|_2}{\sqrt{t}}. \qquad \square$$

If the minimum ℓ_2-norm solution $\vec{y} = \hat{\mathcal{M}}^+\vec{b}$ happens to be a convex combination $\vec{y} = \sum_{i=1}^t w_i \vec{y}_i$ of the (possibly differing Hamming weight) solution vectors $\vec{y}_1, \vec{y}_2, \cdots, \vec{y}_t$, then we can similarly lower bound the length $\|\vec{y}\|$.

Lemma 4.4. *Suppose $\vec{y}_1, \vec{y}_2, \cdots, \vec{y}_t$ are $0/1$ vectors and $\vec{y}_1$ has the minimum Hamming weight. Then every vector in their convex hull A has length (ℓ_2-norm) at least $\|\vec{y}_1\|/\sqrt{t}$.*

---

[5]For the Macaulay matrix construction, let f be $x^2 - x$, the row $(1, f)$ will have a matrix element of magnitude 1.

Proof. Let $\vec{y} = \sum_{i=1}^{t} w_i \vec{y}_i$ be an arbitrary vector in the convex hull A generated by $\vec{y}_1, \vec{y}_2, \cdots, \vec{y}_t$, where $\sum_{i=1}^{t} w_i = 1$ and $w_i \geq 0$. Then

$$\|\vec{y}\|^2 = \sum_{i=1}^{t} \sum_{j=1}^{t} w_i w_j \langle \vec{y}_i, \vec{y}_j \rangle \geq \sum_{i=1}^{t} w_i^2 \langle \vec{y}_i, \vec{y}_i \rangle \geq \sum_{i=1}^{t} w_i^2 \langle \vec{y}_1, \vec{y}_1 \rangle \geq \langle \vec{y}_1, \vec{y}_1 \rangle / t.$$

The first equality is by the definition of $\|\vec{y}\|^2 = \langle \vec{y}, \vec{y} \rangle$. The first inequality is because $w_i w_j \langle \vec{y}_i, \vec{y}_j \rangle \geq 0$ for any pair $i, j \in [t]$. The second inequality is true because $\langle \vec{y}_i, \vec{y}_i \rangle \geq \langle \vec{y}_1, \vec{y}_1 \rangle$ for any $i \in [t]$ as $\vec{y}_i$ are a 0/1 vectors and $\vec{y}_1$ has the minimum Hamming weight. The third inequality follows from Cauchy-Schwarz. $\qquad\square$

Combining (9) with (10)-(11), Lemma 4.3 and Lemma 4.4, we get our first lower bound result.

Theorem 4.5. *Suppose $a_1, a_2, \cdots, a_t \in \{0, 1\}^n$ are the t solutions of $\mathcal{F} = \mathcal{F}_1 \cup \mathcal{F}_2$, where $\mathcal{F}_1 = \{f_1, \ldots, f_m\}$ and $\mathcal{F}_2 = \{x_1^2 - x_1, \ldots, x_n^2 - x_n\}$, and let h be the minimum Hamming weight of the t solutions $a_1, a_2, \cdots, a_t$. Let $\vec{y}_1, \vec{y}_2, \cdots, \vec{y}_t$ be the corresponding solution vectors of the Macaulay linear system $\mathcal{M}\vec{y} = \vec{b}$ under the assignments $a_1, a_2, \cdots, a_t$ respectively.*

If all the t solutions $a_1, a_2, \cdots, a_t$ have the same Hamming weight h or the minimum ℓ_2-norm solution vector $\vec{y} = \mathcal{M}^+ \vec{b}$ is in the convex hull of $\vec{y}_1, \vec{y}_2, \cdots, \vec{y}_t$, then the tQLScn of $\mathcal{M}$ of $\mathcal{F}$ in the Macaulay linear system

- *using max degree is $\kappa_{\vec{b}}(\mathcal{M}) \geq \sqrt{((d+1)^h - 1)/t}$, and*

- *using total degree is $\kappa_{\vec{b}}(\mathcal{M}) \geq \sqrt{\left(\binom{d+h}{h} - 1\right)/t}$.*

In particular in the setup in [CG17b], using max degree $d = 3n$, we have $\kappa_{\vec{b}}(\mathcal{M}) \geq \sqrt{(3n)^h / t}$.

Now we give a lower bound in terms of the smallest Hamming weight in the binary solution set. For this we use the following purely geometrical lemma.

Lemma 4.6 (Shortest vector within an affine subspace). *Let $V \in \mathbb{C}^{n \times k}$ be a matrix with columns $v_1, v_2, \ldots, v_k$, and let A be their (complex) affine hull $A := \{Vx : x \in \mathbb{C}^k \text{ s.t. } \langle x, \mathbf{1} \rangle = 1\}$. Then A contains the origin iff the column space of the Gram matrix $G = V^\dagger V$ does not contain $\mathbf{1}$. Moreover, the length-square γ^* of the shortest vector (with respect to the ℓ_2-norm) in A is*

$$\gamma^* = \max\{\gamma \colon G - \gamma \mathbf{1} \cdot \mathbf{1}^T \succeq 0\}. \tag{12}$$

Furthermore, if A does not contain the origin, then $\gamma^* = 1/\langle \mathbf{1}, G^+ \mathbf{1} \rangle$ and the shortest vector in A is $V \cdot w$ for $w = G^+ \mathbf{1} / \langle \mathbf{1}, G^+ \mathbf{1} \rangle$.

Note that the problem of finding the length of the shortest vector in an affine subspace can also be reformulated as the following SDP:

$$\gamma^* = \min_{\rho \succeq 0} \mathrm{Tr}(G\rho) \qquad \text{subject to: } \mathrm{Tr}(\mathbf{1} \cdot \mathbf{1}^T \rho) = 1,$$

where without loss of generality we can assume that the above optimizer ρ has rank 1.

By the weak duality of SDPs, and utilizing the dual of the above problem, we get:

$$\gamma^* \geq \max_{\gamma \in \mathbb{R}} \gamma \qquad \text{subject to: } G - \gamma \mathbf{1} \cdot \mathbf{1}^T \succeq 0.$$

In fact the following proof of Lemma 4.6 shows that the above inequality is tight, i.e., strong duality always holds for this SDP.

Proof. The shortest vector s in the affine subspace is orthogonal to all vectors of the form $v_i - v_j$, therefore we must have that $\langle s, v_i \rangle$ is constant for all $i \in [k]$, i.e., $V^\dagger s \propto \mathbf{1}$. Since s is in the column space of V, we can write it in the form $s = V \cdot w$ so we get $V^\dagger s = V^\dagger V \cdot w = Gw \propto \mathbf{1}$. If $s \neq 0$ then this implies that $\mathbf{1}$ is in the column space of G, consequently $0 \neq \langle \mathbf{1}, G^+\mathbf{1} \rangle$ and thereby $s = V \cdot w$ for $w = G^+\mathbf{1}/\langle \mathbf{1}, G^+\mathbf{1} \rangle$.[6] From this we can conclude $\gamma^* = \|s\|^2 = \langle w, Gw \rangle = 1/\langle \mathbf{1}, G^+\mathbf{1} \rangle$.

Conversely, if $\mathbf{1}$ is in the column space of G then $s = V \cdot w$ for $w = G^+\mathbf{1}/\langle \mathbf{1}, G^+\mathbf{1} \rangle$ is a non-zero vector, which is the shortest vector in A due to the fact that it is orthogonal to all vectors of the form $v_i - v_j$. We can conclude that $\gamma^* \neq 0$ iff $\mathbf{1}$ is in the column space of G.

If $\gamma^* > 0$, we can formulate a "dual" optimization problem the following way: $\gamma^* = \max\{\gamma \colon 1 - \gamma/\gamma^* \geq 0\} = \max\{\gamma \colon 1 - \gamma\langle \mathbf{1}, G^+\mathbf{1} \rangle \geq 0\} = \max\{\gamma \colon I - \gamma\sqrt{G^+}\mathbf{1} \cdot \mathbf{1}^T\sqrt{G^+} \succeq 0\}$. By multiplying the matrices with $\sqrt{G}$ from both sides we get the following equivalent maximization formulation

$$\gamma^* = \max\{\gamma \colon G - \gamma\sqrt{G}\sqrt{G^+}\mathbf{1} \cdot \mathbf{1}^T\sqrt{G^+}\sqrt{G} \succeq 0\}. \tag{13}$$

Note that $\sqrt{G^+}\sqrt{G} = \sqrt{G}\sqrt{G^+} = (G^+G)$ is the orthogonal projector to the column space of G. Since $\mathbf{1}$ is in the image of G the above equation (13) is equivalent to (12).

On the other hand, if $\gamma^* = 0$, then $\mathbf{1}$ is not in the column space of G and so $(I - G^+G)\mathbf{1} \neq 0$. Observe that $G - \gamma\mathbf{1} \cdot \mathbf{1}^T \succeq 0$ implies $(I - G^+G)G(I - G^+G) - \gamma(I - G^+G)\mathbf{1} \cdot \mathbf{1}^T(I - G^+G) \succeq 0$ or equivalently $-\gamma(I - G^+G)\mathbf{1} \cdot \mathbf{1}^T(I - G^+G) \succeq 0$, which then only holds for $\gamma \leq 0 = \gamma^*$. Consequently, (12) holds even in the case $\gamma^* = 0$. $\qquad\square$

Suppose that the Boolean solution set of the polynomial system is $S = \{a_1, a_2, \cdots, a_t\}$. Let A_S be the affine subspace corresponding to the solution set S, spanned by the monomial solution vectors $y_1, y_2, \cdots, y_t$ of the linear system $\mathcal{M}\vec{y} = \vec{b}$ corresponding to the Boolean solutions $a_1, a_2, \cdots, a_t$ respectively. We wish to lower bound the length of the shortest vector in A_S; for this it suffices to find the length of the shortest vector in an enlarged affine subspace $A_{S'} \supseteq A_S$.

Let S' be the symmetrized solution set of S by applying all possible permutations of the variables of a_i's and taking their union. Let $A_{S'}$ be the affine subspace corresponding to the solution set S' and let v be the shortest vector in $A_{S'}$. For each Hamming weight h that appears in S', there is a symmetrized monomial solution vector $\mathbf{v}_h$. This $\mathbf{v}_h$ equals the average over all monomial solution vectors that are associated to Boolean solutions of Hamming weight h.

Next we will argue that the minimum ℓ_2-norm vector $v \in A_{S'}$ is an affine combination of the symmetrized monomial solution vectors $\mathbf{v}_h$. If we apply an induced permutation on the coordinates of v according to a permutation of the Boolean variables, then the ℓ_2-norm of the resulting vector $u \in A_{S'}$ is equal to the ℓ_2-norm of v. Because v has the minimum ℓ_2-norm in $A_{S'}$ we have $\left\|\frac{u+v}{2}\right\|_2 \geq \|v\|$, and due to $\|u\| = \|v\|$ by the triangle inequality $\left\|\frac{u+v}{2}\right\|_2 \leq \|v\|$ (equality holds if and only if $u = v$), the resulting vector u is equal to v. Therefore, the shortest vector v is invariant under all the possible induced permutations, therefore we can conclude that v is an affine combination of the symmetrized monomial solution vectors $\mathbf{v}_h$.

Now, we can lower bound $\left\|\mathcal{M}_\mathcal{F}^+b\right\|$ by finding the lowest ℓ_2 norm of a vector in the affine subspace spanned by the symmetrized vectors $\mathbf{v}_h$ corresponding to the Hamming weights h that appear in S. This can be achieved by considering the Gram matrix as explained in Lemma 4.6.

In order to compute this Gram matrix, we need to understand the symmetrized vectors $\mathbf{v}_h$. For this, let us introduce the following orthonormal vector system $(\mathbf{b}_s)$, corresponding to the set

---

[6]For the last implication note that we already showed $w = \beta G^+\mathbf{1} + v$, where $v \in \ker(G)$. As $\ker(G) = \ker(V)$ we can assume without loss of generality that $v = 0$. It is easy to see that $VG^+\mathbf{1}/\langle \mathbf{1}, G^+\mathbf{1} \rangle \in A$, and since A is an affine subspace not containing the origin, it can only contain one vector of the form $\beta VG^+\mathbf{1} \colon \beta \in \mathbb{C}$, thus $s = VG^+\mathbf{1}/\langle \mathbf{1}, G^+\mathbf{1} \rangle$. (Indeed, if two distinct vectors x, y are in A and $y = \lambda x$, then $\frac{1}{1-\lambda}y - \frac{\lambda}{1-\lambda}x = 0$ is also in A.)

of monomials $\mathfrak{m}_s^d$ that contain exactly s variables with a non-zero exponent, with the degree of the monomials being at most d, then $\mathbf{b}_s := \frac{1}{\sqrt{|\mathfrak{m}_s^d|}} \sum_{m \in \mathfrak{m}_s^d} e_m$. Also let $\Pi_{\mathfrak{m}_s^d} = \sum_{m \in \mathfrak{m}_s^d} e_m \cdot e_m^T$ be the projector to coordinates in $\mathfrak{m}_s^d$. Finally, let c_s^d be the number of monomials that contain s specific variables with a non-zero exponent and have degree at most d, so that $|\mathfrak{m}_s^d| = \binom{n}{s} c_s^d$.

One can see that for any $a \in \{0,1\}^n$ of Hamming weight h we have $c_s^d \binom{h}{s} = \left\langle \mathbf{1}, \Pi_{\mathfrak{m}_s^d} \vec{y}^{(a)} \right\rangle = \left\langle \mathbf{1}, \Pi_{\mathfrak{m}_s^d} \mathbf{v}_h \right\rangle$. Since $\mathbf{v}_h$ has uniform coordinates over $\mathfrak{m}_s^d$ we have

$$\left\| \Pi_{\mathfrak{m}_s^d} \mathbf{v}_h \right\|^2 = \|\mathfrak{m}_s^d\| \left(\frac{\left\langle \mathbf{1}, \Pi_{\mathfrak{m}_s^d} \mathbf{v}_h \right\rangle}{\|\mathfrak{m}_s^d\|} \right)^2 = c_s^d \binom{n}{s} \left(\frac{c_s^d \binom{h}{s}}{c_s^d \binom{n}{s}} \right)^2 = c_s^d \binom{h}{s}^2 / \binom{n}{s},$$

consequently

$$\mathbf{v}_h = \sum_{s=1}^{h} \sqrt{c_s^d \binom{h}{s}^2 / \binom{n}{s}} \mathbf{b}_s, \tag{14}$$

and the Gram matrix G of the symmetrized vectors has matrix elements

$$G_{ij} = \langle \mathbf{v}_i, \mathbf{v}_j \rangle = \sum_{s=1}^{n} c_s^d \binom{i}{s} \binom{j}{s} / \binom{n}{s}.$$

Together with Lemma 4.6 this enables us to give a lower bound on the smallest ℓ_2-norm solution in terms of the minimal Hamming weight appearing in the solution set as follows:

Theorem 4.7. *Suppose that $\mathcal{F}$ is a Boolean polynomial system with n Boolean variables where each solution have Hamming weight at least h, and $d \geq n$. Then the degree-d Macaulay linear system's tQLScn is lower bounded by*

$$\frac{1}{\left\langle \mathbf{1}, (G^{(h)})^{-1} \mathbf{1} \right\rangle} = \max\{\gamma : G^{(h)} - \gamma \mathbf{1} \cdot \mathbf{1}^T \succeq 0\}, \tag{15}$$

where $G \in \mathbb{R}^{n \times n}$ is the Gram matrix whose (i,j) matrix element is

$$G_{ij} = \sum_{s=1}^{n} c_s^d \binom{i}{s} \binom{j}{s} / \binom{n}{s}, \tag{16}$$

and $c_s^d = d^s$ for max degree, while $c_s^d = \binom{d}{s}$ for total degree, finally $G^{(h)}$ is the bottom-right $(n-h+1)$ by $(n-h+1)$ minor of G.[7]

The expression in (15) is difficult to bound analytically, but it appears to be exponentially large in terms of h for large enough d. In particular, we could verify[8] that for max degree $d = 3n$ Equation (15) is lower bounded by $h^h/2$ for every $h \in [n]$ up to $n = 300$.

---

[7]When $d \geq n$, due to the triangular shape of the non-zero coefficients of the vectors in (14) it is easy to see that G has full rank, i.e., it is positive definit. It follows, that all minors of G are also positive definit, i.e., have full rank, therefore $G^{(h)}$ is invertible.

[8]Aided by symbolic computations executed by Mathematica 12.3 on Linux. See Appendix C for the code.

4.3 Comparison to brute-force search

In case there is a unique Boolean solution we showed that the lower bound of the running time of the quantum algorithm using the HHL algorithm is exponential in the Hamming weight of the unique Boolean solution, and we provided strong evidence that this is also true when there are multiple solutions.

It is useful to compare the HHL-based approach to classical brute-force search and also to using Grover's algorithm. In case we know that the unique solution has Hamming weight h, we can simply classically search through all the $\binom{n}{h}$ different Hamming-weight-h assignments of the original polynomial system. We can also use Grover search to find such an assignment with $\mathcal{O}\left(\sqrt{\binom{n}{h}}\right)$ evaluations of the polynomials. Even if we do not a priory know the Hamming weight, we can classically iterate over increasing Hamming weights w of n-bit strings which requires at most $\mathcal{O}\left(\sum_{w=0}^{h}\binom{n}{w}\right)$ different possible assignments to be checked before finding the solution, which in the case $h \leq n/2$ can be bounded by $\mathcal{O}\left(\sqrt{h}\binom{n}{h}\right)$ as we show in Appendix B. For the $h > n/2$ case, a similar complexity can be achieved by searching through decreasing Hamming weights. In the quantum case, naively iterating through increasing Hamming weights and using Grover's algorithm for each weight gives a complexity bound of $\mathcal{O}\left(h\sqrt{\binom{n}{h}}\right)$.

Moreover, we can use a slight variant of Grover's algorithm for searching through an unknown sized search space[9] which requires only $\mathcal{O}\left(\sqrt[4]{h}\sqrt{\binom{n}{h}}\right)$ evaluations of the polynomials. By comparing this to the lower bounds of Theorem 4.5 on can see that in case $d+h \geq n$ Grover's algorithm performs at least as good as the HHL based algorithm (up to some potential lower order correction $\sqrt[4]{h}$), and the algorithm of Chen and Gao [CG17b] where the max degree $d = 3n$ is definitely outperformed by Grover search.

In case there are multiple solutions, but all their Hamming weights are the same, Theorem 4.5 ensures that we do not get a bigger reduction in the condition number than the analogous speedup we can already achieve by plain Grover search. So the above Grover-based algorithm still performs just as competitively.

In the general case of having multiple solutions with different Hamming weights, the situation is harder to analyze, but we could still obtain an exponential lower bound on tQLScn in terms of the smallest Hamming weight solution up to $n = 300$, providing strong evidence for Chen and Gao's algorithm [CG17b] having a best case complexity that is exponentially large in terms of the minimal Hamming weight of a solution, making it unlikely that their algorithm would give a substantial improvement over brute-force Grover search.

5 The Boolean Macaulay linear system and its tQLScn

In this section we give an equivalent but more efficient way to represent the Macaulay matrix using the fact that we are only searching for 0/1 solutions in $\mathbb{C}$. This results in a smaller lower bound on the tQLScn of size $\Omega(2^{h/2})$. While the quantum algorithm's running time is still exponentially large for larger Hamming weight solutions, for Hamming weight $h = \Theta(\log n)$ the smaller lower bound

---

[9]One can use an algorithm analogous to the "exponential Grover search" [BBHT98] in order to check for a unique solution in subsequently enlarged search spaces corresponding to larger and larger Hamming-weights. By carefully choosing the sequence of upper bounds on the Hamming weights such that the search space expands in each consecutive iteration by a bounded multiplicative factor in $[c, C] \subset (1, \infty)$ the claimed running time bound follows.

leaves open the possibility of a quasipolynomial speedup compared to the classical brute-force search algorithm having running time $\mathcal{O}\big(\binom{n}{h}\big)$.

5.1 The Boolean Macaulay matrix over $\mathbb{C}$

In this section we again include the field polynomials $\mathcal{F}_2 = \{x_1^2 - x_1, \ldots, x_n^2 - x_n\}$ for the field $\mathbb{F}_2$ together with the input polynomials $\mathcal{F}_1$. Solving the system $\mathcal{F} = \mathcal{F}_1 \cup \mathcal{F}_2$ forces the roots to be effectively Boolean even though the underlying field is $\mathbb{C}$. This allows all monomials in an equation to be replaced with equivalent multilinear versions and a reduced Macaulay matrix will be defined that has a more compact form. This was done in [BFSS13] for finite fields, where the extra equations prevented solutions from being in field extensions. We derive the analogous matrix when the solutions are forced to be Boolean but the arithmetic is over $\mathbb{C}$. Additionally, in our case (similarly to Chen and Gao's original construction [CG17b]) the structure of the Boolean solutions makes it possible to extract the Boolean solutions from measuring the quantum state corresponding the the solution vectors (over $\mathbb{C}$).

Let $\psi : R \to R$ map a monomial to its multilinear image as $\psi(\prod_{i=1}^{n} x_i^{a_i}) = \prod_{i=1}^{n} x_i^{\min\{a_i,1\}}$, and extend it to $R = \mathbb{C}[x_1, \ldots, x_n]$ by linearity. For example, $\psi(3x_1^3 x_2 - 1) = 3x_1 x_2 - 1 = \psi(x_1^3 x_2 - 2x_1^2 x_2^2 + 4x_1 x_2 - 1) \neq x_1 - x_1 x_2 - 1 = \psi(x_1^3 - 2x_1^2 x_2^2 + x_1 x_2 - 1)$.

Lemma 5.3 will show that having max degree higher than 1 becomes redundant, so the following definition only has rows up to max degree 1, and in this section we will set the total degree $d = n$, the number of variables. For notation, let $\hat{m}$ and $\hat{m}'$ denote monomials, and let m, m', and m'' denote multilinear monomials (i.e., monomials with max degree at most 1).

Definition 5.1. *The* Boolean Macaulay matrix $\hat{\mathcal{B}}$ *of degree* d *of* $\mathcal{F}_1 = \{f_1, \ldots, f_m\} \subseteq \mathbb{C}[X]$ *is the matrix where each row is labeled by a pair of polynomials* (m, f) *and contains the corresponding coefficient vector of the polynomial* $\psi(mf)$. *The rows range over all* $f \in \mathcal{F}_1$ *and multilinear monomials* m *such that* $\psi(mf)$ *has degree at most* d. *The columns are labeled by the set of multilinear monomials in* $x_1, \ldots, x_n$ *of degree at most* d *and are ordered with respect to a specified monomial ordering. The matrix element in the row corresponding to* (m, f) *and the column corresponding to the monomial* m' *is the coefficient of* m' *in the polynomial* $\psi(mf)$.

Note that compared to the Macaulay matrix, in addition to forcing answers to be Boolean, the Boolean Macaulay matrix is reduced in a certain way, by eliminating polynomials with max degree at least 2. Next we will show that the Boolean Macaulay matrix can be obtained as a submatrix of the Macaulay matrix $\hat{\mathcal{M}}$ of max degree d corresponding to the set of polynomials $\mathcal{F} = \mathcal{F}_1 \cup \mathcal{F}_2$ after Gaussian reduction on the rows.

First we consider the special case when $\mathcal{F}_1 = \emptyset$ and perform the row reduction on the Macaulay matrix of $\mathcal{F}_2$ to show that the field equations $\mathcal{F}_2$ take a special form.

Lemma 5.2. *Let the Macaulay matrix* $\hat{\mathcal{M}}_2$ *of max degree* d *of* $\mathcal{F}_2$ *have its columns ordered such that they are partitioned into two parts the following way: the labels on the right side are multilinear monomials (including the degree 0 monomial 1) and ordered in ascending order with respect to the integer represented by the exponent vector of the multilinear monomial, and let the left side columns be labeled by nonmultilinear monomials and ordered under any monomial order.*

Then using row operations, $\hat{\mathcal{M}}_2 = [L_2 \quad R_2]$ *can be reduced to* $\hat{\mathcal{M}}_2' = [I_2 \quad B_2]$ *where* I_2 *is the identity matrix of dimension* $(d+1)^n - 2^n$ *with rows and columns labeled by nonmultilinear monomials, and rows with zeros are removed.*

Proof. The rows of $\hat{\mathcal{M}}_2$ are indexed by pairs of polynomials $(\hat{m}, x_j^2 - x_j)$, where $\hat{m}$ is a monomial and $\max\deg \hat{m}(x_j^2 - x_j) \leq d$. The approach is to first change each row, which starts with coefficients for a polynomial $\Pi_i x_i^{a_i} - x_j^{-1}\Pi_i x_i^{a_i}$ for some j in $\hat{\mathcal{M}}_2$, to the coefficients of $\Pi_i x_i^{a_i} - \Pi_i x_i^{\min\{a_i,1\}}$. At this point the left side of the matrix has at most one 1 in each row. The second step is to zero out the bottom rows.

For the first step, work in descending total degree of the polynomials, starting at degree nd. Let the current row have the coefficients of $\Pi_i x_i^{a_i} - \Pi_i x_i^{b_i}$ during the algorithm. Let $b_j \geq 2$ for some j, or else this row is reduced. Because $\deg \Pi_i x_i^{b_i} < \deg \Pi_i x_i^{a_i}$, the rows for $\Pi_i x_i^{b_i}$ have not changed yet, and therefore, one of the rows has the coefficients of $\Pi x_i^{b_i} - x_j^{-1}\Pi x_i^{b_i}$. Adding this row changes the current row to $\Pi_i x_i^{a_i} - \Pi_i x_i^{b_i} + (\Pi_i x_i^{b_i} - x_j^{-1}\Pi_i x_i^{b_i}) = \Pi_i x_i^{a_i} - x_j^{-1}\Pi_i x_i^{b_i}$, which has decreased the total degree of the second term by one while keeping the set of variables the same. This is repeated until the row has the coefficients of $\Pi_i x_i^{a_i} - \Pi_i x_i^{\min\{a_i,1\}}$.

At the end of the first step, each column in the left side (i.e., columns indexed by nonmultilinear monomials) has at most one 1. This is in fact a constructive argument showing that

$$\prod_{i=1}^{n} x_i^{a_i} - \prod_{i=1}^{n} x_i^{\min\{a_i,1\}} \in \langle \mathcal{F}_2 \rangle.$$

Consider any two rows indexed by $\hat{m}(x_i^2 - x_i)$ and $\hat{m}'(x_j^2 - x_j)$. If $\hat{m}x_i^2 = \hat{m}'x_j^2$, they have the same set of variables, and therefore $\psi(\hat{m}x_i) = \psi(\hat{m}'x_j)$, so the rows are equal and one can be eliminated (zeroed out). Keep doing this until for every leading nonmultilinear monomial there is only one row where the corresponding coefficient is nonzero.

Because for every column in the left part there is a unique nonzero row with the corresponding leading monomial, the matrix can be written (up to permutation of the rows) as $\begin{bmatrix} I_2 & B_2 \\ 0 & 0 \end{bmatrix}$. $\square$

Lemma 5.3. *Let* $\hat{\mathcal{M}} = \begin{bmatrix} L_1 & R_1 \\ L_2 & R_2 \end{bmatrix}$ *be the Macaulay matrix for* $\mathcal{F}_1 \cup \mathcal{F}_2$ *with the row and column ordering from Lemma 5.2. Using row operations (and then removing some zero rows),* $\hat{\mathcal{M}}$ *can be reduced to* $\hat{\mathcal{M}}' = \begin{bmatrix} 0 & \hat{\mathcal{B}} \\ I_2 & B_2 \end{bmatrix}$ *where* $\hat{\mathcal{B}}$ *is the Boolean Macaulay matrix of* $\mathcal{F}_1$, *and* I_2, B_2 *are as in Lemma 5.2.*

Proof. By Lemma 5.2, using row operations on the $\mathcal{F}_2$ submatrix of $\hat{\mathcal{M}}$ we get a matrix $\begin{bmatrix} L_1' & R_1' \\ I_2 & B_2 \end{bmatrix}$, where zero rows from $\mathcal{F}_2$ are removed.

Row operations utilizing I_2 can then be used to zero out the top left, resulting in $\begin{bmatrix} 0 & R_1'' \\ I_2 & B_2 \end{bmatrix}$. From the polynomial perspective, this maps all the nonmultilinear polynomials to their corresponding multilinear polynomials under ψ, i.e., for each monomial $\prod_{i=1}^{n} x_i^{a_i}$, the map encodes the coefficient vector of $\psi((\prod_{i=1}^{n} x_i^{a_i})f_j), 1 \leq j \leq m$ into the Macaulay matrix as a row vector.

Recall that the Macaulay matrix has rows labeled by pairs; observe that rows $(\hat{m}, f_i)$ and $(\hat{m}', f_i)$ will be equal when $\psi(\hat{m}) = \psi(\hat{m}')$. In particular for any nonmultiliear monomial $\hat{m}$ the rows $(\hat{m}, f_i)$ and $(\psi(\hat{m}), f_i)$ will be equal at this point, so we can eliminate (zero out and then remove) any row indexed by nonmultiliear monomials. To be compatible with Definition 5.1 we choose not to further reduce / remove rows despite the fact $\hat{\mathcal{B}}$ might have zero rows, for example, rows with $\psi(\hat{m}f) = \psi(\hat{m}'f')$, but $f \neq f'$.

15

As claimed, in matrix notation we get $\hat{\mathcal{M}}' = \begin{bmatrix} 0 & \hat{\mathcal{B}} \\ I_2 & B_2 \end{bmatrix}$. $\qquad\square$

As in the general case let $\hat{\mathcal{B}} = [M \quad -\vec{b}]$ define the Boolean Macaulay linear system as $M\vec{y} = \vec{b}$, where the entries of $\vec{y}$ are labeled by the nontrivial multilinear monomials and $\vec{b} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$.

Recall that a matrix is s-sparse if it has at most s entries in any row or column.

Lemma 5.4. *The Boolean Macaulay matrix $\hat{\mathcal{B}}$ of total degree d of $\mathcal{F}_1$ is an $\mathcal{O}(m \cdot \#\mathcal{F}_1)$-sparse matrix.*

Proof. The Boolean Macaulay matrix $\hat{\mathcal{B}}$ is constructed by placing $\psi(tf)$ in a row for a multilinear monomial t and $f \in \mathcal{F}_1$, so the support of each row has size at most $\mathcal{O}(\#\mathcal{F}_1)$.

For the column sparsity first consider the Boolean Maculay matrix of $\{t\}$, which has a 1 matrix element at column m' and row (m'', t) if and only if $m' = \psi(m'' \cdot t)$. This can only happen if t divides m', so we can define $\bar{t} := m'/t$. It is easy to see that $m' = \psi(m'' \cdot t)$ if and only if $m'' = \bar{t} \cdot m_d$ for some monomial m_d that divides t. This implies that the column sparsity of the Boolean Maculay matrix of $\{t\}$ equals the number of divisors of t which is at most 4 if the multilinear monomial t has (total) degree at most 2.

Now consider the Boolean Maculay matrix of $\{f\}$ for some (at most) quadratic polynomial $f \in \mathcal{F}_1$. Observe that f is a linear combination of at most $\#\mathcal{F}_1$ monomials of degree at most 2 and the Boolean Maculay matrix of $\{f\}$ is likewise the linear combination of the Boolean Maculay matrix of these (at most) quadratic monomials. So the column sparsity of the Boolean Maculay matrix of $\{f\}$ is at most $4 \cdot \#\mathcal{F}_1$. Finally, the entire Boolean Macaulay matrix of $\mathcal{F}_1$ is simply given by stacking the Boolean Maculay matrices of $\{f\}$ for $f \in \mathcal{F}_1$, so the total column sparsity is at most $4m \cdot \#\mathcal{F}_1$. $\qquad\square$

Note that this also implies that M, a submatrix of $\hat{\mathcal{B}}$, is also sparse. Moreover, the location and value of the nonzero entries of each column/row of $\hat{\mathcal{B}}$ can be efficiently computed.

Now we show that the Boolean Macaulay linear system is equivalent to the Macaulay linear system. It follows that solving the Boolean Macaulay linear system returns a correct solution of the Boolean polynomial system.[10]

Lemma 5.5. *Let $M_1\vec{y}_1 = \vec{b}_1$ be the Macaulay linear system of a polynomial system $\mathcal{F} = \mathcal{F}_1 \cup \mathcal{F}_2$ and let $M_2\vec{y}_2 = \vec{b}_2$ be the corresponding Boolean Macaulay linear system, where the Macaulay matrix is $\hat{\mathcal{M}} = [M_1 \quad -\vec{b}_1]$, the Boolean Macaulay matrix is $\hat{\mathcal{B}} = [M_2 \quad -\vec{b}_2]$. Then a solution $\hat{y}_2$ of the Boolean Macaulay linear system $M_2\vec{y}_2 = \vec{b}_2$ corresponds to a solution $\hat{y}_1$ of the Macaulay linear system $M_1\vec{y}_1 = \vec{b}_1$.*

Proof. By Lemma 5.3, $\hat{\mathcal{M}} = [M_1 \quad -\vec{b}_1]$ can be reduced to $\begin{bmatrix} 0 & \hat{\mathcal{B}} \\ I_2 & B_2 \end{bmatrix}$ by row operations, where $\hat{\mathcal{B}} = [M_2 \quad -\vec{b}_2]$ is the Boolean Macaulay matrix. Also, $B_2 = [B_2' \quad 0]$ because the last column of the reduced Maculay matrix $\begin{bmatrix} 0 & \hat{\mathcal{B}} \\ I_2 & B_2 \end{bmatrix}$ is indexed by the degree 0 monomial 1 and the polynomials generated from $\mathcal{F}_2$ have no constant terms. Therefore $\begin{bmatrix} 0 & \hat{\mathcal{B}} \\ I_2 & B_2 \end{bmatrix} = \begin{bmatrix} 0 & M_2 & -\vec{b}_2 \\ I_2 & B_2' & 0 \end{bmatrix}$.

---

[10]This observation also implies that the complete solving degree in Chen and Gao's original approach is always at most $n + 2$, tightening their upper bound $3n$.

Since performing row operations on the augmented matrix of a linear system does not change the set of solutions, solving the Macaulay linear system $M_1\vec{y}_1 = \vec{b}_1$ is equivalent to solving the linear system $\begin{bmatrix} 0 & M_2 \\ I_2 & B_2' \end{bmatrix} \begin{bmatrix} \vec{z}_1 \\ \vec{y}_2 \end{bmatrix} = \begin{bmatrix} \vec{b}_2 \\ 0 \end{bmatrix}$, where the entries of $\vec{y}_2$ and $\vec{z}_1$ are indexed by nontrivial multilinear monomials and nonmultilinear monomials respectively.

For the linear system

$$\begin{bmatrix} 0 & M_2 \\ I_2 & B_2' \end{bmatrix} \begin{bmatrix} \vec{z}_1 \\ \vec{y}_2 \end{bmatrix} = \begin{bmatrix} \vec{b}_2 \\ 0 \end{bmatrix}$$

we have $M_2\vec{y}_2 = \vec{b}_2$, which is the Boolean Macaulay linear system, and $\vec{z}_1 + B_2'\vec{y}_2 = 0$.

If $\hat{y}_2$ is a solution of the Boolean Macaulay linear system $M_2\vec{y}_2 = \vec{b}_2$, set $\hat{z}_1$ to be $-B_2'\hat{y}_2$, then $\begin{bmatrix} \hat{z}_1 \\ \hat{y}_2 \end{bmatrix}$ is a solution of the linear system $\begin{bmatrix} 0 & M_2 \\ I_2 & B_2' \end{bmatrix} \begin{bmatrix} \vec{z}_1 \\ \vec{y}_2 \end{bmatrix} = \begin{bmatrix} \vec{b}_2 \\ 0 \end{bmatrix}$. Because the Macaulay linear system is equivalent to the linear system $\begin{bmatrix} 0 & M_2 \\ I_2 & B_2' \end{bmatrix} \begin{bmatrix} \vec{z}_1 \\ \vec{y}_2 \end{bmatrix} = \begin{bmatrix} \vec{b}_2 \\ 0 \end{bmatrix}$, therefore, a solution $\hat{y}_2$ of the Boolean Macaulay linear system $M_2\vec{y}_2 = \vec{b}_2$ corresponds to a solution $\hat{y}_1$ of the Macaulay linear system $M_1\vec{y}_1 = \vec{b}_1$. $\qquad\square$

As M is a $\mathcal{O}(m \cdot \#\mathcal{F})$-sparse row / column computable matrix and we can efficiently prepare the sparse vector $\vec{b}$ as quantum state $|b\rangle$, we can apply a QLS algorithm to "solve" the Boolean Macaulay linear system $M\vec{y} = \vec{b}$, which takes time $\widetilde{\mathcal{O}}(\mathrm{poly}(n)\kappa(M)\log(1/\epsilon))$ [CKS17].

The key parameter in the running time is the condition number of the matrix M. Next we will provide a lower bound of the tQLScn of M and thus also a lower bound on known QLS algorithms.

5.2   Lower bound on the tQLScn $\kappa_{\vec{b}}(M)$

Suppose $a_1, a_2, \cdots, a_t \in \{0,1\}^n$ are the t solutions of $\mathcal{F} = \mathcal{F}_1 \cup \mathcal{F}_2$, where $\mathcal{F}_1 = \{f_1, \ldots, f_m\}$ and $\mathcal{F}_2 = \{x_1^2 - x_1, \ldots, x_n^2 - x_n\}$, and let h be the minimum Hamming weight of the t solutions $a_1, a_2, \cdots, a_t$. Let $\vec{y}_1, \vec{y}_2, \cdots, \vec{y}_t$ be the corresponding solution vectors of the Boolean Macaulay linear system $M\vec{y} = \vec{b}$ under the assignments $a_1, a_2, \cdots, a_t$ respectively.

In this case, we have $\|M\| \geq 1/2$ as M has at least one matrix element which has absolute value at least $1/2$ [11]. Analogously to Theorem 4.5 we get:

Corollary 5.6. *Let $\hat{\mathcal{B}} = [M \quad -\vec{b}]$ be the Boolean Macaulay matrix of $\mathcal{F}$ with columns labeled by multilinear monomials. If all the t solutions $a_1, a_2, \cdots, a_t$ have the same Hamming weight h or the minimum ℓ_2-norm solution vector $\vec{y} = M^+\vec{b}$ is in the convex hull of $\vec{y}_1, \vec{y}_2, \cdots, \vec{y}_t$, then the tQLScn $\kappa_{\vec{b}}(M) \geq \frac{1}{2}\sqrt{(2^h - 1)/t}$ of M of $\mathcal{F}$.*

For $h = \Theta(\log n)$, this lower bound does not rule out the possibility that the Macaulay matrix has a polynomial condition number, which would result in the quantum algorithm beating the brute-force classical algorithm that runs in time $\widetilde{\mathcal{O}}\left(\binom{n}{\log n}\right)$.

---

[11] After applying Red2 there is at least one polynomial f with a constant term of magnitude 1. If f does not have a degree-1 monomial x_i, then $x_i \cdot f$ has a magnitude 1 degree-1 monomial x_i, so the row (x_i, f) will have a matrix element of magnitude 1. Otherwise, suppose the coefficient of x_1 in f is c_1, then the rows $(1, f)$ and (x_1, f) will have a matrix element of magnitude c_1 and $c_1 - 1$ respectively. Therefore, at least one of them has magnitude at least $1/2$.

5.3 Details comparing running times

As we have discussed in Section 4.3 the classical brute-force algorithm tries all $\binom{n}{j}$ choices for the locations of the 1's in the solution a for each $j \leq h$, and its running time can be bounded $\mathcal{O}\left(\sqrt{h}\binom{n}{h}\right)$, where

$$\forall 1 \leq h \leq n : \left(\frac{n}{h}\right)^h \leq \binom{n}{h} \leq \left(\frac{en}{h}\right)^h$$

Comparing the above expression with our tQLScn lower bound, we saw that the Gorver-enhanced brute-force search always outperforms the Maculay matrix approach in case there is a unique solution and $d = n$ (or even $d + h \geq n$). This in particular shows that the quantum algorithm achieves at most a quadratic speed-up compared to classical brute-force search. Moreover, if one chooses $d = 3n$ and works with the max degree as Chen and Gao suggested [CG17b], then $\kappa_{\vec{b}}(\mathcal{M}) \geq (3n)^{h/2}$ and so

- For $h = \Omega(\sqrt{n})$, the classical brute force algorithm is faster than the quantum algorithm.

- For $h = \mathcal{O}(\sqrt{n})$, it is unknown which is faster.

On the other hand in the Boolean case we have only the lower bound $\kappa_{\vec{b}}(M) \geq \frac{1}{2}\sqrt{(2^h - 1)}$, so:

- For $h = pn$, where $p \in (0, \frac{1}{2}]$, the lower bound of $\kappa_{\vec{b}}(M) \geq \frac{1}{2}(2^h - 1)^{1/2}$ is exponentially large and exhaustive search takes time $\mathcal{O}\left(2^{H(p)n}\right)$ where $H(p) = -p\log p - (1-p)\log(1-p)$ is the binary entropy function, as shown in Appendix B.

- For $h = \mathcal{O}(1)$, $\exists$ classical algorithm that takes time $\mathcal{O}\left(\binom{n}{h}\right)$ to solve the problem efficiently by exhaustive search whereas the lower bound of $\kappa(M)$ is a constant $(2^{\mathcal{O}(1)} - 1)^{1/2}$.

- For $h = \Theta(\log n)$, we only know that $\kappa_{\vec{b}}(M) \geq \text{poly}(n)$ whereas classical exhaustive search takes time $\mathcal{O}\left(\binom{n}{\log n}\right)$. Thus, we cannot exclude the possibility that the quantum algorithm might give a quasi-polynomial speedup in this case.

Without loss of generality, let $0 \leq h \leq \frac{n}{2}$ (otherwise we can flip all variables). Then the lower bound on the tQLScn $\kappa(M)$ is always smaller than the time required by brute-force search. Thus, there is a possibility that the quantum algorithm performs better than the exhaustive search approach.

6 Our new improved quantum algorithm

6.1 A Variant of the Quantum Coupon Collector Problem

By [CG17b, Corollary 3.19] and Lemma 5.5, if a set of polynomials $\mathcal{F}$ over $\mathbb{C}[x_1, \ldots, x_n]$ has a unique solution $a = (a_1, a_2, \cdots, a_n) \in \{0, 1\}^n$, then for some d less than or equal to n, the corresponding Boolean Macaulay linear system $M\vec{y} = \vec{b}$ of total degree d has a unique solution $\vec{y} = M^+\vec{b}$, where the entries of $\vec{y}$ are indexed by multilinear monomials in $x_1, \ldots, x_n$ with total degree at most d. Let $U = \{x_1, x_2, \ldots, x_n\}$. There is a one-to-one correspondence between the subsets of U of size at most d and multilinear monomials in $x_1, \ldots, x_n$ with total degree at most d. Let S be the subset of U such that all the variables $x_k \in S$ have assignment $a_k = 1$ and S_d be the set containing all nonempty subsets of S that have size at most d. There is a one-to-one correspondence between the elements of the set S_d and the nonzero entries of $\vec{y}$. Given implicit access to matrix M and sparse vector b, the QLS algorithm outputs the solution vector $\vec{y}$ as the quantum state $|\vec{y}\rangle$, which encodes the nonzero

entries of $\vec{y}$. Because the unique solution $\vec{y} = M^+\vec{b}$ of the Boolean Macaulay linear system is a $0/1$ vector, the quantum state $|\vec{y}\rangle$ can be represented by

$$|\vec{y}\rangle = \frac{1}{\sqrt{|S_d|}} \sum_{R \in S_d} |R\rangle.$$

If we measure the quantum state $|\vec{y}\rangle$, we will get a uniformly random subset $R \in S_d$, where all the variables $x_k \in R$ have assignment $a_k = 1$. Given copies of the quantum state $|\vec{y}\rangle$, the goal is to compute S.

Next, we will reformulate this problem as a variant of the quantum coupon collector problem.

Problem 6.1. *Let $S \subseteq U = \{x_1, x_2, \ldots, x_n\}$ be an unknown subset and S_d be the set containing all nonempty subsets of S that have size at most d. Given copies of the state*

$$|\vec{y}\rangle = \frac{1}{\sqrt{|S_d|}} \sum_{R \in S_d} |R\rangle,$$

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

Specially, when d equals 1, this is the quantum coupon collector problem defined in [ABC+20]. They proved that $\Theta(|S|\log(\min\{|S|, n - |S|\}))$ copies of the states $|\vec{y}\rangle$ are necessary to compute S.

Without loss of generality, we can assume d is at most $|S|$ because when d is greater than $|S|$, the quantum state $|\vec{y}\rangle$ is the same as the case of d equals $|S|$. Then, we have the following result of Problem 6.1.

Theorem 6.2. *Measuring $r = \mathcal{O}((|S|/d)\log(|S|/\varepsilon))$ copies of the quantum superposition state in Problem 6.1, the set S can be computed with probability at least $1 - \varepsilon$.*

Since the only quantum operation is measurement in the computational basis, this is essentially a classical coupon collector problem, where we can sample a uniformly random subset.

Proof. For any $x \in S$, the number of sets $R \in S_d$ containing x is $\sum_{i=1}^{d} \binom{|S|-1}{i-1}$ out of a total number of sets $\sum_{i=1}^{d} \binom{|S|}{i}$ in S_d. Thus, the probability of seeing x equals $\sum_{i=1}^{d} \binom{|S|-1}{i-1} / \sum_{i=1}^{d} \binom{|S|}{i}$. If $0 \le d \le \lfloor \frac{|S|}{3} \rfloor$, we have $\binom{|S|}{d} \le \sum_{i=1}^{d} \binom{|S|}{i} \le \binom{|S|}{d} \frac{|S|-d+1}{|S|-2d+1}$, so $\sum_{i=1}^{d} \binom{|S|-1}{i-1} / \sum_{i=1}^{d} \binom{|S|}{i} \ge \frac{d}{|S|} \cdot \frac{|S|-2d+1}{|S|-d+1}$, where $\frac{|S|-2d+1}{|S|-d+1} < 1$. Hence, when $0 \le d \le \lfloor \frac{|S|}{3} \rfloor$, the probability of not seeing x after r tries is at most $(1 - \frac{d}{|S|} \cdot \frac{1}{2})^r = (1 - \frac{d}{2|S|})^{-\frac{2|S|}{d} \frac{d}{2|S|} r} \le \exp(-\frac{d}{2|S|} r)$. If $\lceil \frac{|S|}{3} \rceil \le d \le |S|$, the probability function $\sum_{i=1}^{d} \binom{|S|-1}{i-1} / \sum_{i=1}^{d} \binom{|S|}{i}$ is an increasing function because $\frac{\binom{|S|-1}{0}}{\binom{|S|}{1}} < \frac{\binom{|S|-1}{1}}{\binom{|S|}{2}} < \cdots < \frac{\binom{|S|-1}{d-1}}{\binom{|S|}{d}}$, and it has the minimum value when $d = \lceil \frac{|S|}{3} \rceil$, which is greater than $\frac{d}{|S|} \cdot \frac{|S|-2d+1}{|S|-d+1} \ge \frac{1}{6}$. Hence, the probability of not finding x after r tries is at most $(1 - \frac{1}{6})^r$.

Let $r = \mathcal{O}((|S|/d)\log(|S|/\varepsilon))$, and by the union bound, the probability of not collecting all the elements x in S is at most ε. That is, if $r = \mathcal{O}((|S|/d)\log(|S|/\varepsilon))$, the entire set S can be recovered with probability at least $1 - \varepsilon$. $\qquad\square$

With respect to the choice of d, there is a trade-off between the number of samples and the memory space:

- For $d = O(1)$, $r = O(|S|\log|S|)$

- For $d = O(\log |S|)$, $r = O(|S|)$

- For $d = O(\frac{|S|}{\log |S|})$, $r = O(\log^2 |S|)$

- For $\frac{|S|}{k} \leq d \leq n$, where k is a positive integer, $r = O(\log |S|)$

6.2   The algorithm

When a set of polynomials $\mathcal{F}$ has a unique solution, Algorithm 1 finds the solution. If a set of polynomials has more than one solution, we apply the Valiant-Vazirani reduction Red1 to get a set of polynomials $\mathcal{F}$ that have a unique solution.

Algorithm 1 Quantum linear system algorithm for $\mathcal{F}$ over $\mathbb{C}$

Input: $\mathcal{F} = \mathcal{F}_1 \cup \mathcal{F}_2 \subseteq \mathbb{C}[x_1, \ldots, x_n]$ where $\mathcal{F}_1 = \{f_1, \ldots, f_m\}$ with $\deg(f_i) = 2$ for $i = 1, \ldots, m$, and $\mathcal{F}_2 = \{x_1^2 - x_1, \ldots, x_n^2 - x_n\}$.
Output: Find the solution $a \in \{0, 1\}^n$ such that $f_1(a) = \cdots = f_m(a) = 0$ over $\mathbb{C}$ when one exists.
  Step 1: Apply a quantum linear system algorithm to the Boolean Macaulay linear system $M\vec{y} = \vec{b}$ of total degree n and get the solution $\vec{y}$ in quantum state

$$|\vec{y}\rangle = \frac{1}{\sqrt{|S_d|}} \sum_{R \in S_d} |R\rangle$$

  Step 2: Perform measurement on the quantum state $|\vec{y}\rangle$ and get outcome $|R\rangle$, then let all the variables in the set R equal 1.
  Step 3: Repeat Step 1 and Step 2 $O(\log n)$ times, and then set all left remaining variables $a_j = 0$.

  Step 4: Return a.

Lemma 6.3. *With high probability Algorithm 1 solves Problem 3.2 in time $\widetilde{\mathcal{O}}(\text{poly}(n)\kappa(M))$.*

Proof. For the Boolean Macaulay linear system $M\vec{y} = \vec{b}$, the matrix M is $\mathcal{O}(m \cdot \#\mathcal{F})$-sparse and the vector $\vec{b}$ can be prepared as $|\vec{b}\rangle = |0\rangle^{n\lceil \log m \rceil}$. Therefore, we can apply a QLS algorithm [CKS17] to the Boolean Macaulay linear system, which takes time $\widetilde{\mathcal{O}}(\text{poly}(n)\kappa(M)\log(1/\epsilon))$. The QLS algorithm outputs a quantum state $|\vec{y}^*\rangle$, which is an approximation of $|\vec{y}\rangle$ with $\||\vec{y}\rangle - |\vec{y}^*\rangle\| \leq \varepsilon$. If we repeat the process r-times, then we essentially prepare the state $|(\vec{y}^*)^{\otimes r}\rangle$ for which we have $\langle (\vec{y})^{\otimes r} | (\vec{y}^*)^{\otimes r} \rangle = (\langle \vec{y} | \vec{y}^* \rangle)^r = (1 - \Theta(\varepsilon^2))^r$. For $\varepsilon = \mathcal{O}(1/r)$ we have that this equals $(1 - \Theta(r\varepsilon^2))$ and so $\||(\vec{y})^{\otimes r}\rangle - |(\vec{y}^*)^{\otimes r}\rangle\| = \Theta(\sqrt{r}\varepsilon)$. Then the total variation distance between the two probability distributions of any measurements on the two states $|(\vec{y})^{\otimes r}\rangle$ and $|(\vec{y}^*)^{\otimes r}\rangle$ is at most $\Theta(\sqrt{r}\varepsilon)$ [dW19, Exercise 4.3] (see also [BV97, Lemma 3.6]), so replacing the ideal state $|(\vec{y})^{\otimes r}\rangle$ by the approximate state $|(\vec{y}^*)^{\otimes r}\rangle$ induces error probability at most $\mathcal{O}(\sqrt{r}\varepsilon)$. By Lemma 6.2, we can extract the solution of the polynomial system $\mathcal{F}$ from $|(\vec{y})^{\otimes r}\rangle$ with high probability by choosing r to be $O(\log n)$. Therefore, letting $\epsilon = 1/\Theta(\sqrt{\log n})$, with high probability Algorithm 1 solves Problem 3.2 in time $\widetilde{\mathcal{O}}(\text{poly}(n)\kappa(M))$. $\qquad\square$

    Compared with Chen and Gao's [CG17b] algorithm, there are two differences with Algorithm 1. First, the size of the Boolean Macaulay matrix in Algorithm 1 is $m2^n \times 2^n$, which leads to a smaller lower bound of the tQLScn and leaves a possibility of superpolynomial speedup using Algorithm 1. By contrast, the size of the Macaulay matrix in Chen and Gao's algorithm is $(m + n)(3n + 1)^n \times$

$(3n + 1)^n$, which leads to a larger lower bound of the tQLScn that prohibits a potential quantum speedup. Second, in Algorithm 1, the polynomial system have a unique solution, so in contrast to [CG17b] the Boolean Macaulay linear system stays the same for every iteration and the number of iterations / measurements required to obtain the solution of the polynomial system is $\mathcal{O}(\log n)$. However, the Valiant-Vazirani reduction needs $\mathcal{O}(n)$ iterations to generate a polynomial system that has a unique solution with high probability. This amounts to $\mathcal{O}(n \log n)$ iterations in total to find a solution. On the other hand, in Chen and Gao's algorithm, the polynomial system could have any finite number of solutions, so the Macaulay linear system needs to be updated after each iteration (measurement) and the number of iterations is $\mathcal{O}(n)$.

7   Discussion

The Boolean Macaulay linear system approach is an interesting framework to study giving insights to the limitations and capabilities of quantum computation. On one hand, a lot of problems such as Factoring, Graph isomorphism, and Learning with binary errors can be put into this single framework. On the other hand, the Factoring problem is known to be inside BQP by Shor's algorithm, therefore, if we can find an approach to get around the curse of the condition number of the Boolean Macaulay linear system derived from the Factoring problem, then we might be able to extend the result to other problems, such as Graph Isomorphism and Learning with binary errors, revealing new capabilities of quantum computation.

Our analytical lower bound on the condition number decreases when there are multiple solutions of the polynomial systems, but the polynomial systems used for cryptography purpose usually have one or few solutions [CG17a], so our result gives strong evidence that the QLS algorithm cannot be used for attacking cryptosystems via the Macaulay matrix approach. Also, we suspect that having many solutions will not make the QLS algorithm to work substantially better. For example consider adding l new field equations $y_i^2 - y_i = 0$, then the number of solution of the new polynomial system will increase by a factor of 2^l, however the length of the shortest vector stays the same – indeed one can see that the shortest vector is an affine combination of solutions where all the new variables are set to 0.[12]

Given an ill-conditioned QLSP, two main approaches have been proposed, one is the truncated QLS algorithm, the other one is the preconditioned QLS algorithm [HHL09]. Our lower bound on the tQLScn prohibits further speedup by the truncated QLS algorithm, however further investigation is needed regarding the possibility of using preconditioned QLS algorithms, such as parallel sparse approximate inverse preconditioner [CJS13], circulant preconditioner [SX18], fast inversion [TAWL20], or develop new preconditioned QLS algorithms for the Boolean Macaulay linear system. A promising feature of the Boolean Macaulay linear system is that it cannot be de-quantized by known classical techniques [CGL+20] since the Boolean Macaulay linear system has high (full) column rank.

The introduced variant of the quantum coupon collector problem provides an example of how to extract the solution efficiently if the values in the solution vector of a QLSP are correlated in a nice pattern. Since applications of the QLS algorithm usually gain restricted access to the solution vector, a generalization of our extraction method, utilizing more general correlated patterns, could have interesting applications.

When the Hamming weight of the solution of a Boolean polynomial system is logarithmic in the number of variables, our lower bound does not rule out a superpolynomial speedup over exhaustive search. Finding a real application that exhibits such a superpolynomial speedup would be very inter-

---

[12]To see this consider the coordinates corresponding to monomials not including the new variables.

esting. However, for applications like polynomial systems over a finite field, the employed reductions usually increase the number of variables in the corresponding polynomial system significantly – likely compromising the potential speedup.

Acknowledgements

References

[Aar15]   Scott Aaronson. Read the fine print. *Nature Physics*, 11(4):291–293, 2015.

[ABC+20]   Srinivasan Arunachalam, Aleksandrs Belovs, Andrew M. Childs, Robin Kothari, Ansis Rosmanis, and Ronald de Wolf. Quantum coupon collector. In *Proceedings of the 15th Conference on the Theory of Quantum Computation, Communication, and Cryptography (TQC)*, pages 10:1–10:17, 2020. arXiv: `2002.07688`

[AFI+04]   Gwénolé Ars, Jean-Charles Faugere, Hideki Imai, Mitsuru Kawazoe, and Makoto Sugita. Comparison between XL and Gröbner basis algorithms. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 338–353. Springer, 2004.

[Amb12]   Andris Ambainis. Variable time amplitude amplification and quantum algorithms for linear algebra problems. In *Proceedings of the 29th Symposium on Theoretical Aspects of Computer Science (STACS)*, pages 636–647, 2012. arXiv: `1010.4458`

[AOAGC18]   Eric R Anschuetz, Jonathan P Olson, Alán Aspuru-Guzik, and Yudong Cao. Variational quantum factoring. arXiv: `1808.08927`, 2018.

[B+18]   Bruno Buchberger et al. Gröbner bases computation by triangularizing Macaulay matrices. In *The 50th Anniversary of Gröbner Bases*, pages 25–33. Mathematical Society of Japan, 2018.

[Bat13]   Kim Batselier. *A numerical linear algebra framework for solving problems with multivariate polynomials*. PhD thesis, KU Leuven (Leuven, Belgium), 2013.

[BBHT98]   Michel Boyer, Gilles Brassard, Peter Høyer, and Alain Tapp. Tight bounds on quantum searching. *Fortschritte der Physik*, 46(4–5):493–505, 1998. arXiv: `quant-ph/9605034`

[BFSS13]   Magali Bardet, Jean-Charles Faugère, Bruno Salvy, and Pierre-Jean Spaenlehauer. On the complexity of solving quadratic boolean systems. *Journal of Complexity*, 29(1):53–75, 2013.

[BKW19]   Andreas Björklund, Petteri Kaski, and Ryan Williams. Solving systems of polynomial equations over GF(2) by a parity-counting self-reduction. In *Proceedings of the 46th International Colloquium on Automata, Languages, and Programming (ICALP)*, 2019.

[Bur02]   Christopher JC Burges. Factoring as optimization. *Microsoft Research MSR-TR-200*, 2002.

[BV97]   Ethan Bernstein and Umesh Vazirani. Quantum complexity theory. *SIAM Journal on computing*, 26(5):1411–1473, 1997.

[BY18]   Daniel J Bernstein and Bo-Yin Yang. Asymptotically faster quantum algorithms to solve multivariate quadratic equations. In *International Conference on Post-Quantum Cryptography*, pages 487–506. Springer, 2018.

[CG17a]   Alessio Caminata and Elisa Gorla. Solving multivariate polynomial systems and an invariant from commutative algebra. arXiv: `1706.06319`, 2017.

[CG17b]   Yu-Ao Chen and Xiao-Shan Gao. Quantum algorithms for Boolean equation solving and quantum algebraic attack on cryptosystems, 2017. arXiv: `1712.06239`

[CGJ19]   Shantanav Chakraborty, András Gilyén, and Stacey Jeffery. The power of block-encoded matrix powers: improved regression techniques via faster Hamiltonian simulation. In *Proceedings of the 46th International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 33:1–33:14, 2019. arXiv: `1804.01973`

[CGL+20]   Nai-Hui Chia, András Gilyén, Tongyang Li, Han-Hsuan Lin, Ewin Tang, and Chunhao Wang. Sampling-based sublinear low-rank matrix arithmetic framework for dequantizing quantum machine learning. In *Proceedings of the 52nd ACM Symposium on the Theory of Computing (STOC)*, page 387–400, 2020. arXiv: `1910.06151`

[CGY18]   Yu-Ao Chen, Xiao-Shan Gao, and Chun-Ming Yuan. Quantum algorithm for optimization and polynomial system solving over finite field and application to cryptanalysis, 2018. arXiv: `1802.03856`

[CJS13]   B David Clader, Bryan C Jacobs, and Chad R Sprouse. Preconditioned quantum linear system algorithm. *Physical review letters*, 110(25):250504, 2013.

[CKPS04]   Nicolas Courtois, Alexander Klimov, Jacques Patarin, and Adi Shamir. Efficient algorithms for solving overdefined systems of multivariate polynomial equations. In *International Conference on the Theory and Applications of Cryptographic Techniques*, pages 392–407. Springer, 2000 (Extended version as of 24 Aug, 2004. `http://www.minrank.org/xlfull.pdf`).

[CKS17]   Andrew M. Childs, Robin Kothari, and Rolando D. Somma. Quantum algorithm for systems of linear equations with exponentially improved dependence on precision. *SIAM Journal on Computing*, 46(6):1920–1950, 2017. arXiv: `1511.02306`

[Die04]   Claus Diem. The XL-algorithm and a conjecture from commutative algebra. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 323–337. Springer, 2004.

[DS13]   Jintai Ding and Dieter Schmidt. Solving degree and degree of regularity for polynomial systems over a finite fields. In *Number Theory and Cryptography*, pages 34–49. Springer, 2013.

[FHK+17]   Jean-Charles Faugere, Kelsey Horan, Delaram Kahrobaei, Marc Kaplan, Elham Kashefi, and Ludovic Perret. Fast quantum algorithm for solving multivariate quadratic equations. arXiv: `1712.07211`, 2017.

[GSLW18]   András Gilyén, Yuan Su, Guang Hao Low, and Nathan Wiebe. Quantum singular value transformation and beyond: exponential improvements for quantum matrix arithmetics [full version], 2018. arXiv: `1806.01838`

[GSLW19]   András Gilyén, Yuan Su, Guang Hao Low, and Nathan Wiebe. Quantum singular value transformation and beyond: exponential improvements for quantum matrix arithmetics. In *Proceedings of the 51st ACM Symposium on the Theory of Computing (STOC)*, pages 193–204, 2019. arXiv: `1806.01838`

[HHL09]   Aram W. Harrow, Avinatan Hassidim, and Seth Lloyd. Quantum algorithm for linear systems of equations. *Physical Review Letters*, 103(15):150502, 2009. arXiv: `0811.3171`

[Juk11]   Stasys Jukna. *Extremal Combinatorics - With Applications in Computer Science (2nd ed.)*. Texts in Theoretical Computer Science. Springer, 2011.

[KP17]   Iordanis Kerenidis and Anupam Prakash. Quantum recommendation systems. In *Proceedings of the 8th Innovations in Theoretical Computer Science Conference (ITCS)*, pages 49:1–49:21, 2017. arXiv: `1603.08675`

[LT20]   Lin Lin and Yu Tong. Optimal polynomial based quantum eigenstate filtering with application to solving quantum linear systems. *Quantum*, 4:361, 2020. arXiv: `1910.14596`

[Per16]   Ludovic Perret. *Bases de Gröbner en Cryptographie Post-Quantique*. PhD thesis, UPMC-Paris 6 Sorbonne Universités, 2016.

[Rob55]   Herbert Robbins. A remark on Stirling's formula. *The American Mathematical Monthly*, 62(1):26–29, 1955.

[SSO19]   Yiğit Subaşı, Rolando D. Somma, and Davide Orsucci. Quantum algorithms for systems of linear equations inspired by adiabatic quantum computing. *Physical Review Letters*, 122(6):060504, 2019. arXiv: `1805.10549`

[SX18]   Changpeng Shao and Hua Xiang. Quantum circulant preconditioner for a linear system of equations. *Physical Review A*, 98(6):062321, 2018.

[TAWL20]   Yu Tong, Dong An, Nathan Wiebe, and Lin Lin. Fast inversion, preconditioned quan-
           tum linear system solvers, and fast evaluation of matrix functions. arXiv: `2008.13295`,
           2020.

[VV86]   Leslie G. Valiant and Vijay V. Vazirani. NP is as easy as detecting unique solutions.
           Theoretical Computer Science, 47:85–93, 1986. Earlier version in STOC'85.

[dW19]   Ronald de Wolf. Quantum computing: Lecture notes, 2019. arXiv: `1907.09415`

[WW15]   Manuela Wiesinger-Widi. *Gröbner bases and generalized sylvester matrices*. PhD thesis,
           Johannes Kepler University Linz, Austria, 2015.

A   Simple proof of the unique solution case

Here we present a simple proof for the correctness of Algorithm 1 for Problem 3.2 when it has a unique solution. Let $a = (a_1, a_2, \ldots a_n) \in \{0, 1\}^n$ be the unique solution of a set of polynomials $\mathcal{F}$. Let $\hat{y} = [a_1, a_2, \ldots, a_i a_j, \ldots, \prod_{i=1}^n a_i]^\top$ be the 0/1 solution vector labeled by the multilinear monomials under the assignment a.

Next, we will show that the Boolean Macaulay linear system $M\vec{y} = \vec{b}$ has the unique solution $\hat{y}$ when $\mathcal{F}$ has the unique solution a. In this case, we have $\hat{y} = M^+\vec{b}$ because the matrix M has linearly independent columns. When $\mathcal{F}$ has more than one solution, the columns of the matrix M are not linearly independent and the solutions of $M\vec{y} = \vec{b}$ form a multidimensional affine subspace.

Lemma A.1. *[AFI+04, Theorem 2] If a set of polynomials $\mathcal{F} \subseteq \mathbb{C}[x_1, \ldots, x_n]$ has a unique solution $a = (a_1, a_2, \ldots, a_n)$, then the following two polynomial ideals coincide*

$$\langle \mathcal{F} \rangle = \langle x_1 - a_1, x_2 - a_2, \ldots, x_n - a_n \rangle.$$

Theorem A.2. *Given a set of polynomials $\mathcal{F} = \mathcal{F}_1 \cup \mathcal{F}_2 \subseteq \mathbb{C}[x_1, \ldots, x_n]$, where $\mathcal{F}_1 = \{f_1, f_2, \ldots, f_m\}$ and $\mathcal{F}_2 = \{x_1^2 - x_1, x_2^2 - x_2, \ldots, x_n^2 - x_n\}$. Suppose $\mathcal{F}$ has a unique solution $a = (a_1, \ldots, a_n)$, where $\mathcal{F}_2$ forces the root of the set of polynomials $\mathcal{F}_1$ to be Boolean. Let $\hat{y}$ be the multilinear monomial solution vector corresponding to the solution a, then the Boolean Macaulay linear system $M\vec{y} = \vec{b}$ of total degree n has the unique solution $\hat{y} = M^+\vec{b}$.*

Proof. First, we prove that for all the nontrivial multilinear monomials X^β, the polynomial $X^\beta - \prod_{i=1}^n a_i^{\beta_i} = \prod_{i=1}^n x_i^{\beta_i} - \prod_{i=1}^n a_i^{\beta_i}$ is in $\langle \mathcal{F} \rangle$, where $\beta \in \{0, 1\}^n \setminus \{0^n\}$. The proof is by induction on the degree d. For the base case $d = 1$, by Lemma A.1, for all $1 \leq k \leq n$, $x_k - a_k \in \langle \mathcal{F} \rangle$. That is, for each k, there exist p_i, q_j such that $x_k - a_k = \sum_{i=1}^m p_i f_i + \sum_{j=1}^n q_j(x_j^2 - x_j)$. Let $B_d = \{$all multilinear monomials with degree d$\}$, and suppose the claim is true for d, that is, for any $X_d^\beta \in B_d$, $X_d^\beta - \prod_{i=1}^n a_i^{\beta_i} \in \langle \mathcal{F} \rangle$. For any $X_{d+1}^\beta \in B_{d+1}$, there exists some x_k and X_d^β such that $X_{d+1}^\beta = x_k \cdot X_d^\beta$, then

$$X_{d+1}^\beta - \prod_{i=1}^n a_i^{\beta_i} = x_k(X_d^\beta - \prod_{i=1}^n a_i^{\beta_i}) + (x_k - a_k)\prod_{i=1}^n a_i^{\beta_i},$$

which implies that $X_{d+1}^\beta - \prod_{i=1}^n a_i^{\beta_i} \in \langle \mathcal{F} \rangle$. Therefore, for all nontrivial multilinear monomials X^β, there exists $p_{i\beta}, q_{j\beta} \in \mathbb{C}[x_1, \ldots, x_n]$ such that $X^\beta - \prod_{i=1}^n a_i^{\beta_i} = \sum_{i=1}^m p_{i\beta} f_i + \sum_{j=1}^n q_{j\beta}(x_j^2 - x_j) \in \langle \mathcal{F} \rangle$, where $i \in [m], j \in [n]$.

The Boolean Macaulay matrix $\begin{bmatrix} M & -\vec{b} \end{bmatrix}$ is the augmented matrix of the Boolean Macaulay linear system $M\vec{y} = \vec{b}$ of the set of polynomials $\mathcal{F}$. Since $X^\beta - \prod_{i=1}^n a_i^{\beta_i} = \psi\left(X^\beta - \prod_{i=1}^n a_i^{\beta_i}\right) = \sum_{i=1}^m \psi(p_{i\beta} f_i)$, and polynomial addition, subtraction, and multiplication in the polynomial ideal $\langle \mathcal{F} \rangle$ correspond to row operations of the Boolean Macaulay matrix $\begin{bmatrix} M & -\vec{b} \end{bmatrix}$, we can perform row operations on the Boolean Macaulay matrix according to $\sum_{i=1}^m \psi(p_{i\beta} f_i)$. By those row operations we can obtain an extended matrix of form $\begin{bmatrix} M & -\vec{b} \\ I & -\vec{y} \end{bmatrix}$, where the columns of the identity matrix I are indexed by the nontrivial multilinear monomials and entries of $\vec{y}$ are values of $\prod_{i=1}^n a_i^{\beta_i}$. As performing row operations on a matrix does not change the matrix rank, the matrix M must have full column rank. Therefore, the Boolean Macaulay linear system has the unique solution $\hat{y} = M^+ \vec{b}$. $\quad\square$

B Bounds on binomial coefficients

In this appendix we derive some standard bounds on binomial coefficients for completeness. First we show that for $h \leq n/2$

$$\sum_{j=0}^h \binom{n}{j} \leq 3\sqrt{h}\binom{n}{h}. \tag{17}$$

We use the following upper bound [Juk11, Corollary 22.9] on binomial coefficients

$$\forall 0 < h \leq n/2: \ \sum_{j=0}^h \binom{n}{j} \leq 2^{n \cdot H(h/n)}$$

$$= 2^{n\left(-\frac{h}{n}\log_2\left(\frac{h}{n}\right) - \frac{n-h}{n}\log_2\left(\frac{n-h}{n}\right)\right)}$$

$$= \left(\frac{n}{h}\right)^h \left(\frac{n}{n-h}\right)^{n-h},$$

in combination with Stirling's approximation [Rob55] $\sqrt{2\pi}\sqrt{n}\left(\frac{n}{e}\right)^n \leq n! \leq e\sqrt{n}\left(\frac{n}{e}\right)^n$, yielding

$$\binom{n}{h} = \frac{n!}{(n-h)!h!} \geq \frac{\sqrt{2\pi}\sqrt{n}\left(\frac{n}{e}\right)^n}{e\sqrt{n-h}\left(\frac{n-h}{e}\right)^{n-h}e\sqrt{h}\left(\frac{h}{e}\right)^h}$$

$$= \frac{\sqrt{2\pi}}{e^2}\sqrt{\frac{n}{n-h}}\frac{1}{\sqrt{h}}\left(\frac{n}{h}\right)^h\left(\frac{n}{n-h}\right)^{n-h} \geq \frac{1}{3\sqrt{h}}\sum_{j=0}^h \binom{n}{j}.$$

Another bound that we use is

$$\sum_{j=1}^h \binom{n}{i} \leq \binom{n}{h}\frac{n-h+1}{n-2h+1}, \tag{18}$$

which can be shown by the summation of an upper bound by a geometric series, i.e.,

$$\sum_{i=1}^{h} \binom{n}{i} = \binom{n}{h}\left(1 + \binom{n}{h-1}/\binom{n}{h} + \binom{n}{h-2}/\binom{n}{h} + \cdots + \binom{n}{1}/\binom{n}{h}\right)$$

$$\leq \binom{n}{h}\left(1 + h/(n-h+1) + h^2/(n-h+1)^2 + \cdots + h^{h-1}/(n-h+1)^{h-1}\right)$$

$$\leq \binom{n}{h}\frac{1}{1 - h/(n-h+1)}$$

$$= \binom{n}{h}\frac{n-h+1}{n-2h+1}.$$

C   Mathematica code for symbolically verifying our lower bound

```
In[1]:= GramMax[n_, d_]:= Table[Table[d^s Binomial[i, s]Binomial[j, s]/Binomial[n, s],
                          {s, 1, Min[i, j]}] // Total, {i, 1, n}, {j, 1, n}];
    BoundMax[n_]:= 2GramMax[n, 3n] - Table[Min[i, j]^Min[i, j], {i, 1, n}, {j, 1, n}];
    ParallelMap[(Print[#]; #[[2]]) & @ Timing[{#,
        PositiveDefiniteMatrixQ[BoundMax[#]]}] &, Range[184], Method -> "FinestGrained"]

Out[1]= {0.000715, {1,True}}
    {0.000922, {2,True}}
    {0.000413, {3,True}}
    ⋮
    {36634.1, {296, True}}
    {37459.0, {297, True}}
    {37858.5, {298, True}}
    {38813.1, {299, True}}
    {39445.0, {300, True}}
```