On the List and Bounded Distance Decodability of Reed-Solomon Codes

Qi Cheng[*]
School of Computer Science
The University of Oklahoma
Norman, OK 73019, USA
qcheng@cs.ou.edu

Daqing Wan[†]
University of California
Irvine, CA 92697, USA
and the Institute of Mathematics
Chinese Academy of Sciences
Beijing, P.R. China
dwan@math.uci.edu

Abstract

For an error-correcting code and a distance bound, the *list decoding problem* is to compute all the codewords within a given distance to a received message. The *bounded distance decoding* problem is to find one codeword if there is at least one codeword within the given distance, or to output the empty set if there is not. Obviously the bounded distance decoding problem is not as hard as the list decoding problem. For a Reed-Solomon code $[n,k]_q$, a simple counting argument shows that for any integer $0 < g < n$, there exists at least one Hamming ball of radius $n-g$, which contains at least $\binom{n}{g}/q^{g-k}$ many codewords. Let $\hat{g}(n,k,q)$ be the smallest positive integer g such that $\binom{n}{g}/q^{g-k} \le 1$. One knows that

$$k-1 \le \hat{g}(n,k,q) \le \sqrt{n(k-1)} \le n.$$

For the distance bound up to $n-\sqrt{n(k-1)}$, it is known that both the list and bounded distance decoding can be solved efficiently. For the distance bound between $n-\sqrt{n(k-1)}$ and $n-\hat{g}(n,k,q)$, we do not know whether the Reed-Solomon code is list, or bounded distance decodable, nor do we know whether there are polynomially many codewords in all balls of the radius. It is generally believed that the answers to both questions are no.

In this paper, we prove: (1) List decoding can not be done for radius $n-\hat{g}(n,k,q)$ or larger, unless the discrete logarithm over $\mathbf{F}_{q^{\hat{g}(n,k,q)-k}}$ is easy.

[*] This work is partially supported by NSF Career Award CCR-0237845.

[†] Partially supported by NSF.

(2) Let h and g be positive integers satisfying $q \geq \max(g^2, (h-1)^{2+\epsilon})$ and $g \geq (\frac{4}{\epsilon}+2)(h+1)$ for a constant $\epsilon > 0$. We show that the discrete logarithm problem over $\mathbf{F}_{q^h}$ can be efficiently reduced by a randomized algorithm to the bounded distance decoding problem of the Reed-Solomon code $[q, g-h]_q$ with radius $q-g$. These results show that the decoding problems for the Reed-Solomon code are at least as hard as the discrete logarithm problem over certain finite fields. For the list decoding problem of Reed-Solomon codes, although the infeasible radius that we obtain is much larger than the radius which is known to be feasible, it is the first non-trivial bound. Our result on the bounded distance decodability of Reed-Solomon codes is also the first of its kind. The main tools to obtain these results are an interesting connection between the problem of list-decoding of Reed-Solomon code and the problem of discrete logarithm over finite fields, and a generalization of Katz's theorem on representations of elements in an extension finite field by products of distinct linear factors.

1 Introduction

An error-correcting code C over a finite alphabet Σ is an injective map $\phi : \Sigma^k \rightarrow \Sigma^n$. When we need to transmit a message of k letters over a noisy channel, we apply the map on the message first (i.e. encode the message) and send its image (i.e. the codeword) of n letters over the channel. The Hamming distance between two sequence of letters of the same length is the number of positions where two sequences differ. A good error-correcting code should have a large *minimum distance* d, which is defined to be the minimum Hamming distance between two distinct codewords in $\phi(\Sigma^k)$. A received message, possibly corrupted, but with no more than $(d-1)/2$ errors, corresponds to a unique codeword and thus may be decoded into the original message despite errors occur during the communication.

Error-correcting codes are widely used in practice. They are mathematically interesting and intriguing. This subject has attracted the attention of theoretical computer science community recently. Several major achievements of theoretical computer science, notably the original proof of PCP theorem and de-randomization techniques, rely heavily on the techniques in error-correcting codes. We refer to the survey [19] for details.

For the purpose of efficient encoding and decoding, Σ is usually set to be the finite field $\mathbf{F}_q$ of q elements, and the map ϕ is $\mathbf{F}_q$-linear. Numerous error correcting codes have been proposed, among them, the Reed-Solomon codes are particularly important. They are deployed to transmit information from and to spaceships, and to store information in optical media [22].

Notation: For a polynomial $f(x)$ and a set $S = \{x_1, \cdots, x_n\}$, we use $(f(x))_{x \in S}$

to denote the vector obtained by evaluating $f(x)$ at the elements in S, that is,

$$(f(x))_{x\in S} = (f(x_1), \cdots, f(x_n)).$$

Let S be a subset of $\mathbf{F}_q$ with $|S| = n$. The Reed-Solomon code $[n,k]_q$, is the map from $(a_0, a_1, \cdots, a_{k-1}) \in \mathbf{F}_q^k$ to

$$(a_0 + a_1 x + \cdots + a_{k-1}x^{k-1})_{x\in S} \in \mathbf{F}_q^n.$$

The choice of S will not affect our results in this paper. Since any two different polynomials with degree $k-1$ can share at most $k-1$ points, the minimum distance of the Reed-Solomon code is $n-k+1$.

1.1 Related works

If the radius of a Hamming ball is less than half of the minimum distance, there is at most one codeword in the Hamming ball. Finding the codeword is called *unambiguous decoding*. It can be efficiently solved, see [2] for a simple algorithm. If we gradually increase the radius, there may be two or more codewords lying in some Hamming balls. Can we efficiently enumerate all the codewords in any Hamming ball of certain radius? This is the so called list decoding problem. The notion was first introduced by Elias [6]. There was virtually no progress on this problem for radius slightly larger than half of the minimum distance, until Sudan published his influential paper [18]. His result was subsequently improved, the current best algorithm [11] solves the list decoding problem for radius as large as $n - \sqrt{n(k-1)}$. The work [11] sheds new light on the list decodability of Reed-Solomon codes. To the other extreme, if the radius is greater than or equal to the minimum distance, there are exponentially many codewords in some Hamming balls.

The decoding problem of Reed-Solomon codes can be reformulated into the problem of *curve fitting* or *polynomial reconstruction*. In this problem, we are given n points

$$(x_1, y_1), (x_2, y_2), \cdots, (x_n, y_n)$$

in $\mathbf{F}_q^2$. The goal is to find polynomials of degree $k-1$ that pass at least g points. In this paper, we only consider the case when the n given points have distinct x-coordinates. If we allow multiple occurrences of x-coordinates, the problem is NP-hard [7, Theorem 6.1], and it is not relevant to the Reed-Solomon decoding problem. If $g \geq (n+k)/2$, it corresponds to the unambiguous decoding of Reed-Solomon codes. If $g > \sqrt{n(k-1)}$, the radius is less than $n - \sqrt{n(k-1)}$, which is essentially the Johnson radius [12, 10], the problem can be solved by the

Guruswami-Sudan algorithm [11]. If $g \leq k$, it is possible that there are exponentially many solutions, but finding one is very easy.

It is known that any Hamming ball of the Johnson radius contains only polynomially many codewords. In this paper, we study the following question: How large can we increase the radius before the list decoding problem or the bounded distance decoding problem become infeasible? The question has been intensively investigated for Reed-Solomon codes and other error-correcting codes. The case of general non-linear codes has been solved [7], where it was proved that there exist codes with exponentially many codewords in Hamming balls of radius a little bigger than Johnson radius. The case for linear codes is much harder. Some partial results have been obtained in [9, 8], where it was proved that there exist linear codes with super-polynomially many codewords in Hamming balls of radius close to Johnson radius. However, none of them apply to Reed-Solomon codes. No negative result is known about the list decoding of Reed-Solomon codes, except for a simple combinatorial bound given by Justesen and Hoholdt [13], which states that for any positive integer $g < n$, there exists at least one Hamming ball of radius $n - g$, which contains at least $\binom{n}{g}/q^{g-k}$ many codewords. This bound matches the intuition well. Consider an imaginary algorithm as follows: randomly select g points from the n input points, and use polynomial interpolation to get a polynomial of degree at most $g - 1$ which passes these g points. Then with probability $1/q^{g-k}$, for a random word in $\mathbf{F}_q^n$, the resulting polynomial has degree $k - 1$. The sample space has size $\binom{n}{g}$. Thus heuristically, the number of codewords in Hamming balls of radius $n - g$ is at least $\binom{n}{g}/q^{g-k}$ on the average. In the same paper, Justesen and Hoholdt also gave an upper bound for the radius of the Hamming balls containing a constant or fewer number of codewords.

1.2 Our results

If we gradually increase g, starting from k and going toward n, then $\binom{n}{g}/q^{g-k}$ will fall below 1 at some point. However, g is still very far away from $\sqrt{n(k-1)}$. Let $\hat{g}(n,k,q)$ be the smallest positive integer such that $\binom{n}{g}/q^{g-k}$ is no great than 1. Roughly speaking, a Hamming ball with a random center and the radius $n - \hat{g}(n,k,q)$ contains on average about one codeword. The following lemma shows that there is a gap between $\hat{g}(n,k,q)$ and $\sqrt{n(k-1)}$.

Lemma 1 *For positive integers $k < g < n$, if $g > \sqrt{nk}$, then $q^{g-k} \geq n^{g-k} > \binom{n}{g}$. This implies that $\hat{g}(n,k,q) \leq \sqrt{nk}$.*

For a fixed rate k/n, the radius $n - \hat{g}(n,k,q)$ has relative radius approaching the relative distance as n approaches infinity. However it is not known whether there exist Reed-Solomon codes such that some Hamming balls of radius $n - \hat{g}(n, k, q)$ contain exponential number of codewords. Then how hard is it to do list decoding for the radius $n - \hat{g}(n, k, q)$?

Instead of trying to find a Hamming ball with a large number of codewords of radius $n - \hat{g}(n, k, q)$, we take another approach. We show that even if there is only a small number of codewords in every Hamming ball of this radius, the list decoding problem is still infeasible, by relating this question to discrete logarithm over finite fields. The discrete logarithm problem in finite field $\mathbf{F}_{q^m}$, is to compute an integer e such that $t = \gamma^e$, given a generator γ of a subgroup of $\mathbf{F}_{q^m}^*$ and t in the subgroup. The general purpose algorithms to solve the discrete logarithm problem are the number field sieve and the function field sieve (for a survey see [16]). They have a conjectured subexponential time complexity

$$\exp(c(\log q^m)^{1/3}(\log\log q^m)^{2/3})$$

for some constant c, when q is small, or m is small.

We prove that if the list decoding of the $[n, k]_q$ Reed-Solomon code is feasible for radius $n - \hat{g}(n, k, q)$, then the discrete logarithm over $\mathbf{F}_{q^{\hat{g}(n,k,q)-k}}$ is easy. In other words, we prove that the list decoding is not feasible for radius $n - \hat{g}(n, k, q)$ or larger, assuming that the discrete logarithm over $\mathbf{F}_{q^{\hat{g}(n,k,q)-k}}$ is hard. Note that it does not rule out the possibility that there are only polynomially many codewords in all Hamming balls of radius $n - \hat{g}(n, k, q)$, even assuming the intractability of the discrete logarithm over $\mathbf{F}_{q^{\hat{g}(n,k,q)-k}}$.

Theorem 1 *If there exists an algorithm solving the list decoding problem of radius $n - \hat{g}(n, k, q)$ for the Reed-Solomon code $[n, k]_q$ in random time $q^{O(1)}$, then discrete logarithm over the finite field $\mathbf{F}_{q^{\hat{g}(n,k,q)-k}}$ can be computed in random time $q^{O(1)}$.*

Let us consider a numerical example. Set $n = 1000$, $k = 401$, $q = 1201$. The unambiguous decoding algorithm can correct up to $\lfloor (n - k + 1)/2 \rfloor = 300$ errors. The Guruswami-Sudan algorithm can correct $\lfloor n - \sqrt{n(k-1)} \rfloor = \lfloor 1000 - \sqrt{1000 * 400} \rfloor = 368$ errors. Can we list decode up to $n - \hat{g}(n, k, q) = 1000 - 499 = 501$ errors in reasonable time? The theorem shows that if we can, then the discrete logarithm over $\mathbf{F}_{1201^{98}}$ can be solved efficiently, which is widely regarded as unlikely at present.

When the list decoding problem is hard for certain radius, or a Hamming ball contains too many codewords for us to enumerate all of them, we can ask for an efficient *bounded distance decoding* algorithm, which only needs to output one of the codewords in the ball, or output the empty set in case that the ball does not contain any codeword. However, we prove that the bounded distance decoding is hard as well.

Theorem 2 *Let q be a prime power and h be a positive integer satisfying $q \geq \max(g^2, (h-1)^{2+\epsilon})$ and $g \geq (\frac{4}{\epsilon}+2)(h+1)$ for any constant $\epsilon > 0$. If the bounded distance decoding problem of radius $q-g$ for the Reed-Solomon code $[q, g-h]_q$ can be solved in random time $q^{O(1)}$, the discrete logarithm problem over $\mathbf{F}_{q^h}$ can be solved in random time $q^{O(1)}$.*

For q, g, h satisfying the conditions in the theorem,

$$\frac{\binom{q}{g}}{q^{g-(g-h)}} \geq \frac{(q/g)^g}{q^h} = \frac{q^{g-h}}{g^g} \geq \frac{(g^2)^{g-h}}{g^g} = g^{g-2h} \geq g^{4h/\epsilon}.$$

Hence there is a Hamming ball of radius $q-g$ containing exponentially many codewords. It is infeasible to do list decoding under these parameters. This result has a drawback that it can only be applied to the low rate codes, since $g-h \leq g \leq \sqrt{q}$.

It is generally believed that the list decoding problem and the bounded distance decoding for Reed-Solomon codes are computationally hard if the number of errors is greater than $n - \sqrt{n(k-1)}$ and less than $n-k$. This problem is even used as a hard problem to build public key cryptosystems and pseudo-random generators [15]. A similar problem, noisy polynomial interpolation [3], was proved to be vulnerable to the attack of lattice reduction techniques, hence is easier than originally thought. This raises concerns on the hardness of polynomial reconstruction problem. Our results confirm the belief that polynomial reconstruction problem is hard for certain parameters, under a well-studied hardness assumption in number theory and hence provide a guideline for selecting parameters for many protocols based on the problem.

1.3 Techniques

We rely on the idea of index calculus to prove these two theorems. Our application of index calculus however is different from its usual applications, in that we use it to prove a hardness result (a computational lower bound), rather than a computational upper bound. We naturally come across the following question in the proofs: In a finite field $\mathbf{F}_{q^h}$, for any α such that $\mathbf{F}_{q^h} = \mathbf{F}_q[\alpha]$, can $\mathbf{F}_q + \alpha$ generate the multiplicative group $(\mathbf{F}_{q^h})^*$? This interesting problem has a lot of applications in graph theory, and it has been studied by several number theorists. Chung [5] proved that if $q > (h-1)^2$, then $(\mathbf{F}_{q^h})^*$ is generated by $\mathbf{F}_q + \alpha$. Wan [21] showed a negative result that if $q^h - 1$ has a divisor $d > 1$ and $h \geq 2(q \log_q d + \log_q(q+1))$, then $(\mathbf{F}_{q^h})^*$ is not generated by $\mathbf{F}_q + \alpha$ for some α. Katz [14] applied the Lang-Weil method, and showed that for every $h \geq 2$ there exists a constant $B(h)$ such that for any finite field $\mathbf{F}_q$ with $q \geq B(h)$, any element in $(\mathbf{F}_{q^h})^*$ can be written as a product of exactly $n = h + 2$ distinct elements from $\mathbf{F}_q + \alpha$. By a simple counting argument, $B(h)$ has to be an exponential function in h. In this paper, we use Weil's character sum estimate and a simple sieving to prove that if $q \geq \max(g^2, (h-1)^{2+\epsilon})$ and $g \geq (\frac{4}{\epsilon} + 2)(h+1)$ for any constant $\epsilon > 0$, then any element in $(\mathbf{F}_{q^h})^*$ can be written as a product of exactly g distinct elements from $\mathbf{F}_q + \alpha$. In comparison to Katz's theorem, we use a bigger n and manage to decrease $B(h)$ to a polynomial function in h.

This paper is organized as follows. In Section 2, we show the connection between the decoding problem of Reed-Solomon codes and the discrete logarithm problem over finite fields. In Section 3, we present the proof of Theorem 1. In Section 4, we present the proof of Theorem 2. In Section 5, we show an interesting duality between the size of a group generated by linear factors, and the list size in Hamming balls of Reed-Solomon codes. In Appendix, Section A, we prove Lemma 1.

2 The decoding problem and the discrete logarithm

Let q be a prime power and let $\mathbf{F}_q$ be the finite field with q elements. Let S be a subset of $\mathbf{F}_q$ of n elements. For a positive integer $g \leq n$, denote

$$\mathcal{P}(S, g) = \{A | A \subseteq S, |A| = g\}.$$

Clearly, the set $\mathcal{P}(S, g)$ has $\binom{n}{g}$ elements. For any $A \in \mathcal{P}(S, g)$, let

$$P_A(x) = \prod_{a \in A} (x - a).$$

This is a monic polynomial of degree g which splits over $\mathbf{F}_q$ as a product of distinct linear factors.

Let $1 < h < g$ be integers. Let $h(x)$ be an irreducible monic polynomial over $\mathbf{F}_q$ of degree h. Define a map

$$\psi : \mathcal{P}(S, g) \rightarrow \mathbf{F}_q[x]/(h(x))$$

by

$$\psi(A) = P_A(x) \pmod{h(x)}.$$

For any $f(x)$ in $\mathbf{F}_q[x]/(h(x))$ with degree at most $h - 1$, if $\psi^{-1}(f(x))$ is not empty, then there exists at least one monic polynomial $t(x) \in \mathbf{F}_q[x]$ of degree $g - h$ and one $A \in \mathcal{P}(S, n)$ such that

$$f(x) + t(x)h(x) = P_A(x).$$

For any $a \in A$, $P_A(a) = 0$ and $t(a) = -f(a)/h(a)$. Recall that $h(x)$ is irreducible over $\mathbf{F}_q$ hence $h(a) \neq 0$ for all $a \in A$. Since $f(x) + t(x)h(x)$ has degree g, there are exactly g elements in S which are the roots of $f(x) + t(x)h(x) = 0$, the curve $y = t(x)$ passes exactly g points in the following set of n points:

$$\{(a, -f(a)/h(a)) | a \in S\}.$$

For any polynomial $f \in \mathbf{F}_q[x]$ of degree at most $h-1$, let $T_{f(x)}$ be the set of monic polynomial $t(x) \in \mathbf{F}_q[x]$ of degree $g - h$ such that $f(x) + t(x)h(x) = P_A(x)$ for some $A \in \mathcal{P}(S, g)$. Let $C_{f(x)}$ be the set of codewords with distance exactly $n - g$ to the received word $(-f(a)/h(a) - a^{g-h})_{a \in S}$ in Reed-Solomon code $[n, g-h]_q$. It is then easy to prove

Lemma 2 There is a one-to-one correspondence between elements of $T_{f(x)}$ and $C_{f(x)}$*, by sending any* $t(x) \in T_{f(x)}$ *to* $(t(a) - a^{g-h})_{a \in S}$.

Remark 1 *According to the pigeonhole principle, there must exist a polynomial* $\hat{f}(x)$ *such that*

$$|\psi^{-1}(\hat{f}(x))| \geq \frac{|\mathcal{P}(S, g)|}{|\mathbf{F}_q[x]/(h(x))|} = \frac{\binom{n}{g}}{q^h}.$$

This provides another proof that there is a Hamming ball of radius $n - g$ *with* $\frac{\binom{n}{g}}{q^h}$ *many codewords.*

Suppose that we know $f(x)$ and $h(x)$, but not $t(x)$, can we still find A? Formally we are asking the following question:

Input: A prime power q, an irreducible polynomial $h(x)$ over $\mathbf{F}_q$ of degree h, a polynomial $f(x) \in \mathbf{F}_q[x]$, a positive integer g and a set $S \subseteq \mathbf{F}_q$.

Problem I: A list of all the subsets $A \in \mathcal{P}(S, g)$ such that

$$f(x) \equiv P_A(x) \pmod{h(x)}.$$

Problem II: One of $A \in \mathcal{P}(S, g)$ such that

$$f(x) \equiv P_A(x) \pmod{h(x)}.$$

Lemma 3 *Problem I can be reduced in polynomial time to the list decoding problem of Reed-Solomon code* $[n, g-h]_q$ *at radius* $n-g$. *Problem II can be reduced in polynomial time to the bounded distance decoding problem of Reed-Solomon code* $[n, g-h]_q$ *at radius* $n - g$.

Proof: The vector $(-f(a)/h(a) - a^{g-h})_{a \in S}$ can be calculated from the input. Using list decoding algorithm or bounded distance decoding algorithm, we can compute $t(x)$ of degree at most $g - h - 1$ such that $t(a) = -f(a)/h(a) - a^{g-h}$ at g many a's. We find A by factoring $f(x) + (t(x) + x^{g-h})h(x)$. □

If A can be found, then in the field $\mathbf{F}_q[x]/(h(x))$, $f(x)$ can be represented as a product of elements from a small set. It is called a *smooth* representation with *factor bases* $x - S$ in computational number theory. The capability of finding smooth representation constitutes a powerful attack against hard number theory problems like integer factorization and the discrete logarithm over finite fields. The lemma implies that decoding Reed-Solomon codes provides a way to find a smooth representation of any field element. Thus naturally an efficient decoding algorithm produces an attack for the discrete logarithm over finite fields. This idea first appeared in [4], and it provides a general framework for the following proofs.

3 The proof of Theorem 1

Given a Reed-Solomon code $[n,k]_q$, let $h = \hat{g}(n,k,q) - k$. Recall that $\hat{g}(n,k,q)$ is the smallest positive integer such that $\binom{n}{g}/q^{g-k}$ is no great than 1, and h is the degree of an irreducible polynomial $h(x)$. We show that there is an efficient algorithm to solve the discrete logarithm over $\mathbf{F}_{q^h} = \mathbf{F}_q[x]/(h(x))$ if there is an efficient list decoding algorithm for the Reed-Solomon code $[n,k]_q$ with radius $n - \hat{g}(n,k,q) = n - k - h$.

Let $\alpha = x \pmod{h(x)}$. Suppose that we are given the base $b(\alpha)$ and we need to find out the discrete logarithm of $v(\alpha)$ with respect to the base, where b and v are polynomials over $\mathbf{F}_q$ of degree at most $h-1$. Select any $S \subseteq \mathbf{F}_q$, $|S| = n$. We use the index calculus algorithm with factor bases $(\alpha - a)_{a \in S}$.

```
Algorithm 1    1. Initialize an empty set of linear equations.
  2. Repeat $n$ times
      (a) Randomly select an integer $i$ between $0$ and $q^h - 2$. Compute $f(x) = b(x)^i \pmod{h(x)}$.
      (b) Apply the list decoding algorithm to find the list of $A \in \mathcal{P}(S, \hat{g}(n,k,q))$ such that $f(x) \equiv P_A(x) \pmod{h(x)}$. If the list is empty, go back to 2a.
      (c) Otherwise we have relations
          $$f(\alpha) = \prod_{a \in A_1} (\alpha - a) = \cdots = \prod_{a \in A_l} (\alpha - a)$$
```

for some $A_1, A_2, \cdots, A_l \in \mathcal{P}(S, \hat{g}(n,k,q))$, where l is the list size. From the relations, we obtain linear equations mod $(q^h - 1)$:

$$i \equiv \sum_{a \in A_1} \log_{b(\alpha)}(\alpha - a) \equiv \cdots \equiv \sum_{a \in A_l} \log_{b(\alpha)}(\alpha - a).$$

Add them to the set of linear equations.

3. *For all $s \in S$ do*

   (a) *Randomly select an integer i between 0 and $q^h - 2$. Compute $f(x) = b(x)^i/(x-a) \pmod{h(x)}$.*

   (b) *Apply the list decoding algorithm to find the list of $A \in \mathcal{P}(S, \hat{g}(n,k,q))$ such that $f(x) \equiv P_A(x) \pmod{h(x)}$. If the list is empty, go back to 3a.*

   (c) *Otherwise we have* relations

   $$f(\alpha) = \prod_{a \in A_1} (\alpha - a) = \cdots = \prod_{a \in A_l} (\alpha - a)$$

   for some $A_1, A_2, \cdots, A_l \in \mathcal{P}(S, \hat{g}(n,k,q))$, where l is the list size. From the relations, we obtain linear equations mod $(q^h - 1)$:

   $$\begin{aligned} i \quad & \equiv \sum_{a \in A_1} \log_{b(\alpha)}(\alpha - a) + \log_{b(\alpha)}(\alpha - s) \equiv \cdots \\ & \equiv \sum_{a \in A_l} \log_{b(\alpha)}(\alpha - a) + \log_{b(\alpha)}(\alpha - s). \end{aligned}$$

   Add them to the set of linear equations.

4. *In these equations, $\log_{b(\alpha)}(\alpha - a)$, $a \in S$, are unknowns. If the system has full rank, solve it; Otherwise go back to Step 2.*

5. *Randomly select an integer i between 0 and $q^h - 2$. Compute $f(x) = b(x)^i v(x) \pmod{h(x)}$.*

6. *Apply the list decoding algorithm to find a list of $A \in \mathcal{P}(S, g)$ such that $f(x) \equiv P_A(x) \pmod{h(x)}$. If the list is empty, go back to 5.*

7. *Otherwise we have a relation*

$$f(\alpha) = \prod_{a \in A} (\alpha - a).$$

Hence

$$i + \log_{b(\alpha)} v(\alpha) = \sum_{a \in A} \log_{b(\alpha)}(\alpha - a),$$

we can solve $\log_{b(\alpha)} v(\alpha)$ *immediately.*

Now we analyze the time complexity of the algorithm. An efficient list decoding algorithm implies:

1. There are only polynomially many codewords in any Hamming ball of radius $n - \hat{g}(n,k,q)$, which along with Lemma 2 implies that $|\psi^{-1}(f)| \le q^c$ for any $f \in \mathbf{F}_{q^h}$ and a constant c. Hence

$$|\psi(\mathcal{P}(S, \hat{g}(n,k,q)))| \ge \frac{\binom{n}{\hat{g}(n,k,q)}}{q^c} \ge \frac{q^{\hat{g}(n,k,q)-k}}{q^c} = \frac{q^h}{q^c}.$$

   Thus in Step 2b, Step 3b and Step 6, since $f(\alpha)$ is a random element in $\mathbf{F}^*_{q^h}$, the list decoding algorithm outputs nonempty list with probability bigger than $1/q^c$. Note that $\frac{1}{q} \le \frac{\binom{n}{\hat{g}(n,k,q)}}{q^{\hat{g}(n,k,q)-k}} \le 1$.

2. And they can be found in polynomial time. Each step will take polynomial time. Thus all steps in the algorithm runs in polynomial time.

So we only need to show

Lemma 4 *The linear system can yield a unique solution with high probability after polynomially number of iterations of the main loop (from Step 2 to Step 4).*

Informally since i is picked randomly, the probability that a new equation is linearly independent to previous ones is very high at the beginning of the algorithm. It would not take long time before we have an independent linear system. Solving the system of equations gives us $\log_{b(\alpha)}(\alpha - a)$ for all $a \in S$.

Proof: (of Lemma 4) The linear system is defined in the ring $\mathbf{Z}/(q^h - 1)$, which is usually not a field. We may proceed with the linear system solver. If the algorithm encounters a zero-divisor in the ring, we can factor $q^h - 1$. We then apply the linear system solver to each modulus. We get the solution for the original system using the Chinese Remainder Theorem. In the case that a modulus is a prime power, we can solve the linear system modulo the prime first, and use Hensel lifting to solve the system modulo the prime power. Since $q^h - 1$ has at most $h \log q$ many distinct prime factors, this issue will slow the algorithm down only by a polynomial factor. Now we may assume that the linear system is defined over a finite field.

Let $w = \lceil 2 \log n \rceil + 3$. Let T be the set of binary vectors with length n and weight g. If the iteration of the main loop is repeated w times, we have selected in Step 2a nw many integers $i_1, i_2, \cdots, i_{nw}$, and in Step 3a nw many integers $i_{nw+1}, \cdots, i_{2nw}$, and obtained relations for $b^{i_j}(\alpha)$ ($1 \leq j \leq nw$) and $b^{i_j}(\alpha)/(\alpha - a)$ ($nw + 1 \leq j \leq 2nw$ and $a \in S$). This amounts to selecting at least $2nw$ vectors from T independently. And $\{\log_{b(\alpha)}(\alpha - s) | s \in S\}$ forms a basis for the linear system. According to the following proposition, proved in [17], we get n independent equations with probability more than $1 - \frac{1}{2n}$. Note that it is *not* required that the vectors are selected uniformly. This finishes the proof of Theorem 1. □

Proposition 1 *Let V be a vector space over a field* $\mathbf{F}$ *with* $\dim V = n$. *Let T be a finite set of vectors in V and let $e_1, \cdots, e_n$ be a basis of V. Let $w = \lceil 2 \log n \rceil + 3$. Suppose we choose $2nw$ vectors $u_1, u_2, \cdots, u_{nw}, v_1, v_2, \cdots, v_{nw}$ independently from T. Let V' be the subspace of V spanned by $u_1, u_2, \cdots, u_{nw}$ and the vectors $e_j + v_{(j-1)w+i}$ for $j = 1, 2, \cdots, n$ and $i = 1, \cdots, w$. Then with probability at least $1 - \frac{1}{2n}$, we have that $V = V'$.*

4 The proof of Theorem 2

We first prove the following number theoretic result.

Theorem 3 *Let h be a positive integer. Assume $q \geq \max(g^2, (h-1)^{2+\epsilon})$ and $g \geq (\frac{4}{\epsilon} + 2)(h + 1)$ for a constant $\epsilon > 0$. Then for any α such that $\mathbf{F}_q[\alpha] = \mathbf{F}_{q^h}$, every element in $\mathbf{F}^*_{q^h}$ can be written as a product of exactly g distinct factors from $\{\alpha + a | a \in \mathbf{F}_q\}$.*

Proof: We follow the method used in [21]. Fix an α such that $\mathbf{F}_q[\alpha] = \mathbf{F}_{q^h}$. For $\beta \in \mathbf{F}^*_{q^h}$, let $N_g(\beta)$ denote the number of solutions of the equation

$$\beta = \prod_{i=1}^{g} (\alpha + a_i), \ a_i \in \mathbf{F}_q,$$

where the a_i's are distinct. Permutations of a_i's are counted as different solutions. We need to show that the number $N_g(\beta)$ is always positive if $q \geq \max(g^2, (h - 1)^{2+\epsilon})$ and $g \geq (\frac{4}{\epsilon} + 2)(h + 1)$.

Let G be the character group of the multiplicative group $\mathbf{F}^*_{q^h}$. That is, G is the set of group homomorphisms from $\mathbf{F}^*_{q^h}$ to $\mathbf{C}^*$. The group G is a cyclic group of order $q^h - 1$ and thus

$$\sum_{\chi \in G} \chi(\prod_{i=1}^{g}(\alpha + a_i)/\beta) = \begin{cases} q^h - 1, & \text{if } \beta = \prod_i(\alpha + a_i), \\ 0, & \text{otherwise.} \end{cases}$$

From this, we deduce

$$N_g(\beta) = \frac{1}{q^h - 1} \sum_{a_i \in \mathbf{F}_q,\ a_i \text{ distinct}} \sum_{\chi \in G} \chi^{-1}(\beta)\chi(\prod_{i=1}^{g}(\alpha + a_i)).$$

Since the second summand is always non-negative, a simple inclusion-exclusion sieving implies that

$$\begin{aligned} N_g(\beta) \quad \geq \quad & \frac{1}{q^h - 1}(\sum_{a_i \in \mathbf{F}_q, 1 \leq i \leq g} - \sum_{1 \leq i_1 < i_2 \leq g} \sum_{a_i \in \mathbf{F}_q, a_{i_1} = a_{i_2}}) \\ & \sum_{\chi \in G} \chi^{-1}(\beta)\chi(\prod_{i=1}^{g}(\alpha + a_i)). \end{aligned}$$

For non-trivial character χ, one has the well-known Weil estimate [21]

$$|\sum_{a \in \mathbf{F}_q} \chi(\alpha + a)| \leq (h-1)\sqrt{q}$$

and thus

$$|\sum_{a_i \in \mathbf{F}_q, 1 \leq i \leq g} \prod_{i=1}^{g} \chi(\alpha + a_i)| \leq (h-1)^g q^{g/2}.$$

If $\chi^2 \neq 1$, then for fixed $i_1 < i_2$, Weil's estimate implies that

$$|\sum_{a_i \in \mathbf{F}_q, a_{i_1} = a_{i_2}} \prod_{i=1}^{g} \chi(\alpha + a_i)| \leq (h-1)^{g-1} q^{(g-1)/2} \leq (h-1)^g q^{g/2}.$$

If $\chi^2 = 1$ but $\chi \neq 1$, then for fixed $i_1 < i_2$, Weil's estimate implies that

$$|\sum_{a_i \in \mathbf{F}_q, a_{i_1} = a_{i_2}} \prod_{i=1}^{g} \chi(\alpha + a_i)| \leq q(h-1)^{g-2} q^{(g-2)/2} \leq (h-1)^g q^{g/2}.$$

Separating the trivial character from the above lower estimate for $N_g(\beta)$, we deduce that

$$N_g(\beta) \geq \frac{q^g - \binom{g}{2} q^{g-1}}{q^h - 1} - (1 + \binom{g}{2})(h-1)^g q^{g/2}.$$

In order for $N_g(\beta) > 0$, it suffices to have the inequality

$$(q - \binom{g}{2})q^{g/2-1-h} > (1 + \binom{g}{2})(h-1)^g.$$

This inequality is clearly satisfied if both $q > 2\binom{g}{2} + 1 = g(g-1) + 1$ and $q^{g/2-1-h} > (h-1)^g$. These two inequalities are satisfied if we take $q \geq \max(g^2, (h-1)^{2+\epsilon})$ and $g \geq (\frac{4}{\epsilon} + 2)(h+1)$. The theorem is proved. □

Remark 2 *As we should see immediately, the expression $N_g(\beta)/g!$ is the number of codewords in the Hamming ball with radius $n - g$ and with center at $(-f(a)/h(a) - a^{g-h})_{a\in\mathbf{F}_q}$ where $h(x)$ is the minimum polynomial of α over $\mathbf{F}_q$ and $f(x)$ is the polynomial of degree at most $h-1$ representing β. Adjusting the parameters will give us an exponential lower bound for $N_g(\beta)/g!$. For example, if $q > 2\binom{g}{2} + 2$ and $g \geq (\frac{4}{\epsilon} + 2)(h+1)$, the same proof shows that $N_g(\beta) \geq q^{g-h-1}$ and thus*

$$\frac{N_g(\beta)}{g!} \geq \frac{q^{g-h-1}}{g!}$$

which is exponential for certain parameters q, g and h.

Let $q \geq \max(g^2, (h-1)^{2+\epsilon})$ and $g \geq (\frac{4}{\epsilon} + 2)(h+1)$. Let $h(x)$ be an irreducible polynomial over $\mathbf{F}_q$ of degree h. Then $\mathbf{F}_{q^h} = \mathbf{F}_q[x]/(h(x))$. Denote $x \pmod{h(x)}$ by α. Suppose that there exists a polynomial time algorithm to do bounded distance decoding of Reed-Solomon code $[q, g-h]_q$ at radius $g - q$. We prove Theorem 2 by describing a polynomial time algorithm to solve the discrete logarithm of $v(\alpha)$ with base $b(\alpha)$ in $\mathbf{F}_{q^h}$, where b and v are polynomials of degree at most $h-1$. We let $S = \mathbf{F}_q$. This algorithm relies on the index calculus idea as in Algorithm 1. It is simpler, as for any polynomial $f(x) \in \mathbf{F}_q[x]$ with degree $g - h - 1$ or less, if we run the bounded decoding algorithm with input word $\{(x, -f(x)/h(x))|x \in \mathbf{F}_q\}$ and distance bound $n - g$, the answer is never empty, according to Theorem 3 and Lemma 2.

Algorithm 2 *1. Initialize an empty set of linear equations.*

```
2. Repeat $n$ times
    (a) Randomly select an integer $i$ between $0$ and $q^h - 2$.
    (b) Compute $b(\alpha)^i$, and let $f(x) = b^i(x) \pmod{h(x)}$.
```

   (c) *Run the bounded distance decoding algorithm to find* $A = \mathcal{P}(\mathbf{F}_q, g)$ *such that*

$$b(\alpha)^i = \prod_{a \in A} (\alpha - a).$$

   We have

$$i \equiv \sum_{a \in A} \log_{b(\alpha)} (\alpha - a) \pmod{q^h - 1}.$$

   Add it to the set of linear equations.

3. *For all* $s \in S$ *do*

   (a) *Randomly select an integer* i *between* 0 *and* $q^h - 2$. *Let* $f(x)$ *denote the element* $b(x)^i/(x - a) \pmod{h(x)}$.

   (b) *Run the bounded distance decoding algorithm to find* $A \in \mathcal{P}(\mathbf{F}_q, g)$ *such that*

$$b(\alpha)^i/(\alpha - s) = \prod_{a \in A} (\alpha - a).$$

   We get

$$i \equiv \sum_{a \in A} \log_{b(\alpha)} (\alpha - a) + \log_{b(\alpha)} (\alpha - s) \pmod{q^h - 1}.$$

   Add it to the set of linear equations.

4. *In these equations,* $\log_{b(\alpha)} (\alpha - a)$, $a \in S$, *are unknowns. If the system has full rank, solve it; Otherwise go back to Step 2.*

5. *Apply the bounded distance decoding algorithm to find relation*

$$v(\alpha) = \prod_{a \in A} (\alpha - a)$$

   Hence

$$\log_{b(\alpha)} v(\alpha) = \sum_{a \in A} \log_{b(\alpha)} (\alpha - a).$$

We can essentially copy the proof of Lemma 4 to prove that we only need to try $O(n \log n)$ many i's before we solve the discrete logarithm of $w(\alpha)$ with base $b(\alpha)$ with probability $1 - \frac{1}{2n}$. It is very crucial here that we have Step 3, because the system may not have the full rank if we only have Step 2. This is the case, if all the A_i's in Step 2 come from a subset of $\mathbf{F}_q$. An easy consequence of Theorem 2 is as follows. Taking $\epsilon = 2$ and $g = 4h + 4$ in Theorem 2, we get

Corollary 1 *Let q be a prime power and let h be a positive integer satisfying $q > \max((4h+4)^2, (h-1)^4)$. If the bounded distance decoding problem of radius $q - 4h - 4$ for the Reed-Solomon code $[q, 3h+4]_q$ can be solved in time $q^{O(1)}$, the discrete logarithm problem over $\mathbf{F}_{q^h}$ can be solved in random time $q^{O(1)}$.*

5 Group size and list size

Let S be a subset of $\mathbf{F}_q$ of n elements. Let α be an element in $\mathbf{F}_{q^h}$ such that $\mathbf{F}_q[\alpha] = \mathbf{F}_{q^h}$, and h is very small compared to q. What is the order of the subgroup generated by $\alpha + S$? This question has an important application in analyzing the performance of the AKS primality testing algorithm [1]. Experimental data suggests that the order is greater than $q^{h/c}$ for some absolute constant c for all $|S| \geq h \log q$. If it can be proved, the space complexity of the AKS algorithm can be cut by a factor of $\log p$ (p is the input prime whose primality certificate is sought), which will make (the random variants of) the algorithm comparable to the primality proving algorithm used in practice. However, the best known lower bound is $(c|S|/h)^h$ for some absolute constant c [20]. We present an interesting duality between the group size and the list size in Hamming balls of certain radius.

Theorem 4 *Let q be a prime power. Let α be an element in the extension of $\mathbf{F}_q$ with degree h. Let $\omega_\alpha(n)$ be the smallest possible order for the group generated by $\alpha + S$ multiplicatively, where $S \in \mathbf{F}_q$ and $|S| = n$. Let $A_q^{RS}(n, d, w)$ be the maximum list size in the Hamming balls of radius w in any Reed-Solomon code with block length n and minimum distance d over $\mathbf{F}_q$. For any integer $k < n - h$, we have*

$$A_q^{RS}(n, n-k, n-k-h) \times \omega_\alpha(n) \geq \binom{n}{k+h}.$$

Proof: Let $\omega_\alpha(n, S)$ be the order of the group generated by $\alpha + S$, where $S \in \mathbf{F}_q$ and $|S| = n$. Let $h(x)$ be the minimum polynomial of α. Consider the mapping:

$$\psi : \mathcal{P}(S, k+h) \rightarrow \mathbf{F}_q[x]/(h(x)).$$

The range of ψ consists of elements which can be represented as products of $k+h$ distinct elements in $\alpha + S$, thus it has cardinality at most $\omega_\alpha(n, S)$. For any element in $\mathbf{F}_q[x]/(h(x))$, the number of its pre-images is at most $A_q^{RS}(n, n-k, n-k-h)$, according to Lemma 2. Hence

$$\begin{aligned} & A_q^{RS}(n, n-k, n-k-h) \times \omega_\alpha(n, S) \\ \geq \quad & |\mathcal{P}(S, k+h)| = \binom{n}{k+h}. \end{aligned}$$

This implies the theorem. □

Corollary 2 *Let k, n be positive integers and q be a prime power. One of the following statements must be true.*

1. *For any constant c_1, there exists a Reed-Solomon code $[n, k]_q$ ($n/3 < k < n/2$), and a Hamming ball of radius $n - \hat{g}(n, k, q)$ containing more than $c_1 1.9^n$ codewords.*

2. *The group generated by $\alpha + S$, has cardinality at least q^{h/c_2} for some absolute constant c_2, where $S \subseteq \mathbf{F}_q$ and $|S| = \lfloor h \log q \rfloor$.*

To prove or disprove the first statement would solve an important open problem about the Reed-Solomon codes. Recall that a Hamming ball with a random center and the radius $n - \hat{g}(n, k, q)$ contains on average one codeword. To prove the second statement would give us a primality proving algorithm much more efficient in term of space complexity than the original AKS and its random variants, hence making the AKS algorithm not only theoretically interesting, but also practically important. However, at this stage we cannot figure out which one is true. What we can prove, however, is that one of them must be true. Note that it is also possible that both statements are true.

Proof: (of Corollary 2) Let $k = \lfloor h \log q/2 \rfloor - h$ and $n = \lfloor h \log q \rfloor$. So the rate k/n is very close to $1/2$ as q gets large. Since $\binom{n}{\lfloor h \log q/2 \rfloor}$ is about $2^{h \log q} = q^h = q^{h \log q/2 - k}$, $\hat{g}(n, k, q) = h \log q/2 + O(1)$. Assume the first statement is false, this means that there exists a constant c_3 such that for any Reed-Solomon code $[n, k]_q$ with $n/3 < k < n/2$, the number of codewords in any Hamming ball of radius $n - \hat{g}(n, k, q)$ is less than $c_3 1.9^n$. That is,

$$A_q^{RS}(n, n-k-h, n - \hat{g}(n, k, q)) \leq c_3 1.9^n$$

Hence the size of group generated by $\alpha + S$ is at least

$$\frac{\binom{n}{\hat{g}(n,k,q)}}{c_3 1.9^n} \geq \frac{q^{h+O(1)}}{c_4 1.9^n} = \frac{q^{h - n \log 1.9/\log q + O(1)}}{c_4} \geq q^{h/c_2}.$$

□

6 Open problems

There is a large gap between $n - \sqrt{n(k-1)}$ and $n - \hat{g}(n, k, q)$, where we do not know list decoding for Reed-Solomon codes $[n, k]_q$ is feasible or not. In Theorem 3, the condition $q \geq g^2$ is still quadratic. It would be very interesting to obtain positive results with only linear condition $q \geq cg$ for some positive constant c. Other interesting open questions include whether there exists a reversal reduction which maps the list or bounded distance decoding problem of Reed-Solomon code for the parameters studied in the paper to discrete logarithm over finite fields, and whether there exists a polynomial time quantum algorithm to solve these decoding problems.

Acknowledgments We thank Professor Chaohua Jia for helpful discussion on the proof of Theorem 3.

References

[1] Manindra Agrawal, Neeraj Kayal, and Nitin Saxena. Primes is in P. http://www.cse.iitk.ac.in/news/primality.pdf, 2002. To appear in *Annals of Mathematics*.

[2] E. Berlekamp and L. Welch. Error correction of algebraic block codes. U.S. Patent Number 4633470, 1986.

[3] Daniel Bleichenbacher and Phong Q. Nguyen. Noisy polynomial interpolation and noisy chinese remaindering. In *Proceedings of EuroCrypto*, volume 1807 of *Lecture Notes in Computer Science*, 2000.

[4] Qi Cheng. On the bounded sum-of-digits discrete logarithm problem in finite fields. In *Proc. of the 24th Annual International Cryptology Conference (CRYPTO)*, pages 201–212. Springer-Verlag, 2004.

[5] F.R.K. Chung. Diameters and eigenvalues. *Journal of American Mathematical Society*, 2(2):187–196, 1989.

[6] Peter Elias. List decoding for noisy channels. In *1957-IRE WESCON Convention Record*, pages 94–104, 1957.

[7] O. Goldreich, R. Rubinfeld, and M. Sudan. Learning polynomials with queries: the highly noisy case. *SIAM Journal on Discrete Mathematics*, 2000.

[8] V. Guruswami. Limits to list decodability of linear codes. In *Proc.* 34*th ACM Symp. on Theory of Computing*, 2002.

[9] V. Guruswami, J. Hastad, M. Sudan, and D. Zuckerman. Combinatorial bounds for list decoding. *IEEE Transactions on Information Theory*, 48(5):1021–1034, 2002.

[10] Venkatesan Guruswami. *List Decoding of Error-Correcting Codes*. PhD thesis, MIT, 2001.

[11] Venkatesan Guruswami and Madhu Sudan. Improved decoding of Reed-Solomon and algebraic-geometry codes. *IEEE Transactions on Information Theory*, 45(6):1757–1767, 1999.

[12] Selmer M. Johnson. A new upper bound for error-correcting codes. *IEEE Transactions on Information Theory*, 8:203–207, 1962.

[13] Jorn Justesen and Tom Hoholdt. Bounds on list decoding of MDS codes. *IEEE Transactions on Information Theory*, 47(4):1604–1609, 2001.

[14] Nicholas M. Katz. Factoring polynomials in finite fields: an application of Lang-Weil to a problem in graph theory. *Mathematische Annalen*, 286:625–637, 1990.

[15] Aggelos Kiayias and Moti Yung. Cryptographic hardness based on the decoding of Reed-Solomon codes. In *Proceedings of ICALP*, volume 2380 of *Lecture Notes in Computer Science*, 2002.

[16] A. M. Odlyzko. Discrete logarithms: The past and the future. *Designs, Codes, and Cryptography*, 19:129–145, 2000.

[17] Carl Pomerance. Fast, rigorous factorization and discrete logarithm algorithms. In *Discrete Algorithms and Complexity*. Academic Press, 1987.

[18] Madhu Sudan. Decoding of Reed-Solomon codes beyond the error-correction bound. *Journal of Complexity*, 13(1):180–193, 1997.

[19] Madhu Sudan. Coding theory: Tutorial & survey. In *Proc. 42th IEEE Symp. on Foundations of Comp. Science*, pages 36–53, 2001.

[20] Jose F. Voloch. On some subgroups of the multiplicative group of finite rings. http://www.ma.utexas.edu/users/voloch/preprint.html, 2003.

[21] Daqing Wan. Generators and irreducible polynomials over finite fields. *Mathematics of Computation*, 66(219):1195–1212, 1997.

[22] Stephen B. Wicker and Vijay K. Bhargava, editors. *Reed-Solomon Codes and Their Applications*. John Wiley and Sons, 1999.

A Proof of Lemma 1

In this section, we prove Lemma 1 by showing the following statement.

Theorem 5 *There are no positive integral solutions for the inequalities*

$$\begin{aligned} \binom{n}{g} &> n^h, & (1) \\ g &> \sqrt{n(g-h)}. & (2) \end{aligned}$$

We first obtain a finite range for h, g and n.

Lemma 5 *If (n, g, h) is a positive integral solution, then $h < 88$.*

Proof: Denote g/h by α and n/h by β. From $g > \sqrt{n(g-h)}$, we have $\alpha > \sqrt{\beta(\alpha-1)}$. Hence $\alpha < \beta < \alpha + 1 + \frac{1}{\alpha-1}$.

Recall that for any positive integer i, $\sqrt{2\pi i}(i/e)^i \leq i! \leq \sqrt{2\pi i}(i/e)^i(1 + \frac{1}{12i-1})$. We have also

$\binom{n}{g} = \binom{\beta h}{\alpha h} \leq (\frac{\beta^\beta}{\alpha^\alpha(\beta-\alpha)^{\beta-\alpha}})^h$.

Thus $\frac{\beta^\beta}{\alpha^\alpha(\beta-\alpha)^{\beta-\alpha}} \geq \beta h$, which implies

$$h \leq \frac{\beta^{\beta-1}}{\alpha^\alpha(\beta-\alpha)^{\beta-\alpha}}.$$

Recall more facts:

1. For $x > 0$, x^x takes the minimum value $0.6922..$ at $x = e^{-1} = 0.36787944....$
2. For $x > 0$, $1 \leq (1 + \frac{1}{x})^x \leq e = 2.7182818284...$

If $\alpha \geq 2$, then $\beta - \alpha \leq 1 + \frac{1}{\alpha-1} \leq 2$. We have

$$\begin{aligned} h &\leq \frac{1.45\beta^{\beta-1}}{\alpha^\alpha} \\ &\leq \frac{1.45(1+\alpha+\frac{1}{\alpha-1})^{(\alpha+\frac{1}{\alpha-1})}}{\alpha^\alpha} \\ &\leq 1.45(1+\alpha+\frac{1}{\alpha-1})^{(\frac{1}{\alpha-1})}(1+\frac{1}{\alpha}+\frac{1}{\alpha(\alpha-1)})^\alpha \\ &\leq 1.45*4*(1+\frac{1}{\alpha-1})^{\alpha-1}(1+\frac{1}{\alpha-1}) \\ &\leq 1.45*4*e*2 < 32. \end{aligned}$$

If $\alpha < 2$, $h \leq \frac{1.45\beta^{\beta-1}}{(\beta-\alpha)^{\beta-\alpha}}$. There are two cases. If $\beta \leq 3$, then

$$h \leq 1.45^2 * 9 < 19.$$

If $\beta > 3$, then

$$\begin{aligned} h &\leq 1.45(\frac{\beta}{\beta-\alpha})^{\beta-1}(\beta-\alpha)^{\alpha-1} \\ &\leq 1.45(\frac{\beta}{\beta-2})^{\beta-1}(1+\frac{1}{\alpha-1})^{\alpha-1} \\ &\leq 1.45 * (1+\frac{2}{\beta-2})^{\beta-2}(1+\frac{2}{\beta-2}) * e \\ &\leq 1.45 * e^2 * 3 * e < 88. \end{aligned}$$

□

Since $\alpha = \frac{g}{h}$, $g > h$ are both positive integers, we easily have,

Corollary 3 *$\alpha \geq 88/87$ and $\beta - \alpha < 88$.*

We are ready to prove the main theorem of this section.

Proof: (of Theorem 5) We claim that $\beta < 178$. If $\alpha < 89$, then $\beta < 178$. If $\alpha \geq 89$, then $\beta - \alpha \leq 1 + 1/88$, but $n - g = (\beta - \alpha)h$ is an integer, and $h \leq 87$, so $\beta - \alpha \leq 1$. This means that $n - g \leq h$, (1) can not hold.

We verify that there is no solution by exhaustively searching for the solutions in the finite range that $h < 88, n < 178 * 88 = 15664$ and $h < g < n$ in a computer. □

Denote $\frac{n}{g-k}$ by γ and $\frac{g}{g-k}$ by δ. To prove the second part of the lemma, it suffices to see that $\binom{n}{g} = \binom{\gamma(g-k)}{\delta(g-k)} \leq c_2^{g-k}$ for some constant c_2 depending only on γ and δ.

Similarly we can show that for any constant c, the inequalities

$$\binom{n}{g} \geq n^{h-c} \quad (3)$$

$$g > \sqrt{n(g-h)} \quad (4)$$

have only finite number of positive integral solutions.