UNIVERSITY OF OKLAHOMA

GRADUATE COLLEGE

STUDY OF MATRICES RELATED TO DISCRETE LOGARITHM IN KUMMER EXTENSIONS

A THESIS

SUBMITTED TO THE GRADUATE FACULTY

in partial fulfillment of the requirements for the

Degree of

MASTER OF SCIENCE

By

XUECHENG CHEN
Norman, Oklahoma

2016

STUDY OF MATRICES RELATED TO DISCRETE LOGARITHM
IN KUMMER EXTENSIONS

A THESIS APPROVED FOR THE
SCHOOL OF COMPUTER SCIENCE

BY

Dr. Qi Cheng, Chair

Dr. Sridhar Radhakrishnan

Dr. S. Lakshmivarahan

Acknowledgements

First of all, I'd like to thank my advisor, my teachers, my fiends and my family for the encouragement and support in pursuing this degree.

With the help of my advisor, Dr. Cheng, I learned so much in knowledge and research in computer science, especially in topics related to security aspect. His solid knowledge base in cryptography and number theory helped me gain sights in this field and his critical thinking of the questions affected me in research methods.

I also want to thank my committee members and teachers in our department for the help of different topics which are interesting and useful. As a transfer student from other major, I have a lot of things to learn about computer science. The courses and activities in thee department helped me to fill the gaps and move on to the new major.

During the last two and half years, my friends and family was always encouraging me in the school and life. My parents cared about me for the health and passion all the time and my wife has always been accompanying with me and built a sweet home together with me.

At last I'd like to mention the tornadoes and earthquakes in Oklahoma, which are unique experiences for an international student who has never seen these before. I'll always remember the great football team , the peaceful campus, and foremost

the university of Sooner and Boomer.

Contents

List of Tables

List of Figures

Abstract

The discrete logarithm problem has been studied during the past decades for its important application in cryptography and other fields. It is very useful in the public key cryptography, which is widely used for Internet safety. Using current computers to solve general discrete logarithm problem seems still not possible within reasonable time, since no polynomial time algorithms has been found for general cases. However, over finite fields of small characteristic, the factor base discrete logarithm can be solved much faster with heuristic polynomial time algorithms.

This thesis is mainly based on the previous study of factor base discrete logarithm in Kummer extension ($\mathbb{F}_{q^{2(q-1)}}$) which is published recently by Xiao-Zhuang-Cheng [14] and we focused on further calculation in this study. The previous research, based on the hypothesis of the determinant of lattices and the discrete logarithm, was confirmed with calculation for all q's such that $\log_2(q^{2(q-1)}) \leq 5000$, and in this thesis we pushed the limit to $\log_2(q^{2(q-1)}) \leq 10000$. During the calculation, we tried different strategies to improve the efficiency, by transferring the matrices and splitting q's into several groups. We achieved 1000% speed-up for most q's in the range and discovered some possible structures to group q's in calculation.

In the thesis, we'll go through the basic backgrounds of the study, and then introduce the main methods and experiments done in the study. We'll discuss the grouping of q's and the efficiency improvement. In the end we'll summarize the progress and the possible future work of this study.

Keywords: Cryptography, Discrete logarithm, Kummer extension, Lattice

Chapter 1

Introduction

1.1 Backgrounds

In the past several decades, public key cryptography was heavily based on discrete logarithm study [5], and we believe that it still takes longer than reasonable time to solve the general cases using current computing method and resources [13]. Although new methods were invented and improved as time went on, this assumption holds unless quantum computers are developed with productivity. However, some recent study showed that if the characteristic of the field is small, the calculation can be accelerated significantly [6, 7, 8, 9, 10, 11, 12]. With the help of index calculus, function field sieve and number field sieve [1, 2], we collect linear relations for the discrete logarithm, and then solve the discrete logarithm with the relations. The smoothness of the polynomial affects the efficiency if exhaustive search is used, but guided searching algorithm can accelerate the process with loss of correctness to assumptions of smoothness [3, 4].

In the public key cryptography, an important part is the Diffie-Hellman key

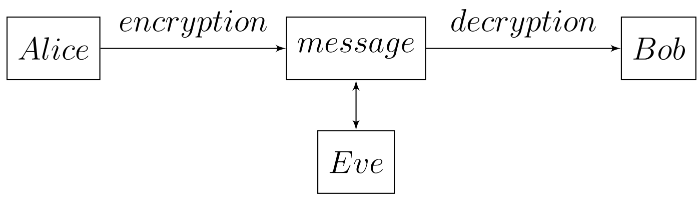

Figure 1.1: Cryptography in communication

exchange based on discrete logarithm. As is shown in Figure 1, if Alice and Bob wants to share the key in a safe channel, they can use this algorithm to get same secret while Eve can't solve it.

Assume Alice and Bob agree to use g and p as the base and modulus. At first Alice hold the secret a and Bob hold the secret b. Alice then transfers $A = g^a$ mod p to Bob and Bob transfers $B = g^b$ mod p to Alice. Now Alice can get the secret $s = B^a$ mod p and Bob can get the same secret $s = A^b$ mod p. Eve can see the g, p, A and B but can't get the secret s. Here we can see the algorithm relies on the difficulty of discrete logarithm with the current computing resources.

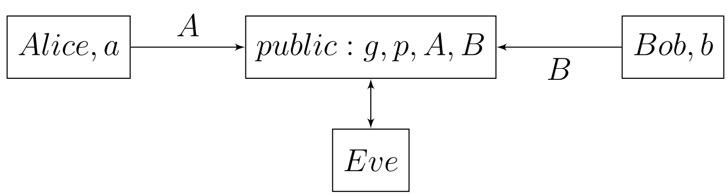

Figure 1.2: Diffie-Hellman public key exchange

In the previous research, matrices related to discrete logarithm in Kummer extension were studied in order to solve the factor base discrete logarithm problems [14]. The research went through the new algorithm from building the matrices related to the index calculus to solving and verifying the determinant of resulting matrices. The detailed steps will not be discussed in this study but main processes and definitions will be introduced in the later chapters.

1.2 Motivation

The previous research provided theoretical evidence to support the algorithm, as well as verified the results for all q's such that $\log_2(q^{2(q-1)}) \leq 5000$. However, it takes quite a long time to reach that limit with the current computers. Since the transformation of matrices related to the lattice is not suitable for parallel computing, it's important to speed up the process by shortening the time of each step. There are several ways of improving the calculation speed and the most efficient one we found was to reduce the size of elements in the matrices during the beginning period. However, it was not always working on current computers due to some possible special structures in the matrices. So we tried to reduce the matrix size by partitioning field at the beginning, and this worked for the outliers from the first case. We also try to minimize the space usage if it's possible although the new algorithm sacrifices space for time in most cases.

1.3 Progress

In this study, the main improvement from the previous work is to push the limit of the q's to $\log_2(q^{2(q-1)}) \leq 10000$ with several attempts in different aspects. The direct speed up from the new calculation of Hermite Normal Form accelerated the calculation by more than 1000% for eligible q's. For the outliers, the new calculation could not be handled by our computers, but by dividing those q's into cases and transforming the calculation to smaller matrices, we also got more than 1000% speed-up. As for the space usage, most cases were affordable in both cases, although the new algorithm has a weaker relation of space usage and q's. The calculation was done on a computer with 3.6GHz CPU and 32G memory.

We used SAGE under LINUX system for the study.

Chapter 2

Methods

2.1 Definitions

2.1.1 Finite Fields

We'll start the definitions from the group theory.

Definition 2.1. Group

The group is a structure of set of elements (G) and operations($\bullet$). To form a group, the operation need to be used on the elements and generate an element with the following properties:

Closure: $\forall$ a, b $\in G$, a $\bullet$ b $\in G$

Associativity: $\forall$ a, b, c $\in G$, (a $\bullet$ b) $\bullet$ c = a $\bullet$ (b $\bullet$ c)

Identity: $\exists$ e, $\forall$ a $\in G$, a $\bullet$ e = e $\bullet$ a = a

Invertibility: $\exists$ b, $\forall$ a $\in G$, a $\bullet$ b = b $\bullet$ a = e

From the definitions above, we denote the e as the identity element and $b = a^{-1}$ as the inverse of a. If the invertibility doesn't hold, we call that a monoid. If further the identity doesn't hold, we call that a semigroup.

Definition 2.2. Abelian Group

The abelian group is a group with the following property:

$\forall$ a, b $\in G$, a $\bullet$ b = b $\bullet$ a

With these definitions, we can see that integers ($\mathbb{Z}$) with addition (+) forms an abelian group $(\mathbb{Z}, +)$. If we remove 0 from $\mathbb{Q}$ (denoted as $\mathbb{Q}^* = \mathbb{Q} \setminus \{0\}$), it also forms an abelian group with multiplication $(\mathbb{Q}^*, *)$, otherwise it's a monoid.

Definition 2.3. Ring

The ring is a set(R) with two binary operations $(\bullet, \circ)$ satisfying the following properties:

$(R, \bullet)$ is an abelian group.

$(R, \circ)$ is a semigroup.

Distributive law holds: a $\circ$ (b $\bullet$ c) = (a $\bullet$ b) $\circ$ (a $\bullet$ c)

For integers it's easy to see that with + and $*$ we can form an integer ring ($\mathbb{Z}$, +, *). For example, $(\mathbb{Z}/m\mathbb{Z}, +, *)$ is an integer ring where $(\mathbb{Z}/m\mathbb{Z}) = \{0, 1, 2, ..., m-1\}$. We can easily verify the properties needed above.

Definition 2.4. Zero divisor

A zero divisor in the ring ($\mathbb{R}$) is, a non-zero element a $\in \mathbb{R}$, $\exists b \neq 0$ such that b * a = 0.

For example, in $(\mathbb{Z}/6\mathbb{Z}, +, *)$, 2 is a zero divisor since 2 * 3 = 0 in $(\mathbb{Z}/6\mathbb{Z}, +, *)$. Also, 3 and 4 are zero divisors in this ring but 5 is not because it has inverse of 5. Zero divisor doesn't have multiplicative inverse.

Definition 2.5. Field

A field $\mathbb{F}$ in number theory is usually defined as a ring in which each nonzero element has its multiplicative inverse.

For example, $(\mathbb{Q}, +, *)$ is a field since every element except 0 has its inverse. For prime number p, $(\mathbb{Z}/p\mathbb{Z}, +, *)$ is a field. Let p = 3, $(\mathbb{Z}/3\mathbb{Z}, +, *) = \{0, 1, 2\}$. We can easily see that 1 has its inverse 1 and 2 has its inverse 2.
For non-prime number m, $(\mathbb{Z}/m\mathbb{Z}, +, *)$ is not a field. Let m = 4, $(\mathbb{Z}/4\mathbb{Z}, +, *) = \{0, 1, 2, 3\}$. We can see that 2 doesn't have its inverse.

Definition 2.6. Characteristic of field
The characteristic of a field $\mathbb{F}$, $Ch(\mathbb{F})$ is defined as 0 or the smallest positive integer n such that $n * 1_{\mathbb{F}} = 0$ if it exists.

Definition 2.7. Finite field
A finite field is a field with finite number of elements.
Denote $\mathbb{F}_q$ be a finite field with q elements, where q is a prime number or a prime power.

With these basic field definitions, now we can describe the discrete logarithm problem.

2.1.2 Discrete logarithm

Discrete logarithm has been used in public key cryptography for a long time, and the definition can be described as follow:

Definition 2.8. Discrete logarithm over finite field.
For $\mathbb{F}_q$, if $\alpha^x = \beta$ was given for $\alpha, \beta \in \mathbb{F}_q^*$, the discrete logarithm problem is to find the integer x.

Here's an example of a simple discrete logarithm problem:

x	1	2	3	4	5	6
β	3	2	6	4	5	1

Table 2.1: A simple discrete logarithm in $\mathbb{F}_7$ when $\alpha = 3$

Here with the table we can find x for given β, but when the field is much larger, it'll take very long time since no polynomial algorithm has been found.

As is mentioned above, the general algorithms solving discrete logarithm problem are sub-exponential time complexity, such as number field sieve and function field sieve.

However, special cases with small characteristic field can be solved much faster. In these cases, the Kummer extension is very useful in testing the algorithms due to its structure. So we focus on these cases in the study.

2.1.3 Kummer Extensions

When we try to solve the discrete logarithm problem, the Kummer extension could be a starting point although it's considered not safe enough to be used in real world cryptography. The Kummer extension can be described as follow:

Definition 2.9. Kummer extension:

In general, a field extension L/K is called a Kummer extension if for some given integer $n > 1$:

K contains n distinct n-th roots of unity.

L/K has abelian Galois group of exponent n.

For example, the Kummer extension $\mathbb{F}_{q^{2(q-1)}}$ could be very interesting since the recent breakthrough suggests fast algorithms in this field and the right hand of the relations would be automatically linear. In this study, $\mathbb{F}_{q^2}[x]/(x^{q-1} - A)$ is used to model the Kummer extension where $A \in \mathbb{F}_{q^2}$ and $x^{q-1} - A$ is irreducible over $\mathbb{F}_{q^2}$.

2.1.4 Lattice

Lattice study is growing popular because of its resistance to quantum computers. In this study, we built the $(q+1)$-dimensional lattice for the conjecture and calculated the determinant related to the matrices.

Definition 2.10. Lattice:

A lattice in $\mathbb{R}^m$ is a subgroup of $\mathbb{R}^m$ and it's isomorphic to $\mathbb{Z}^m$.

In this study, it's defined as

$$\mathcal{L} = \{\sum_{i=1}^{n} x_i \mathbf{b}_i | x_i \in \mathbb{Z}\},$$

where $n \leqslant m$ and $\mathbf{b}_1, \mathbf{b}_2, ...\mathbf{b}_n$ are linearly independent and $\mathbf{b}_i \in \mathbb{R}^n$ for $1 \leqslant i \leqslant n$.

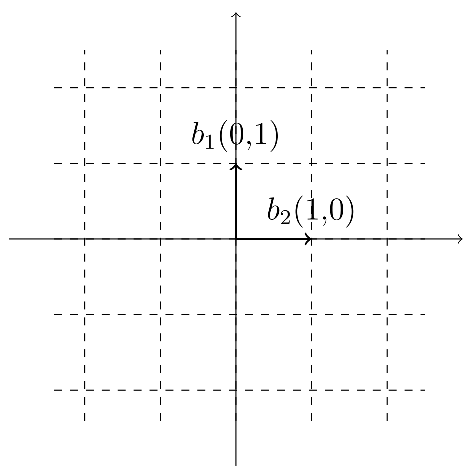

Figure 2.1: Lattice generated by $b_1(0, 1)$ and $b_2(1, 0)$

If m = n in the lattice defined above, the set of vectors $\mathbf{b}_1, \mathbf{b}_2, \cdots, \mathbf{b}_n$ is called

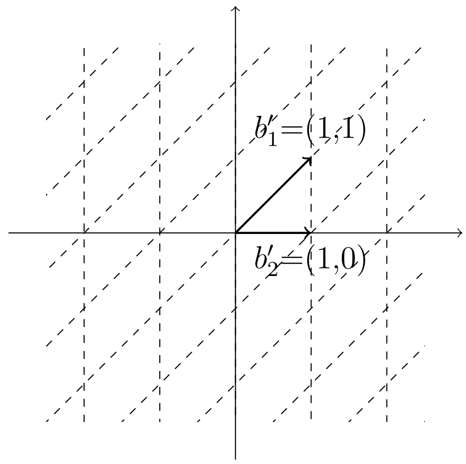

Figure 2.2: Lattice generated by $b'_1(1,1)$ and $b'_2(1,0)$

a lattice basis and it can be represented as

$$B = [\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_n],$$

Definition 2.11. Determinant of a lattice:

For a lattice $\mathcal{L} = \mathcal{L}(B)$, the determinant of the lattice($det(\mathcal{L})$), is the n-dimensional volume of the fundamental parallelepiped $P(B)$.

Now we can go on to the matrices used in the study and explain the transformation chosen on the matrices representing the lattice.

2.1.5 Normal forms

In linear algebra, normal forms are used for matrix transformation in order to generalize and clarify the information of the matrix. Here are the two normal forms used in the study and some transforming methods.

Here we only use row transformation in the study, so the normal forms are transformed by rows unless otherwise specified.

Definition 2.12. Gaussian elimination:

The Gaussian elimination is an operation to change the left lower corner of the matrix into zeros by swapping rows, multiplying rows by non-zero numbers, or adding rows or multiplied rows to another.

Definition 2.13. (Row) Echelon form:

The resulting matrix from the Gaussian elimination is an echelon form.

Definition 2.14. GaussJordan elimination:

If we add the following restriction to Gaussian elimination, it's called Gauss-Jordan elimination: the resulting matrix has leading coefficient of 1 for all rows, and they are the only non-zero numbers in their columns.

Definition 2.15. Reduced (row) echelon form:

The resulting matrix of GaussJordan elimination is called the reduced echelon form. Each matrix only has one unique reduced echelon form.

Note that in this study, row echelon form is used in all calculation if not specified in the following.

Definition 2.16. Hermite normal form:

The Hermite normal form is basically similar with reduced echelon form but it's over integers $\mathbb{Z}$. All transformations should be done over $\mathbb{Z}$. Note that only swapping or adding rows, or addingrows with nonzero integer multiplication are allowed in the transformation since solely multiplying rows with integers may change the determinant.

Definition 2.17. Smith normal form:

The Smith normal form is defined for matrices with entries in a principal ideal domain (such as $\mathbb{Z}$) and calculated by by multiplying the left and right invertible square matrices.

Here's an example of the transformation between normal forms (row transformation).

Say we have matrix A:

$$A = \begin{pmatrix} 1 & 2 & 3 & 9 \\ 2 & 5 & 6 & 7 \\ 3 & 4 & 5 & 6 \\ 4 & 7 & 8 & 9 \end{pmatrix}$$

Here when we transform the matrix to reduced row echelon form (E). It's calculated in $\mathbb{Q}$.

$$E = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Here we transform the matrix to Hermite normal form (H). It's calculated in $\mathbb{Z}$. Note that it has the same absolute determinant value with A.

$$H = \begin{pmatrix} 1 & 0 & 3 & 1 \\ 0 & 1 & 0 & 4 \\ 0 & 0 & 4 & 3 \\ 0 & 0 & 0 & 5 \end{pmatrix}$$

Now we transform it into Smith normal form(S). Here we can see

S=$S_{left} * A * S_{right}$.

$$S = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 20 \end{pmatrix}, S_{left} = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & -3 & -2 \\ -4 & -1 & 14 & 11 \end{pmatrix}, S_{right} = \begin{pmatrix} -2 & -1 & 3 & 15 \\ -1 & -4 & 5 & 24 \\ 2 & 4 & -7 & -33 \\ 0 & 0 & 1 & 4 \end{pmatrix}$$

The previous study was done mainly in Smith normal form, but we changed the verification calculation into Hermite normal form in this study. We'll explain the reason of the change in the later chapters.

2.2 Calculation

2.2.1 Matrix transformation

The first step is to prepare for parameters used in the calculation.

We start from the q used for the verification and calculate the N with the following method:

1. Calculate $q^{2(q-1)} - 1$.

2. Remove prime factors from $q^{2(q-1)} - 1$ until the smallest factor is larger than q^2.

Then we build the field with N from above. Now we can form the matrices need in the calculation:

$$G(x^k) = x^k \sum_{\alpha \in \mathbb{F}_q} x^{\log_g (g+\alpha)}$$

$$T(x^k) = x^k x^{\log_g \frac{Ag^{k(q-1)}-1}{g^q - g}}$$

The next step is to get $M = GT$ and relate it to the lattice. Here we define $\tilde{M}$ with a transformation from $\mathbb{F}_{q^2}$ to $\mathbb{F}_q$, and M_1 as in the matrix

$$M = U^{-1} \begin{pmatrix} M_0 & & & \\ & M_1 & & \\ & & \ddots & \\ & & & M_{q-2} \end{pmatrix} U$$

If we denote L as the map from an integer matrix to the lattice and construct the following:

$$\mathcal{L}_1 = L(\tilde{M}_1 - I) + N\mathbb{Z}^{q+1}.$$

Then We have $N|\det(\mathcal{L}_1)|N^{q+1}$.

The hypothesis need to verify $\det(\mathcal{L}_1) = N$ with q's as large as possible.

2.2.2 Example q = 7

Here's an example when q = 7.

First, we calculate N.

$$q^{2(q-1)} - 1 = 2^5 * 3^2 * 5^2 * 13 * 19 * 43 * 181$$

After removing factors no larger than $q^2 = 49$, we got N = 181.

Then we find the polynomial G.

$$G_{poly} = x^{38} + x^{36} + x^{31} + x^{11} + x^5 + x^2 + x$$

And we get $\tilde{G}$ from G with modulus of $x^{q+1} - q^2$.

$$\tilde{G}_{poly} = 180x^7 + x^5 + 132x^4 + 49x^3 + x^2 + x$$

α	0	1	2	3	4	5	6
DL	1	5	38	36	2	11	31

Table 2.2: Discrete logarithm of $g + \alpha$ in G

Next step is to build the matrix $\tilde{G}$ and $\tilde{T}$.

$$\tilde{G} = \begin{pmatrix} 0 & 1 & 1 & 49 & 132 & 1 & 132 & 180 \\ 132 & 0 & 1 & 1 & 49 & 132 & 1 & 132 \\ 133 & 132 & 0 & 1 & 1 & 49 & 132 & 1 \\ 49 & 133 & 132 & 0 & 1 & 1 & 49 & 132 \\ 133 & 49 & 133 & 132 & 0 & 1 & 1 & 49 \\ 48 & 133 & 49 & 133 & 132 & 0 & 1 & 1 \\ 49 & 48 & 133 & 49 & 133 & 132 & 0 & 1 \\ 49 & 49 & 48 & 133 & 49 & 133 & 132 & 0 \end{pmatrix}$$

$$\tilde{T} = \begin{pmatrix} 0 & 0 & 180 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 48 & 0 \\ 180 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 180 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 132 \\ 0 & 132 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 48 & 0 & 0 \end{pmatrix}$$

Now we calculate M and append the diagonal matrix of N.

$$\tilde{M}_1 = \begin{pmatrix} 179 & 48 & 0 & 132 & 132 & 133 & 48 & 132 \\ 180 & 131 & 49 & 180 & 49 & 1 & 0 & 48 \\ 0 & 48 & 47 & 180 & 1 & 48 & 1 & 133 \\ 49 & 133 & 132 & 180 & 1 & 1 & 49 & 132 \\ 48 & 132 & 48 & 49 & 180 & 180 & 180 & 132 \\ 132 & 132 & 133 & 48 & 132 & 47 & 49 & 0 \\ 48 & 0 & 132 & 132 & 133 & 48 & 131 & 48 \\ 133 & 48 & 132 & 48 & 49 & 0 & 180 & 179 \\ 181 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 181 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 181 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 181 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 181 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 181 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 181 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 181 \end{pmatrix}$$

At last we calculate the Hermite normal form of the matrix above. The resulting matrix will be as follow:

$$HNF = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 155 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 141 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 59 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 170 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 77 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 35 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 46 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 181 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

2.2.3 Verification

We follow the steps above to verify whether $\det(\mathcal{L}_1) = N$. In the previous study, all q's such that $\log_2(q^{2(q-1)}) \leq 5000$, which means $q \leq 307$, was verified. Here we increased the q's until $\log_2(q^{2(q-1)}) \leq 10000$, which means $q \leq 613$, and changed the algorithm of verification.

As for the output, we expect to get the diagonal of the resulting matrix as $[1, 1, 1, ...N]$, and only printed the diagonal instead of the full matrix which should be as follow:

$$HNF = \begin{pmatrix} 1 & & & x_1 \\ & 1 & & x_2 \\ & & \ddots & \\ & & & N \end{pmatrix}$$

2.2.4 Modification

The first change to the algorithm is using Hermite normal form instead of Smith normal form during the calculation. The reason is that we tested different combinations of normal form and transformation, and then found that the Smith normal form is relatively stable under transformation. In other words it's not suitable for the speed up. For the Hermite normal form, we found different speed when we attach the diagonal matrix of N to the $M = GT$ in different directions. Then we picked the faster one as the new algorithm.

The second change is only made for the outliers which doesn't work with the new algorithm. Due to limited time and space, some q's exceeded our computing resources when we apply the new algorithm. So we divided q's into several cases and treated the outliers with different algorithm. According to known factorization of specific structures, we divided the N's into smaller integers, and then applied the old algorithm on those partitions. The other steps are exactly the same.

2.2.5 Evaluation

The main concern of the study is the time efficiency of the verification, so we set a series of reading of CPU time during the calculation. The threshold of the whole process was the calculation of the Hermite normal form. To compare the efficiency of this part, we record the CPU time separately for different calculations.

We also considered the space consumption during the verification. Since some of the q's took way too much memory and caused overflow, we compare the memory usage in different cases.

Chapter 3

Results

3.1 Correctness

All q's such that $311 \leq q \leq 613$ were verified in addition to previous q's such that $q \leq 307$. Although some of the q's doesn't fit in the new algorithm, they were all verified by the second modification. Also, the outliers could be divided into three cases as shown in Table 3.1. According to $q-1$'s factors, we can see that they contain small factors such as 2 or 4, and another large prime factor. But not all the q's with these structures are outliers. For example, 317 and 383 has similar structures but they can go through the new algorithm.

For these outliers of the new algorithm, we partitioned the N into several factors with known factorization. Then we ran the smallest partition with old algorithm and other partitions with new algorithm. All the partitions returned

$4 \mid q-1$	389, 557
$2 \mid q-1$	467, 479, 503, 587
others	311, 313,... 613

Table 3.1: Grouped prime numbers $311 \leq q \leq 613$

correct relation and the product of the determinants is exact N.

We are going to show a sample of this situation, q = 389, in the third section in this chapter.

3.2 Efficiency

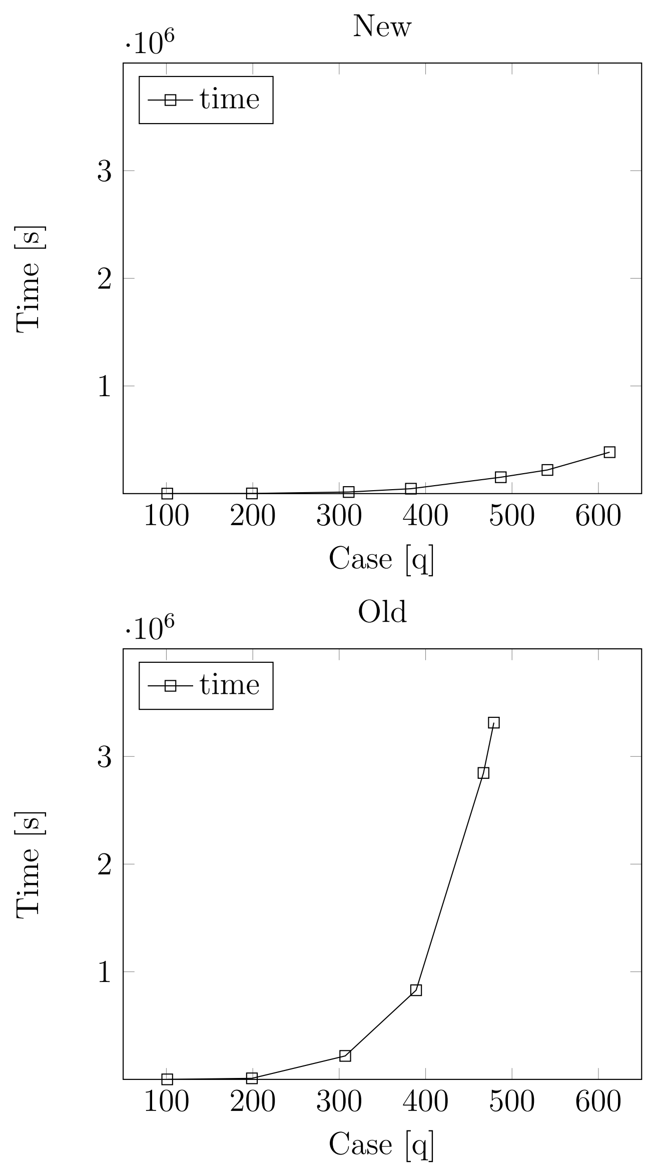

Figure 3.1: Selected HNF time of new and old algorithms

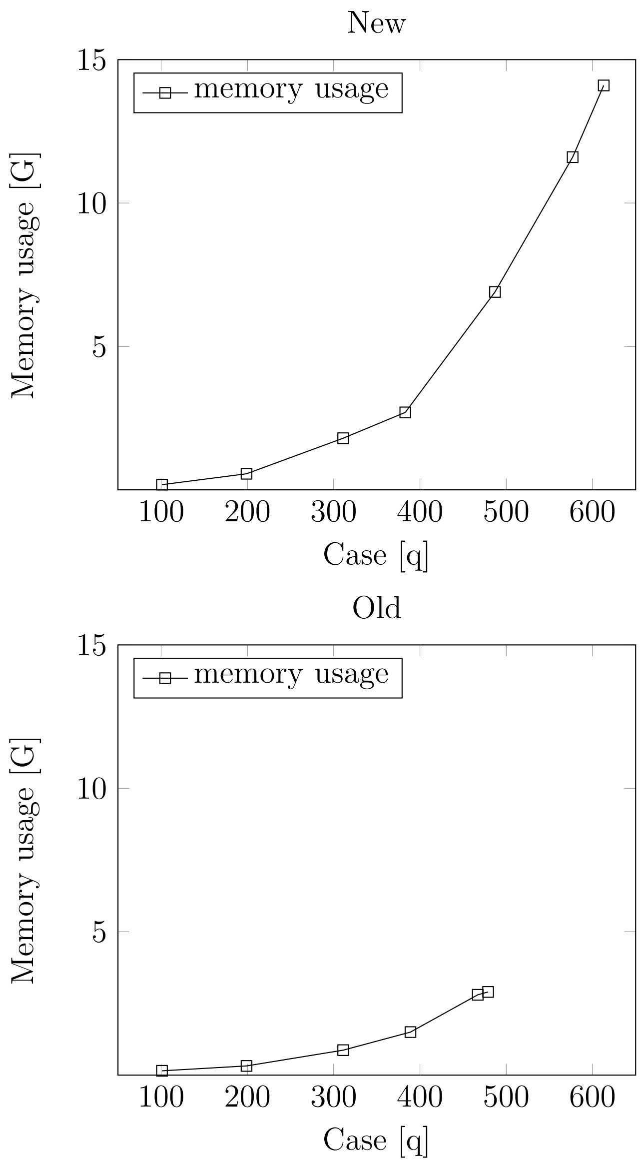

Figure 3.2: Selected memory usage of new and old algorithms

As is shown in Table 3.2, the largest part of time used in calculation is to calculate the HNF. Also, this study doesn't change the calculation method of getting G,T and M. So the efficiency evaluation in this section was referring to HNF time.

The general speed up of the algorithm is more than 1000% and vary with N's in

Time to get	Time s
N	0.644
G, T	873.816
M	40727.372
HNF	384903.052

Table 3.2: Calculation time of different steps in case q = 613

the new algorithm. The threshold of the outliers are using the old algorithm to calculate the smallest partition, but we still found more than 1000% speed-up. With this improvement, we took similar time computing the case where $q = 307$ with old algorithm and $q = 613$ with new algorithm.

3.3 q = 389

The first outlier of the new algorithm is q = 389.

Since $4 \mid q - 1$, we partitioned N into N_1 to N_4 by dividing q as follow: The original method calculates N from $389^{(2*389-2)} - 1$, which could be factored as $389^{388} + 1$, $389^{194} + 1$, $389^{97} + 1$, and $389^{97} - 1$.

Each partition of N is verified individually but we choose different algorithm for each partition. The first three partition are eligible for the new algorithm while the last partition only can be done by the old algorithm. However, due to the decrease of size N, the last part finished much faster than using the old algorithm on original N. The space consumption almost doubled using the partitioning method even without parallel computing. But it was still affordable at this data size.

	HNF time old	HNF time new
$389^{388}+1$	around 270000	around 10000
$389^{194}+1$	around 90000	around 3000
$389^{97}+1$	around 30000	around 1000
$389^{97}-1$	around 30000	unfinished

Table 3.3: Partitions and calculation time of q = 389

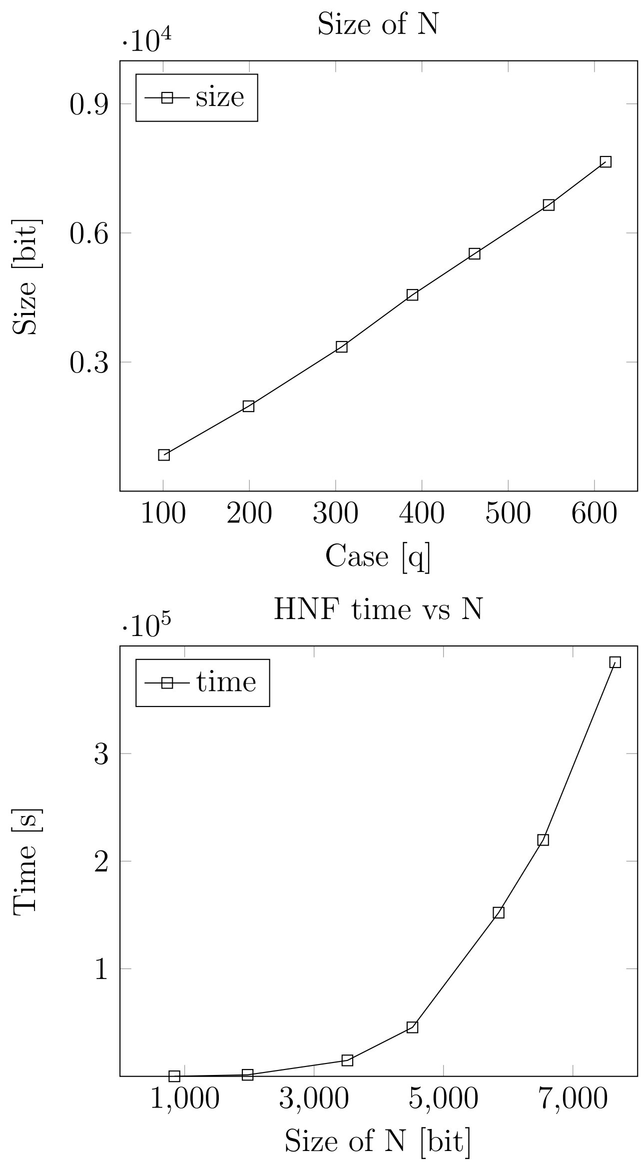

Figure 3.3: Size of N and HNF time vs Size of N

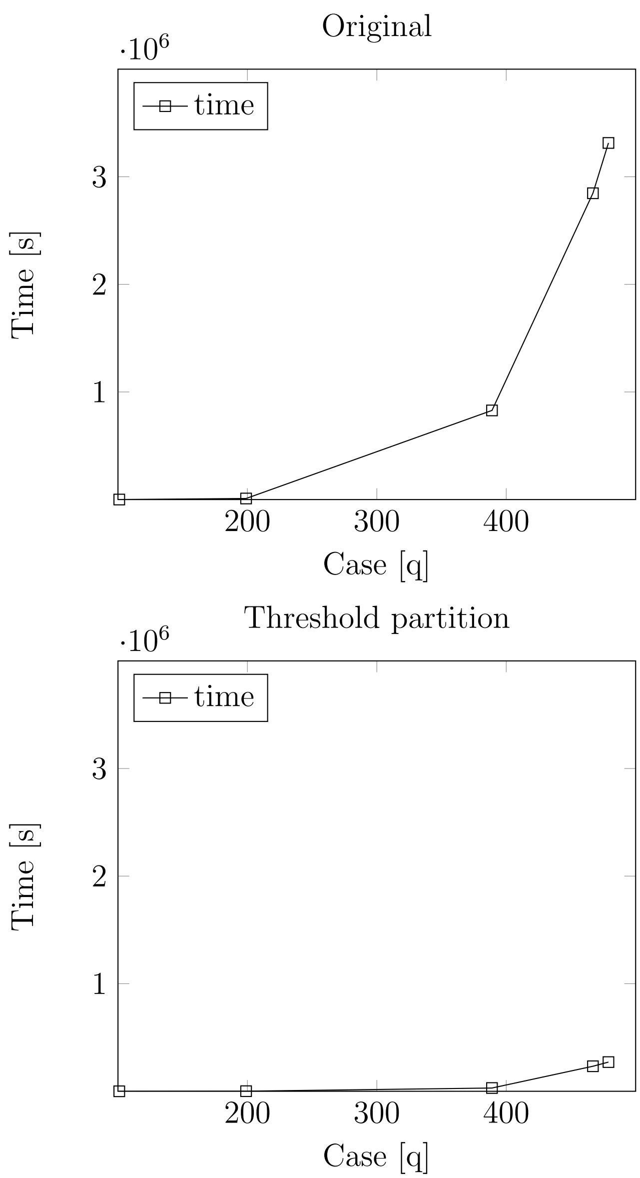

Figure 3.4: HNF time of threshold partitions and original

Chapter 4

Discussion

4.1 Matrices structure

During the calculation, the matrices M could be built in different ways. The previous algorithm appended GT after diagonal matrix with N and the new algorithm choose opposite direction. The difference between the time usage was great: the new algorithm is much faster for eligible q's. However the space usage increased a bit.

$$M_1 = \begin{pmatrix} a_1 & a_2 & a_3 \\ b_1 & b_2 & b_3 \\ c_1 & c_2 & c_3 \\ N & & \\ & N & \\ & & N \end{pmatrix}$$

$$M_2 = \begin{pmatrix} N & & \\ & N & \\ & & N \\ a_1 & a_2 & a_3 \\ b_1 & b_2 & b_3 \\ c_1 & c_2 & c_3 \end{pmatrix}$$

The possible reason for these results could be the default calculation process of the Hermite Normal Form in Sage. Elements in GT might be somehow sparse for eligible q's, which could end up with smaller space usage at the beginning and sharp increase by the end. Space usage is unstable with each q's structure. Smaller elements during the most calculation period reduced the time, but it also varies with q'.

The non-eligible q's could have much more dense structures during the calculation, So the new algorithm with unstable space usage might cause overflow on our computer. But the old algorithm could solve those cases even without partitioning, using longer time.

q	factorization of q-1
311	2 * 5 * 31
317	2 * 2 * 79
383	2 * 191
389	2 * 2 * 97
467	2 * 233
479	2 * 239
487	2 * 3 * 3 * 3 * 3 * 3
503	2 * 251
557	2 * 2 * 139
587	2 * 293
613	2 * 2 * 3 * 3 * 17

Table 4.1: q's and factor of $q-1$

4.2 Grouping q's

The outliers of the new algorithm could be grouped by their factors, since all cases was found to contain only 2's or 4's with another large prime factor of $q-1$'. The cause of the overflow during the matrix solving might be related to those large factors but that is not necessarily related.

If we could try different sequences during the calculation of solving the Hermite Normal Form in the future, we might find the relation to the structures and q's. For example, if printing the matrix during the transformation is possible, we may find out the size of elements and distribution of sizes.

Chapter 5

Conclusion

In summary, this study is an expansion of cases in the previous paper and improvement of algorithm from the old one. The speed-up was significant and space usage was reasonable. It's interesting to find the different cases of q's during the calculation.

In the future, we can look into the cause of grouping with more theoretical mathematics. Also, new way of verifying the determinant would be great if faster calculation or parallel computing could be introduced to the study.

References

[1] Adleman, L.: A subexponential algorithm for the discrete logarithm problem with applications to cryptography. In: Foundations of Computer Science, 1979., 20th Annual Symposium on. pp. 5560. IEEE (1979)

[2] Adleman, L.: The function field sieve. In: Algorithmic number theory ANTS I. Lecture Notes in Comput. Sci., vol. 877, pp. 108121. Springer (1994)

[3] Barbulescu, R., Gaudry, P., Joux, A., Thomé , E.: A heuristic quasi-polynomial algorithm for discrete logarithm in finite fields of small characteristic. In: Advances in Cryptology - EUROCRYPT 2014. LNCS, vol. 8441, pp. 1-16. Springer (2014)

[4] Cheng, Q., Wan, D., Zhuang, J.: Traps to the BGJT-algorithm for discrete logarithms. LMS Journal of Computation and Mathematics 17, 218-229 (2014)

[5] Diffie, W., Hellman, M.: New directions in cryptography. IEEE Transactions on Information Theory 22(6), 644654 (1976)

[6] Gäloğlu, F., Granger R., McGuire G., Zumbrägel J.: On the function field

sieve and the impact of higher splitting probabilities. In: Advances in Cryptology - CRYPTO 2013. LNCS, vol. 8043, pp. 109-128. Springer (2013)

[7] Granger, R., Kleinjung, T., Zumbrägel, J.: On the powers of 2. Cryptology ePrint Archive, Report 2014/300 (2014)

[8] Granger, R., Kleinjung, T., Zumbrägel, J.: On the discrete logarithm problem in finite fields of fixed characteristic. Cryptology ePrint Archive, Report 2015/685 (2015)

[9] Joux, A., Lercier, R.: The function field sieve in the medium prime case. In: Advances in Cryptology EUROCRYPT 2006. Lecture Notes in Comput. Sci., vol. 4005, pp. 254270. Springer (2006)

[10] Joux, A., Lercier, R., Smart, N., Vercauteren, F.: The number field sieve in the medium prime case. In: Advances in Cryptology CRYPTO 2006, Lecture Notes in Comput. Sci., vol. 4117, pp. 326344. Springer (2006)

[11] Joux, A.: Faster index calculus for the medium prime case. Application to 1175-bit and 1425-bit finite fields. In: Advances in Cryptology EUROCRYPT 2013, Lecture Notes in Comput. Sci., vol. 7881, pp. 177193. Springer (2013)

[12] Joux, A.: A new index calculus algorithm with complexity $L(1/4 + o(1))$ in very small characteristic. Cryptology ePrint Archive, Report 2013/095 (2013)

[13] Pohlig, S., Hellman, M.: An improved algorithm for computing logarithms

over $GF(p)$ and its cryptographic significance. IEEE Transactions on Information Theory 24(1), 106-110 (1978)

[14] Xiao, D., Zhuang, J. and Cheng, Q.: Factor base discrete logarithms in Kummer Extensions, Cryptology ePrint Archive: Report 2015/859 (2015)

Appendix

Sage code

q=31 code

```
#########################################################################
# This is the code when q = 31.
# Just change the initialization for other q's.
#########################################################################
# initialize q and calculate N

q=31
Adlog = -1
print "q=",q
t=cputime()

N=q^(2*q-2)-1
for i in range(2,q^2):
        while N%i==0:
                N=N/i
```

```
print "N=", N #calculate N = large prime factors of q^(2*q-2)-1
# print "=", factor(N)

################################################################################
# initialize RING AND FIELD

II=Integers(N)
R.<x>=Integers(N)[]
F2=GF(q^2,'g')
F2_gen=F2.multiplicative_generator()
F1_gen=F2_gen^(q+1)
A=F2_gen^Adlog
# A can be changed here such as A=F2_gen
# Adlog can be changed here such as Adlog=1

################################################################################
# find G

t=cputime()
k=0
for i in (F2):
        if i^q==i:
                l=F2_gen+i
                for j in range (q^2):
```

```
                    if F2_gen^j==1:
                            k=k+x^j
                            break
G=k
print "G=",G

klog = []
# klog stores the logarithm of k
for i in range(q+1):
        k=(A*F2_gen^(i*(q-1))-1)/(F2_gen^q-F2_gen)
        for j in range (q^2-1):
                if F2_gen^j==k:
                        klog.append(j)
                        break



################################################################################
# calculate HNF

def test(q,b,Sbase):
        t=cputime()
        #calculate Gx,Tx
        S = R.quotient(x^(q+1)-Sbase^b, 'a')
        XK=[]
        GK=matrix(R,q+1,q+1)
```

```
YK=[]
TK=matrix(R,q+1,q+1)

for i in range (q+1):
        XK.append(S(G*x^i))
        YK.append(S(x^(i+klog[i])))

for i in range(q+1):
        k=[]
        m=0
        for j in XK[i]:
                k.append(j)
                m+=1
        if m!=q:
                for j in range(m,q):
                        k.append(0)
        GK.set_row(i,k)

for i in range(q+1):
k=[]
m=0
for j in YK[i]:
        k.append(j)
m+=1
if m!=q:
```

```
        for j in range(m,q):
            k.append(0)
TK.set_row(i,k)

GKS=matrix(II,q+1,q+1)
TKS=matrix(II,q+1,q+1)
I = matrix.identity(q+1)

for i in range (q+1):
        GKS[i]=GK[i]
        TKS[i]=TK[i]

M=GKS*TKS
MN= matrix(ZZ,2*q+2,q+1)
for i in range(q+1):
        MN[q+1+i,i]=N
        MN[i]=(M-I)[i]

t=cputime()
M0=MN.hermite_form()
N0=vector(ZZ,q+1)
print 'HNF_time=',cputime()-t
for i in range(q+1):
        N0[i]=M0[i][i]
print N0
```

```
##########################################################################
############################ main ##################################
##########################################################################

print 'begin test'
test(q,1,q^2)
```

q=31 output

begin test

q= 31

N= 142626898851388442193955125501359865760996717321109924032112373711122891

N time= 0.0

G,T time= 0.024

M time= 0.176

HNF time= 0.064

(1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,

142626898851388442193955125501359865760996717321109924032112373711122891)

q	factor of (q-1)	bits of N
2	1	0
3	2	0
5	2^2	5
7	$2 * 3$	5
11	$2 * 5$	36
13	$2^2 * 3$	40
17	2^4	71
19	$2 * 3^2$	58
23	$2 * 11$	123
29	$2^2 * 7$	172
31	$2 * 3 * 5$	162
37	$2^2 * 3^2$	195
41	$2^3 * 5$	248
43	$2 * 3 * 7$	276
47	$2 * 23$	336
53	$2^2 * 13$	386
59	$2 * 29$	446
61	$2^2 * 3 * 5$	418
67	$2 * 3 * 11$	497
71	$2 * 5 * 7$	521
73	$2^3 * 3^2$	524
79	$2 * 3 * 13$	603
83	$2 * 41$	698
89	$2^3 * 11$	746

97	$2^5 * 3$	777
101	$2^2 * 5^2$	841
103	$2 * 3 * 17$	874
107	$2 * 53$	955
109	$2^2 * 3^3$	901
113	$2^4 * 7$	988
127	$2 * 3^2 * 7$	1075
131	$2 * 5 * 13$	1223
137	$2^3 * 17$	1287
139	$2 * 3 * 23$	1284
149	$2^2 * 37$	1433
151	$2 * 3 * 5^2$	1415
157	$2^2 * 3 * 13$	1467
163	$2 * 3^4$	1553
167	$2 * 83$	1656
173	$2^2 * 43$	1731
179	$2 * 89$	1820
181	$2^2 * 3^2 * 5$	1707
191	$2 * 5 * 19$	1929
193	$2^6 * 3$	1935
197	$2^2 * 7^2$	2030
199	$2 * 3^2 * 11$	1973
211	$2 * 3 * 5 * 7$	2092
223	$2 * 3 * 37$	2320
227	$2 * 113$	2425

229	$2^2 * 3 * 19$	2366
233	$2^3 * 29$	2472
239	$2 * 7 * 17$	2564
241	$2^4 * 3 * 5$	2462
251	$2 * 5^3$	2716
257	2^8	2727
263	$2 * 131$	2892
269	$2^2 * 67$	2962
271	$2 * 3^3 * 5$	2802
277	$2^2 * 3 * 23$	2984
281	$2^3 * 5 * 7$	3036
283	$2 * 3 * 47$	3121
293	$2^2 * 73$	3249
307	$2 * 3^2 * 17$	3354
311	$2 * 5 * 31$	3515
313	$2^3 * 3 * 13$	3409
317	$2^2 * 79$	3602
331	$2 * 3 * 5 * 11$	3611
337	$2^4 * 3 * 7$	3758
347	$2 * 173$	4024
349	$2^2 * 3 * 29$	3931
353	$2^5 * 11$	4068
359	$2 * 179$	4184
367	$2 * 3 * 61$	4182
373	$2^2 * 3 * 31$	4311

379	$2*3^3*7$	4261
383	$2*191$	4520
389	2^2*97	4561
397	2^2*3^2*11	4565
401	2^4*5^2	4651
409	2^3*3*17	4765
419	$2*11*19$	4990
421	$2^2*3*5*7$	4782
431	$2*5*43$	5126
433	2^4*3^3	5065
439	$2*3*73$	5256
443	$2*13*17$	5303
449	2^6*7	5333
457	2^3*3*19	5477
461	2^2*5*23	5517
463	$2*3*7*11$	5505
467	$2*233$	5676
479	$2*239$	5875
487	$2*3^5$	5852
491	$2*5*7^2$	5909
499	$2*3*83$	6081
503	$2*251$	6234
509	2^2*127	6291
521	2^3*5*13	6385
523	$2*3^2*29$	6420

541	$2^2 * 3^3 * 5$	6540
547	$2 * 3 * 7 * 13$	6649
557	$2^2 * 139$	7001
563	$2 * 281$	7096
569	$2^3 * 71$	7124
571	$2 * 3 * 5 * 19$	7100
577	$2^6 * 3^2$	7134
587	$2 * 293$	7435
593	$2^4 * 37$	7450
599	$2 * 13 * 23$	7577
601	$2^3 * 3 * 5^2$	7438
607	$2 * 3 * 101$	7702
613	$2^2 * 3^2 * 17$	7653

Table 5.1: Factor of (q-1)'s and bits of N's

Selected q's and determinants

q=311

N= 12220411457481341798648123502938873888169864938486419458061779684519

633773718325807382423254560854121891872381196014855253725161140044501326

840291319765434176918675896823872774734003455372466148895851643435785224

626009714904235854356647470448102513132512659958395237446359110526454795

959177097885577601999780360566406370302177987327533213231942330332426897

893786185740955907652661699531167600216318185100691978130852121936769490

958999056294346567845100249788412392103908381648864114235372837619101634

856648623401207731452963462537018184481684290640409306322381702308295939

247119348497288093733740380345793466966323367569332795813194076393184394

585090377448437440179051089603043396280934495150890826106447800609207045

866225221768826271313973947765248326560131041325812591362336610694982552

719356874239881180561700880555873289430567531921717380049998010517230720

909956655060247530481467964474502154179720952919798058994446256251276835

469241111077252199650513993011290200921954199870571983961084496324555966

103674275690918673335780710423372758684182796492718777256080697208591157

882465160281146160203042977718927562171798903089885996738245775406342229

924627582132962202562863260335072480256211728775630884357861164063291314

443654448871159002687142735391915039896923028993710508489164161294235569

656534675607586762122068181006579255587288309193592733615807492684765354

786331697328026979130749878112737053653471058662859821791094258932270768

354545403030357372180059883131497577986390852673734610006458011514436850

188168696647677

q=317

N= 4021785173154085322655574753137091289006429330886571561660094764082025096818735665065568606226286154152504064112564861937901952925617407568286926568779646055184746606049316606024011899371771665022800676580767034719619583954968280714396440590336994834530118991027936331801150427362939693518242570076363136840365770932966144304372598975781945110481069726261546285340849472299732286650610727616415294009235102545870596043211915424586960233937332337958986039804601053649038967488623456945225989089033401275419329294322513933100734565652842059670311848945669554893129720949447954836951424304175908471967070682772300874480361890979938803146221788105309105727259177491592506197903103055791894056811937490366344252647565457597564766932511281000136846597587810392066535818039165649526995642433608546500096708317548001282282350497793375323900631878335615233643196379667536096467921713929336707535209851385721085526112592135089963195453144246120147416810255297907994560299604353869770277890910576475371951544234112263700193716596807098474711518620524142746755488794787322941516546432932426171929361587888626006289671937521996598178773391476073536684327075334892817166445708005669892476201578305359678934320690158091873793833445606760277427677070493149308305030488142718346015582953457619985800358337241346580760564307466215915620330738784938802568324196853364619202331253493560712153255237113801078736976763236542229449221244087307270718233645259175297320908538008271399073602614627495850660009294692396963268376106946856717947180752801376456672000540771576891420276815577

q=383

N= 145807697907860485797130733205126107593584846052460619489805157752077495221106932588520972734241918732485798853797085970773190101350383098111486896367912933024794946367066801905327711671486423456937272799971540386901101697195533486335856133921155426166323042195462985673651542303182186013840521093943327105530670933517251498028541937966545458857514617362652968672918773886784117748623258822128764993971325593170308095460301556064262854440090893435962828515919017600496019082806907459752774772149654173207731742961935266837232027992561475878997077497364305758960420188022307484695686408178417684708440545982139461121483444966392983266047927438631017956435460617872268689048536223008200601915584845567121435040154255004894352416019083701156336653868479758018963822040230841067925258491403578140056398377200259632163622949623530237558514032980340190971794920703369202249165169714902231036847494140896438546785548350557620139234129697851079309910579637499856308322874165048878317416980329070589555477028424794660770832243516235856253757395848700158548799774104575282732361811885763245707493651012635463603167780987920047275869960119345415739631674289157572504881069378366169038207192195564786884517931414904771270064834787745940295070778488166086343040816145261828784471225312158037216530407121009255667745904658762339035510401085623511888982417436321783954442230138142191402526854409042152090507836707321554478029151783138735123674836002824091755336065669270096841930169867431073229595480504906522894081456268215051621596295121066160620101958319152599732645858807987026779736321720489864918934658890361142498721160269105481169759452959400766323080817853856178349184142891298435315142292331986809901442857189099219517314949338914405526340356801855365944445252481277250224143631547123162141573825856671

013168832075045445040242784141424551583486404827186671835477961314720805
667647392584241256388983691283678436605160723687265941636927616111371892
653376633932050977

q=389

N= 7195331750747484152085419439650343326167001327774918316175974049277
722938228436284822115935720940855808723841592194554852591261106514552587
998792697344006909328597127958485586198824809760493619528438029382516955
802437372015885868652676060704273108648005446756567866454037217153666707
116107955813259897581789965548166873227089745793038056876086093487364660
104165354375638730780144771940820461379144045407410633806525882471616992
313107313458518056156155372469123009498454107816512700653764419802046775
795056390593183426232868729935687347132474597447270299842803551483280382
464345737592680500610492028940899252744848503850147888866006672309743009
707608056808465671630041936151575721737832608056167755557287448599406240
664128644940787322334450386697752135044664759178397183268772058995455906
530214874403118896365825911606545712252543349530273195645766743047004882
772794343150019268224755541621832841084927152609167782631248686519979077
261704778279434581285089309982530290516450254651948711338697696523561128
237131690866523967123809890516412122433312579639731316482517399044222371
648027429394491857914758789554722411217192580708530710563303630609337512
375736374992413197099835298039698176561667502460196281220809685898103151
801536460824061171709409915407076163699749989106951416033064052305415369
172208716680222928232108669911514529435819037003548630822766056743609717
871400339970257506326522184356565974073087427738029035514597912641413281
864721228361243940548875106883409490452585792883551924553356666885340993
409525115851784559602381695501650437380918742644155483811707202620992640
671919692119264818017617725574319133657575175571011358782147102699466133
976139878846733780028611934562593296650810158727671483594135852975270136
343283327218261498217240273563540924577404997311033419523302409280733431

875376399609828318193709239918912628149158776296863968437043483848149113
291753871808384517843168494031720350845780796001506232612214295949526384
219376329330331111096335777802664057

q=467

N= 8193682819735904907936462652762652957783801140282061764680311398642
355238292209136200793745698408979142685271806667880695439656644767365658
962804500507328924591277012527354972867083755868359662342631293941095260
296836469017267512710855619435662717780121618969799191783589121739510344
276659959452330107141588370842890533418683574136420161763154188864032344
051335032890602342690990331545512878722845868516478220655672811984728099
829584400348024077781024883841078871472427192910334006397072591593425575
001582062495270466693923330813787033936932085463161319738777157723719205
663510587261989045516277399792258676669025902061468414338936004777221651
120128177284169726309676767243230700942651892885739372435031266603405194
926736374504661605073635356729678170946671059228219580414102477943739962
010943602405830702183270123588613295844379470805381224397382897311239774
291035219450733243985454552294613261579819999222768501542040828030342324
749500293990959881643288222147454465739982396633029518755264422499169413
800948870145113541709861228904602480297159321730407550425161663814137762
815328504113649745806201609675807776334603770728185435662442974547545292
124484627139481264440306306276174236846371370615288136390337733795900869
799059527013550349978911522961593782262111579795770566157261698765468287
287036428769421593687671844613576392027427659500766509619007109126319228
663188313182821761621096295894335877149917549224508427298360298958806603
741672739378573845250923926503996897651348539802615982069497009131645174
845900515957045945369600419581505046736732678816987857244113512133573495
658735922450650020693185605825145435753481586659509230006830286160027557
753020447567816102711727703321437248277987102699335416867799947690827644
062178290893074122964705126667835005148297475296259944293090593568576122

5629558808205217950847525491504317391051703569813876231387515800743315030493815629343439191884039201706327712011987228295395235819725599070511744265612781790575938199146494834368418874975473661950994875221151762145858853345085590595857580673012926252570320335768017548905129252393215690925593491649839246055867408289640641961552478227647863109357183322194535275642153714290717873417807719280567657903124427847018396645351227162210820649141250124627238609705165559365557672068090556545873648551696644627732118320197614267955377577955441282218071241250736726669821938418201922693091279412607442092475063704700434351443214379177200428908362203557554290621317375594
87

q=479

N= 9618151808687518422857322086615970569614277462595145529845978343454
4486188026930938061743017030419418593485359328699071531010830714110108
0345606363066590053633174140919515192088558243050537910361417168932577848
9635851915467509624636121252747504888668412598846424286996049592018557323
913094689734727755921975094524127950642823880986148497506016970894935903
6485659438676124794055888452505537520064833453800776743035574194321275582
0873342146397137934453304944080623143952670138563549700616936443180578
342918083979210813952453042362800044264328264945159129737220885021342713
346191068726429398553152794681073326673924869399544914873586949715392110
531438902712271275177024677139897963834184485523828965877157232401583590
724538669085315631948916000464357206945552918439460098610548330239378114
122604890421876396849925315449588394081477882496044776662881517998435562
922112467614034116166021538937437658221308477049569621272269637247837934
673746445862102134975710220067727948012646733442558158141041207117963957
330144687116516078841354392849482955694383836421986960742580665512361245
839678127488561770778227718715382062105302566161525401602879855016763348
228945728415309119317294944594581933515923022558243574130162333173574665
114596788216409661778201806711182609580526256789639969059069699814702335
464985047554549072175794921081900428053940948883515585651383933509407464
535316790679550728411775648496877738594644848217099818071547388089609112
011431713647420103559581220011329723647072895374636037206013179803863007
429044971341608413273142857973517743107457111227744181514703731489937526
850686101396429111172298050675100600910677690088021950601559433998023281
835332134627833275107542602172947468095430692109106457093076533472350885
507022280097347231142085342847490879465366555284009919595653337484689307

349840278259161207402657247053492207755420957935013937050896483453379380828096760322973712836650801432534638598289150614333662247482255759866235559272488800042580715888827058657221135097453961914795850259502887785544339667660901930718813936816443652123408869864159822026949526186969957601161164181568852772501922130983139374998154568845631490682925558745817465486295506575815670710489889048343025182906053167837053183750918153073307447086110050331595856422010655107164873054545413592243349550477135845749293027979970580525126115682454706374207919453453022726928557144829026117662443315761792289181179796802022735421053346183377870762458631998915630317689186838149672159569227374084851369323342786149343578458757328878282020478616715216795538097067891

q=487

N= 107827207669750265089506440622206504438721635255378182636161342737
546361849057898469875599327816975774441578412233114599521742614494846887
740703080811148454047060485212052898351429291302607383413701329004937472
110130810594384628498903013384299898677662097586203992584420300123479688
785684393540422293410360973720913164496834267238412112247567596545168837
737144534589039088347178412795877575787183707026931850844179567530895640
048616446605895392791202991591300212536936775209605943209829037959038448
644375408328710718300289456458572242049271728769524363126273853715500193
992389397888943821783652200285128430559853028782821408152591885930253959
343363572383873083165126137898284523398720102296715451247068157020833701
559621319003673098042630112932862628501950564632735466206216603138321249
217171339976696682869969285858086091226851424749306290948023985863737595
479286567482749091395852685584018895956480265983348479431205342847195247
664202184545509444073196750969881471993537723451511123356835172969794677
005895270609159438063149957132594233024220238098666324875715524694568431
877984803514244863991450442722463359917319219876714736136981326149224271
917119983677231104646058922538799501312301306624431782348213725731474980
072268037174861637258178428492282775604518408282182883338962744713428765
351357038539796626115406098868926182873473137617788072302142999686731779
004538770966351787580252505338707237518234022336687560434124912382388971
097424646432301275534386052715373826920904552232580275869239823036727431
833531669999335834386165607257759778153553173938764606955523815952618365
383445341291463783562970897109625720657070919217228163652664563258546056
946645712882883254672309664786508152237835087204624488511667622924336500
614389360778481908856057067993634846367663690999501201392634269969314892

4378932395557253634087976106154814264444461900711295393120661914650623502832379340691253835124226483693428320243605068261288082505623702176148298333266322720608621778068987374449904958132661315380947330602852818657061469919021537196527652277068392161545708863351415110089462235727710111500949006978638726979081832600240994001969907020897723174013212365576625760481287046744192124566549615922774378805380656247559903099690092736600250680226860279698003144880843184623701517154967251909911487616335053831460587215690174974677652703489026202458923686267176112697154042850818617080803258117793872354821201303345158215561917205647447472447952814789172239582686120291642904317414205949691098521792189502811697498929450533471990402912812431381462l9

q=503

N= 1473368343497067970444256568877444893496393852065082746563531128401541888391572039293964139796030534847321703171501238365852999402171240694181311556398207863838638085038378161342212041623706893989106721288301268755357157909771738303463063918827078907052203335286124575801997648359753033590213291565794598350552749850483997113086043600259154077983956819866974601414219145345555946576567784062836791701952470963697658181658083276120040439692190187825735960135657031893628033348143995748664729610249389188858778059289203570339528829626678260619050990371188913243429468325425422147525630440393718710973586028108881642516727403753164854278065290327575990616644477240621517409345148024191763314984008049330582011075587024294493031964087690938731873357913745872907640955800154865844018996297984619540216252263474640786425316846640978554827098598294077636499484913378841997108873474864372307960604721456664417213430417022724644245374520806132484446613392896994990093325463796422974910585450278094427131176351440824504537550362128140991240566411476650357664165576356410796926049739507739201559508177282596657684960554512124445762148271938389264594825957189720014612501417818416742535500533219236029061400096260998179707326730003388295237393692435508419313703605921409976661000246402494613340976817412080872122490890693671140711494582481577361639485069211893105101081400754963736789775658534870634903430446756649788390020316591684936372108511384255937817048764557109179164784948333746974634175971473916412214878621329256808436032281658261463058636397097642452092213128948846480134501137062563440985055394320897871284288879429769491735728793112657299347509685716506066232468182505256249929230759647582910507007542316873022238562178215434643716689589825668757887339814101443452046887024802862021038308140 32

959072070787573966541680556217541265024370180098514347696944640243400100
997466890097305292098160919698936942301801341887349873452648846505852749
093216619469533477857465476218942503264193501871499447399817832749380931
991329581421558944771609938084430666306118203015295611635574723929591216
472701472181576259131075083888792596432994790059810617018106909770573927
054340177187815145464943917452441074440503916244218429981425275233612417
995067889778863996100556131914433557846688192340804030116959156275125115
480483369201728616640893008326260497656665328285815684377580925262658280
993656715254260430554551623193364696224128411333163359895621120907443062
952543262542248720026274818950178977715802855258427963726383761115576970
809697847443899109547973134337018265011068312347873650930050389574335998
518429170353545773457212740809462269322325204877370296048950514552203968
142800188210101729476826029234483572660001

q=557

N= 31436835947815504939735209230786931177885872347178830291388080624466
65259177510823585383450287450162437860174344831157005975750402887282756422406111794977964212225653015351187457378511664305430132940419033000485398065234133793780187980309178569069971128986111686401730791313349850770100952800545670841214036764218346320577545241225122218878596846610289736332816198042132922132749387364003961407363723201106618880753954301016634019144551811180700758951887848912347704734321576916431463553052199072323281417264344524719628010834012350592963698008258387821187520117809592682615565342435006842554887011614968145078022230089005709491794047862538460050494754438350421471094033305412242670381020864498458640528225602590162487889636691894560613484923676600502198235653875597107491144079680114849117192333437397451949040486092448149543407131225218969291720583337278915974762842011437601279116420393720191426806429924565687412007292453485664845882187104933370770391872466605967794295750302443036297407214485816682294666105591522444061623610363792331546901262764154351913139910568152907614349971206244652343602161883484078087976341990201603961163817246123308398422529862831341480472087070722632199350224167040607955070361700105980986148002340834137277097889692981985873645819792171040784769934172758809428538534273301011642822642963809560662839841908781241761022132506031616115969336377944603488326377891573440812153361887841979421706121922025489505307099871230981733643827184885660689633746223802835508189148777935785507686295039424424388821615543235089182079604107961011474395386022301754289022289420668669211039386595621948520705741528372957672622856869340783530926015809419042936720922689683499173236089197876758450673686226404091277634672337252440384577878909991682426853377579961962475939853087000605

4208179411883347769320549138080555021443135377998557798892660487157944688
8849771479995788734733936167261821740816061855298568423218972108731127779

8884664662787578796442880106690009853604110396921677273619975674274672011
1487008481993305957854195725195030525254236271892135599081798081984305299
8309624857885417039524063844963801806548192604772343122135012204475223844
5299484664162056901312957200011162858875157375310077210889024982458439733
4218910850584484407846257111905355618022025935648776971546493506230979055
5322246814040665885379160249262398385877863858160259292796788575143045255
8826121346700260791246549448136065215761212797769590519089764896281540355
4984132852578000768755658654943815334595420840006983581578952288401999722
3430944748128150297157139962320128618540666653855525658719099442007265055
9733159824611182428163837075498226714162322553286021449162601636602487699
2528413019001669025873911588329870874544121630533368340827699590759471988
6919864424301554020639520907538151058724104805982388404871953883844267966
6879230310766301715696315662830173400150166803408319149527613292073511699
2810855884771508927677760130213961217643869293327627396756533060499182888
1579621355892566811256965892960509575400033024164588455387579260206740044
861742776007

q=587

N= 2307128339166220356384869939572283870824394409144539542643795995890193987178273115520241818054503417807301768222837967717792820981566830644702359162800544300509265014734969625319556929612542769704066604717079466606254451383751530392951785763206879488934259762209265908205248236609380495287177671336068049945343470166267870306547763090850450657115720133277400330127216224661244357932421420268386193540179511450902704573515501458381718371765667329495164845114517092530988781212528977795859723717188752369103982160361255244680542941119752704472652144886109616166640911345755877234153924068749709052522654836167607872276394353913884274516483822250816195521614029887820081683411389582650352816790628609894821759194942031618048176625033502545095184450354219959929874159088981017838149904295037811036069180307374841862428006072278059681629016011511702736989138215740236904189013294789763119019389997408126279830675629816248489311246308156841456336916329719257538989706927161394501281723776285349203829295731711005240988383503437043955663238266466773847568327599161319578923621303582477187991682443731564866599626452055766915287099824359072779526276024891688726542369551143073413717064263108242793216650585187078878318906967204999978687403917539485347316385365934781108418846907568834511249001998940032542354888360994610482957042538184871373565948246536501224646804674499078891840377462941835163082393476898641177981774992220062982156897486449527341408379294181912778865815270696310167808906854656327703735516468568474702770635674557351429644669494130508946641549192460064217677136624621713639255158897969699139817786044456222588453327516272595757996675919442037848989678730075126061588436261918155877176047511304356948592278804074568302656098263299446288708068668364230563500761030236549294322384863214

5178532048027881028623936268617183983814440912874270937339452781264048088
7965359732631196763986932846528215326641261600673528504845680039299980738
5165899287404170171568944619548947420212027963956636497377187699349963938
0111836844583629242017746676445411851714586872487099159646872351615398718
2677213730256683805195003450686902632014985649583470903568478088315371498
4744395078425995541780828703544642540856803697288319779640731279992206708
2708050812256686228388765653477131458386149889309546777678757072474706168
6063015307302994094264358349324372805898300892619230958095415543914303738
4916791190342878088125448984891972357411612610323035325113185301355541218
3983226028468717549579537667831217898635760169312158812085597312498675518
4094224149066903057338273116643915035758583456499065102503791607608625328
7126970143964046221280292506080577751716453215753048553237980228559367908
4694736255433241444798134936440152226809154893102176602096552797915136748
6249428310066891761410013303946014599236050207821492350527445586247101438
9706540493350514439269842422114955688833463062664809534071624432123317338
2604289107243243742532269527505384046158239725400667417276657445451521708
1497190739765079026407482428606938130502892654821561379547917379150626798
9407639037467801883802521720476891239434623563148074216110531975913025858
1513875447731865899273310864137441444244457037597147191096978285770199138
2391257016747923168815253647186575286436495449927421825998

q=613

N= 37967508226528476918073262558472181670648196920028343275312848652022
731006137370307020676987442002183994418808143266669430480726600439688808
734612919651875354685336086675317451778836232541703370781797628421233986
621113882114750287550413589331025741556424594247272678625424933369479952
906035575536953222212024061067853595995897027086506801293694247831585472
473283793982504448120963539529045576916940311328207397010118906907733134
438156002463085988688578198624203043542640441172445940351642048479064794
901600135687537440306829864048780686888052216006728049834814225230465843
397553612177806776785312574111404954530337227075837766311047304548539721
614260420572119143737959135000890217515831815911664965053219323781672989
913561714059378748237596737940204485211190572328283387167686145706291348
443009069118595985037354127960220822322841537470642086617577394150689232
688547586814322901506039542526920509034321018921683608705025490003714637
594935004580437503658885677649638722634905006654514996919327942153500929
378573897101905966844125508265745274130123440063013213389366224018794689
444257627064785841185890863061458272161017492632199615983764640925492332
762582215506074702028840054105302840000965050573691005403278102840442887
161806489027237661662844523640602418533550965366513567553213189565130768
788507197701630535892235528643724684269480128823147616081912090105716130
888413273124618815079343538499521907535339652992140668156704855778341615
927579297725266865562505895457468506086032335929177823693142921423490975
569507192881613762168079101192514810330028060443842673771758608106469739
293585975316393005242358974900826646319190415919312161988686957578915167
354265095649519699725676784366285528003991128296794940863508278555518285
768876903309786492861645061353473757393739758753002625407156374758134153

2998897247335710444370503299675384437920052234654005288590829239656438985
5350052674559216459096612068100305821814857097999465332120897993308966674
9434233158649565538933250759726167953829147677321705942685472203327216824
4799811919990277729816834667443739887569630976528621331678051647866641402
6334936670605338224914828323027251394830902484546138414147222175210339
5470105886361735051439234403342449009595617986965445059252906893795739925
1506393797451267547254903683598507380372632130034739822712355665394908184
1811707002679510280695811782162195517932042084393633006681441082836202
7186675566861168761455245503368736567246420532376561946224181505210394295
6557430416125121248631063356964862249395225078079587915303167027595047611
4470208081445518468596451879617512095156994970313079890830722946630569
4276878547718436260925231658374338359946332900564218404388293847313603647
1379083620616907190165484691174248168180253525931487133934451107598512261
0358882306857765845539370780715952131808390251562933011059177280749531
9214314363386055458666580762349248608492688603689876706878297633904508763
4548558518903362670220090458152052571001458043316590019737845508301534041
2951695836425044017995523927102577864708953918610576023698918184457377
6642401901587625138158512605004726731502462201100888296742011370237716511
4089816788703708730656638252510286349748545849788211224429094562823919517
2116079484811027369600456902408830263059555651375168495840197918300145
1079003165122253055867951710959900528175357990749843091473424740609760046
60467