A RESEARCH ON THE VARIATIONS OF THE QUADRATIC SIEVE INTEGER FACTORING ALGORITHM

by

SUAT KARADENİZ

JUNE 2008

A RESEARCH ON THE VARIATIONS OF THE QUADRATIC SIEVE INTEGER FACTORING ALGORITHM

by

Suat KARADENİZ

A thesis submitted to

the Graduate Institute of Sciences and Engineering

of

Fatih University

in partial fulfillment of the requirements for the degree of

Master of Science

in

Mathematics

June 2008
Istanbul,Turkey

APPROVAL PAGE

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science.

Prof. Dr. Hakkı İsmail Erdoğan
Head of Department

This is to certify that I have read this thesis and that in my opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

Prof. Dr. Barış Kendirli
Supervisor

Examining Committee Members

Prof. Dr. Barış KENDİRLİ :

Prof. Dr. A. Göksel AĞARGÜN :

Prof. Dr. Feyzi BAŞAR :

Asst. Prof. Dr. Bahattin YILDIZ :

It is approved that this thesis has been written in compliance with the formatting rules laid down by the Graduate Institute of Sciences and Engineering.

Asst. Prof. Nurullah Arslan
Deputy Director

A RESEARCH ON THE VARIATIONS OF THE QUADRATIC SIEVE INTEGER FACTORING ALGORITHM

Suat KARADENİZ

M. S. Thesis - Mathematics
June 2008

Supervisor: Prof. Dr. Barış KENDİRLİ

ABSTRACT

No polynomial time solutions for the integer factorization problem (IFP) have yet been found. The security of RSA cryptosystem is based on the difficulty of the above problem. With the advent of RSA, the IFP has gained a great deal more practical importance. One of the important methods that has been developed so far is the Quadratic Sieve Method (QS).

The work presented here addresses the quadratic sieve integer factorization algorithm and its variations. We will begin with some elementary factorization algorithms and techniques. And then, the quadratic sieve and its variations will be presented together with some Maple implementations.

Keywords: Integer Factorization, Quadratic Sieve, Multi-Polynomial Quadratic Sieve, Large Prime Variation.

TAMSAYILARI ÇARPANLARA AYIRMA YÖNTEMİ KUADRATİK ELEK ALGORİTMASININ VARYASYONLARININ ARAŞTIRILMASI

Suat KARADENİZ

Yüksek Lisans Tezi – Matematik
Haziran 2008

Tez Yöneticisi: Prof.Dr. Barış KENDİRLİ

ÖZET

Tamsayıları asal çarpanlarına ayırmak için polinom zamanlı bir algoritma henüz bulunamamıştır. RSA şifreleme sisteminin güvenliği bir tamsayıyı asal çarpanlarına ayırmanın zorluğuna dayanır. RSA'nın bulunmasıyla birlikte tamsayıları çarpanlarına ayırma probleminin önemi daha da artmıştır. Bu problemin çözümünde şu ana kadar geliştirilen algoritmaların en önemlilerinden birisi "Kuadratik Elek" metodudur.

Bu tez Kuadratik Elek metodu ve varyasyonları üzerine bir çalışmadır. Öncelikle yukarıdaki problemin çözümünde kullanılan bazı temel algoritmalar ve teknikler verilecektir. Maple uygulamalarıyla birlikte, Kuadratik Elek metodu ve varyasyonları detaylı bir şekilde incelenecektir.

Anahtar Kelimeler: Tamsayıları çarpanlara ayırma, Kuadratik Elek , Çok Polinomlu Kuadratik Elek, Büyük Asal Varyasyonu.

DEDICATION

To my family

ACKNOWLEDGEMENT

I wish to express my deepest gratitude to my thesis advisor, Prof. Dr. Barış KENDİRLİ, for his support and encouragement. My special thanks go to Asst. Prof. Dr. Ibrahim KARATAY, Asst. Prof. Dr. Ali ŞAHİN, Asst. Prof. Dr. Bahattin YILDIZ and PhD student Gökhan KELEBEK for their endless support and assistance in my times of need while completing this thesis.

I would like to express my great appreciation to Asst. Prof. Dr. Tuğrul YANIK for his valuable information and experience.

Finally, I would like to thank everybody who was important to the successful realization of this thesis, as well as expressing my apology that I could not mention personally one by one.

TABLE OF CONTENTS

LIST OF TABLES

TABLE

LIST OF SYMBOLS AND ABBREVIATIONS

SYMBOLS /ABBREVIATIONS

IFP	Integer factorization problem
CFRAC	Continued Fraction Integer Factoring Algorithm
ECM	Elliptic Curve Method
FB	Factor base
$\lvert FB \rvert$	Factor base size
SI	Sieving interval
QS	Quadratic sieve
MPQS	Multi-Polynomial Quadratic Sieve
SIQS	Self-Initializing Quadratic Sieve
$\left(\frac{N}{p}\right)$	Legendre symbol
$\pi(x)$	Number of primes less than or equal to x
$\lfloor\ \rfloor$	The greatest integer(floor) function
$\lceil\ \rceil$	The ceiling function

CHAPTER 1

INTRODUCTION

Two fascinating problems of the computational number theory have been primality testing and integer factorization problem (IFP). Many great mathematicians of the past, Fermat, Euler, Legendre, Gauss and many others, worked on these problems and set the basis of the many of the techniques used today.

The former one deals with the question that for a given positive integer N, to say whether N is a prime number or not. En route to the solution, great progress has been shown. There are many successful probabilistic algorithms and even a deterministic one (Agrawal et al.,2004) running at a polynomial time.

You run a primality test on N and it turned out to be not a prime, i.e. composite. Now, the question: what to do next? How to find the prime factors of N?

Although "the fundamental theorem of arithmetic" says that every integer can be decomposed into a product of primes uniquely, most of the time, to cast out those prime factors is not that easy.

IFP looks for those primes, i.e. to extract out prime factors of a composite integer N, in other words, to decompose N into its prime factors as stated in the fundamental theorem. Even today, with the availability of highly fast computers, there seems to be a long way to go on the way to solve the integer factorization problem.

Anyone with elementary knowledge of arithmetic knows divisibility rules for 2, 3, 5, 9,11 and even more. In the very past, they were enough. Why should we ask for more? Unfortunately, every number is not only divisible by them. Today, at this point we need more complicated tools compared to the very primitive tools of the past.

The question "why should we ask for more?" gained a lot more practical importance with the advent of RSA public-key cryptosystem in 1977. Its security is based on the difficulty of the IFP. All the researches done so far on the security of RSA

suggest that breaking RSA cryptosystem is equivalent to factorizing a positive composite integer N which is the product of two primes. The problem in this case is called hard factorization problem.

When RSA was invented, factoring a 50-digit number was very hard. At that time, the Brillhart-Morrison continued fraction algorithm was the best available method and there were a few others, like Pollard's p-1, Pollard's rho, Shank's SQUFOF. With the genesis of Pomerance's quadratic sieve (QS) in 1982, a giant step was taken on the way to solve the IFP. Factoring numbers more than 100 digit became commonplace, i.e. the length of numbers that could be factored doubled. In 1994, 129-digit RSA challenge number was factorized by a variant of this method. Ironically, in 1976 that number was estimated to be safe for 40 quadrillion years by Martin Gardner in Scientific American.
In the spring of 1996, QS lost its crown to Number Field Sieve (NFS). By NFS, 130-digit RSA challenge number was split successfully in about 15% of the time QS would have required (Pomerance,1996). From that time on so far, NFS has been the factoring champion. With this method, between December 2003 and May 2005, RSA-200 was decomposed and 313-digit special Mersenne number $2^{1039}-1$ was split by Special Number Field Sieve (SNFS) in May 2007 by a group of researchers.

What attracted us about studying Pomerance's QS as a thesis topic then? Because the idea behind QS is very beautiful, simple and still effective. It is the fastest known factorization method for the numbers between 40 to 110 digits long. Since it emerged for the first time, many major improvements have been made and nobody knows what future awaits for us.

In Chapter 2, four different factorization methods, all stemming from different ideas, will be studied with their mathematical bases. They are respectively, Fermat's difference of squares, Pollard's- (p-1), Pollard's and Strassen's method and continued fraction algorithm (CFRAC).

In Chapter 3, a common systematic factorization strategy will be studied. Today, it is used by most of the modern factorization methods except elliptic curve method (ECM).

In Chapter 4, we will explain the basic version of the quadratic sieve and the underlying ideas behind its success.

The last two chapters will be about variations of QS and the experimental results obtained from Maple 10 implementations.

CHAPTER 2

FACTORIZATION METHODS

There are numerous techniques employed to factorize an integer into its prime factors. In this chapter, we will briefly explain some of them to give an idea of the richness of the methods and the mathematical tools behind them.

From now on N will represent the composite number to be factorized.

2.1. Fermat's Factoring Method (Difference of Squares)

Trial division just tests a number for divisibility by a prime and if it turns out to be divisible, then divides by that prime. If not divisible, next prime is tested and so on. This method dates back to B.C. and can only be used to extract out small divisors of a number.

After the trial division, Fermat's method is the oldest systematic way of factoring integers. It has a historical importance and the same idea with improvements lies at the heart of the most modern factoring algorithms.

Fermat's idea was to express N as a difference of squares. Let N be a composite odd number, say $N = a \cdot b$. If we are able to write

$$N = x^2 - y^2,$$

$$N = (x+y)(x-y),$$

then two factors of N can be found immediately. So the question arises now is how such a representation of N can be found. Indeed, it is carried out via the squares of half of the sum and difference of two proper factors whose product is N.

$$N = a \cdot b,$$

$$N = \left(\frac{a+b}{2}\right)^2 - \left(\frac{a-b}{2}\right)^2$$

is the form that we look for.

Since ,

$$\frac{a+b}{2} > \sqrt{ab} = \sqrt{N} \quad \text{(by AM-GM inequality if } a \neq b\text{)},$$

we iteratively start by first computing

$m = \lfloor \sqrt{N} \rfloor + 1$, as an approximation to $\frac{a+b}{2}$,

which is the smallest possible value for $\frac{a+b}{2}$ unless N is a square number.

Next, it must be checked that

$t_1 = m^2 - N$

is a square number or not. If it is, then

$N = m^2 - k^2$ where $t_1 = k^2$,

a factorization is found. Otherwise, calculate

$t_2 = (m+1)^2 - N = (t_1 + 2m + 1)$

and test whether it is a square or not and continue until the difference t_i becomes a square number.

Example 2.1. N=1273, $m = \lfloor \sqrt{N} \rfloor + 1 = 36$

m	2m+1	t
36	73	23
37	75	96
38	77	171
39	79	248
40	81	327
41	83	408
42	85	491
43	87	576

In the last row, $t = 576 = 24^2$ has been found, now using that
N can be written as a difference of squares and factorized as

$$\begin{aligned} N &= 43^2 - 24^2 \\ &= (43+24)(43-24) \\ &= 67 \cdot 19 \end{aligned}$$

Obviously, $m = \frac{67+19}{2} = 43$ and $t = \frac{67-19}{2} = 24$.

Some improvements can be made to Fermat's method. Since t must be a square number, last two-digit of t can be any of the following 22 combinations:

Table 2.1 Possible 2-digit endings of a square number

00	01	04	09	16	21	24	25	29	36	41
44	49	56	61	64	69	76	81	84	89	96

This fact greatly simplifies the search for squares in the column t of the table (Riesel,1994).

Another improvement to Fermat's method exists if it is known that the prime divisors of N have a certain form. For example, Legendre's Theorem states that all prime factors p of the number $N = a^n \pm b^n$, with gcd(a,b)=1, are in the form

$$p = k \cdot n + 1,\ k \in Z^+,$$

apart from those which divide the algebraic factors of the form

$a^m \pm b^m$, $m < n$, of N (Riesel,1994).

Shortly,

$$\begin{cases} p = k \cdot n + 1 \ \ where\ N = a^n - b^n \\ p = 2 \cdot k \cdot n + 1 \ \ where\ N = a^n + b^n \end{cases}$$

Example 2.2. The fifth Fermat number $F_5 = 2^{2^5} + 1 = 2^{32} + 1$ has prime divisors of the form $p = 64k + 1$.

$$F_5 = 641 \cdot 6700417$$

$$641 \equiv 1 \pmod{64}$$

$$6700417 \equiv 1 \pmod{64}$$

as stated by Legendre's Theorem.

Lemma 2.1. Fermat's method can be speeded up by a factor of $2n^2$, if all factors of N are in the form $2 \cdot k \cdot n + 1$.

Proof. Let N satisfy the conditions given in the lemma, then

$$N = p \cdot q = (2 \cdot k_1 \cdot n + 1)(2 \cdot k_2 \cdot n + 1)$$

$$N+1=4\cdot k_1\cdot k_2\cdot n^2+p+q$$

$$\frac{p+q}{2}\equiv\frac{N+1}{2}\ (\mathrm{mod}\ 2n^2).$$

So for the numbers $m\geq\left\lfloor\sqrt{N}\right\rfloor+1$ in the method, only the ones such that

$$m=a+2\cdot k\cdot n^2,\ k\in Z^+\ \text{where}\ a\equiv\frac{N+1}{2}(\mathrm{mod}\ 2n^2)$$

must be tested. This gives us a speed-up of factor $2n^2$.

In 1920's, Maurice Kraitchik developed an idea based on Fermat's difference of squares technique.Today, his idea sets the basis of many factorization algorithms (Rabah,2006). This time, we are not looking for differences of squares equal to N, but a multiple of N, i.e.

$$x^2-y^2=k\cdot N$$

$$x^2\equiv y^2\ (\mathrm{mod}\ N).$$

But finding a non-trivial factor is not guaranteed by the above congruence. However, the probability is still as high as 50%, and the chance to obtain a congruence

$$x^2\equiv y^2\ (\mathrm{mod}\ N),$$

is much higher than finding x and y such that

$$x^2-y^2=N.$$

2.2. Pollard's (p-1)-Method

In 1974, J. M. Pollard found this method. It is suitable for certain composite integers which have a special kind of prime divisor.

Definition 2.1. (Cohen,1996) Let B be a positive integer. A positive integer n said to be *B-smooth* if all the prime divisors of n are less than or equal to B. We will say that n is *B-powersmooth* if all prime powers dividing n are less than or equal to B.

Theorem 2.1. If N has a prime divisor p such that $p-1$ is B-powersmooth , then p can be extracted out by calculating

$$\gcd(a^{lcm[1..B]}-1,N)\ \text{ where }\ \gcd(a,N)=1.$$

Proof. Since $p-1$ is B-powersmooth,

$$lcm[1..B] = k \cdot (p-1),\ k \in Z^+ .$$

By Fermat's Little Theorem,

$$a^{p-1} \equiv 1 (\text{mod } p) , \text{ and}$$

$$a^{lcm[1..B]} = a^{k \cdot (p-1)} \equiv 1 (\text{mod } p) .$$

That implies,

$$p \mid a^{lcm[1..B]} - 1 ,$$

so $\gcd(a^{lcm[1..B]} - 1, N) = p$.

Example 2.3. Let $B = 10$. If N has any of the following prime factors p , the following table shows whether they can be detected by this method or not.

p	$p-1$	B -powersmooth
7	$2 \cdot 3$	yes
11	$2 \cdot 5$	yes
13	$2^2 \cdot 3$	yes
19	$2 \cdot 3^2$	yes
23	$2 \cdot 11$	no
29	$2^2 \cdot 7$	yes

The algorithm for this method proceeds as follows (Riesel,1994):

Generate a list of all primes and prime-powers up to some bound B , say 10^6. For each prime square, cube, etc., write the corresponding prime instead of the prime power. Like,

2,3,2,5,7,2,3,11,13,2,17,...

Next, choose a, generally $a = 2$, and compute recursively

$$b_{i+1} \equiv b_i^{p_i} (\text{mod } N) , \qquad (2.1)$$

where p_i is the i^{th} prime in the list. Start the above sequence with $b_1 = a$ and check $\gcd(b_i - 1, N)$ periodically to see a factor p of N has been found, e.g. at regular intervals of 100 cycles. It is because to calculate $\gcd(b_i - 1, N)$ each time is a costly operation.

As soon as the largest prime power in the factorization of p-1, say $q_i^{\alpha_i}$, has been reached in the list, the factor p can be detected.

Example 2.4. Let N=4087, a=2 and B=5. The list of primes and prime-powers is 2,3,2,5. By setting $b_i = a = 2$, the sequence (2.1) generates the following b_i's .

i	b_i	$\gcd(b_i - 1, N)$
1	2	1
2	4	1
3	64	1
4	9	1
5	1831	61

If we check $\gcd(b_5 - 1, N)$, the prime divisor of N,

$$p = \gcd(b_5 - 1, N) = 61$$

is found. Because $p - 1 = 60 = 2^2 \cdot 3 \cdot 5$ is B-powersmooth over the list.

Then, immediately, $N = 61 \cdot 67$.

Phase 2 of the method:

It is likely that p-1 has only one prime factor q which exceeds B.

$$p - 1 = q \cdot \prod_{i=1}^{m} p_i^{\alpha_i},\ p_i^{\alpha_i} < B,\ i = 1,..,m,\ q : prime > B.$$

If q is in a reasonable range after B, then an efficient continuation to (p-1)-method works nicely. This continuation is called *phase 2* of the method.

Assume no factor is found up to the search limit B. Then another bound B_1 is chosen about 10 to 20 times larger than B. Denote the result at the end of the phase-1 by b.

$$b \equiv a^{lcm[1..B]} \pmod{N}$$

Let $\{q_i\}$ be the set of all primes (in order) between B and B_1, q_1 is the largest prime below B. Prepare a list d_i of differences of primes,

$$d_i = q_{i+1} - q_i .$$

Then, recursively, find the value of

$$b^{q_{i+1}} \equiv b^{q_i} \cdot b^{d_i} \pmod{N}, \quad i = 1, 2, 3, \ldots$$

and check whether

$$\gcd(b^{q_{i+1}} - 1, N) > 1.$$

Since differences are small, one cycle in this recursion runs a lot faster than the one in phase-1 (Riesel,1994). For example, the maximum distance between two consecutive primes in the interval $\left[1,\ 4.444 \times 10^{12}\right]$ is 326.

The biggest factor found by this method is 66-digit prime divisor of $960^{119} - 1$ on 29.06.2006 by T. Nohara. In phase1, $B \approx 10^8$ and in phase 2, $B_1 \approx 10^{10}$ were used as bounds. Today in record trials, B ranges from 10^8 to 10^{11} and B_1 from 10^{10} to 10^{17}.

Inspired from the (p-1)-method, in 1982 H. C. Williams found a similar method based on the decomposition of p+1. It is called Williams' (p+1)-method. We will not give details here but in this method Lucas sequences are used in the computations. There is a problem with this method which is to find(actually to guess) a quadratic non-residue D of p, $\left(\frac{D}{p}\right) = -1$. For details, we refer to (Williams,1982). So far, the biggest prime factor found by p+1 method has been 60–digit long.

Due to the methods (p-1) and (p+1), RSA primes are chosen to have a very large B-powersmoothness bounds for p-1 and p+1.

2.3. Pollard's and Strassen's Method

If $x \equiv k! \pmod{N}$ could be computed quickly for $k \in \left[1..N\right]$, then we should be able to factor N quickly (Dixon,1982). If there existed such a method any factor of N in the range 1,..,k could be found easily. It is because, if

$$x \equiv k! \pmod{N},$$

say, p is a factor of N, $1 < p \le k$.

$$x = k! + a \cdot N,\ a \in Z^+,$$

since $p \mid k!$ and $p \mid N$, p must divide x.

Then, $\gcd(x, N) = p$.

A method partially uses the above idea, due to Pollard and Strassen, was found in 1976 and can be used to find the smallest prime factor of N. The procedure goes as: Let b be the smallest factor of N and define

$$F(x) = (x+1)(x+2).....(x+c),\ 1 \le c \le \sqrt{N},$$

$$f(x) \equiv F(x) \pmod{N}.$$

The coefficients of $f(x)$ can be calculated quickly by a method called FFT (Fast Fourier Transform).

$$(c^2)! \equiv \prod_{0 \le i < c} f(ic) \pmod{N}.$$

This can be derived from calculations easily,

$$f(0) \equiv c! \pmod{N}$$

$$f(c) \equiv (c+1)(c+2)....(2c) \pmod{N}$$

$$\vdots$$

$$\vdots$$

$$f((c-1)c) \equiv ((c-1)c+1)((c-1)c+2)....(c^2) \pmod{N}.$$

Let $g_i \equiv f(ic) \pmod{N},\ i = 0,...,c-1.$

If there exists a factor p, $1 < p \le c^2$, of N, it can be extracted out easily by computing

$$\gcd(g_i, N).$$

Example2.5. Choosing N=1633 and c=7, then set

$f(x) \equiv (x+1)(x+2)...(x+7) \pmod{N}$,

$f(0) = 141,\ \gcd(141, N) = 1,$

$f(7) = 544,\ \gcd(544, N) = 1,$

$f(14) = 160,\ \gcd(160, N) = 1,$

$f(21) = 1518,\ \gcd(1518, N) = 23.$

So $N = 23 \cdot 71.$

The fastest-known fully proven deterministic algorithm is the Pollard-Strassen method .

2.4. Continued Fraction Method(CFRAC)

Definition 2.2. The regular continued fraction expansion of a real number number x is an expression of the form

$$x = b_0 + \cfrac{1}{b_1 + \cfrac{1}{b_2 + \cfrac{1}{\ddots}}} = [b_0, b_1, b_2, ...]$$

where b_0 is an integer and the partial denominators $b_1, b_2, ..$ are all positive integers.

The calculations of b_i's can be achieved by successively computing the numbers given by the following algorithm (Riesel,1994):

$$x_0 = x, \qquad b_0 = \lfloor x_0 \rfloor,$$

$$x_1 = \frac{1}{x_0 - b_0}, \qquad b_1 = \lfloor x_1 \rfloor,$$

$$x_2 = \frac{1}{x_1 - b_1}, \qquad b_2 = \lfloor x_2 \rfloor,$$

....

From now on, we are only interested in regular continued fraction expansion of quadratic irrational numbers. They are infinite and periodic.

Definition2.3. Let $x = [b_0, b_1, b_2, ...]$ and

$$\frac{A_n}{B_n} = [b_0, b_1, b_2, ..., b_n], \text{ where}$$

$$\frac{A_n}{B_n} = b_0 + \cfrac{1}{b_1 + \cfrac{1}{\ddots_{b_{n-1} + \frac{1}{b_n}}}}, \quad \gcd(A_n, B_n) = 1,$$

then the rational numbers $\frac{A_n}{B_n}$ is called the n^{th} *convergent* of the continued fraction.

Example 1.6. $\sqrt{7} = [2,1,1,1,4,1,1,1,4,1,1,1,4,....]$ is an infinite continued fraction with period 4 (i.e. 1,1,1,4 repeats infinitely many times). The convergents are

$$\frac{A_1}{B_1}=[2,1],\ \frac{A_1}{B_1}=2+\frac{1}{1}=3,$$

$$\frac{A_2}{B_2}=[2,1,1],\ \frac{A_2}{B_2}=2+\frac{1}{1+\frac{1}{1}}=\frac{5}{2},$$

$$\frac{A_3}{B_3}=[2,1,1,1],\ \frac{A_3}{B_3}=2+\frac{1}{1+\frac{1}{1+\frac{1}{1}}}=\frac{8}{3},$$

$$\vdots$$

Theorem 2.2. Let $\frac{A_n}{B_n}=[b_0,b_1,b_2,...,b_n]$ be the n[th] convergent of a continued fraction.

If we define $A_{-1}=1$, $B_{-1}=0$, $A_0=b_0$, $B_0=1$ then , $\frac{A_n}{B_n}$ can be computed recursively by the formulas

$$\begin{cases} A_s = b_n A_{s-1} + A_{s-2} \\ B_s = b_s B_{s-1} + B_{s-2} \end{cases}, \quad s \geq 1.$$

Proof can be found on p. 330, Riesel 1994.

After this introduction to regular continued fractions, the factorization method CFRAC will be explained now. It was first introduced by D.H. Lehmer and R.E. Powers in 1931. But at that time, calculations were not suitable to be done by hand computers. Around 1970, Morrison and Brillhart developed a systematic way to implement it on computers and F_7 (seventh Fermat number) was factorized by this method in September 1970.

The method:

Let N be an odd composite integer.

Expand $\sqrt{N}$ or $\sqrt{k \cdot N}$, for some suitably chosen k if $\sqrt{N}$ has a small period length, into its continued fractions,

$\sqrt{N} = \left[b_0, b_1, b_2, ..., b_{n-1}, \left\lfloor \frac{\sqrt{N} + P_n}{Q_n} \right\rfloor \right]$ up to some point.

For each value of $n \geq 1$, the identity

$$A_{n-1}^2 - N \cdot B_{n-1}^2 = (-1)^n \cdot Q_n \text{, where } \frac{A_n}{B_n} \text{ is the n}^{\text{th}} \text{ convergent } of \sqrt{N} \text{,}$$

implies the congruence

$$A_{n-1}^2 \equiv (-1)^n \cdot Q_n \pmod{N}, \quad |Q_n| < 2\sqrt{N}. \qquad (2.2)$$

The details of the identity can be found in any elementary number theory book.

Among the $\left(A_{i-1}, (-1)^i Q_n\right)$ pairs, if a subset S of i's such that the product

$\prod_{i \in S} (-1)^i Q_i$ is a square, say Q^2,

can be found, then

$$A^2 = \prod_{i \in S} A_{i-1}^2 \equiv \prod_{i \in S} (-1)^i Q_i = Q^2 \pmod{N}.$$

$A^2 \equiv Q^2 \pmod{N}$ is obtained (Kraitchik's idea). Then by calculating $\gcd(A-Q, N)$ and $\gcd(A+Q, N)$, we have a 50% chance to find proper factors of N.

A systematic way how to find a set S will be explained in the third chapter.

How about calculations of A_{n-1} and Q_n in (2.2) ?

Here we give the recursive computations for parameters used in the original paper (Morrison,Brillhart,1975).

(i) Set $A_{-2}=0$, $A_{-1}=1$, $Q_{-1}=N$, $r_{-1}=b_0$, $P_0=1$, $Q_0=1$ and $b_0 = \lfloor \sqrt{N} \rfloor$.

(ii) Use $b_0 + P_n = b_n Q_n + r_n$, $0 \leq r_n < Q_n$,

to generate r_n where b_n can be found as explained before in the continued fraction expansion.

(iii) Use $A_n = b_n A_{n-1} + A_{n-2} \pmod{N}$,

to calculate $A_n \pmod{N}$ for $n \geq 0$. (It is not necessary to calculate B_n in this algorithm.)

(iv) Use $b_0 + P_{n+1} = 2b_0 - r_n$ to generate $b_0 + P_{n+1}$ for $n \geq 0$.

(v) Use $Q_{n+1} = Q_{n-1} + b_n(r_n - r_{n-1})$ to find Q_{n+1} for $n \geq 0$.

(vi) Increase n by 1 and return to (ii).

Example 1.7. Let N=767. The following table shows the $\left(A_{i-1},(-1)^i Q_i\right)$ pairs generated by the given algorithm.

i	A_{i-1}	$(-1)^n Q_i$
1	27	-38
2	28	17
3	83	-14
4	277	29
5	360	-23
6	637	26
7	230	-23
8	100	29

Choosing the set $S_1=\{5,7\}$ and to obtain the required congruence of Kraitchik,

$$A_4^2 \cdot A_6^2 \equiv (-1)^5 Q_5 \cdot (-1)^7 Q_7 \pmod{N}$$

is calculated. It leads to the congruence

$$360^2 \cdot 230^2 \equiv 23^2 \pmod{N},$$

and two non-trivial divisors are found by computing

$$\gcd(360 \cdot 230 + 23, N) = 13 \text{ and } \gcd(360 \cdot 230 - 23, N) = 59.$$

Then, N=13·59.

CFRAC was the best algorithm in 70's and in the beginning of 80's. It can be used to factorize numbers up to 50-digit long.

There is another method, Dixon's random squares, that works similar to CFRAC but to generate congruences of the form

$$X_i^{\,2} \equiv Q_i \pmod{N},$$

it uses quadratic polynomials $Q(x) = x^2 - N$.

By taking i's randomly where,

$$x_i = \left\lfloor \sqrt{N} \right\rfloor + i,\ i = 0, \pm 1, \pm 2, \ldots,$$

we obtain congruences

$$x_i^{\,2} \equiv Q(x_i) \pmod{N}.$$

In CFRAC, $|Q_i| < 2\sqrt{N}$ for all i=1,2,3,... but in Dixon's random squares those values get larger, $Q(x_i) \approx 2 \cdot i \cdot \sqrt{N}$, $i = \pm 1, \pm 2, \ldots$. Therefore finding a subset S of i's such that the product $\prod_{i \in S} Q(x_i)$ is a square, becomes harder.

That's why, Dixon's method was no better than CFRAC in that sense until Carl Pomerance incorporated the sieving into this method. His brilliant idea will be studied in fourth chapter.

CHAPTER 3

A COMMON FACTORING STRATEGY

More or less, most of the modern factorization methods, namely, CFRAC, QS, NFS, follow a common strategy to factor an integer. The final goal is to obtain a square congruence in the form (Kraitchik's idea)

$$X^2 \equiv Y^2 \pmod{N}, \qquad (3.1)$$

where N is the number to be factorized. Sometimes, (3.1) is called as Legendre's congruence (Riesel,1994,p.149). And we wil adopt this notation.

Once we have (3.1), by calculating $\gcd(X-Y, N)$ and $\gcd(X+Y, N)$, there is at least a 50% chance of extracting out non-trivial factors of N. When N is the product of only two distinct primes p and q, the probability is $\frac{2}{3}$ (Rabah,2006). If $X \neq \pm Y \pmod{N}$, then two proper divisors of N can be found immediately. But in case, $X \equiv \pm Y \pmod{N}$, only trivial factors can be obtained.

So finding t-different Legendre's congruence means, the probability of finding a proper factor is $1-\left(\frac{1}{2}\right)^t$. The more such square congruences are found, the higher the chance to factorize N.

Example 3.1. Let N=21. The congruence $17^2 \equiv 11^2 \pmod{N}$ leads to $\gcd(17-11, 21) = 3$ and $\gcd(17+11, 21) = 7$. Two proper divisors of N are found.

But generating a Legendre's congruence directly in a reasonable time is nearly impossible with today's techniques and computational power as N gets large.

So how to reach our end-goal (3.1)? An efficient and systematic way to the solution of the above question was found by Morrison and Brillhart. The idea was first used in CFRAC, and that's why it was the factoring champion during the 70's and the beginning of 80's until the Pomerance's quadratic sieve. Still it is at the heart of the many factoring algorithms.

Let's see how it works. Firstly, congruences of kind

$$x_i^2 \equiv a_i \pmod{N}, \quad |a_i| < N, \qquad (3.2)$$

are generated . From now on, a_i's will be called "*quadratic residues*" or "*auxiliary numbers*". Every algorithm has its way of finding (3.2). Then a suitable subset I of i's is selected so that the product of a_i's is a square number. Let

$$Y^2 = \prod_{i \in I} a_i \pmod{N} \text{ and } X^2 = \prod_{i \in I} x_i^2 .$$

It is obvious that $X^2 \equiv Y^2 \pmod{N}$.

Are we done? Has the final goal been reached? No, not yet!

Still we don't know an efficient way of obtaining the subset I. The runtime of the methods is very much related to finding the set I quickly and that depends on

(i) generating small residues,

(ii) availability of the residues for sieving.

The superiority of one method to another lies in (i) and (ii).

For the rest of this chapter, the focus will be on the common terminology and the systematic strategy which will be called "*factor base method*" to determine the subset I efficiently.

3.1. Definitions

Definition 3.1. A set of primes $FB = \{ p_i \mid p_i : prime, i = 1,...,k \}$ is named a *factor base(FB)* satisfying $p_{\max} = p_k < B$, where B is a positive integer depending on the size of N. B is called an *upper bound* or *smoothness bound*. Most of the time -1 is also included in the factor base to take advantage of small negative residues. Generally, FB is constructed from the first k-smallest primes into which the auxiliary numbers are likely to factor.

Definition 3.2. The cardinality of the set FB is called *factor base size*, and denoted by |FB|.

Definition 3.3. An integer m is said to be *B-smooth* if all of its prime factors are $\leq B$ (Pomerance,2000). A number which completely factors over the first k-primes is

called p_k-smooth. The smoothness bound B is determined by the largest prime $p_{\max}$ in the factor base.

Example 3.2. Let $FB = \{-1, 2, 3, 7\}$ then the smoothness bound B is 7. So $a_1 = 36 = 2 \cdot 3^2$ is B-smooth but $a_2 = -44 = -1 \cdot 2^2 \cdot 11$ is not.

Definition 3.4. By the fundamental theorem of arithmetic we know that every integer can be decomposed into the product of prime numbers uniquely. Let $m = \prod_{i=1}^{k} p_i^{\alpha_i}$, where p_i denote the i-th prime. The product is over all primes but only finitely many of the exponents α_i are non-zero. Then the vector $v(m) = (\alpha_1, \alpha_2, ..., \alpha_k)$ is called the *exponent vector* of m. Every integer has its own associated exponent vector. For us, the exponent vectors of numbers which are smooth in our factor base will be important. The associated exponent vectors of these numbers have an infinite sequence of zeros representing the exponents of the primes beyond the factor base. Thus, those zeros are omitted and the dimension of the exponent vector becomes the size of the factor base.

Example 3.3. Let $FB = \{-1, 2, 3, 5\}$ and $m_1 = 36,\ m_2 = 30,\ m_3 = -75$ then the associated exponent vectors are

$m_1 = (-1)^0 \cdot (2)^2 \cdot (3)^2 \cdot (5)^0, \quad v(m_1) = (0, 2, 2, 0)$

$m_2 = (-1)^0 \cdot (2)^1 \cdot (3)^1 \cdot (5)^1, \quad v(m_2) = (0, 1, 1, 1)$

$m_3 = (-1)^1 \cdot (2)^0 \cdot (3)^1 \cdot (5)^2, \quad v(m_3) = (1, 0, 1, 2)$

Definition 3.5. A congruence of the kind

$$X^2 \equiv \prod_{p_i \in FB} p_i^{\alpha_i} \pmod{N} \qquad (3.3)$$

is called a *smooth* or *full relation* with respect to FB. Another kind,

$$X^2 \equiv \prod_{p_i \in FB} p_i^{\alpha_i} \cdot P \pmod{N},\ P: \textit{prime},\ P \notin FB$$

is called a *partial or 1-partial relation* with respect to FB. The product of two partial relations with the same P leads to a full relation because

$$\left.\begin{aligned} X_1^2 &\equiv \prod_{p_i \in FB} p_i^{\alpha_i} \cdot P \pmod N, \\ X_2^2 &\equiv \prod_{q_i \in FB} q_i^{\beta_i} \cdot P \pmod N, \end{aligned}\right\} \Rightarrow X_1^2 \cdot X_2^2 \equiv \prod_{p_i \in FB} p_i^{\alpha_i} \cdot \prod_{q_i \in FB} q_i^{\beta_i} \cdot P^2 \pmod N .$$

In the same manner, there may be 2-partial, 3-partial relations etc. However, to find the combinations of 2-partial, 3-partial relations that make a full relation is more complicated. In this case Graph theory is used.The importance of collecting r-partial relations will be seen later.

Actually, $x_i^2 \equiv a_i \pmod N$, $|a_i| < N$ is a smooth relation if and only if a_i is a smooth number in the factor base, i.e. the greatest prime factor of a_i does not exceed the biggest prime in the FB.

Example 3.4. Let N=989 and $FB = \{-1, 2, 3, 5\}$ then

$33^2 \equiv 100 \pmod N$,

$33^2 \equiv 2^2 \cdot 5^2 \pmod N$ is a smooth relation over the factor base, but

$31^2 \equiv -28 \pmod N$,

$31^2 \equiv -2^2 \cdot 7 \pmod N$ is not.

3.2. How to choose the factor base?

The proper choice of the factor base plays a crucial role in the runtime of the factoring algorithms. Now, the problem is how to decide the FB size, or equivalently the upper bound B of the factor base. The following lemma and the next theorem partially shed light on the solution of the problem.

Lemma 3.1.(Pomerance,2000): If $m_1, m_2, .., m_k$ are positive B-smooth integers, and if $k > \pi(B)$, then some non-empty subsequence (m_i) has product a square.

($\pi(B)$ denotes number of primes $\le B$.)

Proof. For a B-smooth m, look at its exponent vector v(m). If m has the prime factorization $m = \prod_{i=1}^{\pi(B)} p_i^{\alpha_i}$, where p_i is the i^{th} prime number and each α_i is a non-

negative integer, then $v(m) = (\alpha_1, \alpha_2, ..., \alpha_{\pi(B)})$. So, a subsequence $m_{i1}, m_{i2}, .., m_{it}$ has product a square if and only if $v(m_{i1}) + v(m_{i2}) + ... + v(m_{it})$ has all even entries. That is, if and only if the sum of vectors is the 0-vector mod 2. Now the vector space $F_2^{\pi(B)}$, where F_2 is the finite field with 2 elements, has dimension $\pi(B)$. And we have $k > \pi(B)$ vectors. So this sequence of vectors is linearly dependent in this vector space. However, a linear dependence when the field of scalars is F_2 is exactly the same as a subsequence sum being the 0-vector. This completes the proof of the lemma.

What actually can be deduced from the lemma for our purpose is that if |FB|=k, then finding (k+1) smooth relations (i.e. smooth quadratic residues) guarantees us finding at least one Legendre's congruence.

Example 3.5. Let's illustrate the case with one example. Choosing N=1081 and $FB = \{2, 3, 5\}$, we need 4 smooth relations over the factor base. Because |FB|=k=3, so finding k+1=4 smooth relations will make it possible to have at least one Legendre's congruence by lemma 3.1.

(1) $33^2 \equiv 8 \pmod{N}$, $a_1 = 8 = 2^3$

(2) $34^2 \equiv 75 \pmod{N}$, $a_2 = 75 = 3^1 \cdot 5^2$

(3) $37^2 \equiv 288 \pmod{N}$, $a_3 = 288 = 2^5 \cdot 3^2$

(3) $41^2 \equiv 600 \pmod{N}$, $a_4 = 600 = 2^3 \cdot 3^1 \cdot 5^2$

The associated exponent vectors are

$v(a_1) = (3,0,0)$, $v(a_2) = (0,1,2)$, $v(a_3) = (5,2,0)$, $v(a_4) = (3,1,2)$.

Then writing the corresponding vectors over the field F_2,

$v'(a_1) = (1,0,0)$, $v'(a_2) = (0,1,0)$, $v'(a_3) = (1,0,0)$, $v'(a_4) = (1,1,0)$

are obtained.

It is easily seen that

$v'(a_2) + v'(a_3) + v'(a_4) = (0,0,0)$.

This implies that the product $(a_2 \cdot a_3 \cdot a_4)$ is a square number. By using congruences (2), (3), (4), the obtained Legendre's congruence is

$(34 \cdot 37 \cdot 41)^2 \equiv (75 \cdot 288 \cdot 600) \pmod{N}$,

$51578^2 \equiv 3600^2 \pmod{N}$,

$771^2 \equiv 357^2 \pmod{N}$.

And to find the factors of N, we calculate

$d_1 = \gcd(771-357, N)$ and $d_2 = \gcd(771+357, N)$.

$d_1 = 23$ and $d_2 = 47$ turn out to be the only nontrivial factors of N.

$N=23 \cdot 47$ is factorized in this way.

Theorem 3.1. Let X be a number to be factorized, B be an upper bound on prime divisors of X and denote by r, the ratio $\frac{\log X}{\log B}$. Then, if $r << B$, we have the probability of X factorizing fully into primes smaller than B approximately r^{-r} (Kechlibar,2005).

The value r^{-r} indeed is the probability of X being B-smooth. Denoting this probability by p, the following tables give us an idea about p depending on the choice of B for a fixed X.

Table 3.1 Probability of B-smoothness for $X = 10^{10}$

B	p
1000	0.018
500	0.0078
200	0.00168
100	0.00032
50	0.000029

$X = 10^{10}$

Table 3.2 Probability of B-smoothness for $X = 10^{20}$

B	p
10000	0.00032
5000	0.000108
2000	0.000018
1000	0.0000032
500	0.000000358

$X = 10^{20}$

The table next is also helpful visualizing the case from a different angle (Bressoud,1989.p.106).

Table 3.3 Probability of $\sqrt[a]{N}$ to be the upper bound for the largest prime divisor of N

a	Probability that the largest prime divisor of N is $< \sqrt[a]{N}$
2	3.07×10^{-1}
3	4.86×10^{-2}
4	4.91×10^{-3}
5	3.55×10^{-4}
6	1.96×10^{-5}
7	8.75×10^{-7}
8	3.23×10^{-8}
9	1.02×10^{-9}
10	2.80×10^{-11}

Lemma 3.1 says that to reach a Legendre's congruence for sure , the number of smooth relations must be more than the factor base size. Keeping the upper bound B of the factor base small means less number of B-smooth residues are needed, but it can be seen from the tables 3.1, 3.2 and 3.3 that as B gets smaller, the probability of detecting a smooth residue decreases.

Example 3.6. Let the typical auxiliary number X generated by the factoring algorithm be around 10^{10} and B (the upper bound for FB) be chosen 1000.

Then, $|FB| = \pi(1000) = 168$. By looking at the table 3.1, it can be seen that the probability of X to be smooth is nearly 0.018. On avarage, $\frac{1}{0.018} \approx 55$ residues must be examined to find one smooth relation. By lemma 3.1, the number of smooth relations must be more than the factor base size. So to guarantee one Legendre's congruence around $169x55 = 9295$ residues must be tested.

Now choosing B=500, the factor base size becomes $|FB| = \pi(500) = 95$.

The probability of X to be smooth is 0.0078. This time approximately $96x\frac{1}{0.0078} \approx 12307$ residues are required to be examined.

The problem that arises now is, how to determine the optimal value of B as a function of a typical auxiliary number X so that the number of trials is minimized.

An approximate solution to the above question was given in a paper in 1983 by Canfield, Erdös and Pomerance.If X is an estimate for the typical auxiliary number then B must be about $\exp\left(1/2\sqrt{\log X \cdot \log(\log X)}\right)$ and the minimum number of trials is about $\exp\left(2\sqrt{\log X \cdot \log(\log X)}\right)$ (Pomerance,1996).

But in real factoring situations, for a particular reason the B-estimation is not as accurate as desired. The reason is that, in practice the factor base does not consist of all primes up to B. Because the generated residues are not as random as suggested in the theory. That's why, in real applications heuristic arguments are used rather than what the theory suggests.

3.3. Linear Algebra Stage

There is only one point left to be clarified. If the number of smooth residues found is more than the factor base size, it is for certain that the product of some combination of them is a perfect square. But, how to find those combinations of residues? At this stage, linear algebra comes into play. By using Gaussian elimination, it can be carried out easily. The steps are as follows:

a) Let $|FB| = k$ and $k+i$, $(i > 0)$ smooth residues be found. First, a matrix A of dimensions $(k+i)\times k$ is formed. The rows of A are the associated exponent vectors of the residues over F_2. It is convenient to arrange the corresponding vectors according to the exponents of primes from largest to the smallest. First coordinates of the vectors represent the power of the largest prime over F_2 and so on.

b) Then a $(k+i)\times(k+i)$ identity matrix is adjoined to the right of A to keep track of the combinations of residues resulting in a perfect square.

c) The rows of the matrix A are linearly dependent, thus at the end of the Gaussian elimination step there must be at least i-zero rows. Each zero row tells us the product of which combination of residues is a square number. So we start Gaussian elimination.

Turning back to the previous example 3.5, let us illustrate the steps.

Given N=1081, $FB=\{2,3,5\}$, |FB|=3 and the smooth residues found are 8, 75, 288, 600.

The associated exponent vectors over F_2 were calculated before in the example but this time to construct the rows of the matrix A, we use them in reverse order according to the step (a).

$$A=\begin{matrix}5 & 3 & 2\end{matrix}\begin{bmatrix}0 & 0 & 1\\ 0 & 1 & 0\\ 0 & 0 & 1\\ 0 & 1 & 1\end{bmatrix}.$$

Adjoining $I_{4\times 4}$ to the right of A leads to the following matrix

$$\left[\begin{array}{ccc|cccc}0 & 0 & 1 & 1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0 & 1 & 0 & 0\\ 0 & 0 & 1 & 0 & 0 & 1 & 0\\ 0 & 1 & 1 & 0 & 0 & 0 & 1\end{array}\right]$$

At the end of the Gaussian elimination, we get

$$\left[\begin{array}{ccc|cccc}0 & 0 & 1 & 1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0 & 1 & 0 & 0\\ 0 & 0 & 0 & 1 & 0 & 1 & 0\\ 0 & 0 & 0 & 0 & 1 & 1 & 1\end{array}\right]$$

As expected 0-rows are obtained and the 1's in the same row of the adjoined matrix tell us the locations of the residues whose product is a square.

In the example above, two zero rows are found . 3rd row tells us that the product of the first and the third residues is a square. And in the same way, from the 4th row it can be deduced that the residues which lead to a Legendre's congruence are in the second, third and fourth locations.

CHAPTER 4

QUADRATIC SIEVE (QS)

Quadratic sieve algorithm (QS) was the first to introduce the idea of sieving effectively into the factoring world. In 1981, Carl Pomerance came up with the new method inspired from Dixon's random squares and from the well-known sieve of Eratosthenes, used to find primes in a given interval. He modified the sieving to the residues of the quadratic polynomial $Q(x) = x^2 - N$. This can be considered as a milestone in the history of the integer factorization problem (IFP). At the beginning of 1980's , CFRAC was able to factor numbers around 50 digits but with the coming of QS, the number of digits was soon doubled.

By using basic QS, Joseph Gerver managed to factor a 47-digit (a factor of $3^{225} - 1$) number from the Cunningham Project in 1982 (Gerver, 1983) then in 1984 at Sandia Laboratories a number consisting of 71-ones was factored with an improved variant of QS (Pomerance,1996). With the parallel implementation of the algorithm, 100-digit numbers were in the range of QS and in 1994, a team distributing the computation over internet set the record. A 129-digit RSA challenge number was split. It was the first signs of that the security of RSA was at risk for the numbers around that size. So far the biggest number factored using QS has been a 135- digit cofactor of $2^{803} - 2^{402} + 1$. It was a special effort in 2001 to show efficiency of the three prime variation of MPQS although QS was not the factoring champion anymore.

What is the reason for the superiority of QS over CFRAC? At first glance, it seems that using continued fractions is more advantageous than using quadratic polynomials to generate small residues. But the trick lies in the clever idea of Pomerance, quadratic sieving. Until then, the common way to examine a number for smoothness was by trial division, testing the number for divisibility by the primes in the factor base and if divisible, dividing the number by that prime. This entails using costly

and time consuming multi-precission division many times. But employing a new technique, quadratic sieve, enables us to detect smooth numbers by doing far fewer divisions and in later variants requiring no division operation at all.

Contrary to Eratosthenes, this time sieving is not used to detect primes in an interval but to locate places of numbers divisible by a certain prime. So this way we already know which residues are divisible by the primes in the factor base thus eliminating the need to test for divisibility. This seemingly small but indeed giant step put QS in front of CFRAC.

There are three basic steps to the quadratic sieve:

1) Initialization
2) Sieving
3) Linear algebra

In the following subsections, these steps will be explained in details.

4.1. Initializing

Let N be the odd number to be factorized and $k = \lfloor \sqrt{N} \rfloor$ (Here $\lfloor \sqrt{N} \rfloor$ denotes the greatest integer less than or equal to $\sqrt{N}$.)

Set $Q(x) = x^2 - N \in \mathbb{Z}[x]$ where x runs over the integers in the interval $[k-M, k+M]$, $M << k$. $[k-M, k+M]$ is the sieving interval, the optimal value of M depends on N.

The residues then become,

$(k-M)^2 - N, (k-M+1)^2 - N, .., k^2 - N, .., (k+M-1)^2 - N, (k+M)^2 - N$.

Now, a factor base for the residues and a sieving interval must be chosen. The approximate sizes of the residues are known and the tools how to decide the optimal factor base size were developed in third chapter. For the sieving interval and the factor base size, the following table will be used while implementing the basic QS. But these values may vary depending on N.

Table 4.1 Approximate parameters for QS (K:1000)

number of digits of N	factor base size	M
10	40	600
15	70	8K
20	300	50K
23	1000	70K
26	1700	200K
28	2000	400K
30	2500	800K

The following two lemmas will also be useful to decide which primes will be included in the factor base and how to sieve the interval with a prime in the FB.

Lemma 4.1. p is a prime. If $p \mid Q(x)$ and $p \nmid N$ then $\left(\frac{N}{p}\right) = 1$,

where $\left(\ \right)$: is the Legendre symbol.

Proof. $p \mid Q(x)$ implies $p \mid x^2 - N$. Then,

$$x^2 - N \equiv 0 \pmod{p}$$

$$x^2 \equiv N \pmod{p}$$

N is a quadratic residue of p. So $\left(\frac{N}{p}\right) = 1$.

Lemma 4.2. Given $Q(x) = x^2 - N \in \mathbb{Z}[x]$, p is a prime and $\alpha \in \mathbb{Z}^+$. Then

$p^\alpha \mid Q(x_0) \iff p^\alpha \mid Q(x_0 + t \cdot p^\alpha), \forall t \in \mathbb{Z}$.

Proof.
$$\begin{aligned} Q(x_0 + t \cdot p^\alpha) &= x_0^2 + 2 \cdot x_0 \cdot t \cdot p^\alpha + t^2 \cdot p^{2\alpha} - N \\ &= x_0^2 - N + p^\alpha (2 \cdot x_0 \cdot t + t^2 \cdot p^\alpha) \\ &= Q(x_0) + p^\alpha \cdot u, \ u = 2 \cdot x_0 \cdot t + t^2 \cdot p^\alpha \end{aligned}$$

By the above identity, the proof of the lemma is straightforward.

The roots of the quadratic congruence

$$x^2 \equiv N \pmod{p^\alpha},\ p \in FB,\ \alpha \in \mathbb{Z}^+ \qquad (4.1)$$

tells us the places of the residues which are divisible by p^α. By using lemma 4.2, those residues in the sieving interval can be distinguished easily.

As an initializing step of QS, for each $p_i \in FB$, the quadratic congruences $x^2 \equiv N \pmod{p_i}$ must be solved. The necessary procedures for the solutions will be given in Theorem 4.1 and Theorem 4.2. Lemma 4.1 says that if $p \in FB$ then $\left(\frac{N}{p}\right) = 1$. So we are now ready to see how to solve the congruence (4.1).

Theorem 4.1. Let p be a prime in the factor base.

(i) If $p = 4 \cdot k + 3$ then

$x \equiv N^{k+1} \pmod{p}$ is a solution to (4.1).

(ii) If $p = 8 \cdot k + 5$ and $N^{2k+1} \equiv 1 \pmod{p}$, then

$x \equiv N^{k+1} \pmod{p}$ is a solution to (4.1).

(iii) If $p = 8 \cdot k + 5$ and $N^{2k+1} \equiv -1 \pmod{p}$, then

$$x \equiv (4 \cdot N)^{k+1} \times \left(\frac{p+1}{2}\right) \pmod{p},$$

is a solution to (4.1).

Proof. Since $p \in FB$, N is a quadratic residue mod p, then

$$N^{\frac{(p-1)}{2}} \equiv 1 \pmod{p}.$$

(i) It is given that $p = 4 \cdot k + 3$ so

$$\left(N^{k+1}\right)^2 = N^{2k+2} = N \times N^{\frac{(p-1)}{2}} \equiv N \pmod{p}.$$

If $p = 8 \cdot k + 5$, then

$N^{4k+2} \equiv 1 \pmod{p}$, which implies two cases.

Case 1:

(ii) $N^{2k+1} \equiv 1 \pmod{p}$

$$\left(N^{k+1}\right)^2 = N^{2k+1} \times N \equiv N \pmod{p}.$$

Case 2:

(iii) $N^{2k+1} \equiv -1 \pmod{p}$

$$\begin{aligned}(4\cdot N)^{\frac{2k+2}{4}} &= 2^{4k+2}\times N^{2k+2}\\ &\equiv (-1)\times(-N) \pmod{p}\\ &\equiv N \pmod{p}\end{aligned}$$

Since $p = 8\cdot k+5$ implies $\left(\frac{2}{p}\right) = -1$.

What if $p \neq 4\cdot k+3$ or $p \neq 8\cdot k+5$?

Then to solve (4.1), Theorem 4.2 comes into play.

Theorem 4.2. Let N be a quadratic residue modulo an odd prime p and h be chosen so that the Legendre symbol $\left(\frac{h^2-4\cdot N}{p}\right) = -1$.

Define a sequence $V_1, V_2, V_3, \ldots.$ by the recursion

$$V_1 = h,$$

$$V_2 = h^2 - 2\cdot N,$$

$$\vdots$$

$$V_i = h\times V_{i-1} - N\times V_{i-2}.$$

We then have that

$$V_{2i} = V_i^2 - 2\times N^i \quad \text{and}$$

$$V_{2i+1} = V_i\times V_{i+1} - h\times N^i,$$

and a solution of (4.1) is given by

$$x \equiv V_{\frac{p+1}{2}}\times\left(\frac{p+1}{2}\right) \pmod{p}$$

(Bressoud,1989. p. 108).

This algorithm was suggested by D. H. Lehmer in 1969. The proof can be found in any elementary number theory book. It must be noted here that computations in this algorithm take longer time than required in Theorem 4.1.

The congruence $x^2 \equiv N \pmod{p}$, $p \neq 2, p \nmid N$ has two solutions in the interval $\{0,1,,..., p-1\}$. One of the roots x_0 can be calculated with the help of Theorem 4.1 and

Theorem 4.2. Then the second solution becomes $p - x_0$. By Lemma 4.2 , all the other zeros of the polynomial $Q(x) = x^2 - N \pmod p$ are known to be of the form

$$x = x_0 + k \cdot p \quad \text{and} \quad x = (p - x_0) + k \cdot p,\ k \in \mathbb{Z}.$$

In this way, the places of residues divisible by p can be located easily and the need for trial division required in CFRAC is eliminated.

By setting sieving interval, factor base and solving congruences (4.1), we are ready to move to the next stage, sieving.

4.2. Sieving

In the sieving process, the locations of $x's$ in the sieving interval where the residues $Q(x)$ are B-smooth (completely decomposes over the factor base) are found.
It is the most time consuming part of QS. While Gerver was factoring 47-digit number, it took 7 minutes to solve congruences (4.1), 6 minutes for the linear algebra stage but 70 hours of CPU time to do sieving.

The basic sieving algorithm can be formulated as follows:
(Buchmann, Muller,2005)

Input: $M \in \mathbb{N}$, $Q(x) = x^2 - N \in \mathbb{Z}[x]$, the factor base FB

Output: Set $\left\{x : \left|x - \lfloor\sqrt{N}\rfloor\right| \le M, Q(x) \textit{ is factor base smooth}\right\}$

(1) Compute and store $Q(x)$ for every $x \in \left\{\lfloor\sqrt{N}\rfloor - M, .., \lfloor\sqrt{N}\rfloor + M\right\}$.

(2) For every prime $p \in FB$ do, compute x_0 and $x_0' \in \{0,1,..,p-1\}$

with $Q(x_0) \equiv 0 \pmod p$ and $Q(x_0') \equiv 0 \pmod p$.

(3) For every $x = x_0 + k \cdot p$ with $\left|x - \lfloor\sqrt{N}\rfloor\right| \le M$ do,

replace $Q(x)$ by $\dfrac{Q(x)}{p}$.

(4) If $\left(x \ne x_0'\right)$ then for every $x = x_0' + k \cdot p$ with $\left|x - \lfloor\sqrt{N}\rfloor\right| \le M$ do

replace $Q(x)$ by $\dfrac{Q(x)}{p}$.

Return all $x \in \left\{\lfloor\sqrt{N}\rfloor - M, .., \lfloor\sqrt{N}\rfloor + M\right\}$ with $\left|Q(x)\right| = 1$.

Having obtained $x's$ for which $|Q(x)|=1$, the decomposition of $Q(x)$ over FB can easily be constructed by trial division. That means, the associated exponent vector for each $Q(x)$ is generated by trial division.

It may happen that there is a factor $p^{\alpha}, \alpha \in \mathbb{Z}^{+}, \alpha \geq 2$ in the prime decomposition of $Q(x)$. In that case, the sieving is done not only by primes but also by prime powers up to some limit. The solutions to the congruences

$$x^2 \equiv N \pmod{p^{\alpha}},\ \alpha \in \mathbb{Z}^{+},\ \alpha \geq 2$$

can be found by Hensel Lifting. But in practice, sieving is done by primes only and the $x's$ are collected by giving some tolerance to $Q(x)$ values ($|Q(x)| \leq L$). Then by trial division, these candidates are tested whether they are FB smooth or not.

Theorem 4.3. (Hensel Lifting) Let $f(x)$ be a polynomial with integer coefficients, p a prime and $\alpha \geq 1$ an integer. It is known that x_{α} is a solution to $f(x) \equiv 0 \pmod{p^{\alpha}}$, k is a solution to $\frac{f(x_{\alpha})}{p^{\alpha}} + k \cdot f'(x_{\alpha}) \equiv 0 \pmod{p}$ where $0 \leq x_{\alpha} < p_{\alpha}$, $0 \leq k < p$, $f'(x_{\alpha})$ denotes the derivative of the function $f(x)$. Then, $x_{\alpha+1} = x_{\alpha} + k \cdot p^{\alpha}$ is a solution to $f(x) \equiv 0 \pmod{p^{\alpha+1}}$. (Tattersall, 2005)

For a particular $p \in FB$, the values of k for which $x = x_0 + k \cdot p$ falls into the interval $[\sqrt{N} - M, \sqrt{N} + M]$ ranges from $\left\lceil \frac{\sqrt{N} - M - x_0}{p} \right\rceil$ *to* $\left\lfloor \frac{\sqrt{N} + M - x_0}{p} \right\rfloor$.

4.3. Linear Algebra Step

Having collected enough number of smooth residues after the sieving stage, it is now time to find a combination of them which forms a Legendre's congruence. This step is generally called Gaussian elimination, and how it is carried out is fully explained in 3.3. Proportional to the size of N, the factor base size increases. This leads to dealing with very large matrices and this may not be easy to handle with respect to memory and time considerations. Since operations are done over F_2, the matrix turns out to be very sparse, i.e. a matrix populated primarily with zeros. For large systems, there are methods which take advantage of this special form of the matrices,namely:

1) Structured Gaussian Elimination
2) The Coordinate Recurrence Method of Wiedemann
3) The Lanczos Algorithm.

For detailed information on these methods we refer to Chapter 7 of (Buchman, Muller, 2005).

To understand basic QS better, let us illustrate all the steps with one example.

Example 4.1. Let's factorize N=1349 by basic QS.

Step 1: Initialization

$$Q(x) = x^2 - 1349,$$
$$\lfloor \sqrt{N} \rfloor = 36,$$
$$M = 7,$$
$$FB = \{-1, 2, 5, 13\}$$

The sieving interval is $[29, 43]$. $([\sqrt{N} - M, \sqrt{N} + M])$

By solving congruences

(a) $x^2 \equiv 1349 \pmod 2$

(b) $x^2 \equiv 1349 \pmod 5$

(c) $x^2 \equiv 1349 \pmod{13}$

we get (a) has solution set $\{1\}$ so all solutions are of the form $1 + 2 \cdot k$,

(b) has solution set $\{2, 3\}$ so all solutions are of the form $2 + 5 \cdot k$ and $3 + 5 \cdot k$,

(c) has solution set $\{6, 7\}$ so all solutions are of the form $6 + 13 \cdot k$ and $7 + 13 \cdot k$.

Step 2: Sieving

By calculating all residues of $Q(x)$, the following table is obtained.

x	29	30	31	32	33	34	35	36
$Q(x)$	–508	–449	–388	–325	–260	–193	–124	–53

x	37	38	39	40	41	42	43
$Q(x)$	20	95	172	251	332	415	500

We start sieving by the primes in the FB.

1) $p = 2$, the locations of residues divisible by 2 are $1 + 2 \cdot k$, $k = 14,..,21$.

The table after sieving by $p = 2$ is shown below.

x	29	30	31	32	33	34	35	36
$Q(x)$	–254	–449	–194	–325	–130	–193	–62	–53

x	37	38	39	40	41	42	43
$Q(x)$	10	95	86	251	166	415	250

2) $p = 5$, the locations of residues divisible by 5 are $2 + 5 \cdot k$, $k = 6,7,8$ and

$3 + 5 \cdot l$, $l = 6,7,8$

The table after sieving by $p = 5$ becomes

x	29	30	31	32	33	34	35	36
$Q(x)$	–254	–449	–194	–65	–26	–193	–62	–53

x	37	38	39	40	41	42	43
$Q(x)$	2	19	86	251	166	83	50

3) $p = 13$, the locations of residues divisible by 13 are $6 + 13 \cdot k$, $k = 2$ and

$7 + 13 \cdot l$, $l = 2$

The resulting table after sieving by $p = 13$ is

x	29	30	31	32	33	34	35	36
$Q(x)$	–254	–449	–194	–5	–2	–193	–62	–53

x	37	38	39	40	41	42	43
$Q(x)$	2	19	86	251	166	83	50

4) Finally, sieving by powers of 2 and 5, i.e. 2^2 and 5^2,

the table now is

x	29	30	31	32	33	34	35	36
$Q(x)$	−127	−449	−97	−1	−1	−193	−31	−53

x	37	38	39	40	41	42	43
$Q(x)$	1	19	43	251	83	83	5

By collecting values of x such that $|Q(x)| \le 13$, the algorithm returns the set $\{32, 33, 37, 43\}$.

To obtain the associated exponent vectors, we do trial division.

$$Q(32) = -1 \cdot 5^2 \cdot 13,$$

$$Q(33) = -1 \cdot 2^2 \cdot 5 \cdot 13,$$

$$Q(37) = 2^2 \cdot 5,$$

$$Q(43) = 2^2 \cdot 5^3.$$

	13	5	2	−1
$Q(32)$	1	2	0	1
$Q(33)$	1	1	2	1
$Q(37)$	0	1	2	0
$Q(43)$	0	3	2	0

Exponent vectors are $v_1 = (1, 2, 0, 1)$,

$$v_2 = (1, 1, 2, 1),$$

$$v_3 = (0, 1, 2, 0),$$

$$v_4 = (0, 3, 2, 0).$$

The vectors then calculated over F_2 so we get

$$v_1' = (1, 0, 0, 1),$$

$$v_2' = (1, 1, 0, 1),$$

$$v_3' = (0, 1, 0, 0),$$

$$v_4' = (0, 1, 0, 0).$$

After Gaussian elimination step, two Legendre's congruences are found.

First one, $32^2 \cdot 33^2 \cdot 37^2 \equiv 1300^2 \pmod{N}$ gives two trivial factors 1 and N.

But the second one, $32^2 \cdot 33^2 \cdot 43^2 \equiv 6500^2 \pmod{N}$ results in two proper factors 19 and 71 after the calculations $\gcd(32 \cdot 33 \cdot 43 - 6500, N)$ and $\gcd(32 \cdot 33 \cdot 43 + 6500, N)$.

$N = 19 \cdot 71$ is factorized by using basic QS.

CHAPTER 5

VARIATIONS OF QUADRATIC SIEVE

Since the QS emerged for the first time, many improvements have beeen made to it. Some techniques used in CFRAC also adapted to QS. There were two main handicaps with the basic version. First one is that sieving takes a lot of time in the factorization of big numbers since the factor base and sieving interval get large proportional to N's size. The second one is related to the quadratic residues. They increase very rapidly so the probability of the residue to be factor base smooth decreases and this requires sieving very large intervals even to detect one smooth relation. In the rest of this chapter, the methods developed to solve these drawbacks of the basic QS and other improvements will be studied .

5.1. Use of a Multiplier

It is always desirable to have many small primes in our factor base. To enable this, N is multiplied by a suitable number k and k is called a multiplier.So the number to be factorized becomes $k \cdot N$. The multiplier k must be a small square-free integer and obviously the factor base changes accordingly. Generally, k is chosen to be between 1 and 100, otherwise $k \cdot N$ becomes very large and it takes away more than it compensates. Multiplying N by k increases the size of residues by a factor of $\sqrt{k}$. So how to choose multiplier while trying to maximize the number of small primes in the factor base and trying to minimize k ?

The selection of k is done according to the function that will be given next.

We select a value of k that maximizes the modified Knuth-Schroeppel function

$$-\frac{1}{2}\log k + \sum_{p \le B} E_p(k) \cdot \log p \, .$$

Here,

$$E_p(k)=\begin{cases} p=2 & \begin{cases} 2 \ if \ \ N\equiv 1(\mathrm{mod}\,8) \\ 0 \ otherwise \end{cases} \\ else & \begin{cases} \frac{1}{p} \ if \ \ p\,divides\,k \\ \frac{2}{p} \ otherwise \end{cases} \end{cases}$$

where the sum is over those primes $p \le B$ with

$$p=2,\ \left(\frac{kN}{p}\right)=1 \ \ or \ \ p \mid k \qquad \text{(Kurowski,1998).}$$

The quality of the multiplier k and the corresponding value of the modified Knuth-Schroeppel function may reduce the runtime by a factor up to 2.5 (Silverman,1987).

For example, N to be of the form $8k+1$ is a case we want because then $Q(x)=x^2-N$ is divisible by 8 for all odd values of x. Most of the time, setting

$$k = N \bmod 8$$

is a good choice of multiplier for moderate size N since $k \cdot N$ turns into the desired form $8k+1$.

Example 5.1. Let N=923, $FB=\{-1,2,13,19\}$, $SI=[23,37]$ (SI:sieving interval)

In the sieving interval, there is only one FB-smooth residue. But, if a multiplier $k=3\equiv N \pmod 8$ is used, then the parameters become

$N'=3\cdot N=2769$, $FB'=\{-1,2,3,5\}$, $SI'=[45,59]$.

And this time, even if $|SI|=|SI'|$, there are 3-smooth residues detected in the SI'.

5.2. Logarithm Variant

The aim of sieving process is to identify locations of x such that $Q(x)=x^2-N$ factors completely into our factor base. In the basic QS, by solving quadratic congruences $x^2 \equiv N \pmod{p^\alpha}$, places of x's in the sieving interval where $Q(x)$ is divisible by p^α are found and those $Q(x)$'s are then divided by p. The logarithm variant replaces these slow division operation by subtraction or addition. Since latter

ones are much faster than division, in this way sieving time can be reduced greately depending on the size of the factor base and the sieving interval. The steps can be summarized as follows:

(1) Set an array of size equal to the sieving interval, then compute and store the approximate value of $\log|Q(x)|$ in the location corresponding to the argument x, $x \in [\sqrt{N} - M, \sqrt{N} + M]$.

(2) Identify x's in the SI, where $Q(x)$ is divisible by p.

(3) Subtract from those locations the weight $\log p$ associated with p. This step replaces division by faster subtraction.

(4) Repeat the steps (2) and (3) for each $p \in FB$.

(5) Scan the array for residual logs that are close to 0 and these locations correspond to the values of $Q(x)$ that factors completely into our factor base (Pomerance,1985).

(6) Do the usual trial division and linear algebra steps.

In later modified logarithm variants, subtraction is changed to addition in the following way: In step (1), all array entries are initialized to 0 instead of computing $\log|Q(x)|$ for each x. There is no change in step (2). But in the next stage, the weight $\log p$ is added to the corresponding locations rather than subtracting. It is repeated for each $p \in FB$. And this time to identify x's such that $Q(x)$ is FB-smooth, we scan the array for the summed logs that are close to *target* that will be defined next.

If we are sieving over $2 \cdot M + 1$ values, then the logarithm of the absolute value of $(\lfloor \sqrt{N} \rfloor - M + i)^2 - N$, $i = 0,..,2M$ will be approximately

$$target = \frac{\log N}{2} + \log M \text{ . (Bressoud,1989)}$$

This logarithm variant, suggested by Silverman (Silverman,1987), speeds up the sieving process because first step in the subtraction version requires calculating $\log|Q(x)|$, $2 \cdot M + 1$ times. But in Silverman's version, we just initialize all cells of the array to 0. Therefore, proportional to the size of M, a great deal of time is saved.

When the sieving is done by addition, those entries close to *target* are collected. But how close?

In (Silverman,1987), it is suggested to collect indexes where a value bigger than

$$threshold = target - T \times \log p_{\max}$$

is accumulated. Here p_{max} is the biggest prime in the factor base and T is a constant near 2. The Table 5.1 gives optimal values of T with respect to N.

Table 5.1 T-adjustment of the threshold value

Number of digits of N	T
24	1.5
30	1.5
36	1.75
42	2.0
48	2.0
54	2.2
60	2.4
66	2.6

It must be noted that in this implementation a very few number of fully factorable $Q(x)$'s may be missed but the time saved makes up for more than what's lost.

5.3. Small Prime Variation

On avarage, the sieving time takes more than 85% of the total running time (Boender,Riele,1995). So it is important to optimize the sieving process. Another method,called *small prime variation*, saves about 20% of the sieving time. This variation is based on the following idea:

Let $p_i < 10$ and $p_j > 100$, $p_i, p_j \in FB$. In the SI, the multiples of p_i occur at least 10 times as frequently as p_j's. Therefore, sieving with small primes takes a substantial percentage of overall time. And also at the occurences of small primes p_i,

$\log p_i$ is added to the locations, so compared to $\log p_j$'s, they don't contribute much to the sieving. That's why, in practice, it is customary to start sieving with primes bigger than a fixed bound L, generally with $L \approx 30$, and then the threshold value is reduced by

$$\sum_{p<L} \log p \quad \text{or} \quad \sum_{p^{\alpha}<L} \log p,$$

so we don't lose anything. If L is defined ≈ 30, then $\sum_{p<L} \log p < 20$.

But the negative side of this variation is that after sieving, depending on the modified threshold value, it may cause a few false residues to be collected. Nevertheless, the resulting performance is much better.

5.4. Special q-Polynomials

As mentioned earlier at the beginning of this chapter, the rapid growth in the residues of the quadratic polynomial was one of the main drawbacks of basic QS. The larger the residue, the less likely it is to factor over the factor base. While implementing QS at Sandia Lab., Jim Davis developed an important enhancement that mitigated this handicap. He found a way to switch to the other quadratic polynomial after values of the first one, $Q(x) = x^2 - N$, grew uncomfortabley large (Pomerance,1996).

After sieving, it is very likely to come up with a residue at location x_0 such that

$$Q(x_0) = \prod_{i=1}^{|FB|} p_i^{\alpha_i} \cdot q,$$

where $p_i \in FB$ and $p_{\max} < q < p_{\max}^2$.

If q is a prime, and most of the time it is, consider

$$\begin{aligned} Q(x_0 + k \cdot q) &= (x_0 + k \cdot q)^2 - N \\ &= Q(x_0) + 2 \cdot k \cdot q \cdot x_0 + k^2 \cdot q^2 \\ &= Q(k). \end{aligned}$$

Every term of $Q(k)$ is divisible by q and the magnitude of

$$\frac{Q(x_0 + k \cdot q)}{q}$$

is essentially that of $Q(k)$ (Davis,Holdridge,1983). Then the residues to be sieved are generated by $Q(k)$ for $k \in [-M, M]$. The sieving on these residues is done by the usual way. These special q's are easy to find and at no cost. To keep the residues small , sieving is done over small intervals and when we are done, it is switched to another special q-polynomial.

This modification enables to factor big numbers in less time.

For example, Sieving of 58-digit Cunningham number took about one-sixth of the time that single polynomial version required. (Davis, Holdridge,1983)

5.5. The Multi-Polynomial Quadratic Sieve (MPQS)

Another remedy for the uncomfortable growth of the residues was found by

P. Montgomery. In the previous section, special q-polynomials were used to keep residues small by switching to another polynomial. Inspired from this method,

P. Montgomery developed a better strategy than using special q-polynomials. It is called *Multi-Polynomial Quadratic Sieve (MPQS* for short). As the name suggests, several different polynomials are used to keep the sieving interval short and as a result residues small. The basic version of QS can be used to factorize integers up to 50-digit long in a reasonable time(Buchmann,Muller,2005) but with MPQS, this can be up to 100-digits. The idea and the computations of the polynomials will be explained in the rest of this chapter.

As a polynomial set $Q(x) = ax^2 + bx + c$ such that $b^2 - 4ac = N$, $a, b, c \in \mathbb{Z}$. Because only in that case $Q(x)$ generates quadratic residues.

b is selected odd then $b^2 - 4ac \equiv 1 \pmod 4$.

N must be of the form $4k+1$ because of the previous arguments, if not, N is multiplied by a suitable multiplier k to convert into the desired form.

$Q(x) = ax^2 + bx + c$,

$$Q(x) = a\left(x + \frac{b}{2a} \right)^2 - \frac{b^2 - 4ac}{4a}, \qquad (5.1)$$

then

$$Q(x) \equiv a\left(x + \frac{b}{2a} \right)^2 \pmod{N} \qquad (5.2)$$

Let $a = d^2$, d : *prime* and $\left(\frac{N}{d} \right) = 1$.

The equation (5.1) and the congruence (5.2) become

$$Q(x) = \left(dx + \frac{b}{2d} \right)^2 - \frac{b^2 - 4d^2 c}{4d^2}, \qquad (5.3)$$

$$Q(x) \equiv \left(dx + \frac{b}{2d} \right)^2 \pmod{N}. \qquad (5.4)$$

We will adopt (5.4) for the congruences. This idea, choosing $a = d^2$, is Pomerance's. What we want is to make the value of $Q(x)$ as small as possible over $[-M, M]$.

By minimizing,

$\int_{-M}^{M} |Q(x)|\, dx$ with the constraint $b^2 - 4ac = N$, $a, b, c \in \mathbb{Z}$,

it is obtained that,

$$a = \frac{\sqrt{N}}{\sqrt{2} \cdot M},$$

$$b = 0,$$

$$c = \frac{-1}{2\sqrt{2}} M \sqrt{N}.$$

Minimizing $\int_{-M}^{M} |Q(x)|\, dx$ is equivalent to minimizing $Q(x)$ values small because the base of the parabola $2M$ is fixed and the area is directly proportional to the height of the parabola, i.e. $Q(x)$ values. The constructed polynomial $Q(x)$ satisfies the inequality

$$|Q(x)| \le \frac{1}{\sqrt{2}} M\sqrt{N}, \quad x \in [-M, M].$$

A $2\sqrt{2}$ improvement over q-polynomials which take values $\le 2 \cdot M \cdot \sqrt{N}$ in $[-M, M]$.

The selection of the parameters and the computations are done as follows:

(1) Determine the size of FB and the length of the sieving interval $2M+1$.

(2) Select a multiplier k such that $kN \equiv 1 \pmod 4$.

(3) Choose $a = d^2$ where d is a prime with

$$\left(\frac{kN}{d}\right) = 1, \; d \approx \sqrt{\frac{\sqrt{kN/2}}{M}} \text{ and } d \equiv 3 \pmod N.$$

(4) Solve $b^2 \equiv kN \pmod a$ for b. kN must be a quadratic residue modulo d since $b^2 - 4ac = kN$. By an elementary application of Hensel' Lemma (by Wagstaff), $b^2 \equiv kN \pmod a$ can be solved easily.

The solution is

$$b \equiv h_1 + h_2 d \pmod{d^2} \text{ where}$$

$$h_1 \equiv (kN)^{\frac{d+1}{4}} \pmod d \text{ and}$$

$$h_2 \equiv (2h_1)^{-1}\left(\frac{kN - h_1^2}{d}\right) \pmod d.$$

b must be odd, in case b is even, subtract it from d^2.

(5) $\forall p_i \in FB$,

The roots of $Q(x) \equiv 0 \pmod{p_i}$

are $\frac{-b \pm \sqrt{kN}}{2a} \pmod{p_i}$.

Here, $\pm\sqrt{kN}$ stands for the integer solutions of the congruence

$$x^2 \equiv kN \pmod{p_i}.$$

To summarize whole procedure as an algorithm

While (not enough smooth residues found)

Begin

Generate coefficients for the polynomial.

Solve $Q(x) \equiv 0 \pmod{p_i}$, $\forall p_i \in FB$.

Do the sieving.

Scan the sieve array: If any value exceeds the threshold value,

Begin

Compute $Q(x)$ and find its factorization via trial division.

Save the value of H where $Q(x) \equiv H^2 \equiv \left(\frac{2 \cdot a \cdot x + b}{2 \cdot d} \right)^2 \pmod{kN}$

and the exponent vector of $Q(x)$.

End.

End.

(Silverman,1987)

Example 5.2. Given N=13223521. Let us do the necessary computations step by step to generate a polynomial.

Set M=20.

$N \equiv 1 \pmod{4}$, then the multiplier $k = 1$.

$a = d^2$, $d \approx \sqrt{\frac{\sqrt{N/2}}{M}}$. $d = 11$ and $d \equiv 3 \pmod{4}$, $\left(\frac{N}{d} \right) = 1$.

Then $a = 121$.

By solving the congruence

$b^2 \equiv N \pmod{a}$,

the roots are found to be $b = 6$ and $b = 115$.

Since b must be odd, $b = 115$.

Using $b^2 - 4ac = N$, the last coefficient $c = -27294$ is calculated.

$Q(x) = 121x^2 + 115x - 27294$ is the polynomial we want. And obviously,

$$Q(x) = \left(11x + \frac{115}{2 \cdot 11}\right)^2 - \frac{N}{4 \cdot 11^2},$$

$$Q(x) \equiv \left(11x + \frac{115}{2 \cdot 11}\right)^2 \pmod{N}.$$

Some relations found by using $Q(x)$,

$Q(1) = -27058 \equiv 7813915^2 \pmod{N}$ and $Q(0) = -27294 \equiv 7813904^2 \pmod{N}$.

5.6. Self-Inititializing Quadratic Sieve (SIQS)

In MPQS, the polynomial change is costly. For each polynomial, to find locations which are divisible by $p \in FB$,

$$(2a)^{-1} \pmod{p} \qquad (5.5)$$

has to be calculated. For example, for a 60-digit number, the factor base size is about 3000, so every polynomial change requires computing (5.5), 3000 times. The cost of switching to another polynomial is dominated by calculating inverse of $(2a)$ for each prime.

An efficient way of changing polynomials was found by Alford and Pomerance in 1993. This variant of MPQS is called self-initializing quadratic sieve (SIQS). Since polynomials can be changed quickly, in this variant, smaller M than MPQS can be used. Therefore, the residues become smaller.

The calculations of the polynomials are done in the following way:

(i) Choose $k < 30$ primes $d_1, d_2, ..., d_k \in FB$ such that

$$d_i \approx \left(\frac{\sqrt{2N}}{M}\right)^{\frac{1}{2k}} \quad \text{and} \quad \left(\frac{N}{d_i}\right) = 1.$$

(ii) Let $d = d_1 \cdot d_2 \cdot ... \cdot d_k$ and $a = d^2$.

(iii) By the CRT(Chinese Remainder Teorem),

$$b^2 \equiv N \pmod{a}$$

has 2^k solutions for b. And each root satisfies the condition

$$b_i^2 - 4ac = N,\ 1 \le i \le 2^k .$$

But only 2^{k-1} of the solutions are useful as will be explained next.

b is the solution to the system of congruences by CRT,

$$\begin{aligned} x &\equiv +b_1 \pmod{d_1^2} \\ x &\equiv \pm b_2 \pmod{d_2^2} \\ &\vdots \\ x &\equiv \pm b_k \pmod{d_k^2} \end{aligned}$$

where $(\pm b_i)^2 \equiv N \pmod{d_i^2}$. By using only $+b_i$ in the first congruence, there are 2^{k-1} solutions. Because $b_i \equiv \pm b_j \pmod{a}$ implies the generated polynomials $Q_1(x) = (2ax+b)^2 - N$ and $Q_2(x) = (2ax-b)^2 - N$ are mutually symmetric about $x = 0$. Then to use only one such b, i.e. to eliminate the case $b_i \equiv \pm b_j \pmod{a}$, we fix the sign in one of the congruences. There are 2^{k-1} combinations of signs, that means corresponding to each a, 2^{k-1} different b's can be generated, all 2^{k-1} possible polynomials are obtained in this way.

The costly operation $(2a)^{-1} \pmod{p_i}$, $p_i \in FB$ is calculated only once and then stored. When b is changed, i.e. the polynomial, the precomputated values (inverses) are used at no cost. By this variation, a speed-up of a few percent of the total computing time can be gained(Boender, Riele ,1995).

5.7. Large Prime Variation

If the residue at a location x in the sieving interval completely decomposes over the factor base, it is called a full(smooth) relation. The question is now how to make use of partial relations

$$Q(x) = \prod_{p_i \in FB} p_i \cdot P \text{, where } P \notin FB \text{, P:prime.}$$

In chapter 3, it was shown that two 1-partial relations with the same large prime P lead to a full relation. By the Birthday Paradox, there is a reasonable good chance that the same P will appear twice or more after collecting enough number of partial relations.

And finding 1-partial relations is at no extra cost. For example, for the factorization of 102-digit number, 180879 1-partial relations were collected and as a result 11433 additional full relations were obtained (Denny,1993).

To detect large primes which are slightly bigger than p_{max}, the threshold value is kept to be $p_{\max}^2$. In logarithm variant, it is set to be $2\log p_{\max}$. After the trial division by the numbers in the FB, the remainder P must be a prime between

$$p_{\max} < P < p_{\max}^2 \quad .$$

In practice an upper bound L for the threshold value is chosen because collecting all partial relations require a lot of disk space and bigger P's don't contribute much. Generally, $10 < \frac{L}{p_{\max}} < 100$ is a good choice.

In the same way, relations of the form

$$Q(x) = \prod_{p_i \in FB} p_i \cdot P_1 \cdot P_2 \text{ , where } P_1, P_2 \notin FB \text{ , } P_1, P_2 \text{:prime, can be found.}$$

Such relations are called partial-partial (2-partial) relations . To find combinations of 1-partial and 2-partial relations that give a full relation, the cycles in undirected graphs are used (Boender,Riele,1995). In this double prime variation, if the remainder R after the trial division satisfy $L < R < L_1$ and R is composite, then the large prime factors of R can be found by a suitable factoring algorithm. Generally, Shank's SQUFOF or ECM(Elliptic Curve Method) is used. So this variation requires some extra computation.

More large prime variations can be used for factorization of very big numbers. Earlier studies have shown that using one large prime variation is always better than none, and the double prime variation is more efficient when factoring integers with more than 80-digits (Leyland et al., 2001). In 1994, the RSA challenge number (RSA-129) was factored using PPMPQS (Double Large Prime variation of MPQS) and in 2001 a 135-digit number was factored using TMPQS (Three Large Prime variation of MPQS).

CHAPTER 6

EXPERIMENTAL RESULTS

In the previous chapter, the variations of the QS have been introduced. In order to show the effects of them, we wrote maple codes and ran tests on random numbers between 10 to 35 digits. Because the linear algebra stage is fixed for all variations, we did not implement that part. Since the main aim is to reduce the sieving time and the efficiency can be measured with respect to it, we measured it for each variation.

The test numbers we used are as follows:

c10=3605478927,

c15=781254789636521,

c20=86587412021455590143,

c23=42589740114577785544069,

c26=62111072210124773690021303,

c28=7896510144129021400155877441,

c30=325412235487201240012587445447,

c32=12345678909876543210122233301039,

c34=2021458722547810032556987410235477.

Here, c stands for compositeness and the adjoining number next to c denotes the number of digits of the number.

We used the following abbreviations throughout our tests:

M: sieving interval,

FBS: factor base size,

SR: number of smooth relations found,

SR: sieving time,

K: 1000,

s: second.

6.1. Using a Multiplier Test

Number = c10 , M=600, FBS=40

Multiplier	SR
1*	42
7	77

Number = c20 , M=50K, FBS=300

Multiplier	SR
1	140
7	390
17*	562

Number = c26 , M=200K, FBS=1700

Multiplier	SR
1	982
5*	1739
7	1726

It can be observed that using a good multiplier may reduce the runtime of QS up to 3 times. Here, the ones with a (*) shows the multiplier calculated by our maple code.

And we also run test using the multiplier k where k=N mod 8. As a method, basic QS is used.

6.2. Logarithm Variant Test

Number = c23 , M=70K, FBS=1000, Multiplier=1

Method	SR	ST
Basic QS	1006	17.045s
Log Variant (T=1.1)	1001	14.969s

Number = c28 , M=200K, FBS=2000, Multiplier=1

Method	SR	ST
Basic QS	2070	92.171s
Log Variant (T=1.2)	2072	59.984s

Number = c30 , M=500K, FBS=2500, Multiplier=3

Method	SR	ST
Basic QS	1853	408.405s
Log Variant (T=1.2)	1852	215.596s

As the number gets bigger, the efficiency of the logarithm variant increases. The initialization step in the logarithm variant is faster but here it is not calculated.

6.3. Small Prime Variation Test

Number = c23 , M=70K, FBS=1000, Multiplier=1

Method	SR	ST
Basic QS	1006	17.045s
Small Prime Variation	972	7.765s

Number = c28 , M=200K, FBS=2000, Multiplier=1

Method	SR	ST
Basic QS	2070	92.171s
Small Prime Variation	1895	42.890s

Number = c30 , M=500K, FBS=2500, Multiplier=3

Method	SR	ST
Basic QS	1853	408.405s
Small Prime Variation	1831	194.095s

In the small prime variation, sieving has started from the 7^{th} prime in the factor base. A few smooth relations were missed but the time gained compensated for them.

6.4. Comparing Variations

In this section, the sieving times of basic QS, special q-Polynomials and MPQS will be measured. For all of them, small prime and logarithm variant will be used. For the sieving interval and the factor base size, the following table (Bressoud, 1989, p. 118) will be helpful while implementing the special q-Polynomials and MPQS .

Table 6.1 Optimal parameters for MPQS and q-polynomials (K:1000)

number of digits of N	factor base size	M
24	100	5K
30	200	25K
36	400	25K
42	900	50K
48	1200	100K
54	2000	250K
60	3000	350K
66	4500	500K

Number = c15 , Multiplier=1

Method	M	FBS	ST	SR	# of Poly.
Basic QS	8K	70	0.968s	71	1
q-Poly.	700	50	1.752s	65	4
MPQS	700	50	1.515s	66	4

Number = c26 , Multiplier=5

Method	M	FBS	ST	SR	# of Poly.
Basic QS	200K	1700	38.733s	1732	1
q-Poly.	25K	200	17.984s	219	12
MPQS	25K	200	11.218s	221	11

For MPQS, the multiplier was set to be 7, otherwise the number is not of the form 4k+1.

Number = c32 , Multiplier=1

Method	M	FBS	ST	SR	# of Poly.
Basic QS	800K	3500	270.842s	2829	1
q-Poly.	70K	600	82.421	639	16
MPQS	70K	600	51.673	647	13

For MPQS, the multiplier was 59, otherwise the number is not of the form 4k+1.

Number = c34 , Multiplier=97

Method	M	FBS	ST	SR	# of Poly.
Basic QS	-	-	-	-	-
q-Poly.	1000K	1000	167.047s	1045	17
MPQS	1000K	1000	110.923s	1033	17

From the experimental results, it can be observed that MPQS outperforms others and the basic QS is not suitable for the numbers above 30 digits. But for the small numbers basic QS has the best performance.

All the tests were run on Maple 10 using a single PC with the following properties Pentium(R) D CPU 2.80 GHZ and 1GB RAM. You can obtain the codes by contacting me via skaradeniz@fatih.edu.tr.

CHAPTER 7

CONCLUSION

One of the greatest steps taken on the way to solve the IFP was the invention of the QS. It is very simple and still efficient. For the numbers that range from 50 digits to 110 digits, QS is the fastest known algorithm. Since a basic version of it was implemented for the first time, many improvements have been made. In this thesis, we did a thorough survey on these improvements(variations) and by using maple, the effects of some of them have been shown experimentally.

In this research, we applied some variations of the quadratic sieve to some specific randomly chosen lerge integers.

As a result of our experiments, it was observed that using an appropriate multiplier can significantly reduces the runtime of the QS. The logarithm variant and small prime variations are also applied to some large numbers, as a result of which we observed substantial improvements.

We also tested two multi polynomial variants, special q-polynomials and MPQS, and found out that MPQS outperformed all the other variants.

This research suggests that further improvements to the quadratic sieve integer factoring algorithm may be attained in the future.

REFERENCES

Agrawal, M., N. Kayal and N. Saxena, “Primes is in P”, Annals of Math., Vol.160, No.2, pp.781-793, 2004.

Bernstein, J.D., Integer Factorization, 2006, http://cr.yp.to.mirror.dogmap.org/2006.aws/notes-20060306.pdf.

Boender,H. and J. Riele, Factoring Integers with Large PrimeVariations of the Quadratic Sieve, CWI Report, NM-R9513, 1995.

Bressoud, D. M., Factorization and Primality Testing, UTM, Springer-Verlag, 1989.

Buchmann, J. and V. Müller, Algorithms for Factoring Integers, 2005, http://citeseer.ist.psu.edu/219938.htm

Cohen, H., A Course in Computational Algebraic Number Theory, Third Ed., Springer, 1996.

Contini, S. P., Factoring Integers with the Self-Initializing Quadratic Sieve, M.S. Thesis, University of Georgia, 1997.

Davis, J. A. And D. B.Holdridge, Factorization Using the Quadratic Sieve Algorithm, Sandia Report, Sand 83-1346, Sandia National Laboratories,Albuquerque, NM,1983.

Denny, T. F., Foktorisieren mit dem Quadratischen Sieb, Ph.D. Thesis, Universitat des Saarlandes, 1993.

Dixon, J. D., “Factorization and Primality Tests”, The American Mathematical Monthly, Vol.91, No.6, pp.333-352, Jun.-Jul. 1984.

Gathen, J. and J. Gerhard, Modern Computer Algebra, Cambridge University Press, 1999

Gerver, J. L., “Factoring Large Numbers with a Quadratic Sieve”, Mathematics of Computation, Vol.41, No.163, pp.287-294, July 1983.

Kechlibar, M., The Quadratic Sieve-introduction to theory with regard to implementation issues, Ph. D. Thesis, 2005.

Kendirli, B., Number Theory with Cryptographic Applications, Fatih University, Istanbul, 2006.

Kurowsky, B., The Multiple Polynomial Quadratic Sieve, 1998, http://brandt.kurowsky.net/projects/mpqs/paper/mpqs.ps.gz

Landquist, E., The Quadratic Sieve Factoring Algorithm, 2001, http://www.cs.virginia.edu/crab/QFS_Simple.pdf

Leyland, P., A. Lenstra, B. Dodson, A. Muffett and S. Wagstaff, MPQS with Three Large Primes,2001, http://www.leyland.vispa.com/numth/factorization/2,1606L/tmpqs.ps

Montgomery, P., "A Survey of Modern Integer Factorization Algorithms", CWI Quarterly, Vol.7, No.4, pp.337-365, 1994.

Morrison, M. and J. Brillhart, "A Method of Factoring and the Factorization of F_7", Mathematics of Computation, Vol.29, No. 129, pp.183-205, Jan. 1975.

Odlyzko, A. M., "The Future of Integer Factorization", Cryptobytes, Vol.1, No.2, pp.5-12, July 1995.

Pomerance, C., "The Quadratic Sieve Factoring Algorithm", Proc. of the Eurocrypt 84, Workshop on Advances in Cryptology: Theory and Applications of Cryptographic Techniques, Paris-France, pp.169-182, Springer-Verlac,1985.

Pomerance, C., "A Tale of Two Sieves", Notices of the AMS, 43 ,1996.

Pomerance, C., Smooth Numbers and the Quadratic Sieve, 2000, http://www.math.dartmouth.edu /~carlp/PDF/qstalk3.pdf

Rabah, K., "Review of Methods for Integer Factorization Applied to Cryptography", Journal of Applied Sciences, Vol.6, No.1, pp.458-481, 2006.

Riesel, H., Prime Numbers and Computer Methods for Factorization, Second Ed. Birkhauser, Boston, 1994.

Silverman, R. D., “The Multiple Polynomial Quadratic Sieve”, Mathematics of Computation, Vol.48, No.177, pp.329-339, January 1987.

Tattershall, J. J., Elementary Number Theory in Nine Chapters, Second Ed., Cambridge University Press, 2005.

Williams, H.C., “A p+1 Method of Factoring”, Mathematics of Computation, Vol.39, No.159, pp.225-234, July 1982.